







# The Weblate Manual

發  4.15.2

Michal Čihař

2023 年 01 月 25 日

<b>1</b>	<b>用 F 文檔</b>	<b>1</b>
1.1	Weblate 基礎知識	1
1.2	F 和用 F 資料	1
1.3	使用 Weblate 進行翻譯	10
1.4	下載和上傳翻譯	20
1.5	詞 F 表	23
1.6	檢查和修復	26
1.7	搜索	57
1.8	翻譯工作流	62
1.9	常見問題	66
1.10	支持的文件格式	74
1.11	版本控制整合	95
1.12	Weblate 的 REST API	102
1.13	Weblate 客 F 端	150
1.14	Weblate 的 Python API	155
<b>2</b>	<b>管理員文檔</b>	<b>157</b>
2.1	配置手 F	157
2.2	Weblate 部署	221
2.3	升級 Weblate	222
2.4	備份和移動 Weblate	230
2.5	身份核對	237
2.6	存取控制	247
2.7	翻譯專案數	257
2.8	語言定義	274
2.9	持續本地化	277
2.10	翻譯許可	285
2.11	翻譯程序	286
2.12	檢查和修復	292
2.13	配置自動建議	302
2.14	附加元件	312
2.15	翻譯記憶	329
2.16	配置	331
2.17	配置的例子	357
2.18	管理命令	373
2.19	公告	384
2.20	組件列表	386
2.21	Optional Weblate modules	387
2.22	定制 Weblate	392
2.23	管理介面	394
2.24	從 Weblate 獲得支持	403

2.25	Legal documents	405
<b>3</b>	<b>貢獻者文檔</b>	<b>407</b>
3.1	 Weblate 做貢獻	407
3.2	開始  Weblate 貢獻代碼	409
3.3	Weblate 源代碼	413
3.4	調試 Weblate	414
3.5	Weblate  部	416
3.6	Developing add-ons	417
3.7	Weblate 前端	419
3.8	在 Weblate 中匯報問題	420
3.9	Weblate 測試套件與連續集成	420
3.10	數據架構	422
3.11	發布 Weblate	426
3.12	Security and privacy	427
3.13	Contributing to Weblate modules	427
3.14	關於 Weblate	428
3.15	授權	429
<b>4</b>	<b>Change History</b>	<b>430</b>
4.1	Weblate 4.15.2	430
4.2	Weblate 4.15.1	430
4.3	Weblate 4.15	431
4.4	Weblate 4.14.2	431
4.5	Weblate 4.14.1	432
4.6	Weblate 4.14	432
4.7	Weblate 4.13.1	433
4.8	Weblate 4.13	433
4.9	Weblate 4.12.2	434
4.10	Weblate 4.12.1	434
4.11	Weblate 4.12	434
4.12	Weblate 4.11.2	435
4.13	Weblate 4.11.1	435
4.14	Weblate 4.11	435
4.15	Weblate 4.10.1	436
4.16	Weblate 4.10	436
4.17	Weblate 4.9.1	437
4.18	Weblate 4.9	437
4.19	Weblate 4.8.1	438
4.20	Weblate 4.8	438
4.21	Weblate 4.7.2	439
4.22	Weblate 4.7.1	439
4.23	Weblate 4.7	439
4.24	Weblate 4.6.2	440
4.25	Weblate 4.6.1	440
4.26	Weblate 4.6	440
4.27	Weblate 4.5.3	441
4.28	Weblate 4.5.2	441
4.29	Weblate 4.5.1	442
4.30	Weblate 4.5	442
4.31	Weblate 4.4.2	443
4.32	Weblate 4.4.1	443
4.33	Weblate 4.4	443
4.34	Weblate 4.3.2	444
4.35	Weblate 4.3.1	445
4.36	Weblate 4.3	445
4.37	Weblate 4.2.2	446
4.38	Weblate 4.2.1	446

4.39	Weblate 4.2 . . . . .	446
4.40	Weblate 4.1.1 . . . . .	447
4.41	Weblate 4.1 . . . . .	447
4.42	Weblate 4.0.4 . . . . .	449
4.43	Weblate 4.0.3 . . . . .	449
4.44	Weblate 4.0.2 . . . . .	449
4.45	Weblate 4.0.1 . . . . .	450
4.46	Weblate 4.0 . . . . .	450
4.47	Weblate 3.x 系列 . . . . .	451
4.48	Weblate 2.x 系列 . . . . .	462
4.49	Weblate 1.x 系列 . . . . .	474
4.50	Weblate 0.x 系列 . . . . .	478
<b>Python 模組索引</b>		<b>482</b>
<b>HTTP Routing Table</b>		<b>483</b>
<b>索引</b>		<b>486</b>



## 1.1 Weblate 基礎知識

### 1.1.1 Project and component structure

在 Weblate 中，翻譯組織成項目和組件。每個項目可以包含幾個組件，且組件包含各個語言的翻譯。組件相應於一個翻譯文件（例如 *GNU gettext* 或 安卓字符串資源）。項目幫助您將組件組織成邏輯的組（例如，將一個應用中使用的所有翻譯分組）。

默認情況下，每個項目都有對跨組件傳播的公共字符串的翻譯。這減輕了重和多版本翻譯的負擔。但假如翻譯應當有所不同，可以使用 [允許翻譯再用](#) 通過 [組件配置](#) 禁用翻譯傳播。

也參考：

[../devel/integration](#)

## 1.2 Weblate 和用 Weblate 資料

### 1.2.1 Weblate

默認情況下，每個人都可以瀏覽項目，查看翻譯或建議翻譯。只允許 Weblate 用戶實際保存更改，每種翻譯歸功於所做的。

您可以透過以下幾個簡單的步驟：

1. 使用憑據填寫表格。
2. 透過您收到的電子郵件中的連結。
3. （可選）調整您的個人資料以選擇您知道的語言。

## 1.2.2 控制面板

登陸時，您將看到項目和組件的概述，以及各自的翻譯進展。

在 2.5 版本新加入。

默認顯示您正在觀看的項目組件，通過您的首選語言交叉引用。

**提示：** 您可以使用“導航”選項卡切换到不同的視圖。

Preferences

☐ Hide completed translations on the dashboard

**Translation editor mode**

Full editor

**Zen editor mode**

Top to bottom

**Number of nearby strings**

15

Number of nearby strings to show in each direction in the full editor.

☒ Show secondary translations in the Zen mode

☐ Hide source if a secondary translation exists

**Editor link**

Enter a custom URL to be used as link to the source code. You can use `{{branch}}` for branch, `{{filename}}` and `{{line}}` as filename and line placeholders.

**Special characters**

You can specify additional special visual keyboard characters to be shown while translating. It can be useful for characters you use frequently, but are hard to type on your keyboard.

**Default dashboard view**

☒ Watched translations

☐ Suggested translations

Save

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

菜單有這些選項：

- 主菜單：“項目” > “主菜單” 瀏覽所有項目，主菜單中顯示了 WebLte 實例上的每個項目的轉譯狀態。
- 在主菜單中選擇一種語言：Guilabel：“語言”（語言）將顯示所有項目的翻譯狀態，由您的主要語言之一過濾。
- 主菜單：“觀看翻譯” 儀表板中的翻譯將顯示只有您正在觀看的項目的翻譯狀態，由您的主要語言過濾。

此外，DROP-DOWN 還可以顯示 Web2 管理員預先配置的任何數量的 \* 組件列表 \*，請參見 REF：“ComponentLists”。

您可以在“用 配置文件”設置的“用 配置文件”設置中配置“個人默認儀表板”視圖：“首選項”部分。

---

**備注：**當使用以下項目配置 WebBleate 時：設置：`single_project: file: settings.py`文件（請參： `ref: 'config'`），儀表板將不會顯示，因 用 將被重定向到單個項目或組件。

---

### 1.2.3 使用者個人檔案

單擊頂部菜單的右上角的“用 ”圖標可以訪問用 配置文件，然後單擊“ **Guilabel:**”設置“菜單。用 配置文件包含您的首選項。VCS 提交中使用的名稱和電子郵件地址，因此請保持此信息準確。

---

**備注：**所有語言選擇僅提供目前翻譯的語言。

---



---

**提示：**請求或添加要通過單擊按鈕來翻譯的其他語言，以使其成 可用。

---

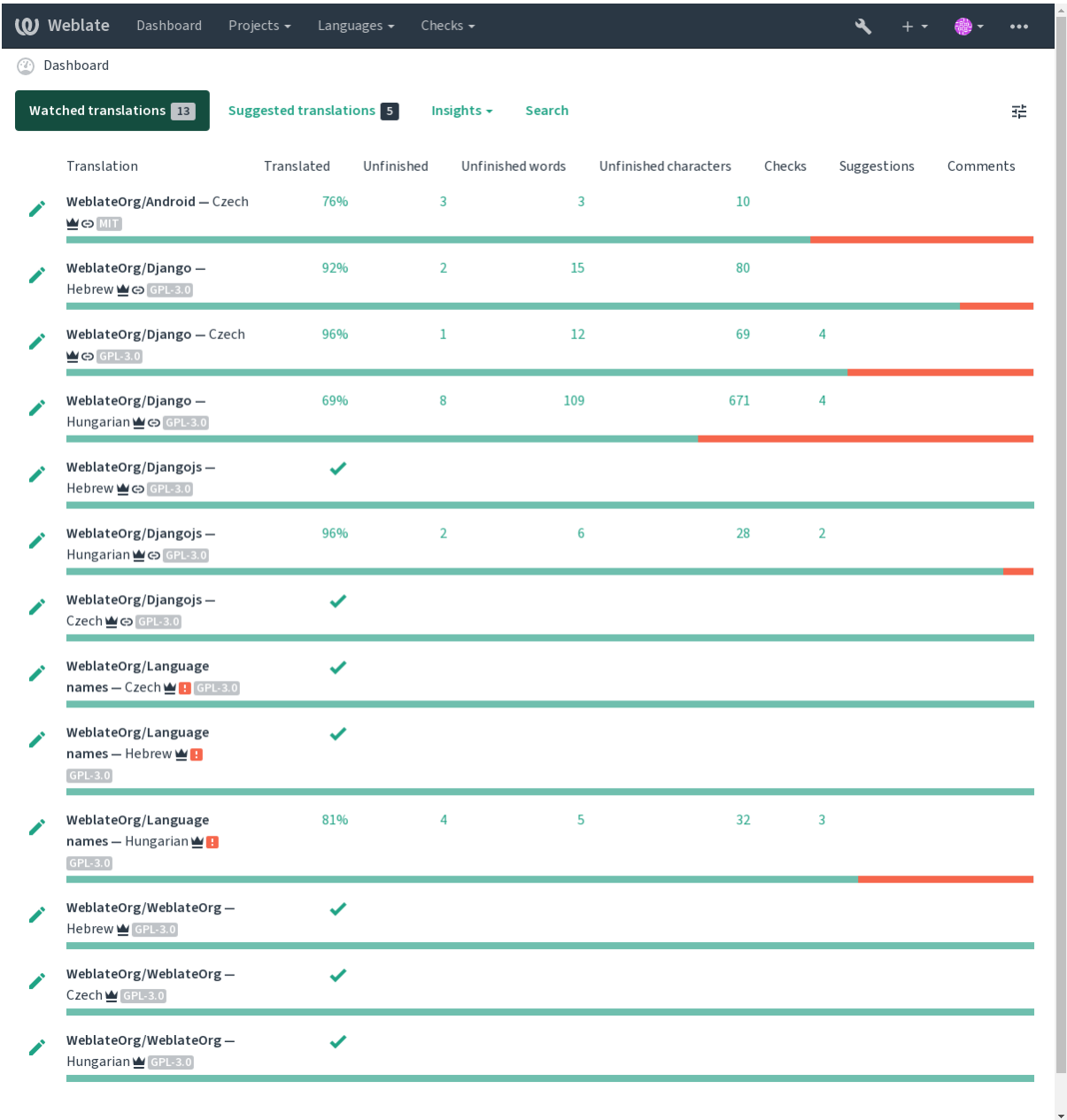
## 語言

### 1.2.4 介面語言

選擇您想要顯示用戶界面的語言。

#### 想翻譯成哪些語言

選擇您更喜歡翻譯的語言，它們將在觀看項目的主頁上提供，因此您可以更容易地訪問每個語言中的所有翻譯。



第二語言

您可以在翻譯時將哪些輔助語言作 指南顯示 指南。在以下圖像中可以看到一個示例，其中希伯來語如下所示：

The screenshot displays the Weblate web interface. At the top, there's a navigation bar with 'Weblate', 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below this, the breadcrumb 'WeblateOrg / Django / Czech / Translate' is shown. The main area features a 'Translation' section with input fields for 'Hebrew', 'English' (containing 'Files'), and 'Czech' (containing 'Soubory'). There are buttons for 'Save and continue', 'Save and stay', 'Suggest', and 'Skip'. A 'Nearby strings' section shows a table with columns 'Language' and 'Target string', listing 'Hebrew' (קבצים), 'Hungarian' (Fájlok), and 'English' (Files). On the right, a 'Glossary' section shows 'English' and 'Czech' with a note 'No related strings found in the glossary.' and an 'Add term to glossary' button. Below that, 'String information' includes 'Screenshot context', 'Explanation', 'Labels', 'Flags', 'Source string location', 'String age', 'Source string age', and 'Translation file'. The footer contains links: 'Powered by Weblate 4.15', 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

## 1.2.5 偏好設定

### 預設控制面板檢視

在：Guilabel：“首選項”選項卡，您可以默認選擇要呈現的可用儀表板視圖。如果選擇：Guilabel：“組件列表”，則必須從以下<sup>1</sup>容中選擇哪些組件列表：Guilabel：“默認組件列表”下拉列表。

也參考：

組件列表

### 編輯器連結

源代碼鏈接顯示在：REF：“組件”中配置的 Web <sup>2</sup>覽器中，默認情<sup>3</sup>下。

**提示：**通過設置：Guilabel：“編輯器”鏈接“，您可以使用本地編輯器打開翻譯字符串的 VCS 源代碼文件。您可以使用：參考：“標記”。

通常是“編輯器：//打開/? 文件 = {{filename}} & line = {{line}}”是一個很好的選擇。

也參考：

您可以在 [Nette 文件](#) 中找到更多關於<sup>4</sup>編輯器自訂 URL 協議的資訊。

## 特殊字元

Additional special characters to include in the *Visual keyboard*.

### 1.2.6 通知

訂閱來自：Guilabel：“通知”選項卡的各種通知。觀看或管理項目的所選事件的通知將每次發送電子郵件發送給您。

其中一些通知僅用於語言中的事件（例如，關於要轉換的新字符串），而組件級別的某些觸發（例如合錯誤）。這兩種通知在設置中可視化。

您可以切換監視項目和管理項目的通知，每個項目和組件都可以進一步調整（或音）。訪問“組件概述”頁面，然後選擇以下選項：Guilabel：“觀看”菜單。

以防：主：“自動觀看貢獻的項目即可用，您將在翻譯字符串時自動開始觀看項目。默認值取於：設置：*default\_auto\_watch*。

---

備：您不會收到您自己的行的通知。

---

---

[Powered by Weblate 4.15](#)
[About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

## 1.2.7 帳號

*Account* 標頁讓您設定基本的帳號資訊細節、連結其他登入帳號服務可透過它們來登入 Weblate、完整的移除您的帳號或打包下載您的使用者資訊 (詳情請參考 [Weblate user data export](#))。

---

**備註：** 服務列表取於您的 WebBlate 配置，但可以包括包括 Gitlab, Github, Google, Facebook 或 Bitbucket 或其他 OAuth 2.0 提供程序等流行網站。

---



W Weblate

DashboardProjectsLanguagesChecks

+...

Your profile

LanguagesPreferencesNotificationsAccountProfileTeamsLicensesAudit logAPI access

Account

Username

testuser

Username may only contain letters, numbers or the following characters: @ . + - \_

Full name

Weblate Test

E-mail

weblate@example.org

You can add another e-mail address below.

Your name and e-mail will appear as commit authorship.

Save

Current user identities

Identity	User ID	Action
Password	testuser	Change password
E-mail	weblate@example.org	Disconnect
Google	weblate@example.org	Disconnect
GitHub	123456	Disconnect
Bitbucket	weblate	Disconnect

Add new association

E-mail

Removal

Account removal deletes all your private data.

Remove my account

User data

You can download all your private data.

Download user data

## 1.2.8 個人檔案

此頁面上的所有欄位皆需選填，且可隨時刪除。一旦填入這些資訊，即表示您同意我們在您的個人檔案出現時分享這些資料。

The commit e-mail will be used instead of your account e-mail in version control commits. Use this to avoid leaking your real e-mail there. Be aware that using different e-mail can disconnect your contributions on other servers (for example your contributions will no longer link to your profile on GitHub). The private e-mail can be turned on site-wide using `PRIVATE_COMMIT_EMAIL_OPT_IN`.

可以為每個用戶顯示化身（取決於：設置：`enable_avatars`）。這些圖像是使用 <https://gravatar.com/> 獲得的。

## 1.2.9 授權方式

### 1.2.10 API 存取

您可以在此處獲取或重置 API 訪問令牌。

### 1.2.11 稽核記錄

審核日誌會跟踪您的帳戶執行的操作。它記錄您的帳戶銷每個重要操作的 IP 地址和瀏覽器。關鍵操作還觸發了主電子郵件地址的通知。

**也參考：**

[在反向代理後面運行](#)

## 1.3 使用 Weblate 進行翻譯

感謝您使用 Weblate 翻譯的興趣。項目可以設置直接翻譯，也可以通過不接受有賬戶的用戶提出的建議。

總的來說，有兩種翻譯方式：

- 該項目接受直接翻譯
- 該項目僅接受建議，一旦達到了已定數量的投票，就會自動驗證

有關翻譯工作流的更多信息，請見[翻譯工作流](#)。

翻譯項目可見性選項：

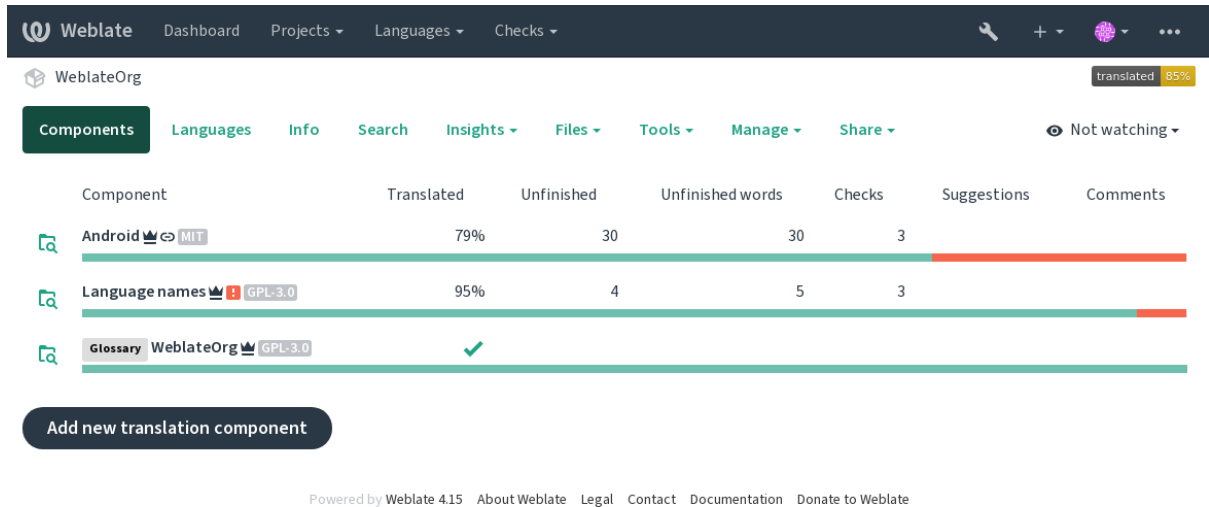
- Publicly visible
- 僅對某一組翻譯人員來看見

**也參考：**

[存取控制](#), [翻譯工作流](#)

### 1.3.1 翻譯專案數

翻譯項目持有相關組件; 用於同一軟件，書籍或項目的資源。

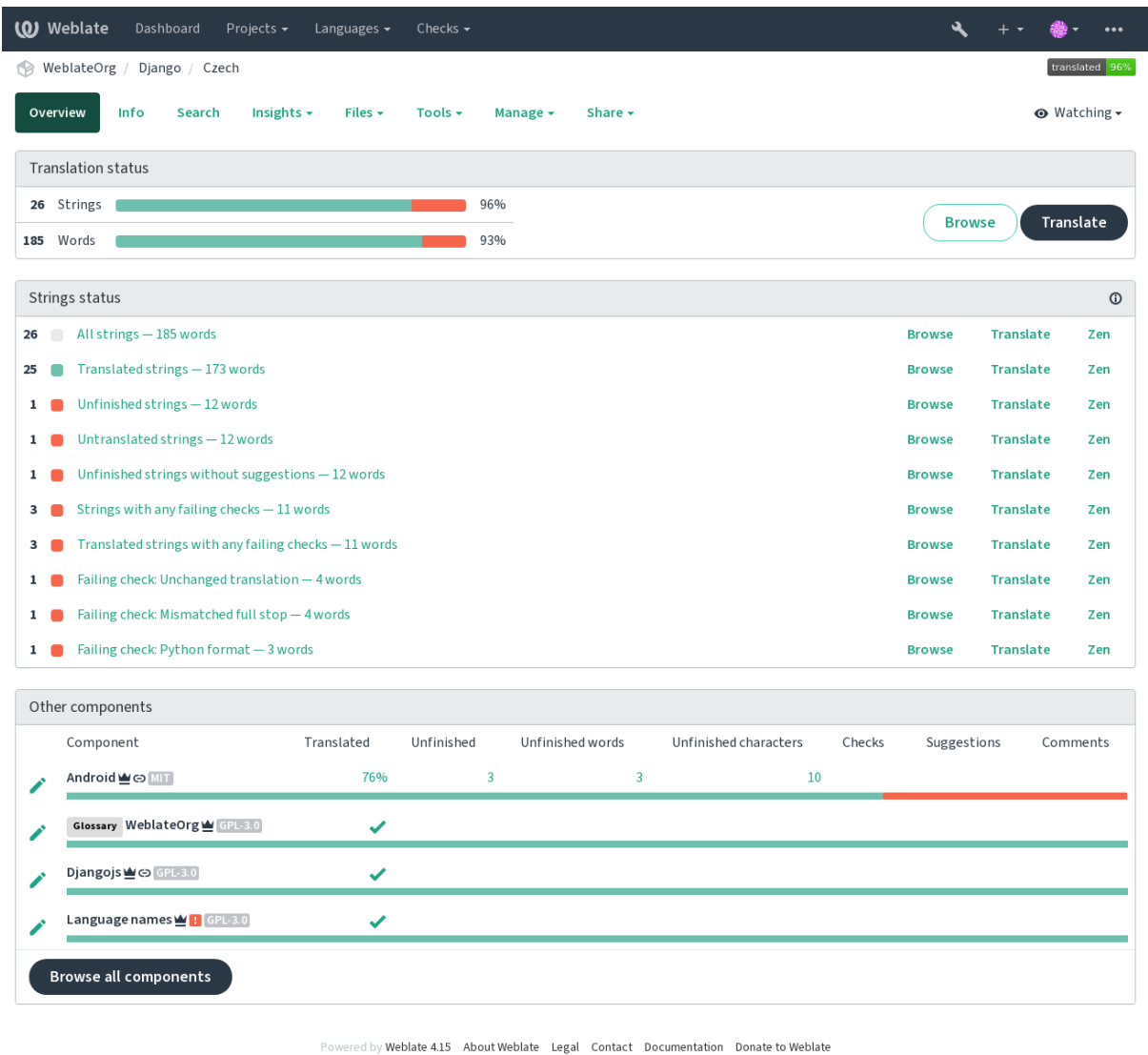


The screenshot shows the Weblate web interface. At the top is a navigation bar with links for Dashboard, Projects, Languages, and Checks. Below this is a header for 'WeblateOrg' with a 'translated 85%' badge. A secondary navigation bar includes tabs for Components, Languages, Info, Search, Insights, Files, Tools, Manage, and Share. The main content area displays a table of translation components with columns for Component, Translated, Unfinished, Unfinished words, Checks, Suggestions, and Comments. Three components are listed: 'Android' (79% translated, 30 unfinished words, 3 checks), 'Language names' (95% translated, 4 unfinished words, 5 checks), and 'Glossary' (100% translated, 0 unfinished words, 0 checks). Each row has a corresponding progress bar. At the bottom of the table is a button 'Add new translation component'. The footer contains links for 'Powered by Weblate 4.15', 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

Component	Translated	Unfinished	Unfinished words	Checks	Suggestions	Comments
Android	79%	30	30	3		
Language names	95%	4	5	3		
Glossary WeblateOrg	100%	0	0	0		

### 1.3.2 Translation links

Having navigated to a component, a set of links lead to its actual translation. The translation is further divided into individual checks, like *Untranslated strings* or *Unfinished strings*. If the whole project is translated, without error, *All strings* is still available. Alternatively you can use the search field to find a specific string or term.



### 1.3.3 建議

備：實際權限可能會有所不同，具體取於您的 Weblate 配置。

匿名用只能（默認情下）前向建議。在簽署的用中仍然可以使用，因在出現過翻譯的不確定性，提示其他翻譯人員審查它。

每天掃描建議，以除與當前翻譯的重和建議。

### 1.3.4 評

可以發布三種類型的釋：用於翻譯，源字符串或在使用以下功能時報告源字符串錯誤：REF: *Project-Source\_Review*。選擇您要討論的適合主題的那個。源字符串釋在任何事件中都適用於在原始字符串上提供反饋，例如應該被重建或提出關於它的問題。

You can use Markdown syntax in all comments and mention other users using @mention.

也參考：

report-source, 源字符串查, 用來源檢

### 1.3.5 變體

變體用於分組串的不同長度變體。然後，項目前端可以使用不同的字符串，具體取於屏幕或窗口大小。

也參考：

variants, 變體

### 1.3.6 標

標用於對項目中的字符串進行分類，以進一步自定義本地化工作流程（例如定義字符串類）。

Following labels are used by Weblate:

自動翻譯

String was translated using 自動翻譯.

來源需要檢

String was marked for review using 源字符串查.

也參考：

labels

### 1.3.7 Translating

在翻譯頁面上，顯示了其翻譯的源字符串和編輯區域。如果翻譯是數，則顯示多個源字符串和編輯區域，每個源字符串和編輯區域都描述標記翻譯語言的數。

所有特殊的空白字符都以紅色調，用灰色符號表示。在紅色中也調了多個後續空間以提醒轉器到在的格式問題。

變型的字元或額外的資訊會被呈現在這頁，它們大部分來自專案本身的原始碼（如：上下文的情境、評論解或是此字串也在哪個位置被使用過）。翻譯欄位也將會將次要語言（如有設定）容顯示在來源字符串上方（參第二語言）。

Below the translation, translators will find suggestion made by others, to be accepted (✓), accepted with changes (⇒), or deleted (❌).

## 數

單詞更改表單以明其數字名稱稱數。每種語言都有自己的數定義。例如，英語支持一個。在例如“汽車”的奇定義中，在多個定義中被引用了一輛車，“汽車”兩個或更多輛車被引用（或汽車的概念作名詞）。例如，捷克或阿拉伯語喜歡的語言有更多的數，且他們的數規則是不同的。

Weblate 在每個相應的語言中都對每個表格中的每一種完全支持（通過單獨翻譯每個數）。在翻譯的應用程序或項目中使用的字段數以及輪流如何取於配置的多個公式。Weblate 顯示基本信息，且 Unicode Consortium 的“語言數規則”是一個更詳細的描述。

也參考：

## 數公式

The screenshot displays the Weblate web interface for translating the string '%(count)s word' from English to Czech. The interface is divided into several sections:

- Top Navigation:** Includes links for Dashboard, Projects, Languages, and Checks.
- Breadcrumb:** Shows the path: WeblateOrg / Django / Czech / Translate.
- Translation Area:**
  - English:** Singular: '%(count)s word', Plural: '%(count)s words'.
  - Czech, One:** '%(count)s slovo'.
  - Czech, Few:** '%(count)s slova'.
  - Czech, Other:** '%(count)s slov'.
  - Plural formula:**  $(n=1) ? 0 : (n>2 \ \&\& \ n<=4) ? 1 : 2$ .
  - Needs editing:** A checkbox to indicate if the translation needs editing.
  - Buttons:** Save and continue, Save and stay, Suggest, Skip.
- Right Sidebar:**
  - Glossary:** Shows the English and Czech terms, with a note that no related strings were found.
  - String information:**
    - Screenshot context:** No screenshot currently associated.
    - Explanation:** No explanation currently provided.
    - Labels:** No labels currently set.
    - Flags:** python-format.
    - Source string location:** weblate/templates/translation.html:149.
    - String age:** 6 seconds ago.
    - Source string age:** 6 seconds ago.
    - Translation file:** weblate/locale/cs/LC\_MESSAGES/django.po, string 5.
- Bottom Section:**
  - Comments:** A section for adding comments on the string.
  - Scope:** A dropdown menu to select the scope of the comment.
  - New comment:** A text area for entering a new comment.
  - Save:** A button to save the comment.

## Alternative translations

在 4.13 版本新加入。

備 F: This is currently only supported with *Multivalue CSV file*.

With some formats, it is possible to have more translations for a single string. You can add more alternative translations using the *Tools* menu. Any blank alternative translations will be automatically removed upon saving.

## 鍵盤快捷鍵

在 2.18 版本變更: 鍵盤快捷鍵已在 2.18 中改進, 以不太可能與 F 瀏覽器或系統默認的碰撞。

在翻譯期間可以使用以下鍵盤快捷鍵:

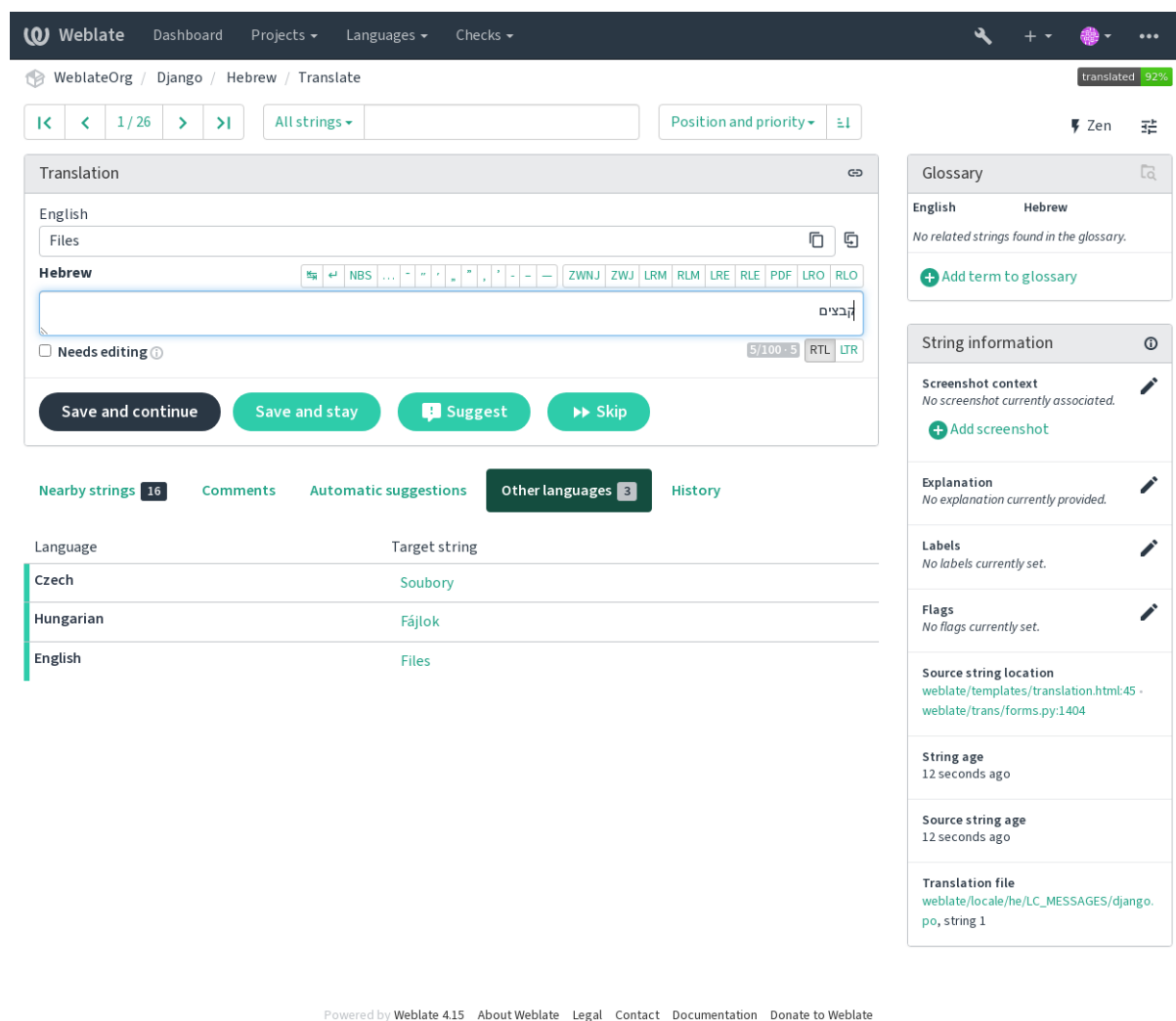
Keyboard shortcut	描述
Alt+Home	Navigate to first translation in current search.
Alt+End	Navigate to last translation in current search.
Alt+PageUp or Ctrl+↑ or Alt+↑ or Cmd+↑	Navigate to previous translation in current search.
Alt+PageDown or Ctrl+↓ or Alt+↓ or Cmd+↓	Navigate to next translation in current search.
Ctrl+Enter or Cmd+Enter	Submit current form; this is same as pressing <i>Save and continue</i> while editing translation.
Ctrl+Shift+Enter or Cmd+Shift+Enter	Unmark translation as needing edit and submit it.
Alt+Enter or Option+Enter	Submit the string as a suggestion; this is same as pressing <i>Suggest</i> while editing translation.
Ctrl+E or Cmd+E	Focus translation editor.
Ctrl+U or Cmd+U	Focus comment editor.
Ctrl+M or Cmd+M	顯示 <i>Automatic suggestions</i> 標 F, 請參見自動建議。
Ctrl+1 to Ctrl+9 or Cmd+1 to Cmd+9	從源字符串中放置給定數字的副本。
: KBD: "Ctrl + M" + : KBD: "1" 直到 : KBD: "9" 或 Cmd+M+1 to 9	將給定號的機器翻譯 F F 到當前翻譯。
: KBD: "Ctrl + I" + : kbd: "I" 到 : kbd: "9" Cmd+I+1 to 9	Ignore one item in the list of failing checks.
Ctrl+J or Cmd+J	Shows the <i>Nearby strings</i> tab.
Ctrl+S or Cmd+S	Focus search field.
Ctrl+O or Cmd+O	Copy source string.
Ctrl+Y or Cmd+Y	Toggle the <i>Needs editing</i> checkbox.

## Visual keyboard

小視覺鍵盤行在翻譯字段之上顯示。這對局部標點符號非常有用（因☞行是每種語言的本地），或者有難以鍵入方便的字符。

顯示的符號分[F]三類：

- User configured 特殊字元 defined in the 使用者個人檔案
- Web20 提供的每語言字符（例如引號或 RTL 特定字符）
- Characters configured using *SPECIAL\_CHARS*



## 翻譯上下文

此上下文描述提供有關當前字符串的相關信息。

## String attributes

像消息 ID, 上下文 (```) 或源代碼中的位置。

## 畫面快照

屏幕截圖可以上傳到 [WebLete](#)，以更好地通知譯者的位置以及如何如何使用字符串，參見：[REF](#)：“[屏幕截圖](#)”。

## 相鄰字串

顯示翻譯文件中相鄰信息。這些信息通常是類似的上下文，**F**保證翻譯的一致性。



### 其他出現位置

如果一條信息出現在多個地方（例如多個組件），若發現它們不一致，這個標會顯示所有的信息（參見不一致）。你可以選擇使用其中之一。

### 翻譯記憶

查找過去曾經翻譯的相識字符串，參見:ref:memory。

### 詞表

顯示當前信息中使用的項目詞表的術語。

### 最近更動

顯示最近通過 Weblate 更改了此信息的人員列表。

### 專案

Project Info，如翻譯人員的，或版本控制系統存儲庫中的字符串的目或鏈接項目使用。

如果要直接鏈接，轉格式必須支持它。

## Translation history

默認情況下，每個更改都是（除非在數據庫中保存的組件設置中關閉），否則可以恢復。可選地，人們還可以還原在底層版本控制系統中的任何內容。

## Translated string length

Weblate 可以在幾種方法中限制翻譯的長度，以確保翻譯的字符串不是太長：

- 翻譯的默認限制比源字符串長十倍。這可以通過以下方式關閉：設置：`limit_translation_length_by_source_length`。如果您正在觸及此情況，則可能也是由單格式轉錯誤地設置雙語的語言，使 Weblate 誤認是實際源串的翻譯鍵。請參：參考：“雙隅猴”以獲取更多信息。
- 由翻譯文件或標定義的字符中的最大長度，請參：ref: `check-max-length`。
- 最大渲染大小以標定義，請參：Ref: “Check-Max-Size”。

### 1.3.8 自動建議

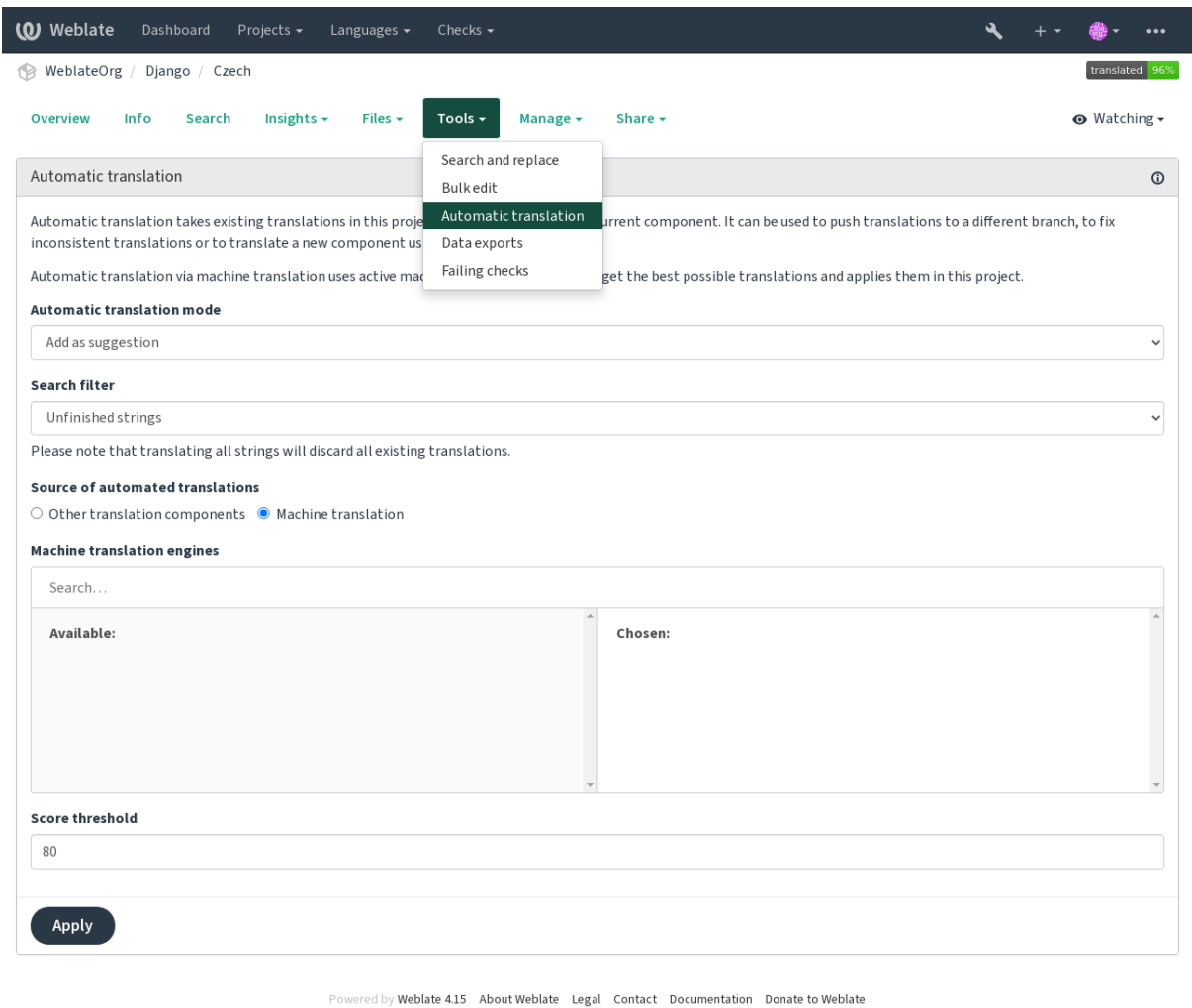
基於配置和翻譯語言，Weblate 提供了來自多種機器翻譯工具的建議和：REF: “翻譯 - 記憶庫”。所有機器翻譯都可以在每個翻譯頁面的單個選項卡中使用。

#### 也參考：

您可以找到支持的工具列表：REF: “計算機翻譯設置”。

### 1.3.9 自動翻譯

您可以使用自動轉基於外部源的引導轉。該工具被稱：主：“自動翻譯”可訪問：Guilabel: “工具”菜單，選擇組件和一種語言後：



兩種操作模式是可能的：

- 使用其他 Web2 個組件作翻譯的源。
- 使用所選機器翻譯服務，轉高於一定的質量值。

您還可以選擇要自動翻譯的字符串。

**警告：** 注意這將覆蓋具有廣泛過濾器（如：Guilabel：‘所有字符串’）覆蓋現有翻譯。

在幾種情況下有用，如在不同組件（例如應用程序及其網站）之間的統一轉（例如應用程序及其網站）或使用現有翻譯（翻譯記憶庫）引導用於新組件的翻譯時。

The automatically translated strings are labelled *Automatically translated*.

**也參考：**

跨組件保持翻譯一致

### 1.3.10 頻次限制

避免濫用界面，速率限制應用於若干操作，如搜索，發送聯表單或翻譯。如果受其影響，則在某個時期被阻止，直到您可以再次執行操作。

默認限制和微調在管理手中介紹，請參：參考：“速率限制”。

### 1.3.11 搜尋取代

使用：Guilabel 有效地改變術語或執行批量固定字符串：“搜索和替”：Guilabel：“工具”菜單。

---

**提示：**不用擔心會弄亂字串。這分成兩個步驟：先預覽編輯字串，才會確認實際更動。

---

### 1.3.12 大量編輯

批量編輯允許在字符串數量上執行一個操作。您可以通過搜索它們來定義字符串設置用於匹配的東西。支持以下操作：

- 更改字符串狀態（例如批准所有未升義的字符串）。
- 調整翻譯標（參見：參考：“定制檢查”）
- 調整字符串標（參見：參考：“標”）

---

**提示：**此工具被稱：Guilabel：“批量編輯”可訪問：Guilabel：“工具”菜單，每個項目，組件或翻譯。

---

**也參考：**

批量編輯插件

### 1.3.13 矩陣視圖

要有效地比較不同的語言，您可以使用矩陣視圖。它在工具菜單下的每個組件頁面上都可用。首先選擇您要比較的所有語言確認您的選擇，然後您可以單擊任何翻譯以快速打開和編輯它。

矩陣視圖也是一個很好的起點，可以找到不同語言的缺失翻譯從一個視圖快速添加它們。

### 1.3.14 模式

Zen 編輯器可以通過在翻譯組件時單擊右上角的 Zen 按鈕來用。它簡化了局移除了額外的 UI 元素，例如相鄰字串或詞表。

You can select the Zen editor as your default editor using the 偏好設定 tab on your 使用者個人檔案. Here you can also choose between having translations listed *Top to bottom* or *Side by side* depending on your personal preference.

## 1.4 下載和上傳翻譯

您可以從轉譯中導出文件，進行更改，再再次導入它們。這允許機器工作，然後將更改合併回現有翻譯。這也有效，即使它在此期間已更改。

---

**備註：** 可用選項可能受到訪問控制設置的限制。

---

### 1.4.1 下載翻譯

從項目或組件儀表板中，可在以下方式下載可翻譯文件：Guilabel: `files` 菜單。

第一個選項是以原始格式下載文件，因為它存儲在存儲庫中。在這種情況下，翻譯中的任何更改都會被提交，最新文件是不帶任何轉譯的數量。

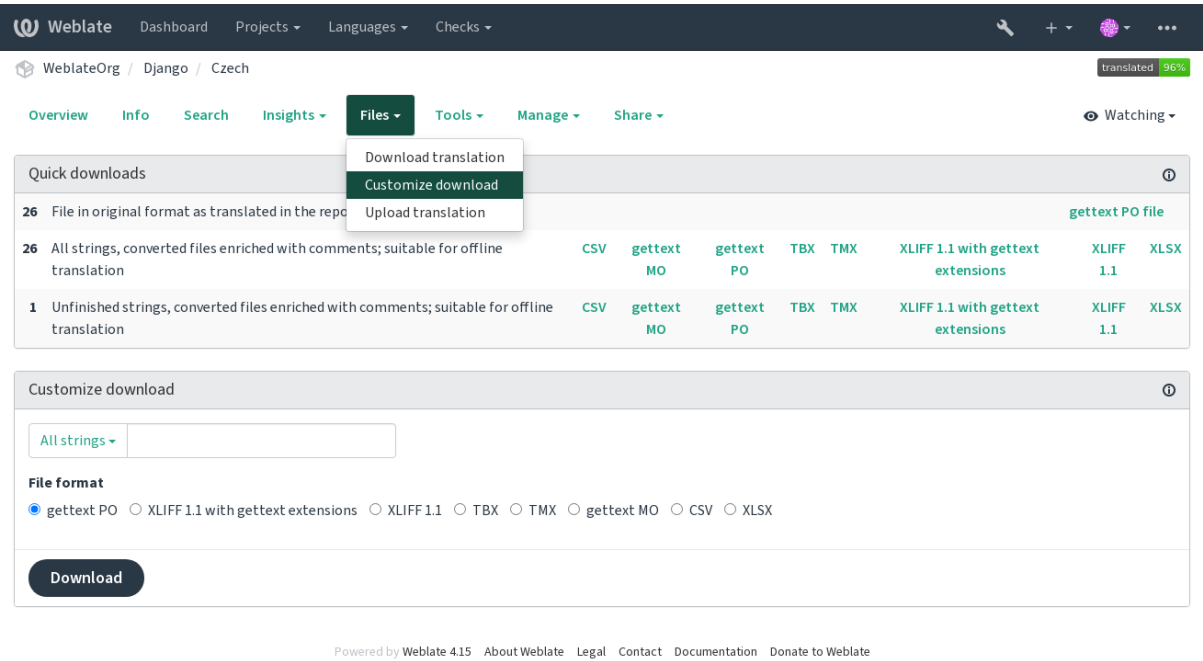
您還可以下載轉譯廣泛使用的本地化格式之一的翻譯。轉譯後的文件將富裕地富裕在 WebLate 中提供的數據；例如其他上下文，評論或標記。通過以下內容提供了多種文件格式：Guilabel: `文件` ↓ 圭： `自定義下載` 菜單：

- gettext PO
- XLIFF 附 gettext 擴展
- XLIFF 1.1
- 術語交匯
- Translation Memory eXchange
- GetText Mo（僅在翻譯使用 GetText Po 時可用）
- CSV
- Excel Open XML
- JSON (only available for monolingual translations)
- Android String Resource (only available for monolingual translations)
- iOS strings (only available for monolingual translations)

---

**提示：** 轉譯文件中可用的內容基於文件格式功能而不同，您可以查找概述：REF: *FMT\_CAPABS*。

---



也參考:

`GET /api/translations/(string:project)/(string:component)/(string:language)/file/`

### 1.4.2 上傳翻譯

當您進行更改時，使用：Guilabel: “上傳翻譯”：Guilabel: “Files”菜單。

The screenshot shows the Weblate web interface. At the top, there's a navigation bar with 'Weblate', 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below it, the breadcrumb is 'WeblateOrg / Django / Czech'. A status bar on the right shows 'translated 96%'. The main navigation has tabs: 'Overview', 'Info', 'Search', 'Insights', 'Files' (selected), 'Tools', 'Manage', and 'Share'. A dropdown menu is open over the 'Files' tab, showing 'Download translation', 'Customize download', and 'Upload translation'. The 'Upload' form is displayed, with a message: 'The uploaded file will be merged with the current file. Use the dropdown menu if you want to overwrite already translated strings.' The form includes a 'File' section with a 'Choose File' button and 'No file chosen' text. Below that is the 'File upload mode' section with four radio buttons: 'Add as translation' (selected), 'Add as suggestion', 'Add as translation needing edit', and 'Replace existing translation file'. The 'Processing of "Needs editing" strings' section has a dropdown menu set to 'Do not import'. The 'Conflict handling' section has a dropdown menu set to 'Change translated strings'. Below these is a note: 'Whether to overwrite existing translations if the string is already translated.' The 'Author name' field contains 'Weblate Test' and the 'Author e-mail' field contains 'weblate@example.org'. At the bottom is an 'Upload' button. The footer shows 'Powered by Weblate 4.15' and links to 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

## 支持的文件格式

可以上載支持的文件格式中的任何文件，但仍然建議使用與用於翻譯的文件格式相同的文件格式，否則可能無法正確翻譯一些功能。

### 也參考：

[支持的文件格式](#), [下載和上傳翻譯](#)

## 導入方法

這些是上傳翻譯文件時提供的選項：

### 新增翻譯 (translate)

導入的字符串作爲現有字符串的翻譯添加。這是最常見的用例，也是默認行爲。

只能從上傳的文件中使用翻譯，而且還有其他內容。

### 新增建議 (suggest)

字符串作爲建議來匯入，當您希望審核已上傳的字符串時執行此操作。

只能從上傳的文件中使用翻譯，而且還有其他內容。

### 新增需要編輯的翻譯 (fuzzy)

字符串作爲需要編輯的翻譯匯入。當您希望使用翻譯但也需要審核時，這可能很有用。

只能從上傳的文件中使用翻譯，而且還有其他內容。

### 取代既有翻譯檔 (replace)

現有文件替換新內容。這可能導致現有翻譯丟失，謹慎使用。

### 更新來源字符串 (source)

更新雙語翻譯文件中的源字符串。這類似於什麼 `addon-weblate.gettext.msgmerge`。

僅某些檔案格式支援此選項。

### 新增字符串 (add)

將新字符串添加到翻譯中。它跳過已經存在的那個。

如果您想要添加新字符串更新現有翻譯，請使用以下時間將文件上傳 添加作翻譯。

此選項僅在打開管理字符串時可用。

上傳檔中僅使用源、翻譯和金鑰（情境）。

### 也參考:

```
POST /api/translations/(string:project)/(string:component)/
(string:language)/file/
```

### 衝突處理

定義如何處理已翻譯的上傳字符串。

### 需要編輯的字串

還有一個選擇如何處理需要在導入的文件中編輯的字符串。這種字符串可以以以下三種方式之一句柄：“不要導入”，“導入作字符串需要編輯”，或“轉”導入。

### 覆蓋作者身份

使用管理員權限，您還可以指定上傳文件的作者身份。如果您以其他方式收到文件希望將其合到現有翻譯中，同時正確明實際作者，這可能會很有用。

## 1.5 詞表

每個項目可以包括一個或多個詞表作存儲術語的速記。術語表可以實現維持翻譯的一致性。

可以自己管理每種語言的詞表，但它們作一個單個組件一起存儲，可幫助項目管理員和多語言翻譯人員來維持一些跨語言一致性。從包含當前翻譯的字符串的詞表中的術語顯示在翻譯編輯器的側欄中。

### 1.5.1 管理詞表

在 4.5 版本變更: 詞表現在是常規的翻譯組件，您可以使用它們上的所有 Weblate 功能 - 評論，存儲在遠程存儲庫中，或添加明。

使用任何組件作詞表，方法是開當作詞表。你可以一個項目創建多個詞表。

使用該項目自動創建給定項目的空詞表。詞表在同一項目的所有組件中共享，且可選地使用其他項目: REF: “組件 - 鏈接” 來自各個詞表組件。

詞表組件看起來像 Weblate 中的任何其他組件，其中添加彩色標:

Webate

Dashboard

Projects

Languages

Checks

WebateOrg / Glossary WebateOrg / Czech

translated 100%

Overview

Info

Search

Insights

Files

Tools

Share

Not watching

Translation status

2 Strings

100%

3 Words

100%

Add new glossary term

Browse

Translate

Strings status

2

All strings — 3 words

Browse

Translate

Zen

2

Translated strings — 3 words

Browse

Translate

Zen

Other components

Component	Translated	Unfinished	Unfinished words	Checks	Suggestions	Comments
<div>Django</div>	96%	1	12	3		
<div>Language names</div>	✓					

Browse all components

Powered by Weblate 4.15

About Weblate

Legal

Contact

Documentation

Donate to Weblate

您可以 覽所有詞 表術語：

Webate

Dashboard

Projects

Languages

Checks

WebateOrg / Glossary WebateOrg / Czech / Browse

translated 100%

<

<

1/1

>

>

All strings

Source string

+

Add new glossary term

English

Czech

machine translation

strojový překlad

project

projekt

Powered by Weblate 4.15

About Weblate

Legal

Contact

Documentation

Donate to Weblate

或將它們編輯 任何翻譯。

### 1.5.2 詞 表術語

詞 表術語翻譯相同的常規字符串。您可以使用以下 容切 其他功能：Guilabel：“工具”菜單，每個術語。



The screenshot shows the Weblate web interface. At the top, there's a navigation bar with 'Weblate', 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below it, the breadcrumb is 'WeblateOrg / Glossary WeblateOrg / Czech / Translate'. The main area is titled 'Glossary term' and contains a form for editing a term. The 'English' field has 'project', and the 'Czech' field has 'projekt'. There's a 'Needs editing' checkbox and an 'Explanation' field. Below the form are buttons: 'Save and continue', 'Save and stay', 'Suggest', 'Skip', and 'Tools'. The 'Tools' dropdown menu is open, showing options: 'Delete string', 'Mark as untranslatable', 'Mark as forbidden translation', 'Mark as terminology', and 'Add variant of this string'. On the right, there's a 'Glossary' sidebar with a search bar and a list of terms. Below it, 'String information' shows 'String age' and 'Source string age' as 'a second ago'. At the bottom, there's a 'Nearby strings' section with a table showing 'machine translation' and 'project' in English and Czech.

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

## 不可翻譯的術語

在 4.5 版本新加入。

Flagging certain glossary term translations `read-only` by bulk-editing, typing in the flag, or by using **Tools** ↓ **Mark as untranslatable** means they can not be translated. Use this for brand names or other terms that should not be changed in other languages. Such terms are visually highlighted in the glossary sidebar.

也參考：

使用標 禁止 自定義行

## 禁止的翻譯

在 4.5 版本新加入。

標記某些詞 表示術語翻譯 “禁止的”，通過批量編輯，鍵入標 禁止，或者使用：**Guilabel**：“工具” ↓ 主 禁止：“標記 禁止翻譯”意味著它們是 **\*\* 不是 \*\*** 使用。當有些詞 味或可能有意外含義時，用這可以澄清翻譯。

也參考：

使用標 禁止 自定義行

## 專有名詞

在 4.5 版本新加入。

將某些詞表術語標記“術語”（批量編輯），或者使用旗幟，或使用工具 `! :guilabel:` 標記術語添加“所有語言”中“詞表”中的條目增加了所有語言。使用此項來是應該得到深思熟慮的重要術語，在所有語言中保留一致的含義。

### 也參考：

使用標自定義行

## 變體

變體是將字符串組合在一起的通用方式。在翻譯時，所有術語變體都列在詞表側欄中。

---

**提示：** 您可以使用它來添加縮寫或縮短表達式。

---

### 也參考：

variants

## 1.6 檢查和修復

質量檢查有助於發現常見的翻譯錯誤，確保翻譯質量良好。如果出現誤報，則可以忽略這些檢查。

提交未通過檢查的翻譯後，將立即向用顯示：

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Django / Czech / Translate
 translated 96%

The translation has been saved, however there are some newly failing checks: Missing plurals, Python format

1 / 1

Custom search

'%(count)s word'

Position

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

English

Singular

%(count)s word

Plural

%(count)s words

Czech, One

Czech, Few

několik slov

Czech, Other

%(count)s slov

Plural formula

(n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2

Needs editing

Save and continue

Save and stay

Suggest

Skip

Nearby strings

Comments

Automatic suggestions

Other languages

History

New comment

Comment on this string for fellow translators and developers to read.

Scope

Translation comment, discussions with other translators

Is your comment specific to this translation, or generic for all of them?

New comment

You can use Markdown and mention users by @username.

Save

Things to check

Python format

Following format strings are missing: %(count)s

Dismiss

For all languages

Missing plurals

Some plural forms are untranslated

Dismiss

For all languages

Glossary

English

Czech

No related strings found in the glossary.

Add term to glossary

String information

Screenshot context

No screenshot currently associated.

Add screenshot

Explanation

No explanation currently provided.

Labels

No labels currently set.

Flags

python-format

Source string location

weblate/templates/translation.html:149

String age

10 seconds ago

Source string age

10 seconds ago

Translation file

weblate/locale/cs/LC\_MESSAGES/django.po, string 5

pending

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

1.6. 檢查和修復

27

### 1.6.1 自動修正

除了質量檢查外，Weblate 還可以自動修復翻譯字符串中的一些常見錯誤。謹慎使用它，不要使其增加翻譯錯誤。

也參考：

`AUTOFIX_LIST`

### 1.6.2 質量檢查

Weblate 對字符串進行了廣泛的質量檢查。以下部分將對它們進行更詳細的描述。還有針對特定語言的檢查。如果有錯誤報告，請將缺陷提交。

也參考：

`CHECK_LIST`，使用標 自定義行

### 1.6.3 翻譯檢查

在每次翻譯更改時執行，幫助翻譯人員保持高質量的翻譯。

#### BBCode 標記

##### 概要

翻譯中的 BBCode 與來源不符

##### 範圍

已翻譯好的字串

##### 檢查類

`weblate.checks.markup.BBCodeCheck`

##### 檢查標識符

`bbcode`

##### 忽略的標

`ignore-bbcode`

BBCode 表示簡單的標記，例如以粗體或斜體突出顯示消息的重要部分。

此檢查確保在翻譯中也找到它們。

---

備：當前檢測 BBCode 的方法非常簡單，因此此檢查可能會生誤報。

---

#### 連續重單字

在 4.1 版本新加入。

##### 概要

文字在同一行中有兩個相同的單字：

##### 範圍

已翻譯好的字串

##### 檢查類

`weblate.checks.duplicate.DuplicateCheck`

##### 檢查標識符

`duplicate`

**忽略的標**

ignore-duplicate

檢查翻譯中是否有連續重的單詞。這通常表示翻譯錯誤。

---

**提示：** 此檢查包括特定於語言的規則，以避免誤報。如果在您的情況下錯誤觸發，請告訴我們。請參閱在 [Weblate](#) 中匯報問題。

---

**與詞表不同**

在 4.5 版本新加入。

**概要**

翻譯未遵循詞表定義的字詞。

**範圍**

已翻譯好的字串

**檢查類**

weblate.checks.glossary.GlossaryCheck

**檢查標識符**

check\_glossary

**Flag to enable**

check-glossary

**忽略的標**

ignore-check-glossary

此檢查必須使用“檢查詞表”標（參見：REF：“定制檢查”）。在用它之前請考慮以下操作：

- 它確實是精確的字符串匹配，預計詞表將包含所有變體中的術語。
- 檢查每個字符串針對詞表昂貴，它將慢 Weblate 中的任何操作，涉及運行等待導入字符串或翻譯。

**也參考：**

詞表, 使用標自定義行, 翻譯旗標

**兩個空白****概要**

翻譯含有兩個空白

**範圍**

已翻譯好的字串

**檢查類**

weblate.checks.chars.DoubleSpaceCheck

**檢查標識符**

double\_space

**忽略的標**

ignore-double-space

檢查翻譯中是否存在雙空格，以避免其他與空格相關的檢查出現誤報。

當在源中找到雙空格時，檢查假，這意味著故意使用雙空格。

## 格式化字符串

檢查字符串格式是否在源和翻譯之間。在翻譯中省略格式字符串通常會導致嚴重的問題，因此字符串中的格式通常應與源匹配。

Weblate 支持檢查幾種語言的格式字符串。僅當適當地標記了字符串時（例如，C 格式 `c-format`），才會自動用該檢查。Gettext 會自動添加它，但是對於其他文件格式，或者如果您的 PO 文件不是由 **xgettext** 生成的，您可能必須手動添加它。

可以按每單位（請參見源字符串另外的信息）或在組件配置中完成此操作。每個組件定義它比較簡單，但是如果該字符串未解釋格式字符串，而碰巧使用了格式化字符串語法，則可能導致誤報。

**提示：** 如果 Weblate 中不提供特定格式的檢查，則可以使用通用位符。

除了檢查，這也將高亮格式化字符串，方便將它們插入到已翻譯字符串：

The screenshot shows the Weblate web interface for a project named 'Django'. The main area is titled 'Translation' and shows the English source string '%(count)s word' and its plural form '%(count)s words'. The Czech translation is currently '%(count)s slovo'. The interface includes a sidebar with 'Glossary' and 'String information' tabs, and a bottom panel showing a list of nearby strings and a detailed view of the current string's translation history.

**Translation Panel:**

- English:** Singular: '%(count)s word', Plural: '%(count)s words'.
- Czech, One:** '%(count)s slovo'.
- Czech, Few:** '%(count)s slova'.
- Czech, Other:** '%(count)s slov'.
- Plural formula:** `(n==1) ? 0 : (n>2 && n<=4) ? 1 : 2`.
- Needs editing:** ☐
- Buttons:** Save and continue, Save and stay, Suggest, Skip.

**String information Panel:**

- Screenshot context:** No screenshot currently associated.
- Explanation:** No explanation currently provided.
- Labels:** No labels currently set.
- Flags:** python-format.
- Source string location:** `weblate/templates/translation.html:149`.
- String age:** 6 seconds ago.
- Source string age:** 6 seconds ago.
- Translation file:** `weblate/locale/cs/LC_MESSAGES/django.po`, string 5.

**Bottom Panel:**

- Nearby strings:** 20.
- Comments:**
- Automatic suggestions:**
- Other languages:** 3.
- History:**

**String History:**

- English:** Singular: '%(count)s word', Plural: '%(count)s words'.
- Czech:** 48 characters edited. Current translation: '%(count)s slovo', Translated: '%(count)s slovo'.

## AngularJS 插值字串

### 概要

AngularJS 插值字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.angularjs.AngularJSInterpolationCheck`

### 檢查標識符

`angularjs_format`

### Flag to enable

`angularjs-format`

### 忽略的標 F

`ignore-angularjs-format`

### Named format string example

你的余額是 `{{uluity}}` `{{currency}}`

### 也參考:

: ref: *check-formats*, **‘angularjs text 插值’** <<https://angular.io/guide/interpolation>>‘\_

## C 格式

### 概要

C 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.CFormatCheck`

### 檢查標識符

`c_format`

### Flag to enable

`c-format`

### 忽略的標 F

`ignore-c-format`

### Simple format string example

這 F 有 %d 顆蘋果

### Position format string example

你的余額是 %1 \$ d%2 \$ s

### 也參考:

格式化字符串,

**‘c 格式字符串’** <[https://www.gnu.org/software/gettext/manual/html\\_node/c\\_002dformat.html](https://www.gnu.org/software/gettext/manual/html_node/c_002dformat.html)>‘\_’, **‘c printf 格式’** <[https://en.wikipedia.org/wiki/printf\\_format\\_string](https://en.wikipedia.org/wiki/printf_format_string)>‘\_

## C# 格式

### 概要

C# 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.CSharpFormatCheck`

### 檢查標識符

`c_sharp_format`

### Flag to enable

`c-sharp-format`

### 忽略的標 F

`ignore-c-sharp-format`

### Position format string example

這 F 有 {0} 顆蘋果

### 也參考:

格式化字符串, [C# String Format](#)

## ECMAScript 模板字面值

### 概要

ECMAScript 模板字面值與來源不相符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.ESTemplateLiteralsCheck`

### 檢查標識符

`es_format`

### Flag to enable

`es-format`

### 忽略的標 F

`ignore-es-format`

### Interpolation example

這 F 有 \${number} 顆蘋果

### 也參考:

格式化字符串, [Template literals](#)



## i18next 插補

在 4.0 版本新加入。

### 概要

i18next 插補與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.I18NextInterpolationCheck`

### 檢查標識符

`i18next_interpolation`

### Flag to enable

`i18next-interpolation`

### 忽略的標<sup>F</sup>

`ignore-i18next-interpolation`

### Interpolation example

這<sup>F</sup>有 {{number}} 顆蘋果

### Nesting example

這<sup>F</sup>有 \$t(number) 顆蘋果

### 也參考:

格式化字符串, [i18next interpolation](#)

## ICU MessageFormat

在 4.9 版本新加入。

### 概要

ICU MessageFormat 字串有語法錯誤或是<sup>F</sup>位符不相符。

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.icu.ICUMessageFormatCheck`

### 檢查標識符

`icu_message_format`

### Flag to enable

`icu-message-format`

### 忽略的標<sup>F</sup>

`ignore-icu-message-format`

### Interpolation example

這<sup>F</sup>有 {數量, <sup>F</sup>數, 一 {一顆蘋果} 其他 {# 顆蘋果}}。

This check has support for both pure ICU MessageFormat messages as well as ICU with simple XML tags. You can configure the behavior of this check by using `icu-flags:*`, either by opting into XML support or by disabling certain sub-checks. For example, the following flag enables XML support while disabling validation of plural sub-messages:

```
icu-message-format, icu-flags:xml:-plural_selectors
```

xml	Enable support for simple XML tags. By default, XML tags are parsed loosely. Stray < characters are ignored if they are not reasonably part of a tag.
strict-xml	Enable support for strict XML tags. All < characters must be escaped if they are not part of a tag.
-highlight	Disable highlighting placeholders in the editor.
-require_other	Disable requiring sub-messages to have an other selector.
-submessage_selector	Skip checking that sub-message selectors match the source.
-types	Skip checking that placeholder types match the source.
-extra	Skip checking that no placeholders are present that were not present in the source string.
-missing	Skip checking that no placeholders are missing that were present in the source string.

Additionally, when `strict-xml` is not enabled but `xml` is enabled, you can use the `icu-tag-prefix:PREFIX` flag to require that all XML tags start with a specific string. For example, the following flag will only allow XML tags to be matched if they start with `<x::`:

```
icu-message-format, icu-flags:xml, icu-tag-prefix:"x:"
```

This would match `<x:link>click here</x:link>` but not `<strong>this</strong>`.

#### 也參考:

*ICU MessageFormat* 語法, 格式化字符串, *ICU: Formatting Messages*, *FormatJS: Message Syntax*

## Java 格式

### 概要

Java 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.JavaFormatCheck`

### 檢查標識符

`java_printf_format`

### Flag to enable

`java-printf-format`

### 忽略的標

`ignore-java-printf-format`

### Simple format string example

這有 %d 顆蘋果

### Position format string example

你的余額是 %1 \$ d %2 \$ s

在 4.14 版本變更: This used to be toggled by `java-format` flag, it was changed for consistency with GNU gettext.

#### 也參考:

:ref:check-formats, [‘java 格式字符串 <https://docs.oracle.com/javase/7/docs/api/java/util/formatter.html>’](https://docs.oracle.com/javase/7/docs/api/java/util/formatter.html)

## Java MessageFormat

### 概要

Java MessageFormat 字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.JavaMessageFormatCheck`

### 檢查標識符

`java_format`

### Flag to enable unconditionally

`java-format`

### Flag to enable autodetection

`auto-java-messageformat` enables check only if there is a format string in the source

### 忽略的標

`ignore-java-format`

### Position format string example

這有 {0} 顆蘋果

在 4.14 版本變更: This used to be toggled by `java-messageformat` flag, it was changed for consistency with GNU gettext.

This check validates that format string is valid for the Java MessageFormat class. Besides matching format strings in the curly braces, it also verifies single quotes as they have a special meaning. Whenever writing single quote, it should be written as `'`. When not paired, it is treated as beginning of quoting and will not be shown when rendering the string.

### 也參考:

格式化字符串, [Java MessageFormat](#)

## JavaScript 格式

### 概要

JavaScript 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.JavaScriptFormatCheck`

### 檢查標識符

`javascript_format`

### Flag to enable

`javascript-format`

### 忽略的標

`ignore-javascript-format`

### Simple format string example

這有 %d 顆蘋果

### 也參考:

格式化字符串, [JavaScript 格式字符串 <https://www.gnu.org/software/gettext/manual/html\\_node/javascript\\_002dformat.html>](https://www.gnu.org/software/gettext/manual/html_node/javascript_002dformat.html)

## Lua 格式

### 概要

Lua 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.LuaFormatCheck`

### 檢查標識符

`lua_format`

### Flag to enable

`lua-format`

### 忽略的標 F

`ignore-lua-format`

### Simple format string example

這 F 有 %d 顆蘋果

### 也參考:

格式化字符串, **Lua 格式字符串** <[https://www.gnu.org/software/gettext/manual/html\\_node/lua\\_002dformat.html#lua\\_002df](https://www.gnu.org/software/gettext/manual/html_node/lua_002dformat.html#lua_002df)>

## Object Pascal 格式

### 概要

Object Pascal 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.ObjectPascalFormatCheck`

### 檢查標識符

`object_pascal_format`

### Flag to enable

`object-pascal-format`

### 忽略的標 F

`ignore-object-pascal-format`

### Simple format string example

這 F 有 %d 顆蘋果

### 也參考:

格式化字符串, Object Pascal formatting strings, Free Pascal formatting strings Delphi formatting strings

## 百分比 位符

在 4.0 版本新加入.

### 概要

百分比 位符與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.PercentPlaceholdersCheck`

### 檢查標識符

`percent_placeholders`

### Flag to enable

`percent-placeholders`

### 忽略的標

`ignore-percent-placeholders`

### Simple format string example

這 有 %number% 顆蘋果

### 也參考:

格式化字符串,

## Perl 格式

### 概要

Perl 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.PerlFormatCheck`

### 檢查標識符

`perl_format`

### Flag to enable

`perl-format`

### 忽略的標

`ignore-perl-format`

### Simple format string example

這 有 %d 顆蘋果

### Position format string example

你的余額是 %1 \$ d %2 \$ s

### 也參考:

格式化字符串, Perl `sprintf`, 'Perl 格式字符串' <[https://www.gnu.org/software/gettext/manual/html\\_node/perl\\_002dformat.html](https://www.gnu.org/software/gettext/manual/html_node/perl_002dformat.html)>‘\_

## PHP 格式

### 概要

PHP 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.PHPFormatCheck`

### 檢查標識符

`php_format`

### Flag to enable

`php-format`

### 忽略的標

`ignore-php-format`

### Simple format string example

這有 %d 顆蘋果

### Position format string example

你的余額是 %1 \$ d%2 \$ s

### 也參考:

: 參考: “檢查格式”, “PHP Sprintf 文檔 <<https://www.php.net/manual/en/function.sprintf.php>>’, ‘php 格式字符串 <[https://www.gnu.org/software/gettext/manual/html\\_node/php\\_002dformat.html](https://www.gnu.org/software/gettext/manual/html_node/php_002dformat.html)>’\_

## Python 大括號格式

### 概要

Python 大括號格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.PythonBraceFormatCheck`

### 檢查標識符

`python_brace_format`

### Flag to enable

`python-brace-format`

### 忽略的標

`ignore-python-brace-format`

### Simple format string

這有 {} 顆蘋果

### Named format string example

你的余額是 {COALE} {COMPORY}

### 也參考:

格式化字符串, Python brace 格式, ‘Python 格式字符串 <[https://www.gnu.org/software/gettext/manual/html\\_node/python\\_002dformat.html](https://www.gnu.org/software/gettext/manual/html_node/python_002dformat.html)>’\_

## Python 格式

### 概要

Python 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.PythonFormatCheck`

### 檢查標識符

`python_format`

### Flag to enable

`python-format`

### 忽略的標

`ignore-python-format`

### Simple format string

這有 %d 顆蘋果

### Named format string example

Your balance is %(amount)d %(currency)s

### 也參考:

格式化字符串, [Python string formatting](https://www.gnu.org/software/gettext/manual/html_node/python_002dformat.html), ‘Python 格式字符串 <[https://www.gnu.org/software/gettext/manual/html\\_node/python\\_002dformat.html](https://www.gnu.org/software/gettext/manual/html_node/python_002dformat.html)>’\_

## Qt 格式

### 概要

Qt 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.qt.QtFormatCheck`

### 檢查標識符

`qt_format`

### Flag to enable

`qt-format`

### 忽略的標

`ignore-qt-format`

### Position format string example

這有 %1 顆蘋果

### 也參考:

格式化字符串, [Qt QString::arg\(\)](#)

## Qt 數格式

### 概要

Qt 數格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.qt.QtPluralCheck`

### 檢查標識符

`qt_plural_format`

### Flag to enable

`qt-plural-format`

### 忽略的標

`ignore-qt-plural-format`

### Plural format string example

“There are %Ln apple(s)” (譯: 無中文使用情境, 因蘋果我們不用加 s)

### 也參考:

: ref: *check-formats*, ‘[qt i18n 指南 <https://doc.qt.io/qt-5/i18n-source-translation.html#andling-purals>](https://doc.qt.io/qt-5/i18n-source-translation.html#andling-purals)’\_

## Ruby 格式

### 概要

Ruby 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.ruby.RubyFormatCheck`

### 檢查標識符

`ruby_format`

### Flag to enable

`ruby-format`

### 忽略的標

`ignore-ruby-format`

### Simple format string example

這有%d 顆蘋果

### Position format string example

你的余額是%1 \$ f%2 \$ s

### Named format string example

“你的余額是%+.2 <COMPELL> S“的2 <金額>

### Named template string

你的余額是%{ulity}%{currency}

### 也參考:

格式化字符串, [Ruby Kernel#sprintf](#)



## Scheme 格式

### 概要

Scheme 格式字串與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.SchemeFormatCheck`

### 檢查標識符

`scheme_format`

### Flag to enable

`scheme-format`

### 忽略的標

`ignore-scheme-format`

### Simple format string example

這有 ~d 顆蘋果

### 也參考:

: 參考: “檢查格式”, “SRFI 28 <<https://srfi.schemers.org/srfi-28/srfi-28.html>>”, ‘chicken scheme 格式 <<https://wiki.call-cc.org/eggref/5/格式>>’, ‘軸套方案格式化輸出 <[https://www.gnu.org/software/guile/manual/html\\_node/formatted-output.html](https://www.gnu.org/software/guile/manual/html_node/formatted-output.html)>’

## Vue I18n 格式

### 概要

Vue I18n 格式與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.format.VueFormattingCheck`

### 檢查標識符

`vue_format`

### Flag to enable

`vue-format`

### 忽略的標

`ignore-vue-format`

### Named formatting

這有 {count} 顆蘋果

### Rails i18n formatting

這有 %{count} 顆蘋果

### Linked locale messages

`@:message.dio @:message.the_world!`

### 也參考:

: ref: *check-formats*, 查看 i18n 格式 <<https://kazupon.github.io/vue-i18n/guide/formatting.html>>, 查看 i18n 鏈接本地消息 <<https://kazupon.github.io/vue-i18n/guide/messages.html#linked-locale-messages>>

## 已經翻譯過

### 概要

字串之前已經有翻譯過

### 範圍

all strings

### 檢查類

`weblate.checks.consistency.TranslatedCheck`

### 檢查標識符

translated

### 忽略的標

ignore-translated

表示已經翻譯了一個字符串。當翻譯在 VCS 中或否則失時，可能會發生這種情。

## 不一致

### 概要

此專案中的這個字串有一種以上的翻譯，或是在某些組件未翻譯。

### 範圍

all strings

### 檢查類

`weblate.checks.consistency.ConsistencyCheck`

### 檢查標識符

inconsistent

### 忽略的標

ignore-inconsistent

Weblate 檢查項目中所有翻譯中相同字符串的翻譯，以幫助您保持一致的翻譯。

檢查失敗了項目中的一個字符串的不同轉。這也可能導致顯示檢查中的不一致。您可以在以下網站上找到此字符串的其他翻譯：Guilabel: “其他出現” 選項卡。

This check applies to all components in a project that have 允許翻譯再用 turned on.

---

**提示:** For performance reasons, the check might not find all inconsistencies, it limits number of matches.

---

---

**備:** This check also fires in case the string is translated in one component and not in another. It can be used as a quick way to manually handle strings which are untranslated in some components just by clicking on the *Use this translation* button displayed on each line in the *Other occurrences* tab.

您可以使用自動翻譯 附加元件來自動翻譯從另一個組件中已翻譯字串到新添加而未翻譯的字串。

---

## 也參考:

跨組件保持翻譯一致

## 使用 Kashida letter

在 3.5 版本新加入.

### 概要

不應該使用裝飾性的卡希達對齊字母

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.KashidaCheck`

### 檢查標識符

`kashida`

### 忽略的標 F

`ignore-kashida`

裝飾 kashida 字母不應該在翻譯中使用。這些也稱 F TatWeel。

### 也參考:

[Kashida on Wikipedia](#)

## Markdown 連結

在 3.5 版本新加入.

### 概要

Markdown 連結與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.markup.MarkdownLinkCheck`

### 檢查標識符

`md-link`

### Flag to enable

`md-text`

### 忽略的標 F

`ignore-md-link`

Markdown 鏈接不匹配源。

### 也參考:

[Markdown links](#)

## Markdown 參照

在 3.5 版本新加入.

### 概要

Markdown 連結參照與來源不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.markup.MarkdownRefLinkCheck`

#### 檢查標識符

md-reflink

#### Flag to enable

md-text

#### 忽略的標[F]

ignore-md-reflink

Markdown 鏈接引用不匹配源。

也參考:

[Markdown links](#)

## Markdown 語法

在 3.5 版本新加入.

#### 概要

Markdown 語法與來源不符

#### 範圍

已翻譯好的字串

#### 檢查類

`weblate.checks.markup.MarkdownSyntaxCheck`

#### 檢查標識符

md-syntax

#### Flag to enable

md-text

#### 忽略的標[F]

ignore-md-syntax

Markdown 語法與來源不符

也參考:

[Markdown span elements](#)

## 翻譯最大長度

#### 概要

翻譯不該超過指定長度

#### 範圍

已翻譯好的字串

#### 檢查類

`weblate.checks.chars.MaxLengthCheck`

#### 檢查標識符

max-length

#### Flag to enable

max-length

#### 忽略的標[F]

ignore-max-length

檢查翻譯的長度是否可匹配可用的空間。這只檢查翻譯字符的長度。

與其他檢查不同，標[F]應該被設置[F] “鍵：價值”（價值）對，如 “Max-Length: 100”。

**提示：** 此檢查查看字符數，使用比例字體呈現文本時可能不是最佳度量。： Ref: “Check-Max-size” 檢查確實檢查了文本的實際渲染。

“替”標可能也很有用，在檢查字符串之前展開 PLATEABLE。

When `xml-text` flag is also used, the length calculation ignores XML tags.

## 翻譯的最大長度

### 概要

翻譯的呈現文字不該超過指定長度

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.render.MaxSizeCheck`

### 檢查標識符

`max-size`

### Flag to enable

`max-size`

### 忽略的標

`ignore-max-size`

在 3.7 版本新加入。

翻譯渲染文本不應超過給定的大小。它呈現出包裝的文本，檢查它是否適合給定邊界。

此檢查需要一個或兩個參數 - 最大寬度和最大行數。如果未提供行數，則考慮一行文本。

您還可以通過 “font- \*” 指令配置已使用的字體（請參： ref: “custom-check”），例如隨圖的翻譯標，使用 `ubuntu` 字體大小 22 呈現的文本應該適合兩行和 500 像素：

```
max-size:500:2, font-family:ubuntu, font-size:22
```

**提示：** 您可能希望設置 “字體 - \*” 指令： REF: “組件”（Component）以具有組件中的所有字符串配置的相同字體。您可以每字符串覆蓋每個字符串的值，以防每個字符串自定義它。

“替”標可能也很有用，在檢查字符串之前展開 PLATEABLE。

When `xml-text` flag is also used, the length calculation ignores XML tags.

## 也參考：

管理字型, 使用標自定義行, 翻譯最大長度

## Mismatched \n

### 概要

譯文中的行數 \n 和源文不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.EscapedNewlineCountingCheck`

### 檢查標識符

`escaped_newline`

**忽略的標**`ignore-escaped-newline`

通常跳字元與行符號對於格式化程序輸出很重要。如果在翻譯中的 `\n` 字義不匹配源，則檢查失敗。

**冒號不相符****概要**

翻譯有和來源一致以冒號結尾

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.EndColonCheck`**檢查標識符**`end_colon`**忽略的標**`ignore-end-colon`

檢查冒號在源和翻譯之間冒號。還檢查了冒號的存在，以便他們不屬於哪種語言（中文或日語）。

**也參考：**

[Colon on Wikipedia](#)

**節號不相符****概要**

翻譯有和來源一致以節號結尾

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.EndEllipsisCheck`**檢查標識符**`end_ellipsis`**忽略的標**`ignore-end-ellipsis`

檢查尾部橢圓在源和轉之間尾部橢圓。這只檢查真正的省略號（...）不適用於三個小點（...）。省略號通常比打印中的三個點更好，且文本到語音更好地聽起來更好。

**也參考：**

[Ellipsis on Wikipedia](#)

**驚嘆號不相符****概要**

翻譯有和來源一致以驚嘆號結尾

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.EndExclamationCheck`

**檢查標識符**`end_exclamation`**忽略的標**`ignore-end-exclamation`

檢查源和翻譯之間 的感嘆號。還檢查了感嘆號的存在，他們不屬於哪種語言（中國，日語，韓國，亞美尼亞，林欄，緬甸或 NKO）。

**也參考：**

Wikipedia 的感嘆號 <[https://en.wikipedia.org/wiki/exclamation\\_mark](https://en.wikipedia.org/wiki/exclamation_mark)>‘\_

**句號不相符****概要**

翻譯 有和來源一致以句號結尾

**範圍**

已翻譯好的字串

**檢查類**

`weblate.checks.chars.EndStopCheck`

**檢查標識符**`end_stop`**忽略的標**`ignore-end-stop`

檢查在源和翻譯之間 完整停止。檢查全部停止的存在，以便他們不屬於哪種語言（中文，日語，Devanagari 或 Urdu）。

**也參考：**

[wikipedia](#)

**問號不相符****概要**

翻譯 有和來源一致以問號結尾

**範圍**

已翻譯好的字串

**檢查類**

`weblate.checks.chars.EndQuestionCheck`

**檢查標識符**`end_question`**忽略的標**`ignore-end-question`

檢查源和翻譯之間 的問號。還檢查了問號的存在，以便他們不屬於哪些語言（亞美尼亞，阿拉伯語，中國，韓國，日語，埃塞俄比亞，vai 或 coptic）。

**也參考：**

‘[wikipedia](https://en.wikipedia.org/wiki/question_mark) 上的問號 <[https://en.wikipedia.org/wiki/question\\_mark](https://en.wikipedia.org/wiki/question_mark)>‘\_

## 分號不相符

### 概要

翻譯 有和來源一致以分號結尾

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.EndSemicolonCheck`

### 檢查標識符

`end_semicolon`

### 忽略的標

`ignore-end-semicolon`

Checks that semicolons at the end of sentences are replicated between both source and translation.

### 也參考:

[Semicolon on Wikipedia](#)

## 斷列符不相配

### 概要

譯文中的 行數和源文不符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.NewLineCountCheck`

### 檢查標識符

`newline-count`

### 忽略的標

`ignore-newline-count`

通常 行符號對於格式化程序輸出很重要。如果在翻譯中的 `\n` 字義不匹配源，則檢查失敗。

## 缺少 數形

### 概要

有些 數形未翻譯

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.consistency.PluralsCheck`

### 檢查標識符

`plurals`

### 忽略的標

`ignore-plurals`

檢查已翻譯所有復數形式的源字符串。可以在字符串定義中找到每個 數表單的具體細節。

在某些情 下，未能填寫 數表格將導致多種形式使用時顯示任何 容。



## F 位符

在 3.9 版本新加入.

### 概要

翻譯缺少一些 F 位符

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.placeholders.PlaceholderCheck`

### 檢查標識符

`placeholders`

### Flag to enable

`placeholders`

### 忽略的標 F

`ignore-placeholders`

在 4.3 版本變更: You can use regular expression as placeholder.

在 4.13 版本變更: With the `case-insensitive` flag, the placeholders are not case-sensitive.

翻譯缺少一些 F 位符。這些是從翻譯文件中提取的，或者使用 “” 手動定義，更多可以用冒號分開，可以引用帶空間的字串：

```
placeholders:$URL$: $TARGET$: "some long text"
```

如果您對占位符有一些語法，則可以使用正則表達式：

```
placeholders:r"%[^% ]%"
```

You can also have case insensitive placeholders:

```
placeholders:$URL$: $TARGET$, case-insensitive
```

## 也參考:

使用標 F 自定義行 F

## 標點符號間距

在 3.9 版本新加入.

### 概要

兩個標點符號前 F 有不斷行空格

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.PunctuationSpacingCheck`

### 檢查標識符

`punctuation_spacing`

### 忽略的標 F

`ignore-punctuation-spacing`

檢查雙標點符號（驚嘆號，問號，分號和冒號）之前存在不可破壞的空間。此規則僅以少數選定的語言使用法語或 BRETON，其中 PARD 標 F 符號前的空間是印刷規則。

也參考:

‘wikipedia 的法語和英語間距 <[https://en.wikipedia.org/wiki/history\\_of\\_sentence\\_spacing#french\\_and\\_english\\_spacing](https://en.wikipedia.org/wiki/history_of_sentence_spacing#french_and_english_spacing)>‘\_

## 正則表達式

在 3.9 版本新加入.

### 概要

翻譯不符正則表達式

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.placeholders.RegexCheck`

### 檢查標識符

`regex`

### Flag to enable

`regex`

### 忽略的標 F

`ignore-regex`

翻譯與正則表達式不匹配。表達式要么從翻譯文件中提取，也可以使用 “(” ) 來定義：

```
regex:^(foo|bar$
```

## 相同 F 數形

### 概要

有些 F 數形 F 有以相同方式翻譯

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.consistency.SamePluralsCheck`

### 檢查標識符

`same-plurals`

### 忽略的標 F

`ignore-same-plurals`

如果在翻譯中復制了一些 F 數，請檢查失敗。在大多數語言中他們必須不同。

## 開頭 F 列

### 概要

翻譯 F 有和來源一致以 F 列符開頭

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.BeginNewlineCheck`

### 檢查標識符

`begin_newline`

**忽略的標**`ignore-begin-newline`

紐丁通常出現在源字符串中，有效原因，遺漏或添加可以導致翻譯文本使用時的格式化問題。

**也參考：**[列結尾](#)**開頭空格****概要**

翻譯有和來源一致以相同數目的空格開頭

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.BeginSpaceCheck`**檢查標識符**`begin_space`**忽略的標**`ignore-begin-space`

字符串開頭的空間通常用於界面中的縮進，因此重要地保持。

**列結尾****概要**

翻譯有和來源一致以列結尾

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.EndNewlineCheck`**檢查標識符**`end_newline`**忽略的標**`ignore-end-newline`

紐丁通常出現在源字符串中，有效原因，遺漏或添加可以導致翻譯文本使用時的格式化問題。

**也參考：**[開頭列](#)**空格結尾****概要**

翻譯有和來源一致以相同數目的空格結尾

**範圍**

已翻譯好的字串

**檢查類**`weblate.checks.chars.EndSpaceCheck`**檢查標識符**`end_space`

**忽略的標**

ignore-end-space

檢查尾隨空格是否在源和轉之間。

尾隨空間通常用於鄰居元素，因此除它可能會破壞局。

**未更動的翻譯****概要**

翻譯與來源文字相同

**範圍**

已翻譯好的字串

**檢查類**

weblate.checks.same.SameCheck

**檢查標識符**

same

**忽略的標**

ignore-same

如果源和相應的轉字符串相同，則會發生到至少一個形式中。忽略所有語言中常見的一些字符串，離各種標記。這少了誤報的數量。

此檢查可以幫助查找錯誤誤解的字符串。

此檢查的默認行是從檢查中從置黑名單中排除單詞。這些是經常未被翻譯的單詞。這對於避免短串上的誤報是有用的，該假陽性僅由單個單詞組成，該單詞具有多種語言中的相同。通過添加“嚴格的”“標記至字符串或組件，可以禁用此黑名單。

**也參考：**

組件配置, 使用標自定義行

**不安全的 HTML**

在 3.9 版本新加入.

**概要**

翻譯使用了不安全的 HTML 標示

**範圍**

已翻譯好的字串

**檢查類**

weblate.checks.markup.SafeHTMLCheck

**檢查標識符**

safe-html

**Flag to enable**

safe-html

**忽略的標**

ignore-safe-html

翻譯使用不安全的 HTML 標記。必須使用“安全 - HTML”標（參見：REF：“定制檢查”）用此檢查。還有伴隨的 AutoFixer，可以自動消毒標記。

---

**提示：** When md-text flag is also used, the Markdown style links are also allowed.

---

**也參考:**

HTML 檢查由 “Mozilla 開發的” 裸照 <<https://blach.readthedocs.io/>>\_ 庫執行。

**網址**

在 3.5 版本新加入。

**概要**

此譯文未包含網址

**範圍**

已翻譯好的字串

**檢查類**

`weblate.checks.markup.URLCheck`

**檢查標識符**

`url`

**Flag to enable**

`url`

**忽略的標**

`ignore-url`

翻譯不包含 URL。僅在單位標記包含 URL 時才會觸發這一點。在這種情況下，翻譯必須是有效的 URL。

**XML 標記****概要**

翻譯中的 XML 標與來源不符

**範圍**

已翻譯好的字串

**檢查類**

`weblate.checks.markup.XMLTagsCheck`

**檢查標識符**

`xml-tags`

**忽略的標**

`ignore-xml-tags`

這通常意味著生的輸出看起來不同。在大多數情況下，這不是改變翻譯的所需結果，但偶爾就是。

檢查源和翻譯之間是否 XML 標記。

---

**備:** This check is disabled by the `safe-html` flag as the HTML cleanup done by it can produce HTML markup which is not valid XML.

---

## XML 語法

在 2.8 版本新加入.

### 概要

翻譯標記 無效 XML

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.markup.XMLValidityCheck`

### 檢查標識符

`xml-invalid`

### 忽略的標

`ignore-xml-invalid`

XML 標記無效。

---

**備:** This check is disabled by the `safe-html` flag as the HTML cleanup done by it can produce HTML markup which is not valid XML.

---

## 零寬度空格

### 概要

翻譯中包含有額外的零寬度的空白字元

### 範圍

已翻譯好的字串

### 檢查類

`weblate.checks.chars.ZeroWidthSpaceCheck`

### 檢查標識符

`zero-width-space`

### 忽略的標

`ignore-zero-width-space`

零寬度空間 (`<u + 200b>`) 字符用於打破單詞中的消息 (Word 包裝)。

由於它們通常被錯誤插入，因此在翻譯中存在後觸發此檢查。使用此字符時，某些程序可能會出現問題。

**也參考:**

[wikipedia](#)

## 1.6.4 Source checks

源檢查可以幫助開發人員提高源字符串的質量。

## 節號

### 概要

該字串使用三個點（…）而非節號（…）

### 範圍

來源字串

### 檢查類

`weblate.checks.source.EllipsisCheck`

### 檢查標識符

`ellipsis`

### 忽略的標

`ignore-ellipsis`

當字符串使用三個點時，這會失敗（…）時，它應該使用省略號字符（“…”）。

在大多數情況下，使用 Unicode 字符更好的方法，看起來更好，且文本到語音可能會更好地聽起來更好。

### 也參考：

[Ellipsis on Wikipedia](#)

## ICU MessageFormat 語法

在 4.9 版本新加入。

### 概要

ICU MessageFormat 字串有語法錯誤。

### 範圍

來源字串

### 檢查類

`weblate.checks.icu.ICUSourceCheck`

### 檢查標識符

`icu_message_format_syntax`

### Flag to enable

`icu-message-format`

### 忽略的標

`ignore-icu-message-format`

### 也參考：

[ICU MessageFormat](#)

## 長期未翻譯

在 4.1 版本新加入。

### 概要

字串很久都沒有翻譯

### 範圍

來源字串

### 檢查類

`weblate.checks.source.LongUntranslatedCheck`

**檢查標識符**`long_untranslated`**忽略的標 F**`ignore-long-untranslated`

When the string has not been translated for a long time, it can indicate a problem in a source string making it hard to translate.

**多項未通過查核****概要**

多個語言的翻譯有未通過查核的項目

**範圍**

來源字串

**檢查類**`weblate.checks.source.MultipleFailingCheck`**檢查標識符**`multiple_failures`**忽略的標 F**`ignore-multiple-failures`

此字符串的許多翻譯都 F 有質量檢查。這通常是可以進行某些東西來改進源字符串的指示。

此檢查失敗通常可以在句子末尾的缺失完整停止 F 失或類似的次要問題傾向於在翻譯中解 F，而在源字符串中將其更好。

**多個未命名變數**

在 4.1 版本新加入。

**概要**

字串中有多個未命名變數，翻譯者將難以重新排序

**範圍**

來源字串

**檢查類**`weblate.checks.format.MultipleUnnamedFormatsCheck`**檢查標識符**`unnamed_format`**忽略的標 F**`ignore-unnamed-format`

字符串中有多個未命名的變量，使翻譯是不可能重新排序的轉 F 器。

考慮使用命名變量，而不是允許翻譯器重新排序。



## 未數化

### 概要

該字串數，但有使用數形

### 範圍

來源字串

### 檢查類

`weblate.checks.source.OptionalPluralCheck`—

### 檢查標識符

`optional_plural`

### 忽略的標

`ignore-optional-plural`

該字符串用作數，但不使用數形式。如果您的翻譯系統支持此情，您應該使用它的數感知變體。例如，在 Python 中用 GetText 可能是：

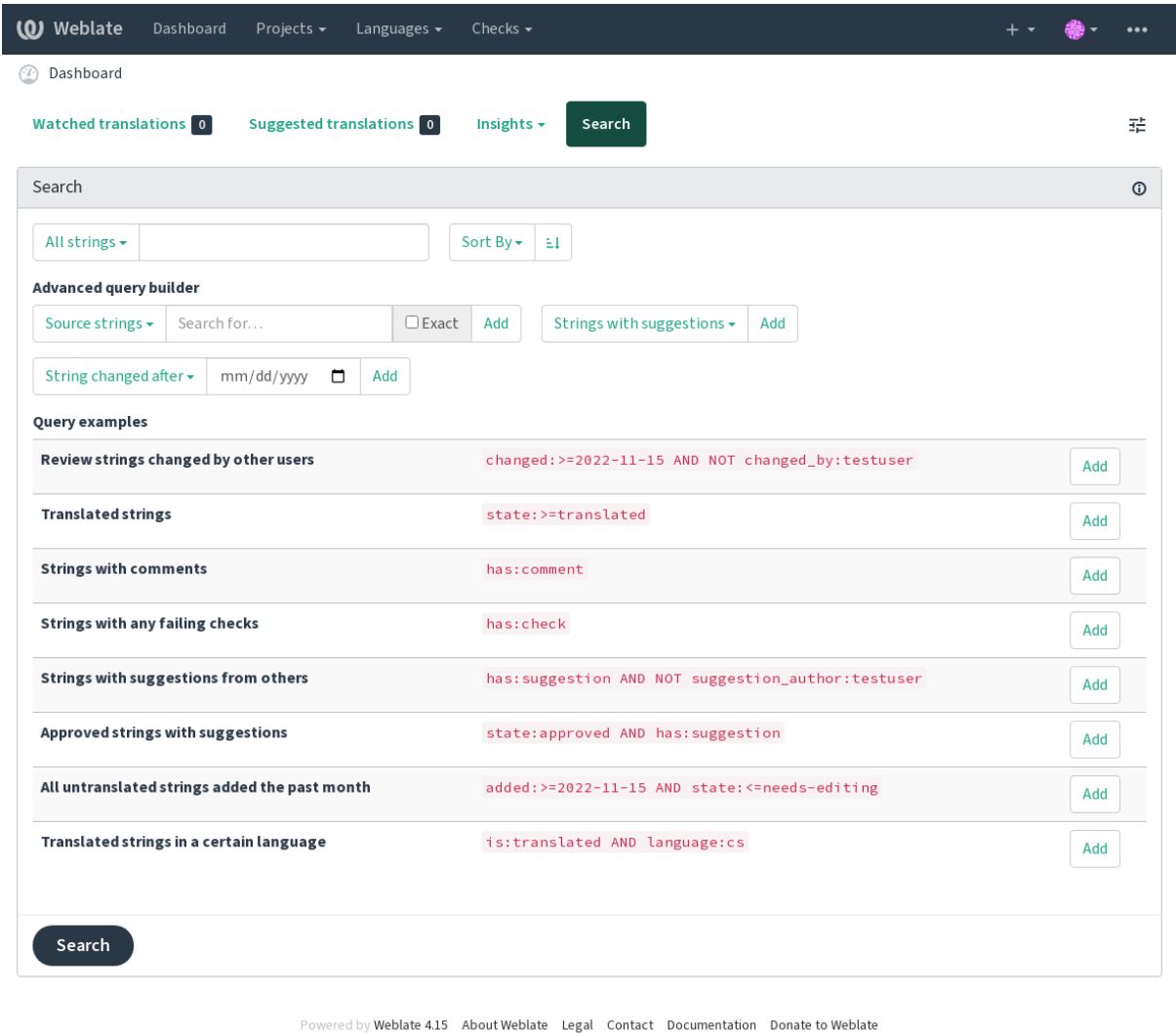
```
from gettext import ngettext
print(ngettext("Selected %d file", "Selected %d files", files) % files)
```

## 1.7 搜索

在 3.9 版本新加入。

使用布爾操作，括號或字段特定查找的高級查詢可用於查找所需的字符串。

When no field is defined, the lookup happens on source, target, and context strings.



### 1.7.1 Simple search

鍵入搜索框中的任何短語都分成單詞。顯示包含任何一個的字符串。要查找一個確切的短語，將“SearchPhrase”放入引號（單個（`『`）和雙（`“`）引號將工作）：”這是一個引用的字符串“````或`’`另一個引用的字符串`』`“。

### 1.7.2 Fields

**source:TEXT**  
Source string case-insensitive search.

**target:TEXT**  
Target string case-insensitive search.

**context:TEXT**  
Context string case-insensitive search.

**key:TEXT**  
Key string case-insensitive search.

**note:TEXT**  
Source string description case-insensitive search.

**location:TEXT**

Location string case-insensitive search.

**priority:NUMBER**

String priority.

**added:DATETIME**

將字符串添加到 Weblate 時的時間戳。

**state:TEXT**

Search for string states (approved, translated, needs-editing, empty, read-only), supports *Field operators*.

**pending:BOOLEAN**

在彎曲以洗到 VCS。

**has:TEXT**

Search for string having attributes - plural, context, suggestion, comment, check, dismissed-check, translation, variant, screenshot, flags, explanation, glossary, note, label.

**is:TEXT**

搜尋擱置中的翻譯 (pending)。也可搜尋其他狀態的翻譯字串 (approved, translated, untranslated, needs-editing, read-only)。

**language:TEXT**

String target language.

**component:TEXT**

Component slug or name case-insensitive search, see [組件標識串](#) and [組件名憲](#).

**project:TEXT**

Project slug, see *URL slug*.

**changed\_by:TEXT**

作者使用給定用名更改字符串。

**changed:DATETIME**

字符串容在日期更改，支持：參考：“搜索操作員”。

**change\_time:DATETIME**

字符串已在日期更改，支持：REF：“搜索操作員”，與“已更改”（更改）提供此包含不會更改容的事件，且您可以使用“Change\_Action”“”應用“使用”自定義操作過濾“。

**change\_action:TEXT**

在更改動作上的過濾器，與“change\_time”一起有用。接受更改操作的英語名稱，引用以及帶有連字符替的空格或小寫和空格。請參：參考：“搜索 - 更改”以獲取示例。

**check:TEXT**

String has failing check, see [檢查和修復](#) for check identifiers.

**dismissed\_check:TEXT**

String has dismissed check, see [檢查和修復](#) for check identifiers.

**comment:TEXT**

Search in user comments.

**resolved\_comment:TEXT**

Search in resolved comments.

**comment\_author:TEXT**

Filter by comment author.

**suggestion:TEXT**

Search in suggestions.

**suggestion\_author:TEXT**

Filter by suggestion author.



### 1.7.8 Predefined queries

您可以在搜索頁面上選擇預定義查詢，這允許您快速訪問最常見的搜索：

Webplate

Dashboard

Projects

Languages

Checks

WebplateOrg / Django / Czech / Translate

translated 96%

<<

<

1/1

>

>>

Custom search

'%(count)s word'

Position and priority

⌵

Translation

English

Singular

%(count)s word

Plural

%(count)s words

Czech, One

%(count)s slovo

Czech, Few

%(count)s slova

Czech, Other

%(count)s slov

Plural formula: (n==1) ? 0 : (n>=2 && n<=4) ? 1 : 2

Needs editing

Save and continue

Save and stay

Suggest

Skip

Untranslated strings • state:empty

Unfinished strings • state:<translated

Translated strings • state:>=translated

Strings marked for edit • state:needs-editing

Strings with suggestions • has:suggestion

Strings with variants • has:variant

Strings with labels • has:label

Strings with context • has:context

Unfinished strings without suggestions • state:<translated AND NOT has:suggestion

Strings with comments • has:comment

Strings with any failing checks • has:check

Approved strings • state:approved

Strings waiting for review • state:translated

Nearby strings 20

Comments

Automatic suggestions

Other languages 3

History

New comment

Comment on this string for fellow translators and developers to read.

Scope

Translation comment, discussions with other translators

Is your comment specific to this translation, or generic for all of them?

New comment

You can use Markdown and mention users by @username.

Save

Glossary

English

Czech

No related strings found in the glossary.

Add term to glossary

String information

Screenshot context

No screenshot currently associated.

Add screenshot

Explanation

No explanation currently provided.

Labels

No labels currently set.

Flags

python-format

Source string location

weblate/templates/translation.html149

String age

6 seconds ago

Source string age

6 seconds ago

Translation file

weblate/locale/cs/LC\_MESSAGES/django.p

o, string 5

## 1.7.9 Ordering the results

根據您的需要，有許多選擇字符串：

The screenshot shows the Weblate web interface for a project named 'Django' in the 'Czech' language. The main content area displays a translation form for the string 'The string uses three dots (...) instead of an ellipsis character (...)'. The form includes fields for the source string, target string, and a 'Needs editing' checkbox. Below the form are buttons for 'Save and continue', 'Save and stay', 'Suggest', and 'Skip'. A dropdown menu is open, showing options for ordering the results: Position and priority, Position, Priority, Labels, Source string, Target string, String age, Number of words, Number of comments, Number of failing checks, and Key. The right sidebar contains a 'Glossary' section and a 'String information' section, which includes details such as 'Screenshot context', 'Explanation', 'Labels', 'Flags', 'Source string location', 'String age', 'Source string age', and 'Translation file'.

## 1.8 翻譯工作流

使用 Weblate 是一個流程，使您的用戶更接近您，通過讓您更接近您的翻譯。由您決定您想要使用的功能數量。

以下不是配置 Weblate 的方法的完整列表。您可以將其他工作流基於此處列出的最常見的示例。

### 1.8.1 翻譯訪問

ref:access control <access-control> 有作一個整體在工作流程中被詳細討論，因它的大多數選項都可以應用於任何工作流。請參考有關如何管理對翻譯的訪問的相關文檔。

在以下各章中，任何用戶都是指有權訪問翻譯的用戶。如果項目是公共項目，則可以是任何經過身份驗證的用戶，也可以是具有項目 *Translate* 權限的用戶。

## 1.8.2 翻譯狀態

每個翻譯的字符串可以處於以下狀態之一：

### 未翻譯

翻譯是空的，取於文件格式，翻譯是否可能存儲在文件中。

### 需要編輯

翻譯需要編輯，這通常是源字符串更改、模糊匹配或譯員操作的結果。翻譯存儲在文件中，具體取於文件格式，它可能被標需要編輯（例如，當它在 Gettext 文件中獲得一個 fuzzy 標記）。

### 等候檢

翻譯已完成，但未進行審核。它作有效翻譯存儲在文件中。

### 已核可

翻譯已在審核中得到批准。翻譯者不能再更改它，只能由審者更改。譯者只能向其中添加建議。

This state is only available when reviews are enabled.

### 建議

建議僅存儲在 Weblate 中，而不存儲在翻譯文件中。

The states are represented in the translation files when possible.

---

**提示：** In case file format you use does not support storing states, you might want to use 將未變動的翻譯標記「需要編輯」 add-on to flag unchanged strings as needing editing.

---

### 也參考：

翻譯類型功能, 翻譯工作流

## 1.8.3 直接翻譯

這是小型團隊最常用的設置，任何人都可以直接翻譯。這也是 Weblate 中的默認設置。

- 任何用都可以編輯翻譯。
- 當翻譯者不確定更改時，建議是建議更改的可選方法。

Setting	Value	備
用檢	關閉	Configured at project level.
用建議	開	用可以在不確定時提出建議，這很有用。
建議投票	關閉	
自動接受建議	0	
翻譯組	使用者	或使用 <i>per-project access control</i> 的翻譯權限。
審核者組	N/A	不曾用過。

## 1.8.4 同行評審

使用此工作流程，任何人都可以添加建議，且需要其他成員的同意才能被接受翻譯。

- 任何用都可以添加建議。
- 任何用都可以對建議投票。
- 當給定預定數量的投票時，建議就變成翻譯。

Setting	Value	備
用檢	關閉	Configured at project level.
用建議	開	
建議投票	關閉	
自動接受建議	1	您可以設置更高的值，以要求更多的同行評審。
翻譯組	使用者	或使用 <i>per-project access control</i> 的翻譯權限。
審核者組	N/A	未使用，所有翻譯員都審。

### 1.8.5 專門的審核者

在 2.18 版本新加入：從 Weblate 2.18 開始，支持正確的審核工作流。

使用專門的審核者，您有兩組用，一組可以提交翻譯，而另一組可以審核它們以確保翻譯一致且質量良好。

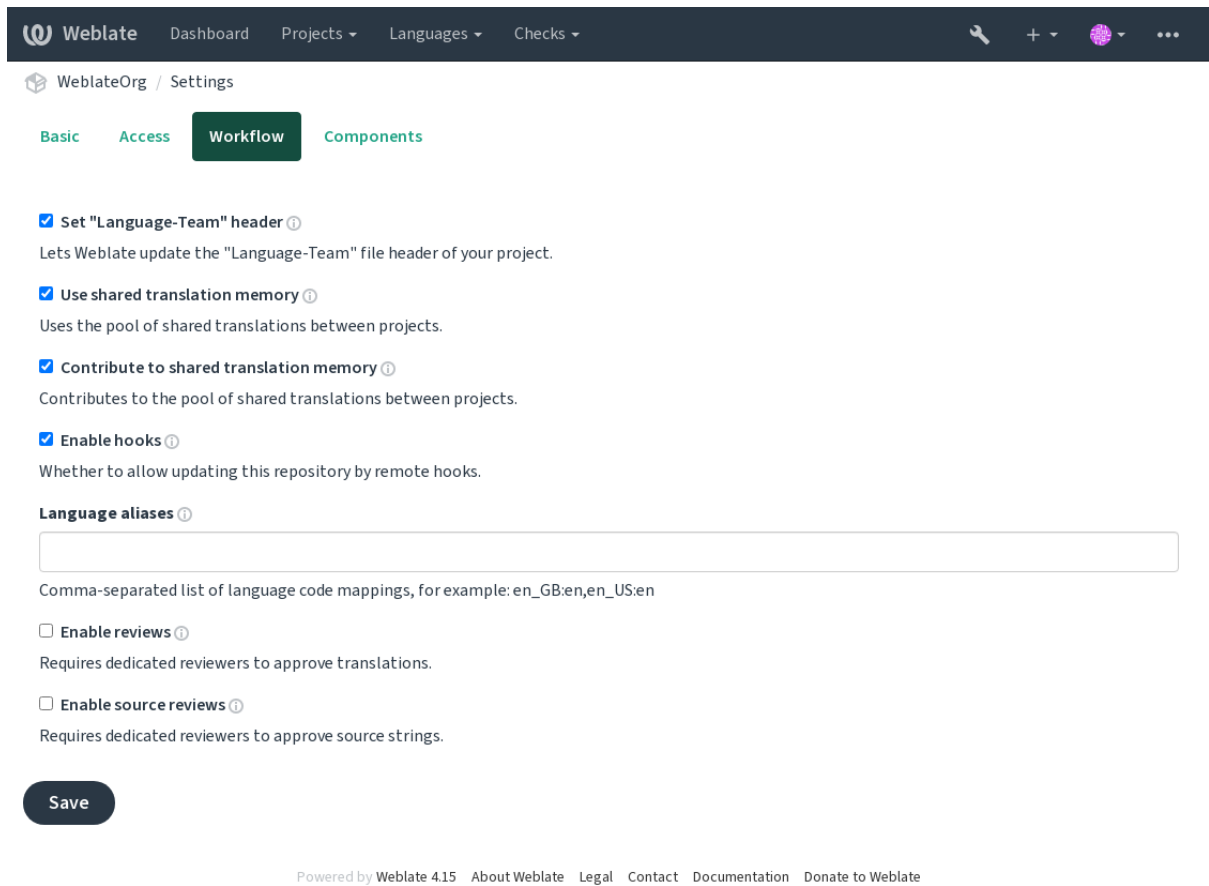
- 任何用都可以編輯未批准的翻譯。
- *Reviewer* can approve / unapprove strings.
- 審核者可以編輯所有翻譯（包括批准的翻譯）。
- 建議還可以用於建議更改已批准的字符串。

Setting	Value	備
用檢	開	Configured at project level.
用建議	關閉	用可以在不確定時提出建議，這很有用。
建議投票	關閉	
自動接受建議	0	
翻譯組	使用者	或使用 <i>per-project access control</i> 的翻譯權限。
審核者組	審核員	或使用 <i>per-project access control</i> 的權限進行 審查。

### 1.8.6 打開審核

可以在項目設置的 *Workflow* 子頁面中的項目配置中打開審核（位於 *Manage* → *Settings* 菜單中）：



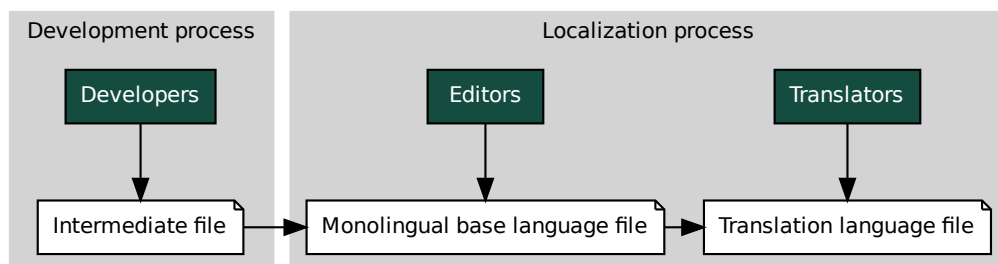


備註：根據 Weblate 配置的不同，該設置可能對您不可用。例如，在 Hosted Weblate 上，這不適用於免費管轄的項目。

### 1.8.7 源字符串的質量網關

在許多情況下，原始源語言字符串來自開發人員，因為它們編寫代碼提供初始字符串。然而，開發人員通常不是源語言中的母語者，且不提供所需的源字符串質量。中間轉可以幫助您解決此問題 - 開發人員和翻譯人員和用之間的字符串存在額外的質量網關。

通過設置 REF：“組件中間”，此文件將用作字符串的源，但它將被編輯源語言以波動它。一旦字符串在源語言中準備就緒，它也可以用於翻譯成轉其他語言。



**也參考:**

中間語言檔案, 單語的基底語言檔, 雙語和單語格式

## 1.8.8 源字符串查

通過允許用來源檢, 查過程可以應用到源字符串上。一旦允許, 用可以匯報源字符串種的情。實際過程依賴於使用雙語言還是單語言格式。

對於單語言格式, 源字符串查的行與專門的審核者相似——一旦源字符串匯報了情, 就會被標記 *Needs editing*。

雙語言格式不允許直接編輯源字符串 (他們典型地是從源代碼種直接提取的)。在這種情下, *Source needs review* 標會貼到翻譯者回到的字符串上。可以復查這樣的字符串, 在源中編輯或除標。

**也參考:**

雙語和單語格式, 專門的審核者, labels, 評

## 1.9 常見問題

### 1.9.1 配置

#### 如何創建自動化工作流?

Weblate 可以您半自動處理所有翻譯工作。如果授予它對倉庫的推送訪問權限, 則翻譯可以在有交互的情下進行, 除非發生某些合衝突。

1. 新建您的 Git 倉庫, 來告知 Weblate 何時有任何更改, 請參通知勾以獲取有關如何執行此操作的信息。
2. 在 Weblate 中的組件配置配置中設置推送 URL, 這使 Weblate 可以將更改推送到倉庫。
3. 在 Weblate 中打開 component 上的:ref:'component-push\_on\_commit', 這將使 Weblate 在 Weblate 發生更改時將更改推送到倉庫。

**也參考:**

持續本地化, *Avoiding merge conflicts*

#### 如何通過 SSH 訪問倉庫?

設置 SSH 密鑰的信息請參訪問存儲庫。

#### 如何修復翻譯中的合衝突?

在翻譯文件在 Weblate 和上游倉庫同時更改時, 合衝突不時發生。通常可以通過在對翻譯文件進行更改前 (例如在運行 msgmerge 前) 合倉庫來避免發生這個情。只要告訴 Weblate 執行起所有的翻譯 (可以在 Manage 菜單中的 *Repository maintenance* 中去做), 合倉庫即可 (如果自動推送有打開的話)。

If you've already encountered a merge conflict, the easiest way to solve all conflicts locally on your machine, is to add Weblate as a remote repository, merge it into upstream and fix any conflicts. Once you push changes back, Weblate will be able to use the merged version without any other special actions.

---

**備:** 取於你的設置, 訪問 Weblate 代碼庫可能需要身份驗證。使用 Weblate 中建的:ref:'git-exporter'時, 你使用名和 API 密鑰進行身份驗證。

---

```
# Commit all pending changes in Weblate, you can do this in the UI as well:
wlc commit
# Lock the translation in Weblate, again this can be done in the UI as well:
wlc lock
# Add Weblate as remote:
git remote add weblate https://hosted.weblate.org/git/project/component/
# You might need to include credentials in some cases:
git remote add weblate https://username:APIKEY@hosted.weblate.org/git/project/
→component/

# Update weblate remote:
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts:
edit ...
git add ...
...
git commit

# Rebase changes (if Weblate is configured to do rebases)
git rebase origin/main

# Push changes to upstream repository, Weblate will fetch merge from there:
git push

# Open Weblate for translation:
wlc unlock
```

如果在 Weblate 中使用多個分支，那麼可以對所有的分支做相同的事：

```
# Add and update Weblate remotes
git remote add weblate-one https://hosted.weblate.org/git/project/one/
git remote add weblate-second https://hosted.weblate.org/git/project/second/
git remote update weblate-one weblate-second

# Merge QA_4_7 branch:
git checkout QA_4_7
git merge weblate-one/QA_4_7
... # Resolve conflicts
git commit

# Merge main branch:
git checkout main
git merge weblate-second/main
... # Resolve conflicts
git commit

# Push changes to the upstream repository, Weblate will fetch the merge from there:
git push
```

在 gettext PO 文件的情況下，有一種方式以半自動方式來合併衝突：

取回並保存 Weblate Git 倉庫的本地克隆。還要得到上游 Git 倉庫的第二個新的本地克隆（也就是需要上游 Git 倉庫的兩份文件：完整的文件和工作文件）：

```
# Add remote:
git remote add weblate /path/to/weblate/snapshot/

# Update Weblate remote:
```

(繼續下一頁)

(繼續上一頁)

```
git remote update weblate

# Merge Weblate changes:
git merge weblate/main

# Resolve conflicts in the PO files:
for PO in `find . -name '*.po'` ; do
    msgcat --use-first /path/to/weblate/snapshot/$PO\
              /path/to/upstream/snapshot/$PO -o $PO.merge
    msgmerge --previous --lang=${PO%.po} $PO.merge domain.pot -o $PO
    rm $PO.merge
    git add $PO
done
git commit

# Push changes to the upstream repository, Weblate will fetch merge from there:
git push
```

**也參考:**

如何導出 Weblate 使用的 Git 倉庫?, 持續本地化, *Avoiding merge conflicts*, Weblate 客戶端

**如何立刻翻譯幾個分支?**

Weblate 支持在一個項目配置 推送翻譯更改。對於每個將其打開的組件配置 (默認行), 所作的更改自動傳遞給其它組件。即使分支本身已經非常多樣化了, 也能以這種方式保持同步, 且不能在他們之間簡單地合翻譯更改。

一旦從 Weblate 合了更改, 就會不得不合這些分支 (依賴於您的開發工作流程), 而差:

```
git merge -s ours origin/maintenance
```

**也參考:**

跨組件保持翻譯一致

**如何翻譯多平台項目?**

Weblate 支持大範圍的文件格式 (請參見 [支持的文件格式](#)), 而最容易的方式時對每個平台使用本地格式。

一旦在一個項目中將所有平台的翻譯文件作組件添加 (請參見: [ref:adding-projects](#)), 就可以立刻使用翻譯傳播特性 (默認打開, 且可以在: [ref:component](#) 中關閉), 來翻譯所有平台的字符串了。

**也參考:**

跨組件保持翻譯一致

**如何導出 Weblate 使用的 Git 倉庫?**

倉庫有什特殊的, 它存在於 `DATA_DIR` 目下面, 被命名 `vcs/<project>/<component>/`。如果有 SSH 範文這台機器, 那可以直接使用倉庫。

對於匿名訪問, 您會想要運行 Git 服務器, 讓它外部世界提供倉庫服務。

此外, 可以在 Weblate 替代使用 *Git 導出器* 而使其自動化。

### 將更改推送回上游的選項是什麼？

這嚴重依賴於您的設置，Weblate 在這個領域是非常靈活的。這是使用 Weblate 的一些工作流程的例子：

- Weblate 自動推送合更改（請參見[如何創建自動化工作流？](#)）。
- 您需要手動告訴 Weblate 去推送（推送需要訪問上游倉庫）。
- 一些人將 Weblate git 倉庫的更改手動合到上游倉庫中。
- 一些人重寫由 Weblate 生成的歷史（例如通過除合提交），合更改，告訴 Weblate 重置上游倉庫中的內容。

當然您可以按您的意願自由混用所有這些方法。

### 如何限制 Weblate 只訪問翻譯，而不向它暴露源代碼？

可以使用 `git submodule` 將翻譯從源代碼中分離出來，而仍將它們至於版本控制之下。

1. 以您的文藝文件創建倉庫。
2. 將其作子模塊添加到您的代碼中：

```
git submodule add git@example.com:project-translations.git path/to/translations
```

3. 將 Weblate 連接到這個倉庫上，它不再需要訪問包含您源代碼的倉庫。
4. 可以從 Weblate 更新帶有翻譯的主倉庫，通過：

```
git submodule update --remote path/to/translations
```

更多細節請詢 `git submodule` 文檔。

### 如何檢查 Weblate 是否被正確地設置？

Weblate 包括一組配置檢查，可以在管理面板中看到，只需按照管理面板中的 *Performance report* 連接，或直接打開 `/manage/performance/` URL 即可。

也參考：

監測 *Weblate, Monitoring Celery status*

### 什麼所有提價由 Weblate <noreply@weblate.org> 執行？

This is the default committer name, configured by `DEFAULT_COMMITTER_EMAIL` and `DEFAULT_COMMITTER_NAME`.

（如果下層的版本管理系統 VCS 支持它的話），每個提交的作者仍會被正確地記進行翻譯的用。

For commits where no authorship is known (for example anonymous suggestions or machine translation results), the authorship is credited to the anonymous user (see `ANONYMOUS_USER_NAME`). You can change the name and e-mail in the management interface.

也參考：

組件配置

## How to move files in the repository without losing history in Weblate?

To keep the history, comments, or screenshots linked to strings after changing the files location you need to ensure that these strings are never deleted in Weblate. These removals can happen in case the Weblate repository is updated, but the component configuration still points to the old files. This makes Weblate assume that it should delete all the translations.

The solution to this is to perform the operation in sync with Weblate:

1. Lock the affected component in Weblate.
2. Commit any pending changes and merge them into the upstream repository.
3. Disable receiving webhooks the 項目配置; this prevents Weblate from immediately seeing changes in the repository.
4. Do any needed changes in the repo (for example using `git mv`), push them to the upstream repository.
5. Change the 組件配置 to match the new setup; upon changing configuration, Weblate will fetch the updated repository and notice the changed locations while keeping existing strings.
6. Unlock the component and re-enable hooks in the project configuration.

## 1.9.2 Usage

### 如何複查其他人的翻譯？

- 在 Weblate 中，基於工作流程有幾種查方式，請參見翻譯工作流。
- 可以訂通知 中做出的任何更改，然後檢查當它們通過電子郵件進入時檢查其他人的貢獻。
- 在翻譯視圖底部有個查工具，可以在那選擇覽給定時間以來其他人進行的翻譯。

也參考：

翻譯工作流

### 如何提供源字符串的反饋？

在翻譯下面的文本標上，可以使用 *Comments* 標來提供源字符串的反饋，或者與其他翻譯者討論。

也參考：

report-source, 評

### 翻譯時如何使用現有的翻譯？

- All translations within Weblate can be used thanks to shared translation memory.
- You can import existing translation memory files into Weblate.
- 使用導入功能將概要導入作翻譯、建議或需要復查的翻譯。這是使用概要或相似的翻譯數據庫一次性翻譯的最好方法。
- You can set up *tmserver* with all databases you have and let Weblate use it. This is good when you want to use it several times during translation.
- 另一個選項是在單一 Weblate 事件中翻譯所有相關項目，這樣同樣可以從其它項目中自動地拾取翻譯。

也參考：

配置自動建議, 自動建議, 翻譯記憶

## Weblate 除了更新翻譯，還更新翻譯文件嗎？

Weblate 嘗試將翻譯文件中的更改限制到最小。對於有些文件格式，很不幸會導致將文件重新格式化。如果想要將文件保持自己的格式化方式，請其使用預提交子。

也參考：

[updating-target-files](#)

## 語言定義來自何處以及如何添加自己的語言定義？

語言定義的基本組包括在 Weblate 和 Translate-toolkit 中。這覆蓋了超過 150 種語言，且包括了數形式和文本方向的信息。

您可以在管理界面自由定義自己的語言，只需要提供與之相關的信息。

也參考：

[語言定義](#)

## Weblate 能將模糊字符串中的更改高亮嗎？

Weblate 支持這個功能，然而它需要數據來顯示差異。

對於 Gettext PO 文件，更新 PO 文件時，必須將參數 `--previous` 傳遞給 `msgmerge`，例如：

```
msgmerge --previous -U po/cs.po po/phpmyadmin.pot
```

對於單語言翻譯，Weblate 可以通過 ID 找到之前的字符串，因此可以自動顯示差異。

## 當已經更新了模板時，什麼 Weblate 仍然顯示舊的字符串？

Weblate 除了允許翻譯者翻譯之外，不會嘗試以任何方式操作翻譯文件。因此當模板或源代碼更改時，它也同樣不會更新翻譯文件。您必須簡單地手動去做，將更改推送到倉庫，然後 Weblate 會自動拾取更改。

---

**備註：** 在更新翻譯文件之前在 Weblate 中完成更改到，這通常是個好主意，因否則的話通常會以一些衝突來結束。

---

例如對於 gettext PO 文件，可以使用 `msgmerge` 工具來更新翻譯文件：

```
msgmerge -U locale/cs/LC_MESSAGES/django.mo locale/django.pot
```

In case you want to do the update automatically, you can install add-on [更新 PO 檔以符合 POT \(msgmerge\)](#).

也參考：

[updating-target-files](#)

## 1.9.3 Troubleshooting

### 請求有時失敗，錯誤信息 “too many open files”（打開文件過多）

有時當您的 Git 倉庫增長太多您有太多倉庫時會發生。壓縮 Git 倉庫會改善這種情況。這樣做的最容易方式是運行：

```
# Go to DATA_DIR directory
cd data/vcs
# Compress all Git repositories
for d in */* ; do
    pushd $d
    git gc
    popd
done
```

也參考：

`DATA_DIR`

### 當訪問網站時，“Bad Request (400)”（錯誤的請求）錯誤信息

這很可能因 `ALLOWED_HOSTS` 配置不當而生。需要包含在 Weblate 上訪問的所有主機名。例如：

```
ALLOWED_HOSTS = ["weblate.example.com", "weblate", "localhost"]
```

也參考：

*Allowed hosts setup*

### “There are more files for the single language (en)”（單一語言‘英語’有多個文件）是什麼意思？

當源語言有翻譯時這會典型發生。Weblate 跟踪源字符串，此保留源字符串。相同語言的附加字符串不會被處理。

- In case the translation to the source language is desired, please change the 來源語言 in the component settings. You might want to use *English (Developer)* as a source language, or utilize 源字符串的質量網關。
- 如果是不需要用到來源語言的翻譯檔，請從儲存庫中將它移除。
- In case the translation file for the source language is needed, but should be ignored by Weblate, please adjust the 語言篩選 to exclude it.

---

**提示：**對於其它語言可能也會得到相似的錯誤信息。在這種情況下最可能的原因是在 Weblate 中幾個文件映射到單語言上。

這可能是由於將過時的語言編碼與新的一起使用（對於日語是“ja”和“jp”），或者包括了特定國家的和通用的語言編碼（`fr``和``fr_FR`）。更多細節請參見 *Parsing language codes*。

---



## 1.9.4 功能

### Weblate 支持 Git 和 Mercurial 意外的 VCS 嗎？

Weblate 當前不支持 `ref:vcs-git`（擴展支持 `ref:vcs-github`、*Gerrit* 和 `ref:vcs-git-svn`）和 `ref:vcs-mercurial` 以外的任何 VCSes，但能對其它 VCSes 寫後端。

還可以在 Git 中使用 *Git remote helpers* 來訪問其它 VCSes。

Weblate 還支持無 VCS 的操作，請參見 *Local files*。

---

**備註：**除了本地支持其它 VCSes，Weblate 需要使用分式 VCS，可能調整來與 Git 和 Mercurial 以外的其它任何 VCSes 工作，但必須有人應用這項支持。

---

#### 也參考：

版本控制整合

### Weblate 如何記錄翻譯者？

Weblate 中所做的每個更改都將以翻譯者的名稱提交到版本控制系統（VCS）中。這樣，每個更改都具有適當的作者身份，您可以使用用於代碼的標準版本控制系統（VCS）工具來進行跟踪。

此外，如果翻譯文件格式支持，則文件頭會更新包含翻譯者的名稱。

#### 也參考：

`list_translators`, `../devel/reporting`

### 什麼 Weblate 制在單一樹中顯示所有 PO 文件？

Weblate 的設計方式是將每個 PO 文件表示單個組件。這對翻譯人員來是有好處的，這樣他們就知道他們實際上在翻譯什麼。

在 4.2 版本變更：Translators can translate all the components of a project into a specific language as a whole.

### Weblate 什麼使用 `sr_Latn` 或 `zh_Hant` 這樣的語言編碼？

These are language codes defined by [RFC 5646](#) to better indicate that they are really different languages instead previously wrongly used modifiers (for @latin variants) or country codes (for Chinese).

Weblate 仍然理解傳統的語言編碼將它們映射到當前的便馬上——例如 “sr@latin” 將被處理 “sr\_Latn” 或者 “zh@CN” 處理 “zh\_Hans”。

---

**備註：** Weblate defaults to POSIX style language codes with underscore, see [語言定義](#) for more details.

---

#### 也參考：

語言定義, 語言代碼類型, 添加新的翻譯

## 1.10 支持的文件格式

Weblate 支持 `translate-toolkit` 理解的大多數翻譯格式，但是每種格式都略有不同，可能會出現未經良好測試的格式問題。

也參考：

[Translation Related File Formats](#)

備：您的應用程序選擇文件格式時，最好在您使用的工具箱/平台中保留一些公認的格式。這樣，您的翻譯人員可以額外使用他們習慣使用的任何工具，且更有可能您的項目做出貢獻。

### 1.10.1 雙語和單語格式

支持 `monolingual` 和 `bilingual` 格式。雙語格式在單個文件中存儲兩種語言——源和翻譯（典型示例是 `GNU gettext`，`XLIFF` 或蘋果 `iOS` 字符串 字符串）。另一方面，單語格式通過 ID 識字符串，每個語言文件僅包含那些語言到任何給定語言（通常是 `Android 字符串資源`）的映射。兩種變體都使用某些文件格式，請參見下面的詳細明。

了正確使用單語文件，Weblate 要求訪問一個包含完整字符串列表的文件，以與其源一起翻譯——該文件在 Weblate 中稱 `ref:component-template`，管命名方式可能會有所不同。

另外，可以利用 `中間語言檔案` 擴展此工作流程，以包括開發人員提供的字符串，但不要在最終的字符串中使用。

### 1.10.2 自動偵測

Weblate 可以自動檢測幾種常用的文件格式，但是這種檢測會損害您的性能，且會限制特定於給定文件格式的功能（例如，自動添加新翻譯）。

### 1.10.3 翻譯類型功能

所有受支持格式的功能：

Format	語 言 能 力 <sup>Page 76, 1</sup>	數 <sup>Page 76, 2</sup>	Descrip- tions <sup>Page 76, 3</sup>	語 境 <sup>Page 76, 4</sup>	位 置 <sup>Page 76, 5</sup>	標 記 <sup>Page 76, 8</sup>	附 加 狀 態 <sup>Page 76, 6</sup>
<i>GNU gettext</i>	bilingual	yes	yes	yes	yes	是 <sup>9</sup>	需要編輯
單 語 <i>get-text</i>	mono	yes	yes	yes	yes	是 <sup>9</sup>	需要編輯
<i>XLIFF</i>	both	yes	yes	yes	yes	是 <sup>10</sup>	需要編輯，已同意
<i>Java</i> 屬性	both	no	yes	no	no	no	
<i>mi18n lang files</i>	mono	no	yes	no	no	no	
<i>GWT</i> 屬性	mono	yes	yes	no	no	no	
<i>Joomla</i> 翻譯	mono	no	yes	no	yes	no	
<i>Qt Linguist .ts</i>	both	yes	yes	no	yes	是 <sup>Page 76, 10</sup>	需要編輯
<code>Android 字符串資源</code>	mono	yes	是 <sup>7</sup>	no	no	是 <sup>Page 76, 10</sup>	

繼續下一頁

表格 1 – 繼續上一頁

Format	語 言 能 力 <sup>Page 76, 1</sup>	F 數 <sup>2</sup>	Descriptions <sup>3</sup>	語境 <sup>4</sup>	位置 <sup>5</sup>	標記 <sup>8</sup>	附加狀態 <sup>6</sup>
蘋果 iOS 字符串	both	no	yes	no	no	no	
PHP 字串	mono	否 <sup>11</sup>	yes	no	no	no	
JSON files	mono	no	no	no	no	no	
JSON i18next files	mono	yes	no	no	no	no	
go-i18n JSON files	mono	yes	no	no	no	no	
gotext JSON files	mono	yes	yes	no	yes	no	
ARB File	mono	yes	yes	no	no	no	
WebExtension JSON	mono	yes	yes	no	no	no	
.XML resource files	mono	no	yes	no	no	是 <sup>Page 76, 10</sup>	
Resource-Dictionary files	mono	no	no	no	no	是 <sup>Page 76, 10</sup>	
CSV 文件	both	no	yes	yes	yes	no	需要編輯
YAML files	mono	no	yes	no	no	no	
Ruby YAML files	mono	yes	yes	no	no	no	
DTD files	mono	no	no	no	no	no	
Flat XML files	mono	no	no	no	no	是 <sup>Page 76, 10</sup>	
Windows RC files	mono	no	yes	no	no	no	
Excel Open XML	mono	no	yes	yes	yes	no	需要編輯
App store 中介資料檔	mono	no	no	no	no	no	
Subtitle files	mono	no	no	no	yes	no	
HTML files	mono	no	no	no	no	no	
Open-Document Format	mono	no	no	no	no	no	
IDML Format	mono	no	no	no	no	no	
INI 翻譯	mono	no	no	no	no	no	
Inno Setup INI translations	mono	no	no	no	no	no	
TermBase eXchange format	bilingual	no	yes	no	no	是 <sup>Page 76, 10</sup>	
文字檔	mono	no	no	no	no	no	
Stringsdict format	mono	yes	yes	no	no	no	

繼續下一頁

表格 1 – 繼續上一頁

Format	語 言 能 力 <sup>Page 76, 1</sup>	數 <sup>2</sup>	Descrip- tions <sup>3</sup>	語境 <sup>4</sup>	位置 <sup>5</sup>	標記 <sup>8</sup>	附加狀態 <sup>6</sup>
<i>Fluent format</i>	mono	no <sup>12</sup>	yes	no	no	no	

## 唯讀字串

在 3.10 版本新加入。

將包含來自翻譯文件的只讀字符串，但不能在 Weblate 中編輯。此功能本功能僅由少量格式支持（：REF: *xliff* 和：ref: *resource*），但可以通過添加一個 “”。

### 1.10.4 GNU gettext

Most widely used format for translating libre software.

通過調整文件頭或鏈接到相應的源文件，可以支持存儲在文件中的語境信息。

雙語 gettext PO 文件通常如下所示：

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "Monday"
msgstr "Pondělí"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "Tuesday"
msgstr "Úterý"

#: weblate/accounts/avatar.py:163
msgctxt "No known user"
msgid "None"
msgstr "Žádný"
```

#### 典型的 Weblate 組件配置

文件掩碼	po/* .po
單語的基底語言檔	<i>Empty</i>
新翻譯的模板	po/messages.pot
檔案格式	<i>Gettext PO file</i>

## 也參考：

devel/gettext, devel/sphinx, Gettext 在 Wikipedia 的條目, PO Files, 更新「configure」檔案中的 *ALL\_LINGUAS* 變數, 自訂 gettext 輸出, 更新 *LINGUAS* 檔案, 生成 *MO* 檔, 更新 *PO* 檔以符合 *POT* (msgmerge)

<sup>1</sup> 請參見雙語和單語格式

<sup>2</sup> 將帶有不同數量的字符串正確本地化時復數是必要的。

<sup>3</sup> Source string descriptions can be used to pass additional info about the string to translate.

<sup>4</sup> 文本用於不同範圍中使用的相同字符串（例如‘Sun’可以用作「Sunday」名稱的縮寫，或用作我們最近的星）。

<sup>5</sup> 源代碼中字符串的位置會幫助熟練的譯者識出字符串如何使用。

<sup>8</sup> 請參見使用標自定義行

<sup>6</sup> Additional states supported by the file format in addition to 「Untranslated」 and 「Translated」.

<sup>9</sup> gettext 類型的釋用作標記。

<sup>10</sup> 對於基於格式所有 XML，標記從非標準的屬性 weblate-flags 中提取。此外，max-length:N 通過 “maxwidth” attribute <http://docs.oasis-open.org/xliff/v1.2/os/xliff-core.html#maxwidth>\_ 如在 XLIFF 標準中定義的，請參見:ref:xliff-flags。

<sup>7</sup> XML comment placed before the <string> element, parsed as a source string description.

<sup>11</sup> 只對於 Laravel 來支持數，在字符串語法中使用來定義它們，請參見 Localization in Laravel。

<sup>12</sup> Plurals are handled in the syntax of the strings and not exposed as plurals in Weblate.

## 單語 gettext

一些項目固定使用 gettext 作單語格式——它們僅在源代碼中編碼 ID，然後將字符串翻譯成所有語言，包括英語。支持此功能，管在將組件導入 Weblate 時必須明確選擇此文件格式。

單語言的 gettext PO 文件通常如下所示：

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-monday"
msgstr "Pondělí"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-tuesday"
msgstr "Úterý"

#: weblate/accounts/avatar.py:163
msgid "none-user"
msgstr "Žádný"
```

基本語言文件將是：

```
#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-monday"
msgstr "Monday"

#: weblate/media/js/bootstrap-datepicker.js:1421
msgid "day-tuesday"
msgstr "Tuesday"

#: weblate/accounts/avatar.py:163
msgid "none-user"
msgstr "None"
```

典型的 Weblate 組件配置	
文件掩碼	po/*.po
單語的基底語言檔	po/en.po
新翻譯的模板	po/messages.pot
檔案格式	Gettext PO 文件（單語）

### 1.10.5 XLIFF

創建基於 XML 的格式來標準化翻譯文件，但最終它是該領域中許多標準之一。

XML Localization Interchange File Format (XLIFF) 通常用作雙語言格式，但 Weblate 也支持其作單語言格式。

Weblate supports XLIFF in several variants:

#### **XLIFF translation file**

Simple XLIFF file where content of the elements is stored as plain text (all XML elements being escaped).

#### **XLIFF with placeables support**

Standard XLIFF supporting placeables and other XML elements.

#### **XLIFF with gettext extensions**

XLIFF enriched by XLIFF 1.2 Representation Guide for Gettext PO to support plurals.

也參考：

XML Localization Interchange File Format (XLIFF) specification, XLIFF 1.2 Representation Guide for Gettext PO

## 翻譯狀態

在 3.3 版本變更: 3.3 版本前的 Weblate 忽略 state 屬性。

文件中的 “state” 屬性在 Weblate 中被部分處理映射到「Needs edit」（需要編輯）的狀態（如果出現目標的話，後面的狀態用於將字符串標記為需要編輯: new ``、``needs-translation、needs-adaptation、needs-l10n）。如果應該省略 state 屬性，只要 <target> 元素存在，那麼字符串被認為需要翻譯。

如果翻譯字符串具有 “approved=「yes」“，那麼它還將被導入 Weblate 作為「Approved」（同意的），任何其它內容被導入作為「Waiting for review」（等待復核，這與 XLIFF 規範匹配）。

當存儲時，Weblate 如無必要不會添加其它屬性：

- state 屬性只在字符串標記為需要編輯的情況下添加。
- approved 屬性只在字符串已經被檢查的情況下添加。
- 在其它情況下不添加屬性，但在它們出現的情況下更新。

這表示當使用 XLIFF 格式時，強烈推薦打開 Weblate 審查過程，從而能看到更改字符串的審核狀態。

相似地在導入這樣的文件時（在上傳表格中），應該選擇 *Processing of strings needing edit* 之下的 *Import as translated*。

### 也參考:

專門的審核者

## XLIFF 中的空白字符和新行符

在 XML 格式中不區分通常類型或量的空白字符。如果想要保留，必須將 `xml:space="preserve"` 標記添加到字符串中。

例如：

```
<trans-unit id="10" approved="yes">
  <source xml:space="preserve">hello</source>
  <target xml:space="preserve">Hello, world!
</target>
</trans-unit>
```

## 指定翻譯標記

還可以通過使用 `weblate-flags` 屬性指定附加的翻譯標記（請參見使用標記自定義行）。Weblate 還理解來自 XLIFF 規範的 `maxwidth` 和 `font` 屬性：

```
<trans-unit id="10" maxwidth="100" size-unit="pixel" font="ubuntu;22:bold">
  <source>Hello %s</source>
</trans-unit>
<trans-unit id="20" maxwidth="100" size-unit="char" weblate-flags="c-format">
  <source>Hello %s</source>
</trans-unit>
```

解析 `font` 屬性用於字體集、尺寸和粗細，上面的示例顯示了全部，儘管只需要字符集。字符集中的任何空白字符被轉換為下劃線，所以 `Source Sans Pro` 變成 `Source_Sans_Pro`，當命名字符集時請記住這些請參見管理字型）。

## String keys

Weblate 通過 `resname` 屬性在它出現的情況下識 XLIFF 文件中的單元，並退回到 `id`（如果出現的話與 `file` 標一起）。

`resname` 屬性被認為是人類友好的單元識別符，對 Weblate 比 `id` 更適於顯示。在整個 XLIFF 文件中 `resname` 具有一致性。這是 Weblate 需要的，且不被 XLIFF 標準覆蓋——它不在這個屬性上放入任何獨特性限制。

用於雙語言 XLIFF 的典型 Weblate 組件配置	
文件掩碼	<code>localizations/*.xliff</code>
單語的基底語言檔	<i>Empty</i>
新翻譯的模板	<code>localizations/en-US.xliff</code>
檔案格式	<i>XLIFF</i> 翻譯文件

用於單語言 XLIFF 的典型 Weblate 組件配置	
文件掩碼	<code>localizations/*.xliff</code>
單語的基底語言檔	<code>localizations/en-US.xliff</code>
新翻譯的模板	<code>localizations/en-US.xliff</code>
檔案格式	<i>XLIFF</i> 翻譯文件

也參考：

**‘xliff 在 wikipedia 上 <<https://en.wikipedia.org/wiki/xliff>>’**，`doc: #`：格式/*xliff*，xliff 中的 `font` 屬性 1.2 `<http: //docs.oasis-open.org / xliff / v1.2 / os / xliff-core.html # font>`，`maxlength` 屬性在 xliff 1.2

### 1.10.6 Java 屬性

用於翻譯的本地 Java 格式。

Java 屬性通常用作單語言翻譯。

Weblate 支持這個格式的 ISO-8859-1、UTF-8 和 UTF-16 變體。它們所有都支持存儲 Unicode 字符，只是編碼不同。在 ISO-8859-1 中，使用了 Unicode 轉義序列（例如 `zkou\u0161ka`），所有其它編碼字符直接或者在 UTF-8 中或者在 UTF-16 中。

備：加載轉義序列也在 UTF-8 模式中工作，因此請小心選擇正確的編碼組，與您應用的需要匹配。

典型的 Weblate 組件配置	
文件掩碼	<code>src/app/Bundle_*.properties</code>
單語的基底語言檔	<code>src/app/Bundle.properties</code>
新翻譯的模板	<i>Empty</i>
檔案格式	<i>Java Properties (ISO-8859-1)</i>

也參考：

Java properties on Wikipedia, Mozilla and Java properties files, *mi18n lang files*, *GWT* 屬性, *updating-target-files*, 格式化 *Java properties* 檔案, 清理翻譯檔

### 1.10.7 mi18n lang files

在 4.7 版本新加入。

File format used for JavaScript localization by `mi18n`. Syntactically it matches *Java* 屬性。

典型的 Weblate 組件配置	
文件掩碼	<code>*.lang</code>
單語的基底語言檔	<code>en-US.lang</code>
新翻譯的模板	<i>Empty</i>
檔案格式	<i>mi18n lang file</i>

也參考：

`mi18n`, Mozilla and Java properties files, *Java* 屬性, `updating-target-files`, 格式化 *Java properties* 檔案, 清理翻譯檔

### 1.10.8 GWT 屬性

用於翻譯的本地 GWT 格式。

GWT 屬性通常用作單語言翻譯。

典型的 Weblate 組件配置	
文件掩碼	<code>src/app/Bundle_*.properties</code>
單語的基底語言檔	<code>src/app/Bundle.properties</code>
新翻譯的模板	<i>Empty</i>
檔案格式	<i>GWT 屬性</i>

也參考：

GWT localization guide, GWT Internationalization Tutorial, Mozilla and Java properties files, `updating-target-files`, 格式化 *Java properties* 檔案, 清理翻譯檔

### 1.10.9 INI 翻譯

在 4.1 版本新加入。

用於翻譯的 INI 文件格式。

INI 翻譯通常用作單語言翻譯。

典型的 Weblate 組件配置	
文件掩碼	<code>language/*.ini</code>
單語的基底語言檔	<code>language/en.ini</code>
新翻譯的模板	<i>Empty</i>
檔案格式	<i>INI File</i>

---

備註： Weblate 只從 INI 文件 的章節中提取鍵。在您的 INI 文件缺乏章節的情況下，會想要替代使用 *Joomla* 翻譯 或 *Java* 屬性。

---

也參考：

INI Files, *Java* 屬性, *Joomla* 翻譯, *Inno Setup INI translations*



### 1.10.10 Inno Setup INI translations

在 4.1 版本新加入。

Inno Setup INI file format for translations.

Inno Setup INI translations are usually used as monolingual translations.

---

備：唯一需要注意的差別是 *INI* 翻譯支持 %n 和 %t 位符用於行和表符。

---

典型的 Weblate 組件配置	
文件掩碼	language/*.isl
單語的基底語言檔	language/en.isl
新翻譯的模板	<i>Empty</i>
檔案格式	<i>Inno Setup INI File</i>

---

備：當前只支持 Unicode 文件 (.isl)，當前不支持 ANSI 變體 (.isl)。

---

也參考：

[INI Files](#), [Joomla 翻譯](#), [INI 翻譯](#)

### 1.10.11 Joomla 翻譯

在 2.12 版本新加入。

用於翻譯的本地 Joomla 格式。

Joomla 翻譯通常用作單語言翻譯。

典型的 Weblate 組件配置	
文件掩碼	language/*/com_foobar.ini
單語的基底語言檔	language/en-GB/com_foobar.ini
新翻譯的模板	<i>Empty</i>
檔案格式	<i>Joomla 語言文件</i>

也參考：

[Mozilla and Java properties files](#), [INI 翻譯](#), [Inno Setup INI translations](#)

### 1.10.12 Qt Linguist .ts

基於 Qt 的應用中使用的翻譯格式。

Qt Linguist 文件既用作雙語翻譯，也用作單語翻譯。

用作雙語言時典型的 Weblate 組件配置	
文件掩碼	i18n/app.*.ts
單語的基底語言檔	<i>Empty</i>
新翻譯的模板	i18n/app.de.ts
檔案格式	<i>Qt Linguist 翻譯文件</i>

用作單語言時典型的 Weblate 組件配置	
文件掩碼	i18n/app.*.ts
單語的基底語言檔	i18n/app.en.ts
新翻譯的模板	i18n/app.en.ts
檔案格式	Qt Linguist 翻譯文件

也參考:

Qt Linguist 手, Qt .ts, 雙語和單語格式

### 1.10.13 安卓字符串資源

用於翻譯應用的安卓特定文件格式。

Android string resources are monolingual, the 單語的基底語言檔 is stored in a different location from the other files `-res/values/strings.xml`.

典型的 Weblate 組件配置	
文件掩碼	res/values-*/strings.xml
單語的基底語言檔	res/values/strings.xml
新翻譯的模板	Empty
檔案格式	安卓字符串資源

也參考:

安卓字符串資源文檔, Android string resources

備: 當前不支持安卓的 `string-array` 架構。了解這個問題，可以將字符串數組分開：

```
<string-array name="several_strings">
  <item>First string</item>
  <item>Second string</item>
</string-array>
```

變:

```
<string-array name="several_strings">
  <item>@string/several_strings_0</item>
  <item>@string/several_strings_1</item>
</string-array>
<string name="several_strings_0">First string</string>
<string name="several_strings_1">Second string</string>
```

指向 `string` 元素的 `string-array` 應存儲在不同文件中，且不翻譯所用。

這個本可以幫助預處理現有的 `strings.xml` 文件和翻譯：<https://gist.github.com/paour/11291062>

提示: To avoid translating some strings, these can be marked as non-translatable. This can be especially useful for string references:

```
<string name="foobar" translatable="false">@string/foo</string>
```

### 1.10.14 蘋果 iOS 字符串

File format typically used for translating Apple iOS applications, but also standardized by PWG 5100.13 and used on NeXTSTEP/OpenSTEP.

Apple iOS strings are usually used as monolingual.

典型的 Weblate 組件配置	
文件掩碼	Resources/*.lproj/Localizable.strings
單語的基底語言檔	Resources/en.lproj/Localizable.strings 或 Resources/Base.lproj/Localizable.strings
新翻譯的模板	<i>Empty</i>
檔案格式	<i>iOS Strings (UTF-8)</i>

也參考:

*Stringsdict format*, Apple 「strings files」 documentation, Message Catalog File Format in PWG 5100.13, Mac OSX strings

### 1.10.15 PHP 字串

PHP 翻譯通常只包含一種語言，因此建議指定 (最常見的) 帶英語字符串的模板文件。

範例檔案:

```
<?php
$LANG['foo'] = 'bar';
$LANG['foo1'] = 'foo bar';
$LANG['foo2'] = 'foo bar baz';
$LANG['foo3'] = 'foo bar baz bag';
```

典型的 Weblate 組件配置	
文件掩碼	lang/*/texts.php
單語的基底語言檔	lang/en/texts.php
新翻譯的模板	lang/en/texts.php
檔案格式	<i>PHP strings</i>

### Laravel PHP 字串

在 4.1 版本變更.

Laravel PHP 本地化文件也支持函數:

```
<?php
return [
    'welcome' => 'Welcome to our application',
    'apples' => 'There is one apple|There are many apples',
];
```

也參考:

PHP, 在 Laravel 中使用在地化

### 1.10.16 JSON files

在 2.0 版本新加入。

在 2.16 版本變更: 自從 Weblate 2.16 和最早 2.2.4 版本的 [translate-toolkit](#), 也支持嵌套結構的 JSON 文件。

在 4.3 版本變更: 即使對於在之前發布版本中中斷的<sup>[1]</sup>雜情<sup>[2]</sup>, JSON 文件的結構也適當保留。

JSON 格式主要用於翻譯 Javascript 中實施的應用。

Weblate 當前支持 JSON 翻譯的幾個變體:

- 簡單的鍵/值文件, 由例如 *vue-i18n* 或 *react-intl* 使用。
- 具有嵌套鍵的文件。
- *JSON i18next files*
- *go-i18n JSON files*
- *gotext JSON files*
- *WebExtension JSON*
- *ARB File*

JSON 翻譯通常是單語言的, 因此推薦指定帶有 (最經常使用的) 英語字符串的翻譯模板文件。

範例檔案:

```
{
  "Hello, world!\n": "Ahoj světe!\n",
  "Orangutan has %d banana.\n": "",
  "Try Weblate at https://demo.weblate.org/!\n": "",
  "Thank you for using Weblate.": ""
}
```

也支持嵌套文件 (要求請參見上面), 這樣的文件看起來像:

```
{
  "weblate": {
    "hello": "Ahoj světe!\n",
    "orangutan": "",
    "try": "",
    "thanks": ""
  }
}
```

**提示:** *JSON file* 和 *JSON nested structure file* 都可以處理相同類型的文件。翻譯時都保留現有的 JSON 架構。

它們之間的唯一區別是使用 Weblate 添加新字符串時。嵌套結構格式會解析新添加的鍵<sup>[1]</sup>將新字符串插入匹配結構。例如, “app.name” 鍵插入<sup>[2]</sup>:

```
{
  "app": {
    "name": "Weblate"
  }
}
```

#### 典型的 Weblate 組件配置

文件掩碼	langs/translation-*.json
單語的基底語言檔	langs/translation-en.json
新翻譯的模板	<i>Empty</i>
檔案格式	<i>JSON nested structure file</i>

**也參考:**

JSON, updating-target-files, 自訂 JSON 輸出, 清理翻譯檔,

**1.10.17 JSON i18next files**

在 2.17 版本變更: 由於 Weblate 2.17 和最少 2.2.5 的“翻譯 - 工具包”, 因此還支持具有復數的 I18Next JSON 文件。

在 4.15.1 版本變更: Support for v4 variant of this format was added.

**提示:** In case you use plurals, it is recommended to use v4 as that aligned plural handling with CLDR. Older versions have different plural rules for some languages which are not correct.

i18next <<https://www.i18next.com/>>\_ 是編寫的框架和 javascript 的國際化框架。Weblate 支持其本地化文件, 其中包含多個功能。

i18next 翻譯是單語, 因此建議使用 (最常見的) 英語字符串指定基本文件。

**備註:** Weblate supports the i18next JSON v3 and v4 variants. Please choose correct file format matching your environment.

The v2 and v1 variants are mostly compatible with v3, with exception of how plurals are handled.

**範例檔案:**

```
{
  "hello": "Hello",
  "apple": "I have an apple",
  "apple_plural": "I have {{count}} apples",
  "apple_negative": "I have no apples"
}
```

典型的 Weblate 組件配置	
文件掩碼	langs/*.json
單語的基底語言檔	langs/en.json
新翻譯的模板	Empty
檔案格式	i18next JSON file v3

**也參考:**

JSON, i18next JSON 格式, updating-target-files, 自訂 JSON 輸出, 清理翻譯檔

**1.10.18 go-i18n JSON files**

在 4.1 版本新加入.

Go-i18n 翻譯是單向的, 因此建議使用 (最常見的) 英語字符串指定基本文件。

**備註:** Weblate 支持 Go-i18N JSON V1 格式, 適用於平面 JSON 格式, 請使用: REF: “JSON”。目前不支持 v2 格式與哈希。

典型的 Weblate 組件配置	
文件掩碼	langs/*.json
單語的基底語言檔	langs/en.json
新翻譯的模板	<i>Empty</i>
檔案格式	<i>go-i18n JSON file</i>

也參考:

: doc: *tt*: 格式/*json*, *go-i18n*, : ref: 更新 - 目標文件, : ref: *addon-Web2.json.Customize*, : ref: *addon-weblate.cleanup.generic*,

### 1.10.19 gotext JSON files

在 4.15.1 版本新加入.

gotext translations are monolingual, so it is recommended to specify a base file with (what is most often the) English strings.

典型的 Weblate 組件配置	
文件掩碼	internal/translations/locales/*/messages.gotext.json
單語的基底語言檔	internal/translations/locales/en-GB/messages.gotext.json
新翻譯的模板	<i>Empty</i>
檔案格式	<i>gotext JSON file</i>

也參考:

JSON, I18n in Go: Managing Translations, updating-target-files, 自訂 *JSON* 輸出, 清理翻譯檔,

### 1.10.20 ARB File

在 4.1 版本新加入.

ARB 翻譯是單語, 因此建議使用 (最常見的) 英語字符串指定基本文件。

典型的 Weblate 組件配置	
文件掩碼	lib/l10n/intl_*.arb
單語的基底語言檔	lib/l10n/intl_en.arb
新翻譯的模板	<i>Empty</i>
檔案格式	<i>ARB file</i>

也參考:

JSON, 應用程式中文包規格, 國際化 Flutter 應用程式, updating-target-files, 自訂 *JSON* 輸出, 清理翻譯檔

### 1.10.21 WebExtension JSON

在 2.16 版本新加入: 自 2.16 以來支持這一點, 且在最小 2.2.4 中使用 '翻譯-Togrukit'。

在翻譯 Mozilla Firefox 或 Google Chromium 的擴展時使用的文件格式。

---

備註: 雖然這種格式稱作 JSON, 但其規範允許包括註釋, 這些註釋不是 JSON 規範的一部分。WebLte 目前不支持註釋。

---

範例檔案:

```
{
  "hello": {
    "message": "Ahoj světe!\n",
    "description": "Description",
    "placeholders": {
      "url": {
        "content": "$1",
        "example": "https://developer.mozilla.org"
      }
    }
  },
  "orangutan": {
    "message": "Orangutan has $count$ bananas",
    "description": "Description",
    "placeholders": {
      "count": {
        "content": "$1",
        "example": "5"
      }
    }
  },
  "try": {
    "message": "",
    "description": "Description"
  },
  "thanks": {
    "message": "",
    "description": "Description"
  }
}
```

#### 典型的 Weblate 組件配置

文件掩碼	_locales/*/messages.json
單語的基底語言檔	_locales/en/messages.json
新翻譯的模板	Empty
檔案格式	WebExtension JSON file

#### 也參考:

: doc: [tt: 格式/json](#), [google chrome.i18n](#), **‘mozilla 擴展 Internationalization <<https://開發人員。Mozilla.org/en-us/docs/mozilla/add-ons/webextensions/internationalization>>’**

### 1.10.22 .XML resource files

在 2.3 版本新加入。

一個.xml 資源 (.resx) 文件使用 Microsoft .NET 應用程序中使用的單格式 XML 文件格式。它是“互用”的.resw, 在使用相同的語法到.resx <<https://lingohub.com/developers/resource-files/resw-resx-localization>>’。

#### 典型的 Weblate 組件配置

文件掩碼	Resources/Language.*.resx
單語的基底語言檔	Resources/Language.resx
新翻譯的模板	Empty
檔案格式	.NET resource file

#### 也參考:

.NET Resource files (.resx), [updating-target-files](#), [清理翻譯檔](#)

### 1.10.23 ResourceDictionary files

在 4.13 版本新加入。

ResourceDictionary is a monolingual XML file format used to package localizable string resources for Windows Presentation Foundation (WPF) applications.

典型的 Weblate 組件配置	
文件掩碼	Languages/*.xaml
單語的基底語言檔	Language/en.xaml
新翻譯的模板	<i>Empty</i>
檔案格式	<i>ResourceDictionary file</i>

也參考：

[Flat XML](#), [Flat XML files](#), [updating-target-files](#), [清理翻譯檔](#)

### 1.10.24 CSV 文件

在 2.4 版本新加入。

CSV 文件可以包含源和翻譯的簡單列表。Weblate 支持以下文件：

- 帶有標頭定義字段 (location, source, target, ID, fuzzy, context, translator\_comments, 'developer\_comments') 的文件。這是推薦的方法，因 它最不容易出錯。挑選 *CSV file* 作 一種文件格式。
- 具有兩個字段的文件——源和翻譯（按此順序），選擇 *Simple CSV file* 作 文件格式。
- 無標頭文件字段順序由 'translate-toolkit': location、source、target、ID、fuzzy、context、translator\_comments、developer\_comments 所定義。選擇 *CSV file* 作 一種文件格式。
- 請記住定義：REF: “組件 - 模板”（參見：REF: ‘Bimono’）。

**提示：** By default, the CSV format does autodetection of file encoding. This can be unreliable in some corner cases and causes performance penalty. Please choose file format variant with encoding to avoid this (for example *CSV file (UTF-8)*).

**警告：** CSV 格式當前會自動檢測 CSV 文件的方言。在某些情 下，自動檢測可能會失敗， 且您會得到不同的結果。對於值中包含 行符的 CSV 文件尤其如此。作 一種解 方法，推薦省略引用的字符。

範例檔案：

Thank you for using Weblate.,Děkujeme za použití Weblate.

雙語 CSV 文件的典型 Weblate 組件配置	
文件掩碼	locale/*.csv
單語的基底語言檔	<i>Empty</i>
新翻譯的模板	locale/en.csv
檔案格式	<i>CSV 文件</i>



單語 CSV 文件的典型 Weblate 組件配置	
文件掩碼	locale/*.csv
單語的基底語言檔	locale/en.csv
新翻譯的模板	locale/en.csv
檔案格式	<i>Simple CSV file</i>

### Multivalue CSV file

在 4.13 版本新加入.

This variant of the CSV files allows storing multiple translations per string.

也參考:

CSV

### 1.10.25 YAML files

在 2.9 版本新加入.

具有字符串鍵和值的普通 yaml 文件。Weblate 還從列表或詞典中提取字符串。

yaml 文件的示例:

```
weblate:
  hello: ""
  orangutan: ""
  try: ""
  thanks: ""
```

典型的 Weblate 組件配置	
文件掩碼	translations/messages/*.yaml
單語的基底語言檔	translations/messages.en.yaml
新翻譯的模板	<i>Empty</i>
檔案格式	<i>YAML file</i>

也參考:

YAML, *Ruby YAML files*

### 1.10.26 Ruby YAML files

在 2.9 版本新加入.

Ruby i18N Yaml 文件用語言作根節點。

示例 Ruby I18N Yaml 文件:

```
cs:
  weblate:
    hello: ""
    orangutan: ""
    try: ""
    thanks: ""
```

典型的 Weblate 組件配置	
文件掩碼	translations/messages.*.yaml
單語的基底語言檔	translations/messages.en.yaml
新翻譯的模板	Empty
檔案格式	Ruby YAML file

也參考:

[YAML](#), [YAML files](#)

### 1.10.27 DTD files

在 2.18 版本新加入.

Example DTD file:

```
<!ENTITY hello "">
<!ENTITY orangutan "">
<!ENTITY try "">
<!ENTITY thanks "">
```

典型的 Weblate 組件配置	
文件掩碼	locale/*.dtd
單語的基底語言檔	locale/en.dtd
新翻譯的模板	Empty
檔案格式	DTD file

也參考:

[Mozilla DTD format](#)

### 1.10.28 Flat XML files

在 3.9 版本新加入.

平面 XML 文件的示例:

```
<?xml version='1.0' encoding='UTF-8'?>
<root>
  <str key="hello_world">Hello World!</str>
  <str key="resource_key">Translated value.</str>
</root>
```

典型的 Weblate 組件配置	
文件掩碼	locale/*.xml
單語的基底語言檔	locale/en.xml
新翻譯的模板	Empty
檔案格式	Flat XML file

也參考:

[Flat XML](#)

### 1.10.29 Windows RC files

在 4.1 版本變更: 對 Windows RC 文件的支持已被重寫。

備註: 對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

Example Windows RC file:

```
LANGUAGE LANG_CZECH, SUBLANG_DEFAULT

STRINGTABLE
BEGIN
    IDS_MSG1                "Hello, world!\n"
    IDS_MSG2                "Orangutan has %d banana.\n"
    IDS_MSG3                "Try Weblate at http://demo.weblate.org/!\n"
    IDS_MSG4                "Thank you for using Weblate."
END
```

典型的 Weblate 組件配置	
文件掩碼	狼/* . 如此
單語的基底語言檔	狼/恩-US . 如此
新翻譯的模板	狼/恩-US . 如此
檔案格式	<i>RC file</i>

也參考:

Windows RC files

### 1.10.30 App store 中介資料檔

在 3.5 版本新加入.

可以翻譯用於在各種應用商店中發布應程序的應程序的元數據。目前以下工具兼容:

- Triple-T gradle-play-publisher
- Fastlane
- F-Droid

元數據由多個 TextFile 組成，Web2 將作單獨的字符串呈現要轉。

典型的 Weblate 組件配置	
文件掩碼	fastlane/android/metadata/*
單語的基底語言檔	fastlane/android/metadata/en-US
新翻譯的模板	fastlane/android/metadata/en-US
檔案格式	<i>App store metadata files</i>

**提示:** 如果您不想翻譯某些字符串 (例如 changelogs)，則標記只讀 (請參: REF: “自定義檢查”)。這可以通過以下方式自動化: REF: “addon-weblate.flags.bulk”。

### 1.10.31 Subtitle files

在 3.7 版本新加入.

Weblate can translate various subtitle files:

- SubRip 字幕檔 (\*.srt)
- MicroDVD 字幕檔 (\*.sub)
- 高級變電站 alpha 字幕文件 (\* .ass)
- 變電站字母字幕文件 (\* .ssa)

典型的 Weblate 組件配置	
文件掩碼	path/*.srt
單語的基底語言檔	path/en.srt
新翻譯的模板	path/en.srt
檔案格式	<i>SubRip subtitle file</i>

也參考:

[Subtitles](#)

### 1.10.32 Excel Open XML

在 3.2 版本新加入.

Excel 打開 XML (.xlsx) 文件可以導入和導出。

在上傳用於翻譯的 XLSX 文件時，請注意，只考慮活動工作表，且必須至少包含一個名“源字符串”的列以及稱“目標”（包含的列）譯文）。此外，應該有一個名“context”的列（其中包含翻譯字符串的上下文路徑）。如果使用 XLSX 下載將翻譯導出到 Excel 工作簿中，則您已經獲得了具有正確文件格式的文件。

### 1.10.33 HTML files

在 4.1 版本新加入.

---

備：對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

---

可翻譯容從 HTML 文件中提取，翻譯提供。

也參考:

[HTML](#)

### 1.10.34 文字檔

在 4.6 版本新加入.

---

備：對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

---

可翻譯容從純文本文件中提取，翻譯提供。每個段落都被翻譯單獨的字符串。

There are three flavors of this format:

- 純文字檔案

- DokuWiki 文本文件
- MediaWiki 文本文件

也參考:

[Simple Text Documents](#)

### 1.10.35 OpenDocument Format

在 4.1 版本新加入。

---

備: 對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

---

可翻譯容從 OpenDocument 文件中提取，翻譯提供。

也參考:

[OpenDocument Format](#)

### 1.10.36 IDML Format

在 4.1 版本新加入。

---

備: 對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

---

從 Adobe Indesign 標記語言文件中提取可翻譯容，翻譯提供。

### 1.10.37 TermBase eXchange format

在 4.5 版本新加入。

TBX 是術語數據交的 XML 格式。

典型的 Weblate 組件配置	
文件掩碼	<code>tbx/*.*tbx</code>
單語的基底語言檔	<i>Empty</i>
新翻譯的模板	<i>Empty</i>
檔案格式	<i>TermBase eXchange file</i>

也參考:

**tbx 在 wikipedia 上** <[https://en.wikipedia.org/wiki/termbase\\_exchange](https://en.wikipedia.org/wiki/termbase_exchange)>\_: doc: *tt*: 格式/*tbx*: ref: 詞表

### 1.10.38 Stringsdict format

在 4.8 版本新加入。

備：對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

XML based format used by Apple which is able to store plural forms of a string.

典型的 Weblate 組件配置	
文件掩碼	Resources/*.lproj/Localizable.stringsdict
單語的基底語言檔	Resources/en.lproj/Localizable.stringsdict or Resources/Base.lproj/Localizable.stringsdict
新翻譯的模板	Empty
檔案格式	Stringsdict file

也參考：

蘋果 iOS 字符串, Stringsdict File Format

### 1.10.39 Fluent format

在 4.8 版本新加入。

備：對此格式的支持目前在 Beta 中，歡迎您的測試反饋。

Fluent is a monolingual text format that focuses on asymmetric localization: a simple string in one language can map to a complex multi-variant translation in another language.

典型的 Weblate 組件配置	
文件掩碼	locales/*/messages.ftl
單語的基底語言檔	locales/en/messages.ftl
新翻譯的模板	Empty
檔案格式	Fluent file

也參考：

Project Fluent website

### 1.10.40 Supporting other formats

支持序列化的大多數格式可以輕鬆支持支持序列化，但他們有（尚未）接收任何測試。在大多數情況下，Weblate 需要一些薄層以隱藏不同 ‘Translate-Toolkit’\_Storage 的行差。

要新格式添加支持，首先要在 “translate-toolkit”\_中其實現支持。

也參考：

Translation Related File Formats

## 1.11 版本控制整合

Weblate currently supports *Git* (with extended support for *GitHub pull requests*, *GitLab merge requests*, *Gitea pull requests*, *Gerrit* and *Subversion*) and *Mercurial* as version control back-ends.

### 1.11.1 訪問存儲庫

Weblate 必須訪問您要使用的 VCS 存儲庫。使用公開的存儲庫，您只需輸入正確的 URL (例如 `https://github.com/WeblateOrg/weblate.git`)，但對於私人存儲庫或推送 URL，設置更複雜且需要驗證。

#### 從 GitHub 的 Weblate 訪問存儲庫

對於 GitHub 的 Web2，有一個專用推送用 GitHub, Bitbucket, Codeberg 和 Gitlab 上 (帶有 Username: Guilabel: “Weblate”，電子郵件 “@ Weblate.org”和，名稱: Guilabel: Bublate Push)。您需要將此用添加協作者，存儲庫提供適當的權限 (只讀對於克隆也可以，按下寫入)。根據服務和組織設置，這會立即發生，或者需要在 Weblate 方面確認。

: 主: Github 上的 “Weblate” 用在五分鐘自動接受邀請函。在其他服務可能需要手動處理，所以請耐心等待。

曾經: 主: “WebBlate” 用已添加，您可以配置: 參考: “組件 - repo”和: ref: *component-push* 使用 *ssh* 協議 (例如 `git@github.com: weblateorg / weblate.git`)。

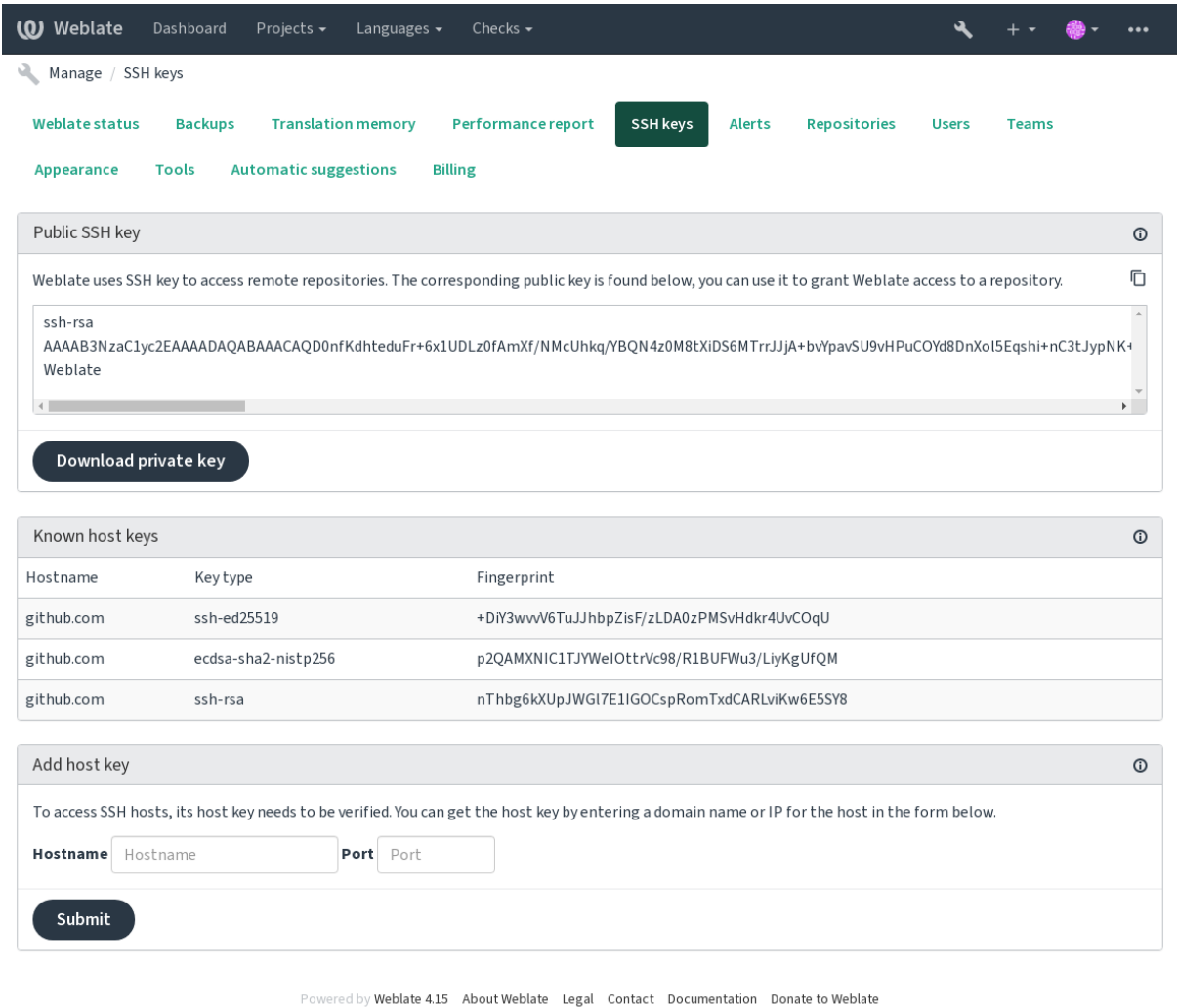
#### SSH 倉庫

訪問私有倉庫的最常用方法是基於 SSH。授權公共 Weblate SSH 密鑰 (請參閱 [Weblate SSH 密鑰](#)) 以這種方式訪問上游倉庫。

**警告:** 在 Github 上，每個密鑰只能用一次，見 [GitHub repositories](#) 和從 GitHub 的 Weblate 訪問存儲庫。

Weblate 還會在首次連接時存儲主機密鑰指紋，且在以後進行更改時將無法連接到主機 (請參閱 [驗證 SSH 主機密鑰](#))。

如需調整，請從 Weblate 管理介面進行：



Weblate SSH 密鑰

Weblate 公鑰對 [About](#) 頁面的所有用 [F](#) 可見。

管理員可以在管理界面登 [F](#) 頁面的連接部分（從 *SSH keys*）生成或顯示 Weblate 當前使用的公共密鑰。

備 [F](#)：相應的私有 SSH 密鑰當前無法使用密碼，因此請確保已受到良好的保護。

提示：對生成的私有 Weblate SSH 密鑰進行備份。

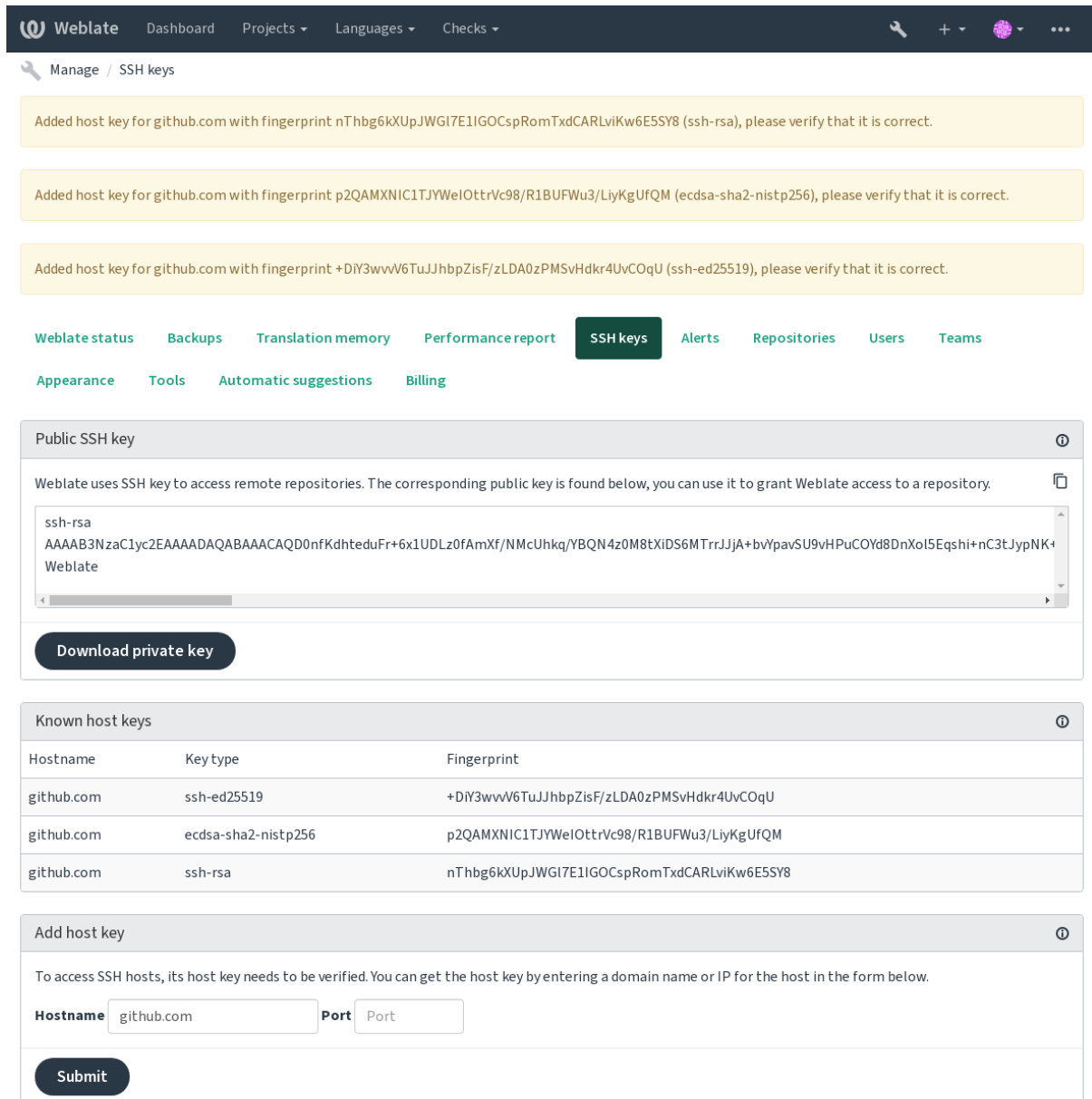


## 驗證 SSH 主機密鑰

Weblate 在第一次訪問時自動存儲 SSH 主機密鑰，記住它們以備將來使用。

如果要在連接到倉庫之前對密鑰指紋進行驗證，請從管理界面的同一部分:guilabel: 『add host key』添加你要訪問的服務器的 SSH 主機密鑰。輸入您要訪問的主機名（例如 `gitlab.com`），然後按 *Submit*。驗證其指紋與您添加的服務器匹配。

帶指紋的添加鍵顯示在確認消息中：



The screenshot displays the 'SSH keys' section of the Weblate management interface. It features a navigation bar at the top with 'SSH keys' highlighted. Below the navigation bar, there are three yellow notification boxes showing added host keys for github.com with their respective fingerprints and key types (ssh-rsa, ecdsa-sha2-nistp256, and ssh-ed25519). The main content area is divided into three sections: 'Public SSH key', 'Known host keys', and 'Add host key'. The 'Public SSH key' section shows a key for Weblate. The 'Known host keys' section is a table listing known host keys for github.com. The 'Add host key' section is a form to add a new host key.

Hostname	Key type	Fingerprint
github.com	ssh-ed25519	+DiY3wvW6TuJJhbpZisF/zLDA0zPMSvHdkr4UvCOqU
github.com	ecdsa-sha2-nistp256	p2QAMXNIC1TJYWeIOttrVc98/R1BUFWu3/LiyKgUfQM
github.com	ssh-rsa	nThbg6kXUpJWG17E1IGOCspRomTxdCARLviKw6E5SY8

## GitHub repositories

可以訪問通過 SSH（參見：REF: ‘SSH-REPOS’），但在需要訪問多個存儲庫的情況下，您將在允許的 SSH 密鑰使用情況下擊中 GitHub 限制（因為每個鍵只能使用一次）。

如果：ref: ‘component-push\_branch’ 未設置，則項目被叉，通過叉子推出的變化。如果設置，則將更改推送到上游存儲庫和所選分支。

對於較小的部署，請使用 HTTPS 身份驗證使用個人訪問令牌和 GitHub 帳戶，參見“創建用於命令行使用”的訪問令牌。

對於更大的設置，通常更好地 Web2RETE 創建專用，將其分配它在 Webleate 中生成的公共 SSH 密鑰（請參見：Ref: “WebLate-Ssh-key”）授予您訪問要轉的所有存儲庫的訪問權限。這種方法也用於管的 Weberate，有專門的：主: “WebLate” 用此。

### 也參考:

從管的 Weberate 訪問存儲庫

## Weblate internal URLs

通過引用其放置在不同組件之間共享一個存儲庫設置，作其他（鏈接）組件的 “Web2: // Project / Component” 這種方式鏈接組件使用 Main（參考）組件的 VCS 存儲庫配置。

**警告：** 除主組件也會除鏈接組件。

如果在創建組件時，WebLlate 會自動調整存儲庫 URL，如果它找到具有匹配的存儲庫設置的組件。您可以在組件配置的最後一步中覆蓋這一點。

Reasons to use this:

- 保存服務器上的磁盤空間，存儲庫僅存儲一次。
- 使更新更快，只更新一個存儲庫。
- 只有單一導出的存儲庫，具有 WebLlate 翻譯（請參見：參考: “Git-Exporter”）。
- Some add-ons can operate on multiple components sharing one repository, for example 匯 Git 提交.

## HTTPS repositories

要訪問受保護的 HTTPS 存儲庫，請在 URL 中包含用戶名和密碼。擔心，Webleate 會在 URL 向用戶顯示 URL 時離此信息（如果允許查看存儲庫 URL）。

例如，添加了具有身份驗證的 GitHub URL 可能看起來像：https: // user: your\_access\_token @ github.com / weblateorg / webbleate.git。

**備：** 如果您的用戶名或密碼包含特殊字符，那必須是 URL 編碼的，例如 “https: //user%40example.com: %24password%23 @ bitbucket.org / ...”。

## Using proxy

如果需要使用代理服務器訪問 HTTP / HTTPS VCS 存儲庫，請配置 VCS 以使用它。

這可以使用 “http\_proxy”，“https\_proxy”和 “”或者通過在 VCS 配置中執行它，例如：

```
git config --global http.proxy http://user:password@proxy.example.com:80
```

**備：** 代理設定需要在使用者使用 Weblate 下完成設定。（參：文件系統權限） 設置“HOME=\$DATA\_DIR/home”（請參：DATA\_DIR），否則 Weblate 將無法透過 Git 執行它。

也參考：

curl manpage, ‘git 配置文件 <<https://git-cm.com/docs/git-config>>’\_

### 1.11.2 Git

**提示：** Weblate needs Git 2.12 or newer.

也參考：

請參：REF: “VCS-REPOS” 用於如何訪問不同類型的存儲庫的信息。

#### Git 使用制推送

這表現得比很像 Git 本身，唯一的區是它總是制推動。這僅在使用單獨的存儲庫進行翻譯的情下。

**警告：** 謹慎使用，因這很容易導致上游存儲庫中的失。

## Customizing Git configuration

Web2Rate 用 “Home = \$ Data\_Dir / Home” 調用所有 VCS 命令（參見：設置：data\_dir），因此需要在 “data\_dir / home / .git”中編輯用配置。

## Git remote helpers

您還可以使用 Git‘Reary Helpers’\_ 另外支持其他版本控制系統，但准備好調試問題，這可能會導致。

此時，在 Github 上的單獨存儲庫中提供了 Bazaar 和 Mercurial 的助手：“Git-remote-Hg”\_ 和 ‘git-remote-bzr’\_。手動下載將其放在搜索路徑中（例如：文件：/bin）。確保安裝了相應的版本控制系統。

安裝這些已安裝後，此類控制器可用於在 WebLate 中指定存儲庫。

使用 Bazaar 來在 Launchpad 上的 gnuhello 專案

```
bzr::lp:gnuhello
```

hello 儲存庫在 selenic.com 上使用 Mercurial

```
hg::http://selenic.com/repo/hello
```

**警告：** 使用 Git 遠程幫助程序的不便例如使用 Mercurial，遠程幫助器有時會在推送更改時創建新的提示。

### 1.11.3 GitHub pull requests

在 2.3 版本新加入。

這在：Ref： “GCS-Git” 中添加了一個薄層： 使用 “github api\_” 以允許將翻譯更改推回拉請求，而不是直接推向存儲庫。

： REF： “VCS-GIT” 將更改直接轉到存儲庫，而參考： REF： “VCS-GitHub” 創建拉出請求。僅訪問 Git 存儲庫不需要後者。

You need to configure API credentials (*GITHUB\_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *GitHub* option when selecting 版本控制系統。

**也參考：**

推送 *Weblate* 的更改, *GITHUB\_USERNAME*, *GITHUB\_TOKEN*, *GITHUB\_CREDENTIALS*

### 1.11.4 GitLab merge requests

在 3.9 版本新加入。

這只需在： REF： “VCS-GIT” 上使用 “GITLAB API” \_ 使用 “GITLAB API” 允許將翻譯更改回合請求，而不是直接推送到存儲庫。

不需要將此用於訪問 Git 存儲庫，普通： REF： “VCS-GIT” 運行相同，唯一的區別是如何處理推送存儲庫。使用： 參考： “VCS-Git” 更改直接推送到存儲庫，而 ref： ‘vcs-gitlab’ 創建合請求。

You need to configure API credentials (*GITLAB\_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *GitLab* option when selecting 版本控制系統。

**也參考：**

推送 *Weblate* 的更改, *GITLAB\_USERNAME*, *GITLAB\_TOKEN*, *GITLAB\_CREDENTIALS*

### 1.11.5 Gitea pull requests

在 4.12 版本新加入。

This just adds a thin layer atop *Git* using the *Gitea API* to allow pushing translation changes as pull requests instead of pushing directly to the repository.

There is no need to use this to access Git repositories, ordinary *Git* works the same, the only difference is how pushing to a repository is handled. With *Git* changes are pushed directly to the repository, while *Gitea pull requests* creates pull requests.

You need to configure API credentials (*GITEA\_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *Gitea* option when selecting 版本控制系統。

**也參考：**

推送 *Weblate* 的更改, *GITEA\_USERNAME*, *GITEA\_TOKEN*, *GITEA\_CREDENTIALS*

### 1.11.6 Pagure merge requests

在 4.3.2 版本新加入。

這只是在 REF 中添加了一個薄層：ref: ‘VCS-GIT’ 使用 ‘PAGURE API’ 允許將翻譯更改合併請求，而不是直接推送到存儲庫。

不需要將此用於訪問 Git 存儲庫，普通：REF: “VCS-GIT” 運行相同，唯一的區別是如何處理推送存儲庫。使用：參考: “VCS-Git” 更改直接推向存儲庫，而參考：參考: “VCS-PAGURE”: “VCS-PAGURE” 創建合併請求。

You need to configure API credentials (*PAGURE\_CREDENTIALS*) in the Weblate settings to make this work. Once configured, you will see a *Pagure* option when selecting 版本控制系統。

**也參考:**

推送 Weblate 的更改, *PAGURE\_USERNAME*, *PAGURE\_TOKEN*, *PAGURE\_CREDENTIALS*

### 1.11.7 Gerrit

在 2.2 版本新加入。

使用 *git-review* 工具加入一層審查層 *Git*，如透過 Gerrit 審查所有推送來的翻譯變更，而不是將翻譯變更直接推送到專案儲存庫。

GERRIT 文檔有關於設置此類存儲庫所需的配置的詳細信息。

### 1.11.8 Mercurial

在 2.1 版本新加入。

Mercurial 是另一個您可以直接在 Weblate 中使用的 VC。

---

**備註:** 它應該與任何 Mercurial 版本合作，但有時對命令行界面有時不兼容，可打破 WebBlate 集成。

---

**也參考:**

請參閱：REF: “VCS-REPOS” 用於如何訪問不同類型的存儲庫的信息。

### 1.11.9 Subversion

在 2.8 版本新加入。

Weblate 使用 ‘git-svn’ 與 ‘subversion’ 存儲庫進行交互。它是一個 perl 本，它允許 git 客戶端使用 subversion，使用功能維護局部存儲庫的完整克隆在本地提交。

---

**備註:** WebLete 嘗試自動檢測 Subversion 存儲庫局 - 它支持具有標準局的分支或存儲庫的直接 URL (分支/, 標記/和中繼/)。有關此內容的更多信息將在 “git-svn 文檔 <<https://git-scm.com/docs/git-svn#documentation/git-svn.txt>>” 中找到。如果您的存儲庫有標準局且您遇到錯誤，請嘗試在存儲庫 URL 中包含分支名稱將分支空。

---

在 2.19 版本變更: 在此之前，僅支持使用標準局的存儲庫。

## Subversion credentials

Weberate 希望您在前面接受證書（如果需要，您的憑據）。它將查看將它們插入：設置：`data_dir`。使用 “SVN”（使用 “\$HOME” 環境變量設置：設置：‘`DATA_DIR`’：

```
# Use DATA_DIR as configured in Weblate settings.py, it is /app/data in the Docker
HOME=${DATA_DIR}/home svn co https://svn.example.com/example
```

也參考：

`DATA_DIR`

### 1.11.10 Local files

### 1.11.11 Git

**提示：** Underneath, this uses *Git*. It requires Git installed and allows you to switch to using Git natively with full history of your translations.

在 3.8 版本新加入。

Weblate 也可以在有遠程 VCS 的情況下運行。通過上傳它們導入初始翻譯。稍後您可以通過文件上載替換單個文件，或直接從 Weberate 添加轉字符串（目前僅適用於單語翻譯）。

在後台，WebBlate 您創建一個 Git 存儲庫，跟踪所有更改。如果您稍後定使用 VCS 存儲翻譯，則您已經在 Weblate 中擁有存儲庫可以基於您的集成。

## 1.12 Weblate 的 REST API

在 2.6 版本新加入：從 Weblate 2.6 開始可以使用 REST API。

API 可以在 `/api/` URL 上訪問，且它基於 Django REST framework。你可以直接使用或參考 *Weblate 客戶端*。

### 1.12.1 身份驗證和通用參數

公共項目 API 不需要身份驗證就可用，但管理有身份驗證的請求導致嚴重的瓶頸（默認每天 100 個請求），所以推薦使用身份驗證。身份驗證使用令牌，這可以在你的簡介中得到。在 Authorization 標頭中使用它：

**ANY /**

對於 API 的普通請求行，標頭、狀態編碼和參數在這也應用於所有端點。

#### 查詢參數

- **format** –響應格式（覆蓋了 `Accept`）。可能的值依賴於 REST 框架設置，默認支持 `json` 和 `api`。後者 API 提供了 web 瀏覽器接口。
- **page** –Returns given page of paginated results (use *next* and *previous* fields in response to automate the navigation).

#### 請求標頭

- **Accept** –相應內容的類型依賴於 `Accept` 標頭
- **Authorization** –optional token to authenticate as `Authorization: Token YOUR-TOKEN`

#### 響應標頭

- **Content-Type** –這依賴於請求的標頭 **Accept**
- **Allow** –對象允許的 HTTP 方法的列表

### Response JSON Object

- **detail** (*string*) –結果的詳細描述（對於 200 OK 以外的 HTTP 狀態編碼）
- **count** (*int*) –對象列表的總項目計數
- **next** (*string*) –對象列表的下一頁 URL
- **previous** (*string*) –對象列表的上一頁 URL
- **results** (*array*) –對象列表的結果
- **url** (*string*) –使用 API 訪問這個資源的 URL
- **web\_url** (*string*) –使用瀏覽器訪問這個資源的 URL

### 狀態編碼

- 200 OK –當請求被正確地處理時
- 201 Created –當成功創建新對象時
- 204 No Content –當一個對象成功刪除時
- 400 Bad Request –當缺少表格參數時
- 403 Forbidden –當訪問被拒絕時
- 429 Too Many Requests –當出現瓶頸時

## Authentication tokens

在 4.10 版本變更: Project scoped tokens were introduced in the 4.10 release.

Each user has his personal access token which can be obtained in the user profile. Newly generated user tokens have the `wlu_` prefix.

It is possible to create project scoped tokens for API access to given project only. These tokens can be identified by the `wlp_` prefix.

## 身份驗證的例子

### 示例請求:

```
GET /api/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
Authorization: Token YOUR-TOKEN
```

### 示例響應:

```
HTTP/1.0 200 OK
Date: Fri, 25 Mar 2016 09:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, HEAD, OPTIONS

{
  "projects": "http://example.com/api/projects/",
```

(繼續下一頁)

(繼續上一頁)

```

"components": "http://example.com/api/components/",
"translations": "http://example.com/api/translations/",
"languages": "http://example.com/api/languages/"
}

```

**CURL 示例:**

```

curl \
  -H "Authorization: Token TOKEN" \
  https://example.com/api/

```

**Passing Parameters Examples**

對於 **POST** 方法，參數可以指定 表格提交 (*application/x-www-form-urlencoded*) 或 **JSON** (*application/json*)。

**Form request example:**

```

POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/x-www-form-urlencoded
Authorization: Token TOKEN

operation=pull

```

**JSON request example:**

```

POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation": "pull"}

```

**CURL 示例:**

```

curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/

```

**CURL JSON example:**

```

curl \
  --data-binary '{"operation": "pull"}' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/

```



## API 頻次限制

這個 API 請求限制了速率；對於匿名用 默認配置限制 每天 100 個請求，對於身份驗證的用 限制 每小時 5000 個請求。

速率限制可以在 `file:settings.py` 中調整；如何配置它的更多細節請參見 ‘Throttling in Django REST framework documentation’ <<https://www.django-rest-framework.org/api-guide/throttling/>>。

In the Docker container this can be configured using `WEBLATE_API_RATELIMIT_ANON` and `WEBLATE_API_RATELIMIT_USER`.

速率限制在後面的標頭中報告：

X-RateLimit-Limit	要執行的對速率限制進行限制的請求
X-RateLimit-Remaining	保持限制的請求
X-RateLimit-Reset	直到速率限制窗口重置時的秒數

在 4.1 版本變更：添加速率限制狀態的標頭。

也參考：

頻次限制, 頻次限制, `WEBLATE_API_RATELIMIT_ANON`, `WEBLATE_API_RATELIMIT_USER`

### 1.12.2 API Entry Point

**GET** `/api/`

API 根入口點。

示例請求：

```
GET /api/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
Authorization: Token YOUR-TOKEN
```

示例響應：

```
HTTP/1.0 200 OK
Date: Fri, 25 Mar 2016 09:46:12 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, HEAD, OPTIONS

{
  "projects": "http://example.com/api/projects/",
  "components": "http://example.com/api/components/",
  "translations": "http://example.com/api/translations/",
  "languages": "http://example.com/api/languages/"
}
```

### 1.12.3 使用者

在 4.0 版本新加入。

**GET /api/users/**

返回用 列表，如果有權限查看管理用的話。如果有，那會只看到自己的具體信息。

也參考：

用 對象屬性被歸檔在 `GET /api/users/(str:username)/`。

**POST /api/users/**

創建新用。

#### Parameters

- **username** (*string*) – 使用者名稱
- **full\_name** (*string*) – User full name
- **email** (*string*) – 用 電子郵件
- **is\_superuser** (*boolean*) – 用 是超級用 嗎？（可選的）
- **is\_active** (*boolean*) – 用 是活動用 嗎？（可選的）
- **is\_bot** (*boolean*) – Is user bot? (optional) (used for project scoped tokens)

**GET /api/users/(str: username) /**

返回用的信息。

#### Parameters

- **username** (*string*) – 用的用 名

#### Response JSON Object

- **username** (*string*) – 用的用 名
- **full\_name** (*string*) – 用的全名
- **email** (*string*) – 用的電子郵件
- **is\_superuser** (*boolean*) – 用 是否是超級用
- **is\_active** (*boolean*) – 用 是否是活動用
- **is\_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date\_joined** (*string*) – 創建用的日期
- **groups** (*array*) – 連接到關聯的組；請參見 `GET /api/groups/(int:id)/`

Example JSON data:

```
{
  "email": "user@example.com",
  "full_name": "Example User",
  "username": "exampleusername",
  "groups": [
    "http://example.com/api/groups/2/",
    "http://example.com/api/groups/3/"
  ],
  "is_superuser": true,
  "is_active": true,
  "is_bot": false,
  "date_joined": "2020-03-29T18:42:42.617681Z",
  "url": "http://example.com/api/users/exampleusername/",
  "statistics_url": "http://example.com/api/users/exampleusername/statistics/"
}
```

**PUT** `/api/users/ (str: username) /`

更改用 參數。

#### Parameters

- **username** (*string*) – 用的用 名

#### Response JSON Object

- **username** (*string*) – 用的用 名
- **full\_name** (*string*) – 用的全名
- **email** (*string*) – 用的電子郵箱
- **is\_superuser** (*boolean*) – 用 是否是超級用 用
- **is\_active** (*boolean*) – 用 是否是活動用 用
- **is\_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date\_joined** (*string*) – 創建用 的日期

**PATCH** `/api/users/ (str: username) /`

更改用 參數。

#### Parameters

- **username** (*string*) – 用的用 名

#### Response JSON Object

- **username** (*string*) – 用的用 名
- **full\_name** (*string*) – 用的全名
- **email** (*string*) – 用的電子郵箱
- **is\_superuser** (*boolean*) – 用 是否是超級用 用
- **is\_active** (*boolean*) – 用 是否是活動用 用
- **is\_bot** (*boolean*) – whether the user is bot (used for project scoped tokens)
- **date\_joined** (*string*) – 創建用 的日期

**DELETE** `/api/users/ (str: username) /`

除所有的用 信息 將用 標記 不活動用 。

#### Parameters

- **username** (*string*) – 用的用 名

**POST** `/api/users/ (str: username) /groups/`

將群組與用 關聯。

#### Parameters

- **username** (*string*) – 用的用 名

#### 表格參數

- **string group\_id** – 唯一的群組 ID

**DELETE** `/api/users/ (str: username) /groups/`

在 4.13.1 版本新加入。

Remove user from a group.

#### Parameters

- **username** (*string*) – 用的用 名

#### 表格參數

- **string group\_id** –唯一的群組 ID

**GET /api/users/ (str: username) /statistics/**  
用 的統計數據列表。

#### Parameters

- **username (string)** –用的用 名

#### Response JSON Object

- **translated (int)** –Number of translations by user
- **suggested (int)** –Number of suggestions by user
- **uploaded (int)** –Number of uploads by user
- **commented (int)** –Number of comments by user
- **languages (int)** –Number of languages user can translate

**GET /api/users/ (str: username) /notifications/**  
用的訂 列表。

#### Parameters

- **username (string)** –用的用 名

**POST /api/users/ (str: username) /notifications/**  
將訂 與用 關聯。

#### Parameters

- **username (string)** –用的用 名

#### Request JSON Object

- **notification (string)** – 通知的名稱
- **scope (int)** –來自可用選擇的通知範圍
- **frequency (int)** –通知的頻率選擇

**GET /api/users/ (str: username) /notifications/**  
**int: subscription\_id/**  
獲得與用 關聯的訂 。

#### Parameters

- **username (string)** –用的用 名
- **subscription\_id (int)** –ID of notification registered

**PUT /api/users/ (str: username) /notifications/**  
**int: subscription\_id/**  
編輯與用 關聯的訂 。

#### Parameters

- **username (string)** –用的用 名
- **subscription\_id (int)** –ID of notification registered

#### Request JSON Object

- **notification (string)** – 通知的名稱
- **scope (int)** –來自可用選擇的通知範圍
- **frequency (int)** –通知的頻率選擇

**PATCH /api/users/ (str: username) /notifications/**  
**int: subscription\_id/**

編輯與用 關聯的訂 。

#### Parameters

- **username** (*string*) – 用的用 名
- **subscription\_id** (*int*) – ID of notification registered

#### Request JSON Object

- **notification** (*string*) – 通知的名稱
- **scope** (*int*) – 來自可用選擇的通知範圍
- **frequency** (*int*) – 通知的頻率選擇

**DELETE** /api/users/(**str**: *username*)/notifications/  
**int**: *subscription\_id*/

除與用 關聯的訂 。

#### Parameters

- **username** (*string*) – 用的用 名
- **subscription\_id** – 通知的名稱
- **subscription\_id** – int

## 1.12.4 群組

在 4.0 版本新加入。

**GET** /api/groups/

返回群組列表，如果有權限看到管理群組的話，如果 有，那 會只看到用 所在的群組。

也參考：

群組對象屬性歸檔在 [GET /api/groups/\(int:id\)/](#)。

**POST** /api/groups/

創建新的群組。

#### Parameters

- **name** (*string*) – 群組名稱
- **project\_selection** (*int*) – 給定選項的項目選擇的群組
- **language\_selection** (*int*) – 給定選項的語言選擇的群組
- **defining\_project** (*str*) – link to the defining project, used for 管理單一專案的存取控制; see [GET /api/projects/\(string:project\)/](#)

**GET** /api/groups/(**int**: *id*)/

返回群組的信息。

#### Parameters

- **id** (*int*) – 群組的 ID

#### Response JSON Object

- **name** (*string*) – 群組的名稱
- **project\_selection** (*int*) – 相應於對象群組的整數
- **language\_selection** (*int*) – 相應於語言群組的整數
- **roles** (*array*) – 相關聯角色的連接；請參見 [GET /api/roles/\(int:id\)/](#)

- **projects** (*array*) –相關聯項目的連接；請參見 `GET /api/projects/(string:project)/`
- **components** (*array*) –相關聯組件的連接；請參見 `GET /api/components/(string:project)/(string:component)/`
- **componentlists** (*array*) –相關聯組件列表的連接；請參見 `GET /api/component-lists/(str:slug)/`
- **defining\_project** (*str*) –link to the defining project, used for 管理單一專案的存取控制; see `GET /api/projects/(string:project)/`

Example JSON data:

```
{
  "name": "Guests",
  "defining_project": null,
  "project_selection": 3,
  "language_selection": 1,
  "url": "http://example.com/api/groups/1/",
  "roles": [
    "http://example.com/api/roles/1/",
    "http://example.com/api/roles/2/"
  ],
  "languages": [
    "http://example.com/api/languages/en/",
    "http://example.com/api/languages/cs/"
  ],
  "projects": [
    "http://example.com/api/projects/demo1/",
    "http://example.com/api/projects/demo/"
  ],
  "componentlist": "http://example.com/api/component-lists/new/",
  "components": [
    "http://example.com/api/components/demo/weblate/"
  ]
}
```

**PUT** `/api/groups/(int: id)/`

更改群組參數。

#### Parameters

- **id** (*int*) –群組的 ID

#### Response JSON Object

- **name** (*string*) –群組的名稱
- **project\_selection** (*int*) –相應於對象群組的整數
- **language\_selection** (*int*) –相應於語言群組的整數

**PATCH** `/api/groups/(int: id)/`

更改群組參數。

#### Parameters

- **id** (*int*) –群組的 ID

#### Response JSON Object

- **name** (*string*) –群組的名稱
- **project\_selection** (*int*) –相應於對象群組的整數
- **language\_selection** (*int*) –相應於語言群組的整數

**DELETE** /api/groups/(int: id)/

Deletes the group.

#### Parameters

- **id**(int) – 群組的 ID

**POST** /api/groups/(int: id)/roles/

將角色與群組關聯。

#### Parameters

- **id**(int) – 群組的 ID

#### 表格參數

- **string** **role\_id** – 唯一的角色 ID

**POST** /api/groups/(int: id)/components/

將組件與群組關聯。

#### Parameters

- **id**(int) – 群組的 ID

#### 表格參數

- **string** **component\_id** – 唯一的組件 ID

**DELETE** /api/groups/(int: id)/components/

**int:** *component\_id*

從群組中刪除組件。

#### Parameters

- **id**(int) – 群組的 ID
- **component\_id**(int) – 唯一的組件 ID

**POST** /api/groups/(int: id)/projects/

將項目與群組關聯。

#### Parameters

- **id**(int) – 群組的 ID

#### 表格參數

- **string** **project\_id** – 唯一的項目 ID

**DELETE** /api/groups/(int: id)/projects/

**int:** *project\_id*

從群組中刪除項目。

#### Parameters

- **id**(int) – 群組的 ID
- **project\_id**(int) – 唯一的項目 ID

**POST** /api/groups/(int: id)/languages/

將語言與群組關聯。

#### Parameters

- **id**(int) – 群組的 ID

#### 表格參數

- **string** **language\_code** – 唯一的語言代碼

**DELETE** /api/groups/(int: id)/languages/

**string:** *language\_code*

從群組中刪除語言。

#### Parameters

- `id(int)` – 群組的 ID
- `language_code(string)` – 唯一的語言代碼

**POST** `/api/groups/(int: id)/componentlists/`

將組件列表與群組關聯。

#### Parameters

- `id(int)` – 群組的 ID

#### 表格參數

- `string component_list_id` – 唯一的組件列表 ID

**DELETE** `/api/groups/(int: id)/componentlists/`

`int: component_list_id`

從群組中刪除組件列表。

#### Parameters

- `id(int)` – 群組的 ID
- `component_list_id(int)` – 唯一的組件列表 ID

## 1.12.5 角色

**GET** `/api/roles/`

返回與用 ID 關聯的所有角色列表。如果用 0 是超級用戶，那 0 返回所有現有角色的列表。

也參考：

角色對象屬性歸檔在 `GET /api/roles/(int:id)/`。

**POST** `/api/roles/`

創建新角色。

#### Parameters

- `name(string)` – 角色名稱
- `permissions(array)` – 權限編碼名稱的列表

**GET** `/api/roles/(int: id) /`

返回角色的信息。

#### Parameters

- `id(int)` – 角色 ID

#### Response JSON Object

- `name(string)` – 角色名稱
- `permissions(array)` – 權限編碼名稱的列表

Example JSON data:

```
{
  "name": "Access repository",
  "permissions": [
    "vcs.access",
    "vcs.view"
  ]
}
```

(繼續下一頁)



(繼續上一頁)

```

    ],
    "url": "http://example.com/api/roles/1/",
  }

```

**PUT** `/api/roles/(int: id) /`

更改角色參數。

**Parameters**

- **id** (*int*) – 角色的 ID

**Response JSON Object**

- **name** (*string*) – 角色名稱
- **permissions** (*array*) – 權限編碼名稱的列表

**PATCH** `/api/roles/(int: id) /`

更改角色參數。

**Parameters**

- **id** (*int*) – 角色的 ID

**Response JSON Object**

- **name** (*string*) – 角色名稱
- **permissions** (*array*) – 權限編碼名稱的列表

**DELETE** `/api/roles/(int: id) /`

Deletes the role.

**Parameters**

- **id** (*int*) – 角色的 ID

## 1.12.6 語言

**GET** `/api/languages/`

返回所有語言的列表。

**也參考:**語言對象屬性存檔在 `GET /api/languages/(string: language) /`。**POST** `/api/languages/`

創建新的語言。

**Parameters**

- **code** (*string*) – 語言名稱
- **name** (*string*) – 語言名稱
- **direction** (*string*) – 文字方向
- **population** (*int*) – 語言使用者數量
- **plural** (*object*) – 語言 F 數形式與數字

**GET** `/api/languages/(string: language) /`

返回語言的信息。

**Parameters**

- **language** (*string*) – 語言碼

**Response JSON Object**

- **code** (*string*) – 語言碼
- **direction** (*string*) – 文字方向
- **plural** (*object*) – 語言 數信息的對象
- **aliases** (*array*) – 語言 名的數組

**Request JSON Object**

- **population** (*int*) – 語言使用者數量

**Example JSON data:**

```
{
  "code": "en",
  "direction": "ltr",
  "name": "English",
  "population": 159034349015,
  "plural": {
    "id": 75,
    "source": 0,
    "number": 2,
    "formula": "n != 1",
    "type": 1
  },
  "aliases": [
    "english",
    "en_en",
    "base",
    "source",
    "eng"
  ],
  "url": "http://example.com/api/languages/en/",
  "web_url": "http://example.com/languages/en/",
  "statistics_url": "http://example.com/api/languages/en/statistics/"
}
```

**PUT** /api/languages/ (*string: language*) /

更改語言參數。

**Parameters**

- **language** (*string*) – 語言的編碼

**Request JSON Object**

- **name** (*string*) – 語言名稱
- **direction** (*string*) – 文字方向
- **population** (*int*) – 語言使用者數量
- **plural** (*object*) – Language plural details

**PATCH** /api/languages/ (*string: language*) /

更改語言參數。

**Parameters**

- **language** (*string*) – 語言的編碼

**Request JSON Object**

- **name** (*string*) – 語言名稱
- **direction** (*string*) – 文字方向

- **population** (*int*) – 語言使用者數量
- **plural** (*object*) – Language plural details

**DELETE** /api/languages/ (**string:** *language*) /

Deletes the language.

#### Parameters

- **language** (*string*) – 語言的編碼

**GET** /api/languages/ (**string:** *language*) /**statistics/**

返回語言的統計數據。

#### Parameters

- **language** (*string*) – 語言碼

#### Response JSON Object

- **total** (*int*) – 字符串的總數
- **total\_words** (*int*) – 詞的總數
- **last\_change** (*timestamp*) – 語言的上一次更改
- **recent\_changes** (*int*) – 更改的總數
- **translated** (*int*) – 已翻譯的字符串數量
- **translated\_percent** (*float*) – 已翻譯字符串的百分比
- **translated\_words** (*int*) – 已翻譯詞的數量
- **translated\_words\_percent** (*int*) – 已翻譯詞的百分比
- **translated\_chars** (*int*) – 已翻譯字符的數量
- **translated\_chars\_percent** (*int*) – 已翻譯字符的百分比
- **total\_chars** (*int*) – 總字符的數量
- **fuzzy** (*int*) – number of fuzzy (marked for edit) strings
- **fuzzy\_percent** (*int*) – 模糊字符串（標記  需要編輯）的百分比
- **failing** (*int*) – 失敗字符串的數量
- **failing** – 失敗字符串的百分比

## 1.12.7 專案

**GET** /api/projects/

返回所有項目的列表。

也參考：

項目對象的屬性存檔在 [GET /api/projects/ \(\*\*string:\*\* \*project\*\) /](#)。

**POST** /api/projects/

在 3.9 版本新加入。

創建新項目。

#### Parameters

- **name** (*string*) – 專案名稱
- **slug** (*string*) – 項目標識串
- **web** (*string*) – 專案網站

**GET** /api/projects/(string: project) /

返回項目的信息。

#### Parameters

- **project** (string) –專案 URL slug

#### Response JSON Object

- **name** (string) –專案名稱
- **slug** (string) –項目標識串
- **web** (string) –項目網站
- **components\_list\_url** (string) –組件列表的 URL; 請參見 [GET /api/projects/\(string:project\)/components/](#)
- **repository\_url** (string) –倉庫狀態的 URL; 請參見 [GET /api/projects/\(string:project\)/repository/](#)
- **changes\_list\_url** (string) –更改列表的 URL; 請參見 [GET /api/projects/\(string:project\)/repository/](#)
- **translation\_review** (boolean) –[用檢](#)
- **source\_review** (boolean) –[用來源檢](#)
- **set\_language\_team** (boolean) –設定「*Language-Team*」檔案標頭
- **enable\_hooks** (boolean) –[用](#)[勾](#)
- **instructions** (string) –翻譯指示
- **language\_aliases** (string) –語言[名](#)

Example JSON data:

```
{
  "name": "Hello",
  "slug": "hello",
  "url": "http://example.com/api/projects/hello/",
  "web": "https://weblate.org/",
  "web_url": "http://example.com/projects/hello/"
}
```

**PATCH** /api/projects/(string: project) /

在 4.3 版本新加入。

透過 [PATCH](#) 請求來編輯一個專案。

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug

**PUT** /api/projects/(string: project) /

在 4.3 版本新加入。

透過 [PUT](#) 請求來編輯一個專案。

#### Parameters

- **project** (string) –專案 URL slug

**DELETE** /api/projects/(string: project) /

在 3.9 版本新加入。

Deletes a project.

**Parameters**

- **project** (*string*) –專案 URL slug

**GET** /api/projects/(string: project)/changes/

返回項目更改的列表。這本質上是仔細檢查的項目 `GET /api/changes/` 接收相同的參數。

**Parameters**

- **project** (*string*) –專案 URL slug

**Response JSON Object**

- **results** (*array*) –組件對象的矩陣；請參見 `GET /api/changes/(int:id)/`

**GET** /api/projects/(string: project)/repository/

返回版本控制系統（VCS）倉庫狀態的信息。這個端點只包含項目所有倉庫的整體概況。為了得到更多細節，請使用 `GET /api/components/(string:project)/(string:component)/repository/`。

**Parameters**

- **project** (*string*) –專案 URL slug

**Response JSON Object**

- **needs\_commit** (*boolean*) –是否有待定的更改要提交
- **needs\_merge** (*boolean*) –是否有上游更改要合併
- **needs\_push** (*boolean*) –是否有本地更改要推送

Example JSON data:

```
{
  "needs_commit": true,
  "needs_merge": false,
  "needs_push": true
}
```

**POST** /api/projects/(string: project)/repository/

在版本控制系統（VCS）倉庫上執行給定的操作。

**Parameters**

- **project** (*string*) –專案 URL slug

**Request JSON Object**

- **operation** (*string*) –要執行的操作: one of push, pull, commit, reset, cleanup, file-sync

**Response JSON Object**

- **result** (*boolean*) –操作的結果

CURL 示例:

```
curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/repository/
```

JSON request example:

```
POST /api/projects/hello/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation": "pull"}
```

**JSON response example:**

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"result": true}
```

**GET /api/projects/(string: project)/components/**

返回給定項目的翻譯組件列表。

**Parameters**

- **project** (*string*) –專案 URL slug

**Response JSON Object**

- **results** (*array*) –組件對象的矩陣；請參見 *GET /api/components/(string:project)/(string:component)/*

**POST /api/projects/(string: project)/components/**

在 3.9 版本新加入。

在 4.3 版本變更: *zipfile* 和 *docfile* 參數現在可被無 VCS 的組件所接受，見 *Local files*。

在 4.6 版本變更: 克隆的存儲庫現在使用: REF: “[☐部 URL](#)”，現在自動在項目中自動共享。使用 “isable\_autoshare” 關閉。

在給定的項目中新建翻譯組件。

---

**提示:** 使用: REF: 從單個 VCS 存儲庫創建多個組件時的 “[☐部 URL](#)”。

---



---

**備☐:** 多數組件的新建發生在後台。檢查新建組件的 *task\_url* 屬性，☐按照那☐的步驟進行。

---

**Parameters**

- **project** (*string*) –專案 URL slug

**表格參數**

- **file zipfile** –上傳到 Weblate 用於翻譯初始化的 ZIP 文件
- **file docfile** –要翻譯的文件
- **boolean disable\_autoshare** –禁用自動存儲庫共享: 參考: “[☐部 URL](#)”。

**Request JSON Object**

- **object** -Component parameters, see `GET /api/components/(string:project)/(string:component)/`

### Response JSON Object

- **result** (object) - 新建組件對象; 請參見 `GET /api/components/(string:project)/(string:component)/`

使用 “ZipFile“和 “””) 無法使用 JSON。數據必須上傳 F: MimeType: *multipart/form-data*。

### CURL form request example:

```
curl \
  --form docfile=@strings.html \
  --form name=Weblate \
  --form slug=weblate \
  --form file_format=html \
  --form new_lang=add \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

### CURL JSON request example:

```
curl \
  --data-binary '{
    "branch": "main",
    "file_format": "po",
    "filemask": "po/*.po",
    "name": "Weblate",
    "slug": "weblate",
    "repo": "https://github.com/WeblateOrg/hello.git",
    "template": "",
    "new_base": "po/hello.pot",
    "vcs": "git"
  }' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

### JSON request to create a new component from Git:

```
POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "name": "Weblate",
  "slug": "weblate",
  "repo": "https://github.com/WeblateOrg/hello.git",
  "template": "",
  "new_base": "po/hello.pot",
  "vcs": "git"
}
```

### JSON request to create a new component from another one:

```

POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
  "file_format": "po",
  "filemask": "po/*.po",
  "name": "Weblate",
  "slug": "weblate",
  "repo": "weblate://weblate/hello",
  "template": "",
  "new_base": "po/hello.pot",
  "vcs": "git"
}

```

**JSON response example:**

```

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "Weblate",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",
  "vcs": "git",
  "web_url": "http://example.com/projects/hello/weblate/"
}

```

**GET** /api/projects/(string: *project*)/languages/



對項目 的所有語言返回編頁的統計數據。

在 3.8 版本新加入。

#### Parameters

- **project** (*string*) –專案 URL slug

#### Response JSON Object

- **results** (*array*) –翻譯統計數據對象的矩陣
- **language** (*string*) –語言名稱
- **code** (*string*) –語言代碼
- **total** (*int*) –字符串的總數
- **translated** (*int*) –已翻譯的字符串數量
- **translated\_percent** (*float*) –已翻譯字符串的百分比
- **total\_words** (*int*) –詞的總數
- **translated\_words** (*int*) –已翻譯詞的數量
- **words\_percent** (*float*) –已翻譯詞的百分比

**GET** /api/projects/(string: project)/statistics/  
返回項目的統計數據。

在 3.8 版本新加入。

#### Parameters

- **project** (*string*) –專案 URL slug

#### Response JSON Object

- **total** (*int*) –字符串的總數
- **translated** (*int*) –已翻譯的字符串數量
- **translated\_percent** (*float*) –已翻譯字符串的百分比
- **total\_words** (*int*) –詞的總數
- **translated\_words** (*int*) –已翻譯詞的數量
- **words\_percent** (*float*) –已翻譯詞的百分比

## 1.12.8 組件

---

**提示:** Use `POST /api/projects/(string:project)/components/` to create new components.

---

**GET** /api/components/

返回翻譯組件的列表。

#### 也參考:

組件對象屬性存檔在 `GET /api/components/(string:project)/(string:component)/`。

**GET** /api/components/(string: project) /  
string: component/

返回翻譯組件的信息。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **project** (*object*) – 翻譯項目；請參見 `GET /api/projects/(string:project)/`
- **name** (*string*) –組件名
- **slug** (*string*) –組件標識串
- **vcs** (*string*) –版本控制系統
- **repo** (*string*) –源代碼儲存庫
- **git\_export** (*string*) –已匯出儲存庫 URL
- **branch** (*string*) –儲存庫分支
- **push\_branch** (*string*) –推送分支
- **filemask** (*string*) –文件掩碼
- **template** (*string*) –單語的基底語言檔
- **edit\_template** (*string*) –編輯基底檔
- **intermediate** (*string*) –中間語言檔案
- **new\_base** (*string*) –新翻譯的模板
- **file\_format** (*string*) –檔案格式
- **license** (*string*) –翻譯授權條款
- **agreement** (*string*) –貢獻者協議書
- **new\_lang** (*string*) –加入新翻譯
- **language\_code\_style** (*string*) –語言代碼類型
- **source\_language** (*object*) –源語言對象；請參見 `GET /api/languages/(string:language)/`
- **push** (*string*) –儲存庫推送 URL
- **check\_flags** (*string*) –翻譯旗標
- **priority** (*string*) –優先度
- **enforced\_checks** (*string*) –制查核
- **restricted** (*string*) –受限制的訪問
- **repoweb** (*string*) –儲存庫 瀏覽器
- **report\_source\_bugs** (*string*) –來源字串臭蟲回報位址
- **merge\_style** (*string*) –合 類型
- **commit\_message** (*string*) –Commit, add, delete, merge, add-on, and merge request messages
- **add\_message** (*string*) –Commit, add, delete, merge, add-on, and merge request messages
- **delete\_message** (*string*) –Commit, add, delete, merge, add-on, and merge request messages
- **merge\_message** (*string*) –Commit, add, delete, merge, add-on, and merge request messages

- **addon\_message** (*string*) - Commit, add, delete, merge, add-on, and merge request messages
- **pull\_message** (*string*) - Commit, add, delete, merge, add-on, and merge request messages
- **allow\_translation\_propagation** (*string*) - 允許翻譯再用
- **enable\_suggestions** (*string*) - F 用建議
- **suggestion\_voting** (*string*) - 建議投票
- **suggestion\_autoaccept** (*string*) - 自動接受建議
- **push\_on\_commit** (*string*) - 提交時 F 推送
- **commit\_pending\_age** (*string*) - 更動後提交的經過時間
- **auto\_lock\_error** (*string*) - 有錯誤時鎖定
- **language\_regex** (*string*) - 語言篩選
- **variant\_regex** (*string*) - 變體的正則表達式
- **repository\_url** (*string*) - 倉庫狀態的 URL; 請參見 `GET /api/components/(string:project)/(string:component)/repository/`
- **translations\_url** (*string*) - 翻譯列表的 URL; 請參見 `GET /api/components/(string:project)/(string:component)/translations/`
- **lock\_url** (*string*) - 鎖定狀態的 URL; 請參見 `GET /api/components/(string:project)/(string:component)/lock/`
- **changes\_list\_url** (*string*) - 更改的列表的 URL; 請參見 `GET /api/components/(string:project)/(string:component)/changes/`
- **task\_url** (*string*) - 後台任務 URL (如果有的話); 請參見 `GET /api/tasks/(str:uuid)/`

Example JSON data:

```
{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "Weblate",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "source_language": {
    "code": "en",
```

(繼續下一頁)

(繼續上一頁)

```

    "direction": "ltr",
    "population": 159034349015,
    "name": "English",
    "url": "http://example.com/api/languages/en/",
    "web_url": "http://example.com/languages/en/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",
  "vcs": "git",
  "web_url": "http://example.com/projects/hello/weblate/"
}

```

**PATCH** /api/components/(string: project) /  
string: component/

透過 **PATCH** 請求來編輯一個組件。

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug
- **source\_language** (string) –項目源語言代碼 (可選)

#### Request JSON Object

- **name** (string) –name of component
- **slug** (string) –slug of component
- **repo** (string) –VCS repository URL

#### CURL 示例:

```

curl \
  --data-binary '{"name": "new name"}' \
  -H "Content-Type: application/json" \
  -H "Authorization: Token TOKEN" \
  PATCH http://example.com/api/projects/hello/components/

```

#### JSON request example:

```

PATCH /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{
  "name": "new name"
}

```

#### JSON response example:

```

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en

```

(繼續下一頁)

(繼續上一頁)

```

Allow: GET, POST, HEAD, OPTIONS

{
  "branch": "main",
  "file_format": "po",
  "filemask": "po/*.po",
  "git_export": "",
  "license": "",
  "license_url": "",
  "name": "new name",
  "slug": "weblate",
  "project": {
    "name": "Hello",
    "slug": "hello",
    "source_language": {
      "code": "en",
      "direction": "ltr",
      "population": 159034349015,
      "name": "English",
      "url": "http://example.com/api/languages/en/",
      "web_url": "http://example.com/languages/en/"
    },
    "url": "http://example.com/api/projects/hello/",
    "web": "https://weblate.org/",
    "web_url": "http://example.com/projects/hello/"
  },
  "repo": "file:///home/nijel/work/weblate-hello",
  "template": "",
  "new_base": "",
  "url": "http://example.com/api/components/hello/weblate/",
  "vcs": "git",
  "web_url": "http://example.com/projects/hello/weblate/"
}

```

**PUT** /api/components/(string: project) /  
 string: component/

透過 **PUT** 請求來編輯一個組件。

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug

#### Request JSON Object

- **branch** (string) –VCS repository branch
- **file\_format** (string) –翻譯的文件格式
- **filemask** (string) –倉庫中翻譯的文件掩碼
- **name** (string) –name of component
- **slug** (string) –slug of component
- **repo** (string) –VCS repository URL
- **template** (string) –但語言翻譯的譯文模板文件
- **new\_base** (string) –用於添加新翻譯的譯文模板文件
- **vcs** (string) –version control system

**DELETE** /api/components/(string: project) /  
 string: component/

在 3.9 版本新加入.

Deletes a component.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

**GET** /api/components/ (**string:** *project*) /  
**string:** *component/changes/*

返回組件更改的列表。這本質上是仔細檢查的組件:[http:~get:api/changes/](http://get:api/changes/)接相同參數。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **results** (*array*) –組件對象的矩陣; 請參見[GET /api/changes/ \(int:id\)/](#)

**GET** /api/components/ (**string:** *project*) /  
**string:** *component/file/*

在 4.9 版本新加入.

Downloads all available translations associated with the component as an archive file using the requested format.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### 查詢參數

- **format** (*string*) –The archive format to use; If not specified, defaults to zip; Supported formats: zip

**GET** /api/components/ (**string:** *project*) /  
**string:** *component/screenshots/*

返回組件屏幕截圖的列表。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **results** (*array*) –組件屏幕截圖的矩陣; 請參見[GET /api/screenshots/ \(int:id\)/](#)

**GET** /api/components/ (**string:** *project*) /  
**string:** *component/lock/*

返回組件鎖定狀態。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **locked** (*boolean*) –組件是否因更新而鎖定

Example JSON data:

```
{
  "locked": false
}
```

**POST** `/api/components/(string: project) /`  
**string:** `component/lock/`

設置組件鎖定狀態。

響應時間與 `http:GET:/api/components/(string:project)/(string:component)/lock/` 相同。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Request JSON Object

- **lock** –是否鎖定的布爾值。

CURL 示例:

```
curl \
  -d lock=true \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

JSON request example:

```
POST /api/components/hello/weblate/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"lock": true}
```

JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"locked": true}
```

**GET** `/api/components/(string: project) /`  
**string:** `component/repository/`

返回版本控制系統 (VCS) 倉庫狀態的信息。

響應與 `GET /api/projects/(string:project)/repository/` 的相同。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **needs\_commit** (*boolean*) – 是否有待定的更改要提交
- **needs\_merge** (*boolean*) – 是否有上游更改要合併
- **needs\_push** (*boolean*) – 是否有本地更改要推送
- **remote\_commit** (*string*) – Remote commit information
- **status** (*string*) – 由版本控制系統 (VCS) 報告的 VCS 狀態
- **merge\_failure** – 描述合併失敗的文本, 有的話空

**POST** `/api/components/(string: project) /`  
**string:** `component/repository/`

在版本控制系統 (VCS) 倉庫執行給定的操作。

文檔請參見 `POST /api/projects/(string:project)/repository/`。

#### Parameters

- **project** (*string*) – 專案 URL slug
- **component** (*string*) – 組件 URL slug

#### Request JSON Object

- **operation** (*string*) – 執行的操作: push, pull, commit, reset, “cleanup” 之一

#### Response JSON Object

- **result** (*boolean*) – 操作的結果

#### CURL 示例:

```
curl \
  -d operation=pull \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/components/hello/weblate/repository/
```

#### JSON request example:

```
POST /api/components/hello/weblate/repository/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"operation": "pull"}
```

#### JSON response example:

```
HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{"result": true}
```

**GET** `/api/components/(string: project) /`  
**string:** `component/monolingual_base/`

單語言翻譯下載譯文模板文件。



**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

**GET** /api/components/ (*string: project*) /  
*string: component/new\_template/*

新的翻譯下載模板文件。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

**GET** /api/components/ (*string: project*) /  
*string: component/translations/*

返回給定組件中翻譯對象的列表。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

**Response JSON Object**

- **results** (*array*) –翻譯對象的矩陣；請參見 *GET /api/translations/ (string:project) / (string:component) / (string:language) /*

**POST** /api/components/ (*string: project*) /  
*string: component/translations/*

在給定組件中新建新的翻譯。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

**Request JSON Object**

- **language\_code** (*string*) –翻譯語言代碼；請參見 *GET /api/languages/ (string:language) /*

**Response JSON Object**

- **result** (*object*) –新建的新翻譯對象

**CURL 示例：**

```
curl \
  -d language_code=cs \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/projects/hello/components/
```

**JSON request example:**

```
POST /api/projects/hello/components/ HTTP/1.1
Host: example.com
Accept: application/json
Content-Type: application/json
Authorization: Token TOKEN
Content-Length: 20

{"language_code": "cs"}
```

**JSON response example:**

```

HTTP/1.0 200 OK
Date: Tue, 12 Apr 2016 09:32:50 GMT
Server: WSGIServer/0.1 Python/2.7.11+
Vary: Accept, Accept-Language, Cookie
X-Frame-Options: SAMEORIGIN
Content-Type: application/json
Content-Language: en
Allow: GET, POST, HEAD, OPTIONS

{
  "failing_checks": 0,
  "failing_checks_percent": 0,
  "failing_checks_words": 0,
  "filename": "po/cs.po",
  "fuzzy": 0,
  "fuzzy_percent": 0.0,
  "fuzzy_words": 0,
  "have_comment": 0,
  "have_suggestion": 0,
  "is_template": false,
  "is_source": false,
  "language": {
    "code": "cs",
    "direction": "ltr",
    "population": 1303174280
    "name": "Czech",
    "url": "http://example.com/api/languages/cs/",
    "web_url": "http://example.com/languages/cs/"
  },
  "language_code": "cs",
  "id": 125,
  "last_author": null,
  "last_change": null,
  "share_url": "http://example.com/engage/hello/cs/",
  "total": 4,
  "total_words": 15,
  "translate_url": "http://example.com/translate/hello/weblate/cs/",
  "translated": 0,
  "translated_percent": 0.0,
  "translated_words": 0,
  "url": "http://example.com/api/translations/hello/weblate/cs/",
  "web_url": "http://example.com/projects/hello/weblate/cs/"
}

```

**GET** `/api/components/(string:project)/string:component/statistics/`

對組件 F 所有的翻譯返回分頁的統計數據。

在 2.7 版本新加入。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **results** (*array*) – 翻譯統計數據對象的矩陣；請參見 `GET /api/translations/(string:project)/(string:component)/(string:language)/statistics/`

**GET** `/api/components/(string:project)/string:component/links/`

Returns projects linked with a component.

在 4.5 版本新加入.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### Response JSON Object

- **projects** (*array*) –associated projects; see `GET /api/projects/(string:project)/`

**POST /api/components/(string: project) /**  
**string: component/links/**

Associate project with a component.

在 4.5 版本新加入.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug

#### 表格參數

- **string project\_slug** –項目標識串

**DELETE /api/components/(string: project) /**  
**string: component/links/string: project\_slug/**

Remove association of a project with a component.

在 4.5 版本新加入.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **project\_slug** (*string*) –Slug of the project to remove

## 1.12.9 翻譯

**GET /api/translations/**

返回翻譯的列表。

也參考:

翻譯對象屬性存檔在 `GET /api/translations/(string:project)/(string:component)/(string:language)/`。

**GET /api/translations/(string: project) /**  
**string: component/string: language/**

返回翻譯的信息。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

#### Response JSON Object

- **component** (*object*) – 組件對象；請參見 `GET /api/components/(string:project)/(string:component)/`
- **failing\_checks** (*int*) – number of strings failing checks
- **failing\_checks\_percent** (*float*) – percentage of strings failing checks
- **failing\_checks\_words** (*int*) – number of words with failing checks
- **filename** (*string*) – 翻譯文件名
- **fuzzy** (*int*) – number of fuzzy (marked for edit) strings
- **fuzzy\_percent** (*float*) – 模糊字符串（標記需要編輯）的百分比
- **fuzzy\_words** (*int*) – 模糊（標記編輯）字符串中的單詞數
- **have\_comment** (*int*) – 帶有釋的字符串數量
- **have\_suggestion** (*int*) – 帶有建議的字符串數量
- **is\_template** (*boolean*) – whether the translation has a monolingual base
- **language** (*object*) – 源語言對象；請參見 `GET /api/languages/(string:language)/`
- **language\_code** (*string*) – 倉庫中使用的語言代碼；這可以不同於語言對象中的語言代碼
- **last\_author** (*string*) – 最後一位作者的姓名
- **last\_change** (*timestamp*) – last change timestamp
- **revision** (*string*) – 文件的修訂哈希值
- **share\_url** (*string*) – 用於分享導向約定頁面的 URL
- **total** (*int*) – 字符串的總數
- **total\_words** (*int*) – 詞的總數
- **translate\_url** (*string*) – URL for translating
- **translated** (*int*) – 已翻譯的字符串數量
- **translated\_percent** (*float*) – 已翻譯字符串的百分比
- **translated\_words** (*int*) – 已翻譯詞的數量
- **repository\_url** (*string*) – 倉庫狀態的 URL；請參見 `GET /api/translations/(string:project)/(string:component)/(string:language)/repository/`
- **file\_url** (*string*) – 文件對象的 URL；請參見 `GET /api/translations/(string:project)/(string:component)/(string:language)/file/`
- **changes\_list\_url** (*string*) – 更改的列表的 URL；請參見 `GET /api/translations/(string:project)/(string:component)/(string:language)/changes/`
- **units\_list\_url** (*string*) – 字符串列表的 URL；請參見 `GET /api/translations/(string:project)/(string:component)/(string:language)/units/`

Example JSON data:

```
{
  "component": {
    "branch": "main",
    "file_format": "po",
```

(繼續下一頁)

(繼續上一頁)

```

    "filemask": "po/*.po",
    "git_export": "",
    "license": "",
    "license_url": "",
    "name": "Weblate",
    "new_base": "",
    "project": {
      "name": "Hello",
      "slug": "hello",
      "source_language": {
        "code": "en",
        "direction": "ltr",
        "population": 159034349015,
        "name": "English",
        "url": "http://example.com/api/languages/en/",
        "web_url": "http://example.com/languages/en/"
      },
      "url": "http://example.com/api/projects/hello/",
      "web": "https://weblate.org/",
      "web_url": "http://example.com/projects/hello/"
    },
    "repo": "file:///home/nijel/work/weblate-hello",
    "slug": "weblate",
    "template": "",
    "url": "http://example.com/api/components/hello/weblate/",
    "vcs": "git",
    "web_url": "http://example.com/projects/hello/weblate/"
  },
  "failing_checks": 3,
  "failing_checks_percent": 75.0,
  "failing_checks_words": 11,
  "filename": "po/cs.po",
  "fuzzy": 0,
  "fuzzy_percent": 0.0,
  "fuzzy_words": 0,
  "have_comment": 0,
  "have_suggestion": 0,
  "is_template": false,
  "language": {
    "code": "cs",
    "direction": "ltr",
    "population": 1303174280,
    "name": "Czech",
    "url": "http://example.com/api/languages/cs/",
    "web_url": "http://example.com/languages/cs/"
  },
  "language_code": "cs",
  "last_author": "Weblate Admin",
  "last_change": "2016-03-07T10:20:05.499",
  "revision": "7ddfafe6daaf57fc8654cc852ea6be212b015792",
  "share_url": "http://example.com/engage/hello/cs/",
  "total": 4,
  "total_words": 15,
  "translate_url": "http://example.com/translate/hello/weblate/cs/",
  "translated": 4,
  "translated_percent": 100.0,
  "translated_words": 15,
  "url": "http://example.com/api/translations/hello/weblate/cs/",
  "web_url": "http://example.com/projects/hello/weblate/cs/"
}

```

```

DELETE /api/translations/(string: project) /
      string: component/string: language/

```

在 3.9 版本新加入.

Deletes a translation.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

**GET** /api/translations/ (**string**: *project*) /  
**string**: *component* / **string**: *language* / **changes** /

返回翻譯更改的列表。這本質上是仔細檢查的翻譯:<http://get:api/changes/>接受相同參數。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

#### Response JSON Object

- **results** (*array*) –組件對象的矩陣; 請參見 [GET /api/changes/ \(int:id\) /](#)

**GET** /api/translations/ (**string**: *project*) /  
**string**: *component* / **string**: *language* / **units** /

返回翻譯單元的列表。

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code
- **q** (*string*) –搜索查詢字符串:ref:‘Searching’ (可選)

#### Response JSON Object

- **results** (*array*) –組件對象的矩陣; 請參見 [GET /api/units/ \(int:id\) /](#)

**POST** /api/translations/ (**string**: *project*) /  
**string**: *component* / **string**: *language* / **units** /

Add new unit.

#### Parameters

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

#### Request JSON Object

- **key** (*string*) –翻譯單元的名稱
- **value** (*array*) –Source strings (use single string if not creating plural)

#### Response JSON Object

- **unit** (*object*) –newly created unit; see [GET /api/units/ \(int:id\) /](#)

也參考:

管理字串, adding-new-strings

**POST** /api/translations/(string: project) /  
 string: component/string: language/autotranslate/  
 觸發自動翻譯。

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug
- **language** (string) –Translation language code

#### Request JSON Object

- **mode** (string) –自動翻譯模式
- **filter\_type** (string) –自動翻譯篩選類型
- **auto\_source** (string) –Automatic translation source - mt or others
- **component** (string) –開 對專案的共享翻譯記憶作貢獻，以取得其他組件的存取權。
- **engines** (array) –機器翻譯引擎
- **threshold** (string) –分數 值

**GET** /api/translations/(string: project) /  
 string: component/string: language/file/  
 下載存儲在 VCS 中的當前翻譯文件（不帶“format”參數）或將其轉 另一格式（見 下載翻譯）。

---

備：這個 API 端點使用了不同於 API 其余的邏輯來輸出，它在整個文件而不是在數據上操作。接受的“format”參數組不同， 有這個參數會將翻譯文件存儲在版本控制系統（VCS）中。

---

#### 查詢參數

- **format** –使用的文件格式；如果不指定，則不會進行格式轉 ；支持的文件格式有：po, mo, xiff, xiff11, txb` ` , ``csv, xlsx, json, aresource, strings
- **q** (string) –Filter downloaded strings, see search, only applicable when conversion is in place (format is specified).

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug
- **language** (string) –Translation language code

**POST** /api/translations/(string: project) /  
 string: component/string: language/file/  
 上傳帶有翻譯的新文件。

#### Parameters

- **project** (string) –專案 URL slug
- **component** (string) –組件 URL slug
- **language** (string) –Translation language code

#### 表格參數

- **string conflicts** –如何處理衝突 (ignore, replace-translated or replace-approved)

- **file file** –上傳文件
- **string email** –作者郵件信箱
- **string author** –作者姓名
- **string method** –上傳方法 (translate, approve, suggest, fuzzy, replace, source, add), 見:ref: *upload-method*
- **string fuzzy** –模糊 (標記需要編輯) 的字符串處理 (*empty*, *process*, *approve*)

**CURL 示例:**

```
curl -X POST \
  -F file=@strings.xml \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/translations/hello/android/cs/file/
```

**GET /api/translations/(string: project) /**  
**string: component/string: language/repository/**

返回版本控制系統 (VCS) 倉庫狀態的信息。

響應與: `http:GET:/api/components/(string:project)/(string:component)/repository/` 的相同。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

**POST /api/translations/(string: project) /**  
**string: component/string: language/repository/**

在版本控制系統 (VCS) 倉庫上執行給定的操作。

文檔請參見 `POST /api/projects/(string:project)/repository/`。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

**Request JSON Object**

- **operation** (*string*) –執行的操作: push, pull, commit, reset, “cleanup” 之一

**Response JSON Object**

- **result** (*boolean*) –操作的結果

**GET /api/translations/(string: project) /**  
**string: component/string: language/statistics/**

返回具體的翻譯統計數據。


在 2.7 版本新加入。

**Parameters**

- **project** (*string*) –專案 URL slug
- **component** (*string*) –組件 URL slug
- **language** (*string*) –Translation language code

**Response JSON Object**



- **code**(*string*) – 語言代碼
- **failing**(*int*) – 檢查失敗的數量
- **failing\_percent**(*float*) – 檢查失敗的百分比
- **fuzzy**(*int*) – number of fuzzy (marked for edit) strings
- **fuzzy\_percent**(*float*) – 模糊字符串（標記需要編輯）的百分比
- **total\_words**(*int*) – 詞的總數
- **translated\_words**(*int*) – 已翻譯詞的數量
- **last\_author**(*string*) – 最後一位作者的姓名
- **last\_change**(*timestamp*) – 上次更改的日期
- **name**(*string*) – 語言名稱
- **total**(*int*) – 字符串的總數
- **translated**(*int*) – 已翻譯的字符串數量
- **translated\_percent**(*float*) – 已翻譯字符串的百分比
- **url**(*string*) – 訪問翻譯的 URL（約定的 URL）
- **url\_translate**(*string*) – 訪問翻譯的 URL（真實翻譯的 URL）

### 1.12.10 記憶

在 4.14 版本新加入。

**GET** /api/memory/

Returns a list of memory results.

**DELETE** /api/memory/(int: *memory\_object\_id*) /

Deletes a memory object

#### Parameters

- **memory\_object\_id** – Memory Object ID

### 1.12.11 Units

“單位”是一個翻譯的單曲，它對具有相應翻譯的字符串對的源字符串，也包含一些相關元數據。該術語來自“翻譯工具包”<<http://docs.translatehouse.org/projects/translate-toolkit/en/latest/api/storage.html#translate.storage.base.translationUnit>>\_ 和 xlfiff。

在 2.10 版本新加入。

**GET** /api/units/

返回翻譯單元的列表。

#### Parameters

- **q**(*string*) – 搜索查詢字符串:ref:‘Searching’（可選）

也參考:

單元對象屬性存檔在 `GET /api/units/(int:id)/`。

**GET** /api/units/(int: *id*) /

在 4.3 版本變更: `target` 和 `source` 現在是矩陣，來適當的處理多個字符串。

返回翻譯單元的信息。

#### Parameters

- **id**(*int*) –單元 ID

#### Response JSON Object

- **translation**(*string*) –相關翻譯對象的 URL
- **source**(*array*) –來源字串
- **previous\_source**(*string*) –用於模糊匹配的之前的源字符串
- **target**(*array*) –目標字符串
- **id\_hash**(*string*) –單元的唯一識文字
- **content\_hash**(*string*) –源字符串的唯一識文字
- **location**(*string*) –源代碼中單元的位置
- **context**(*string*) –翻譯單元的語境
- **note**(*string*) –翻譯單元的解
- **flags**(*string*) –翻譯單元的標記
- **labels**(*array*) –translation unit labels, available on source units
- **state**(*int*) –unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved, 100 - read only
- **fuzzy**(*boolean*) –whether the unit is fuzzy or marked for review
- **translated**(*boolean*) –whether the unit is translated
- **approved**(*boolean*) –whether the translation is approved
- **position**(*int*) –翻譯文件中的單元位置
- **has\_suggestion**(*boolean*) –whether the unit has suggestions
- **has\_comment**(*boolean*) –whether the unit has comments
- **has\_failing\_check**(*boolean*) –whether the unit has failing checks
- **num\_words**(*int*) –源詞的數量
- **priority**(*int*) –翻譯優先級; 100 默認值
- **id**(*int*) –單元識問題
- **explanation**(*string*) –字符串的解釋, 可在源單元獲得, 請參見源字符串另外的信息
- **extra\_flags**(*string*) –另外的字符串標記, 可在源單元獲得, 請參見使用標自定義行
- **web\_url**(*string*) –單元可以被編輯的 URL
- **source\_unit**(*string*) –源單元鏈接; 請參見 `GET /api/units/(int:id)/`
- **pending**(*boolean*) –whether the unit is pending for write
- **timestamp**(*timestamp*) –string age

**PATCH** `/api/units/(int: id) /`

在 4.3 版本新加入.

Performs partial update on translation unit.

#### Parameters

- **id**(*int*) –單元 ID

#### Request JSON Object

- **state** (*int*) –unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see 專門的審核者)
- **target** (*array*) –目標字符串
- **explanation** (*string*) –字符串的解釋，可在源單元獲得，請參見源字符串另外的信息
- **extra\_flags** (*string*) –另外的字符串標記，可在源單元獲得，請參見使用標 自定義行

#### Response JSON Object

- **labels** (*array*) –labels, available on source units

**PUT** /api/units/ (*int: id*) /

在 4.3 版本新加入。

Performs full update on translation unit.

#### Parameters

- **id** (*int*) –單元 ID

#### Request JSON Object

- **state** (*int*) –unit state, 0 - untranslated, 10 - needs editing, 20 - translated, 30 - approved (need review workflow enabled, see 專門的審核者)
- **target** (*array*) –目標字符串
- **explanation** (*string*) –字符串的解釋，可在源單元獲得，請參見源字符串另外的信息
- **extra\_flags** (*string*) –另外的字符串標記，可在源單元獲得，請參見使用標 自定義行

#### Response JSON Object

- **labels** (*array*) –labels, available on source units

**DELETE** /api/units/ (*int: id*) /

在 4.3 版本新加入。

Deletes a translation unit.

#### Parameters

- **id** (*int*) –單元 ID

## 1.12.12 更動

在 2.10 版本新加入。

**GET** /api/changes/

在 4.1 版本變更: 更改的篩選在 4.1 版本引入。

返回翻譯更改的列表。

#### 也參考:

更改對象的屬性存檔在 `GET /api/changes/(int:id)/`。

#### 查詢參數

- **user** (*string*) –篩選用 的 名
- **action** (*int*) –篩選的動作，可以多次使用

- **timestamp\_after** (*timestamp*) –ISO 8601 格式的時間標, 列出此時間之後的更改
- **timestamp\_before** (*timestamp*) –ISO 8601 格式的時間標, 列出此時間之前的更改

**GET** /api/changes/ (*int: id*) /

返回有關翻譯更改的信息。

#### Parameters

- **id** (*int*) –更改的 ID

#### Response JSON Object

- **unit** (*string*) –相關單元對象的 URL
- **translation** (*string*) –相關翻譯對象的 URL
- **component** (*string*) –相關組件對象的 URL
- **user** (*string*) –相關用對象的 URL
- **author** (*string*) –相關作者對象的 URL
- **timestamp** (*timestamp*) –時間的時間標
- **action** (*int*) –動作的幾種識
- **action\_name** (*string*) –動作的文本描述
- **target** (*string*) –更改的事件的文本或細節
- **id** (*int*) –更改的識文字

### 1.12.13 畫面快照

在 2.14 版本新加入。

**GET** /api/screenshots/

返回屏幕截圖字符串信息的列表。

#### 也參考:

屏幕截圖對象的屬性存檔在 [GET /api/screenshots/\(int:id\)/](#)。

**GET** /api/screenshots/ (*int: id*) /

返回與屏幕截圖信息有關的信息。

#### Parameters

- **id** (*int*) –屏幕截圖的 ID

#### Response JSON Object

- **name** (*string*) –屏幕截圖的名稱
- **component** (*string*) –相關組件對象的 URL
- **file\_url** (*string*) –下載文件的 URL; 請參見 [GET /api/screenshots/\(int:id\)/file/](#)
- **units** (*array*) –與源字符串信息相關的鏈接; 請參見 [GET /api/units/\(int:id\)/](#)

**GET** /api/screenshots/ (*int: id*) /file/

下載屏幕截圖的圖片。

#### Parameters

- **id**(*int*) – 屏幕截圖的 ID

**POST /api/screenshots/(int: id)/file/**

替 屏幕截圖。

#### Parameters

- **id**(*int*) – 屏幕截圖的 ID

#### 表格參數

- **file image** – 上傳文件

#### CURL 示例:

```
curl -X POST \
  -F image=@image.png \
  -H "Authorization: Token TOKEN" \
  http://example.com/api/screenshots/1/file/
```

**POST /api/screenshots/(int: id)/units/**

與屏幕截圖相關的源字符串。

#### Parameters

- **id**(*int*) – 屏幕截圖的 ID

#### 表格參數

- **string unit\_id** – 單元 ID

#### Response JSON Object

- **name**(*string*) – 屏幕截圖的名稱
- **translation**(*string*) – 相關翻譯對象的 URL
- **file\_url**(*string*) – 下載文件的 URL; 請參見 [GET /api/screenshots/\(int:id\)/file/](#)
- **units**(*array*) – 與源字符串信息相關的鏈接; 請參見 [GET /api/units/\(int:id\)/](#)

**DELETE /api/screenshots/(int: id)/units/**

**int: unit\_id**

Remove source string association with screenshot.

#### Parameters

- **id**(*int*) – 屏幕截圖的 ID
- **unit\_id** – Source string unit ID

**POST /api/screenshots/**

新建新的屏幕截圖。

#### 表格參數

- **file image** – 上傳文件
- **string name** – 畫面快照名稱
- **string project\_slug** – 項目標識串
- **string component\_slug** – 組件標識串
- **string language\_code** – 語言碼

#### Response JSON Object

- **name**(*string*) – 屏幕截圖的名稱

- **component** (*string*) – 相關組件對象的 URL
- **file\_url** (*string*) – 下載文件的 URL；請參見 `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – 與源字符串信息相關的鏈接；請參見 `GET /api/units/(int:id)/`

**PATCH /api/screenshots/(int: id) /**

Edit partial information about screenshot.

#### Parameters

- **id** (*int*) – 屏幕截圖的 ID

#### Response JSON Object

- **name** (*string*) – 屏幕截圖的名稱
- **component** (*string*) – 相關組件對象的 URL
- **file\_url** (*string*) – 下載文件的 URL；請參見 `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – 與源字符串信息相關的鏈接；請參見 `GET /api/units/(int:id)/`

**PUT /api/screenshots/(int: id) /**

Edit full information about screenshot.

#### Parameters

- **id** (*int*) – 屏幕截圖的 ID

#### Response JSON Object

- **name** (*string*) – 屏幕截圖的名稱
- **component** (*string*) – 相關組件對象的 URL
- **file\_url** (*string*) – 下載文件的 URL；請參見 `GET /api/screenshots/(int:id)/file/`
- **units** (*array*) – 與源字符串信息相關的鏈接；請參見 `GET /api/units/(int:id)/`

**DELETE /api/screenshots/(int: id) /**

Delete screenshot.

#### Parameters

- **id** (*int*) – 屏幕截圖的 ID

## 1.12.14 附加元件

在 4.4.1 版本新加入。

**GET /api/addons/**

Returns a list of add-ons.

#### 也參考：

附加組件對象屬性記 F 在 `GET /api/units/(int:id)/`。

**GET /api/addons/(int: id) /**

Returns information about add-on information.

#### Parameters

- **id** (*int*) –Add-on ID

#### Response JSON Object

- **name** (*string*) –name of an add-on
- **component** (*string*) –相關組件對象的 URL
- **configuration** (*object*) –Optional add-on configuration

也參考:

附加元件

**POST** /api/components/ (*string*: *project*) /  
*string*: *component/addons/*

Creates a new add-on.

#### Parameters

- **project\_slug** (*string*) –項目標識串
- **component\_slug** (*string*) –組件標識串

#### Request JSON Object

- **name** (*string*) –name of an add-on
- **configuration** (*object*) –Optional add-on configuration

**PATCH** /api/addons/ (*int*: *id*) /

Edit partial information about add-on.

#### Parameters

- **id** (*int*) –Add-on ID

#### Response JSON Object

- **configuration** (*object*) –Optional add-on configuration

**PUT** /api/addons/ (*int*: *id*) /

Edit full information about add-on.

#### Parameters

- **id** (*int*) –Add-on ID

#### Response JSON Object

- **configuration** (*object*) –Optional add-on configuration

**DELETE** /api/addons/ (*int*: *id*) /

Delete add-on.

#### Parameters

- **id** (*int*) –Add-on ID

### 1.12.15 組件列表

在 4.0 版本新加入。

**GET** `/api/component-lists/`

返回組件列表的列表。

**也參考：**

組件列表對象屬性存檔在 `GET /api/component-lists/(str:slug)/`。

**GET** `/api/component-lists/(str: slug) /`

返回組件列表的信息。

#### Parameters

- **slug** (*string*) –組件列表的標識串

#### Response JSON Object

- **name** (*string*) –組件列表的名稱
- **slug** (*string*) –組件列表的表示串
- **show\_dashboard** (*boolean*) –是否在控制台上顯示
- **components** (*array*) –相關聯組件的連接；請參見 `GET /api/components/(string:project)/(string:component)/`
- **auto\_assign** (*array*) –自動分配規則

**PUT** `/api/component-lists/(str: slug) /`

更改組件列表參數。

#### Parameters

- **slug** (*string*) –組件列表的標識串

#### Request JSON Object

- **name** (*string*) –組件列表的名稱
- **slug** (*string*) –組件列表的表示串
- **show\_dashboard** (*boolean*) –是否在控制台上顯示

**PATCH** `/api/component-lists/(str: slug) /`

更改組件列表參數。


#### Parameters

- **slug** (*string*) –組件列表的標識串

#### Request JSON Object

- **name** (*string*) –組件列表的名稱
- **slug** (*string*) –組件列表的表示串
- **show\_dashboard** (*boolean*) –是否在控制台上顯示

**DELETE** `/api/component-lists/(str: slug) /`

 除組件列表。

#### Parameters

- **slug** (*string*) –組件列表的標識串



**POST** `/api/component-lists/(str: slug)/components/`

使組件與組件列表相關。

#### Parameters

- **slug** (*string*) –組件列表的標識串

#### 表格參數

- **string component\_id** –組件 ID

**DELETE** `/api/component-lists/(str: slug)/components/`

**str:** *component\_slug*

將組件與組件列表接觸相關性。

#### Parameters

- **slug** (*string*) –組件列表的標識串
- **component\_slug** (*string*) –組件標識串

### 1.12.16 詞表

在 4.5 版本變更: 表格現在存儲常規組件, 翻譯和字符串, 請使用相應的 API。

### 1.12.17 Tasks

在 4.4 版本新加入.

**GET** `/api/tasks/`

Listing of the tasks is currently not available.

**GET** `/api/tasks/(str: uuid) /`

Returns information about a task

#### Parameters

- **uuid** (*string*) –任務 UUID

#### Response JSON Object

- **completed** (*boolean*) –Whether the task has completed
- **progress** (*int*) –Task progress in percent
- **result** (*object*) –任務結果或過程細節
- **log** (*string*) –Task log

### 1.12.18 Metrics

**GET** `/api/metrics/`

Returns server metrics.

#### Response JSON Object

- **units** (*int*) –Number of units
- **units\_translated** (*int*) –Number of translated units
- **users** (*int*) –Number of users
- **changes** (*int*) –變更數
- **projects** (*int*) –Number of projects

- **components** (*int*) – 組件數目
- **translations** (*int*) – Number of translations
- **languages** (*int*) – Number of used languages
- **checks** (*int*) – Number of triggered quality checks
- **configuration\_errors** (*int*) – Number of configuration errors
- **suggestions** (*int*) – Number of pending suggestions
- **celery\_queues** (*object*) – Celery 排程序列的長度。參 使用 *Celery* 的後台任務
- **name** (*string*) – Configured server name

### 1.12.19 通知勾

通知子允許外部應用來通知 Weblate 版本控制系統（VCS）倉庫已經更新。

可以項目、組件和翻譯使用倉庫端點來更新各自的倉庫；文檔請參見 `POST /api/projects/(string:project)/repository/`。

**GET /hooks/update/(string: project) /**  
**string: component/**

在 2.6 版本開始 用：請使用 `POST /api/components/(string:project)/(string:component)/repository/` 來替代，它使用 ACL 限制的身份驗證而工作正常。

觸發組件的更新（從版本控制系統 VCS 拉取掃描翻譯的更改）。

**GET /hooks/update/(string: project) /**

在 2.6 版本開始 用：請使用 `POST /api/projects/(string:project)/repository/` 來替代，它使用 ACL 限制的身份驗證而工作正常。

觸發項目中所有組件的更新（從版本控制系統 VCS 拉取掃描翻譯的更改）。

**POST /hooks/github/**

處理 Github 通知與自動更新匹配組件的特殊子。

---

備：Github 包括了對通知 Weblate 的直接支持：在倉庫設置中動 Weblate 服務子，將 URL 設置你的 Weblate 安裝的 URL。

---

也參考：

從 *GitHub* 自動接收更改

關於設置 Github 集成的指令

<https://docs.github.com/en/get-started/customizing-your-github-workflow/exploring-integrations/about-webhooks>

GitHub Webhooks 的一般信息

**ENABLE\_HOOKS**

關於對整個 Weblate 動子

**POST /hooks/gitlab/**

處理 GitLab 通知自動更新匹配組件的特殊子。

也參考：

從 *GitLab* 自動接收更改

關於設置 GitLab 集成的指示

<https://docs.gitlab.com/ee/user/project/integrations/webhooks.html>

關於 GitLab Webhooks 的一般信息

#### **ENABLE\_HOOKS**

關於對整個 Weblate 啟用

#### **POST /hooks/bitbucket/**

處理 Bitbucket 通知自動更新匹配的組件的特殊子。

也參考:

#### 從 *Bitbucket* 自動接收更改

關於設置 Bitbucket 集成的指示

<https://support.atlassian.com/bitbucket-cloud/docs/manage-webhooks/>

關於 Bitbucket Webhooks 的一般信息

#### **ENABLE\_HOOKS**

關於對整個 Weblate 啟用

#### **POST /hooks/pagure/**

在 3.3 版本新加入。

處理 Pagure 通知自動更新匹配的組件的特殊子。

也參考:

#### 從 *Pagure* 自動接受更改

關於設置 Pagure 集成的指示

[https://docs.pagure.org/pagure/usage/using\\_webhooks.html](https://docs.pagure.org/pagure/usage/using_webhooks.html)

關於 Pagure Webhooks 的一般信息

#### **ENABLE\_HOOKS**

關於對整個 Weblate 啟用

#### **POST /hooks/azure/**

在 3.8 版本新加入。

Special hook for handling Azure DevOps notifications and automatically updating matching components.

---

備註: Please make sure that *Resource details to send* is set to *All*, otherwise Weblate will not be able to match your Azure repository.

---

也參考:

#### 從 *Azure Repos* 自動接收更改

關於設置 Azure 集成的指示

<https://learn.microsoft.com/en-us/azure/devops/service-hooks/services/webhooks?view=azure-devops>

Generic information about Azure DevOps Web Hooks

#### **ENABLE\_HOOKS**

關於對整個 Weblate 啟用

#### **POST /hooks/gitea/**

在 3.9 版本新加入。

處理 Gitea Webhook 通知自動更新匹配的組件的特殊子。

也參考:

### 從 Gitea Repos 自動接收更改

關於設置 Gitea 集成的指示

<https://docs.gitea.io/en-us/webhooks/>

關於 Gitea Webhooks 的一般信息

### ENABLE\_HOOKS

關於對整個 Weblate 啟動子

**POST /hooks/gitee/**

在 3.9 版本新加入。

處理 Gitee Webhook 通知自動更新匹配的組件的特殊子。

也參考：

### 從 Gitee Repos 自動接收更改

關於設置 Gitee 集成的指示

<https://gitee.com/help/categories/40>

關於 Gitee Webhooks 的一般信息

### ENABLE\_HOOKS

關於對整個 Weblate 啟動子

## 1.12.20 Exports

Weblate 提供各種導出，允許進一步處理數據。

**GET /exports/stats/(string: project) /**  
**string: component/**

查詢參數

- **format** (*string*) –輸出格式：json 或 csv

在 2.6 版本開始用：請替代使用：[http://get:/api/components/\(string:project\)/\(string:component\)/statistics/](http://get:/api/components/(string:project)/(string:component)/statistics/) 和 [http://get:/api/translations/\(string:project\)/\(string:component\)/\(string:language\)/statistics/](http://get:/api/translations/(string:project)/(string:component)/(string:language)/statistics/)；它也允許訪問 ACL 控制的項目。

給定的組件以給定的格式檢索統計數據。

示例請求：

```
GET /exports/stats/weblate/main/ HTTP/1.1
Host: example.com
Accept: application/json, text/javascript
```

示例響應：

```
HTTP/1.1 200 OK
Vary: Accept
Content-Type: application/json

[
  {
    "code": "cs",
    "failing": 0,
    "failing_percent": 0.0,
    "fuzzy": 0,
    "fuzzy_percent": 0.0,
    "last_author": "Michal Čihař",
    "last_change": "2012-03-28T15:07:38+00:00",
    "name": "Czech",
```

(繼續下一頁)

(繼續上一頁)

```

    "total": 436,
    "total_words": 15271,
    "translated": 436,
    "translated_percent": 100.0,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/cs/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/cs/"
  },
  {
    "code": "nl",
    "failing": 21,
    "failing_percent": 4.8,
    "fuzzy": 11,
    "fuzzy_percent": 2.5,
    "last_author": null,
    "last_change": null,
    "name": "Dutch",
    "total": 436,
    "total_words": 15271,
    "translated": 319,
    "translated_percent": 73.2,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/nl/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/nl/"
  },
  {
    "code": "el",
    "failing": 11,
    "failing_percent": 2.5,
    "fuzzy": 21,
    "fuzzy_percent": 4.8,
    "last_author": null,
    "last_change": null,
    "name": "Greek",
    "total": 436,
    "total_words": 15271,
    "translated": 312,
    "translated_percent": 71.6,
    "translated_words": 3201,
    "url": "http://hosted.weblate.org/engage/weblate/el/",
    "url_translate": "http://hosted.weblate.org/projects/weblate/main/el/"
  }
]

```

### 1.12.21 RSS 消息來源

翻譯的更改導出到 RSS 頻道。

**GET** `/exports/rss/(string: project) /`  
**string: component/string: language/**  
 用翻譯近期的更改檢索 RSS 頻道。

**GET** `/exports/rss/(string: project) /`  
**string: component/**  
 用組件的近期更改檢索 RSS 頻道。

**GET** `/exports/rss/(string: project) /`  
 用項目的近期更改檢索 RSS 頻道。

**GET** `/exports/rss/language/ (string: language) /`

用語言的近期更改檢索 RSS 頻道。

**GET** `/exports/rss/`

用 Weblate 事件的近期更改檢索 RSS 頻道。

**也參考:**

[RSS on Wikipedia](#)

## 1.13 Weblate 客端

在 2.7 版本新加入: 自從 Weblate 2.7 以來, 已經有完整的 `wlc` 實用程序支持。如果您使用的是舊版本, 則可能會與 API 發生某些不兼容。

### 1.13.1 安裝

Weblate 客端是分開上市的, 包括 Python 模塊。要使用下面的命令, 您需要安裝 `wlc`:

```
pip install wlc
```

### 1.13.2 Docker usage

Web2 客端也可作 Docker Image 提供。

該圖像在 Docker Hub 上發布: <https://hub.docker.com/r/weblate/wlc>

Installing:

```
docker pull weblate/wlc
```

Docker Container 使用 Weblate 的默認設置連接到位於 `localhost` 中部署的 API。API URL 和 API\_KEY 可以通過 Weblate 接受的參數配置。

啟動容器的命令使用以下語法:

```
docker run --rm weblate/wlc [WLC_ARGS]
```

例:

```
docker run --rm weblate/wlc --url https://hosted.weblate.org/api/ list-projects
```

您可能希望通過您的: REF: “WLC-CONFIG” (WLC-CONFIG 給 Docker 容器, 最簡單的方法是將您當前的目錄添加: file: `:/home/weblate` 卷:

```
docker run --volume $PWD:/home/weblate --rm weblate/wlc show
```

### 1.13.3 入門

`wlc` 配置存儲在 `~/.config/weblate` 中（其他位置參見 [ref:wlc-config](#)），請創建它以匹配您的環境：

```
[weblate]
url = https://hosted.weblate.org/api/

[keys]
https://hosted.weblate.org/api/ = APIKEY
```

然後，您可以在默認服務器上調用命令：

```
wlc ls
wlc commit sandbox/hello-world
```

也參考：

[配置文件](#)

### 1.13.4 Synopsis

```
wlc [arguments] <command> [options]
```

命令實際上指示應該執行哪個操作。

### 1.13.5 描述

Weblate 客戶端是一個 Python 庫和命令行實用程序，可使用 *Weblate* 的 *REST API* 遠程管理 Weblate。命令行實用程序可以作 `wlc` 調用，且置在 `wlc` 上。

#### Arguments

程序接受以下參數來定義輸出格式或使用哪個 Weblate 實例。這些參數必須位於任何命令之前。

**--format** {csv,json,text,html}

指定輸出格式。

**--url** URL

指定 API URL。覆蓋在配置文件中找到的任何值，請參[配置文件](#)。該網址應以 `/api/` 結尾，例如 `https://hosted.weblate.org/api/`。

**--key** KEY

指定要使用的 API 用密鑰。覆蓋在配置文件中找到的任何值，請參[配置文件](#)。您可以在 Weblate 的個人資料中找到密鑰。

**--config** PATH

覆蓋配置文件路徑，請參[配置文件](#)。

**--config-section** SECTION

覆蓋正在使用的配置文件部分，請參[配置文件](#)。

## Commands

以下命令可用：

### **version**

打印當前版本。

### **list-languages**

列出 Weblate 中使用的語言。

### **list-projects**

列出 Weblate 中的項目。

### **list-components**

Lists components in Weblate.

### **list-translations**

Lists translations in Weblate.

### **show**

顯示 Weblate 對象（翻譯，組件或項目）。

### **ls**

列出 Weblate 對象（翻譯，組件或項目）。

### **commit**

提交在 Weblate 對象（翻譯，組件或項目）中所做的更改。

### **pull**

拉取遠程倉庫的更改到 Weblate 對象中（翻譯，組件或項目）。

### **push**

將 Weblate 對象更改推送到遠程倉庫（翻譯，組件或項目）。

### **reset**

在 0.7 版本新加入: Supported since wlc 0.7.

重置 Weblate 對象中的更改以匹配遠程存儲庫（翻譯，組件或項目）。

### **cleanup**

在 0.9 版本新加入: Supported since wlc 0.9.

刪除 Weblate 對象中所有未跟踪的更改以匹配遠程倉庫（翻譯，組件或項目）。

### **repo**

顯示給定 Weblate 對象（翻譯，組件或項目）的倉庫狀態。

### **stats**

顯示給定 Weblate 對象（翻譯，組件或項目）的詳細統計數據。

### **lock-status**

在 0.5 版本新加入: Supported since wlc 0.5.

顯示鎖定狀態。

### **lock**

在 0.5 版本新加入: Supported since wlc 0.5.

鎖定組件以防止在 Weblate 中進一步翻譯。

### **unlock**

在 0.5 版本新加入: Supported since wlc 0.5.

解鎖 Weblate 組件的翻譯。



**changes**

在 0.7 版本新加入: 從 wlc 0.7 和 Weblate 2.10 開始受支持。

顯示給定對象的更改。

**download**

在 0.7 版本新加入: Supported since wlc 0.7.

Downloads a translation file.

**--convert**

轉換文件格式, 如果未指定, 則在服務器上不進行任何轉換, 且將文件原樣下載到倉庫中。

**--output**

指定要保存輸出的文件, 如果未指定, 則將其打印到 stdout。

**upload**

在 0.9 版本新加入: Supported since wlc 0.9.

Uploads a translation file.

**--overwrite**

上傳時覆蓋現有翻譯。

**--input**

從中讀取內容的文件, 如果未指定, 則從 stdin 中讀取。

**--method**

Upload method to use, see [導入方法](#).

**--fuzzy**

模糊 (標記需要編輯) 的字符串處理 (*empty*, *process*, *approve*)

**--author-name**

Author name, to override currently authenticated user

**--author-email**

Author e-mail, to override currently authenticated user

---

**提示:** 您可以通過傳遞 “--help” 獲取更多有關調用單個命令的更詳細信息: `wlc ls - help`.

---

### 1.13.6 配置文件

**.weblate, .weblate.ini, weblate.ini**

在 1.6 版本變更: 也接受了具有 ‘.ini’ 擴展的文件。

Per project configuration file

**C:\Users\NAME\AppData\weblate.ini**

在 1.6 版本新加入.

在 Windows 上使用配置文件。

**~/.config/weblate**

用戶配置文件

**/etc/xdg/weblate**

System wide configuration file

該程序遵循 XDG 規範, 因此您可以通過環境變量 `XDG_CONFIG_HOME` 或 `XDG_CONFIG_DIRS` 來調整配置文件的位置。在 Windows 系統上, ‘APPDATA’ 目錄是配置文件的首選位置。

可以在 [weblate] 部分中配置以下設置 (您可以通過 `--config-section` 進行自定義):

**key**

用於訪問 Weblate 的 API KEY。

**url**

API 服務器網址，默認 F `http://127.0.0.1:8000/api/`。

**translation**

默認翻譯的路徑——組件或項目。

配置文件是一個 INI 文件，例如：

```
[weblate]
url = https://hosted.weblate.org/api/
key = APIKEY
translation = weblate/application
```

另外，API 密鑰可以存儲在 [keys] 部分中：

```
[keys]
https://hosted.weblate.org/api/ = APIKEY
```

這樣，您就可以在版本控制系統（VCS）倉庫中使用 `.weblate` 配置時，將密鑰存儲在個人設置中，以便 `wlc` 知道它應該與哪個服務器通信。

## 1.13.7 Examples

Print current program version:

```
$ wlc version
version: 0.1
```

列出所有項目：

```
$ wlc list-projects
name: Hello
slug: hello
url: http://example.com/api/projects/hello/
web: https://weblate.org/
web_url: http://example.com/projects/hello/
```

Upload translation file:

```
$ wlc upload project/component/language --input /tmp/hello.po
```

您還可以指定 `wlc` 應該從事的項目：

```
$ cat .weblate
[weblate]
url = https://hosted.weblate.org/api/
translation = weblate/application

$ wlc show
branch: main
file_format: po
source_language: en
filemask: weblate/locale/*/LC_MESSAGES/django.po
git_export: https://hosted.weblate.org/git/weblate/application/
license: GPL-3.0+
license_url: https://spdx.org/licenses/GPL-3.0+
name: Application
new_base: weblate/locale/django.pot
```

(繼續下一頁)

(繼續上一頁)

```
project: weblate
repo: git://github.com/WeblateOrg/weblate.git
slug: application
template:
url: https://hosted.weblate.org/api/components/weblate/application/
vcs: git
web_url: https://hosted.weblate.org/projects/weblate/application/
```

通過此設置，可以輕鬆地提交當前項目中待定的更改：

```
$ wlc commit
```

## 1.14 Weblate 的 Python API

### 1.14.1 安裝

The Python API is shipped separately, you need to install the *Weblate* 客戶端 (wlc) to have it.

```
pip install wlc
```

### 1.14.2 wlc

#### **WeblateException**

**exception** `wlc.WeblateException`

所有異常的基類。

#### **Weblate**

**class** `wlc.Weblate` (*key=""*, *url=None*, *config=None*)

參數

- **key** (*str*) –User key
- **url** (*str*) –API Server URL，如果有指定默認值
- **config** (`wlc.config.WeblateConfig`) –配置對象，覆蓋任何其他參數。

訪問 API 的類，定義 API 鍵和可選的 API URL。

**get** (*path*)

參數

**path** (*str*) –Request path

回傳型 對象

object

執行單個 API 呼叫。

**post** (*path*, *\*\*kwargs*)

參數

**path** (*str*) –Request path

回傳型 對象

object

執行單個 API 呼叫。

### 1.14.3 wlc.config

#### WeblateConfig

```
class wlc.config.WeblateConfig (section='wlc')
```

參數

**section** (*str*) – Configuration section to use

XDG 規範之後的配置文件解析器。

```
load (path=None)
```

參數

**path** (*str*) – 從中加載配置的路徑。

從文件中加載配置，如果 有指定，則從 “WLC” 配置文件 (：“文件: /*.config* / *wlc*) 中加載 (：“file: /*etc* / *xdg* / *wlc*)。

### 1.14.4 wlc.main

```
wlc.main.main (settings=None, stdout=None, args=None)
```

參數

- **settings** (*list*) – 作元組列表覆蓋的設置
- **stdout** (*object*) – 用於打印輸出的 stdout 文件對象，使用 “sys.stdout”
- **args** (*list*) – 要處理的命令行參數，使用 “sys.args”

命令行界面的主要入口點。

```
@wlc.main.register_command (command)
```

Command` 裝飾器類 在主要解析程式:func:`main` 中。

#### Command

```
class wlc.main.Command (args, config, stdout=None)
```

用於調用命令的主類。

## 2.1 配置手冊

### 2.1.1 安裝 Weblate

#### Installing using Docker

通過 dockerized Weblate 部署，您可以在幾秒鐘內啟動並運行您的個人 Weblate 實例。Weblate 的所有依賴項已包含在內。PostgreSQL 被新建默認數據庫。

#### 硬件要求

Weblate 應該可以在任何現代硬件上正常運行，以下是在單個主機（Weblate，數據庫和 Web 服務器）上運行 Weblate 所需的最低配置：

- 2 GB 的內存
- 2 CPU cores
- 1 GB 的存儲空間

內存越多越好——用於所有級的緩存（文件系統，數據庫和 Weblate）。

許多應用會增加所需的 CPU 核數量。對於數百個翻譯組件，推薦至少有 4 GB 的內存。

典型的數據庫存儲用量大約每 1 百萬單詞 300 MB。克隆倉庫所需的存儲空間會變化，但 Weblate 試圖通過淺克隆將其大小最小化。

---

**備註：** 根據 Weblate 中管理的翻譯大小，安裝 Weblate 的實際要求差異很大。

---

## 安裝

以下示例假設您擁有一個工作正常的 Docker 環境，已安裝了 docker-compose。請查看 Docker 文檔以獲取說明。

1. 克隆 weblate-docker 存儲庫：

```
git clone https://github.com/WeblateOrg/docker-compose.git weblate-docker
cd weblate-docker
```

2. 建立一個 docker-compose.override.yml 檔案以調整您所需的設定。參見 [Docker environment variables](#) 以了解更多相關設定參數。

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SERVER_EMAIL: weblate@example.com
      WEBLATE_DEFAULT_FROM_EMAIL: weblate@example.com
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for the admin user
      WEBLATE_ADMIN_EMAIL: weblate.admin@example.com
```

---

備註：如果未設置 `WEBLATE_ADMIN_PASSWORD`，則使用首次啟動時顯示的隨機密碼創建管理員用戶。

提供的例子使 Weblate 偵聽端口 80，在 docker-compose.override.yml 文件中編輯端口映射來更改。

---

3. Start Weblate containers:

```
docker-compose up
```

享受您的 Weblate 部署，可以在 weblate 容器的端口 80 上進行訪問。

在 2.15-2 版本變更：最近更改了設置，以前有單獨的 web 服務器容器，因 2.15-2 開始，web 服務器已嵌入 Weblate 容器中。

在 3.7.1-6 版本變更：在 2019 年 7 月（從 3.7.1-6 標本開始）中，容器未以 root 用戶身份運行。這已將裸露端口從 80 更改為 8080。

### 也參考：

[Invoking management commands](#)

## Choosing Docker hub tag

You can use following tags on Docker hub, see <https://hub.docker.com/r/weblate/weblate/tags/> for full list of available ones.

Tag name	描述	用例
latest	Weblate stable release, matches latest tagged release	Rolling updates in a production environment
<VERSION>-<PATCH>	Weblate stable release	Well defined deploy in a production environment
edge	Weblate stable release with development changes in the Docker container (for example updated dependencies)	Rolling updates in a staging environment
edge-<DATE>-<PATCH>	Weblate stable release with development changes in the Docker container (for example updated dependencies)	Well defined deploy in a staging environment
bleeding	Development version Weblate from Git	Rolling updates to test upcoming Weblate features
bleeding-<DATE>-<PATCH>	Development version Weblate from Git	Well defined deploy to test upcoming Weblate features

Every image is tested by our CI before it gets published, so even the *bleeding* version should be quite safe to use.

## 具有 HTTPS 支持的 Docker 容器

請參閱安裝以獲取常規部署說明，本節僅提及與之相比的差別。

### 使用自己的 SSL 證書

在 3.8-3 版本新加入。

如果您要使用自己的 SSL 證書，只需將文件放入 Weblate 數據卷中（請參閱 [Docker container volumes](#)）：

- `ssl/fullchain.pem` 包含證書，包括任何需要的 CA 證書
- `ssl/privkey.pem` 包含有私鑰

擁有這兩個文件的用戶必須與啟動 docker 容器將文件掩碼設置 600（僅擁有可讀可寫的用戶同一用戶）。

此外，Weblate 容器現在將在端口 4443 上接受 SSL 連接，您將要在 docker compose override 中包括 HTTPS 的端口轉發：

```
version: '3'
services:
  weblate:
    ports:
      - 80:8080
      - 443:4443
```

如果您已經在同一服務器上管理其他站點，則反向代理（例如 NGINX）可能會使用端口 80 和 443。要將 HTTPS 連接從 NGINX 傳遞到 docker 容器，可以使用以下配置：

```
server {
    listen 443;
    listen [::]:443;

    server_name <SITE_URL>;
    ssl_certificate /etc/letsencrypt/live/<SITE>/fullchain.pem;
    ssl_certificate_key /etc/letsencrypt/live/<SITE>/privkey.pem;
```

(繼續下一頁)

(繼續上一頁)

```

location / {
    proxy_set_header HOST $host;
    proxy_set_header X-Forwarded-Proto https;
    proxy_set_header X-Real-IP $remote_addr;
    proxy_set_header X-Forwarded-For $proxy_add_x_forwarded_for;
    proxy_set_header X-Forwarded-Host $server_name;
    proxy_pass https://127.0.0.1:<EXPOSED_DOCKER_PORT>;
}

```

將 <SITE\_URL>, <SITE> 和 <EXPOSED\_DOCKER\_PORT> 替換為您環境中的實際值。

## 使用 Let's Encrypt 自動生成 SSL 證書

如果要在公共安裝中使用 Let's Encrypt <<https://letsencrypt.org/>> 自動生成的 SSL 證書，則需要在其他 Docker 容器中添加反向 HTTPS 代理，這將使用 'https-portal <<https://hub.docker.com/r/steveltm/https-portal/>>'。這是在 docker-compose-https.yml 文件中使用的。然後使用您的設置創建一個 docker-compose-https.override.yml 文件：

```

version: '3'
services:
  weblate:
    environment:
      WEBLATE_EMAIL_HOST: smtp.example.com
      WEBLATE_EMAIL_HOST_USER: user
      WEBLATE_EMAIL_HOST_PASSWORD: pass
      WEBLATE_SITE_DOMAIN: weblate.example.com
      WEBLATE_ADMIN_PASSWORD: password for admin user
  https-portal:
    environment:
      DOMAINS: 'weblate.example.com -> http://weblate:8080'

```

每當調用 **docker-compose** 時，您都需要將兩個文件都傳遞給它，然後執行以下操作：

```

docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml
↪build
docker-compose -f docker-compose-https.yml -f docker-compose-https.override.yml up

```

## 升級 Docker 容器

通常，只更新 Weblate 容器並保持 PostgreSQL 容器的版本是一個好主意，因為升級 PostgreSQL 會很痛苦，並且在大多數情況下不會帶來很多好處。

在 4.10-1 版本變更：Since Weblate 4.10-1, the Docker container uses Django 4.0 what requires PostgreSQL 10 or newer, please upgrade it prior to upgrading Weblate. See [Upgrade from 4.9 to 4.10](#) and [Upgrading PostgreSQL container](#).

您可以通過堅持使用現有的 docker-compose，並且只是拉取最新映像，然後重新啟動，來執行此操作：

```

# Fetch latest versions of the images
docker-compose pull
# Stop and destroy the containers
docker-compose down
# Spawn new containers in the background
docker-compose up -d
# Follow the logs during upgrade
docker-compose logs -f

```



Weblate 數據庫應在首次啟動時自動遷移，且不需要其他手動操作。

**備註：** Upgrades across major versions are not supported by Weblate. For example, if you are on 3.x series and want to upgrade to 4.x, first upgrade to the latest 4.0.x-y image (at time of writing this it is the 4.0.4-5), which will do the migration and then continue upgrading to newer versions.

You might also want to update the `docker-compose` repository, though it's not needed in most case. See [Upgrading PostgreSQL container](#) for upgrading the PostgreSQL server.

## Upgrading PostgreSQL container

PostgreSQL containers do not support automatic upgrading between version, you need to perform the upgrade manually. Following steps show one of the options of upgrading.

也參考：

<https://github.com/docker-library/postgres/issues/37>

1. Stop Weblate container:

```
docker-compose stop weblate cache
```

2. Backup the database:

```
docker-compose exec database pg_dumpall --clean --username weblate > backup.sql
```

3. Stop the database container:

```
docker-compose stop database
```

4. Remove the PostgreSQL volume:

```
docker-compose rm -v database
docker volume remove weblate-docker_postgres-data
```

5. Adjust `docker-compose.yml` to use new PostgreSQL version.

6. Start the database container:

```
docker-compose up -d database
```

7. Restore the database from the backup:

```
cat backup.sql | docker-compose exec -T database psql --username weblate --
↳ dbname postgres
```

8. Start all remaining containers:

```
docker-compose up -d
```

## Admin sign in

設置容器之後，您可以使用 `WEBLATE_ADMIN_PASSWORD` 中提供的密碼以管理員身份登入，或者如果未設置該密碼，則在首次啟動時生成隨機密碼。

要重置管理員密碼，請在 `WEBLATE_ADMIN_PASSWORD` 設置新密碼的情況下重啟容器。

也參考：

`WEBLATE_ADMIN_PASSWORD`, `WEBLATE_ADMIN_NAME`, `WEBLATE_ADMIN_EMAIL`

## 過程數量和記憶體消耗

基於 CPU 的數量，自動確定 UWSGI 和 CELERY 的工人流程數。這適用於大多數雲主機，因為它們通常具有很少的 CPU 和良好的記憶體。

在你有很多 CPU 核心且碰到記憶體用盡問題情況下，嘗試減少 worker 的數量：

```
environment:
  WEBLATE_WORKERS: 2
```

你還可以微調單個 worker 類型：

```
environment:
  WEB_WORKERS: 4
  CELERY_MAIN_OPTIONS: --concurrency 2
  CELERY_NOTIFY_OPTIONS: --concurrency 1
  CELERY_TRANSLATE_OPTIONS: --concurrency 1
```

也參考：

`WEBLATE_WORKERS`, `CELERY_MAIN_OPTIONS`, `CELERY_NOTIFY_OPTIONS`,  
`CELERY_MEMORY_OPTIONS`, `CELERY_TRANSLATE_OPTIONS`, `CELERY_BACKUP_OPTIONS`,  
`CELERY_BEAT_OPTIONS`, `WEB_WORKERS`

## Scaling horizontally

在 4.6 版本新加入。

您可以運行多個 WebBleate 容器以水平縮放服務。介紹：文件：`/app/data` 卷必須由所有容器共享，建議使用群集文件系統，如 *Glusterfs*。：文件：`/app/cache` 卷應該每個容器分開。

每個 WebLete 容器都有定義的角色：envvar: `weblate_service` 環境變量。請仔細跟蹤文檔，因為某些服務應該在群集中僅運行一次，且服務的訂單也是如此。

您可以在 “docker-compose“ repo 中”找到示例設置 `“docker-compose-split.yml <https://github.com/weblateorg/docker-compose/blob/main/docker-compose-split.yml>“。`

## Docker environment variables

Many of Weblate’s 配置 can be set in the Docker container using the environment variables described below.

If you need to define a setting not exposed through Docker environment variables, see *Configuration beyond environment variables*.

## 通用設置

### WEBLATE\_DEBUG

使用 *DEBUG* 配置 Django 調試模式。

示例：

```
environment:
  WEBLATE_DEBUG: 1
```

也參考：

*Disable debug mode*

### WEBLATE\_LOGLEVEL

配置日誌記錄的詳細程度。

### WEBLATE\_LOGLEVEL\_DATABASE

Configures the logging of the database queries verbosity.

### WEBLATE\_SITE\_TITLE

更改所有頁面頁眉上顯示的站點標題。

### WEBLATE\_SITE\_DOMAIN

Configures the site domain. This parameter is required.

也參考：

設置正確的網站域名, *SITE\_DOMAIN*

### WEBLATE\_ADMIN\_NAME

### WEBLATE\_ADMIN\_EMAIL

配置站點管理員的姓名和電子郵件。它用於 *ADMINS* 設置和創建 管理員用 (有關此信息，請參閱 *WEBLATE\_ADMIN\_PASSWORD*)。

示例：

```
environment:
  WEBLATE_ADMIN_NAME: Weblate admin
  WEBLATE_ADMIN_EMAIL: noreply@example.com
```

也參考：

*Admin sign in, Properly configure admins, ADMINS*

### WEBLATE\_ADMIN\_PASSWORD

設置 管理員用的密碼。

- 如果未設置且 管理員用不存在，則會使用首次啟動容器時顯示的隨機密碼來創建它。
- 如果未設置且 管理員用存在，則不執行任何操作。
- 如果設置，則在每次容器啟動時都會對 管理員用進行調整，以匹配 *WEBLATE\_ADMIN\_PASSWORD*，*WEBLATE\_ADMIN\_NAME* 和 *WEBLATE\_ADMIN\_EMAIL*。

**警告：** 將密碼存儲在配置文件中可能會帶來安全風險。考慮僅將此變量用於初始設置（或讓 Weblate 在初始啟動時生成隨機密碼）或用於密碼恢復。

也參考：

：參考：“docker-admin-login”

**WEBLATE\_ADMIN\_PASSWORD\_FILE**

設置指向包含 管理員用密碼的一個文件。

也參考:

*WEBLATE\_ADMIN\_PASSWORD*

**WEBLATE\_SERVER\_EMAIL**

The email address that error messages are sent from.

也參考:

*SERVER\_EMAIL, Configure e-mail sending*

**WEBLATE\_DEFAULT\_FROM\_EMAIL**

配置外發電子郵件的地址。

也參考:

*DEFAULT\_FROM\_EMAIL, Configure e-mail sending*

**WEBLATE\_CONTACT\_FORM**

配置聯表單行，請參：設置： *contact\_form*。

**WEBLATE\_ALLOWED\_HOSTS**

使用 *ALLOWED\_HOSTS* 配置允許的 HTTP 主機名。

默認 \* 來允許所有的主機名稱。

示例:

```
environment:
  WEBLATE_ALLOWED_HOSTS: weblate.example.com,example.com
```

也參考:

*ALLOWED\_HOSTS, Allowed hosts setup*, 設置正確的網站域名

**WEBLATE\_REGISTRATION\_OPEN**

通過切 *REGISTRATION\_OPEN* 配置是否打開。

示例:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
```

**WEBLATE\_REGISTRATION\_ALLOW\_BACKENDS**

配置可用於通過 *REGISTRATION\_ALLOW\_BACKENDS* 創建新帳的身份驗證方法。

示例:

```
environment:
  WEBLATE_REGISTRATION_OPEN: 0
  WEBLATE_REGISTRATION_ALLOW_BACKENDS: azuread-oauth2,azuread-tenant-
  ↪oauth2
```

**WEBLATE\_TIME\_ZONE**

在 Weblate 中配置使用的時區，請參 *TIME\_ZONE*。

---

備：了更改 Docker 自己的時區，使用 TZ 環境變量。

---

示例:

```
environment:
  WEBLATE_TIME_ZONE: Europe/Prague
```

**WEBLATE\_ENABLE\_HTTPS**

讓 Weblate 假定在反向 HTTPS 代理後面操作，這使 Weblate 在電子郵件和 API 鏈接中使用 HTTPS，或者在 cookies 上設置安全標記。

**提示：** 可能的警告請參見 [ENABLE\\_HTTPS](#) 文檔。

**備註：** 這不會使 Weblate 容器接受 HTTPS 連接，您同樣需要配置它，例子請參見具有 [HTTPS](#) 支持的 *Docker* 容器。

**示例：**

```
environment:
  WEBLATE_ENABLE_HTTPS: 1
```

**也參考：**

[ENABLE\\_HTTPS](#) 設置正確的網站域名, [WEBLATE\\_SECURE\\_PROXY\\_SSL\\_HEADER](#)

**WEBLATE\_INTERLEDGER\_PAYMENT\_POINTERS**

在 4.12.1 版本新加入。

Lets Weblate set the *meta[name=monetization]* field in the head of the document. If multiple are specified, chooses one randomly.

**也參考：**

[INTERLEDGER\\_PAYMENT\\_POINTERS](#)

**WEBLATE\_IP\_PROXY\_HEADER**

讓 Weblate 從任何給定的 HTTP 標頭中取回 IP 地址。在使用 Weblate 容器之前的反向代理時使用它。

允許 [IP\\_BEHIND\\_REVERSE\\_PROXY](#) 設置 [IP\\_PROXY\\_HEADER](#)。

**備註：** 格式必須符合 Django 的要求。Django [transforms](#) 原始 HTTP 標頭如下命名：

- 將所有字符裝大寫
- 用下劃替任何連字符
- 前面添加 HTTP\_ 前綴字

所以 X-Forwarded-For 將被映射到 HTTP\_X\_FORWARDED\_FOR。

**示例：**

```
environment:
  WEBLATE_IP_PROXY_HEADER: HTTP_X_FORWARDED_FOR
```

**WEBLATE\_SECURE\_PROXY\_SSL\_HEADER**

代表 HTTP 標頭/值的組合的元組，用於表達請求，這樣的元組是安全的。當 Weblate 在進行終止 SSL 的反向代理之後運行時，這是需要的，終止 SSL 不通過標準 HTTPS 標頭。

**示例：**

```
environment:
  WEBLATE_SECURE_PROXY_SSL_HEADER: HTTP_X_FORWARDED_PROTO,https
```

**也參考：**

[SECURE\\_PROXY\\_SSL\\_HEADER](#)

**WEBLATE\_REQUIRE\_LOGIN**

用 `WEBLATE_REQUIRE_LOGIN` 而在整個 Weblate 上限制認證。

示例：

```
environment:
  WEBLATE_REQUIRE_LOGIN: 1
```

**WEBLATE\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS****WEBLATE\_ADD\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS****WEBLATE\_REMOVE\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS**

使用 `WEBLATE_LOGIN_REQUIRED_URLS_EXCEPTIONS` 來在整個 Weblate 安裝所需的身份驗證添加 URL 例外。

可以替換整個設置，或者使用 `ADD` 和 `REMOVE` 變量修改默認值。

**WEBLATE\_GOOGLE\_ANALYTICS\_ID**

通過 `WEBLATE_GOOGLE_ANALYTICS_ID` 來配置用於 Google Analytics 的 ID。

**WEBLATE\_GITHUB\_USERNAME****WEBLATE\_GITHUB\_TOKEN****WEBLATE\_GITHUB\_HOST**

Configures GitHub pull-requests integration by changing `WEBLATE_GITHUB_CREDENTIALS` (if `WEBLATE_GITHUB_HOST` is set), or `WEBLATE_GITHUB_USERNAME` and `WEBLATE_GITHUB_TOKEN`.

也參考：

*GitHub pull requests*

**WEBLATE\_GITLAB\_USERNAME****WEBLATE\_GITLAB\_TOKEN****WEBLATE\_GITLAB\_HOST**

Configures GitLab merge-requests integration by changing `WEBLATE_GITLAB_CREDENTIALS` (if `WEBLATE_GITLAB_HOST` is set), or `WEBLATE_GITLAB_USERNAME` and `WEBLATE_GITLAB_TOKEN`.

也參考：

*GitLab merge requests*

**WEBLATE\_GITEA\_USERNAME****WEBLATE\_GITEA\_TOKEN****WEBLATE\_GITEA\_HOST**

Configures Gitea pull-requests integration by changing `WEBLATE_GITEA_CREDENTIALS` (if `WEBLATE_GITEA_HOST` is set), or `WEBLATE_GITEA_USERNAME` and `WEBLATE_GITEA_TOKEN`.

也參考：

*Gitea pull requests*

**WEBLATE\_PAGURE\_USERNAME****WEBLATE\_PAGURE\_TOKEN****WEBLATE\_PAGURE\_HOST**

Configures Pagure merge-requests integration by changing `WEBLATE_PAGURE_CREDENTIALS` (if `WEBLATE_PAGURE_HOST` is set), or `WEBLATE_PAGURE_USERNAME` and `WEBLATE_PAGURE_TOKEN`.

也參考：

*Pagure merge requests*

**WEBLATE\_DEFAULT\_PULL\_MESSAGE**

Configures the default title and message for pull requests via API by changing *DEFAULT\_PULL\_MESSAGE*

也參考:

*DEFAULT\_PULL\_MESSAGE*

**WEBLATE\_SIMPLIFY\_LANGUAGES**

配置語言簡化策略，請參見*SIMPLIFY\_LANGUAGES*。

**WEBLATE\_DEFAULT\_ACCESS\_CONTROL**

F 新項目配置默認的存取控制，請參見*DEFAULT\_ACCESS\_CONTROL*。

**WEBLATE\_DEFAULT\_RESTRICTED\_COMPONENT**

F 新組件的受限制的訪問 配置默認值，請參見*DEFAULT\_RESTRICTED\_COMPONENT*。

**WEBLATE\_DEFAULT\_TRANSLATION\_PROPAGATION**

F 新組件的允許翻譯再用 配置默認值，請參見*DEFAULT\_TRANSLATION\_PROPAGATION*。

**WEBLATE\_DEFAULT\_COMMITER\_EMAIL**

配置*DEFAULT\_COMMITER\_EMAIL*。

**WEBLATE\_DEFAULT\_COMMITER\_NAME**

配置*DEFAULT\_COMMITER\_NAME*。

**WEBLATE\_DEFAULT\_SHARED\_TM**

配置*DEFAULT\_SHARED\_TM*。

**WEBLATE\_AKISMET\_API\_KEY**

配置 Akismet API 密鑰，請參見*AKISMET\_API\_KEY*。

**WEBLATE\_GPG\_IDENTITY**

配置提交的 GPG 簽名，請參見*WEBLATE\_GPG\_IDENTITY*。

也參考:

簽署 *GnuPG* 的 *Git* 承諾

**WEBLATE\_URL\_PREFIX**

配置 Weblate 運行的 URL 前綴，請參見*URL\_PREFIX*。

**WEBLATE\_SILENCED\_SYSTEM\_CHECKS**

配置您不想要顯示的檢查，請參見 *SILENCED\_SYSTEM\_CHECKS*。

**WEBLATE\_CSP\_SCRIPT\_SRC****WEBLATE\_CSP\_IMG\_SRC****WEBLATE\_CSP\_CONNECT\_SRC****WEBLATE\_CSP\_STYLE\_SRC****WEBLATE\_CSP\_FONT\_SRC**

允許定制 Content-Security-Policy HTTP 標頭。

也參考:

: 參考: “CSP”, : 設置: “CSP\_Script\_src”,

**WEBLATE\_LICENSE\_FILTER**

配置*LICENSE\_FILTER*。

**WEBLATE\_LICENSE\_REQUIRED**

配置*LICENSE\_REQUIRED*

**WEBLATE\_WEBSITE\_REQUIRED**

配置 *WEBSITE\_REQUIRED*

**WEBLATE\_HIDE\_VERSION**

配置 *HIDE\_VERSION*。

**WEBLATE\_BASIC\_LANGUAGES**

配置 *BASIC\_LANGUAGES*。

**WEBLATE\_DEFAULT\_AUTO\_WATCH**

配置 *DEFAULT\_AUTO\_WATCH*。

**WEBLATE\_RATELIMIT\_ATTEMPTS**

**WEBLATE\_RATELIMIT\_LOCKOUT**

**WEBLATE\_RATELIMIT\_WINDOW**

在 4.6 版本新加入。

Configures rate limiter.

---

**提示：** 您可以設置任何速率限制器範圍的配置。在此設置的任何設置添加 “Web21”：ref：速率限制。

---

**也參考：**

頻次限制, *RATELIMIT\_ATTEMPTS*, *RATELIMIT\_WINDOW*, *RATELIMIT\_LOCKOUT*

**WEBLATE\_API\_RATELIMIT\_ANON**

**WEBLATE\_API\_RATELIMIT\_USER**

在 4.11 版本新加入。

Configures API rate limiting. Defaults to 100/day for anonymous and 5000/hour for authenticated users.

**也參考：**

*API* 頻次限制

**WEBLATE\_ENABLE\_HOOKS**

在 4.13 版本新加入。

Configures *ENABLE\_HOOKS*。

**WEBLATE\_ENABLE\_AVATARS**

在 4.6.1 版本新加入。

配置 *ENABLE\_AVATARS*。

**WEBLATE\_AVATAR\_URL\_PREFIX**

在 4.15 版本新加入。

Configures *AVATAR\_URL\_PREFIX*。

**WEBLATE\_LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH**

在 4.9 版本新加入。

Configures *LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH*。

**WEBLATE\_SSH\_EXTRA\_ARGS**

在 4.9 版本新加入。

Configures *SSH\_EXTRA\_ARGS*。



**WEBLATE\_BORG\_EXTRA\_ARGS**

在 4.9 版本新加入。

Configures *BORG\_EXTRA\_ARGS*.

**WEBLATE\_ENABLE\_SHARING**

在 4.14.1 版本新加入。

Configures *ENABLE\_SHARING*.

**WEBLATE\_EXTRA\_HTML\_HEAD**

在 4.15 版本新加入。

Configures *EXTRA\_HTML\_HEAD*.

**WEBLATE\_PRIVATE\_COMMIT\_EMAIL\_TEMPLATE**

在 4.15 版本新加入。

Configures *PRIVATE\_COMMIT\_EMAIL\_TEMPLATE*.

**WEBLATE\_PRIVATE\_COMMIT\_EMAIL\_OPT\_IN**

在 4.15 版本新加入。

Configures *PRIVATE\_COMMIT\_EMAIL\_OPT\_IN*.

**Automatic suggestion settings**

在 4.13 版本變更: Automatic suggestion services are now configured in the user interface, see [配置自動建議](#).

The existing environment variables are imported during the migration to Weblate 4.13, but changing them will not have any further effect.

**身份驗證設置****LDAP****WEBLATE\_AUTH\_LDAP\_SERVER\_URI****WEBLATE\_AUTH\_LDAP\_USER\_DN\_TEMPLATE****WEBLATE\_AUTH\_LDAP\_USER\_ATTR\_MAP****WEBLATE\_AUTH\_LDAP\_BIND\_DN****WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD****WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD\_FILE**

Path to the file containing the LDAP server bind password.

也參考:

*WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD*

**WEBLATE\_AUTH\_LDAP\_CONNECTION\_OPTION\_REFERRALS****WEBLATE\_AUTH\_LDAP\_USER\_SEARCH****WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_FILTER****WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_UNION**

**WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_UNION\_DELIMITER**

LDAP authentication configuration.

直接綁定的例子：

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_USER_DN_TEMPLATE: uid=%(user)s,ou=People,dc=example,dc=net
  # map weblate 'full_name' to ldap 'name' and weblate 'email' attribute to
  ↪ 'mail' ldap attribute.
  # another example that can be used with OpenLDAP: 'full_name:cn,email:mail'
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
```

搜索與綁定的例子：

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN=Users,DC=example,DC=com
```

聯合搜索與綁定的例子：

```
environment:
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH_UNION: ou=users,dc=example,
  ↪ dc=com|ou=otherusers,dc=example,dc=com
```

針對活動目錄 F (Active Directory) 的搜索與綁定的例子：

```
environment:
  WEBLATE_AUTH_LDAP_BIND_DN: CN=ldap,CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_BIND_PASSWORD: password
  WEBLATE_AUTH_LDAP_SERVER_URI: ldap://ldap.example.org
  WEBLATE_AUTH_LDAP_CONNECTION_OPTION_REFERRALS: 0
  WEBLATE_AUTH_LDAP_USER_ATTR_MAP: full_name:name,email:mail
  WEBLATE_AUTH_LDAP_USER_SEARCH: CN=Users,DC=example,DC=com
  WEBLATE_AUTH_LDAP_USER_SEARCH_FILTER: (sAMAccountName=%(user)s)
```

也參考：

[LDAP 身份驗證](#)

**GitHub**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_SECRET**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_SECRET**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_NAME**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_SECRET**

WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_ID

允許 *GitHub* 驗證。

### Bitbucket

WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_OAUTH2\_KEY

WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_OAUTH2\_SECRET

WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_KEY

WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_SECRET

允許 *Butbucket* 驗證。

### Facebook

WEBLATE\_SOCIAL\_AUTH\_FACEBOOK\_KEY

WEBLATE\_SOCIAL\_AUTH\_FACEBOOK\_SECRET

允許 *Facebook* 驗證 2.

### Google

WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_KEY

WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_SECRET

WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_WHITELISTED\_DOMAINS

WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_WHITELISTED\_EMAILS

允許 *Google OAuth 2*。

### GitLab

WEBLATE\_SOCIAL\_AUTH\_GITLAB\_KEY

WEBLATE\_SOCIAL\_AUTH\_GITLAB\_SECRET

WEBLATE\_SOCIAL\_AUTH\_GITLAB\_API\_URL

允許 *GitLab* 驗證 2。

### Gitea

WEBLATE\_SOCIAL\_AUTH\_GITEA\_API\_URL

WEBLATE\_SOCIAL\_AUTH\_GITEA\_KEY

WEBLATE\_SOCIAL\_AUTH\_GITEA\_SECRET

Enables Gitea authentication.

## Azure Active Directory

**WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_OAUTH2\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_OAUTH2\_SECRET**

允許 Azure 活動目錄身份驗證，請參見微軟 *Azure Active Directory*。

## 帶有租支持的 Azure 活動目錄

**WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_SECRET**

**WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_TENANT\_ID**

允許帶有租支持的 Azure 活動目錄身份驗證，請參見微軟 *Azure Active Directory*。

## Keycloak

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_SECRET**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_PUBLIC\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_ALGORITHM**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_AUTHORIZATION\_URL**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_ACCESS\_TOKEN\_URL**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_TITLE**

**WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_IMAGE**

允許 Keycloak 身份驗證，請參見 [documentation](#)。

## Linux 銷售商

您可以通過將後面的變量設置任何值，使用 Linux 銷售商身份驗證服務來允許身份驗證。

**WEBLATE\_SOCIAL\_AUTH\_FEDORA**

**WEBLATE\_SOCIAL\_AUTH\_OPENSUSE**

**WEBLATE\_SOCIAL\_AUTH\_UBUNTU**

## Slack

**WEBLATE\_SOCIAL\_AUTH\_SLACK\_KEY**

**SOCIAL\_AUTH\_SLACK\_SECRET**

允許 Slack 身份驗證，請參見 *Slack*。

## OpenID Connect

在 4.13-1 版本新加入。

**WEBLATE\_SOCIAL\_AUTH\_OIDC\_OIDC\_ENDPOINT**

**WEBLATE\_SOCIAL\_AUTH\_OIDC\_KEY**

**WEBLATE\_SOCIAL\_AUTH\_OIDC\_SECRET**

**WEBLATE\_SOCIAL\_AUTH\_OIDC\_USERNAME\_KEY**

Configures generic OpenID Connect intergration.

也參考:

[OIDC \(OpenID Connect\)](#)

## SAML

在第一次啟動容器時自動生成自簽名的 SAML 密鑰。在您想要使用自己的密鑰的情況下，將證書和私鑰放置在 `file:/app/data/ssl/saml.crt` 和 `file:/app/data/ssl/saml.key` 中。

**WEBLATE\_SAML\_IDP\_ENTITY\_ID**

**WEBLATE\_SAML\_IDP\_URL**

**WEBLATE\_SAML\_IDP\_X509CERT**

**WEBLATE\_SAML\_IDP\_IMAGE**

**WEBLATE\_SAML\_IDP\_TITLE**

SAML 身份提供者設置，請參見 [SAML 身份驗證](#)。

## Other authentication settings

**WEBLATE\_NO\_EMAIL\_AUTH**

Disables e-mail authentication when set to any value. See [關閉密碼身份驗證](#).

## PostgreSQL database setup

數據庫由 `docker-compose.yml` 建立，所以這些設置影響 Weblate 和 PostgreSQL 容器。

也參考:

[Weblate 的數據庫設置](#)

**POSTGRES\_PASSWORD**

PostgreSQL 密碼。

**POSTGRES\_PASSWORD\_FILE**

包含 PostgreSQL 密碼的文件的路徑。用作 `postgres_password` 的替代品。

**POSTGRES\_USER**

PostgreSQL 用戶名。

**POSTGRES\_DATABASE**

PostgreSQL database name.

**POSTGRES\_HOST**

PostgreSQL 服務器主機名或 IP 地址。默認 `database`。

**POSTGRES\_PORT**

PostgreSQL 服務器端口。默認 F 無（使用默認值）。

**POSTGRES\_SSL\_MODE**

配置 PostgreSQL 如何處理 SSL 連接到服務器，可能的選項請參見 [SSL Mode Descriptions](#)

**POSTGRES\_ALTER\_ROLE**

在遷移過程中配置要改變的角色名稱，請參見配置 [Weblate](#) 來使用 *PostgreSQL*。

**POSTGRES\_CONN\_MAX\_AGE**

在 4.8.1 版本新加入。

The lifetime of a database connection, as an integer of seconds. Use 0 to close database connections at the end of each request (this is the default behavior).

Enabling connection persistence will typically, cause more open connection to the database. Please adjust your database configuration prior enabling.

配置的例子：

```
environment:
  POSTGRES_CONN_MAX_AGE: 3600
```

也參考：

[CONN\\_MAX\\_AGE](#), [Persistent connections](#)

**POSTGRES\_DISABLE\_SERVER\_SIDE\_CURSORS**

在 4.9.1 版本新加入。

Disable server side cursors in the database. This is necessary in some **pgbouncer** setups.

配置的例子：

```
environment:
  POSTGRES_DISABLE_SERVER_SIDE_CURSORS: 1
```

也參考：

[DISABLE\\_SERVER\\_SIDE\\_CURSORS](#), [Transaction pooling and server-side cursors](#)

## Database backup settings

也參考：

下載的數據用於備份

**WEBLATE\_DATABASE\_BACKUP**

使用 [DATABASE\\_BACKUP](#) 配置每日數據庫轉儲。默認 F plain。

## Caching server setup

Weblate F 烈推薦使用 Redis，在 Docker 中運行 Weblate 時您必須提供 Redis 事例。

也參考：

允許緩存

**REDIS\_HOST**

Redis 服務器主機名稱或 IP 地址。默認 F cache。

**REDIS\_PORT**

Redis 服務器端口。默認 F 6379。

**REDIS\_DB**

Redis 數據庫編號，默認 1。

**REDIS\_PASSWORD**

Redis 服務器密碼，默認不使用。

**REDIS\_PASSWORD\_FILE**

Path to the file containing the Redis server password.

也參考：

`REDIS_PASSWORD`

**REDIS\_TLS**

允許使用 SSL 進行 Redis 連接。

**REDIS\_VERIFY\_SSL**

可以用於禁止 Redis 連接的 SSL 身份認證。

**Email server setup**

要使外發電子郵件正常工作，您需要提供一個郵件服務器。

Example TLS configuration:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
```

Example SSL configuration:

```
environment:
  WEBLATE_EMAIL_HOST: smtp.example.com
  WEBLATE_EMAIL_PORT: 465
  WEBLATE_EMAIL_HOST_USER: user
  WEBLATE_EMAIL_HOST_PASSWORD: pass
  WEBLATE_EMAIL_USE_TLS: 0
  WEBLATE_EMAIL_USE_SSL: 1
```

也參考：

*Configuring outgoing e-mail*

**WEBLATE\_EMAIL\_HOST**

郵件服務器主機名或 IP 地址。

也參考：

`WEBLATE_EMAIL_PORT`, `WEBLATE_EMAIL_USE_SSL`, `WEBLATE_EMAIL_USE_TLS`,  
`EMAIL_HOST`

**WEBLATE\_EMAIL\_PORT**

郵件服務器端口，默認 25。

也參考：

`EMAIL_PORT`

**WEBLATE\_EMAIL\_HOST\_USER**

E-mail authentication user.

也參考：

`EMAIL_HOST_USER`

#### **WEBLATE\_EMAIL\_HOST\_PASSWORD**

E-mail authentication password.

也參考:

[EMAIL\\_HOST\\_PASSWORD](#)

#### **WEBLATE\_EMAIL\_HOST\_PASSWORD\_FILE**

Path to the file containing the e-mail authentication password.

也參考:

[WEBLATE\\_EMAIL\\_HOST\\_PASSWORD](#)

#### **WEBLATE\_EMAIL\_USE\_SSL**

與 SMTP 服務器通信時是否使用隱式 TLS（安全）連接。在大多數電子郵件文檔中，這種 TLS 連接類型稱爲 SSL。通常在端口 465 上使用。如果遇到問題，請參閱顯式 TLS 設置 [WEBLATE\\_EMAIL\\_USE\\_TLS](#)。

在 4.11 版本變更: The SSL/TLS support is automatically enabled based on the [WEBLATE\\_EMAIL\\_PORT](#).

也參考:

[WEBLATE\\_EMAIL\\_PORT](#), [WEBLATE\\_EMAIL\\_USE\\_TLS](#), [EMAIL\\_USE\\_SSL](#)

#### **WEBLATE\_EMAIL\_USE\_TLS**

與 SMTP 服務器通訊時是否使用 TLS（安全）連接。這用於顯式 TLS 連接，通常在端口 587 或 25 上。如果您遇到起起的連接，請參見隱式 TLS 設置 [WEBLATE\\_EMAIL\\_USE\\_SSL](#)。

在 4.11 版本變更: The SSL/TLS support is automatically enabled based on the [WEBLATE\\_EMAIL\\_PORT](#).

也參考:

[WEBLATE\\_EMAIL\\_PORT](#), [WEBLATE\\_EMAIL\\_USE\\_SSL](#), [EMAIL\\_USE\\_TLS](#)

#### **WEBLATE\_EMAIL\_BACKEND**

將 Django 後端配置用於發送電子郵件。

也參考:

[Configure e-mail sending](#), [EMAIL\\_BACKEND](#)

#### **WEBLATE\_AUTO\_UPDATE**

Configures if and how Weblate should update repositories.

也參考:

[AUTO\\_UPDATE](#)

---

備註: 這是布林值的設定（使用 "true" 或 "false"）。

---

### Site integration

#### **WEBLATE\_GET\_HELP\_URL**

配置 [GET\\_HELP\\_URL](#).

#### **WEBLATE\_STATUS\_URL**

配置 [STATUS\\_URL](#).

#### **WEBLATE\_LEGAL\_URL**

配置: setting: [LEGAL\\_URL](#).

#### **WEBLATE\_PRIVACY\_URL**

Configures [PRIVACY\\_URL](#).



## 錯誤報告

推薦從安裝中系統地收集錯誤，請參見 [Collecting error reports](#)。

要用對 Rollbar 的支持，請進行以下設置：

### ROLLBAR\_KEY

您的 Rollbar 發布服務器訪問令牌。

### ROLLBAR\_ENVIRONMENT

您的 Rollbar 環境，默認 `production`。

要用對 Sentry 的支持，請進行以下設置：

### SENTRY\_DSN

Your Sentry DSN.

### SENTRY\_ENVIRONMENT

您的 Sentry 環境（可選）。

## Localization CDN

### WEBLATE\_LOCALIZE\_CDN\_URL

### WEBLATE\_LOCALIZE\_CDN\_PATH

在 4.2.1 版本新加入。

:ref:‘addon-weblate.cdn.cdnjs’的配置。

`WEBLATE_LOCALIZE_CDN_PATH` 是容器的路徑。它應該存儲在持久卷上，而不能存儲在瞬態存儲器中。

一種可能性是存儲在 Weblate 數據目錄中：

```
environment:
  WEBLATE_LOCALIZE_CDN_URL: https://cdn.example.com/
  WEBLATE_LOCALIZE_CDN_PATH: /app/data/l10n-cdn
```

備：您負責設置 Weblate 生成的文件的服務，它只在配置的位置存儲文件。

### 也參考：

weblate-cdn, `LOCALIZE_CDN_URL`, `LOCALIZE_CDN_PATH`

## Changing enabled apps, checks, add-ons or autofixes

在 3.8-5 版本新加入。

The built-in configuration of enabled checks, add-ons or autofixes can be adjusted by the following variables:

### WEBLATE\_ADD\_APPS

### WEBLATE\_REMOVE\_APPS

### WEBLATE\_ADD\_CHECK

### WEBLATE\_REMOVE\_CHECK

### WEBLATE\_ADD\_AUTOFIX

**WEBLATE\_REMOVE\_AUTOFIX**

**WEBLATE\_ADD\_ADDONS**

**WEBLATE\_REMOVE\_ADDONS**

示例:

```
environment:
  WEBLATE_REMOVE_AUTOFIX: weblate.trans.autofixes.whitespace.
  ↪SameBookendingWhitespace
  WEBLATE_ADD_ADDONS: customize.addons.MyAddon,customize.addons.OtherAddon
```

也參考:

*CHECK\_LIST*, *AUTOFIX\_LIST*, *WEBLATE\_ADDONS*, *INSTALLED\_APPS*

## Container settings

**WEBLATE\_WORKERS**

在 4.6.1 版本新加入。

在容器中運行的工人進程基數。未設置時，基於可用的 CPU 核心數，在容器啟動時自動確定。

It is used to determine *CELERY\_MAIN\_OPTIONS*, *CELERY\_NOTIFY\_OPTIONS*, *CELERY\_MEMORY\_OPTIONS*, *CELERY\_TRANSLATE\_OPTIONS*, *CELERY\_BACKUP\_OPTIONS*, *CELERY\_BEAT\_OPTIONS*, and *WEB\_WORKERS*. You can use these settings to fine-tune.

**CELERY\_MAIN\_OPTIONS**

**CELERY\_NOTIFY\_OPTIONS**

**CELERY\_MEMORY\_OPTIONS**

**CELERY\_TRANSLATE\_OPTIONS**

**CELERY\_BACKUP\_OPTIONS**

**CELERY\_BEAT\_OPTIONS**

這些變量允許您調整 Celery worker 選項。它可以用於調整並發性 (`--concurrency 16`)，或使用不同的池實現 (`--pool=gevent`)。

默認情況下，開發工作人員的數量基於：envvar: *weblate\_workers*。

示例:

```
environment:
  CELERY_MAIN_OPTIONS: --concurrency 16
```

也參考:

: Doc: “Celery Worker” 選項 <Celery: Reference / Celery.Bin.Worker>, : Ref: “芹菜”

**WEB\_WORKERS**

配置應該執行多少個 uWSGI workers。

It defaults to *WEBLATE\_WORKERS*.

示例:

```
environment:
  WEB_WORKERS: 32
```

**WEBLATE\_SERVICE**

定義應在容器執行哪些服務，使用對象 *Scaling horizontally*。

Following services are defined:

**celery-beat**

Celery 任務調度程序，只能運行一個實例。該容器也負責數據庫結構遷移，且應該在其他之前動。

**celery-backup**

凱利工人備份，只能運行一個例子。

**celery-celery**

Generic Celery worker.

**celery-memory**

Translation memory Celery worker.

**celery-notify**

Notifications Celery worker.

**celery-translate**

Automatic translation Celery worker.

**web**

Web 服務器。

**Docker container volumes**

Weblate 容器導出了兩個卷（數據和緩存）。其他服務容器（PostgreSQL 或 Redis）也具有其數據卷，但本文檔未涵蓋這些數據卷。

數據卷用於存儲 Weblate 持久數據（例如克隆的倉庫）或自定義 Weblate 安裝。

Docker 卷在主機系統上的位置取於您的 Docker 配置，但通常存儲在 `/var/lib/docker/volumes/weblate-docker_weblate-data/_data/` 中。（該路徑由你的 `docker-compose` 目的名稱、容器和卷的名稱組成）。在容器中，它載 `/app/data`。

緩存卷已安裝：file: `/app/cache`，用於存儲態文件。它的容在集裝箱動時重新創建，且可以使用截至 “TMPFS”（如 “TMPFS”）安裝卷。

When creating the volumes manually, the directories should be owned by UID 1000 as that is user used inside the container.

**也參考：**

Docker 卷文檔

**Configuration beyond environment variables**

*Docker environment variables* are intended to expose most *configuration settings* of relevance for Weblate installations.

If you find a setting that is not exposed as an environment variable, and you believe that it should be, feel free to *ask for it to be exposed in a future version of Weblate*.

If you need to modify a setting that is not exposed as a Docker environment variable, you can still do so, either *from the data volume* or *extending the Docker image*.

**也參考：**

定制 Weblate

## Overriding settings from the data volume

You can create a file at `/app/data/settings-override.py`, i.e. at the root of the *data volume*, to extend or override settings defined through environment variables.

## Overriding settings by extending the Docker image

To override settings at the Docker image level instead of from the data volume:

1. *Create a custom Python package.*
2. Add a module to your package that imports all settings from `weblate.settings_docker`.

For example, within the example package structure defined at [建立 Python 模塊](#), you could create a file at `weblate_customization/weblate_customization/settings.py` with the following initial code:

```
from weblate.settings_docker import *
```

3. Create a custom Dockerfile that inherits from the official Weblate Docker image, and then installs your package and points the `DJANGO_SETTINGS_MODULE` environment variable to your settings module:

```
FROM weblate/weblate

USER root

COPY weblate_customization /usr/src/weblate_customization
RUN pip install --no-cache-dir /usr/src/weblate_customization
ENV DJANGO_SETTINGS_MODULE=weblate_customization.settings

USER 1000
```

4. Instead of using the official Weblate Docker image, build a custom image from this Dockerfile file.

There is *no clean way* to do this with `docker-compose.override.yml`. You *could* add `build: .` to the `weblate` node in that file, but then your custom image will be tagged as `weblate/weblate` in your system, which could be problematic.

So, instead of using the `docker-compose.yml` straight from the [official repository](#), unmodified, and extending it through `docker-compose.override.yml`, you may want to make a copy of the official `docker-compose.yml` file, and edit your copy to replace `image: weblate/weblate` with `build: ..`

See the [Compose file build reference](#) for details on building images from source when using `docker-compose`.

5. Extend your custom settings module to define or redefine settings.

You can define settings before or after the import statement above to determine which settings take precedence. Settings defined before the import statement can be overridden by environment variables and setting overrides defined in the data volume. Setting defined after the import statement cannot be overridden.

You can also go further. For example, you can reproduce some of the things that `weblate.docker_settings` *does*, such as exposing settings as environment variables, or allow overriding settings from Python files in the data volume.

## 替換 logo 和其它態文件

在 3.8-5 版本新加入。

Weblate 附帶的態文件可以通過放置到 `/app/data/python/customize/static` 中來覆蓋（請參見 [Docker container volumes](#)）。例如，創建 `/app/data/python/customize/static/favicon.ico` 將替換 favicon。

**提示：** 在容器啟動時，這些文件被复制到相應的位置，因此需要在更改卷的內容後重新啟動 Weblate。

This approach can be also used to override Weblate templates. For example 法律 documents can be placed into `/app/data/python/customize/templates/legal/documents`.

或者，您也可以包括自己的模塊（請參見 [定制 Weblate](#)），將其作為單獨的添加到 Docker 容器中，例如：

```
weblate:
  volumes:
    - weblate-data:/app/data
    - ./weblate_customization/weblate_customization:/app/data/python/weblate_
      ↪ customization
  environment:
    WEBLATE_ADD_APPS: weblate_customization
```

## Configuring PostgreSQL server

The PostgreSQL container uses default PostgreSQL configuration and it won't effectively utilize your CPU cores or memory. It is recommended to customize the configuration to improve the performance.

The configuration can be adjusted as described in *Database Configuration* at [https://hub.docker.com/\\_/postgres](https://hub.docker.com/_/postgres). The configuration matching your environment can be generated using <https://pgtune.leopard.in.ua/>.

## Container internals

The container is using **supervisor** to start individual services. In case of *Scaling horizontally*, it only starts single service in a container.

To check the services status use:

```
docker-compose exec --user weblate weblate supervisorctl status
```

There are individual services for each Celery queue (see [使用 Celery 的後台任務](#) for details). You can stop processing some tasks by stopping the appropriate worker:

```
docker-compose exec --user weblate weblate supervisorctl stop celery-translate
```

## 在 Debian 和 Ubuntu 上安裝

### 硬件要求

Weblate 應該可以在任何現代硬件上正常運行，以下是在單個主機（Weblate，數據庫和 Web 服務器）上運行 Weblate 所需的最低配置：

- 2 GB 的內存
- 2 CPU cores
- 1 GB 的存儲空間

內存越多越好——用於所有級別的緩存（文件系統，數據庫和 Weblate）。

許多應用會增加所需的 CPU 核數量。對於數百個翻譯組件，推薦至少有 4 GB 的內存。

典型的數據庫存儲用量大約每 1 百萬單詞 300 MB。克隆倉庫所需的存儲空間會變化，但 Weblate 試圖通過淺克隆將其大小最小化。

---

**備註：** 根據 Weblate 中管理的翻譯大小，安裝 Weblate 的實際要求差異很大。

---

## 安裝

### 系統要求

安裝所需的依賴包，來建立 Python 模塊（參見軟件要求）：

```
apt install -y \
    libxml2-dev libxslt-dev libfreetype6-dev libjpeg-dev libz-dev libyaml-dev \
    libffi-dev libcairo-dev gir1.2-pango-1.0 libgirepository1.0-dev \
    libacl1-dev libssl-dev libpq-dev libjpeg-dev build-essential \
    python3-gdbm python3-dev python3-pip python3-virtualenv virtualenv git
```

根據您想要使用的特性來安裝想要的可選依賴包（參見可選依賴性）：

```
apt install -y \
    tesseract-ocr libtesseract-dev liblibleptonica-dev \
    libldap2-dev libldap-common libsasl2-dev \
    libxmlsec1-dev
```

可選地安裝生服務器運行所需要的軟件，參見運行服務器、Weblate 的數據庫設置、使用 Celery 的後台任務。根據於您的安裝所的空間，您會想要在特定的服務器上運行這些組件。

本地安裝的使用說明：

```
# Web server option 1: NGINX and uWSGI
apt install -y nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
apt install -y apache2 libapache2-mod-wsgi-py3

# Caching backend: Redis
apt install -y redis-server

# Database server: PostgreSQL
apt install -y postgresql postgresql-contrib

# SMTP server
apt install -y exim4
```

## Python 模塊

**提示：** 我們使用 `virtualenv` 在與您的系統隔開的環境安裝 Weblate。如果您不熟悉，查看 [virtualenv User Guide](#)。

1. F Weblate 新建 `virtualenv`：

```
virtualenv ~/weblate-env
```

2. F Weblate 激活 `virtualevn`：

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check [可選依賴性](#) for fine-tuning of optional dependencies.

**備 F：** On some Linux distributions running Weblate fails with `libffi` error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi
→ library seen at runtime is different from the 'ffi.h' file seen at compile-
→ time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

## 配置 Weblate

**備 F：** The following assumes the `virtualenv` used by Weblate is activated (by executing `. ~/weblate-env/bin/activate`). If not, specify the full path to the **weblate** command as `~/weblate-env/bin/weblate`.

1. 將文件 `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` F F F: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`。
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [調整配置](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Weblate 的數據庫設置](#) for a production-ready setup):

```
weblate migrate
```

4. Create an account for the administrator user and copy its password to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect the static files for your web server (see [運行服務器](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress the JavaScript and CSS files (optional, see *Compressing client assets*):

```
weblate compress
```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. 使用 *Celery* 的後台任務 has more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

8. Start the development server (運行服務器 details a production setup):

```
weblate runserver
```

## 安裝後

配置，您的 Weblate 服務器現在運行了，您可以使用它來啟動。

- 您可以在 `http://localhost:8000/` 訪問 Weblate。
- Sign in with admin credentials obtained during installation or register with new users.
- 現在，您可以在 Weblate virtualenv 活動時使用 **weblate** 命令來運行 Weblate 命令。
- 您可以使用 Ctrl+C 來停止測試的服務器。
- Review potential issues with your installation either on `/manage/performance/` URL (see 管理介面) or using **weblate check --deploy**, see 生 設置.

## 添加翻譯

1. 打開管理界面 (`http://localhost:8000/create/project/`)，新建您想要翻譯的項目。更多細節請參見 項目配置。

這所有需要您指定的只是項目名稱及其網站。

2. 新建組件，它是翻譯的真實對象——它執行版本控制系統（VCS）倉庫，用於選擇那個文件被翻譯。更多細節請參見 組件配置。

The important fields here are: 組件名, 源代碼儲存庫, and 文件掩碼 for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, 安卓字符串資源, 蘋果 iOS 字符串, *Java* 屬性, *Stringsdict format* or *Fluent format*, see 支持的文件格式 for more details.

3. 一旦完成上面的工作（根據您的版本控制系統 VCS 倉庫的大小，以及需要翻譯的信息數量，這可能是個漫長的過程），您就可以開始翻譯了。

## 在 SUSE 和 openSUSE 上安裝

### 硬件要求

Weblate 應該可以在任何現代硬件上正常運行，以下是在單個主機（Weblate，數據庫和 Web 服務器）上運行 Weblate 所需的最低配置：

- 2 GB 的內存
- 2 CPU cores
- 1 GB 的存儲空間



記憶體越多越好——用於所有級的緩存（文件系統，數據庫和 Weblate）。

許多應用會增加所需的 CPU 核數量。對於數百個翻譯組件，推薦至少有 4 GB 的記憶體。

典型的數據庫存儲用量大約每 1 百萬單詞 300 MB。克隆倉庫所需的存儲空間會變化，但 Weblate 試圖通過淺克隆將其大小最小化。

---

**備註：** 根據 Weblate 中管理的翻譯大小，安裝 Weblate 的實際要求差異很大。

---

## 安裝

### 系統要求

安裝所需的依賴包，來建立 Python 模塊（參見軟件要求）：

```
zypper install \
    libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
    libyaml-devel libffi-devel cairo-devel pango-devel \
    gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
    python3-devel git
```

根據您想要使用的特性來安裝想要的可選依賴包（參見可選依賴性）：

```
zypper install tesseract-ocr tesseract-devel leptonica-devel
zypper install libldap2-devel libsasl2-devel
zypper install libxmlsec1-devel
```

可選地安裝生服務器運行所需要的軟件，參見運行服務器、Weblate 的數據庫設置、使用 Celery 的後台任務。根據於您的安裝所的空間，您會想要在特定的服務器上運行這些組件。

本地安裝的使用說明：

```
# Web server option 1: NGINX and uWSGI
zypper install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
zypper install apache2 apache2-mod_wsgi

# Caching backend: Redis
zypper install redis-server

# Database server: PostgreSQL
zypper install postgresql postgresql-contrib

# SMTP server
zypper install postfix
```

## Python 模塊

---

**提示：** 我們使用 virtualenv 在與您的系統隔開的環境安裝 Weblate。如果您不熟悉，查看 [virtualenv User Guide](#)。

---

1. 新建 Weblate virtualenv：

```
virtualenv ~/weblate-env
```

2. F Weblate 激活 `virtualenv` :

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check 可選依賴性 for fine-tuning of optional dependencies.

備 F: On some Linux distributions running Weblate fails with `libffi` error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi
→library seen at runtime is different from the 'ffi.h' file seen at compile-
→time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

## 配置 Weblate

備 F: The following assumes the `virtualenv` used by Weblate is activated (by executing `. ~/weblate-env/bin/activate`). If not, specify the full path to the **weblate** command as `~/weblate-env/bin/weblate`.

1. 將文件: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` F F F: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`。
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see 調整配置.
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Weblate 的數據庫設置](#) for a production-ready setup):

```
weblate migrate
```

4. Create an account for the administrator user and copy its password to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect the static files for your web server (see [運行服務器](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress the JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. 使用 [Celery 的後台任務](#) has more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

8. Start the development server ([運行服務器](#) details a production setup):

```
weblate runserver
```

## 安裝後

配置，您的 Weblate 服務器現在運行了，您可以使用它來啟動。

- 您可以在 `http://localhost:8000/` 訪問 Weblate。
- Sign in with admin credentials obtained during installation or register with new users.
- 現在，您可以在 Weblate virtualenv 活動時使用 **weblate** 命令來運行 Weblate 命令。
- 您可以使用 Ctrl+C 來停止測試的服務器。
- Review potential issues with your installation either on `/manage/performance/` URL (see [管理介面](#)) or using **weblate check --deploy**, see [生產設置](#).

## 添加翻譯

1. 打開管理界面 (`http://localhost:8000/create/project/`)，新建您想要翻譯的項目。更多細節請參見 [項目配置](#)。  
這需要您指定的只是項目名稱及其網站。
2. 新建組件，它是翻譯的真實對象——它執行版本控制系統（VCS）倉庫，用於選擇那個文件被翻譯。更多細節請參見 [組件配置](#)。  
The important fields here are: 組件名義, 源代碼儲存庫, and 文件掩碼 for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, 安卓字符串資源, 蘋果 iOS 字符串, *Java* 屬性, *Stringsdict format* or *Fluent format*, see [支持的文件格式](#) for more details.
3. 一旦完成上面的工作（根據您的版本控制系統 VCS 倉庫的大小，以及需要翻譯的信息數量，這可能是個漫長的過程），您就可以開始翻譯了。

## 在 Redhat、Fedora 和 CentOS 上安裝

### 硬件要求

Weblate 應該可以在任何現代硬件上正常運行，以下是在單個主機（Weblate，數據庫和 Web 服務器）上運行 Weblate 所需的最低配置：

- 2 GB 的內存
- 2 CPU cores
- 1 GB 的存儲空間

內存越多越好——用於所有級別的緩存（文件系統，數據庫和 Weblate）。

許多應用會增加所需的 CPU 核數量。對於數百個翻譯組件，推薦至少有 4 GB 的內存。

典型的數據庫存儲用量大約每 1 百萬單詞 300 MB。克隆倉庫所需的存儲空間會變化，但 Weblate 試圖通過淺克隆將其大小最小化。

---

**備註：** 根據 Weblate 中管理的翻譯大小，安裝 Weblate 的實際要求差異很大。

---

## 安裝

### 系統要求

安裝所需的依賴包，來建立 Python 模塊（參見軟件要求）：

```
dnf install \
    libxslt-devel libxml2-devel freetype-devel libjpeg-devel zlib-devel \
    libyaml-devel libffi-devel cairo-devel pango-devel \
    gobject-introspection-devel libacl-devel python3-pip python3-virtualenv \
    python3-devel git
```

根據您想要使用的特性來安裝想要的可選依賴包（參見可選依賴性）：

```
dnf install tesseract-langpack-eng tesseract-devel leptonica-devel
dnf install libldap2-devel libsasl2-devel
dnf install libxmlsec1-devel
```

可選地安裝生 F 服務器運行所需要的軟件，參見運行服務器、Weblate 的數據庫設置、使用 Celery 的後台任務。根據於您的安裝所 F 的空間，您會想要在特定的服務器上運行這些組件。

本地安裝的使用 F 明：

```
# Web server option 1: NGINX and uWSGI
dnf install nginx uwsgi uwsgi-plugin-python3

# Web server option 2: Apache with ``mod_wsgi``
dnf install apache2 apache2-mod_wsgi

# Caching backend: Redis
dnf install redis

# Database server: PostgreSQL
dnf install postgresql postgresql-contrib

# SMTP server
dnf install postfix
```

## Python 模塊

**提示：** 我們使用 virtualenv 在與您的系統隔開的環境安裝 Weblate。如果您不熟悉，查看 [virtualenv User Guide](#)。

1. F Weblate 新建 virtualenv：

```
virtualenv ~/weblate-env
```

2. F Weblate 激活 virtualevn：

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check 可選依賴性 for fine-tuning of optional dependencies.

備 F: On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi
→library seen at runtime is different from the 'ffi.h' file seen at compile-
→time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

## 配置 Weblate

備 F: The following assumes the virtualenv used by Weblate is activated (by executing `. ~/weblate-env/bin/activate`). If not, specify the full path to the **weblate** command as `~/weblate-env/bin/weblate`.

1. 將文件: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` F F F: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`。
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [調整配置](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Weblate 的數據庫設置](#) for a production-ready setup):

```
weblate migrate
```

4. Create an account for the administrator user and copy its password to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect the static files for your web server (see [運行服務器](#) and *Serving static files*):

```
weblate collectstatic
```

6. Compress the JavaScript and CSS files (optional, see *Compressing client assets*):

```
weblate compress
```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. 使用 *Celery* 的後台任務 has more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

8. Start the development server ([運行服務器](#) details a production setup):

```
weblate runserver
```

## 安裝後

配置，您的 Weblate 服務器現在運行了，您可以使用它來啟動。

- 您可以在 `http://localhost:8000/` 訪問 Weblate。
- Sign in with admin credentials obtained during installation or register with new users.
- 現在，您可以在 Weblate virtualenv 活動時使用 **weblate** 命令來運行 Weblate 命令。
- 您可以使用 Ctrl+C 來停止測試的服務器。
- Review potential issues with your installation either on `/manage/performance/` URL (see [管理介面](#)) or using **weblate check --deploy**, see [生成設置](#).

## 添加翻譯

1. 打開管理界面 (`http://localhost:8000/create/project/`)，新建您想要翻譯的項目。更多細節請參見 [項目配置](#)。

這所有需要您指定的只是項目名稱及其網站。

2. 新建組件，它是翻譯的真實對象——它執行版本控制系統（VCS）倉庫，用於選擇那個文件被翻譯。更多細節請參見 [組件配置](#)。

The important fields here are: 組件名義, 源代碼儲存庫, and 文件掩碼 for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, 安卓字符串資源, 蘋果 iOS 字符串, *Java* 屬性, *Stringsdict format* or *Fluent format*, see [支持的文件格式](#) for more details.

3. 一旦完成上面的工作（根據您的版本控制系統 VCS 倉庫的大小，以及需要翻譯的信息數量，這可能是個漫長的過程），您就可以開始翻譯了。

## Installing on macOS

### 硬件要求

Weblate 應該可以在任何現代硬件上正常運行，以下是在單個主機（Weblate，數據庫和 Web 服務器）上運行 Weblate 所需的最低配置：

- 2 GB 的內存
- 2 CPU cores
- 1 GB 的存儲空間

內存越多越好——用於所有級的緩存（文件系統，數據庫和 Weblate）。

許多內存會增加所需的 CPU 核數量。對於數百個翻譯組件，推薦至少有 4 GB 的內存。

典型的數據庫存儲用量大約每 1 百萬單詞 300 MB。克隆倉庫所需的存儲空間會變化，但 Weblate 試圖通過淺克隆將其大小最小化。

---

**備註：** 根據 Weblate 中管理的翻譯大小，安裝 Weblate 的實際要求差異很大。

---

## 安裝

### 系統要求

安裝所需的依賴包，來建立 Python 模塊（參見軟件要求）：

```
brew install python pango cairo gobject-introspection libffi glib libyaml
pip install virtualenv
```

確認 pip 能 找到 homebrew 提供的 libffi 版本——將在安裝編譯步驟中需要它。

```
export PKG_CONFIG_PATH="/usr/local/opt/libffi/lib/pkgconfig"
```

根據您想要使用的特性來安裝想要的可選依賴包（參見可選依賴性）：

```
brew install tesseract
```

可選地安裝生 服務器運行所需要的軟件，參見運行服務器、*Weblate* 的數據庫設置、使用 *Celery* 的後台任務。根據於您的安裝所 的空間，您會想要在特定的服務器上運行這些組件。

本地安裝的使用 明：

```
# Web server option 1: NGINX and uWSGI
brew install nginx uwsgi

# Web server option 2: Apache with ``mod_wsgi``
brew install httpd

# Caching backend: Redis
brew install redis

# Database server: PostgreSQL
brew install postgresql
```

### Python 模塊

**提示：** 我們使用 virtualenv 在與您的系統隔開的環境安裝 Weblate。如果您不熟悉，查看 [virtualenv User Guide](#)。

1. 新建 Weblate virtualenv：

```
virtualenv ~/weblate-env
```

2. Weblate 激活 virtualenv：

```
. ~/weblate-env/bin/activate
```

3. Install Weblate including all optional dependencies:

```
# Install Weblate with all optional dependencies
pip install "Weblate[all]"
```

Please check 可選依賴性 for fine-tuning of optional dependencies.

**備：** On some Linux distributions running Weblate fails with libffi error:

```
ffi_prep_closure(): bad user_data (it seems that the version of the libffi_
→library seen at runtime is different from the 'ffi.h' file seen at compile-
→time)
```

This is caused by incompatibility of binary packages distributed via PyPI with the distribution. To address this, you need to rebuild the package on your system:

```
pip install --force-reinstall --no-binary :all: cffi
```

## 配置 Weblate

備 F: The following assumes the virtualenv used by Weblate is activated (by executing `. ~/weblate-env/bin/activate`). If not, specify the full path to the **weblate** command as `~/weblate-env/bin/weblate`.

1. 將文件: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings_example.py` F F F: `file:~/weblate-env/lib/python3.9/site-packages/weblate/settings.py`。
2. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see [調整配置](#).
3. Create the database and its structure for Weblate (the example settings use PostgreSQL, check [Weblate 的數據庫設置](#) for a production-ready setup):

```
weblate migrate
```

4. Create an account for the administrator user and copy its password to the clipboard, and also save it for later use:

```
weblate createadmin
```

5. Collect the static files for your web server (see [運行服務器](#) and [Serving static files](#)):

```
weblate collectstatic
```

6. Compress the JavaScript and CSS files (optional, see [Compressing client assets](#)):

```
weblate compress
```

7. Start the Celery workers. This is not necessary for development purposes, but strongly recommended otherwise. 使用 [Celery 的後台任務](#) has more info:

```
~/weblate-env/lib/python3.9/site-packages/weblate/examples/celery start
```

8. Start the development server ([運行服務器](#) details a production setup):

```
weblate runserver
```



## 安裝後

配置，您的 Weblate 服務器現在運行了，您可以使用它來啟動。

- 您可以在 `http://localhost:8000/` 訪問 Weblate。
- Sign in with admin credentials obtained during installation or register with new users.
- 現在，您可以在 Weblate virtualenv 活動時使用 **weblate** 命令來運行 Weblate 命令。
- 您可以使用 Ctrl+C 來停止測試的服務器。
- Review potential issues with your installation either on `/manage/performance/` URL (see 管理介面) or using **weblate check --deploy**, see 生成設置。

## 添加翻譯

1. 打開管理界面 (`http://localhost:8000/create/project/`)，新建您想要翻譯的項目。更多細節請參見 項目配置。

這所有需要您指定的只是項目名稱及其網站。

2. 新建組件，它是翻譯的真實對象——它執行版本控制系統 (VCS) 倉庫，用於選擇那個文件被翻譯。更多細節請參見 組件配置。

The important fields here are: 組件名、源代碼儲存庫, and 文件掩碼 for finding translatable files. Weblate supports a wide range of formats including *GNU gettext*, 安卓字符串資源, 蘋果 iOS 字符串, *Java* 屬性, *Stringsdict format* or *Fluent format*, see 支持的文件格式 for more details.

3. 一旦完成上面的工作（根據您的版本控制系統 VCS 倉庫的大小，以及需要翻譯的信息數量，這可能是個漫長的過程），您就可以開始翻譯了。

## Installing from sources

1. Please follow the installation instructions for your system first up to installing Weblate:

- 在 *Debian* 和 *Ubuntu* 上安裝
- 在 *SUSE* 和 *openSUSE* 上安裝
- 在 *Redhat*、*Fedora* 和 *CentOS* 上安裝

2. 使用 Git 來抓取最新的 Weblate 資源（或下載 tarball 包將其解壓）：

```
git clone https://github.com/WeblateOrg/weblate.git weblate-src
```

另外，您可以發布檔案。從可以從我們的網站 <<https://weblate.org/>> 來下載。下載是加密簽名的，參見 *Verifying release signatures*。

3. 將當前的 Weblate 代碼安裝到 virtualenv 中：

```
. ~/weblate-env/bin/activate
pip install -e weblate-src
```

4. 將 `weblate/settings_example.py` 重命名為 `weblate/settings.py`。
5. Adjust the values in the new `settings.py` file to your liking. You will need to provide at least the database credentials and Django secret key, but you will want more changes for production setup, see 調整配置。
6. 新建 Weblate 使用的數據庫，參見 *Weblate 的數據庫設置*。
7. 建立 Django 表、態文件和初始數據（參見 填滿數據庫 和 填滿數據庫）：

```
weblate migrate
weblate collectstatic
weblate compress
```

備：無論任何時候更新倉庫時，都應該重這一步驟。

## Installing on OpenShift

使用 OpenShift Weblate 模板，您可以在幾秒鐘動運行您的個人 Weblate 實例。Weblate 的所有依賴項都已經包含在。PostgreSQL 被設置默認數據庫，且使用持久化卷聲明。

你可以在 <<https://github.com/WeblateOrg/openshift/>> 找到模板。

## 安裝

下面的示例假設您有一個正常運作的 OpenShift v3.x 環境，它已經安裝“oc”客戶端工具。請查看 OpenShift 文檔中的明。

：文件：*template.yml* 非常適合運行 OpenShift 中的所有組件。還有：文件：“模板 - 外部 PostgreSQL.yml” 不動 PostgreSQL 服務器，允許您配置外部 PostgreSQL 服務器。

## Web 控制台

從 ‘*template.yml*’ <<https://github.com/WeblateOrg/openshift/blob/main/template.yml>> 原始容，將它們導入你的項目，然後在 OpenShift web 控制台使用 “Create” 按鈕來新建你的應用。web 控制台將提示你模板使用的所有參數的值。

## CLI

了將 Weblate 模板上傳到你當前項目的模板庫中，使用後面的命令傳遞 *template.yml* 文件：

```
$ oc create -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
→template.yml \
  -n <PROJECT>
```

現在模板可以使用 CLI 的 web 控制台以供選擇。

## Parameters

模板的 *parameters* 部分列出了你可以覆蓋的參數。你可以通過使用後面的命令指定要使用的文件通過 CLI 列出它們：

```
$ oc process --parameters -f https://raw.githubusercontent.com/WeblateOrg/
→openshift/main/template.yml

# If the template is already uploaded
$ oc process --parameters -n <PROJECT> weblate
```

## Provisioning

還可以使用 CLI 來處理模板，使用生成的配置來立即新建對象。

```
$ oc process -f https://raw.githubusercontent.com/WeblateOrg/openshift/main/
→template.yml \
  -p APPLICATION_NAME=weblate \
  -p WEBLATE_VERSION=4.3.1-1 \
  -p WEBLATE_SITE_DOMAIN=weblate.app-openshift.example.com \
  -p POSTGRESQL_IMAGE=docker-registry.default.svc:5000/openshift/postgresql:9.6 \
  -p REDIS_IMAGE=docker-registry.default.svc:5000/openshift/redis:3.2 \
  | oc create -f
```

在成功地遷移部署特定的“WEBLATE\_SITE\_DOMAIN”參數後，Weblate 事件就應該可用了。

設置容器之後，您可以使用 WEBLATE\_ADMIN\_PASSWORD 中提供的密碼以管理員身份登入，或者如果未設置密碼，則使用首次啟動時生成的隨機密碼。

要重置管理員密碼，請在將“WEBLATE\_ADMIN\_PASSWORD”設置相應的“Secret”中的新密碼的情況下，重啟容器。

## Eliminate

```
$ oc delete all -l app=<APPLICATION_NAME>
$ oc delete configmap -l app= <APPLICATION_NAME>
$ oc delete secret -l app=<APPLICATION_NAME>
# ATTENTION! The following command is only optional and will permanently delete
→all of your data.
$ oc delete pvc -l app=<APPLICATION_NAME>

$ oc delete all -l app=weblate \
  && oc delete secret -l app=weblate \
  && oc delete configmap -l app=weblate \
  && oc delete pvc -l app=weblate
```

## 配置

通過處理模板，將新建各自的 ConfigMap，並且可以用於定制 Weblate 映像。ConfigMap 直接作環境變量載，並且在每次更改時觸發新的部署。對於更多配置選項，環境變量的完成列表請參見 *Docker environment variables* for full list of environment variables。

## 在 Kubernetes 上安裝

---

備註：本手冊尋找有 Kubernetes 經驗的貢獻者來詳細介紹安裝過程。

---

憑藉 Kubernetes Helm 圖表，您可以在幾秒鐘內運行您的個人 Weblate 實例。Weblate 的所有依賴項都已經包含在內。PostgreSQL 被設置為默認數據庫，並且使用持久化卷聲明。

您可以在 <<https://github.com/WeblateOrg/helm/>> 找到圖標，並且它可以在 <<https://artifacthub.io/packages/helm/weblate/weblate>> 顯示。

## 安裝

```
helm repo add weblate https://helm.weblate.org
helm install my-release weblate/weblate
```

## 配置

For further configuration options, see *Docker environment variables* for full list of environment variables.

根據你的設置和經驗，你選擇適當的安裝方法：

- *Installing using Docker*, 推薦用於生產安裝。
- Virtualenv 安裝，推薦用於開發設置：
  - 在 *Debian* 和 *Ubuntu* 上安裝
  - 在 *SUSE* 和 *openSUSE* 上安裝
  - 在 *Redhat*、*Fedora* 和 *CentOS* 上安裝
  - *Installing on macOS*
- *Installing from sources*, recommended for development.
- *Installing on OpenShift*
- 在 *Kubernetes* 上安裝

### 2.1.2 軟件要求

#### 操作系統

Weblate 已知運行在 Linux、FreeBSD 和 macOS 上。其他類 Unix 的系統也很可能支持運行。

Windows 不支持 Weblate。但仍然可能工作，願意接受補丁。

#### 其他服務

Weblate 其操作使用其他服務。至少需要後面的服務運行：

- PostgreSQL 數據庫服務器，請參見 *Weblate* 的數據庫設置。
- Redis 服務器，用於緩存和任務隊列，請參見使用 *Celery* 的後台任務。
- SMTP 服務器，用於發送電子郵件，請參見 *Configuring outgoing e-mail*。

#### Python 依賴性

Weblate 用 Python 編寫，且支持 Python 3.6 或更新版本。可以使用 pip 或你的發布包來安裝依賴性，完全列表可在 `requirements.txt` 中找到。

Most notable dependencies:

##### Django

[HTTPS://www.djangoproject.com/](https://www.djangoproject.com/)

##### Celery

<https://docs.celeryq.dev/>

##### Translate Toolkit (翻譯工具包)

[HTTPS://toolkit.translatehouse.org/](https://toolkit.translatehouse.org/)

**translation-finder**[HTTPS://GitHub.com/Web late org/translation-finder](https://github.com/Web-late-org/translation-finder)**Python Social Auth**[HTTPS://Python-social-AUTH.read the doc S.IO/](https://python-social-auth.readthedocs.io/)**Django REST Framework**[HTTPS://鳴鳴鳴.Django-rest-framework.org/](https://www.django-rest-framework.org/)**可選依賴性**

後面的模塊對 Weblate 的一些特性是必須的。可以在 `requirements-optional.txt` 中找到。

**Mercurial (optional for *Mercurial* repositories support)**[HTTPS://鳴鳴鳴.mercurial-三重門.org/](https://www.mercurial-scm.org/)**phply (optional for *PHP* 字串)**[HTTPS://GitHub.com/vi raptor/PHP 有](https://github.com/viraptor/PHP-ly)**tesseract (optional for OCR in 字符串的可見語境)**[HTTPS://GitHub.com/私人房子/特殊色如 OCR](https://github.com/tesseract-ocr/tesseract)**python-akismet (optional for 針對垃圾郵件的保護)**<https://github.com/Nekmo/python-akismet>**ruamel.yaml (對 *YAML files* 是可選的)**[HTTPS://GitHub.com/私人房子/特殊色如 OCR](https://github.com/ruamel/yaml)**Zeep (optional for *Microsoft Terminology*)**[HTTPS://GitHub.com/私人房子/特殊色如 OCR](https://github.com/zeep/zeep)**aeidon (對 *Subtitle files* 是可選的)**<https://pypi.org/project/aeidon/>**fluent.syntax (optional for *Fluent format*)**<https://projectfluent.org/>

提示: When installing using pip, you can directly specify desired features when installing:

```
pip install "Weblate[PHP,Fluent]"
```

Or you can install Weblate with all optional features:

```
pip install "Weblate[all]"
```

Or you can install Weblate without any optional features:

```
pip install Weblate
```

**Database backend dependencies**

Weblate 支持 PostgreSQL、MySQL 和 MariaDB 數據庫，更多細節請參見 *Weblate* 的數據庫設置 和後端文件。

## Other system requirements

後面的依賴性必須安裝在系統上：

**Git**

<https://git-scm.com/>

**Pango, Cairo and related header files and GObject introspection data**

<https://cairographics.org/>, <https://pango.gnome.org/>, 請參見 *Pango and Cairo*

**git-review** (對 Gerrit 支持是可選的)

<https://pypi.org/project/git-review/>

**git-svn** (對 Subversion 的支持是可選的)

<https://git-scm.com/docs/git-svn>

**tesseract** 及其數據 (對截屏 OCR 是可選的)

<https://github.com/tesseract-ocr/tesseract>

**licensee** (當新建組件時可選用與 除許可)

<https://github.com/licensee/licensee>

## 構建時的依賴

為了構建一些 *Python* 依賴性，你可能需要安裝其依賴庫。這取決於你如何安裝它們，因此請參考單獨包的文檔。如果你使用預編譯 wheels 使用 `pip` 安裝或使用分發包時，你不會需要那些。

## Pango and Cairo

在 3.7 版本變更。

Weblate 使用 Pango 和 Cairo 來提供位圖 widget (請參參見 *promotion*) 提供檢查 (請參見 *管理字型*)。為了適當地安裝 Python 綁定需要首先安裝系統庫的那些——Cairo 和 Pango 都是需要的，由此需要 Glib。所有那些需要與開發文件和 GObject 省數據一起安裝。

## 2.1.3 Verifying release signatures

Weblate 的發布由發布開發者通過密碼簽發。當前是 Michal Čihař。他的 PGP 密鑰指紋是：

```
63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

並且可以從 <https://keybase.io/nijel> 得到更多識別信息。

應該驗證簽名與下載的壓縮檔案匹配。這種方式可以確保你使用發布的相同編碼。還應該核實簽名的日期，確定下載了最新的版本。

每個壓縮檔案伴隨 .asc 文件在一起，它包含了需要使用的 PGP 簽名。一旦它們處於相同的文件夾，就可以驗證簽名了：

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:                using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Can't check signature: public key not found
```

正如你所看到的，GPG 抱怨它不知道公鑰。此時應該執行以下步驟之一：

- 使用 *wkd* 來下載密鑰：

```
$ gpg --auto-key-locate wkd --locate-keys michal@cihar.com
pub  rsa4096 2009-06-17 [SC]
    63CB1DF1EF12CF2AC0EE5A329C27B31342B7511D
uid          [ultimate] Michal Čihař <michal@cihar.com>
uid          [ultimate] Michal Čihař <nijel@debian.org>
uid          [ultimate] [jpeg image of size 8848]
uid          [ultimate] Michal Čihař (Braiiins) <michal.cihar@braiins.cz>
sub  rsa4096 2009-06-17 [E]
sub  rsa4096 2015-09-09 [S]
```

- 從 Michal's server 下載鑰匙鏈，然後將其導入：

```
$ gpg --import wmxth3chu9jfxdxywj1skpmhsj311mzm
```

- 從一個密鑰服務器下載並導入密鑰：

```
$ gpg --keyserver hkp://pgp.mit.edu --recv-keys 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: key 9C27B31342B7511D: "Michal Čihař <michal@cihar.com>" imported
gpg: Total number processed: 1
gpg:          unchanged: 1
```

這會將情況改善一點——在這點上可以驗證給定密鑰的簽名是正確的，但仍然不能相信密鑰中使用的名稱：

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Ne 3. března 2019, 16:43:15 CET
gpg:          using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:          aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:          aka "[jpeg image of size 8848]" [ultimate]
gpg:          aka "Michal Čihař (Braiiins) <michal.cihar@braiins.cz>" [ultimate]
gpg: WARNING: This key is not certified with a trusted signature!
gpg:          There is no indication that the signature belongs to the owner.
Primary key fingerprint: 63CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D
```

The problem here is that anybody could issue the key with this name. You need to ensure that the key is actually owned by the mentioned person. The GNU Privacy Handbook covers this topic in the chapter [Validating other keys on your public keyring](#). The most reliable method is to meet the developer in person and exchange key fingerprints, however you can also rely on the web of trust. This way you can trust the key transitively through signatures of others, who have met the developer in person.

一旦信任了密鑰，警告就不會發生：

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: assuming signed data in 'Weblate-3.5.tar.xz'
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:          using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: Good signature from "Michal Čihař <michal@cihar.com>" [ultimate]
gpg:          aka "Michal Čihař <nijel@debian.org>" [ultimate]
gpg:          aka "[jpeg image of size 8848]" [ultimate]
gpg:          aka "Michal Čihař (Braiiins) <michal.cihar@braiins.cz>" [ultimate]
```

如果簽名非法（壓縮的檔案被更改），那會得到清晰的錯誤，而無論密鑰是否可信：

```
$ gpg --verify Weblate-3.5.tar.xz.asc
gpg: Signature made Sun Mar  3 16:43:15 2019 CET
gpg:          using RSA key 87E673AF83F6C3A0C344C8C3F4AA229D4D58C245
gpg: BAD signature from "Michal Čihař <michal@cihar.com>" [ultimate]
```



### 2.1.4 文件系統權限

Weblate 進程需要能讀寫它保存數據的目錄 - setting: `DATA_DIR`。該目錄下的所有文件都應該由運行所有 Weblate 進程的用戶擁有和可寫入（通常是 WSGI 和 Celery，見運行服務器 and 使用 Celery 的後台任務）。

默認的配置放置在 Weblate 源的相同樹下，然而你會想要將這些移動到更好的位置，如： `/var/lib/weblate`。

Weblate 試圖自動建立這些文件夾，但當有權限去執行時會失敗。

當運行管理命令時應該小心，它們應該由 Weblate 自己運行的相同用戶來運行，否則一些文件的權限會是錯誤的。

In the Docker container, all files in the `/app/data` volume have to be owned by the `weblate` user inside the container (UID 1000).

也參考：

*Serving static files*

### 2.1.5 Weblate 的數據庫設置

推薦使用 PostgreSQL 數據庫服務器來運行 Weblate。

也參考：

使用力的數據庫引擎, Databases, 從其它數據庫遷移到 PostgreSQL

#### PostgreSQL

PostgreSQL 通常是基於 Django 的網站的最好選擇。它是實現 Django 數據庫層而使用的參考數據庫。

---

備註：Weblate 使用三字母的擴展名，在某些情況下需要單獨安裝。查找 `postgresql-contrib` 或類似命名的包。

---

也參考：

PostgreSQL notes

#### 建立 PostgreSQL 數據庫

在另一個單獨的數據庫中運行 Weblate，將用戶賬戶分開通常是個好方法：

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser --superuser --pwprompt weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

---

提示：如果不想 Weblate 在 PostgreSQL 中使用超級用戶，可以省略掉。在模式中必須作 PostgreSQL 超級用戶，來手動執行一些遷移步驟的情況下，Weblate 將使用：

```
CREATE EXTENSION IF NOT EXISTS pg_trgm WITH SCHEMA weblate;
CREATE EXTENSION IF NOT EXISTS btree_gin WITH SCHEMA weblate;
```

---



## 配置 Weblate 來使用 PostgreSQL

PostgreSQL 的 settings.py 片段：

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.postgresql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL,
        # use in case role name is different than user used for authentication.
        # "ALTER_ROLE": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "database.example.com",
        # Set to empty string for default
        "PORT": "",
    }
}
```

數據庫的合規執行了 Weblate 使用的 `ALTER ROLE` 數據庫角色。在多數情況下，角色的名稱與用戶名匹配。在更富有的設置中，角色的名稱與用戶名不同，而在數據庫合規過程中會得到不存在的角色的錯誤信息（`psycopg2.errors.UndefinedObject: role "weblate@hostname" does not exist` ``，角色“weblate@hostname”不存在）。已知這會在 PostgreSQL 的 Azure 數據庫時發生，但不僅限於這種環境。請將 `ALTER_ROLE` 設置在數據庫合規過程中 Weblate 應該更改的角色名稱。

## MySQL and MariaDB

**提示：**一些 Weblate 特性使用 *PostgreSQL* 會執行得更好。這包括搜索與翻譯記憶庫，它們都使用了數據庫中的全文本特性，而 PostgreSQL 的實施更勝一籌。

Weblate 還可以使用 MySQL 或 MariaDB，使用與兩個數據庫相關的 Django 而導致的警告，請參見 [MySQL notes](#) 和 [MariaDB notes](#)。由於這些限制，我們建議對新安裝使用 *PostgreSQL*。

Weblate 需要至少 5.7.8 版的或至少 10.2.7 版的 MariaDB。

推薦 Weblate 使用後面的設置：

- 使用 `utf8mb4` 字符集來允許表示更高的 Unicode 平面（例如 emojis 表情符號）。
- 用 `innodb_large_prefix` 配置服務器，以允許在文本字段上有更長的索引。
- 設置隔離級 `READ COMMITTED`。
- SQL 模式應該設置 `STRICT_TRANS_TABLES`。

MySQL 8.x，MariaDB 10.5.x 或較新具有合理的默認配置，以便不需要使用服務器調整，並且可以在客戶端配置所有所需的服務器。

下面是用於 8GB 存服務器的 `/etc/my.cnf.d/server.cnf` 的例子。這些設置應該足以用於多數安裝。MySQL 和 MariaDB 具有來提高服務器性能的可調整部分，除非計劃有大量用戶訪問系統，否則這些可調整部分被認為是不必要的。這些細節請查看各廠商的文檔。

在運行您的 Weblate 前設置好 `innodb_file_per_table` 設置重啟 MySQL/MariaDB，這對少安裝時的問題對重要。

```
[mysqld]
character-set-server = utf8mb4
character-set-client = utf8mb4
collation-server = utf8mb4_unicode_ci

datadir=/var/lib/mysql

log-error=/var/log/mariadb/mariadb.log

innodb_large_prefix=1
innodb_file_format=Barracuda
innodb_file_per_table=1
innodb_buffer_pool_size=2G
sql_mode=STRICT_TRANS_TABLES
```

---

**提示：** 如遇 #1071 - Specified key was too long; max key length is 767 bytes 錯誤，請更新你的配置以包含上方的 innodb 設置 F 重新 F 動你的安裝。

---

---

**提示：** 在得到錯誤信息 #2006 - MySQL server has gone away 的形 F 下，配置 `CONN_MAX_AGE` 可能會有幫助。

---

## Configuring Weblate to use MySQL/MariaDB

MySQL 和 MariaDB 的 `settings.py` 片段：

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
        # Database user
        "USER": "weblate",
        # Database password
        "PASSWORD": "password",
        # Set to empty string for localhost
        "HOST": "127.0.0.1",
        # Set to empty string for default
        "PORT": "3306",
        # In case you wish to use additional
        # connection options
        "OPTIONS": {},
    }
}
```

開始安裝前還應該在 MySQL 或 MariaDB 中創建 weblate 用 F 賬 F。使用下面的命令來實現：

```
GRANT ALL ON weblate.* to 'weblate'@'localhost' IDENTIFIED BY 'password';
FLUSH PRIVILEGES;
```

## 2.1.6 其他配置

### Configuring outgoing e-mail

Weblate 在各種情況下會發出電子郵件——用於激活賬戶，以及用配置過的各種通知。對於這些需要訪問 SMTP 服務器。

郵件服務器使用這些設置進行配置: `EMAIL_HOST`, `EMAIL_HOST_PASSWORD`, `EMAIL_USE_TLS`, `EMAIL_USE_SSL`, django: `EMAIL_HOST_USER` and `EMAIL_PORT`。從名稱就可以大概知道它們的含義，但是你可以在 Django 文檔中找到更多信息。

---

**提示:** 在得到有關不支持的認證的情況下（例如 SMTP AUTH extension not supported by server），這最可能因使用不安全的鏈接且服務器拒以這種方式認證而導致。在這種情況下嘗試啟動 `EMAIL_USE_TLS`。

---

#### 也參考:

: 參考: “調試郵件”, : 參考: “在 Docker 容器中配置傳出電子郵件 <docker-mail>”

### 在反向代理後面運行

Weblate 的幾個特性依賴於能得到客戶端 IP 地址。這包括頻次限制、針對垃圾郵件的保護或稽核記錄。在默認設置中，Weblate 從 WSGI 句柄設置的 `REMOTE_ADDR` 中解析 IP 地址。

在運行反向代理的情況下，這個字段很可能包含其地址。需要配置 Weblate 來信任附加的 HTTP 標頭，從中解析 IP 地址。這不能默認允許，因在不使用反向代理的安裝時，這會允許 IP 地址欺騙。允許 `IP_BEHIND_REVERSE_PROXY` 對多數常見設置就足夠了，但你還必須調整 `IP_PROXY_HEADER` 和 `IP_PROXY_OFFSET`。

Another thing to take care of is the Host header. It should match to whatever is configured as `SITE_DOMAIN`. Additional configuration might be needed in your reverse proxy (for example use `ProxyPreserveHost On` for Apache or `proxy_set_header Host $host;` with nginx).

#### 也參考:

: 參考: Ref: “垃圾郵件保護”, : REF: “速率限制”, : REF: “審計日誌”, : “`IP_BEHIND_REVERSE_PROXY`”:  
django: `secure_proxy_ssl_header`

### HTTP 代理

Weblate 執行版本控制系統 (VCS) 命令，且那些從環境中接受代理配置。推薦的方法是在 `settings.py` 中定義代理設置:

```
import os

os.environ["http_proxy"] = "http://proxy.example.com:8080"
os.environ["HTTPS_PROXY"] = "http://proxy.example.com:8080"
```

#### 也參考:

Proxy Environment Variables

## 2.1.7 調整配置

### 也參考:

配置的例子

將 `weblate/settings_example.py` 複製到 `weblate/settings.py`, 且調整它與你的設置匹配。你可能想要調整後面的選項: `ADMINS`

網站管理者的列表, 當發生故障時它們接收通知, 例如合失敗或 Django 錯誤的通知。

### 也參考:

`ADMINS`, *Properly configure admins*

`ALLOWED_HOSTS`

需要設置這個, 來列出你的網站支持服務的主機。例如:

```
ALLOWED_HOSTS = ["demo.weblate.org"]
```

另外可以包括通配符:

```
ALLOWED_HOSTS = ["*"]
```

### 也參考:

`ALLOWED_HOSTS`, `WEBLATE_ALLOWED_HOSTS`, *Allowed hosts setup*

`SESSION_ENGINE`

配置如何存儲會話。在保持默認的數據庫後端引擎的情況下, 應該安排 **weblate clearsessions** 從數據庫中刪除舊的會話。

如果使用 Redis 作緩存 (請參見允許緩存), 推薦也使用它作會話:

```
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
```

### 也參考:

*Configuring the session engine*, `SESSION_ENGINE`

`DATABASES`

到數據庫服務器的連接性, 細節請查看 Django 的文件。

### 也參考:

*Weblate 的數據庫設置*, `DATABASES`, `Databases`

`DEBUG`

對於任何生產服務器禁止這項。允許調試模式時, Django 會在出錯的情況下向用戶顯示回溯信息, 當禁止時, 錯誤將每封電子郵件發送到 `ADMINS` (請參見上面)。

調試模式還使 Weblate 變慢, 因在這種情況下 Django 存儲了非常多的信息。

### 也參考:

`DEBUG`, *Disable debug mode*

`DEFAULT_FROM_EMAIL`

用於發送電子郵件的電子郵件發件人地址, 例如用戶電子郵件。

### 也參考:

`DEFAULT_FROM_EMAIL`

`SECRET_KEY`

金鑰會被 Django 簽署相同資訊在瀏覽器 cookies 中，參 [Django secret key](#) 解更多資訊。

**也參考：**

`SECRET_KEY`

`SERVER_EMAIL`

用作向管理員發送電子郵件的發送者地址的郵箱，例如通知失敗的合

**也參考：**

`SERVER_EMAIL`

### 2.1.8 填滿數據庫

在配置準備好之後，可以運行 `weblate migrate` 來建立數據庫結構。現在你將能使用管理界面建立翻譯項目。

在想要非交互式地運行安裝的情況下，可以使用 `weblate migrate --noinput`，然後使用 `createadmin` 命令來建立管理用。

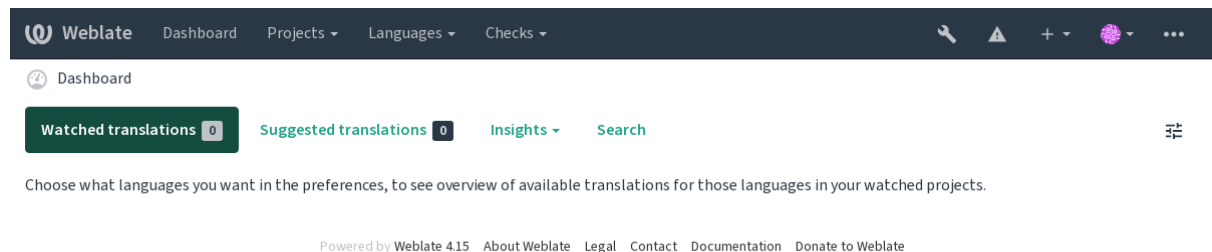
一旦完成，你將可以在管理界面檢查 *Performance report*，它會提示你網站上在的非最優的配置。

**也參考：**

配置, 特殊權限列表與建角色

### 2.1.9 生設置

對於生設置，可以進行後面的章節中描述的調整。最嚴格的設置將觸發警告，如果超級用登的話，警告由頂部條的感嘆標記來指示：



同樣也推薦查看由 Django 觸發的檢查（管可能不需要修復所有的檢查）：

```
weblate check --deploy
```

還可以復查來自管理介面的每個檢查表。

**也參考：**

Deployment checklist

## Disable debug mode

禁止 Django 的調試模式 (*DEBUG*):

```
DEBUG = False
```

在調試模式打開時，Django 存儲所有執行的查詢，將錯誤的回溯顯示給用戶，這在生產設置中是不需要的。

也參考:

[調整配置](#)

## Properly configure admins

將正確的管理地址設置到 *ADMINS* 設置中，來確定服務器出現一些故障時誰接收電子郵件，例如：

```
ADMINS = (("Your Name", "your_email@example.com"),)
```

也參考:

[調整配置](#)

## 設置正確的網站域名

在管理界面調整網站名稱和域名，否則 RSS 中的鏈接或電子郵件地址將不工作。這使用 *SITE\_DOMAIN* 來配置，它應該包含網站域名。

在 4.2 版本變更: 在 4.2 版本之前，替代使用了 Django 網站框架，請參見 [The “sites” framework](#)。

也參考:

: 參考: [ref: “生 - 主機”](#); REF: [“生-SSL”](#); 設置: *Site\_Domain*; envvar: *weblate\_site\_domain*; 設置: *enable\_https*

## Correctly configure HTTPS

強烈推薦使用加密的 HTTPS 協議運行 Weblate。將其允許後，可以在設置中設置 *ENABLE\_HTTPS* :

```
ENABLE_HTTPS = True
```

---

**提示:** 你還會想要新建 HSTS，更多細節請參見 [SSL/HTTPS](#)。

---

也參考:

[ENABLE\\_HTTPS](#), [Allowed hosts setup](#), [設置正確的網站域名](#)

## Set properly SECURE\_HSTS\_SECONDS

如果你的網站基於 SSL 上提供服務，那必須考慮 *settings.py* 中 *SECURE\_HSTS\_SECONDS* 的設置值，來允許 HTTP Strict Transport Security (HTTP 本傳輸安全)。默認設置 0，如下所示。

```
SECURE_HSTS_SECONDS = 0
```

如果設置非 0 整數，那在所有還不曾具有它的響應時，*django.middleware.security.SecurityMiddleware* 設置 HTTP Strict Transport Security 標頭。

**警告：** 不正確地設置這項會導致你的網站不可逆地（有時）崩潰。請首先閱讀 [HTTP Strict Transport Security](#) 文件。

## 使用力的數據庫引擎

- 請使用 PostgreSQL 作生產環境，更多信息請參見 [Weblate 的數據庫設置](#)。
- Use adjacent location for running the database server, otherwise the networking performance or reliability might ruin your Weblate experience.
- Check the database server performance or tweak its configuration, for example using PG Tune.

### 也參考：

[Weblate 的數據庫設置](#), [從其它數據庫遷移到 PostgreSQL](#), [調整配置](#), [Databases](#)

## 允許緩存

如果可能，通過調整 CACHES 配置變量來使用來自 Django 的 Redis，例如：

```
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/0",
        # If redis is running on same host as Weblate, you might
        # want to use unix sockets instead:
        # 'LOCATION': 'unix:///var/run/redis/redis.sock?db=0',
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
}
```

**提示：** 在緩存更改 Redis 設置的情況下，也會需要 Celery 來調整，請參見使用 [Celery](#) 的後台任務。

### 也參考：

[個人頭像快取](#), Django's cache framework

## 個人頭像快取

除了 Django 的緩存，Weblate 還執行頭像緩存。推薦使用單獨的、文件後端緩存來用於這個目的：

```
CACHES = {
    "default": {
        # Default caching backend setup, see above
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "unix:///var/run/redis/redis.sock?db=0",
        "OPTIONS": {
            "CLIENT_CLASS": "django_redis.client.DefaultClient",
            "PARSER_CLASS": "redis.connection.HiredisParser",
        },
    },
    "avatar": {
        "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
        "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
    },
}
```

(繼續下一頁)

(繼續上一頁)

```

    "TIMEOUT": 604800,
    "OPTIONS": {
        "MAX_ENTRIES": 1000,
    },
},
}

```

**也參考:**

: 設置: *enabent\_avatars*, : 設置: *avatar\_url\_prefix*, : ref: *avatars*, : ref: *spections-cache*, : doc: *django*: 主題/緩存

**Configure e-mail sending**

Weblate 需要在幾種情況下發送電子郵件，這些電子郵件應具有正確的發送者地址，請配置:setting:*SERVER\_EMAIL* 和 *DEFAULT\_FROM\_EMAIL*，與你的環境匹配，例如：

```

SERVER_EMAIL = "admin@example.org"
DEFAULT_FROM_EMAIL = "weblate@example.org"

```

**備註：** 為了禁止 Weblate 發送電子郵件，將 *EMAIL\_BACKEND* 設置為 *django.core.mail.backends.dummy.EmailBackend*。

這將禁止 所有電子郵件的投遞，包括 或密碼重置電子郵件。

**也參考:**

: 參考: “配置”, : REF: “外郵件”, : STD: 設置: *Django: Email\_Backend*, : std: 設置: *django: default\_mail*, : std: 設置: *django: server\_email*

**Allowed hosts setup**

Django 需要 *ALLOWED\_HOSTS* 保存你的網站允許服務的域名列表，將其保持空置會屏蔽任何請求。

在 有配置來匹配 HTTP 服務器的情況下，會得到錯誤信息，如 *Invalid HTTP\_HOST header: '1.1.1.1'. You may need to add '1.1.1.1' to ALLOWED\_HOSTS.*

**提示：** 在 Docker 容器上，這可以使用， *WEBLATE\_ALLOWED\_HOSTS*。

**也參考:**

*ALLOWED\_HOSTS*, *WEBLATE\_ALLOWED\_HOSTS*, 設置正確的網站域名

**Django secret key**

*SECRET\_KEY* 設置由 Django 使用來進行 cookies 簽名，應該真正 生自己的值，而不是使用來自舉例的設置的值。

可以使用與 Weblate 一起上市的 **weblate-generate-secret-key**，來 生新的密鑰。

**也參考:**

*SECRET\_KEY*



## Running maintenance tasks

為了優化性能，在後台運行一些維護任務是個好方法。現在這由使用 *Celery* 的後台任務自動進行，且包括後面的任務：

- 配置健康性的檢查（每小時）。
- 提交待定的更改（每小時），請參見簡易提交和 `commit_pending`。
- 更新組件警告（每天）。
- 更新遠程分支（每晚），請參見 `AUTO_UPDATE`。
- 翻譯記憶庫備份到 JSON（每天），請參見 `dump_memory`。
- 全文本 and 數據庫維護任務（每天和每任務），請參見 `cleanuptrans`。

在 3.2 版本變更：從 3.2 版本開始，執行這些任務的默認方式是使用 *Celery*，且 *Weblate* 已經具有一些適當的配置，請參見使用 *Celery* 的後台任務。

## 系統的地區與編碼

系統的地區應該設置兼容 UTF-8 的。在多數 Linux 發布中這是默認的設置。在你的系統不能兼容的情況下，請將地區更改 UTF-8 變體。

例如通過編輯 `/etc/default/locale` 設置 `LANG="C.UTF-8"`。

In some cases the individual services have separate configuration for locales. This varies between distribution and web servers, so check documentation of your web server packages for that.

Apache on Ubuntu uses `/etc/apache2/envvars`:

```
export LANG='en_US.UTF-8'
export LC_ALL='en_US.UTF-8'
```

Apache on CentOS uses `/etc/sysconfig/httpd` (or `/opt/rh/httpd24/root/etc/sysconfig/httpd`):

```
LANG='en_US.UTF-8'
```

## 使用定制的證書授權

*Weblate* 在 HTTP 請求時驗證 SSL 證書。在使用定制的證書授權的情況下，這樣定制的證書授權在默認 bundles 的中不被信任，你必須將其證書添加可信任。

傾向使用的方法是在系統層次進行，更多細節請查看你的發布的文件（例如在 *debian* 中，這可以通過將 CA 證書放置在 `file:/usr/local/share/ca-certificates/`，運行 `command:update-ca-certificates` 來完成）。

一旦完成，系統工具就會信任證書，這包括 *Git*。

對於 Python 代碼，需要配置請求來使用系統 CA bundle，而不是與它一起上市的那個。這可以通過將後面的模板放到 `settings.py` 來實現（路徑是 *Debian* 特有的）：

```
import os

os.environ["REQUESTS_CA_BUNDLE"] = "/etc/ssl/certs/ca-certificates.crt"
```

## Compressing client assets

Weblate 帶有一組 JavaScript 和 CSS 文件。由於性能的原因，在將其發送到客戶端前最好進行壓縮。在默認配置中，這通過耗費一點經常資源而在運行中完成。在大型安裝中，推薦允許離線壓縮模式。這需要在配置中完成，並且必須在每次 weblate 升級時觸發壓縮。

配置切很簡單，通過允許 `django.conf.settings.COMPRESS_OFFLINE`，配置 `django.conf.settings.COMPRESS_OFFLINE_CONTEXT`（後者已經包括在例子的配置中）：

```
COMPRESS_OFFLINE = True
```

在每個部署中，您需要壓縮文件來匹配當前的版本：

```
weblate compress
```

---

**提示：** 官方 Docker 鏡像已經允許了這個特性。

---

**也參考：**

[Common Deployment Scenarios](#), [Serving static files](#)

### 2.1.10 運行服務器

---

**提示：** 如果您有在下面描述的服務經歷，您可能需要嘗試：[Doc：安裝 / docker](#)。

---

需要幾個服務來運行 Weblate，推薦的設置包括：

- 數據庫服務器（請參見 [Weblate 的數據庫設置](#)）
- 緩存服務器（請參見 [允許緩存](#)）
- 用於態文件和終結 SSL 的前端 web 服務器（請參見 [Serving static files](#)）
- 用於動態容的 WSGI 服務器（請參見 [NGINX](#) 和 [uWSGI](#) 的配置例子）
- 用於執行後台任務的 Celery（請參見 [使用 Celery 的後台任務](#)）

---

**備：** 這些服務之間由一些依賴性，例如當動 Celery 或 uwsgi 進程時，緩存和數據庫應該運行。

---

在多數情況下，需要在單一（擬）服務器上運行所有服務，但在您的安裝是重載的情況下，可以將這些服務拆開。對此的唯一限制是 Celery 和 Wsgi 服務器需要訪問 `DATA_DIR`。

---

**備：** WSGI 進程和 Celery 進程必須在同一用下被執行，否則 `DATA_DIR` 中的文件將以混合的所有權來存儲，導致運行問題。

還請參見 [文件系統權限](#) 和 [使用 Celery 的後台任務](#)。

---

## Running web server

運行 Weblate 與運行其他任何基於 Django 的程序不同。Django 通常作 uWSGI 或 fcgi 執行（請參見下面不同 web 服務器的例子）。

為了檢測的目的，您可以在 Django 中使用建 web 服務器：

```
weblate runserver
```

**警告：** 在生設置中不要使用這個服務器。它還有通過安全審查或性能檢測。還請參見 `runserver` 上的 Django 文件。

**提示：** Django 建服務只通過允許 `DEBUG` 來態文件提供服務，因它只用於開發的目的。對於生使用，請參見 *NGINX* 和 *uWSGI* 的配置例子、*Apache* 的配置例子 *Apache* 和 *Gunicorn* 的配置例子 和 *Serving static files* 中的 `wsgi` 設置。

## Serving static files

在 2.4 版本變更：在 2.4 版本之前，Weblate 不能正常使用 Django 態文件框架，且設置更難。

Django 需要將其態文件收集在一個單一文件夾中。此，執行 `weblate collectstatic --noinput`。這會將態文件到 `STATIC_ROOT` 設置指定的文件夾中（這默認 `DATA_DIR` 的 `static` 文件夾）。

推薦直接從你的 web 服務器態文件提供服務，對於後面的路徑應該使用它：

**/static/**

Weblate 的態文件和管理界面（由 `STATIC_ROOT` 定義）提供服務。

**/media/**

用於上傳用媒體（例如截屏）。

**/favicon.ico**

應該重寫，重寫規則 `/static/favicon.ico` 提供服務。

**也參考：**

*NGINX* 和 *uWSGI* 的配置例子, *Apache* 的配置例子, *Apache* 和 *Gunicorn* 的配置例子, *Compressing client assets*, *How to deploy Django*, `django :howto/static-files/deployment`

## Content security policy

默認的 Weblate 配置允許 `weblate.middleware.SecurityMiddleware` 中間件，它設置與 HTTP 標頭相關的安全，如 `Content-Security-Policy` 或 `X-XSS-Protection`。這些被默認新建，與 Weblate 及其配置一起工作，但這對你的環境需要定制化。

**也參考：**

：設置 `csp_script_src`，設置 `csp_img_src`，設置 `csp_connect_src`，設置 `csp_style_src`，設置 `csp_font_src`

## NGINX 和 uWSGI 的配置例子

為了運行生 web 服務器，使用與 Weblate 一起安裝的 wsgi 封裝（在擬 env 的情況下，它安裝在 `file:~/weblate-env/lib/python3.9/site-packages/weblate/wsgi.py`）。別忘了將 Python 的搜索路徑同樣設置在您的擬 env（例如在 uWSGI 中使用 `“virtualenv = /home/user/weblate-env”`）。

後面的配置將 Weblate 作為 NGINX web 服務器下的 uWSGI 來運行。

NGINX 的配置（還作 `weblate/examples/weblate.nginx.conf` 來獲得）：

```
#
# nginx configuration for Weblate
#
# You will want to change:
#
# - server_name
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
server {
    listen 80;
    server_name weblate;
    # Not used
    root /var/www/html;

    location ~ ^/favicon.ico$ {
        # DATA_DIR/static/favicon.ico
        alias /home/weblate/data/static/favicon.ico;
        expires 30d;
    }

    location /static/ {
        # DATA_DIR/static/
        alias /home/weblate/data/static/;
        expires 30d;
    }

    location /media/ {
        # DATA_DIR/media/
        alias /home/weblate/data/media/;
        expires 30d;
    }

    location / {
        include uwsgi_params;
        # Needed for long running operations in admin interface
        uwsgi_read_timeout 3600;
        # Adjust based to uwsgi configuration:
        uwsgi_pass unix:///run/uwsgi/app/weblate/socket;
        # uwsgi_pass 127.0.0.1:8080;
    }
}
```

uWSGI 的配置（還作 `weblate/examples/weblate.uwsgi.ini` 來獲得）：

```
#
# uWSGI configuration for Weblate
#
# You will want to change:
#
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
```

(繼續下一頁)

(繼續上一頁)

```

# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
[uwsgi]
plugins      = python3
master       = true
protocol     = uwsgi
socket       = 127.0.0.1:8080
wsgi-file    = /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/wsgi.
↳py

# Add path to Weblate checkout if you did not install
# Weblate by pip
# python-path = /path/to/weblate

# In case you're using virtualenv uncomment this:
virtualenv = /home/weblate/weblate-env

# Needed for OAuth/OpenID
buffer-size  = 8192

# Reload when consuming too much of memory
reload-on-rss = 250

# Increase number of workers for heavily loaded sites
workers      = 8

# Enable threads for Sentry error submission
enable-threads = true

# Child processes do not need file descriptors
close-on-exec = true

# Avoid default 0000 umask
umask = 0022

# Run as weblate user
uid = weblate
gid = weblate

# Enable harakiri mode (kill requests after some time)
# harakiri = 3600
# harakiri-verbose = true

# Enable uWSGI stats server
# stats = :1717
# stats-http = true

# Do not log some errors caused by client disconnects
ignore-sigpipe = true
ignore-write-errors = true
disable-write-exception = true

```

**也參考:**

How to use Django with uWSGI

## Apache 的配置例子

推薦當 Weblate 使用 WSGI 時使用 prefork MPM。

後面的配置將 Weblate 作 WSGI 來運行，您需要允許“mod\_wsgi”（作 weblate/examples/apache.conf 來獲得）：

```
#
# VirtualHost for Weblate
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate_
↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias / /home/weblate/weblate-env/lib/python3.9/site-packages/
↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>
</VirtualHost>
```

備：Weblate 需要 Python 3，所以請確認您運行 modwsgi 的 Python 3 變體。它通常作獨立的包來獲得，例如 libapache2-mod-wsgi-py3。

### 也參考：

系統的地區與編碼, How to use Django with Apache and mod\_wsgi

## Apache 和 Gunicorn 的配置例子

後面的配置在 Gunicorn 和 Apache 2.4 中運行 Weblate (作 `weblate/examples/apache.gunicorn.conf` 獲得):

```
#
# VirtualHost for Weblate using gunicorn on localhost:8000
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:443>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/https_cert.cert
    SSLCertificateKeyFile /etc/apache2/ssl/https_key.pem
    SSLProxyEngine On

    ProxyPass /favicon.ico !
    ProxyPass /static/ !
    ProxyPass /media/ !

    ProxyPass / http://localhost:8000/
    ProxyPassReverse / http://localhost:8000/
    ProxyPreserveHost On
</VirtualHost>
```

也參考:

How to use Django with Gunicorn

## 在路徑下運行 Weblate

在 1.3 版本新加入。

推薦當 Weblate 使用 WSGI 時使用 prefork MPM。

“/weblate” 下的 Weblate 提供服務的 Apache 配置的例子。再次使用 mod\_wsgi （還作 weblate/examples/apache-path.conf 獲得）：

```
#
# VirtualHost for Weblate, running under /weblate path
#
# You will want to change:
#
# - ServerAdmin and ServerName
# - change /home/weblate/weblate-env to location where Weblate virtualenv is placed
# - change /home/weblate/data to match your DATA_DIR
# - change python3.9 to match your Python version
# - change weblate user to match your Weblate user
#
<VirtualHost *:80>
    ServerAdmin admin@weblate.example.org
    ServerName weblate.example.org

    # DATA_DIR/static/favicon.ico
    Alias /weblate/favicon.ico /home/weblate/data/static/favicon.ico

    # DATA_DIR/static/
    Alias /weblate/static/ /home/weblate/data/static/
    <Directory /home/weblate/data/static/>
        Require all granted
    </Directory>

    # DATA_DIR/media/
    Alias /weblate/media/ /home/weblate/data/media/
    <Directory /home/weblate/data/media/>
        Require all granted
    </Directory>

    # Path to your Weblate virtualenv
    WSGIDaemonProcess weblate python-home=/home/weblate/weblate-env user=weblate_
↪request-timeout=600
    WSGIProcessGroup weblate
    WSGIApplicationGroup %{GLOBAL}

    WSGIScriptAlias /weblate /home/weblate/weblate-env/lib/python3.9/site-packages/
↪weblate/wsgi.py process-group=weblate
    WSGIPassAuthorization On

    <Directory /home/weblate/weblate-env/lib/python3.9/site-packages/weblate/>
        <Files wsgi.py>
            Require all granted
        </Files>
    </Directory>
</VirtualHost>
```

此外，您必須調整 weblate/settings.py：

```
URL_PREFIX = "/weblate"
```



### 2.1.11 使用 Celery 的後台任務

在 3.2 版本新加入。

Weblate uses Celery to execute regular and background tasks. You are supposed to run a Celery service that will execute these. For example, it is responsible for handling following operations (this list is not complete):

- Receiving webhooks from external services (see [通知勾](#)).
- Running regular maintenance tasks such as backups, cleanups, daily add-ons, or updates (see [備份和移動 Weblate, BACKGROUND\\_TASKS](#), [附加元件](#)).
- Running [自動翻譯](#).
- Sending digest notifications.
- Offloading expensive operations from the wsgi process.
- Committing pending changes (see [簡易提交](#)).

A typical setup using Redis as a backend looks like this:

```
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL
```

也參考:

: 參考: [芹菜 < 芹菜: Broker-Redis-Configuration >](#) “redis broker 配置”

您應該啟動 Celery worker 來處理任務，且定時任務，這可以直接在命令行完成（調試和開發時最有用）:

```
./weblate/examples/celery start
./weblate/examples/celery stop
```

**備註:** Celery 進程和 WSGI 進程必須在同一用下被執行，否則 `DATA_DIR` 中的文件將以混合的所有權來存儲，導致運行問題。

還請參見 [文件系統權限](#) 和 [運行服務器](#)。

### Executing Celery tasks in the wsgi using eager mode

**備註:** This will have severe performance impact on the web interface, and will break features depending on regular trigger (for example committing pending changes, digest notifications, or backups).

For development, you might want to use eager configuration, which does process all tasks in place:

```
CELERY_TASK_ALWAYS_EAGER = True
CELERY_BROKER_URL = "memory://"
CELERY_TASK_EAGER_PROPAGATES = True
```

## 運行 Celery 作系統服務

您更可能想要運行 Celery 作守護進程，這由 [Daemonization](#) 來涵蓋。對於使用 [systemd](#) 的最通常的 Linux 設置，您可以使用與下面列出的 `examples` 文件夾一起上市的例子文件。

Systemd 單元作 `/etc/systemd/system/celery-weblate.service` 放置：

```
[Unit]
Description=Celery Service (Weblate)
After=network.target

[Service]
Type=forking
User=weblate
Group=weblate
EnvironmentFile=/etc/default/celery-weblate
WorkingDirectory=/home/weblate
RuntimeDirectory=celery
RuntimeDirectoryPreserve=restart
LogsDirectory=celery
ExecStart=/bin/sh -c '${CELERY_BIN} multi start ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'
ExecStop=/bin/sh -c '${CELERY_BIN} multi stopwait ${CELERYD_NODES} \
  --pidfile=${CELERYD_PID_FILE}'
ExecReload=/bin/sh -c '${CELERY_BIN} multi restart ${CELERYD_NODES} \
  -A ${CELERY_APP} --pidfile=${CELERYD_PID_FILE} \
  --logfile=${CELERYD_LOG_FILE} --loglevel=${CELERYD_LOG_LEVEL} ${CELERYD_OPTS}'

[Install]
WantedBy=multi-user.target
```

環境配置作 `/etc/default/celery-weblate` 放置：

```
# Name of nodes to start
CELERYD_NODES="celery notify memory backup translate"

# Absolute or relative path to the 'celery' command:
CELERY_BIN="/home/weblate/weblate-env/bin/celery"

# App instance to use
# comment out this line if you don't use an app
CELERY_APP="weblate.utils"

# Extra command-line arguments to the worker,
# increase concurrency if you get weblate.E019
CELERYD_OPTS="--beat:celery --queues:celery=celery --prefetch-multiplier:celery=4 \
  --queues:notify=notify --prefetch-multiplier:notify=10 \
  --queues:memory=memory --prefetch-multiplier:memory=10 \
  --queues:translate=translate --prefetch-multiplier:translate=4 \
  --concurrency:backup=1 --queues:backup=backup --prefetch-multiplier:backup=2"

# Logging configuration
# - %n will be replaced with the first part of the nodename.
# - %I will be replaced with the current child process index
# and is important when using the prefork pool to avoid race conditions.
CELERYD_PID_FILE="/run/celery/weblate-%n.pid"
CELERYD_LOG_FILE="/var/log/celery/weblate-%n%I.log"
CELERYD_LOG_LEVEL="INFO"
```

使用 **logrotate** 來旋轉 Celery 記的額外配置將被放置 `/etc/logrotate.d/celery`：

```
/var/log/celery/*.log {
    weekly
    missingok
    rotate 12
    compress
    notifempty
}
```

### 使用 Celery beat 的周期性任務

Weblate 帶有建定的定時任務設置。然而您可以在 `settings.py` 中定義另外的任務，例如請參見簡易提交。

任務應該由 Celery beats 守護進程執行。在不能正常工作的情況下，它可能不會運行，或者其數據庫崩潰。在這樣的情況下檢查 Celery 動日，來找出根本原因。

### Monitoring Celery status

You can find current length of the Celery task queues in the 管理介面 or you can use `celery_queues` on the command-line. In case the queue will get too long, you will also get configuration error in the admin interface.

**警告：** Celery 錯誤默認之存儲在 Celery 日中，且用不可見。在您想要了解故障概的情況下，推薦配置 *Collecting error reports*。

#### 也參考：

監測 Weblate, 如何檢查 Weblate 是否被正確地設置?, Configuration and defaults, Workers Guide, Daemonization, Monitoring and Management Guide, `celery_queues`

## 2.1.12 監測 Weblate

Weblate 提供 `/healthz/` URL 作簡單的健康檢查來使用，例如使用 Kubernetes。Docker 容器具有使用此 URL 的置運行狀檢查。

用於監視 Weblate 的指標，您可以使用：`http: get: /api / metrics / api` 端點。

#### 也參考：

如何檢查 Weblate 是否被正確地設置?, *Monitoring Celery status*, *Weblate plugin for Munin*

## 2.1.13 Collecting error reports

與其他任何軟件一樣，Weblate 可能會失敗。了收集有用的故障狀態，我們推薦使用第三方服務來收集此類信息。這在 Celery 任務失敗的情況下尤其有用，否則將只會向日報告錯誤，而您不會收到有關它們的通知。Weblate 支持以下服務：

## Sentry

Weblate 配置了對 Sentry 的支持。要使用它，只需在 `settings.py` 中設置 `SENTRY_DSN`:

```
SENTRY_DSN = "https://id@your.sentry.example.com/"
```

## Rollbar

Weblate 具有對 Rollbar 的配置支持。要使用它，只需遵循 [Rollbar notifier for Python](#) 的說明即可。

簡而言之，您需要調整 `settings.py`：

```
# Add rollbar as last middleware:
MIDDLEWARE = [
    # ... other middleware classes ...
    "rollbar.contrib.django.middleware.RollbarNotifierMiddleware",
]

# Configure client access
ROLLBAR = {
    "access_token": "POST_SERVER_ITEM_ACCESS_TOKEN",
    "client_token": "POST_CLIENT_ITEM_ACCESS_TOKEN",
    "environment": "development" if DEBUG else "production",
    "branch": "main",
    "root": "/absolute/path/to/code/root",
}
```

將會自動整合所有必要的設定，您將會收集到來自伺服器或是使用端的錯誤訊息。

### 2.1.14 將 Weblate 遷移到其他服務其中

將 Weblate 遷移到其他服務器應該非常簡單，然而它將數據存儲在幾個位置，您應該小心遷移。最佳的方式時停止 Weblate 再遷移。

#### 遷移數據庫

Depending on your database backend, you might have several options to migrate the database. The most straightforward approach is to use database native tools, as they are usually the most effective (e.g. `mysqldump` or `pg_dump`). Alternatively you can use replication in case your database supports it.

#### 也參考:

Migrating between databases described in [從其它數據庫遷移到 PostgreSQL](#).

#### Migrating VCS repositories

存儲在 `DATA_DIR` 下的版本控制系統 (VCS) 同樣需要遷移。您可以簡單地複製它們，或使用 `rsync` 來更有效地遷移。

## 其他釋

不要忘記移動 Weblate 會使用的其他服務，如 Redis、Cron 任務或定制的身份驗證後端。

## 2.2 Weblate 部署

如果多個組件分享相同的倉庫，需要分將他們全部鎖定：

- *Installing using Docker*
- *Installing on OpenShift*
- 在 *Kubernetes* 上安裝

### 2.2.1 Third-party deployments for Weblate

備：後面的部署不是由 Weblate 團隊開發或支持的。部分設置會與本文件中描述的有偏差。

#### Bitnami Weblate stack

Bitnami 很多平台提供 Weblate 棧 <<https://bitnami.com/stack/weblate>>。設置在安裝過程中調整，更多文件請參見 <<https://bitnami.com/stack/weblate/README.txt>>。

#### Weblate Cloudron Package

Cloudron 是自管 web 應用的平台。安裝有 Cloudron 的 Weblate 會自動更新。軟件包由 Cloudron 團隊在它們的 Weblate package repo 上維護。



#### Weblate in YunoHost

自管項目 YunoHost 提供了包。一旦安裝了 YunoHost，就可以同其它應用一樣安裝 Weblate。它還提供帶有備份和恢復的完全工作棧，但你必須特定應用編輯設置文件。

可以使用管理界面，或這個按鈕（它將帶你到你的服務器）：



還能使用命令行界面：

```
yunohost app install https://github.com/YunoHost-Apps/weblate_ynh
```

## 2.3 升級 Weblate

### 2.3.1 Docker 鏡像升級

官方 Docker 鏡像（參見 *Installing using Docker*）集成了所有 Weblate 升級步驟。除了拉取最新版本外，通常不需要手動步驟。

也參考：

升級 Docker 容器

### 2.3.2 通用的升級

在升級前，請檢查當前的軟件要求，因他們可能被更改。一旦所有的要求被安裝或升級，請調整你的 `settings.py`，來匹配配置中的更改（正確的值請詢 `settings_example.py`）。

在升級前總是查看版本特定。在你跳過一些版本的情況下，請遵從在升級中您跳過的所有版本的指示。有時最好升級到一些中間版本，來確保平滑遷移。跨多發行版本的升級應該可以工作，但還像單一版本升級一樣測試過。

---

**備：** 推薦在升級前執行全數據庫備份，使你可以在升級失敗的情況下回數據庫，請參見 *備份和移動 Weblate*。

---

1. 停止 WSGI 和 Celery 進程。升級可能執行數據庫的不兼容更改，因此在升級中避免舊的進程運行總是安全的。
2. 升級 Weblate 代碼。

對於 pip 安裝，可以通過後面的來實現：

```
pip install -U "Weblate[all]==version"
```

或者，如果您只想獲取最新發的版本：

```
pip install -U "Weblate[all]"
```

If you don't want to install all of the optional dependencies do:

```
pip install -U Weblate
```

通過 Git 核實，你需要取回新的源代碼升級你的安裝：

```
cd weblate-src
git pull
# Update Weblate inside your virtualenv
. ~/weblate-env/bin/pip install -e .
# Install dependencies directly when not using virtualenv
pip install --upgrade -r requirements.txt
# Install optional dependencies directly when not using virtualenv
pip install --upgrade -r requirements-optional.txt
```

3. New Weblate release might have new 可選依賴性, please check if they cover features you want.
4. 升級配置文件，所需的步驟請參考 `settings_example.py` 或版本特定。
5. Upgrade database structure:

```
weblate migrate --noinput
```

6. 收集升級的狀態文件（請參見 *運行服務器* 和 *Serving static files*）：

```
weblate collectstatic --noinput --clear
```

7. 壓縮 JavaScript 和 CSS 文件（可選步驟，請參見 *Compressing client assets*）：

```
weblate compress
```

8. 如果你運行來自 Git 的版本，每次升級時還應該重新生成 locale 文件。可以通過調用後面的來進行：

```
weblate compilemessages
```

9. 驗證您的設置合理（還請參見 *生 設置*）：

```
weblate check --deploy
```

10. Restart Celery worker (see *使用 Celery 的後台任務*).

### 2.3.3 版本特定 明

#### Upgrade from 2.x

如果從 2.x 發布版本升級，首先總是升級到 3.0.1，然後繼續在 3.x 系列中升級。跳過這步的升級不被支持，且會中斷。

#### 也參考：

Weblate 3.0 文檔中關於從 2.20 升級到 3.0

#### Upgrade from 3.x

如果從 3.x 發布版本升級，首先總是升級到 4.0.4 或 4.1.1，然後繼續在 4.x 系列中升級。跳過這步的升級不被支持，且會中斷。

#### 也參考：

‘Weblate 4.0 文檔中關於從 3.11 升級到 4.0 <<https://docs.weblate.org/en/weblate-4.0.4/admin/upgrade.html#upgrade-from-3-11-to-4-0>>’

#### 從 4.0 升級到 4.1

請按照 *通用的升級 明* 來執行升級。

显著的配置与依赖性更改：

- 在 `settings_example.py` 中有几项更改，最显著的是中间件的更改，请由此调整你的设置。
- 有幾個新的文件格式，在修改 `WEBLATE_FORMATS` 的情 下，你會想要將他們包括進來。
- 有幾個新的質量檢查，在修改 `CHECK_LIST` 的情 下，你會想要將他們包括進來。
- 在 `DEFAULT_THROTTLE_CLASSES` 設置中有幾項更改，來允許在 API 中報告速率限制。
- 有幾個新的且更新的要求。
- 在 `INSTALLED_APPS` 中有一些更改。
- The `MT_DEEPL_API_VERSION` setting has been removed in Version 4.7. The *DeepL* machine translation now uses the new `MT_DEEPL_API_URL` instead. You might need to adjust `MT_DEEPL_API_URL` to match your subscription.

#### 也參考：

*通用的升級 明*

## 從 4.1 升級到 4.2

請按照[通用的升級說明](#)來執行升級。

显著的配置与依赖性更改：

- 從 3.x 發布版本升級不再支持，請首先升級到 4.0 或 4.1。
- 有幾個新的且更新的要求。
- 在 `settings_example.py` 中有几项更改，最显著的是新中间件和更改的应用订购。
- 基於 JSON 格式的密鑰是不再包括前導的點。在數據庫遷移過程中調整字符串，但在你依賴於導出或 API 中的密鑰時，外部組件會需要調整。
- Celery 配置更改，不再使用 `memory` 隊列。請調整你的[啟動本](#)和 `CELERY_TASK_ROUTES` 設置。
- 現在在設置中配置 Weblate 域，請參見 `SITE_DOMAIN``（或：`envvar: `WEBLATE_SITE_DOMAIN``）。在運行 Weblate 前你將不得不配置它。
- 用[數據庫上的用名](#)和電子郵件字段現在應該不因大小寫敏感而不同。它之前錯誤地有被 PostgreSQL 制。

也參考：

[通用的升級說明](#)

## 從 4.2 升級到 4.3

請按照[通用的升級說明](#)來執行升級。

显著的配置与依赖性更改：

- 在質量檢查中有一些更改，在你調整 `CHECK_LIST` 的情下會想將他們包括進來。
- 源語言屬性從項目移動到 API 中暴露的組件。在使用時你會需要更新 [Weblate 客戶端](#)。
- 根據翻譯的字符串數量，數據庫遷移到 4.3 會花費很長時間（期望每 10 萬個字符串的遷移時間大約 1 小時）。
- 在 `INSTALLED_APPS` 中有一些更改。
- 有個新的設置 `SESSION_COOKIE_AGE_AUTHENTICATED`，補充了 `SESSION_COOKIE_AGE`。
- 在使用 `hub`` 或：`command: `lab`` 與 GitHub 或 GitLab 集成的情下，需要重新配置它，請參見：`setting: `GITHUB_CREDENTIALS` 和 `GITLAB_CREDENTIALS`。

在 4.3.1 版本變更：

- Celery 配置更改，加入了 `memory` 隊列。請調整你的[啟動本](#)和 `CELERY_TASK_ROUTES` 設置。

在 4.3.2 版本變更：

- The `post_update` method of add-ons now takes extra `skip_push` parameter.

也參考：

[通用的升級說明](#)



### 從 4.3 升級到 4.4

請按照[通用的升級說明](#)來執行升級。

显著的配置与依赖性更改：

- 在 `INSTALLED_APPS` 中有一處更改，必須將 `weblate.configuration` 添加在那裏。
- 現在需要 Django 3.1。
- 在使用 MySQL 或 MariaDB 的情況下，需要的最低版本提高了，請參見[MySQL and MariaDB](#)。

在 4.4.1 版本變更：

- 單語 *gettext* now uses both `msgid` and `msgctxt` when present. This will change identification of translation strings in such files breaking links to Weblate extended data such as screenshots or review states. Please make sure you commit pending changes in such files prior upgrading and it is recommended to force loading of affected component using *loadpo*.
- 增加了 `translate-toolkit` 的最低需求版本，來處理幾個文件格式問題。

也參考：

[通用的升級說明](#)

### 從 4.4 升級到 4.5

請按照[通用的升級說明](#)來執行升級。

显著的配置与依赖性更改：

- 如果你有大詞表，遷移可能需要相當長的時間。
- Glossaries are now stored as regular components.
- 刪除詞表 API，使用定期翻譯 API 來訪問詞表。
- `:setting:django:INSTALLED_APPS` 中有一處更改，- “`weblate.metrics`” 應被添加。

在 4.5.1 版本變更：

- “Pyahocorasick” 模塊有一個新的依賴。

也參考：

[通用的升級說明](#)

### 從 4.5 升級到了 4.6

請按照[通用的升級說明](#)來執行升級。

显著的配置与依赖性更改：

- 有幾個新的文件格式，在修改 `WEBLATE_FORMATS` 的情況下，你會想要將他們包括進來。
- 創建組件的 API 現在自動使用 *Weblate internal URLs*，見：[http://post:/api/projects/\(string:project\)/components/](http://post:/api/projects/(string:project)/components/)。
- 依賴關係和：設置：`django: password_hashers` 更喜歡 `argon2` 以獲取密碼散列。

也參考：

[通用的升級說明](#)

## 從 4.6 升級到了 4.7

請按照[通用的升級 F 明](#) 來執行升級。

显著的配置与依赖性更改：

- 在 `settings_example.py` 中有几项更改，最显著的是中间件的更改 (`MIDDLEWARE`)，请由此调整你的设置。
- The *DeepL* machine translation now has a generic `MT_DEEPL_API_URL` setting to adapt to different subscription models more flexibly. The `MT_DEEPL_API_VERSION` setting is no longer used.
- Django 3.2 is now required.

**也參考：**

[通用的升級 F 明](#)

## Upgrade from 4.7 to 4.8

請按照[通用的升級 F 明](#) 來執行升級。

There are no additional upgrade steps needed in this release.

**也參考：**

[通用的升級 F 明](#)

## Upgrade from 4.8 to 4.9

請按照[通用的升級 F 明](#) 來執行升級。

- There is a change in storing metrics, the upgrade can take long time on larger sites.

**也參考：**

[通用的升級 F 明](#)

## Upgrade from 4.9 to 4.10

請按照[通用的升級 F 明](#) 來執行升級。

- 這是會影響每一專案的改動，這個改動會在數千個專案上，[F](#)在網站上花點時間執行。
- Django 4.0 has made some incompatible changes, see [Backwards incompatible changes in 4.0](#). Weblate still supports Django 3.2 for now, in case any of these are problematic. Most notable changes which might affect Weblate:
  - Dropped support for PostgreSQL 9.6, Django 4.0 supports PostgreSQL 10 and higher.
  - Format of `CSRF_TRUSTED_ORIGINS` was changed.
- The Docker container now uses Django 4.0, see above for changes.

**也參考：**

[通用的升級 F 明](#)

## Upgrade from 4.10 to 4.11

請按照通用的升級 明 來執行升級。

- Weblate now requires Python 3.7 or newer.
- The implementation of 管理單一專案的存取控制 has changed, removing the project prefix from the group names. This affects API users.
- Weblate now uses `charset-normalizer` instead of `chardet` module for character set detection.
- **Changed in 4.11.1:** There is a change in `REST_FRAMEWORK` setting (removal of one of the backends in `DEFAULT_AUTHENTICATION_CLASSES`).

也參考:

通用的升級 明

## Upgrade from 4.11 to 4.12

請按照通用的升級 明 來執行升級。

- There are no special steps required.

也參考:

通用的升級 明

## Upgrade from 4.12 to 4.13

請按照通用的升級 明 來執行升級。

- The 語言定義 are now automatically updated on upgrade, use `UPDATE_LANGUAGES` to disable that.
- Handling of context and location has been changed for *Windows RC files*, *HTML files*, *IDML Format*, and 文 字檔 file formats. In most cases the context is now shown as location.
- The machine translation services are now configured using the user interface, settings from the configuration file will be imported during the database migration.

也參考:

通用的升級 明

## Upgrade from 4.13 to 4.14

請按照通用的升級 明 來執行升級。

- The Java formatting checks now match GNU gettext flags. The flags set in Weblate will be automatically migrated, but third-party scripts will need to use `java-printf-format` instead of `java-format` and `java-format` instead of `java-messageformat`.
- The *jellyfish* dependency has been replaced by *rapidfuzz*.
- **Changed in 4.14.2:** Deprecated insecure configuration of VCS service API keys via `_TOKEN/_USERNAME` configuration instead of `_CREDENTIALS` list. In Docker, please add matching `_HOST` directive. For example see `WEBLATE_GITHUB_HOST` and `GITHUB_CREDENTIALS`.

也參考:

通用的升級 明

## Upgrade from 4.14 to 4.15

請按照通用的升級說明來執行升級。

- Weblate now requires `btree_gin` extension in PostgreSQL. The migration process will install it if it has sufficient privileges. See [建立 PostgreSQL 數據庫](#) for manual setup.
- The Docker image no longer enables debug mode by default. In case you want it, enable it in the environment using `WEBLATE_DEBUG`.
- The database migration may take hours on larger instances due to recreating some of the indexes.
- **Changed in 4.15.1:** The default value for `DEFAULT_PAGINATION_CLASS` in rest framework settings was changed.

也參考:

通用的升級說明

## 2.3.4 從 Python 2 升級到 Python 3

Weblate no longer supports Python older than 3.6. In case you are still running on older version, please perform migration to Python 3 first on existing version and upgrade later. See [Upgrading from Python 2 to Python 3 in the Weblate 3.11.1 documentation](#).

## 2.3.5 從其它數據庫遷移到 PostgreSQL

If you are running Weblate on other database than PostgreSQL, you should consider migrating to PostgreSQL as Weblate performs best with it. The following steps will guide you in migrating your data between the databases. Please remember to stop both web and Celery servers prior to the migration, otherwise you might end up with inconsistent data.

### 建立 PostgreSQL 數據庫

在另一個單獨的數據庫中運行 Weblate，將用戶名與密碼分開通常是個好方法：

```
# If PostgreSQL was not installed before, set the main password
sudo -u postgres psql postgres -c "\password postgres"

# Create a database user called "weblate"
sudo -u postgres createuser -D -P weblate

# Create the database "weblate" owned by "weblate"
sudo -u postgres createdb -E UTF8 -O weblate weblate
```

### 使用 Django JSON 轉儲來遷移

最簡單的遷移方法是使用 Django JSON 轉儲。這對於較小的安裝工作得很好。在更大的網站，你會想要使用 `pgloader` 代替，請參見 [使用 pgloader 遷移到 PostgreSQL](#)。

1. 添加 PostgreSQL 作為 `file:settings.py` 的另外的數據庫連接：

```
DATABASES = {
    "default": {
        # Database engine
        "ENGINE": "django.db.backends.mysql",
        # Database name
        "NAME": "weblate",
```

(繼續下一頁)

(繼續上一頁)

```

# Database user
"USER": "weblate",
# Database password
"PASSWORD": "password",
# Set to empty string for localhost
"HOST": "database.example.com",
# Set to empty string for default
"PORT": "",
# Additional database options
"OPTIONS": {
    # In case of using an older MySQL server, which has MyISAM as a
    ↪ default storage
    # 'init_command': 'SET storage_engine=INNODB',
    # Uncomment for MySQL older than 5.7:
    # 'init_command': "SET sql_mode='STRICT_TRANS_TABLES'",
    # If your server supports it, see the Unicode issues above
    "charset": "utf8mb4",
    # Change connection timeout in case you get MySQL gone away error:
    "connect_timeout": 28800,
},
},
"postgresql": {
    # Database engine
    "ENGINE": "django.db.backends.postgresql",
    # Database name
    "NAME": "weblate",
    # Database user
    "USER": "weblate",
    # Database password
    "PASSWORD": "password",
    # Set to empty string for localhost
    "HOST": "database.example.com",
    # Set to empty string for default
    "PORT": "",
},
}

```

2. 運行遷移，F 將任何插入到表格中的數據 drop 掉：

```

weblate migrate --database=postgresql
weblate sqlflush --database=postgresql | weblate dbshell --database=postgresql

```

3. 將遺留數據庫進行轉儲，F 導入 PostgreSQL

```

weblate dumpdata --all --output weblate.json
weblate loaddata weblate.json --database=postgresql

```

4. 調整 DATABASES 而只使用 PostgreSQL 數據庫作 F 默認，將遺留連接 F 除掉。

現在 Weblate 應該準備好從 PostgreSQL 數據庫運行了。

## 使用 pgloader 遷移到 PostgreSQL

pgloader 是通用遷移工具，將數據遷移到 PostgreSQL。你可以使用它來遷移 Weblate 數據庫。

1. 調整 `settings.py` 文件而將 PostgreSQL 用作數據庫。
2. 遷移 PostgreSQL 中的模式：

```
weblate migrate
weblate sqlflush | weblate dbshell
```

3. 運行 pgloader 來轉移數據。後面的脚本可以用於遷移數據庫，但你會想要學習更多關於 pgloader 的知識，來理解它做什麼以及調整它來匹配你的設置：

```
LOAD DATABASE
FROM      mysql://weblate:password@localhost/weblate
INTO      postgresql://weblate:password@localhost/weblate

WITH include no drop, truncate, create no tables, create no indexes, no_
foreign keys, disable triggers, reset sequences, data only

ALTER SCHEMA 'weblate' RENAME TO 'public'
;
```

### 2.3.6 Migrating from Pootle

因 Weblate 開始編寫出來替 Pootle，所以支持從 Pootle 遷移用戶賬。你可以將 Pootle 的用戶轉儲，使用 `importusers` 將他們導入。

## 2.4 備份和移動 Weblate

### 2.4.1 Project level backups

在 4.14 版本新加入。

**警告：** 恢復備份功能只支援使用 PostgreSQL 或 MariaDB 10.5+ 的資料庫。

專案會從 Weblate 備份所有翻譯內容（專案、組件、翻譯、字串評、建議或查核）。這適合用在轉移專案到另一個 Weblate 翻譯平台。

你可以執行一個專案的備份，透過 *Manage ↓ Backups*。而備份可以在建立新專案的時候恢復（參：添加翻譯項目和組件）。

目前備份資料不包含存取控制與歷史資訊。

評與建議會與其建立者之名稱一同備份。匯入時若有相符則指定到其使用者。若無相符使用者，將會指定匿名者名稱。

The generated backups are kept on the server as configured by `PROJECT_BACKUP_KEEP_DAYS` and `PROJECT_BACKUP_KEEP_COUNT` (it defaults to keep at most 3 backups for 30 days).

## 2.4.2 Automated backup using BorgBackup

在 3.9 版本新加入.

Weblate 設置了對使用 [BorgBackup](#) 創建服務備份的支持。Borg 創建了節省空間的加密備份，可以安全地存儲在雲中。可以從管理界面中的 *Backups* 選項卡上控制備份。

在 4.4.1 版本變更: PostgreSQL 和 MySQL/MariaDB 數據庫都包括在自動備份中。

使用 Borg 的備份是遞增的，Weblate 配置保留後面的備份：

- Daily backups for 14 days back
- 每周備份 8 回來
- 每月備份 6 個月回來

Weblate
 Dashboard Projects Languages Checks

Manage / Backups

Backup process triggered

Weblate status Backups Translation memory Performance report SSH keys Alerts Repositories Users Teams

Appearance Tools Automatic suggestions Billing

Backup service: /tmp/tmpwavy2px5weblate

Backup service credentials Dec. 16, 2022

Backup repository /tmp/tmpwavy2px5weblate

Passphrase uNUWnvM8M^Ja5mmT@HxuREDXn^v5wvU!%!r\*x\*VJS%44s\$PhzNG  
 The passphrase is used to encrypt the backups and is necessary to restore them.

SSH key Download private key  
 The private key is needed to access the remote backup repository.

Deleted the oldest backups Dec. 16, 2022

Backup performed Dec. 16, 2022

Repository initialization Dec. 16, 2022

Turn off Perform backup Delete

Activate support package

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

Activate Purchase support package

Add backup service

Backup repository URL

Use /path/to/repo for local backups or user@host:/path/to/repo or ssh://user@host:port/path/to/backups for remote SSH backups.

Add

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate



## Borg encryption key

BorgBackup 創建加密的備份，如果你有密碼，你將無法恢復它們。密碼是在添加新的備份服務時生成的，你應該將它將其保存在一個安全的地方。

如果你在使用 *Weblate provisioned backup storage*，請同樣備份你的私有 SSH 密鑰，因它用來訪問你的備份。

也參考：

`borg init`

## Customizing backup

- The database backup can be configured via `DATABASE_BACKUP`.
- The backup creation can be customized using `BORG_EXTRA_ARGS`.

### 2.4.3 Weblate provisioned backup storage

備份你的 Weblate 實例的最簡單方式是在 [weblate.org](https://weblate.org) 購買『備份服務』<<https://weblate.org/support/#backup>>。這是你如何讓它運行起來的方式：

1. 在 <https://weblate.org/support/#backup> 購買 備份服務。
2. 在管理界面輸入得到的密鑰，請參見集成支持。
3. Weblate 將連接到雲服務，獲取備份的訪問信息。
4. 從 *Backups* 標開新的備份配置。
5. Backup your Borg credentials to be able to restore the backups, see *Borg encryption key*.

---

**提示：**為了安全起見，有打開所有東西的手動步驟。只有你的同意，就不會有數據發送到通過步驟得到的備份倉庫。

---

### 2.4.4 使用客戶的備份存儲

也可以使用自己的存儲來備份。SSH 可以用於在遠程目的地存儲備份，目標服務器需要安裝 BorgBackup。

也參考：

General 在 Borg 文件中

## 本地文件系統

推薦去指定本地備份的對路徑，例如 `/path/to/backup`。該目錄必須可由運行 weblate 的用戶寫入（請參見文件系統權限）。在目錄不存在的情況下，Weblate 會嘗試新建它，但需要適當的權限才能這做。

---

**提示：**在 Docker 中運行 Weblate 時，請確保備份位置暴露來自 Weblate 容器的一個卷。否則，備份文件將在其所在的容器重啟時被 Docker 刪除。

一個選項是將備份放置在一個現有的目錄中。例如，`/app/data/borgbackup`。這是容器中的一個現有的目錄。

你也可以在 Docker 的編寫文件中備份目的添加一個新的容器，例如使用 `/borgbackup`：

```
services:
  weblate:
    volumes:
      - /home/weblate/data:/app/data
      - /home/weblate/borgbackup:/borgbackup
```

備份所存儲的目錄由 UID 1000 所有，否則 Weblate 將無法把備份寫入那。

---

## Remote backups

For creating remote backups, you will have to install [BorgBackup](#) onto another server that's accessible for your Weblate deployment via SSH using the Weblate SSH key:

1. Prepare a server where your backups will be stored.
2. Install the SSH server on it (you will get it by default with most Linux distributions).
3. Install [BorgBackup](#) on that server; most Linux distributions have packages available (see [Installation](#)).
4. Choose an existing user or create a new user that will be used for backing up.
5. Add Weblate SSH key to the user so that Weblate can SSH to the server without a password (see [Weblate SSH 密鑰](#)).
6. Configure the backup location in Weblate as `user@host:/path/to/backups` or `ssh://user@host:port/path/to/backups`.

---

**提示:** [Weblate provisioned backup storage](#) provides you automated remote backups without any effort.

---

也參考:

[Weblate SSH 密鑰](#), [General](#)

## 2.4.5 Restoring from BorgBackup

1. 恢復功能會訪問你的備份倉庫，準備備份密碼。
2. 用“`borg list REPOSITORY`”列出服務器上的所有備份。
3. 使用 `borg extract REPOSITORY::ARCHIVE` 將所需備份恢復到當前目錄。
4. 從放置在 Weblate 數據目錄下 backup 目錄中的 SQL 備份中恢復數據庫（請參見下載的數據用於備份）。
5. 將 Weblate 配置（`backups/settings.py`，請參見下載的數據用於備份）到正確的位置，請參見調整配置。

When using Docker container, the settings file is already included in the container and you should restore the original environment variables. The `environment.yml` file might help you with this (see 下載的數據用於備份).

6. 將整個存儲的數據目錄到 `DATA_DIR` 所配置的位置。

When using Docker container place the data into the data volume, see [Docker container volumes](#).

Please make sure the files have correct ownership and permissions, see [文件系統權限](#).

Borg 會話可能看上去是這個樣子的：

```
$ borg list /tmp/xxx
Enter passphrase for key /tmp/xxx:
2019-09-26T14:56:08          Thu, 2019-09-26 14:56:08
→ [de0e0f13643635d5090e9896bdaceb92a023050749ad3f3350e788f1a65576a5]
$ borg extract /tmp/xxx::2019-09-26T14:56:08
Enter passphrase for key /tmp/xxx:
```

也參考:

[borg list](#), [borg extract](#)

## 2.4.6 手動備份

取<sup>[1]</sup>於你想保存什<sup>[2]</sup>，Weblate 存儲的類型數據會備份在各自的位置。

**提示:** 如果你正進行手動備份，你也許想要關閉 Weblate 關於缺乏備份的警告，方法是添加 “weblate.I028” 到 `file:settings.py` 中的 `:setting:django:SILENCED_SYSTEM_CHECKS`；對於 Docker，則是 `envvar:WEBLATE_SILENCED_SYSTEM_CHECKS`。

```
SILENCED_SYSTEM_CHECKS.append("weblate.I028")
```

## 數據庫

實際存儲位置依賴於數據庫的設置。

**提示:** 數據庫是最重要的存儲。定期對數據庫進行備份。<sup>[3]</sup>有數據庫，所有的翻譯都會消失。

## Native database backup

推薦的方式是使用數據庫的本地工具如 `pg_dump` 或 `mysqldump` 來保存數據庫的轉儲。這通常比 Django 備份表現得好，<sup>[4]</sup>且可以連同數據一道，恢復完整表格。

你可以在較新的 Weblate 發行版中恢復這個備份，當運行於 `migrate` 時，它將執行所有必需的遷移。請參考[升級 Weblate](#) 了解如何在版本間升級的更多詳細信息。

## Django database backup

另外，可以使用 Django 的 `dumpdata` 命令備份你的數據庫。那種方式是不依托數據庫的，<sup>[5]</sup>且可以用於先要更改數據庫後端的情<sup>[6]</sup>。

恢復數據庫之前，你需要確保恢復備份和執行備份的實例運行的是完全相同的 Weblate 版本。這是必要的，因<sup>[7]</sup>數據庫結構在不同版本之間會發生變化，你可能會以某種方式破壞數據。安裝相同版本後，用 `migrate` 運行所有數據庫遷移。

之後，一些條目將已經在數據庫中創建，你也會在數據庫備份中看到它們。推薦的方法是使用管理 shell 手動<sup>[8]</sup>除這些條目（見：[ref:invoke-manage](#)）：

```
weblate shell
>>> from weblate.auth.models import User
>>> User.objects.get(username='anonymous').delete()
```

## 檔案

如果你有足夠的備份空間，只需備份整個 `DATA_DIR`。這是一個安全帶，即使它包含一些你不想要的文件。下面的部分詳細描述了應該備份和可以跳過的內容。

### 下載的數據用於備份

在 4.7 版本變更: The environment dump was added as `environment.yml` to help in restoring in the Docker environments.

存儲在 `DATA_DIR/backups` 中。

Weblate 這個備份各種數據，可以包括這些文件用於更完整的備份。文件每日更新（需要運行 Celery beat 服務器，請參見使用 *Celery* 的後台任務）。當前，這包括：

- Weblate 設置 `settings.py`（還有擴展版，在 `settings-expanded.py`）。
- PostgreSQL 數據庫備份 `database.sql`。
- Environment dump as `environment.yml`。

數據庫備份默認保存純文本，但也可以通過 `setting: 'DATABASE_BACKUP'` 來進行壓縮或整個跳過。

To restore the database backup load it using database tools, for example:

```
psql --file=database.sql weblate
```

### Version control repositories

存儲在 `DATA_DIR"/vcs"` 中。

版本控制存儲庫包含帶有 Weblate 更改的上游存儲庫的副本。如果你對所有翻譯組件用了 `:ref:component-push_on_commit`，那麼所有 Weblate 變化都包括在上游。不需要在 Weblate 端備份存儲庫，因為它們可以從上游位置再次克隆，不會丟失數據。

### SSH 和 GPG 密鑰

存儲在 `DATA_DIR/ssh` 和 `DATA_DIR/home` 中。

如果正在使用 Weblate 生成的 SSH 或 GPG 密鑰，你應該備份這些位置。否則將丟失私有密鑰，且你將不得不重新生成新的密鑰。

### User uploaded files

存儲在 `DATA_DIR/media` 中。

你應當備份所有有用上傳的文件（例如字符串的可見語境）。

## Celery 任務

Celery 任務隊列可能會包含一些信息，但通常無需進行備份。你最多會遺失尚未被翻譯記憶庫處理的更新。無論如何，建議在恢復時執行全文或存儲庫更新，這樣就不會有遺失這些內容的問題。

### 也參考：

使用 *Celery* 的後台任務

## 手動備份的命令行

使用 cron 作業，您可以設置一條每天執行的 Bash 命令，例如：

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz \
↪backups vcs ssh home media fonts secret
```

‘XZ\_OPT’後面引號之間的字符串允許你選擇自己的 xz 選項，例如用於壓縮的塊大小；見 <https://linux.die.net/man/1/xz>

你可以根據需要調整文件夾和文件的列表。為了避免保存翻譯記憶庫（在備份文件夾中），你可以使用：

```
$ XZ_OPT="-9" tar -Jcf ~/backup/weblate-backup-$(date -u +%Y-%m-%d_%H%M%S).xz \
↪backups/database.sql backups/settings.py vcs ssh home media fonts secret
```

## 2.4.7 Restoring manual backup

1. 將已經備份的所有數據恢復。
2. 使用 `updategit` 更新所有倉庫。

```
weblate updategit --all
```

## 2.4.8 移動 Weblate 安裝

按照上面備份與恢復說明，將你的安裝遷移到不同系統。

### 也參考：

從 *Python 2* 升級到 *Python 3*，從其它數據庫遷移到 *PostgreSQL*

## 2.5 身份核對

### 2.5.1 使用者

Weblate 的默認設置使用 `python-social-auth`，網站上處理新用戶的一種形式。確定電子郵箱後，新用戶可以通過使用一種第三方服務來貢獻或證實。

還可以使用 `REGISTRATION_OPEN` 關閉新用戶。

身份驗證嘗試服從於頻次限制。

## 2.5.2 身份驗證後台

Django 的 `django.contrib.auth` 方案用途是進行身份驗證，包括用各種社交登入選項進行驗證。使用它意味著可以導入基於 Django 其他項目的用 `django.contrib.auth` 數據庫（請參見 *Migrating from Poolle*）。

也可以另外新建 Django，相對於其他方式進行身份驗證。

也參考：

[身份驗證設置](#) 描述如何配置官方 Docker 鏡像的身份驗證。

## 2.5.3 社交身份驗證

由於 [Welcome to Python Social Auth's documentation!](#)，Weblate 支持很多使用第三方服務的身份驗證，如 GitLab、Ubuntu、Fedora 等。

請檢查 [Django Framework](#) 中的通用配置指示的文件。

**備註：** Weblate 默認依賴於第三方身份驗證服務來提供合法的電子郵件地址。如果想要使用的一些服務不支持，請通過其配置 `FORCE_EMAIL_VALIDATION`，來制 Weblate 網站上的電子郵件驗證：

```
SOCIAL_AUTH_OPENSUSE_FORCE_EMAIL_VALIDATION = True
```

也參考：

[Pipeline](#)

用單獨的後端非常簡單，只需添加一個條目至設置：`setting: django:AUTHENTICATION_BACKENDS` 即可（可能還需一個給定的驗證方式添加密鑰）請注意，一些後端默認不提供用電子郵件，你必須明確地請求，否則 Weblate 無法將功勞歸於作出貢獻的用。

**提示：** 大部分的後端授權需要 HTTPS。一旦用網頁服務上的 HTTPS，請設定 Weblate 環境參數 `ENABLE_HTTPS` 或 Docker 環境參數 `WEBLATE_ENABLE_HTTPS`。

也參考：

: doc: `python seamlial auth stend <psa: backends / index>`

## OpenID 驗證

對於基於 OpenID 的服務，通常只要用它們就行了。後面的部分關於對於 OpenSUSE、Fedora 和 Ubuntu 允許 OpenID 身份驗證：

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.suse.OpenSUSEOpenId",
    "social_core.backends.ubuntu.UbuntuOpenId",
    "social_core.backends.fedora.FedoraOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

也參考：

[OpenID](#)

## GitHub 驗證

需要在 GitHub 上[1][2]一個 OAuth 應用，然後告訴 Weblate 所有的 secrets：

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.github.GithubOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = "GitHub Client ID"
SOCIAL_AUTH_GITHUB_SECRET = "GitHub Client Secret"
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]
```

應該配置 GitHub 具有回調 URL 作[3] <https://example.com/accounts/complete/github/>。

這[4]有與 Github 組織版及團隊版類似的後端授權方式。他們的設定名稱[5] `SOCIAL_AUTH_GITHUB_ORG_*` 與 `SOCIAL_AUTH_GITHUB_TEAM_*` [6]且需要額外的設定參數 `SOCIAL_AUTH_GITHUB_ORG_NAME` 或 `SOCIAL_AUTH_GITHUB_TEAM_ID` 其返回應答網址（callback URLs）[7]: <https://example.com/accounts/complete/github-org/> 與 <https://example.com/accounts/complete/github-teams/>。

---

備[8]: Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

---

也參考：

GitHub

## Butbucket 驗證

需要在 Bitbucket 上[1][2]應用，然後告訴 Weblate 所有的秘密：

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.bitbucket.BitbucketOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = "Bitbucket Client ID"
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = "Bitbucket Client Secret"
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True
```

---

備[8]: Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

---

也參考：

Bitbucket

## Google OAuth 2

F 了使用 Google OAuth 2，可以在 <<https://console.developers.google.com/>> 上 F F 應用，F 允許 Google+ API。

重定向 URL F [https://WEBLATE\\_SERVER/accounts/complete/google-oauth2/](https://WEBLATE_SERVER/accounts/complete/google-oauth2/)

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.google.GoogleOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = "Client ID"
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = "Client secret"
```

---

備 F: Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

---

也參考:

[Google](#)

## Facebook 驗證 2

通常通過 OAuth2 服務，需要用 Facebook 來 F F 應用。一旦完成，就可以新建 Weblate 來使用了：

重定向 URL F [https://WEBLATE\\_SERVER/accounts/complete/facebook/](https://WEBLATE_SERVER/accounts/complete/facebook/)

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.facebook.FacebookOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_FACEBOOK_KEY = "key"
SOCIAL_AUTH_FACEBOOK_SECRET = "secret"
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
```

---

備 F: Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

---

也參考:

[Facebook](#)



## GitLab 驗證 2

為了使用 GitLab OAuth 2，需要在 <<https://gitlab.com/profile/applications>> 上應用。

重定向 URL 為 `https://WEBLATE_SERVER/accounts/complete/gitlab/`，確保你標記 `read_user` 範圍。

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.gitlab.GitLabOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_GITLAB_KEY = "Application ID"
SOCIAL_AUTH_GITLAB_SECRET = "Secret"
SOCIAL_AUTH_GITLAB_SCOPE = ["read_user"]

# If you are using your own GitLab
# SOCIAL_AUTH_GITLAB_API_URL = 'https://gitlab.example.com/'
```

備註：Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

也參考：

GitLab

## 微軟 Azure Active Directory

可以配置 Weblate，使用一般或特定租戶進行身份驗證。

常見的重定向 URL 為 “`https://WEBLATE_SERVER/accounts/complete/azuread-oauth2/`”，或 “`https://WEBLATE_SERVER/accounts/complete/azuread-tenant-oauth2/`” 用於租戶特定身份驗證。

```
# Azure AD common

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread.AzureADOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_OAUTH2_KEY = ""
SOCIAL_AUTH_AZUREAD_OAUTH2_SECRET = ""
```

```
# Azure AD Tenant

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.azuread_tenant.AzureADTenantOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# OAuth2 keys
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_KEY = ""
```

(繼續下一頁)

(繼續上一頁)

```
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_SECRET = ""
# Tenant ID
SOCIAL_AUTH_AZUREAD_TENANT_OAUTH2_TENANT_ID = ""
```

備註：Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

### 也參考：

Microsoft Azure Active Directory

## Slack

為了使用 Slack OAuth 2，需要在 <<https://api.slack.com/apps>> 上配置應用。

重定向 URL 為 `https://WEBLATE_SERVER/accounts/complete/slack/`。

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.slack.SlackOAuth2",
    "social_core.backends.email.EmailAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_SLACK_KEY = ""
SOCIAL_AUTH_SLACK_SECRET = ""
```

備註：Weblate 在身份驗證時提供的回調 URL。在得到 URL 不匹配的錯誤時，可以根據需要來修復，請參見設置正確的網站域名。

### 也參考：

Slack

## Overriding authentication method names and icons

你可以寫認證方式的顯示名稱與圖示使用 `SOCIAL_AUTH_<NAME>_IMAGE` 與 `SOCIAL_AUTH_<NAME>_TITLE` 參數設定。例如寫 Auth0 的名稱可以看起來像這樣：

```
SOCIAL_AUTH_AUTH0_IMAGE = "custom.svg"
SOCIAL_AUTH_AUTH0_TITLE = "Custom auth"
```

## 關閉密碼身份驗證

通過從 `AUTHENTICATION_BACKENDS` 移除 `social_core.backends.email.EmailAuth`，可以關閉電子郵件和密碼身份驗證。總是將 `weblate.accounts.auth.WeblateUserBackend` 保留在那裡，它用於 Weblate 核心功能。

Disabling e-mail authentication will disable all e-mail related functionality – user invitation or password reset feature.

**小訣竅：** You can still use password authentication for the admin interface, for users you manually create there. Just navigate to `/admin/login/`.

例如，使用後面的設置可以實現只是用 openSUSE Open ID 的身份驗證：

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.suse.OpenSUSEOpenId",
    "weblate.accounts.auth.WeblateUserBackend",
)
```

## 2.5.4 密碼驗證

默認 settings.py 與一組合理的設置 AUTH\_PASSWORD\_VALIDATORS 在一起：

- 密碼不能與其它個人信息太相似。
- 密碼需至少 10 個字元。
- 密碼不能是常見的密碼。
- 密碼不能只含數字。
- 密碼不能包括單個字符或只有空白字符。
- 密碼與你過去使用的密碼不匹配。

可以自定義這個設置來匹配密碼政策。

可以另外安裝 `django-zxcvbn-password` 這會非常實際地估計密碼的難度，允許拒絕低於下面適當值的密碼。

## 2.5.5 SAML 身份驗證

在 4.1.1 版本新加入。

請遵守 Python Social Auth 的指示來配置。显著的差异有：

- Weblate 支持單一 IDP，在 SOCIAL\_AUTH\_SAML\_ENABLED\_IDPS 中被稱 weblate。
- SAML XML 元數據 URL `/accounts/metadata/saml/`。
- 後面的設置自動填入：SOCIAL\_AUTH\_SAML\_SP\_ENTITY\_ID、SOCIAL\_AUTH\_SAML\_TECHNICAL\_CONTACT、SOCIAL\_AUTH\_SAML\_SUPPORT\_CONTACT

配置的例子：

```
# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    "social_core.backends.saml.SAMLAuth",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Social auth backends setup
SOCIAL_AUTH_SAML_SP_ENTITY_ID = f"https://{SITE_DOMAIN}/accounts/metadata/saml/"
SOCIAL_AUTH_SAML_SP_PUBLIC_CERT = "-----BEGIN CERTIFICATE-----"
SOCIAL_AUTH_SAML_SP_PRIVATE_KEY = "-----BEGIN PRIVATE KEY-----"
SOCIAL_AUTH_SAML_ENABLED_IDPS = {
    "weblate": {
        "entity_id": "https://idp.testshib.org/idp/shibboleth",
        "url": "https://idp.testshib.org/idp/profile/SAML2/Redirect/SSO",
        "x509cert": "MIIEDjCCAvagAwIBAgIBADA ... 8Bbn1+ev0peYzxFyF5sQA=",
        "attr_name": "full_name",
        "attr_username": "username",
        "attr_email": "email",
    }
}
```

(繼續下一頁)

(繼續上一頁)

```

}
SOCIAL_AUTH_SAML_ORG_INFO = {
    "en-US": {
        "name": "example",
        "displayname": "Example Inc.",
        "url": "http://example.com"
    }
}
SOCIAL_AUTH_SAML_TECHNICAL_CONTACT = {
    "givenName": "Tech Gal",
    "emailAddress": "technical@example.com"
}
SOCIAL_AUTH_SAML_SUPPORT_CONTACT = {
    "givenName": "Support Guy",
    "emailAddress": "support@example.com"
}

```

預設的設定檔會萃取出使用者資訊透過以下的屬性值，請設定你的 IDP 來提估給他們：

屬性值	SAML URI reference
全名	urn:oid:2.5.4.3
名字	urn:oid:2.5.4.42
Last name	urn:oid:2.5.4.4
郵件信箱	urn:oid:0.9.2342.19200300.100.1.3
使用者名稱	urn:oid:0.9.2342.19200300.100.1.1

**提示：** 在以上的範例中與被定義 IDP 的 Docker 映像檔名 weblate。你可能需要設定此字串 Relay 在你的 IDP 中。

### 也參考：

：參考：“在 Docker <docker-collect>‘中配置收集，： doc: psa: beftends / collect

## 2.5.6 LDAP 身份驗證

LDAP 身份驗證可以使用 *django-auth-ldap* 軟件包而最好地實現。可以使用通常的方式安裝：

```

# Using PyPI
pip install django-auth-ldap>=1.3.0

# Using apt-get
apt-get install python-django-auth-ldap

```

**提示：** 此包包含於 Docker 容器中，見 *Installing using Docker*。

**備：** 在 Python LDAP 3.1.0 模塊中有一些不兼容，導致可能無法使用那個版本。如果得到錯誤信息 ‘AttributeError: [‘module’] object has no attribute [‘\_trace\_level’]’ <<https://github.com/python-ldap/python-ldap/issues/226>>\_，將 python-ldap 降回到 3.0.0 版可能會有幫助。

一旦安裝了軟件包，就可以將其入 Django 身份驗證了：

```

# Add LDAP backed, keep Django one if you want to be able to sign in
# even without LDAP for admin account
AUTHENTICATION_BACKENDS = (
    "django_auth_ldap.backend.LDAPBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# LDAP server address
AUTH_LDAP_SERVER_URI = "ldaps://ldap.example.net"

# DN to use for authentication
AUTH_LDAP_USER_DN_TEMPLATE = "cn=%(user)s,o=Example"
# Depending on your LDAP server, you might use a different DN
# like:
# AUTH_LDAP_USER_DN_TEMPLATE = 'ou=users,dc=example,dc=com'

# List of attributes to import from LDAP upon sign in
# Weblate stores full name of the user in the full_name attribute
AUTH_LDAP_USER_ATTR_MAP = {
    "full_name": "name",
    # Use the following if your LDAP server does not have full name
    # Weblate will merge them later
    # 'first_name': 'givenName',
    # 'last_name': 'sn',
    # Email is required for Weblate (used in VCS commits)
    "email": "mail",
}

# Hide the registration form
REGISTRATION_OPEN = False

```

備 註： 你 應 當 從 設 置 的 `setting:django:AUTHENTICATION_BACKENDS` 部 分 移 除 `social_core.backends.email.EmailAuth`， 否 則 用 它 不 能 在 Weblate 中 設 置 他 們 的 密 碼， 使 用 它 進 行 身 份 驗 證。 了 生 成 權 限 和 方 便 匿 名 用 戶， 仍 需 保 留 `weblate.accounts.auth.WeblateUserBackend`。 它 還 允 許 你 使 用 一 個 本 地 管 理 帳 登 陸， 如 果 你 已 經 創 建 了 它（如， 通 過 使 用 `:djadmin:createadmin`）。

## 使用 Bind Password

如果 可 以 身 份 驗 證 使 用 直 接 綁 定， 那 麼 需 要 使 用 搜 索， 用 搜 索 提 供 綁 定， 例 如：

```

import ldap
from django_auth_ldap.config import LDAPSearch

AUTH_LDAP_BIND_DN = ""
AUTH_LDAP_BIND_PASSWORD = ""
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "ou=users,dc=example,dc=com", ldap.SCOPE_SUBTREE, "(uid=%(user)s)"
)

```

## Active Directory integration

```
import ldap
from django_auth_ldap.config import LDAPSearch, NestedActiveDirectoryGroupType

AUTH_LDAP_BIND_DN = "CN=ldap,CN=Users,DC=example,DC=com"
AUTH_LDAP_BIND_PASSWORD = "password"

# User and group search objects and types
AUTH_LDAP_USER_SEARCH = LDAPSearch(
    "CN=Users,DC=example,DC=com", ldap.SCOPE_SUBTREE, "(sAMAccountName=%(user)s)"
)

# Make selected group a superuser in Weblate
AUTH_LDAP_USER_FLAGS_BY_GROUP = {
    # is_superuser means user has all permissions
    "is_superuser": "CN=weblate_AdminUsers,OU=Groups,DC=example,DC=com",
}

# Map groups from AD to Weblate
AUTH_LDAP_GROUP_SEARCH = LDAPSearch(
    "OU=Groups,DC=example,DC=com", ldap.SCOPE_SUBTREE, "(objectClass=group)"
)
AUTH_LDAP_GROUP_TYPE = NestedActiveDirectoryGroupType()
AUTH_LDAP_FIND_GROUP_PERMS = True

# Optionally enable group mirroring from LDAP to Weblate
# AUTH_LDAP_MIRROR_GROUPS = True
```

也參考:

Django Authentication Using LDAP, Authentication

## 2.5.7 CAS 身份驗證

可以使用軟件包如 *django-cas-ng* 來實現 CAS 身份驗證。

第一步通過 CAS 揭示了用 電子郵箱字段。這必須在 CAS 服務器自身來配置，需要至少運行 CAS v2，因 CAS v1 不支持屬性。

第二步更新 Weblate，來使用 CAS 服務器和屬性。

安裝 *django-cas-ng*:

```
pip install django-cas-ng
```

一旦安裝了軟件包，就可以通過修改 `settings.py` 文件將其連到 Django 身份驗證系統：

```
# Add CAS backed, keep the Django one if you want to be able to sign in
# even without LDAP for the admin account
AUTHENTICATION_BACKENDS = (
    "django_cas_ng.backends.CASBackend",
    "weblate.accounts.auth.WeblateUserBackend",
)

# CAS server address
CAS_SERVER_URL = "https://cas.example.net/cas/"

# Add django_cas_ng somewhere in the list of INSTALLED_APPS
INSTALLED_APPS = (... , "django_cas_ng")
```

最後，可以使用信號將電子郵箱字段投射到用 對象上。了生效，必須將信號從 *django-cas-ng* 軟件包導入，將你的代碼與這個信號連接。在設置文件中這樣做可能生問題，這樣建議將它放進去：

- 在你的 app 配置的 `django.apps.AppConfig.ready()` 方法
- 在項目的 `urls.py` 文件中（當有模塊存在時）

```
from django_cas_ng.signals import cas_user_authenticated
from django.dispatch import receiver

@receiver(cas_user_authenticated)
def update_user_email_address(sender, user=None, attributes=None, **kwargs):
    # If your CAS server does not always include the email attribute
    # you can wrap the next two lines of code in a try/catch block.
    user.email = attributes["email"]
    user.save()
```

也參考:

Django CAS NG <<https://github.com/django-cas-ng/django-cas-ng>> ‘Django CAS NG

## 2.5.8 配置第三方 Django 身份驗證

一般地，任何 Django 身份認證插件應該可以在 Weblate 上工作。只需要按照插件的說明，只記住安裝了 Weblate 用後台。

也參考:

LDAP 身份驗證, CAS 身份驗證

典型的安裝包括，將身份驗證後台添加到 `AUTHENTICATION_BACKENDS`，將身份驗證 app（如果有的話）安裝到 `INSTALLED_APPS`：

```
AUTHENTICATION_BACKENDS = (
    # Add authentication backend here
    "weblate.accounts.auth.WeblateUserBackend",
)

INSTALLED_APPS += (
    # Install authentication app here
)
```

## 2.6 存取控制

Weblate 具有一個精細的權利系統，可以整個實例或在有限範圍分配使用者權限。

在 3.0 版本變更: 在 Weblate 3.0 之前的權利系統是基於 Django 的，但現在是專門 Weblate 構建的。如果你使用的是舊版本，請查該版本的文件，此處的資訊將不適用。

### 2.6.1 簡單的訪問控制

如果你不是管理整個 Weblate 安裝，而只是有權限管理某些項目（比如在 ‘管 Weblate <<https://hosted.weblate.org/>>’），你的存取控制管理選項僅限於以下設定。如果你不需要任何複雜的設定，這些就足够了。

## 專案存取控制

備註：此功能不適用於運行在 Weblate 伺服器的 Libre 方案。

您可以透過選擇不同的 `guilabel:‘存取控制’` 設定來限制使用者對單個項目的存取。可用的選項有：

### 公開

公開可見，但只有已經登入的使用者可以翻譯。

### 受保護

公開可見，但僅對選定使用者可翻譯。

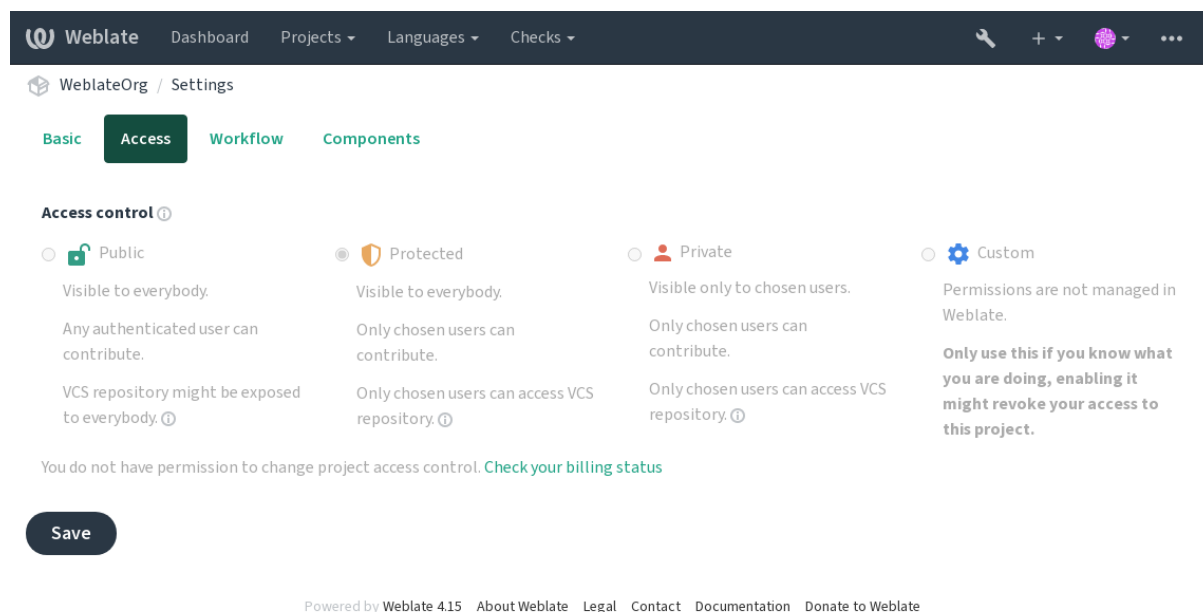
### 私人

僅選定使用者可以瀏覽和翻譯。

### 自訂

`:ref:‘使用者管理 <manage-acl>’` 功能將被停用；預設情況下，所有使用者都被禁止在專案上進行任何操作。你必須使用 `:ref:‘custom-acl’` 設定所有的權限。

可以在每個項目的配置（`guilabel: ‘Manage’`、`guilabel: ‘Settings’`）的 `guilabel: ‘Access’` 選項卡中更改 Access 控制。



可以透過 `DEFAULT_ACCESS_CONTROL` 修改預設值。

備註：即使對於“私有”項目，也會公開有關項目的一些信息：儘管具有訪問控制設置，所有項目的計數都會包含在整個實例的統計信息和語言摘要頁面。你的項目名稱和其他信息不會暴露。

備註：項目中默認提供“公開”，“受保護”，及“私有”的用戶權限組合，Weblate 的實例管理員也可以使用 `:ref:‘custom settings <custom-acl>’` 進行個性化定制。

## 也參考：

存取控制



## 管理單一專案的存取控制

Users with the *Manage project access* privilege (see [特殊權限列表與建立角色](#)) can manage users in projects via adding them to the teams. The initial collection of teams is provided by Weblate, but additional ones can be defined providing more fine-grained access control. You can limit teams to languages and assign them designated access roles (see [特殊權限列表與建立角色](#)).

以下的團隊將隨著每一專案自動建立：

於‘公開’、‘保護’與‘隱私’專案：

### 管理

Includes all permissions available for the project.

### 查 (僅限:ref:review workflow <reviews>設置打開)

可以在復查時批准翻譯。

只限‘受保護’及‘私有’項目：

### 翻譯

可以翻譯項目，將離的翻譯上傳。

### 原文

可以編輯源字符串 (如果在 *project settings* 中允許的話) 及源字符串信息。

### 語言

Can manage translated languages (add or remove translations).

### 詞表

Can manage glossary (add or remove entries, also upload).

### 記憶

Can manage translation memory.

### 畫面快照

Can manage screenshots (add or remove them, and associate them to source strings).

### 自動翻譯

Can use automatic translation.

### VCS

可以管理版本控制系統 (VCS) 訪問導出的倉庫。

### 帳單

可以訪問賬單信息和設置 (請參見 [帳單](#))。

The screenshot shows the Weblate interface with the 'Access control' section active. It includes tabs for 'Users', 'Teams', and 'API access'. The 'Users' tab displays a table of users with columns for Username, Full name, E-mail, Last sign in, and Teams. A user named 'testuser' is listed. Below the table is a note: 'Once all its permissions are removed, the user will be removed from the project.' There are three main panels for user management: 'Add a user' (with a text input and 'Add' button), 'Block user' (with a text input, 'Block duration' dropdown, and 'Block' button), and 'Invite new user' (with 'E-mail', 'Username', and 'Full name' inputs, and an 'Invite' button). The footer contains links: 'Powered by Weblate 4.15', 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

這些功能可在 *Access control* 頁面上找到，頁面訪問路徑 F 項目 menu *Manage* ↓ *Users*。

## 團隊管理員

在 4.15 版本新加入。

每一團隊可以擁有團隊的管理員，他可以新增或移除團隊中的成員。對於想要建立自治的團隊是很有用的。

## New user invitation

Also, besides adding an existing user to the project, it is possible to invite new ones. Any new user will be created immediately, but the account will remain inactive until signing in with a link in the invitation sent via an e-mail. It is not required to have any site-wide privileges in order to do so, access management permission on the project's scope (e.g. a membership in the *Administration* team) would be sufficient.

**提示：** 如果被邀請的用 F 錯過了邀請的有效性，則可以在密碼重置表單中使用被邀請的電子郵件地址設置密碼，因 F 已經創建了帳 F。

在 3.11 版本新加入: 重新發送用 邀請電子郵件是有可能的 (使任何之前發送的邀請無效)。

同樣的邀請函數可以從管理界面 :guilabel: 用 標 。

## Blocking users

在 4.7 版本新加入.

避免一些行惡意的使用者在您的專案中，您可以限制他們貢獻。被限制的使用者若有權限依舊能看到專案，但是他不能做出貢獻。

## 單一專案權限管理

將專案設定 保護、隱私 與 :ref:manage users <manage-acl> 在 Weblate 介面中管理每個專案的使用者。

By default this prevents Weblate from granting access provided by *Users* and *Viewers* *default teams* due to these teams' own configuration. This doesn't prevent you from granting permissions to those projects site-wide by altering default teams, creating a new one, or creating additional custom settings for individual component as described in 自訂存取控制 below.

One of the main benefits of managing permissions through the Weblate user interface is that you can delegate it to other users without giving them the superuser privilege. In order to do so, add them to the *Administration* team of the project.

## 2.6.2 自訂存取控制

---

備: 此功能不適用於運行在 Weblate 伺服器的 Libre 方案。

---

The permission system is based on teams and roles, where roles define a set of permissions, and teams link them to users and translations, see *Users, roles, teams, and permissions* for more details.

Weblate's Access Control 系統的最 大功能僅通過以下方式提供: REF: *Django* 管理員界面 <admin-interface>. 您可以使用它來管理任何項目的權限。您不一定必須將其切到 “自定義”: REF: “訪問控制 <ACL>” 使用它。但是，您必須具有超級用 權限才能使用它。

如果您對實現的詳細信息不感興趣, 且只想根據默認值創建一個簡單的配置, 或者 有對整個 WebLlate 安裝的站點廣泛訪問 (如 “管的 WebLlate <https://hosted.weblate.org/>”), 請參: REF: “訪問 - 簡單” 部分。

## 常見設定

此部分包含了一些您可能感興趣的常用配置選項的概覽。

## Site-wide permission management

To manage permissions for a whole instance at once, add users to appropriate *default teams*:

- *Users* (this is done by default by the *automatic team assignment*).
- 審稿人員 (如果您使用的是: REF: “評論工作流程 <評論>” 與專用評論者)。
- “經理” (如果您想將大多數管理操作委派給其他人)。

You should keep all projects configured as *Public* (see 專案存取控制), otherwise the site-wide permissions provided by membership in the *Users* and *Reviewers* teams won't have any effect.

You may also grant some additional permissions of your choice to the default teams. For example, you may want to give a permission to manage screenshots to all the *Users*.

You can define some new custom teams as well. If you want to keep managing your permissions site-wide for these teams, choose an appropriate value for the *Project selection* (e.g. *All projects* or *All public projects*).

### Custom permissions for languages, components or projects

您可以創建自己的專用團隊來管理不同對象（如語言，組件和項目）的權限。您不能透過新增一組自定義的團隊來撤銷任何來自全站設定的權限或某一團隊的權限，即使這些團隊只能授予額外的權限。

#### 示例：

如果您想要（無論出於何種原因）允許翻譯到特定語言（讓我們“捷克語”）僅在一組封閉式的可靠翻譯器，同時保持翻譯到其他語言，您將不得不：

1. Remove the permission to translate *Czech* from all the users. In the default configuration this can be done by altering the *Users default team*.

表格 1: Group *Users*

語言選取	<i>As defined</i>
語言	除 捷克語以外

2. Add a dedicated team for *Czech* translators.

表格 2: Group *Czech translators*

角色	<i>Power users</i>
專案選取	<i>All public projects</i>
語言選取	<i>As defined</i>
語言	<i>Czech</i>

3. Add users you wish to give the permissions to into this team.

正如您所看到的，這種方式的權限管理是大的，但可能是相當繁瑣的工作。除非授予超級用權限，否則您無法將其委派給另一個用。

### Users, roles, teams, and permissions

身份驗證模型包括幾個對象：

#### 許可

Weblate 定義的個人權限。權限不能分配給用。這只能通過分配角色來完成。

#### 角色

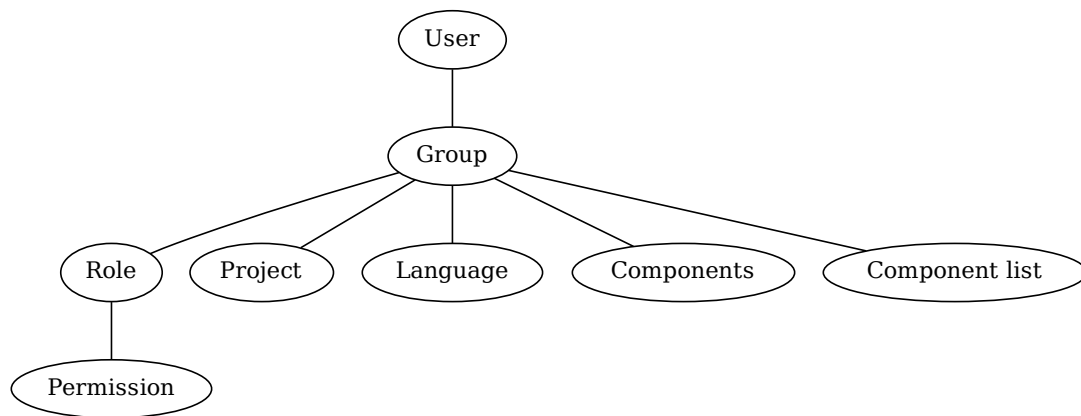
角色定義一組權限。這允許在幾個地方重用這些組，使管理更容易。

#### 使用者

User can belong to several teams.

#### 群組

用組連接角色、用和身份驗證對象（項目、語言和組件列表）。



備註: A team can have no roles assigned to it, in that case access to browse the project by anyone is assumed (see below).

### Access for browse to a project

A user has to be a member of a team linked to the project, or any component inside that project. Having membership is enough, no specific permissions are needed to browse the project (this is used in the default *Viewers* team, see 隊列表).

### Access for browse to a component

用 一旦能 訪問組件的項目，就可以不受限制地訪問組件。（將擁有該項目授予用的所有權限）。在開 受限制的訪問 的情況下，訪問組件需要對該組件（或該組件所在的組件列表）具有顯式權限。

### Scope of teams

The scope of the permission assigned by the roles in the teams are applied by the following rules:

- If the team specifies any *Component list*, all the permissions given to members of that team are granted for all the components in the component lists attached to the team, and an access with no additional permissions is granted for all the projects these components are in. *Components* and *Projects* are ignored.
- If the team specifies any *Components*, all the permissions given to the members of that team are granted for all the components attached to the team, and an access with no additional permissions is granted for all the projects these components are in. *Projects* are ignored.
- Otherwise, if the team specifies any *Projects*, either by directly listing them or by having *Projects selection* set to a value like *All public projects*, all those permissions are applied to all the projects, which effectively grants the same permissions to access all projects *unrestricted components*.
- The restrictions imposed by a team's *Languages* are applied separately, when it's verified if a user has an access to perform certain actions. Namely, it's applied only to actions directly related to the translation process itself like reviewing, saving translations, adding suggestions, etc.

提示: 使用 *Language selection* 或 *Project selection* 來自動包括所有語言或項目。

示例:

Let's say there is a project `foo` with the components: `foo/bar` and `foo/baz` and the following team:

表格 3: 群組 西班牙語管理審查員

角色	審核字串, 管理儲存庫
組件	foo/bar
語言	Spanish

Members of that team will have following permissions (assuming the default role settings):

- 一般 (F覽) 訪問整個項目 “foo”，包括它的兩個組件: `foo / bar`` 和 ``foo / baz`。
- 審核字串 `foo/bar` 在西班牙語翻譯中（不在其他地方）。
- 管理整個 “foo / bar” 的 VCS。提交待定的翻譯人員對所有語言進行的更改。

## Automatic team assignments

On the bottom of the *Group* editing page in the *Django admin interface*, you can specify *Automatic team assignments*, which is a list of regular expressions used to automatically assign newly created users to a team based on their e-mail addresses. This assignment only happens upon account creation.

The most common use-case for the feature is to assign all new users to some default team. In order to do so, you will probably want to keep the default value (`^.*$`) in the regular expression field. Another use-case for this option might be to give some additional privileges to employees of your company by default. Assuming all of them use corporate e-mail addresses on your domain, this can be accomplished with an expression like `^.*@mycompany.com`.

---

**備 F:** Automatic team assignment to *Users* and *Viewers* is always recreated when upgrading from one Weblate version to another. If you want to turn it off, set the regular expression to `^$` (which won't match anything).

---



---

**備 F:** As for now, there is no way to bulk-add already existing users to some team via the user interface. For that, you may resort to using the *REST API*.

---

## Default teams and roles

After installation, a default set of teams is created (see [團隊列表](#)).

These roles and teams are created upon installation. The built-in roles are always kept up to date by the database migration when upgrading. You can't actually change them, please define a new role if you want to define your own set of permissions.

## 特殊權限列表與 F 建角色

範圍	權限	角色
帳單 (參 F 帳單)	檢視帳單資訊	<i>Administration, Billing</i>
更動	下載更動處	<i>Administration</i>
評 F	張貼評 F	管理權限、編輯來源、超級使用者、審核字串、翻譯
	F 除評 F	<i>Administration</i>
	解 F 評 F	管理權限、審核字串
組件	編輯組件設定	<i>Administration</i>
	鎖定組件, 防止翻譯	<i>Administration</i>

表格 4 – 繼續上一頁

範圍	權限	角色
詞彙表	增加詞彙表條目	Administration, Manage glossary, Power user
	編輯詞彙表條目	Administration, Manage glossary, Power user
	刪除詞彙表條目	Administration, Manage glossary, Power user
	上傳詞彙表條目	Administration, Manage glossary, Power user
自動建議	使用自動建議	管理權限、編輯來源、超級使用者、審核字串、翻譯
翻譯記憶	編輯翻譯記憶	Administration, Manage translation memory
	刪除翻譯記憶	Administration, Manage translation memory
專案	編輯專案設定	Administration
	管理專案存取權	Administration
回報	下載報表	Administration
畫面快照	加入畫面快照	Administration, Manage screenshots
	編輯畫面快照	Administration, Manage screenshots
	刪除畫面快照	Administration, Manage screenshots
來源字串	編輯額外字串資訊	Administration, Edit source
字串	新增新字串	Administration
	移除字串	Administration
	略過未通過查核	管理權限、編輯來源、超級使用者、審核字串、翻譯
	編輯字串	管理權限、編輯來源、超級使用者、審核字串、翻譯
	檢閱字串	管理權限、審核字串
	當施行建議時編輯字串	管理權限、審核字串
	編輯來源字串	Administration, Edit source, Power user
建議	接受建議	管理權限、編輯來源、超級使用者、審核字串、翻譯
	新增建議	管理權限、編輯來源、新增建議、超級使用者、審核字串、翻譯
	刪除建議	Administration, Power user
	建議的投票	管理權限、編輯來源、超級使用者、審核字串、翻譯
翻譯	加入新語言的翻譯	Administration, Power user, Manage languages
	執行自動翻譯	Administration, Automatic translation
	刪除既有的翻譯	Administration, Manage languages
	下載翻譯檔案	管理權限、編輯來源、存取儲存庫、超級使用者、審核字串、翻譯、
	加入多種新語言的翻譯	Administration, Manage languages
上傳	定義上傳翻譯的作者	Administration
	以上傳內容覆蓋現在的翻譯	管理權限、編輯來源、超級使用者、審核字串、翻譯
	上傳翻譯	管理權限、編輯來源、超級使用者、審核字串、翻譯
VCS	存取內部儲存庫	Administration, Access repository, Power user, Manage repository
	將更動提交到內部儲存庫	Administration, Manage repository
	從內部儲存庫推入更動	Administration, Manage repository
	重設內部儲存庫中的更動	Administration, Manage repository
	檢視上游儲存庫的位置	Administration, Access repository, Power user, Manage repository
	更新內部儲存庫	Administration, Manage repository
全網站範圍的特權	使用管理介面	
	加入新的專案	
	加入語言定義	
	管理語言定義	
	管理團隊	
	管理使用者	
	管理角色	
	管理公告	
	管理翻譯記憶	
	管理機器翻譯	
	管理組件列表	

備註：站點範圍的特權不會被授予任何默認角色。它們功能強大，非常接近超級用戶的地位。它們中的

大多數都會影響到你的 Weblate 安裝中的所有項目。

---

## 團隊列表

The following teams are created upon installation (or after executing *setupgroups*) and you are free to modify them. The migration will, however, re-create them if you delete or rename them.

### 訪客

Defines permissions for non-authenticated users.

This team only contains anonymous users (see *ANONYMOUS\_USER\_NAME*).

You can remove roles from this team to limit permissions for non-authenticated users.

默認角色: *Add suggestion, Access repository*

### 審查員

This role ensures visibility of public projects for all users. By default, all users are members of this team.

By default, *automatic team assignment* makes all new accounts members of this team when they join.

預設角色: 無

### 使用者

Default team for all users.

By default, *automatic team assignment* makes all new accounts members of this team when they join.

默認角色: *Power user*

### 審核員

審核員專用的群組（參閱翻譯工作流）。

預設角色: 審核字串

### 管理員

管理員專用群組。

預設角色: 管理權限

**警告:** Never remove the predefined Weblate teams and users as this can lead to unexpected problems! If you have no use for them, you can removing all their privileges instead.

## 2.6.3 Additional access restrictions

If you want to use your Weblate installation in a less public manner, i.e. allow new users on an invitational basis only, it can be done by configuring Weblate in such a way that only known users have an access to it. In order to do so, you can set *REGISTRATION\_OPEN* to *False* to prevent registrations of any new users, and set *REQUIRE\_LOGIN* to */.\** to require signing in to access all the site pages. This is basically the way to lock your Weblate installation.

---

**提示:** 您可以使用 建立的 `:ref:invite-user` 新增使用者。

---



## 2.7 翻譯專案數

### 2.7.1 翻譯組織

項目/組件的可翻譯的版本控制系統（VCS）內容，由 Weblate 組織成樹狀結構。

- 底層對象是項目配置，該項目配置應將所有翻譯歸在一起（例如，多個版本的應用程序翻譯和/或隨附的文檔）。
- 在上面的級上，`:ref:component`（實際上是要翻譯的組件），您定義要使用的版本控制系統（VCS）倉庫以及要翻譯的文件的掩碼。
- 在`:ref:component`上方，有單獨的翻譯，當版本控制系統（VCS）倉庫中出現翻譯文件（與`:ref:component`中定義的`:ref:component-filemask`匹配）時，Weblate 會自動處理這些翻譯。

Weblate 支持 Translate Toolkit 支持的多種翻譯格式（雙語和單語），請參閱[支持的文件格式](#)。

**備註：** 您可以使用 [Weblate internal URLs](#) 共享克隆的版本控制系統（VCS）倉庫。當您有許多共享同一版本控制系統（VCS）的組件時，[強烈推薦](#)使用此功能。它提高了性能並[減少](#)了所需的磁盤空間。

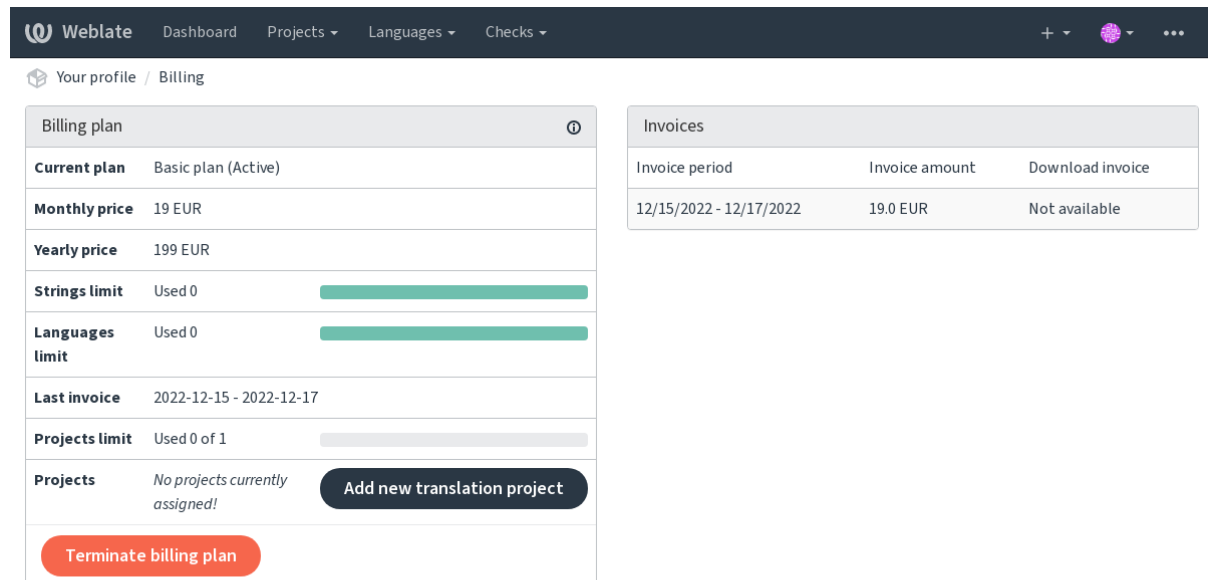
### 2.7.2 添加翻譯項目和組件

在 3.2 版本變更：已包含用於添加項目和組件的界面，您不再需要使用 [Django 管理界面](#)。

在 3.4 版本變更：現在，添加組件的過程是多階段的，可以自動發現大多數參數。

根據你的權限，新的翻譯項目和組件可以被創建。具備`guilabel:Add new projects`權限的用戶總是可以這做。如果使用付費賬管，你還可以從管理賬單的用戶賬基於套餐限額創建它們。

您可以在單獨的頁面上查看當前的結算方案：



The screenshot shows the Weblate web interface. At the top is a navigation bar with 'Weblate', 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below this is a sub-header 'Your profile / Billing'. The main content area is divided into two panels. The left panel, titled 'Billing plan', shows the 'Current plan' as 'Basic plan (Active)', 'Monthly price' as '19 EUR', and 'Yearly price' as '199 EUR'. It also displays usage for 'Strings limit' and 'Languages limit' as 'Used 0' with progress bars. The 'Last invoice' is for the period '2022-12-15 - 2022-12-17'. The 'Projects limit' is 'Used 0 of 1'. At the bottom of this panel is a red button 'Terminate billing plan'. The right panel, titled 'Invoices', shows a table with columns 'Invoice period', 'Invoice amount', and 'Download invoice'. The first row shows the period '12/15/2022 - 12/17/2022', amount '19.0 EUR', and 'Not available' for download. At the bottom of the right panel is a dark blue button 'Add new translation project'.

Billing plan	
Current plan	Basic plan (Active)
Monthly price	19 EUR
Yearly price	199 EUR
Strings limit	Used 0
Languages limit	Used 0
Last invoice	2022-12-15 - 2022-12-17
Projects limit	Used 0 of 1
Projects	No projects currently assigned!
<a href="#">Add new translation project</a>	
<a href="#">Terminate billing plan</a>	

Invoices		
Invoice period	Invoice amount	Download invoice
12/15/2022 - 12/17/2022	19.0 EUR	Not available

Powered by Weblate 4.15 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

您可以從此處開始創建項目，也可以使用導航條中的菜單來填寫翻譯項目的基本信息以完成添加：

Create project

Add new translation project Import translation project

**Project name** ⓘ  
WeblateOrg

Display name

**URL slug** ⓘ  
weblateorg  
Name used in URLs and filenames.

**Project website** ⓘ  
https://weblate.org/  
Main website of translated project.

**Translation instructions** ⓘ  
https://weblate.org/contribute/

You can use Markdown and mention users by @username.

**Billing** ⓘ  
Weblate Test (Basic plan)

Save

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

創建項目後，您將直接進入項目頁面：

WeblateOrg translated 100%

Components Languages Info Search Insights Files Tools Manage Share Watching

Nothing to list here.

Add new translation component

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

只需單擊一次即可自動創建新翻譯組件的操作。創建組件的過程是多階段的，自動檢測大多數翻譯參數。有幾種創建組件的方法：

### 來自版本控制

從遠程版本控制倉庫創建組件。

### 來自既有組件

通過選擇不同的文件現有組件創建其他組件。

### 額外分支

僅針對不同分支，現有組件創建其他組件。

### 上傳翻譯檔

如果您有版本控制或不想將其與 Weblate 集成，則將翻譯文件上傳到 Weblate。您以後可以使用

網絡界面或 *Weblate* 的 *REST API* 更新內容。

### 翻譯文件

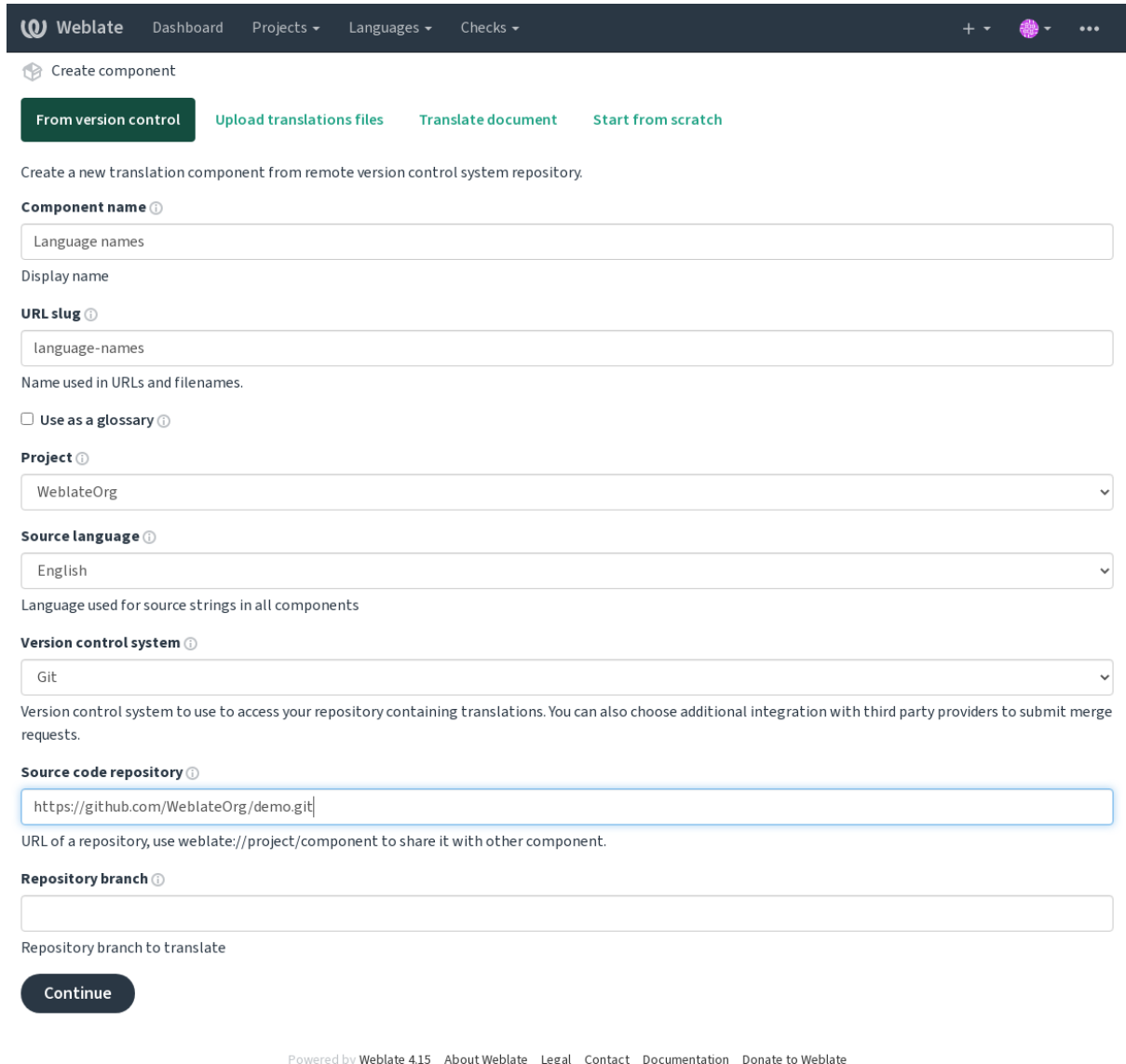
Upload single document or translation file and translate that.

### 從頭開始

創建空白翻譯項目並手動添加字符串。


一旦有了現有的翻譯組件，就可以使用同一倉庫輕鬆地添加其他文件或分支添加新的組件。

首先，您需要填寫名稱和倉庫位置：



The screenshot shows the 'Create component' form in the Weblate web interface. The form is titled 'Create component' and has a dark header bar with the Weblate logo and navigation links: 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below the header, there are four tabs: 'From version control' (selected), 'Upload translations files', 'Translate document', and 'Start from scratch'. The form itself is titled 'Create a new translation component from remote version control system repository.' and contains several fields: 'Component name' (with a hint icon) containing 'Language names', 'Display name' (empty), 'URL slug' (with a hint icon) containing 'language-names', 'Name used in URLs and filenames.' (empty), a checkbox 'Use as a glossary' (unchecked), 'Project' (dropdown menu showing 'WeblateOrg'), 'Source language' (dropdown menu showing 'English'), 'Language used for source strings in all components' (empty), 'Version control system' (dropdown menu showing 'Git'), 'Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.' (empty), 'Source code repository' (with a hint icon) containing 'https://github.com/WeblateOrg/demo.git', 'URL of a repository, use weblate://project/component to share it with other component.' (empty), 'Repository branch' (with a hint icon) (empty), and 'Repository branch to translate' (empty). At the bottom of the form is a 'Continue' button. Below the form, there is a footer bar with the text 'Powered by Weblate 4.15' and links to 'About Weblate', 'Legal', 'Contact', 'Documentation', and 'Donate to Weblate'.

在下一頁上，將顯示已發現的可翻譯資源的列表：

 Weblate Dashboard Projects Languages Checks

Create component

Add new translation component ⓘ

Choose translation files to import ⓘ

☐ Specify configuration manually

☐ File format `Android String Resource`, File mask `app/src/main/res/values-*/strings.xml`

☐ File format `gettext PO file`, File mask `weblate/langdata/locale/*/LC_MESSAGES/django.po`

☐ File format `gettext PO file`, File mask `weblate/locale/*/LC_MESSAGES/django.po`

☐ File format `gettext PO file`, File mask `weblate/locale/*/LC_MESSAGES/djangojs.po`

Continue

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

最後，您檢查翻譯組件信息 F 填寫可選詳細信息：

Weblate

Dashboard

Projects

Languages

Checks

+

▼

●

...

Create component

Add new translation component

Project

WeblateOrg

Component name

Language names

Display name

URL slug

language-names

Name used in URLs and filenames.

Version control system

Git

Version control system to use to access your repository containing translations. You can also choose additional integration with third party providers to submit merge requests.

Source code repository

https://github.com/WeblateOrg/demo.git

URL of a repository, use weblate://project/component to share it with other component.

Repository branch

Repository branch to translate

Repository push URL

URL of a push repository, pushing is turned off if empty.

Push branch

Branch for pushing changes, leave empty to use repository branch

Repository browser

https://github.com/WeblateOrg/demo/blob/{{branch}}/{{filename}}#L{{line}}

Link to repository browser, use {{branch}} for branch, {{filename}} and {{line}} as filename and line placeholders. You might want to strip leading directory by using {{filename|parentdir}}.

File format

gettext PO file

File mask

app/src/main/res/values-\*/strings.xmlweblate/langdata/locale/\*/LC\_MESSAGES/django.po

Path of files to translate relative to repository root, use \* instead of language code, for example: po/\* or locale/\*/LC\_MESSAGES/django.po.

Monolingual base language file

app/src/main/res/values/strings.xml

Filename of translation base file, containing all strings and their source; it is recommended for monolingual translation formats.

☒ Edit base file

Whether users will be able to edit the base file for monolingual translations.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Adding new translation

Create new language file

How to handle requests for creating new translations.

Template for new translations

weblate/langdata/locale/django.pot

Filename of file used for creating new translations. For gettext choose .pot file.

Translation license

GNU General Public License v3.0 or later

Language code style

Default based on the file format

Customize language code used to generate the filename for translations created by Weblate.

Language filter

^(cs|he|hu)\$

Regular expression used to filter translation files when scanning for file mask.

Source language

English

Language used for source strings in all components

☐ Use as a glossary

You will be able to edit more options in the component settings after creating it.

Save

Powered by Weblate 4.15

About Weblate

Legal

Contact

Documentation

Donate to Weblate

也參考:

*Django* 管理界面, 項目配置, 組件配置

### 2.7.3 項目配置

創建一個翻譯項目，然後在其中添加一個新的翻譯組件。這個項目就像一個架子，上面堆放著真正的翻譯。同一項目中的所有組件共享建議及其字典；翻譯也將自動傳播到單個項目中的所有組件（除非在組件配置中關閉），請參見翻譯記憶。

也參考:

/devel/integration

這些基本屬性被新建通知翻譯人員項目：

#### 專案名稱

詳細的項目名稱，用於顯示項目名稱。

也參考:

`PROJECT_NAME_RESTRICT_RE`

#### URL slug

適用於 URL 的項目名稱。

#### 專案網站

譯者可以在其中找到有關該項目的更多信息的 URL。

這是必需參數，除非關閉：設置：`websents_required`。

也參考:

`PROJECT_WEB_RESTRICT_RE`

#### 翻譯指示

Text describing localization process in the project, and any other information useful for translators. Markdown can be used for text formatting or inserting links.

#### 設定「Language-Team」檔案標頭

Weblate 是否應管理 Language-Team 頭（目前這是僅 *GNU gettext* 功能）。

## 使用共享的翻譯記憶

是否使用共享翻譯記憶庫，有關更多詳細信息，請參見[共享的翻譯記憶](#)。

The default value can be changed by `DEFAULT_SHARED_TM`.

## 貢獻至共享翻譯記憶

是否貢獻到共享翻譯記憶庫，請參見[共享的翻譯記憶](#)以獲取更多詳細信息。

The default value can be changed by `DEFAULT_SHARED_TM`.

## 存取控制

配置每個項目的訪問控制，請參見[專案存取控制](#)以獲取更多詳細信息。

可以透過 `DEFAULT_ACCESS_CONTROL` 修改預設值。

## 用檢

允許复核翻譯的工作流程，請參見[專門的審核者](#)。

## 用來源檢

用審核來源字串的工作流程，參見[源字符串](#)[查](#)。

**也參考：**

`report-source`, [評](#)

## 用勾

是否將未經身份驗證的[通知](#)[勾](#)用於此倉庫。

**也參考：**

[中間語言檔案](#), [源字符串的質量網關](#), [雙語和單語格式](#), [語言定義](#)

## 語言名

將翻譯導入到 Weblate 時定義語言代碼映射。當您的存儲庫中的語言代碼不一致，[且](#)您希望在 Weblate 中獲得一致的視圖，或者如果您想使用翻譯文件的非標準命名時，可以使用此方法。

典型的使用情[會](#)是將美國英語映射到英語：`en_US:en`

由逗號分隔的多個映射：`en_GB:en,en_US:en`

使用非標準編碼：`ia_FOO:ia`

---

**提示：** 當匹配翻譯文件時映射語言代碼，[且](#)映射是大小寫敏感的，所以您確保使用與文件名中使用的形式相同的源語言代碼。

---

**也參考：**

[添加新的翻譯](#), [語言碼](#), [Parsing language codes](#)

## 2.7.4 組件配置

組件是用於翻譯的☐容的分組。您輸入版本控制系統（VCS）倉庫位置和想要翻譯那個文件的掩碼，Weblate 會自動地從這個版本控制系統（VCS）中取回，☐找到所有匹配的翻譯文件。

**也參考：**

[/devel/integration](#)

您可以在[支持的文件格式](#) 中找到一些典型配置的例子。

---

**備☐：** It is recommended to keep translation components to a reasonable size - split the translation by anything that makes sense in your case (individual apps or add-ons, book chapters or websites).

Weblate 能☐輕鬆處理 10000 個字符串，但大的翻譯組件的分割工作和翻譯者之間的協調更困難。

---

如果翻譯的語言定義☐失，會新建一個空的定義，☐且命名☐「cs\_CZ (generated)」。您應該調整定義，☐將其反饋給 Weblate 的作者，從而☐失的語言可以包括在下一次的發布版本中。

使用版本控制系統（VCS）工作的所有重要參數都包含在組件中，☐且從中取出翻譯：

### 組件名稱

冗長組件名稱，用於顯示組件的名稱。

### 組件標識串

適用於 URLs 的組件名稱。

### 組件項目

組件所屬的[項目配置](#)。

### 版本控制系統

使用的版本控制系統（VCS），細節請參見：[版本控制整合](#)。

**也參考：**

[推送 Weblate 的更改](#)

### 源代碼儲存庫

版本控制系統（VCS）倉庫，用於拉取更改。

**也參考：**

指定 URLs 的更多細節請參見[訪問存儲庫](#)。

---

**提示：** 這可以或者是真實的版本控制系統（VCS）的 URL，或者是 `weblate://project/component`，指示了倉庫應該與其它組件分享。更多細節請參見 [Weblate internal URLs](#)。

---



## 儲存庫推送 URL

用於推送的倉庫 URL。這個設置用於 *Git* 和 *Mercurial*，且當這個空白時推送支持這些關閉。

For linked repositories, this is not used and setting from linked component applies.

### 也參考:

關於如何指定倉庫 URL 的更多細節請見 [訪問存儲庫](#)，且關於從 Weblate 推送更改的更多細節，請參見 [推送 Weblate 的更改](#)。

## 儲存庫瀏覽器

用於顯示源文件（已使用消息的位置）倉庫瀏覽器的 URL。當空白時將不生成這樣的連接。您可以使用 [模板標記](#)。

例如在 GitHub 上，使用像：“<https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename}}#L{{line}}>”那樣的一些東西

In case your paths are relative to different folder (path contains `..`), you might want to strip leading directory by `parentdir` filter (see [模板標記](#)): `https://github.com/WeblateOrg/hello/blob/{{branch}}/{{filename|parentdir}}#L{{line}}`

## 已匯出儲存庫 URL

由 Weblate 進行的更改被導出的 URL。當不使用 [持續本地化](#) 時，或者當需要手動合更改時，這是重要的。您可以 Git 倉庫使用 *Git* 導出器，來將其自動化。

## 儲存庫分支

從版本控制系統（VCS）核實哪個分支，以及從哪尋找翻譯。

For linked repositories, this is not used and setting from linked component applies.

## 推送分支

用於推送更改的分支，留空白來使用 [儲存庫分支](#)。

For linked repositories, this is not used and setting from linked component applies.

---

**備註：** 此功能目前只支持 Git、GitLab 和 GitHub，無法在其他 VCS 集成中工作。

---

### 也參考:

[推送 Weblate 的更改](#)

## 文件掩碼

要翻譯的文件的掩碼，包括路徑。它應包含一個 “\*” 替語言代碼（有關處理方式的信息，請參 [語言定義](#)）。如果您的倉庫包含多個翻譯文件（例如，多個 gettext 域），則您需要每個文件創建一個組件。

例如 “`pol.po`” 或 “`locale/LC_MESSAGES/django.po`”。

如果文件名包含特殊字符（例如 “[, ]”），則需要將這些特殊字符轉義 [ ] 或 [ ]。


### 也參考:

雙語和單語格式，“*There are more files for the single language (en)*”（單一語言‘英語’有多個文件）是什麼意思？

## 單語的基底語言檔

包含字符串定義的譯文模板文件，用於單語言組件。



### 也參考：

雙語和單語格式，“*There are more files for the single language (en)*”（單一語言‘英語’有多個文件）是什  意思？

## 編輯基底檔


對於單語言組件 是否允許編輯譯文模板文件。

## 中間語言檔案

對於單語言組件 的單一語言文件。在多數情  下，這是開發者提供的翻譯文件，  且在新建真正的源字符串時使用。

When set, the source strings are based on this file, but all other languages are based on 單語的基底語言檔. In case the string is not translated into the source language, translating to other languages is prohibited. This provides 源字符串的質量網關.

### 也參考：

源字符串的質量網關，雙語和單語格式，“*There are more files for the single language (en)*”（單一語言‘英語’有多個文件）是什  意思？

## 新翻譯的模板

用於生成新翻譯的譯文模板文件，例如 gettext 的 .pot 文件。

---

**提示：** In many monolingual formats Weblate starts with empty file by default. Use this in case you want to have all strings present with empty value when creating new translation.

---


### 也參考：

: ref: 添加翻譯, : ref: 'new-translations', : ref: component-new\_lang, : ref: bimono, : ref: faq-duplicate-files



## 檔案格式

翻譯文件格式，還請參見 [支持的文件格式](#)。

## 來源字符串臭虫回報位址

用於匯報上游缺陷的電子郵件地址。Weblate 中做出的任何字符串  釋的通知，也由這個地址接收。

## 允許翻譯再用

您可以關閉項目  從其它組件到這個組件的翻譯的傳播。這真正依賴於您在翻譯的是什 ，有時最好多次使用同一個翻譯。

對於單語言翻譯，除非您跨越整個項目中使用相同的 ID，通常關閉它是個好主意。

默認值可以通過 `DEFAULT_TRANSLATION_PROPAGATION` 來更改。


### 也參考：

[跨組件保持翻譯一致](#)

## 用建議

對於這個組件，建議的翻譯是否被接受。

## 建議投票

 建議打開投票，請參見 [建議投票](#)。

## 自動接受建議

自動接收被投票的建議，請參見 [建議投票](#)。



## 翻譯旗標

質量檢查和其他 Weblate 行  的定制，請參見 [使用標 !\[\]\(5b78f4d8e2942ab203be44f938cc0a7c\_img.jpg\) 自定義行 !\[\]\(1f09aec1483927ae51093bfc72ceaa0e\_img.jpg\)](#)。

## 制查核

檢查哪個不能被忽視的列表，請參見 [制檢查](#)。

---

備 ：執行檢查不會自動  用它，您仍然應該使用：REF：“自定義檢查”：REF：“Component-Check\_Flags”或：Ref：“其他 “。

---

## 翻譯授權條款

翻譯的許可（不需要與源代碼的許可相同）。

## 貢獻者協議書

使用者必須先同意使用者協議才能翻譯此組件。

## 加入新翻譯

如何處理創建新語言的請求。可用選項：

### 聯絡維護者

用 可以選擇所需的語言，項目維護者將收到有關該語言的通知。由他們 定是否向倉庫添加（或不添加）語言。

### 指向翻譯指示 URL

向用 顯示的頁面鏈接描述了開始新翻譯的過程。如果需要更正式的流程（例如，在開始實際翻譯之前組成人員團隊），請使用此選項。

### 建立新語言檔

用 可以選擇語言，然後 Weblate 會自動 其新建文件 開始翻譯。

### 停用加入新翻譯

用 將無法選擇開始新的翻譯。

---

**提示：**項目管理員可以添加新的翻譯，即使它是可能的（參考：Ref：‘Component-New\_base’或從空文件開始的文件格式支持）。

---

### 也參考：

adding-translation, 添加新的翻譯

## 管理字串

在 4.5 版本新加入。

配置 WebLte 中的用 是否將被允許添加新字符串 除現有字符串。調整此項以匹配您的本地化工作流程 - 如何介紹新字符串。

對於雙語格式，通常從源代碼中提取字符串（例如，通過使用：程序：*xgettext*） 禁用 Web2 中添加新字符串（在下次更新翻譯文件時將被 ）。在 WebLte 中，您可以 每個翻譯管理字符串， 且它不會在所有翻譯中 制執行字符串。

對於單晶格式，字符串僅在源語言上管理， 在翻譯中自動添加或 除。一旦翻譯，字符串會在翻譯文件中出現。

### 也參考：

雙語和單語格式, adding-new-strings, `POST /api/translations/(string:project)/(string:component)/(string:language)/units/`

## 語言代碼類型

用來生成使用 Weblate 建立之翻譯檔名的自訂語言代碼。

### 也參考：

添加新的翻譯, 語言碼, *Parsing language codes*

## 合併類型

You can configure how updates from the upstream repository are handled. The actual implementation depends on VCS, see [版本控制整合](#).

### 衍合

Rebases Weblate commits on top of upstream repository on update. This provides clean history without extra merge commits.

在複雜融合的情況下，變基可能使你生麻煩，因此請仔細考慮是否允許它們。

You might need to enable force pushing by choosing [Git 使用制推送](#) as 版本控制系統, especially when pushing to a different branch.

### 合併

Upstream repository changes are merged into Weblate one. This setting utilizes fast-forward when possible. This is the safest way, but might produce a lot of merge commits.

### 不快轉合併

Upstream repository changes are merged into Weblate one with doing a merge commit every time (even when fast-forward would be possible). Every Weblate change will appear as a merge commit in Weblate repository.

默認值可以由 `:setting:DEFAULT_MERGE_STYLE` 更改。

## Commit, add, delete, merge, add-on, and merge request messages

當提交翻譯時使用的消息，請參見模板標記。

Default value can be changed by `DEFAULT_ADD_MESSAGE`, `DEFAULT_ADDON_MESSAGE`, `DEFAULT_COMMIT_MESSAGE`, `DEFAULT_DELETE_MESSAGE`, `DEFAULT_MERGE_MESSAGE`, `DEFAULT_PULL_MESSAGE`.

## 提交時一推送

是否自動推送已提交的項目到上游倉庫。用時，當 Weblate 將更改提交到其基礎存儲庫中，就會自動推送。（請參見[簡易提交](#)）。了能真正用推送功能，需配置 `Repository push URL`。

## 更動後提交的經過時間

設置在後台任務或 `djadmin:commit_pending` 管理命令提交更改前，這些更改存在的時長（以小時單位）。一旦存在至少一個比該時長更舊的更改，便會提交組件中的所有更改。

默認值可以由 `COMMIT_PENDING_HOURS` 更改。

---

**提示：**有其他情況可能會提交起的更改，請參：REF：“懶惰提交”。

---

## 有錯誤時鎖定

鎖定組件（及關聯的組件，見 `ref:internal-urls`），觸發條件是第一個失敗的推送，合到其上游存儲庫，或從中拉出。這樣可以避免添加其他衝突，這些衝突必須手動解。

一旦倉庫有故障留下來了，組件將會自動解鎖。

## 來源語言

用於源字符串的語言。如果您要翻譯的不是英語，請更改此選項。

**提示：**如果你正在從英語翻譯雙語文件，但又希望能 在英語翻譯中進行修復，選擇 *English (Developer)* 作一種源語言以避免源語言和現有翻譯之間名稱上的衝突。

對於單語言翻譯，您可以使用這種情 下的中間翻譯，請參見 [中間語言檔案](#)。

## 語言篩選

Regular expression used to filter the translation when scanning for file mask. It can be used to limit the list of languages managed by Weblate.

**備：**單出現在文件名中時，您需要列出語言代碼。

過濾的一些例子：

過濾器的描述	正則表達式
Selected languages only	<code>^(cs de es)\$</code>
排除的語言	<code>^(?! (it fr) \$) .+\$</code>
只篩選兩個字母的代碼	<code>^..\$</code>
排除非語言文件	<code>^(?! (blank) \$) .+\$</code>
包括所有文件（默認）	<code>^[^.] +\$</code>

## 變體的正則表達式

用於確定字符串變體的正則表達式，請見 [variants](#)。

**備：**Most of the fields can be edited by project owners or administrators, in the Weblate interface.

## 也參考：

[Weblate 支持 Git 和 Mercurial 意外的 VCS 嗎？](#), [alerts](#)

## 優先度

較高優先度的組件會優先提供給翻譯者。

在 4.15 版本變更: This now also affects ordering of matched glossary terms.

## 受限制的訪問

組件默認對訪問項目的任何人都可見，即使不能在組件中進行任何更改。這會容易地使翻譯在項目 保持一致。

無論項目級 權限如何，限制組件的訪問或組件列表級 將接管到組件的訪問權限。您必須明確授予對其的訪問權限。這可以通過授予對新用 組的訪問 將用 放入其中，或使用默認的“自定義”或“私有”訪問控制組。

默認設置可在 `DEFAULT_RESTRICTED_COMPONENT` 中更改。

---

**提示：** 這也應用於項目管理員—請確認切狀態後，您不會失對組件的訪問。

---

## 分享專案

可以選擇組件可見的附加項目。這在分享不同項目間使用的庫時是有用的。

---

**備：** 分享組件不更改其訪問控制。這樣做只是讓它在覽其它項目時可見。用仍然需要訪問實際組件的權限來覽或翻譯它。

---

## 當作詞表

在 4.5 版本新加入。

允許使用該組件作術語表。您可以配置其列出方式，借助詞表色彩。

詞表將在以下所有項目中訪問：REF: “組件鏈接”。

建議用：參考: “組件 - Manage\_Units”，以便它們添加新單詞。

**也參考：**

詞表

## 詞表色彩

顯示色，用於顯示單詞匹配時使用的詞表。

## 2.7.5 模板標記

Weblate 在需要提供文本的幾個地方使用簡單的標記語言。它基於 [The Django template language](#)，因此能非常。

當前它用在：

- 提交消息格式，請參見 [組件配置](#)
- **Several add-ons**
  - 組件探索
  - 統計數據生成器
  - *Executing scripts from add-on*

可以在組件模板中得到後面的變量：

```
{{ language_code }}
```

語言碼

```
{{ language_name }}
```

語言名稱

```
{{ component_name }}
```

組件名稱

```
{{ component_slug }}
```

組件標識串

```
{{ project_name }}
```

專案名稱

```
{{ project_slug }}
    項目標識串

{{ url }}
    翻譯 URL

{{ filename }}
    翻譯檔名

{{ stats }}
    翻譯統計數據，這具有進一步的屬性，示例如下。

{{ stats.all }}
    Total strings count

{{ stats.fuzzy }}
    需要審核的字串數量

{{ stats.fuzzy_percent }}
    需要審核的字串數量百分比

{{ stats.translated }}
    Translated strings count

{{ stats.translated_percent }}
    Translated strings percent

{{ stats.allchecks }}
    檢查失敗的字符串數量

{{ stats.allchecks_percent }}
    檢查失敗的字符串百分比

{{ author }}
    當前提交的作者，只在提交範圍可用。

{{ addon_name }}
    Name of currently executed add-on, available only in the add-on commit message.
```

後面的變量在倉庫 F 覽器或編輯器模板中可用：

```
{{branch}}
    當前的分支

{{line}}
    line in file
```

```
{{filename}}
```

文件名，您也可以使用 `parentdir` 過濾器，例如 `{{filename|parentdir}}`，來 F 除前導部分  
您可以將它們與過濾器結合：

```
{{ component|title }}
```

您可以使用條件：

```
{% if stats.translated_percent > 80 %}Well translated!{% endif %}
```

有另外的標 F 用於替 F 字符：

```
{% replace component "-" " " %}
```

您可以將它與過濾器結合：

```
{% replace component|capfirst "-" " " %}
```

還有另外的過濾器來操作文件名：



```

Directory of a file: {{ filename|dirname }}
File without extension: {{ filename|striptext }}
File in parent dir: {{ filename|parentdir }}
It can be used multiple times: {{ filename|parentdir|parentdir }}

```

……以及其他 Django 模板特性。

## 2.7.6 導入速度

取回版本控制系統（VCS）倉庫，[將翻譯導入 Weblate](#)，依賴於您的翻譯的大小，這會是漫長的過程。這[是](#)一些提示：

### 優化配置

對於測試[調試 Weblate](#)，默認的配置是有用的，當用於生[成](#)設置時，您應該進行一些調整。它們中的很多都對形成具有巨大的衝擊。特[是](#)，更多細節請查看[生](#)[成](#)設置：

- 配置 Celery 來執行後台任務（請參見[使用 Celery 的後台任務](#)）
- [允許緩存](#)
- [使用](#)[力](#)的數據庫引擎
- [Disable debug mode](#)

### Check resource limits

如果導入巨大的翻譯或倉庫，您會遭到服務器資源限制的打擊。

- 檢查空[間](#)[存](#)的量，通過操作系統來緩存翻譯，將極大地提高性能。
- 如果有很多字符串需要處理的話，磁盤操作會是瓶頸——磁盤被 Weblate 和數據庫施加壓力。
- 另外的 CPU 核心會幫助提高後台任務的性能（請參見[使用 Celery 的後台任務](#)）。

### Disable unneeded checks

一些質量檢查可以使非常昂貴的，而如果不需要，在導入時省略可以節省一些時間。配置的信息請參見[CHECK\\_LIST](#)。

## 2.7.7 自動新建組件

In case your project has dozen of translation files (e.g. for different gettext domains, or parts of Android apps), you might want to import them automatically. This can either be achieved from the command-line by using [import\\_project](#) or [import\\_json](#), or by installing the [組件探索](#) add-on.

To use the add-on, you first need to create a component for one translation file (choose the one that is the least likely to be renamed or removed in future), and install the add-on on this component.

對於管理命令，您需要新建包含所有組件的項目，然後運行[import\\_project](#) 或 [import\\_json](#)。

也參考：

[管理命令](#), [組件探索](#)

## 2.8 語言定義

為了恰當地呈現不同的翻譯，需要提供有關語言名稱、文本方向、`code` 數定義和語言代碼的信息。

### 2.8.1 Built-in language definitions

Weblate 中包括了大約 600 種語言的定義，且每次發時列表都在擴大。無論何時更新 Weblate 時（更特地是無論何時執行 `weblate migrate` 時，請參見：[ref:generic-upgrade-instructions](#)）。語言數據庫都被更新，來包括 Weblate 上市時的所有語言定義。

這個特性可以使用 `UPDATE_LANGUAGES` 來禁止。還可以使用 `django-admin:setuplang` 來制更新數據庫，從而匹配 Weblate 建數據。

也參考：

[Extending built-in language definitions](#), [Current language definitions](#)

### 2.8.2 Parsing language codes

While parsing translations, Weblate attempts to map language code (usually the ISO 639-1 one) from the 文件掩碼 to any existing language object.

您可以通過語言名在項目層次來進一步調整這種映射。

如果無法找到精確的匹配，將嘗試把其融入一種現有的語言。已嘗試以下措施：

- 大小寫不敏感的查詢。
- Normalizing underscores and dashes.
- Looking up built-in language aliases.
- Looking up by language name.
- 忽略給定語言的默認國家代碼—選擇 `cs` 而非 `cs_CZ`。

如果這也失敗了，將使用默認值創建一個新的語言定義（從左到右的文本方向，一個數）。自動創建的帶有代碼 `xx_XX` 的語言將被命名 `xx_XX (generated)`。您可能想稍後在管理界面中更改這個（見：[ref:changing-languages](#)），將它報告給問題跟踪器（見：[ref:contributing](#)），這樣的話，正確的定義就可以添加到即將發布的 Weblate 版本中。

---

**提示：** 在您看到有些不想要的容作語言的情下，您會想要調整語言篩選，當分析翻譯時忽略這樣的文件。

---

也參考：

[語言碼](#), [添加新的翻譯](#)

### 2.8.3 Changing language definitions

您可以在語言界面來更改語言定義（`/languages/` URL）。

當編輯時，確認所有字段都是正確的（特別是數和正文方向），否則譯者將不能正常編輯這些翻譯。

## 2.8.4 Ambiguous language codes and macrolanguages

In many cases it is not a good idea to use macrolanguage code for a translation. The typical problematic case might be Kurdish language, which might be written in Arabic or Latin script, depending on actual variant. To get correct behavior in Weblate, it is recommended to use individual language codes only and avoid macrolanguages.

也參考:

宏語言定義 <<https://iso639-3.sil.org/about/scope#Macrolanguages>>, 宏語言列表 <[https://iso639-3.sil.org/code\\_tables/macrolanguage\\_mappings/data](https://iso639-3.sil.org/code_tables/macrolanguage_mappings/data)>

## 2.8.5 語言定義

每種語言都包括後面的字段:


### 語言碼

Code identifying the language. Weblate prefers two letter codes as defined by ISO 639-1, but uses ISO 639-2 or ISO 639-3 codes for languages that do not have two letter code. It can also support extended codes as defined by BCP 47.

也參考:

*Parsing language codes*, 添加新的翻譯


### 語言名稱

語言的可見名稱。還要根據用  界面語言將 Weblate 中包括的語言名稱進行本地化。

### 文字方向

確定語言是從右向左還是從左向右書寫。對於多數語言這個屬性都能  正確地自動監測。

### 數數量

語言中使用的  數的數量。

### 數公式

Gettext 兼容的  數公式，用於確定根據給定的數量使用哪種  數形式。

也參考:

 數, GNU gettext 工具:  數形式 <[https://www.gnu.org/software/gettext/manual/html\\_node/Plural-forms.html](https://www.gnu.org/software/gettext/manual/html_node/Plural-forms.html)>,   
\*Language Plural Rules by the Unicode Consortium ‘\_

## 語言使用者數量

Number of worldwide speakers of this language.

### 2.8.6 添加新的翻譯

在 2.18 版本變更: 在 2.18 以前的版本中, 添加新的翻譯的行因文件格式而不同。

Weblate 可以所有文件格式自動開始新的翻譯。

Some formats expect to start with an empty file and only translated strings to be included (for example 安卓字符串資源), while others expect to have all keys present (for example *GNU gettext*). The document-based formats (for example *OpenDocument Format*) start with a copy of the source document and all strings marked as needing editing. In some situations this really doesn't depend on the format, but rather on the framework you use to handle the translation (for example with *JSON files*).

當在組件配置中指定新翻譯的模板時, Weblate 將使用這個文件開始新的翻譯。當執行時任何現有翻譯將從文件中除。

當新翻譯的模板是空的, 且文件格式支持時, 新建空文件, 一旦新的字符串被翻譯就添加進去。

語言代碼類型 允許在生成的文件名中將語言代碼個性化:

#### 預設基於檔案格式

依賴於文件格式, 對於其中的多數使用 POSIX。

#### POSIX 樣式, 使用底當作分隔符號

典型地由 gettext 和相關工具使用, 生成像 pt\_BR 那樣的語言代碼。

#### POSIX 樣式, 使用底當作分隔符號, 包含地區碼

POSIX 風格的語言代碼即使不必要時也包括國家代碼; 例如 cs\_CZ)。

#### BCP 樣式, 使用連字號當作分隔符號

典型地在 web 平台上使用, 生成像 pt-BR 那樣的語言編碼。

#### BCP 樣式, 使用連字號當作分隔符號, 包含地區碼

BCP 風格的語言代碼即使不必要時也包括國家代碼; 例如 cs-CZ)。

#### BCP 樣式, 使用連字號作分隔符號、舊式語言代碼

Uses legacy codes for Chinese and BCP style notation.

#### BCP 樣式, 使用連字號作分隔符號, 小寫

BCP style notation, all in lower case (for example cs-cz)。

#### Apple App Store 中介資料樣式

Style suitable for uploading metadata to Apple App Store.

#### Google Play 中介資料樣式

Style suitable for uploading metadata to Google Play Store.

#### Android 樣式

只在安卓 apps 中使用, 生成像 pt-rBR 那樣的語言代碼。

#### Linux 樣式

Locales as used by Linux, uses legacy codes for Chinese and POSIX style notation.

此外, 語言名中定義的任何映射都反向應用。

---

備: Weblate 當解析翻譯文件是識所有這些, 上面的設置只影響如何新建新的文件。

---

#### 也參考:

語言碼, 語言名, *Parsing language codes*

## 2.9 持續本地化

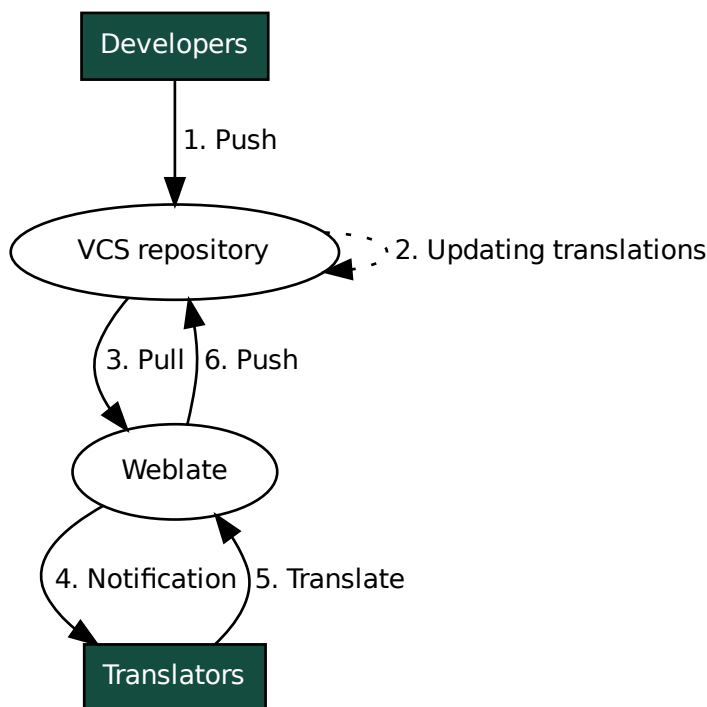
有適當的基礎結構，因此你的翻譯緊隨開發。這樣，翻譯人員可以一直進行翻譯，而不必在發布之前處理大量的新文本。

### 也參考：

/dev/integration 描述了將您的開發集成到 Weblate 中的基本方式。

這是過程：

1. 開發人員進行更改 將其推送到版本控制系統（VCS）倉庫。
2. 可以選擇更新翻譯文件（這取 於文件格式，請參 當已經更新了模板時， 什 Weblate 仍然顯示舊的字符串？）。
3. Weblate 從版本控制系統（VCS）倉庫中拉取更改，請參 更新倉庫。
4. 一旦 Weblate 檢測到翻譯更改，便會根據翻譯者的訂 設置通知他們。
5. 翻譯者使用 Weblate Web 界面提交翻譯，或上傳離 更改。
6. 翻譯者完成後，Weblate 會將更改提交到本地倉庫（請參 簡易提交），如果有權限將其推回（請參 推送 Weblate 的更改）。



## 2.9.1 更新倉庫

應該新建一些從他們的源來更新後端倉庫的方式。

- 使用通知勾 來與多數常見的代碼管服務集成：
  - 從 *GitHub* 自動接收更改
  - 從 *GitLab* 自動接收更改
  - 從 *Bitbucket* 自動接收更改
  - 從 *Pagure* 自動接受更改
  - 從 *Azure Repos* 自動接收更改
- 在倉庫管理中或使用 *Weblate* 的 *REST API* 或 *Weblate* 客戶端 來手動觸發更新
- 允許 *AUTO\_UPDATE* 在你的 *Weblate* 事例上自動更新所有組件
- 執行 *updategit* (選擇項目, 或 *--all* 來更新全部)

每當 *Weblate* 更新存儲庫時, 更新後插件都將被觸發, 請參見: 附加元件。

### Avoiding merge conflicts

當相同的文件在 *Weblate* 之與之外都更改時導致 *Weblate* 的合衝突。有兩種方法來處理——避免在 *Weblate* 之外編輯, 或者將 *Weblate* 集成到你的更新過程中, 從而在更新 *weblate* 之外的文件之前刷新更改。

第一種方法容易用於單語言文件——可以添加 *Weblate* 之的新字符串, 將文件的整個編輯留在那。對於雙語言文件, 通常存在某種消息提取過程而從源代碼生翻譯文件。在一些情況下, 這可以分成兩部分——一部分用於提取過程生模板 (例如使用 `program:xgettext` 生 `gettext POT`), 然後下一步過程將它合到真正的翻譯中 (例如使用 `program:msgmerge` 更新 `gettext PO` 文件)。可以在 *Weblate* 中執行第二步, 它將確認在這個操作前所有待定的更改都包括進去了。

第二種方法可以這樣實現, 使用 *Weblate* 的 *REST API*, 當你在自己一側進行更改時, 制 *Weblate* 推送所有待定的更改, 鎖定翻譯。

進行更新的本看起來像這樣:

```
# Lock Weblate translation
wlc lock
# Push changes from Weblate to upstream repository
wlc push
# Pull changes from upstream repository to your local copy
git pull
# Update translation files, this example is for Django
./manage.py makemessages --keep-pot -a
git commit -m 'Locale updates' -- locale
# Push changes to upstream repository
git push
# Tell Weblate to pull changes (not needed if Weblate follows your repo
# automatically)
wlc pull
# Unlock translations
wlc unlock
```

如果多個組件分享相同的倉庫, 需要分將他們全部鎖定:

```
wlc lock foo/bar
wlc lock foo/baz
wlc lock foo/baj
```

備註：例子使用了 [Weblate 客戶端](#)，這需要配置（API 密鑰）來遠程控制 Weblate。可以通過使用 HTTP 客戶端代替 `wlc`，例如 `curl` 來實現，請參見 [Weblate 的 REST API](#)。

也參考：

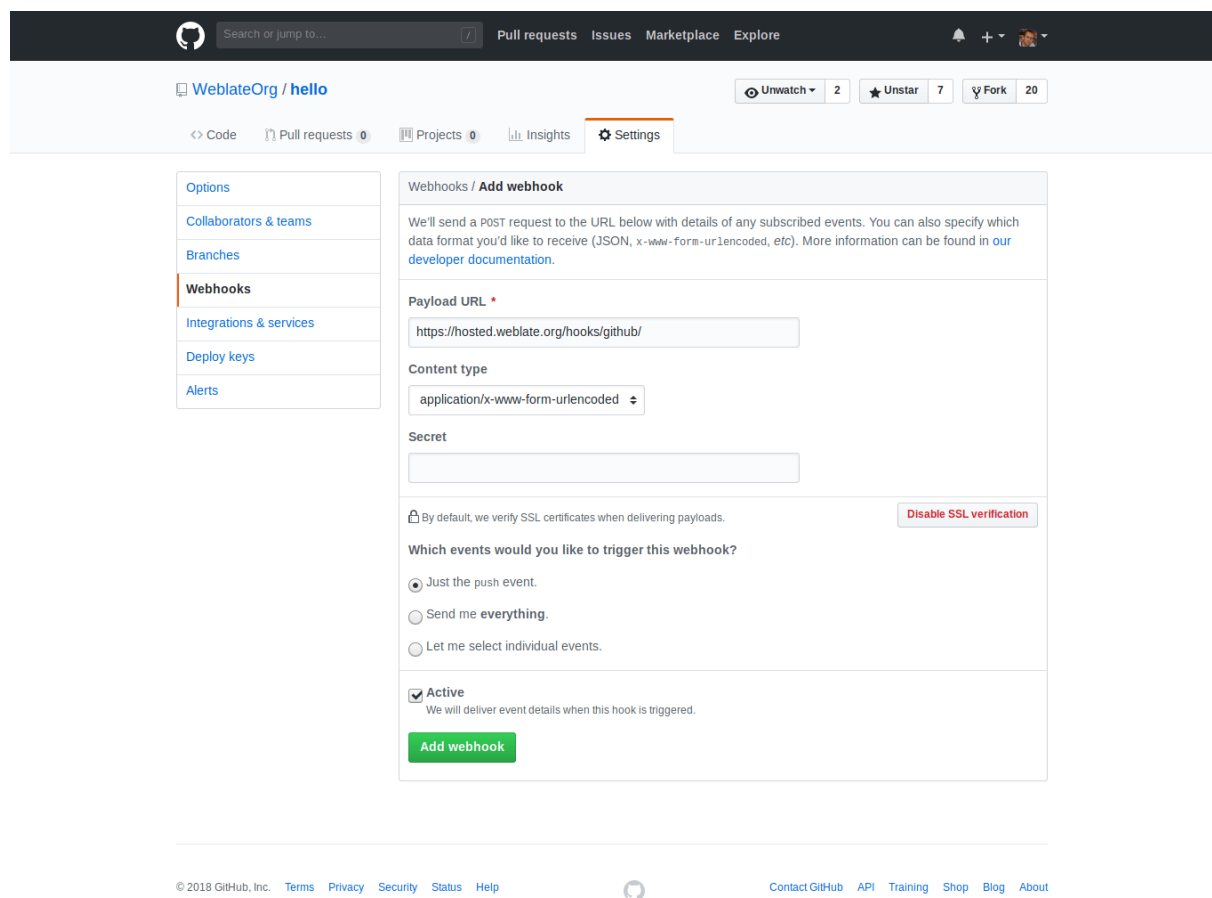
[Weblate 客戶端](#)

## 從 GitHub 自動接收更改

Weblate 伴隨 GitHub 本地支持。

如果使用 Hosted Weblate，推薦的方法是安裝 [Weblate app](#)，該方法能得到正確的設置，而不必設置很多東西。它還可以用於將更改推送回來。

為了在每次推送到 GitHub 倉庫時接收通知，將 Weblate Webhook 添加到倉庫設置（*Webhooks*）中，如下圖所示：



對於負載 URL，將 `/hooks/github/` 增補到你的 Weblate URL 中，例如對於 Hosted Weblate 服務，這是 `https://hosted.weblate.org/hooks/github/`。

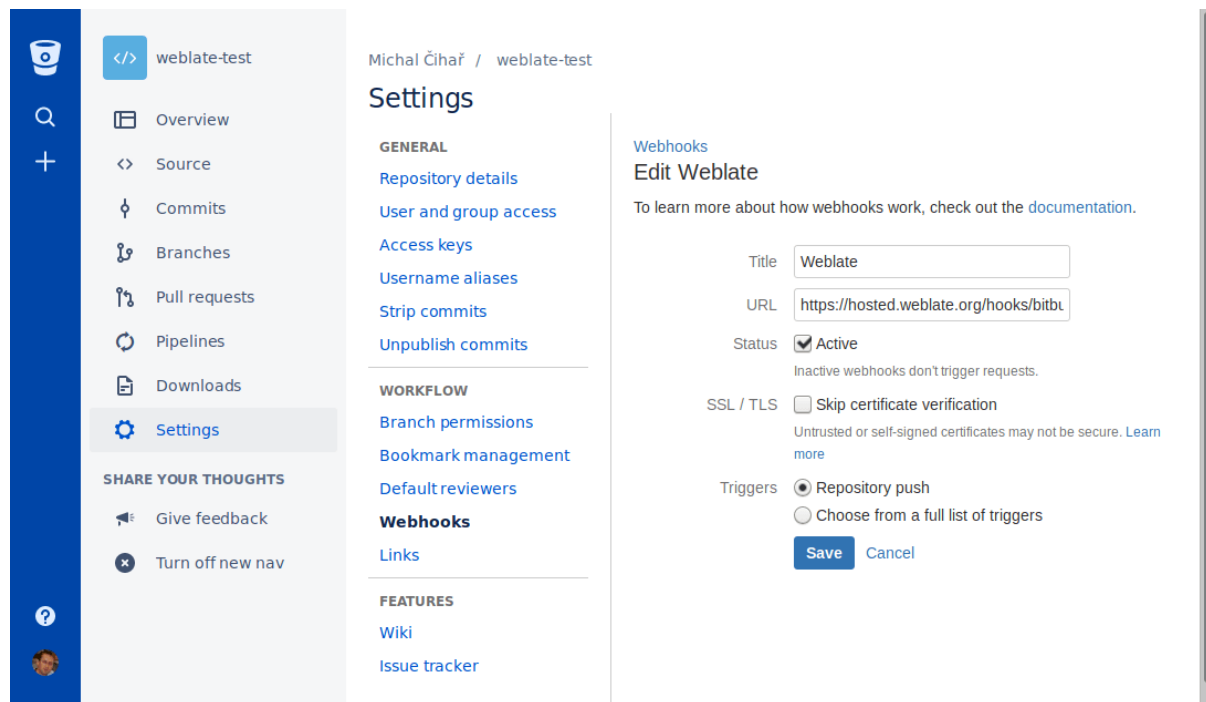
可以將其他設置保留默認值（Weblate 可以處理內容類型，只消費 *push* 事件）。

也參考：

`POST /hooks/github/`，從 [Weblate](#) 訪問存儲庫

## 從 Bitbucket 自動接收更改

Weblate 已經支持 Bitbucket webhooks，添加倉庫推送時觸發的 webhook，目的地是你的 Weblate 安裝上的“/hooks/bitbucket/”（例如“https://hosted.weblate.org/hooks/bitbucket/”）。



### 也參考:

`POST /hooks/bitbucket/`, 從 Bitbucket 訪問存儲庫

## 從 GitLab 自動接收更改

Weblate 已經支持 GitLab hooks，添加項目的 webhook，目的地是你的 Weblate 安裝上的“/hooks/gitlab/”（例如“https://hosted.weblate.org/hooks/gitlab/”）。

### 也參考:

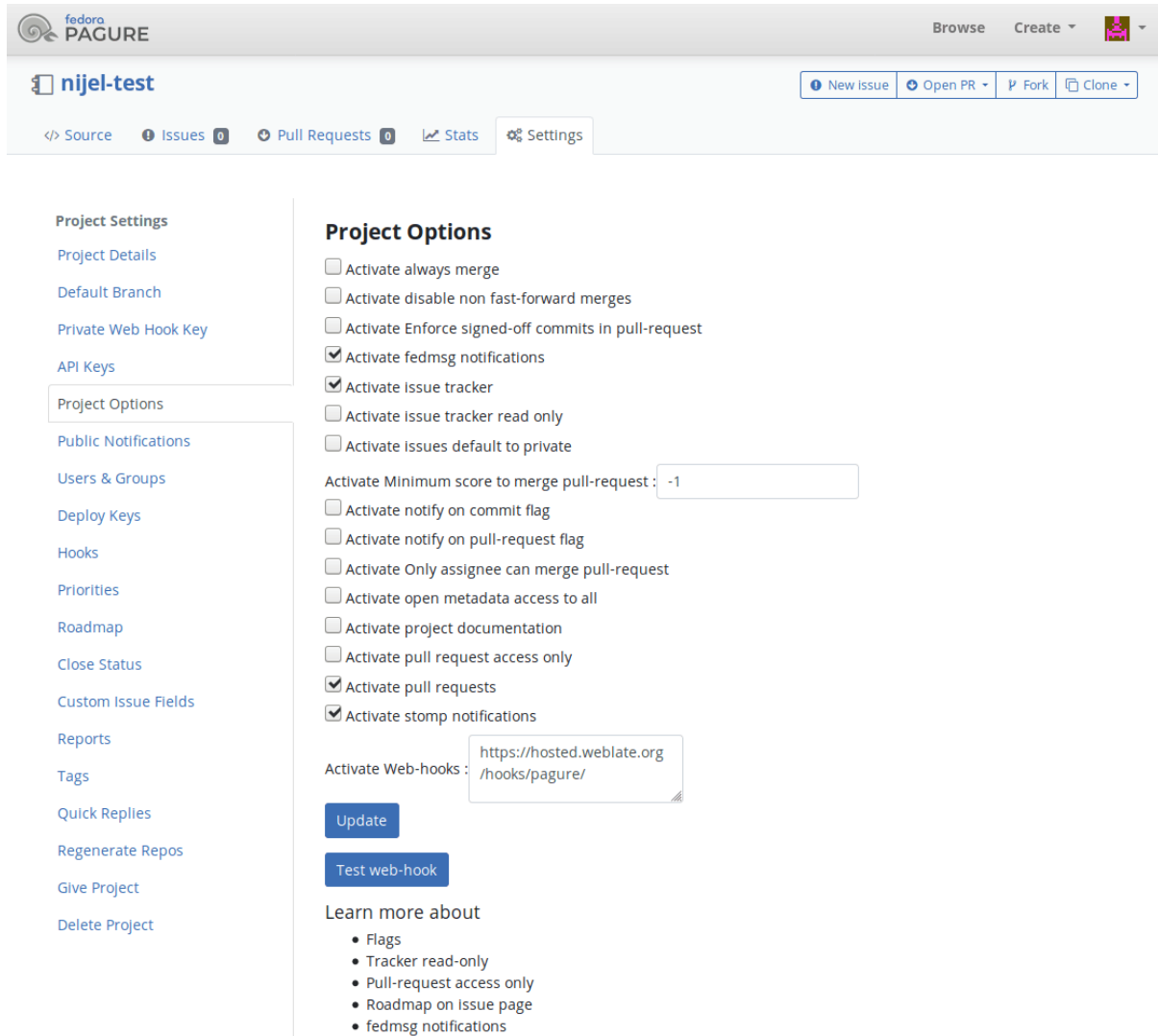
`POST /hooks/gitlab/`, 從 GitLab 訪問存儲庫



## 從 Pagure 自動接受更改

在 3.3 版本新加入。

Weblate 已經支持 Pagure hooks，添加項目的 webhook，目的地你的 Weblate 安裝上的 “/hooks/pagure/” (例如 “https://hosted.weblate.org/hooks/pagure/”)。這可以在 *Project options* 之下的 *Activate Web-hooks* 中完成：



也參考：

`POST /hooks/pagure/`, 從 [Weblate](#) 訪問存儲庫

## 從 Azure Repos 自動接收更改

在 3.8 版本新加入。

Weblate 已經支持 Azure Repos web hooks, [\[E\]:guilabel:Code pushed](#) 事件添加 webhook, 目的地你的 Weblate 安裝上的 “/hooks/azure/” URL (例如 “https://hosted.weblate.org/hooks/azure/”)。這可以在 *Project settings* 之下的 *Service hooks* 中完成。

也參考：

Web hooks in Azure DevOps manual, `POST /hooks/azure/`, 從 [Weblate](#) 訪問存儲庫

## 從 Gitea Repos 自動接收更改

在 3.9 版本新加入。

Weblate 已經支持 Gitea webhooks, 在 `Push events` 事件添加 `Gitea Webhook`, 目的地你的 Weblate 安裝上的 `/hooks/gitea/` URL (例如 `https://hosted.weblate.org/hooks/gitea/`)。這可以在 *Settings* 之下的 *Webhooks* 中完成。

### 也參考:

Gitea 手冊中的 Webhooks <<https://docs.gitea.io/en-us/webhooks/>>, `POST /hooks/gitea/`, 從 Weblate 訪問存儲庫

## 從 Gitee Repos 自動接收更改

在 3.9 版本新加入。

Weblate 已經支持 Gitee webhooks, 在 `Push` 事件添加 `Webhook`, 目的地你的 Weblate 安裝上的 `/hooks/gitee/` URL (例如 `https://hosted.weblate.org/hooks/gitee/`)。這可以在 *Management* 之下的 *Webhooks* 中完成。

### 也參考:

Gitee 手冊中的 Webhooks <<https://gitee.com/help/categories/40>>, `POST /hooks/gitee/`, 從 Weblate 訪問存儲庫

## 每晚自動更新倉庫

Weblate 在後面合入更改時, 每晚自動獲取遠程倉庫來提高性能。可以選擇將其同樣轉入進行每晚合入, 通過允許 `AUTO_UPDATE`。

## 2.9.2 推送 Weblate 的更改

每個翻譯組件可以新建推送 URL (請參見 *儲存庫推送 URL*), 在那種情況下 Weblate 能將更改推送到遠程倉庫。Weblate 還可以配置在每次提交時自動推送更改 (這是默認的, 請參見 *提交時一推送*)。如果不想更改自動給推送, 可以在 *Repository maintenance* 之下手動進行, 或通過 `wlc push` 使用 API。

推送選項根據使用的版本控制整合而不同, 更多細節可以在那個章節中找到。

In case you do not want direct pushes by Weblate, there is support for *GitHub pull requests*, *GitLab merge requests*, *Gitea pull requests*, *Pagure merge requests* pull requests or *Gerrit* reviews, you can activate these by choosing *GitHub*, *GitLab*, *Gitea*, *Gerrit* or *Pagure* as 版本控制系統 in 組件配置。

整體上, Git、GitHub 和 GitLab 可以具有後面的選項:

需要的設置	版本控制系統	儲存庫推送 URL	推送分支
不推送	<i>Git</i>	空	空
直接推送	<i>Git</i>	SSH URL	空
Push to separate branch	<i>Git</i>	SSH URL	分支名稱
來自叉子的 GitHub 拉取請求	<i>GitHub pull requests</i>	空	空
來自分支的 GitHub 拉取請求	<i>GitHub pull requests</i>	SSH URL <sup>1</sup>	分支名稱
來自叉子的 GitLab 結合請求	<i>GitLab merge requests</i>	空	空
來自分支的 GitLab 結合請求	<i>GitLab merge requests</i>	SSH URL <sup>1</sup>	分支名稱
Gitea merge request from fork	<i>Gitea pull requests</i>	空	空
Gitea merge request from branch	<i>Gitea pull requests</i>	SSH URL <sup>1</sup>	分支名稱
來自分叉的 Pagure 合入請求	<i>Pagure merge requests</i>	空	空
來自分支的 Pagure 合入請求	<i>Pagure merge requests</i>	SSH URL <sup>1</sup>	分支名稱

<sup>1</sup> 在源代碼儲存庫支持推送的情況下可以空。

備註：還可以允許 Weblate 提交後更改的自動推送，這可以在提交時一推送 中完成。

#### 也參考：

請參見訪問存儲庫 來設置 SSH 密鑰，和簡易提交 獲得關於 Weblate 提交更改的信息。

### 受保護的分支

如果在受保護的分支上使用 Weblate，可以配置使用拉取請求，執行翻譯的實際檢查（對你不知道的語言可能有問題）。另一個方法是去掉對 Weblate 推送用的這個限制。

例如在 GitHub，這可以在倉庫配置中進行：

☒ **Require pull request reviews before merging**  
When enabled, all commits must be made to a non-protected branch and submitted via a pull request with the required number of approving reviews and no changes requested before it can be merged into a branch that matches this rule.

Required approving reviews: 1



☐ **Dismiss stale pull request approvals when new commits are pushed**  
New reviewable commits pushed to a matching branch will dismiss pull request review approvals.

☐ **Require review from Code Owners**  
Require an approved review in pull requests including files with a designated code owner.

☒ **Restrict who can dismiss pull request reviews**  
Specify people or teams allowed to dismiss pull request reviews.

Search for people or teams

**People and teams that can dismiss reviews.**

-  **Organization and repository administrators**  
These members can always dismiss.
-  **weblate**  
Weblate push user

## 2.9.3 Interacting with others

Weblate 通過使用它的 API，使與他人的交流更容易。

#### 也參考：

*Weblate 的 REST API*

## 2.9.4 簡易提交

Weblate 會盡可能將同一作者的提交分組到一個提交中。這大大減少了提交的數量，但是如果你想同步版本控制系統（VCS）倉庫，例如 Git，你可能需要明確地告訴它去做提交（這對 `guilabel:Managers` 組是默認允許的，參見 [ref:privileges](#)）。

如果多個組件分享相同的倉庫，需要分組將他們全部鎖定：

- 某人另外更改了已經被更改的字符串。
- 來自上游的結合發生了。
- 明確地請求了提交。
- A file download is requested.
- 更改比 [組件配置](#) 上定義 `Age of changes to commit` 的時間段更陳舊。

---

**提示：** Commits are created for every component. So in case you have many components you will still see lot of commits. You might utilize [Git 提交 add-on](#) in that case.

---

如果想要更頻繁地提交更改，且不檢查新舊，可以安排定期時性任務來執行提交：

```
CELERY_BEAT_SCHEDULE = {
    # Unconditionally commit all changes every 2 minutes
    "commit": {
        "task": "weblate.trans.tasks.commit_pending",
        # Omitting hours will honor per component settings,
        # otherwise components with no changes older than this
        # won't be committed
        "kwargs": {"hours": 0},
        # How frequently to execute the job in seconds
        "schedule": 120,
    }
}
```

## 2.9.5 用本地處理倉庫

The way to customize how Weblate interacts with the repository is [附加元件](#). Consult *Executing scripts from add-on* for info on how to execute external scripts through add-ons.

## 2.9.6 跨組件保持翻譯一致

一旦具有多個翻譯組件，你會想要確保相同的字符串具有相同的翻譯。這可以在幾個層次實現。

### 翻譯宣傳

With [允許翻譯再用](#) enabled (what is the default, see [組件配置](#)), all new translations are automatically done in all components with matching strings. Such translations are properly credited to currently translating user in all components.

---

**備註：** 翻譯宣傳需要密鑰來匹配單語言翻譯格式，因此在建立翻譯密鑰匙請記住。

---

## 一致性檢查

在字串不同時不一致會自動檢查。您可以藉此來手動檢查其差異，選擇正確的翻譯。

## 自動翻譯

Automatic translation based on different components can be way to synchronize the translations across components. You can either trigger it manually (see 自動翻譯) or make it run automatically on repository update using add-on (see 自動翻譯).

## 2.10 翻譯許可

您可以指定翻譯輸入哪種授權方式。當翻譯對公公開時這很重要，這樣明確規定如何使用。

您應該指定組件配置版權信息。您應該避免那些需要獲得貢獻者版權協議的情況，儘管這是可能的。

### 2.10.1 版權信息

在指定版權信息的時候（版權名稱和 URL），這個信息顯示在各個組件配置的翻譯信息部分。

如果不需要特意的同意的話，這通常是放置許可信息的最佳位置。如果您的項目或翻譯不是開源項目，那麼您最可能需要事先同意。

### 2.10.2 貢獻者協議書

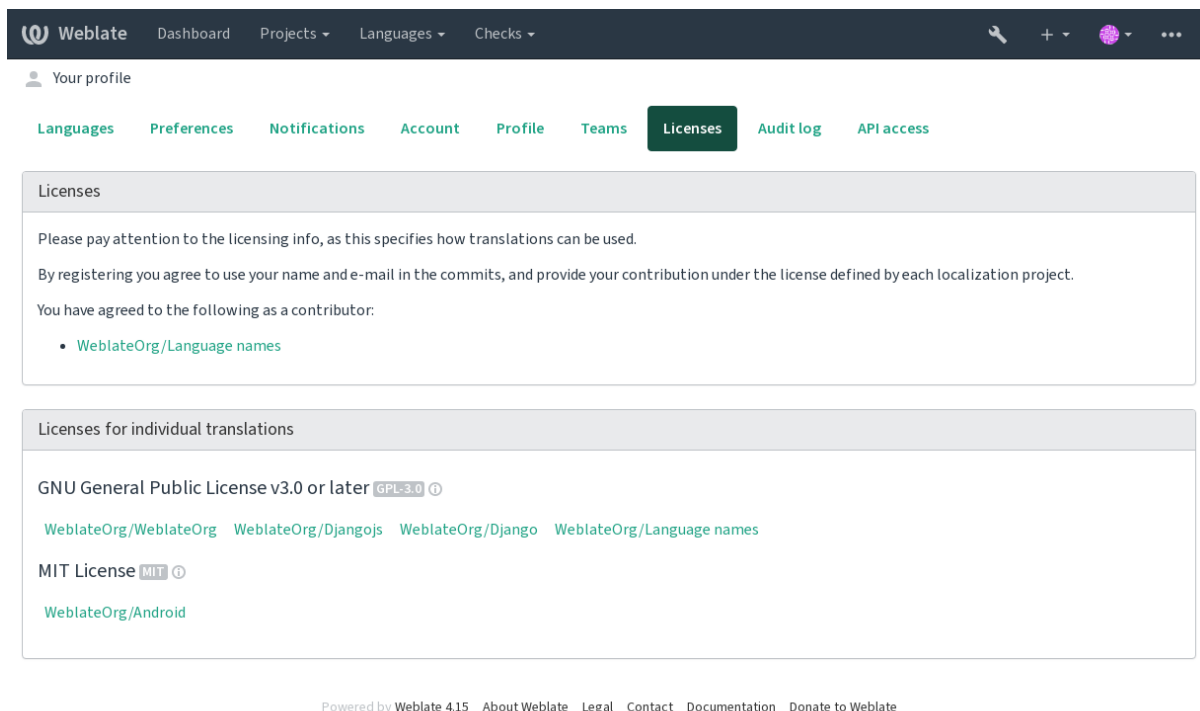
如果您指定了貢獻者版權協議，那麼只有同意協議的用戶才能做出貢獻。當進入到翻譯時，這是清晰的步驟：

Language	Translated	Unfinished	Unfinished words	Unfinished characters	Checks	Suggestions	Comments
Czech	✓						
Hebrew	✓						
Hungarian	81%	4	5	32	3		
English	✓						

輸入的文本被格式化段落，而非且可以包括外部連接。不能使用 HTML 標記。

## 2.10.3 用 版權

任何用 可以在其簡介的實例中看到所有公開項目的所有翻譯版權：



## 2.11 翻譯程序

### 2.11.1 建議投票

Everyone can add suggestions by default, to be accepted by signed in users. Suggestion voting can be used to make use of a string when more than one signed-in user agrees, by setting up the 組件配置 with *Suggestion voting* to turn on voting, and *Autoaccept suggestions* to set a threshold for accepted suggestions (this includes a vote from the user making the suggestion if it is cast).

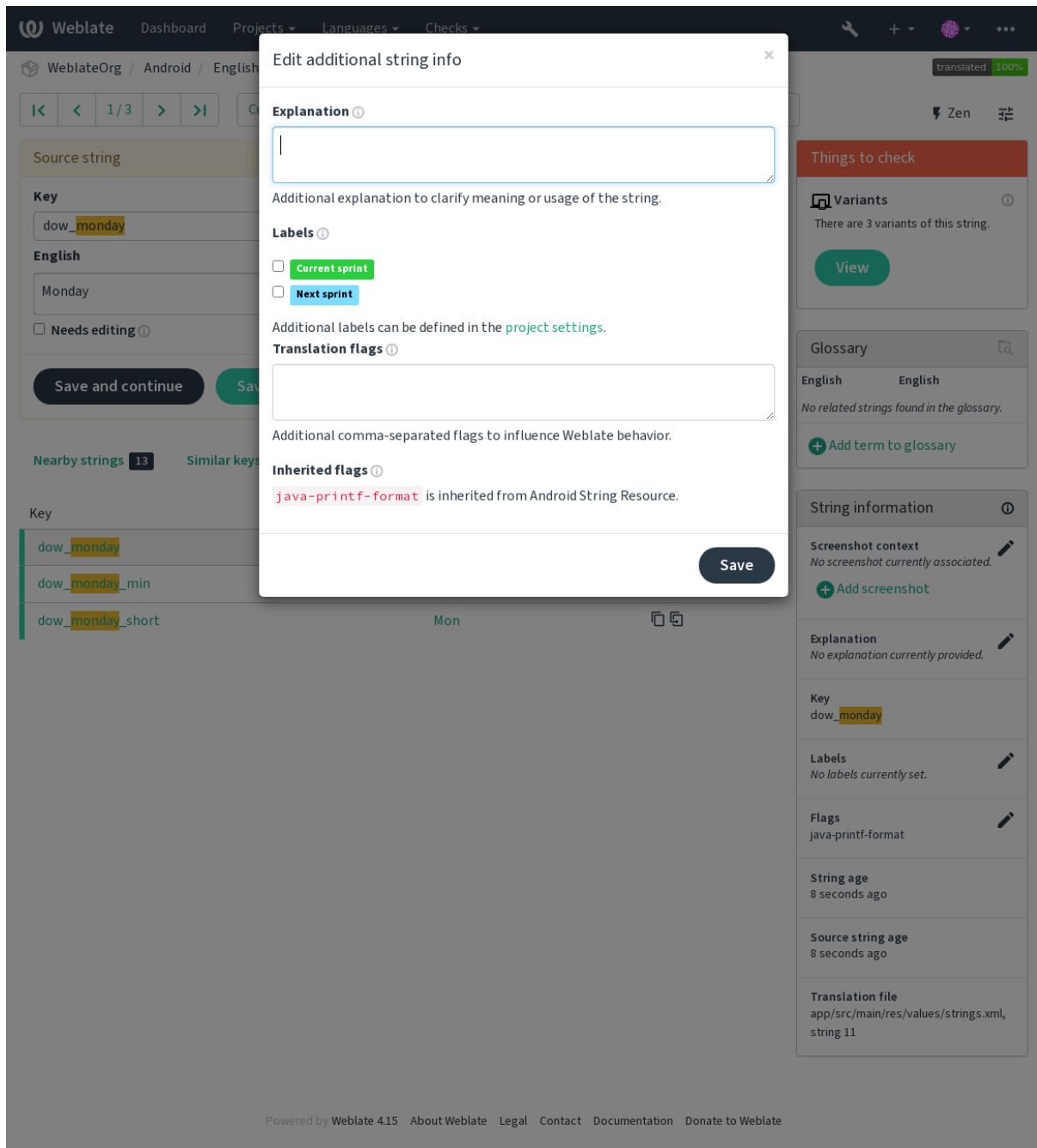
**備：** 一旦新建了自動接受，那 普通用 會失去直接保存翻譯或接受建議的特權。要繞過這一點，可以通過 當建議 制時編輯字符串 *permission*。

您可以將這些與:ref:access control <access-control> 合 到以下設置中的一個：

- 用 建議 對建議進行投票， 且有限的組控制接受什 。——打開投票。——關閉自動接受。——不要讓用 存儲翻譯。
- 用 建議 對建議進行投票，一旦他們之中確定的數量同意則自動接受。——打開投票。——設置自動接受所需要的投票數量。
- 對建議的可選投票。（當用 對做出的多個建議不確定時，可以由用 可選地使用。）——只打開投票。

### 2.11.2 源字符串另外的信息

通過向字符串添加額外的信息來增進翻譯過程，這些信息包括解釋、字符串優先級、檢查標記和可視化上下文。有些信息可以從翻譯文件中提取，有些可以通過編輯額外的字符串信息添加：



可以通過點擊緊鄰 *Screenshot context* 或 *Flags* 的「Edit」標記而從翻譯界面直接訪問。

Weblate
 Dashboard · Projects · Languages · Checks

Translated 90%

<

>

11 / 26

All strings

Position and priority

Zen

Translation

**Explanation**

Help text for automatic translation tool

English

Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.

Czech

Automatický překlad prostřednictvím strojového překládu používá aktivní enginy strojového překládu pro získání nejlepších možných překladů a použije je na tento projekt.

☐ Needs editing

Save and continue Save and stay Suggest Skip

Nearby strings 26 Comments Automatic suggestions Other languages 4 History

Context	English	Czech	Actions
	Files	Soubory	
	Automatic translation	Automatický překlad	
	Add new translation string	Add new translation string	
	Translation status	Stav překladu	
	<b>Singular</b> %(count)s word <b>Plural</b> %(count)s words	<b>One</b> %(count)s slovo <b>Few</b> %(count)s slova <b>Other</b> %(count)s slov	
	Other components	Další součásti	
	Translation file	Soubor s překladem	
	Download	Stáhnout	
	Browse all translation changes	Procházet všechny změny v překladu.	
	Automatic translation takes existing translations in this project and applies them to the current component. It can be used to push translations to a different branch, to fix inconsistent translations or to translate a new component using translation memory.	Automatický překlad použije stávající překlady v projektu na tuto součást. Může být užitečný pro sloučení překladů z jiné větve, opravu nekonzistentních překladů nebo překlad nové součásti pomocí překladové paměti.	
	Automatic translation via machine translation uses active machine translation engines to get the best possible translations and applies them in this project.	Automatický překlad prostřednictvím strojového překládu používá aktivní enginy strojového překládu pro získání nejlepších možných překladů a použije je na tento projekt.	
	You can add new translation string here, it will automatically appear in all translations.	Zde můžete přidat nový řetězec k překladu, automaticky se objeví ve všech jazycích.	
	The uploaded file will be merged with the current translation. In case you want to overwrite already translated strings, don't forget to enable it.	Nahrany soubor bude sloučen se stávajícími překlady. Pokud chcete přepsat již přeložené řetězce, nezapomeňte to povolit.	
	The uploaded file will be merged with the current translation.	Nahrany soubor bude sloučen se stávajícími překlady.	
	The fulltext search might not work properly as the fulltext index for this translation is not yet up to date.	Fulltextové vyhledávání nemusí fungovat správně, protože fulltextový index pro tento překlad ještě není plně zpracován.	
	Review	Kontrola	
	Review translations touched by other users.	Zkontrolovat překlady od ostatních uživatelů.	
	Start review	Začít kontrolu	
	Percent	Procenta	
	Total	Celkem	
	Failing check	Neúspěšných kontrol	
	Last activity	Poslední aktivity	
	Last change	Poslední změna	
	Last author	Poslední autor	
	Question for a mathematics-based CAPTCHA, the %s is an arithmetic problem	Kolik to je?	
	The string uses three dots (...) instead of an ellipsis character (...)		

Glossary

English Czech

machine translation project strojový překlad projekt

+ Add term to glossary

String information

**Screenshot context**

No screenshot currently associated.

+ Add screenshot

**Explanation**

Help text for automatic translation tool

**Labels**

No labels currently set.

**Flags**

No flags currently set.

**Source string location**

website/templates/translation.html:212

**String age**

a second ago

**Source string age**

2 seconds ago

**Translation file**

website/locale/cs/LC\_MESSAGES/djan-go.po, string 11



## 字符串優先級

在 2.0 版本新加入。

使用 `priority` 標識可以更改字符串優先級來 [F](#) 字符串提供更高優先級，以便更早地進行翻譯。

---

**提示：** 這可以用於以邏輯的方式將翻譯流程排序。

---

### 也參考：

[質量檢查](#)

## 翻譯旗標

在 2.4 版本新加入。

在 3.3 版本變更：之前被稱 [F](#) *Quality checks flags*，它不再只配置檢查了。

質量檢查和其他 Weblate 行 [F](#) 的定制，請參見使用標 [F](#) 自定義行 [F](#)。

The string flags are also inherited from the [翻譯旗標](#) at [組件配置](#) and flags from the translation file (see [支持的文件格式](#)).

### 也參考：

[質量檢查](#), 使用標 [F](#) 自定義行 [F](#)

## 解釋

在 4.1 版本變更：在以前的版本中這被稱 [F](#) *Extra context*。

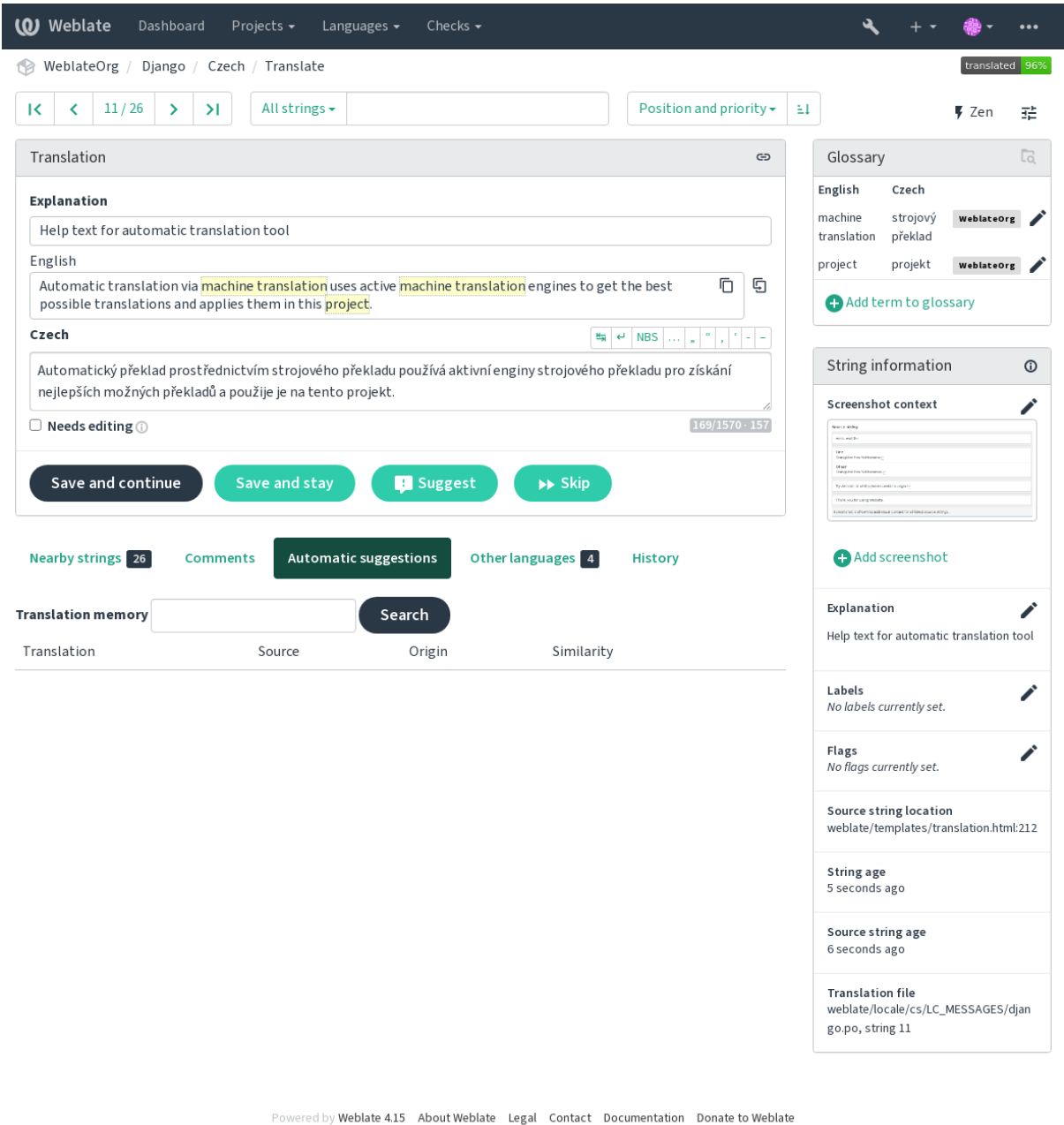
使用解釋來明確翻譯的範圍或翻譯的使用。您可以使用 Markdown 來包括連接和其它標 [F](#)。

## 字符串的可見語境

在 2.9 版本新加入。

你可以將顯示你程序中使用的給定源字符串的截屏上傳。這幫助譯者理解它用在哪 [F](#)，[F](#) 且應該如何翻譯。

上傳的截屏在翻譯語境的側邊條中顯示：



additional 除了源字符串另外的信息，截屏在 *Tools* 菜單下有個單獨的管理界面。上傳截屏，將它們手動分配給源字符串，或者使用光學字符識F（OCR）來進行。

一旦上傳了截屏，那F這個界面處理管理以及源字符串的聯F：

WebateDashboardProjectsLanguagesChecks

WebateOrg / Django / Screenshots / Automatic translation

Screenshot has been uploaded, you can now assign it to source strings.

Assigned source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			
Screenshot is shown to add visual context for all listed source strings.			

Assign source strings

English	Location	Assigned screenshots	Actions
No matching strings found.			

Source string search

Search

Automatically recognize

Image

Source string

Hello, world!

OneOrangutan has %d banana.

OtherOrangutan has %d bananas.

Try Weblate at <http://demo.weblate.org/>!

Thank you for using Weblate.

Screenshot is shown to add visual context for all listed source strings.

Edit screenshot

Screenshot name

Automatic translation

Image

Currently: screenshots/screenshot.png

Change:

Choose File

No file chosen

Upload JPEG or PNG images up to 2000x2000 pixels.

Save

Screenshot details

Created	now
Uploaded by	testuser
Language	English

Delete screenshot

Deleting screenshot will remove it from all associated source strings.

Delete

## 2.12 檢查和修復

### 2.12.1 Custom automatic fixups

還可以應用除了自動修正以外自己的自動修正，`☐`將它們包括到`AUTOFIX_LIST`。

自動修復很`☐`大，但可能導致損壞；寫`☐`本的時候要小心。

例如，後面的自動修復會將每次出現的字符串 `foo` 在翻譯中替`☐☐` `bar`：

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

from django.utils.translation import gettext_lazy as _

from weblate.trans.autofixes.base import AutoFix

class ReplaceFooWithBar(AutoFix):
    """Replace foo with bar."""

    name = _("Foobar")

    def fix_single_target(self, target, source, unit):
        if "foo" in target:
            return target.replace("foo", "bar"), True
        return target, False
```

`☐`了安裝定制的檢查，在`AUTOFIX_LIST` 中`☐` Python 類提供完全合規的路徑，請參見*Custom quality checks, add-ons and auto-fixes*。

### 2.12.2 使用標`☐`自定義行`☐`

You can fine-tune the Weblate behavior by using flags. This can be done on the source string level (see 源字符串另外的信息), or in the 組件配置 (翻譯旗標). Some file formats also allow to specify flags directly in the format (see 支持的文件格式).

標記用逗號分隔，參數用冒號分隔。可以在字符串中使用引號來包含空白字符或特定字符。例如：

```
placeholders:"special:value":"other value", regex:.*
```

Both single and double quotes are accepted, special characters are being escaped using backslash:

```
placeholders:"quoted \"string\"":"'single \'quoted\'"
```

這`☐`是現在能接受的標記的列表：

#### **rst-text**

將文本視`☐` reStructuredText 文檔，影響未更動的翻譯。

#### **dos-eol**

使用 DOS 的行末標記，而不是 Unix 的 (`\r\n` instead of `\n`)。

#### **read-only**

字符串應該只讀，`☐`且不能在 Weblate 中編輯。請參見唯讀字串。

#### **priority:N**

字符串的優先級。高優先級的字符串首先出現被翻譯。默認的優先級是 100，字符串的優先級越高，就會越早安排翻譯。

**max-length:N**

將字符串的最大長度限制為 N 個字符，請參見翻譯最大長度。

**xml-text**

將文本看作 XML 文件，影響 XML 語法和 XML 標記。

**font-family:NAME**

定義 font-family 來提供檢查，請參見管理字型。

**font-weight:WEIGHT**

定義 font-weight 來提供檢查，請參見管理字型。

**font-size:SIZE**

定義 font-size 來提供檢查，請參見管理字型。

**font-spacing:SPACING**

定義渲染檢查的字母間隔，請參見管理字型。

**icu-flags:FLAGS**

Define flags for customizing the behavior of the *ICU MessageFormat* quality check.

**icu-tag-prefix:PREFIX**

Set a required prefix for XML tags for the *ICU MessageFormat* quality check.

**placeholders:NAME:NAME2:...**

翻譯中需要的位字符串，請參見位符。

**replacements:FROM:TO:FROM2:TO2...**

當檢查結果文本參數時執行替換（例如在翻譯的最大長度或翻譯最大長度中）。這一典型應用的情況拓展了非譯元素，確保匹配那些即使使用了長值的文本，例如 `placements:%s:「 John Doe」`。

**variants:SOURCE**

將此字符串標記為具有匹配源的字符串的變體。見 variants。

**regex:REGEX**

Regular expression to match translation, see 正則表達式。

**forbidden**

表示術語表中的禁止翻譯，請參見 REF: “詞表禁止”。

**strict-same**

使用建立的單詞黑名單，來避免“有翻譯”的檢查提示。請參見未更動的翻譯。

**check-glossary**

Enable the 與詞表不同 quality check.

**angularjs-format**

Enable the *AngularJS* 插值字符串 quality check.

**c-format**

Enable the *C* 格式 quality check.

**c-sharp-format**

Enable the *C#* 格式 quality check.

**es-format**

Enable the *ECMAScript* 模板字面值 quality check.

**i18next-interpolation**

Enable the *i18next* 插補 quality check.

**icu-message-format**

Enable the *ICU MessageFormat* quality check.

**java-printf-format**

Enable the *Java* 格式 quality check.

**java-format**

Enable the *Java MessageFormat* quality check.

**javascript-format**

Enable the *JavaScript* 格式 quality check.

**lua-format**

Enable the *Lua* 格式 quality check.

**object-pascal-format**

Enable the *Object Pascal* 格式 quality check.

**percent-placeholders**

Enable the 百分比 F 位符 quality check.

**perl-format**

Enable the *Perl* 格式 quality check.

**php-format**

Enable the *PHP* 格式 quality check.

**python-brace-format**

Enable the *Python* 大括號格式 quality check.

**python-format**

Enable the *Python* 格式 quality check.

**qt-format**

Enable the *Qt* 格式 quality check.

**qt-plural-format**

Enable the *Qt* F 數格式 quality check.

**ruby-format**

Enable the *Ruby* 格式 quality check.

**scheme-format**

Enable the *Scheme* 格式 quality check.

**vue-format**

Enable the *Vue 118n* 格式 quality check.

**md-text**

Treat text as a Markdown document. Enable *Markdown* 連結, *Markdown* 參照, and *Markdown* 語法 quality checks.

**case-insensitive**

Adjust checks behavior to be case-insensitive. Currently affects only F 位符 quality check.

**safe-html**

Enable the 不安全的 *HTML* quality check.

**url**

The string should consist of only a URL. Enable the 網址 quality check.

**ignore-all-checks**

Ignore all quality checks.

**ignore-bbcode**

Skip the *BBCode* 標記 quality check.

**ignore-duplicate**

Skip the 連續重 F 單字 quality check.

**ignore-check-glossary**

Skip the 與詞 F 表不同 quality check.

**ignore-double-space**

Skip the 兩個空白 quality check.

**ignore-angularjs-format**

Skip the *AngularJS* 插值字串 quality check.

**ignore-c-format**Skip the *C* 格式 quality check.**ignore-c-sharp-format**Skip the *C#* 格式 quality check.**ignore-es-format**Skip the *ECMAScript* 模板字面值 quality check.**ignore-i18next-interpolation**Skip the *i18next* 插補 quality check.**ignore-icu-message-format**Skip the *ICU MessageFormat* quality check.**ignore-java-format**Skip the *Java MessageFormat* quality check.**ignore-java-printf-format**Skip the *Java* 格式 quality check.**ignore-javascript-format**Skip the *JavaScript* 格式 quality check.**ignore-lua-format**Skip the *Lua* 格式 quality check.**ignore-object-pascal-format**Skip the *Object Pascal* 格式 quality check.**ignore-percent-placeholders**

Skip the 百分比 F 位符 quality check.

**ignore-perl-format**Skip the *Perl* 格式 quality check.**ignore-php-format**Skip the *PHP* 格式 quality check.**ignore-python-brace-format**Skip the *Python* 大括號格式 quality check.**ignore-python-format**Skip the *Python* 格式 quality check.**ignore-qt-format**Skip the *Qt* 格式 quality check.**ignore-qt-plural-format**Skip the *Qt* F 數格式 quality check.**ignore-ruby-format**Skip the *Ruby* 格式 quality check.**ignore-scheme-format**Skip the *Scheme* 格式 quality check.**ignore-vue-format**Skip the *Vue I18n* 格式 quality check.**ignore-translated**

Skip the 已經翻譯過 quality check.

**ignore-inconsistent**

Skip the 不一致 quality check.

**ignore-kashida**Skip the 使用 *Kashida letter* quality check.

**ignore-md-link**

Skip the *Markdown* 連結 quality check.

**ignore-md-reflink**

Skip the *Markdown* 參照 quality check.

**ignore-md-syntax**

Skip the *Markdown* 語法 quality check.

**ignore-max-length**

Skip the 翻譯最大長度 quality check.

**ignore-max-size**

Skip the 翻譯的最大長度 quality check.

**ignore-escaped-newline**

Skip the *Mismatched* \n quality check.

**ignore-end-colon**

Skip the 冒號不相符 quality check.

**ignore-end-ellipsis**

Skip the F 節號不相符 quality check.

**ignore-end-exclamation**

Skip the 驚嘆號不相符 quality check.

**ignore-end-stop**

Skip the 句號不相符 quality check.

**ignore-end-question**

Skip the 問號不相符 quality check.

**ignore-end-semicolon**

Skip the 分號不相符 quality check.

**ignore-newline-count**

Skip the 斷列符不相配 quality check.

**ignore-plurals**

Skip the 缺少 F 數形 quality check.

**ignore-placeholders**

Skip the F 位符 quality check.

**ignore-punctuation-spacing**

Skip the 標點符號間距 quality check.

**ignore-regex**

Skip the 正則表達式 quality check.

**ignore-same-plurals**

Skip the 相同 F 數形 quality check.

**ignore-begin-newline**

Skip the 開頭 F 列 quality check.

**ignore-begin-space**

Skip the 開頭空格 quality check.

**ignore-end-newline**

Skip the F 列結尾 quality check.

**ignore-end-space**

Skip the 空格結尾 quality check.

**ignore-same**

Skip the 未更動的翻譯 quality check.



**ignore-safe-html**

Skip the 不安全的 *HTML* quality check.

**ignore-url**

Skip the 網址 quality check.

**ignore-xml-tags**

Skip the *XML* 標記 quality check.

**ignore-xml-invalid**

Skip the *XML* 語法 quality check.

**ignore-zero-width-space**

Skip the 零寬度空格 quality check.

**ignore-ellipsis**

Skip the 節號 quality check.

**ignore-icu-message-format-syntax**

Skip the *ICU MessageFormat* 語法 quality check.

**ignore-long-untranslated**

Skip the 長期未翻譯 quality check.

**ignore-multiple-failures**

Skip the 多項未通過查核 quality check.

**ignore-unnamed-format**

Skip the 多個未命名變數 quality check.

**ignore-optional-plural**

Skip the 未數化 quality check.

---

**備註：** 通常規則是，任何檢查都使用識文字來命名 `ignore-*`，所以能將規則應用在定制檢查中。

---

每個源字符串的設置，在組件配置 設置中，且在翻譯文件自身中（例如在 GNU gettext 中），能理解這些標記。

### 2.12.3 強制檢查

在 3.11 版本新加入。

You can configure a list of checks which can not be ignored by setting 強制查核 in 組件配置. Each listed check can not be dismissed in the user interface and any string failing this check is marked as *Needs editing* (see 翻譯狀態).

---

**備註：** Turning on check enforcing doesn't enable it automatically. The check can be turned on by adding the corresponding flag to string or component flags.

---

#### 也參考：

源字符串另外的信息, 翻譯旗標

---

## 2.12.4 管理字型

在 3.7 版本新加入。

**提示：**上傳到 Weblate 的字體純粹用於：REF: “Check-Max-Size” 檢查，它們在 WebLte 用 界面中 有效果。

用於計算呈現文本需要的尺寸的:ref: 『check-max-size』檢查需要字體被加載進 Weblate 被一個翻譯標識選中（見使用標 自定義行 ）。。

在您的翻譯項目 *Manage* 菜單下 *Fonts* 中的 Weblate 字體管理工具提供了接口來上傳 管理字體。可以上傳 TrueType 或 OpenType 字體，設置 font-groups 在檢查中使用它們。

字型組允許 不同語言確定不同字型，這是非拉丁語言中典型需要的：

Font group

<b>Name</b>	default-font		
<b>Default font</b>	Source Sans 3 Bold		
<b>Japanese</b>	language override	Droid Sans Fallback Regular	Remove
<b>Korean</b>	language override	Droid Sans Fallback Regular	Remove
Delete			

Add language override

**Language**

-----

**Font**

-----

Save

Edit font group

**Font group name**

default-font

Identifier you will use in checks to select this font group. Avoid whitespaces and special characters.

**Default font**

Source Sans 3 Bold

Default font is used unless per language override matches.

Save

Powered by Weblate 4.15   About Weblate   Legal   Contact   Documentation   Donate to Weblate

字型組通過名稱識，名稱不能包含空白字符或特殊字符，這使它能容易地用在檢查定義中：

Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Fonts

Font groups
 Fonts

Group name	Default font	Language overrides	
default-font	Source Sans 3 Bold	Japanese: Droid Sans Fallback Regular Korean: Droid Sans Fallback Regular	Edit

Add font group

**Font group name**  
  
 Identifier you will use in checks to select this font group. Avoid whitespaces and special characters.

**Default font**  
  
 Default font is used unless per language override matches.

Save

Powered by Weblate 4.15
 About Weblate
 Legal
 Contact
 Documentation
 Donate to Weblate

字型族和樣式在上傳後自動識 F:

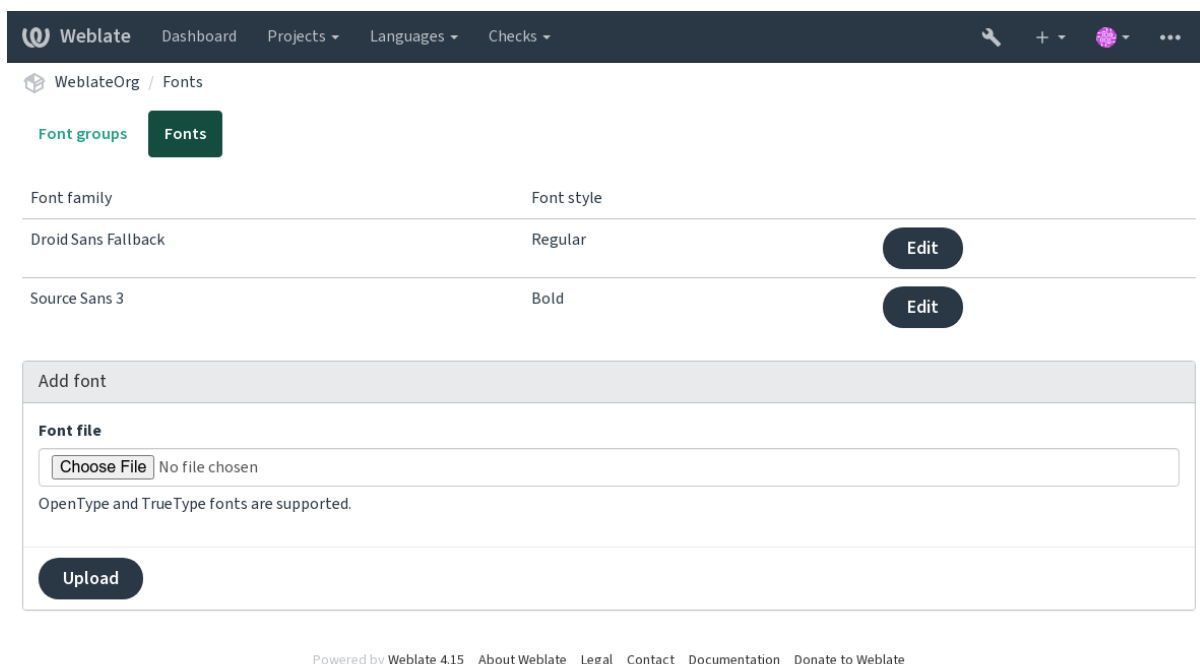
Weblate
 Dashboard Projects Languages Checks

WeblateOrg / Fonts / Droid Sans Fallback Regular

Font	
Font family	Droid Sans Fallback
Font style	Regular
File size	3939852
Created	now
Uploaded by	testuser
Used in groups	
Delete	

Powered by Weblate 4.15
 About Weblate
 Legal
 Contact
 Documentation
 Donate to Weblate

可以將幾種字型加載到 Weblate 中：



為了使用字型來檢查字符串長度，將適當的標記傳遞給它（請參見使用標 F 自定義行 F）。可能會需要後面這些：

**max-size:500**

Defines maximal width in pixels.

**font-family:ubuntu**

確定字型組，通過指定其識 F 文字來使用。

**font-size:22**

Defines font size in pixels.

## 2.12.5 Writing own checks

Weblate F 建了很大範圍的質量檢查，（請參見質量檢查），F 管可能 F 有覆蓋想要檢查的所有 F 容。可以使用 `CHECK_LIST` 來調整執行檢查的列表，也可以添加定制的檢查。

1. Subclass the `weblate.checks.Check`
2. 設置一些屬性。
3. 應用 `check`（如果想要處理代碼中的 F 數的話）或 `check_single` 方法（它將 F 你完成）。

一些例子：

為了安裝定制的檢查，在 `CHECK_LIST` 中 F Python 類提供完全合格的路徑，請參見 *Custom quality checks, add-ons and auto-fixes*。

## 檢查翻譯文本不含有“foo”

這是非常簡單的檢查，只檢查翻譯中是否 F 失了字符串“foo”。

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Simple quality check example."""

from django.utils.translation import gettext_lazy as _
from weblate.checks.base import TargetCheck

class FooCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_single(self, source, target, unit):
        return "foo" in target
```

## 檢查捷克語翻譯文本的 F 數差 F

使用語言信息來檢查，驗證捷克語中的兩種 F 數形式不同。

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Quality check example for Czech plurals."""

from django.utils.translation import gettext_lazy as _
from weblate.checks.base import TargetCheck

class PluralCzechCheck(TargetCheck):

    # Used as identifier for check, should be unique
    # Has to be shorter than 50 characters
    check_id = "foo"

    # Short name used to display failing check
    name = _("Foo check")

    # Description for failing check
    description = _("Your translation is foo")

    # Real check code
    def check_target_unit(self, sources, targets, unit):
        if self.is_language(unit, ("cs",)):
```

(繼續下一頁)

(繼續上一頁)


```
        return targets[1] == targets[2]
    return False





def check_single(self, source, target, unit):
    """We don't check target strings here."""
    return False
```


## 2.13 配置自動建議

在 4.13 版本變更: Prior to Weblate 4.13, the services were configured in the [配置](#).

The support for several machine translation and translation memory services is built-in. Each service can be turned on by the administrator for whole site or at the project settings:


[Dashboard](#)
[Projects ▾](#)
[Languages ▾](#)
[Checks ▾](#)


[WeblateOrg](#) / [Automatic suggestions](#)

Configured automatic suggestion services ⓘ

There are no services currently installed.

Available automatic suggestion services ⓘ

AWS ⓘ	Install
Amagama ⓘ	Install
Apertium APy ⓘ	Install
Baidu ⓘ	Install
DeepL ⓘ	Install
Glosbe ⓘ	Install
Google Translate ⓘ	Install
Google Translate API v3 ⓘ	Install
LibreTranslate ⓘ	Install
Microsoft Terminology ⓘ	Install
Microsoft Translator ⓘ	Install
ModernMT ⓘ	Install
MyMemory ⓘ	Install
Netease Sight ⓘ	Install
SAP Translation Hub ⓘ	Install
Weblate ⓘ	Install
Weblate Translation Memory ⓘ	Install
Yandex ⓘ	Install
Youdao Zhiyun ⓘ	Install
tmserver ⓘ	Install

Some services will ask for additional configuration during installation.

Powered by [Weblate 4.15](#)
[About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

備 F: They come subject to their terms of use, so ensure you are allowed to use them how you want.

The services translate from the source language as configured at [組件配置](#), see [來源語言](#).

**也參考:**

[自動建議](#)

### 2.13.1 Amagama

**服務編號**

amagama

**配置**

*This service has no configuration.*

Special installation of *tmserver* run by the authors of Virtaal.

**也參考:**

Installing amaGama, Amagama, ‘amaGama 翻譯記憶 <<https://amagama.translatehouse.org/>>’\_

### 2.13.2 Apertium APy

**服務編號**

apertium-apy

**配置**

url	API URL	
-----	---------	--

開源原件機器翻譯平台提供一組有限語言的翻譯。

使用 Apertium 的推薦方式是運行您自己的 Apertium-APy 服務器。

**也參考:**

[Apertium website](#), [Apertium APy documentation](#)

### 2.13.3 AWS

在 3.1 版本新加入.

**服務編號**

aws

**配置**

key	存取金鑰 ID	
secret	API 密金鑰	
region	區域名稱	

Amazon Translate 是神經機器翻譯服務，用於將英語與廣泛支持的語言進行互譯。

**也參考:**

[Amazon Translate Documentation](#)



### 2.13.4 Baidu

在 3.2 版本新加入.

服務編號

baidu

配置

key	用 端 ID	
secret	用 端秘密金鑰 (secret)	

由百度提供的機器翻譯服務。

這項服務使用 API, 且您需要從百度獲得 ID 和 API 密鑰來使用它。

也參考:

[Baidu Translate API](#)

### 2.13.5 DeepL

在 2.20 版本新加入.

服務編號

deepl

配置

url	API URL	
key	API 金鑰	

DeepL 是付費服務, 提供一些語言的良好機器翻譯。您需要購買 *DeepL API* 訂, 或者您可以使用傳統的 *DeepL Pro (classic)* 計劃。

API URL to use with the DeepL service. At the time of writing, there is the v1 API as well as a free and a paid version of the v2 API.

**<https://api.deepl.com/v2/>** (在 Weblate 預設使用)

Is meant for API usage on the paid plan, and the subscription is usage-based.

**<https://api-free.deepl.com/v2/>**

Is meant for API usage on the free plan, and the subscription is usage-based.

**<https://api.deepl.com/v1/>**

Is meant for CAT tools and is usable with a per-user subscription.

此前 Weblate 被 DeepL 分類 計算機輔助翻譯工具, 因此應該使用 v1 API, 但現在應該使用 v2 API。這樣默認 v2, 在你有現有的計算機輔助翻譯工具訂, 想要 Weblate 使用它的情下, 可以將其更改 v1。

The easiest way to find out which one to use is to open an URL like the following in your browser:

[https://api.deepl.com/v2/translate?text=Hello&target\\_lang=FR&auth\\_key=XXX](https://api.deepl.com/v2/translate?text=Hello&target_lang=FR&auth_key=XXX)

Replace the XXX with your auth\_key. If you receive a JSON object which contains 「Bonjour」, you have the correct URL; if not, try the other three.

Weblate supports DeepL formality, it will choose matching one based on the language (for example, there is de@formal and de@informal).

也參考:

[DeepL website](#), [DeepL pricing](#), [DeepL API documentation](#)

### 2.13.6 Glosbe

服務編號

glosbe

配置

*This service has no configuration.*

幾乎每一種活語言的免費字典與翻譯服務。

The API is gratis to use, but usage of the translations is subject to the license of the used data source. There is a limit of calls that may be done from one IP in a set period of time, to prevent abuse.

也參考:

[Glosbe website](#)

### 2.13.7 谷歌翻譯

服務編號

google-translate

配置

key	API 金鑰	
-----	--------	--

Google 提供的機器翻譯服務。

這項服務使用了 Google Translation API，您需要得到 API 密鑰，[F](#)在 Google API 控制台打開記費。

也參考:

[Google translate documentation](#)

### 2.13.8 Google Translate API v3

服務編號

google-translate-api-v3

配置

credentials	Google 翻譯服務帳號資訊	
project	Google 翻譯專案	
location	Google Translate 位置	

Google 雲服務提供的機器翻譯服務。

也參考:

[Google translate documentation](#), [Authenticate to Cloud services using client libraries](#), [Creating Google Translate project](#), [Google Cloud App Engine locations](#)

### 2.13.9 LibreTranslate

在 4.7.1 版本新加入。

**服務編號**

libretranslate

**配置**

url	API URL	
key	API 金鑰	

LibreTranslate is a free and open-source service for machine translations. The public instance requires an API key, but LibreTranslate can be self-hosted and there are several mirrors available to use the API for free.

**<https://libretranslate.com/>** (官方公開服務)

Requires an API key to use outside of the website.

**也參考:**

[LibreTranslate website](#), [LibreTranslate repository](#), [LibreTranslate mirrors](#)

### 2.13.10 Microsoft Terminology

在 2.19 版本新加入。

**服務編號**

microsoft-terminology

**配置**

*This service has no configuration.*

Microsoft Terminology Service API 允許您通過 Web 服務，可編程地訪問 Language Portal 上可用的術語、定義和用⌘界面（UI）字符串。

**也參考:**

[微軟術語服務 API](#)

### 2.13.11 微軟的 Translator

在 2.10 版本新加入。

**服務編號**

microsoft-translator

**配置**

key	API 金鑰	
endpoint_url	應用程式端點 URL	
base_url	應用程式基礎 URL	Available choices: api.cognitive.microsofttranslator.com –Global (non-regional) api-apc.cognitive.microsofttranslator.com –Asia Pacific api-eur.cognitive.microsofttranslator.com –Europe api-nam.cognitive.microsofttranslator.com –North America api.translator.azure.cn –中國
region	應用程式區域	

由 Microsoft 在 Azure 門 F 提供的機器翻譯服務，作 F Cognitive Services 的一種。

Weblate 實施了 Translator API V3.

### Translator Text API 第二版

您用於 Translator API V2 的密鑰可以用於 API 3。

### Translator Text API 第三版

You need to register at Azure portal and use the key you obtain there. With new Azure keys, you also need to set region to locale of your service.

---

**提示：** 對於 Azure 中國，請使用你的 Azure Portal 的端點。

---

#### 也參考：

Cognitive Services - Text Translation API, Microsoft Azure Portal, Base URLs, 「Authenticating with a Multi-service resource」 「Authenticating with an access token」 section

### 2.13.12 ModernMT

在 4.2 版本新加入。

**服務編號**

modernmt

**配置**

url	API URL	
key	API 金鑰	

**也參考:**

[ModernMT API](#),

### 2.13.13 MyMemory

**服務編號**

mymemory

**配置**

email	聯絡信箱	
username	使用者名稱	
key	API 金鑰	

使用機器翻譯的巨量翻譯記憶庫。

Free, anonymous usage is currently limited to 100 requests/day, or to 1000 requests/day when you provide a contact e-mail address in `email`. You can also ask them for more.

**也參考:**

[MyMemory website](#)

### 2.13.14 Netease Sight

在 3.3 版本新加入。

**服務編號**

netease-sight

**配置**

key	用Ⓔ端 ID	
secret	用Ⓔ端秘密金鑰 (secret)	

Machine translation service provided by NetEase.

這項服務使用 API，兵器額您需要從網易得到密鑰和密碼。

**也參考:**

[NetEase Sight Translation Platform](#)

## 2.13.15 SAP Translation Hub

### 服務編號

sap-translation-hub

### 配置

url	API URL	
key	API 金鑰	
username	SAP 使用者名稱	
password	SAP 密碼	
enable_mt	啟用機器翻譯	

SAP 提供的機器翻一下服務。

You need to have a SAP account (and the SAP Translation Hub enabled in the SAP Cloud Platform) to use this service.

You can also configure whether to also use machine translation services, in addition to the term database.

備註: To access the Sandbox API, you need to set url and key.

為了存取正式環境的 API，您需要設定 url, username 與 password。

### 也參考:

[SAP Translation Hub API, Building the Base URL of SAP Translation Hub](#)

## 2.13.16 tmserver

### 服務編號

tmserver

### 配置

url	API URL	
-----	---------	--

您可以通過使用 Translate-toolkit 綁定的一個服務器與之對話，來運行您自己的翻譯服務器。您還可以將它與 amaGama 服務器一起使用，它是 tmserver 的增進版本。

1. 首先您會想要將一些數據導入翻譯記憶庫：

```
build_tmdb -d /var/lib/tm/db -s en -t cs locale/cs/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t de locale/de/LC_MESSAGES/django.po
build_tmdb -d /var/lib/tm/db -s en -t fr locale/fr/LC_MESSAGES/django.po
```

2. 啟動 tmserver 來收聽您的請求：

```
tmserver -d /var/lib/tm/db
```

3. 設定 Weblate 與它建立連，預設的 URL 是 `http://localhost:8888/tmserver/`。

### 也參考:

[tmserver Installing amaGama, Amagama, Amagama Translation Memory](#)

### 2.13.17 Weblate

#### 服務編號

weblate

#### 配置

*This service has no configuration.*

Weblate machine translation service can provide translations for strings that are already translated inside Weblate. It looks for exact matches in the existing strings.

### 2.13.18 Weblate Translation Memory

在 2.20 版本新加入.

#### 服務編號

weblate-translation-memory

#### 配置

*This service has no configuration.*

Use 翻譯記憶 as a machine translation service. Any string that has been translated in past (or uploaded to the translation memory) can be translated in this way.

### 2.13.19 Yandex

#### 服務編號

yandex

#### 配置

key	API 金鑰	
-----	--------	--

Yandex 提供的機器翻譯服務。

這項服務使用翻譯 API，您需要從 Yandex 得到 API 密鑰。

#### 也參考:

[Yandex Translate API, Powered by Yandex.Translate](#)

### 2.13.20 Youdao Zhiyun

在 3.2 版本新加入.

#### 服務編號

youdao-zhiyun

#### 配置

key	用 F 端 ID	
secret	用 F 端秘密金鑰 (secret)	

有道提供的機器翻譯服務。

這項服務使用 API，您需要從有道獲得 ID 和 API 密鑰。

#### 也參考:

[Youdao Zhiyun Natural Language Translation Service](#)

### 2.13.21 Custom machine translation

您還可以通過使用一些 Python 代碼來實施您自己的機器翻譯服務。這個例子使用 `dictionary` Python 模塊來實施一組固定語言的機器翻譯：

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Machine translation example."""

import dictionary

from weblate.machinery.base import MachineTranslation

class SampleTranslation(MachineTranslation):
    """Sample machine translation interface."""

    name = "Sample"

    def download_languages(self):
        """Return list of languages your machine translation supports."""
        return {"cs"}

    def download_translations(
        self,
        source,
        language,
        text: str,
        unit,
        user,
        search: bool,
        threshold: int = 75,
    ):
        """Return tuple with translations."""
        for t in dictionary.translate(text):
            yield {"text": t, "quality": 100, "service": self.name, "source": text}
```

You can list your own class in `WEBLATE_MACHINERY` and Weblate will start using that.

## 2.14 附加元件

在 2.19 版本新加入。






附加組件提供了自定義和自動化翻譯工作流程的方法。管理員可以從每個相應翻譯組件的 *Manage* ↓ :guilabel: Addons 菜單添加和管理附加組件。


---

**提示：** You can also configure add-ons using *API*, *DEFAULT\_ADDONS*, or *install\_addon*.

---




 Weblate
 Dashboard Projects Languages Checks
 




 WeblateOrg / Language names / Add-ons

Installed add-ons ⓘ


There are no add-ons currently installed.

Available add-ons ⓘ

 Automatic translation ⓘ

Automatically translates strings using machine translation or other components.


Install

 Add missing languages ⓘ

Ensures a consistent set of languages is used for all components within a project.

project wide


Install

 Component discovery ⓘ

Automatically adds or removes project components based on file changes in the version control system.


repository wide

Install

 Bulk edit ⓘ


Bulkedit flags, labels, or states of strings.

Install

 Statistics generator ⓘ


Generates a file containing detailed info about the translation status.

Install

 Prefill translation with source ⓘ


Fills in translation strings with source string.

Install

 Pseudolocale generation ⓘ


Generates a translation by adding prefix and suffix to source strings automatically.

Install

 Contributors in comment ⓘ


Updates the comment part of the PO file header to include contributor names and years of contributions.

Install

 Customize gettext output ⓘ


Allows customization of gettext output behavior, for example line wrapping.

Install

 Generate MO files ⓘ


Automatically generates a MO file for every changed PO file.

Install

 Update PO files to match POT (msgmerge) ⓘ

Updates all PO files (as configured by "File mask") to match the POT file (as configured by "Template for new translations") using msgmerge.


Install

 Squash Git commits ⓘ

Squash Git commits prior to pushing changes.

repository wide


Install

 Stale comment removal ⓘ

Set a timeframe for removal of comments.

project wide

Install

 Stale suggestion removal ⓘ

Set a timeframe for removal of suggestions.

project wide

Install

Some add-ons will ask for additional configuration during installation.

## 2.14.1 Built-in add-ons

### 自動翻譯

在 3.9 版本新加入。

#### Add-on ID

weblate.autotranslate.autotranslate

#### 配置

mode	自動翻譯模式	Available choices: suggest -Add as suggestion translate -Add as translation fuzzy -Add as 「Needing edit」
filter-type	搜尋篩選	請注意：翻譯所有字串會包含所有現有翻譯。 Available choices: all -All strings nottranslated -Untranslated strings todo -Unfinished strings fuzzy -Strings marked for edit check:inconsistent -錯誤檢查：不一致
auto-source	自動翻譯的來源	Available choices: others -Other translation components mt -Machine translation
component-engine	組件機器翻譯引擎	輸入要用作來源的組件 slug。留空則使用本專案的所有組件。
threshold	分數值	

#### 觸發

component update, daily

使用機器翻譯或其他組件自動翻譯字串。

It is triggered:

- When new strings appear in a component.
- 每個組件的一個月，可以使用以下配置：設置： *background\_tasks*。

#### 也參考：

自動翻譯, 跨組件保持翻譯一致

## JavaScript 在地化 CDN

在 4.2 版本新加入。

#### Add-on ID

weblate.cdn.cdnjs

#### 配置

threshold	最低翻譯數	翻譯收 的最低門檻。
css_selector	CSS 選擇器	用來偵測可在地化元素的 CSS 選擇器。
cookie_name	語言 Cookie 名稱	儲存語言偏好設定的 Cookie 名稱。
files	從 HTML 檔案 取字串	在目前儲存庫或遠端 URL 中，要解析其可翻譯字串的檔案名稱清單。

**觸發**

每日，儲存庫提交後、儲存庫更新後

將翻譯發布給 容遞交網路，供 JavaScript 或 HTML 在地化處理使用。

可以用於本地化 態 HTML 網頁，或者在 JavaScript 代碼中加載本地化文件。

你的組件生成一個唯一的 URL，你可以將其包含在 HTML 頁面中以本地化它們。詳情見 weblate-cdn。

**也參考：**

cdn-addon-config, weblate-cdn, cdn-addon-extract, cdn-addon-html

**移除空白字串**

在 4.4 版本新加入。

**Add-on ID**

weblate.cleanup.blank

**配置**

*This add-on has no configuration.*

**觸發**

儲存庫提交後、儲存庫更新後

從翻譯檔中移除 有譯文的字串。

使用此方法可以使翻譯文件中不存在任何空字符串 (例如，如果你的本地化庫將它們顯示 缺失，而不是退回到源字符串)。

**也參考：**

*Weblate 除了更新翻譯，還更新翻譯文件嗎？*

**清理翻譯檔****Add-on ID**

weblate.cleanup.generic

**配置**

*This add-on has no configuration.*

**觸發**

儲存庫提交前、儲存庫更新前

更新所有翻譯檔以符合單語言基礎檔。對大多數檔案格式而言，這代表會移除基礎檔中不再存在的失時效翻譯金鑰。

**也參考：**

*Weblate 除了更新翻譯，還更新翻譯文件嗎？*

## 加入遺漏語言

### Add-on ID

```
weblate.consistency.languages
```

### 配置

*This add-on has no configuration.*

### 觸發

daily, repository post-add

確保對一個項目 F 所有組件使用一致的一組語言。

每隔 24 小時，和在 Weblate 中加入新語言時，會檢查一次缺失的語言。

不像其他多數附加組件，這個附加組件影響整個項目。

---

**提示：** 用 [自動翻譯](#) 自動翻譯新添加的字符串。

---

## 組件探索

### Add-on ID

```
weblate.discovery.discovery
```

### 配置

match	用來比對出翻譯檔的正則表達式	
file_format	檔案格式	
name_template	自訂組件名稱	
base_file_template	定義單語的基礎檔名	保留雙語翻譯檔空白。
new_base_file_template	定義新翻譯的基礎檔	用來建立新翻譯的檔案名稱。例如 <code>gettext</code> ，請選擇 <code>.pot</code> 檔。
intermediate_file_template	中間語言檔案	中間翻譯檔的檔名。大部分情況下，這即是開發者提供的翻譯檔，會在建立實際來源字串時使用。
language_regex	語言篩選	掃描檔案遮罩時用以篩選過濾翻譯檔的正則表達式。
copy_addons	從主組件 F 附加元件至新建立的組件	
remove	移除檔案不存在的組件	
confirm	我能確認上述符合項目看起來正確	

### 觸發

repository post-update

根據版本控制系統中的檔案更動自動加入或移除組件專案。

每次更新 VCS 時觸發，在其他方 F 類似 `djadmin:import_project` 管理命令。通過這種方式，你可以在一個 VCS 中跟踪多個翻譯組件。

該匹配是使用允許 F 雜配置的正則表達式完成的，但這樣做需要一些知識。一些常見用例的例子可以在附加組件幫助部分找到。

一旦點擊了 *Save*，將顯示匹配組件的預覽，可以檢查配置是否匹配於自己的需要：

Weblate

Dashboard

Projects

Languages

Checks

+

▼

WeblateOrg

/

Language names

/

Add-ons

/

Component discovery

Configure add-on

Please review and confirm the matched components.

Component

Matched files

The following components would be created

Djangojs

File mask: `weblate/locale/*/LC_MESSAGES/djangojs.po`  
`weblate/locale/hu/LC_MESSAGES/djangojs.po` (hu)  
`weblate/locale/he/LC_MESSAGES/djangojs.po` (he)  
`weblate/locale/cs/LC_MESSAGES/djangojs.po` (cs)

Django

File mask: `weblate/locale/*/LC_MESSAGES/django.po`  
`weblate/locale/he/LC_MESSAGES/django.po` (he)  
`weblate/locale/hu/LC_MESSAGES/django.po` (hu)  
`weblate/locale/cs/LC_MESSAGES/django.po` (cs)

☐ I confirm the above matches look correct

Regular expression to match translation files against

`weblate/locale/(?P<language>["/"]*)/LC_MESSAGES/(?P<component>["/"]*)\.po`

File format

gettext PO file

Customize the component name

`{{ component|title }}`

Define the monolingual base filename

Leave empty for bilingual translation files.

Define the base file for new translations

`weblate/locale/{{ component }}.pot`

Filename of file used for creating new translations. For gettext choose .pot file.

Intermediate language file

Filename of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Language filter

`^[cs|he|hu]$`

Regular expression to filter translation files against when scanning for file mask.

☒ Clone add-ons from the main component to the newly created ones

☐ Remove components for inexistent files

The regular expression to match translation files has to contain two named groups to match component and language, some examples:

Regular expression	Example matched files	Description
<code>(?P&lt;language&gt;["/"].)*/(?P&lt;component&gt;["/"]*)\.po</code>	<code>cs/application.po</code> <code>cs/website.po</code> <code>de/application.po</code> <code>de/website.po</code>	One folder per language containing translation files for components.
<code>locale/(?P&lt;language&gt;["/"].)*LC_MESSAGES/(?P&lt;component&gt;["/"]*)\.po</code>	<code>locale/cs/LC_MESSAGES/application.po</code> <code>locale/cs/LC_MESSAGES/website.po</code> <code>locale/de/LC_MESSAGES/application.po</code> <code>locale/de/LC_MESSAGES/website.po</code>	Usual structure for storing gettext PO files.
<code>src/locale/(?P&lt;component&gt;["/"]*)\. (?P&lt;language&gt;["/"].)*\.po</code>	<code>src/locale/application.cs.po</code> <code>src/locale/website.cs.po</code> <code>src/locale/application.de.po</code> <code>src/locale/website.de.po</code>	Using both component and language name within filename.
<code>locale/(?P&lt;language&gt;["/"].)*/(?P&lt;component&gt;["/"]*)/(?P&lt;language&gt;["/"].)*\.po</code>	<code>locale/cs/application/cs.po</code> <code>locale/cs/website/cs.po</code> <code>locale/de/application/de.po</code> <code>locale/de/website/de.po</code>	Using language in both path and filename.
<code>res/values-(?P&lt;language&gt;["/"].)*strings-(?P&lt;component&gt;["/"]*)\.xml</code>	<code>res/values-cs/strings-about.xml</code> <code>res/values-cs/strings-help.xml</code> <code>res/values-de/strings-about.xml</code> <code>res/values-de/strings-help.xml</code>	Android resource strings, split into several files.
<code>(?P&lt;originalHierarchy&gt;.*)(?P&lt;component&gt;["/"]*)/src/main/resources/ApplicationResources_(?P&lt;language&gt;["/"].)*.properties</code>	<code>parent/module1/submodule/src/main/resources/ApplicationResources_fr.properties</code> <code>parent/module1/submodule/src/main/resources/ApplicationResource_es.properties</code> <code>parent/module2/src/main/resources/ApplicationResource_de.properties</code> <code>parent/module2/src/main/resources/ApplicationResource_ro.properties</code>	Multi-module Maven project with Java properties translations.

You can use Django template markup in both component name and the monolingual base filename, for example:

`{{ component }}`  
Component filename match  
`{{ component|title }}`  
Component filename with upper case first letter

Save

Powered by Weblate 4.15

About Weblate

Legal

Contact

Documentation

Donate to Weblate

**提示：** 組件發現附加用途：參考：“部 URL”。它是在多個組件之間共享 VCS 設置的便捷方式。鏈接組件使用通過填寫“webbleate: // project / main-component“中設置的主要組件的本地存儲庫：ref: ‘component-repo’字段（： guilabel: ‘管理’↓： guilabel: “設置” ➤ “： 締約方標： 每個相應組件的” 版本控制系統 “）。這也可以節省時間和系統資源。

**也參考：**

模板標記

大量編輯

在 3.11 版本新加入.

Add-on ID

weblate.flags.bulk

配置

q	查詢	
state	要設定的狀態	Available choices: -1 –Do not change 10 –Needs editing 20 –Translated 30 –Approved
add_flags	要加入的翻譯旗標	
remove_flag	要移除的翻譯標	
add_labels	要加入的標	
remove_labels	要移除的標	

觸發

component update

大量編輯字串的旗標、標或狀態。

要自動化標操作，從搜索字符串“NOT has:label“入手， 添加標直到所有字符串都有所需的標。也可以完成 Weblate 元數據的其他自動化操作。

**範例：**

表格 5: 自動標於新字符串

Search query	NOT has:label
要加入的標	recent

表格 6: Marking all App store 中介資料檔 changelog strings read-only

Search query	language:en AND key:changelogs/
要加入的翻譯旗標	read-only

**也參考：**

大量編輯, 使用標自定義行, labels

### 將未變動的翻譯標記「需要編輯」

在 3.1 版本新加入.

**Add-on ID**

`weblate.flags.same_edit`

**配置**

*This add-on has no configuration.*

**觸發**

元件建立前

每當新的可翻譯字串從 VCS 匯入且符合來源字串時，會在 Weblate 中標記需要編輯。這對於未翻譯字串有來源字串的檔案格式來特別有用。

---

**提示：** 您或許也想要加未更動的翻譯的檢查透過加入 `strict-same` 標記到翻譯旗標 中。

---

**也參考：**

[翻譯狀態](#)

### 標記新來源字串「需要編輯」

**Add-on ID**

`weblate.flags.source_edit`

**配置**

*This add-on has no configuration.*

**觸發**

元件建立前

每當新來源字串由 VCS 匯入時，會在 Weblate 中標記需要編輯。這樣您可以輕鬆過濾編輯開發者寫的來源字串。

**也參考：**

[翻譯狀態](#)

### 將新翻譯標記「需要編輯」

**Add-on ID**

`weblate.flags.target_edit`

**配置**

*This add-on has no configuration.*

**觸發**

元件建立前

每當新的可翻譯字串從 VCS 匯入時，會在 Weblate 中標記需要編輯。這樣您可以輕鬆過濾編輯開發者建立的翻譯。

**也參考：**

[翻譯狀態](#)

## 統計數據生成器

## Add-on ID

weblate.generate.generate

## 配置

filename	生成的檔案名稱	
template	生成的檔案 F 容	

## 觸發

repository pre-commit

生成 F 含翻譯狀態詳細資訊的檔案。

You can use a Django template in both filename and content, see [模板標記](#) for a detailed markup description.

For example generating a summary file for each translation:

## 生成的檔案名稱

locale/{{ language\_code }}.json

## F 容

```
{
  "language": "{{ language_code }}",
  "strings": "{{ stats.all }}",
  "translated": "{{ stats.translated }}",
  "last_changed": "{{ stats.last_changed }}",
  "last_author": "{{ stats.last_author }}"
}
```

## 也參考:

[模板標記](#)

## 將原文預先填充進翻譯

在 4.11 版本新加入.

## Add-on ID

weblate.generate.prefill

## 配置

*This add-on has no configuration.*

## 觸發

component update, daily

用來源字串填充翻譯字串。

所有在此組件中未翻譯的字串將會使用來源字串取代，F 標記 F 需要編輯。請使用此功能當您不允許有空字串於翻譯檔案。



語系生

在 4.5 版本新加入.

Add-on ID

```
weblate.generate.pseudolocale
```

配置

source	來源字串	
target	目標翻譯	本翻譯中的所有字串都將被覆寫
prefix	固定字串前綴	
var_prefix	可變字串前綴	
suffix	固定字串後綴	
var_suffix	可變字串後綴	
var_multiplier	可變部分乘數	重可變部分的次數多寡，取於來源字串的長度。
include_readonly	納入唯讀字串	

觸發

```
component update, daily
```

藉由在來源字串加前綴與後綴，來自動生成語系翻譯。

locale 對於查找未準備好進行本地化的字符串很有用。這是通過修改所有可翻譯的源字符串來實現的，使得在用 locale 語言運行應用程序時很容易發現未修改的字符串。

發現本地化對應物可能不適合局的字符串也是可能的。

使用變數部分可以查找在本地化之後可能無法適合使用者界面的字串 - 它基於來源字串長度來擴充。變數部分可以藉由重字數來表達。例如：使用 \_ 當作 你好的後綴字及其數量代表長度可 你好 - 這的 \_ 重幾次代表相對於來源字串多少字。

字串將以下述模式生：

```
Fixed string prefix Variable string prefix Source string Variable string suffix Fixed string suffix
```

**提示：** 可以使用真正的語言進行檢測，但在 Weblate 中有專用的假語言環境——en\_XA 和 ar\_XB。

**提示：** 您可以使用這個擴充套件來開始翻譯現有或相似的語言。一旦您加入翻譯到這組件，也將會套用擴充套件。例如：您有 fr 且想加入 fr\_CA 翻譯，只需要將 fr 設定來源語言，fr\_CA 目標翻譯，將前後綴留空。

在填滿新翻譯後卸載擴充套件以防止 Weblate 更改後的翻譯。

## 評 中的貢獻者紀

## Add-on ID

```
weblate.gettext.authors
```

## 配置

*This add-on has no configuration.*

## 觸發

```
repository pre-commit
```

更新 PO 檔標頭中的評 部分以納入貢獻者姓名及貢獻年份。

PO 文件頭看上去是這樣的：

```
# Michal Čihař <michal@weblate.org>, 2012, 2018, 2019, 2020.
# Pavel Borecki <pavel@example.com>, 2018, 2019.
# Filip Hron <filip@example.com>, 2018, 2019.
# anonymous <noreply@weblate.org>, 2019.
```

## 更新「configure」檔案中的 ALL\_LINGUAS 變數

## Add-on ID

```
weblate.gettext.configure
```

## 配置

*This add-on has no configuration.*

## 觸發

```
repository post-add, daily
```

當新的翻譯添加時，更新 configure 、 configure.in 或任何 configure.ac 文件中的 ALL\_LINGUAS 變量。

## 自訂 gettext 輸出

## Add-on ID

```
weblate.gettext.customize
```

## 配置

width	長列	預設情況下，gettext 會在第 77 個字元和新列字元處 ；加上-no-wrap 參數後，則僅在新列字元處 。
	列	
		Available choices:
		77 –Wrap lines at 77 characters and at newlines (xgettext default)
		65535 –Only wrap lines at newlines (like 『xgettext -no-wrap』)
		-1 –No line wrapping

## 觸發

儲存後載入

允許自訂 gettext 輸出行 ，例如 。

提供了後面的選項：

- 於第 77 個字元處與新列處 。
- 僅在新列處 。
- 不要 。

備： By default gettext wraps lines at 77 characters and at newlines. With the `--no-wrap` parameter, wrapping is only done at newlines.

更新 LINGUAS 檔案

Add-on ID

`weblate.gettext.linguas`

配置

*This add-on has no configuration.*

觸發

repository post-add, daily

當新增翻譯時更新 LINGUAS 檔。

生成 MO 檔

Add-on ID

`weblate.gettext.mo`

配置

<code>path</code>	生成的 MO 檔路徑	如果未指定，將使用 PO 檔的位置。
-------------------	------------	--------------------

觸發

repository pre-commit

每次 PO 檔有變動時便自動生成 MO 檔。

生成的 MO 文件的位置可以定制化，且其字段使用模板標記。

更新 PO 檔以符合 POT (msgmerge)

Add-on ID

`weblate.gettext.msgmerge`

配置

<code>previous</code>	保留翻譯字串上次的 msgid	
<code>no_location</code>	移除翻譯字串的位置	
<code>fuzzy</code>	使用模糊比對	

觸發

repository post-update

Updates all PO files (as configured by 文件掩碼) to match the POT file (as configured by 新翻譯的模板) using **msgmerge**.

Triggered whenever new changes are pulled from the upstream repository. Most msgmerge command-line options can be set up through the add-on configuration.

也參考：

*Weblate 除了更新翻譯，還更新翻譯文件嗎？*

匯 F Git 提交

Add-on ID

weblate.git.squash

配置

squash	提交匯 F	Available choices: all –All commits into one language –Per language file –每一檔案 author –每一作者
append_translation_comments	將額外資訊加入至壓縮提交的訊息	額外資訊的列看起來類似於 RFC 822 電子信箱標頭，位於提交訊息的其他自由格式部分的末尾，例如「Co-authored-by: …」。
commit_message	提交訊息	此提交訊息將用來取代從壓縮提交組合而成的提交訊息。

觸發

repository post-commit

推送更動前先匯 F Git 提交。

以下模式之一中，Git 提交可以在推送更改之前被壓縮：

- 所有提交匯 F F 一
- 每個語言
- 每個檔案
- 每位作者

Original commit messages are kept, but authorship is lost unless *Per author* is selected, or the commit message is customized to include it.

The original commit messages can optionally be overridden with a custom commit message.

預告（提交行像 Co-authored-by: …）可選地從原始提交信息中去掉，F 且添加在去掉的提交信息後面。這還可以 F 每一位翻譯者 F 生適當的 Co-authored-by: 信譽。

自訂 JSON 輸出

Add-on ID

weblate.json.customize

配置

sort_keys	短 JSON 金鑰	
indent	JSON 縮排	
style	JSON 縮排類型	Available choices: spaces –空格 tabs –Tabs

觸發

儲存後載入

允許調整 JSON 輸出行 F，例如縮排或排序。

## 格式化 Java properties 檔案

### Add-on ID

weblate.properties.sort

### 配置

*This add-on has no configuration.*

### 觸發

repository pre-commit

格式化 排序 Java properties 檔案。

- 將新行合 Unix ones。
- Unicode 跳 符號的大寫格式（如果有的話）。
- Strips blank lines and comments.
- Sorts the strings by the keys.
- Drops duplicate strings.

## 陳舊評 移除

在 3.7 版本新加入。

### Add-on ID

weblate.removal.comments

### 配置

age	保留的天數	
-----	-------	--

### 觸發

每日

設定移除評 的時間間隔。

這可以用於 除可能變得過時的陳舊 釋。小心使用，因 陳舊的 釋不意味著失去了重要性。

## 陳舊建議移除

在 3.7 版本新加入。

### Add-on ID

weblate.removal.suggestions

### 配置

age	保留的天數	
votes	最低投票數	移除之最低票數。當停用投票時，這個欄位不會有效果。

### 觸發

每日

設定移除建議的時間間隔。

此附加組件在與建議投票一道用來 除在給定的時間 有得到足 的正面投票的建議方面非常有用。

更新 RESX 檔案

在 3.9 版本新加入.

**Add-on ID**  
weblate.resx.update

**配置**  
*This add-on has no configuration.*

**觸發**  
repository post-update

更新所有翻譯檔以符合單語言上游基底檔。未使用的字串將移除，新增加的字串將根據來源字串加入。

**提示：** 如果只想刪除陳舊的翻譯鍵，那使用清理翻譯檔。

也參考：

Weblate 除了更新翻譯，還更新翻譯文件嗎？

Customize XML output

在 4.15 版本新加入.

**Add-on ID**  
weblate.xml.customize

**配置**

closing_tags	Include closing tag for blank XML tags
--------------	--

**觸發**  
儲存後載入

Allows adjusting XML output behavior, for example closing tags instead of self- closing tags for empty tags.

自訂 YAML 輸出

在 3.10.2 版本新加入.

**Add-on ID**  
weblate.yaml.customize

**配置**

indent	YAML 縮排	
width	長 列 列	Available choices: 80 –Wrap lines at 80 chars 100 –Wrap lines at 100 chars 120 –Wrap lines at 120 chars 180 –Wrap lines at 180 chars 65535 –No line wrapping
line_break	行 符號	Available choices: dos –DOS (\r\n) unix –UNIX (\n) mac –MAC (\r)

**觸發**

儲存後載入

允許調整 YAML 輸出行，例如列長或列。

## 2.14.2 Customizing list of add-ons

附加組件列表由 `WEBLATE_ADDONS` 配置。要添加其他附加組件，只需在這個設置中包含對類名稱即可。

## 2.14.3 Writing add-on

You can write your own add-ons too, create a subclass of `weblate.addons.base.BaseAddon` to define the add-on metadata, and then implement a callback to do the processing.

**也參考:**

*Developing add-ons*

## 2.14.4 Executing scripts from add-on

附加逐漸還可以用於執行外部本。這曾經集成在 Weblate 中，但現在必須寫一些代碼，將本包裹在附加組件中。

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

"""Example pre commit script."""

from django.utils.translation import gettext_lazy as _

from weblate.addons.events import EVENT_PRE_COMMIT
from weblate.addons.scripts import BaseScriptAddon


class ExamplePreAddon(BaseScriptAddon):
    # Event used to trigger the script
    events = (EVENT_PRE_COMMIT,)
    # Name of the addon, has to be unique
    name = "weblate.example.pre"
    # Verbose name and long description
    verbose = _("Execute script before commit")
    description = _("This add-on executes a script.")

    # Script to execute
    script = "/bin/true"
    # File to add in commit (for pre commit event)
    # does not have to be set
    add_file = "po/{{ language_code }}.po"
```

安裝方法請參見 *Custom quality checks, add-ons and auto-fixes*。

對於任何給定的組件，當前路徑設置版本控制系統（VCS）倉庫的根目時，執行本。

此外，可以訪問後面的環境參數：

**WL\_VCS**

使用的版本控制系統。

**WL\_REPO**

上游儲存庫的 URL。

**WL\_PATH**

版本控制系統（VCS）倉庫的對路徑。

**WL\_BRANCH**

在 2.11 版本新加入。

當前組件配置的倉庫分支。

**WL\_FILEMASK**

當前組件的 File mask。

**WL\_TEMPLATE**

單語言翻譯模板的文件名（可以空）。

**WL\_NEW\_BASE**

在 2.14 版本新加入。

建立新的翻譯所使用文件的文件名（可以空）。

**WL\_FILE\_FORMAT**

File format used in current component.

**WL\_LANGUAGE**

當前處理的翻譯的語言（對於組件級的子不可用）。

**WL\_PREVIOUS\_HEAD**

更新後的上個 HEAD (僅在運行更新後子後可用)。

**WL\_COMPONENT\_SLUG**

在 3.9 版本新加入。

組件標識串用於構建 URL。

**WL\_PROJECT\_SLUG**

在 3.9 版本新加入。

項目標識串用於構建 URL。

**WL\_COMPONENT\_NAME**

在 3.9 版本新加入。

組件名稱。

**WL\_PROJECT\_NAME**

在 3.9 版本新加入。

專案名稱。

**WL\_COMPONENT\_URL**

在 3.9 版本新加入。

組件 URL。

**WL\_ENGAGE\_URL**

在 3.9 版本新加入。

專案參與 URL。

**也參考:**

[組件配置](#)



## Post-update repository processing

當 VCS 的上游源發生變化時，可用於更新翻譯文件。為了實現這個功能，請記住 Weblate 只看到提交給版本控制系統（VCS）的文件，所以需要同意更改作爲本的一部分。

例如 Gulp，可以使用後面的代碼來執行：

```
#!/bin/sh
gulp --gulpfile gulp-i18n-extract.js
git commit -m 'Update source strings' src/languages/en.lang.json
```

## Pre-commit processing of translations

在將翻譯提交到存儲庫之前，使用 commit 本自動更改翻譯。

它作組成當前翻譯文件名的單一參數而通過。

## 2.15 翻譯記憶

在 2.20 版本新加入。

Weblate 帶有建的翻譯記憶庫，包括下面的：

- 手動導入翻譯記憶庫（請參見用界面）。
- 自動存儲 Weblate 中進行的翻譯（依賴於 *Translation memory scopes*）。
- 自動導入以前的翻譯。

翻譯記憶庫中的容可以以兩種方式之一來應用：

- 手動，自動建議 當翻譯時查看。
- Automatically, by translating strings using 自動翻譯, or 自動翻譯 add-on.

For installation tips, see *Weblate Translation Memory*, which is turned on by default.

### 2.15.1 Translation memory scopes

在 3.2 版本新加入：在較早的版本中，翻譯記憶庫只能從相應於當前導入的翻譯記憶庫範圍的文件中加載。

翻譯記憶庫的範圍這樣允許私有或翻譯者共享，來適應所需要的行。

## Imported translation memory

使用 *import\_memory* 命令導入任意翻譯記憶庫數據，使記憶的容可用於所有的用和項目。

## 每名用 的翻譯記憶庫

在每個單獨用 的個人翻譯記憶庫中自動存儲用 的翻譯。

## 每個項目的翻譯記憶庫

項目 的所有翻譯都自動存儲在項目翻譯記憶庫中，這個翻譯記憶庫只在項目 可用。

## 共享的翻譯記憶

翻譯記憶庫分享打開的項目的所有翻譯，都存儲在分享的翻譯記憶庫中，可用於所有項目。

對於分享的 Weblate 安裝，請仔細考慮是否打開這個特性，因 可能導致嚴重的影響：

- 翻譯可以被任何人使用。
- 這會導致 露秘密信息。

## 2.15.2 Managing translation memory

### 用 界面

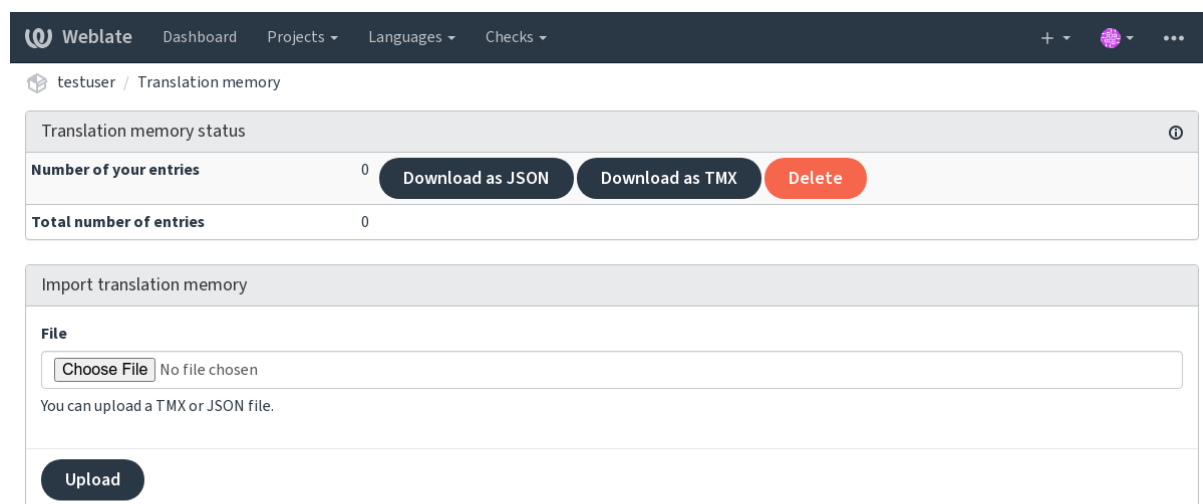
在 3.2 版本新加入。

在基本上用 界面上，可以管理每名用 每個項目的項目翻譯記憶庫。它可以用於下載、消除或導入翻譯記憶庫。

**提示：**JSON 的翻譯記憶庫可以導入 Weblate，提供了 TMX 與其他工具進行互操作。

### 也參考：

*Weblate Translation Memory Schema*



Powered by Weblate 4.15 | [About Weblate](#) | [Legal](#) | [Contact](#) | [Documentation](#) | [Donate to Weblate](#)

## 管理介面

有幾個管理命令來操作翻譯記憶庫的內容。它們整體操作翻譯記憶庫，不會被範圍來篩選（除非被參數請求）：

`dump_memory`

將記憶庫導入 JSON

`import_memory`

將 TMX 或 JSON 文件導入翻譯記憶庫

## 2.16 配置

所有的設置存儲在 `settings.py` 中，（如 Django 通常那樣）。

---

**備註：** 在更改這些設置的任何一部分後，需要重新啟動 Weblate —— WSGI 和 Celery 兩個過程。

在它作 `mod_wsgi` 運行的情況下，需要重新啟動 Apache，來重新加載配置。

---

### 也參考：

還要請查看 [Django's documentation](#) 中關於配置 Django 自身的參數。

### 2.16.1 AKISMET\_API\_KEY

Weblate 可以使用 Akismet 檢查到來的對垃圾郵件的匿名建議。請訪問 [akismet.com](https://akismet.com) 來購買 API 密鑰，將它與網站關聯。

### 2.16.2 ANONYMOUS\_USER\_NAME

未登錄用戶的用戶名。

### 也參考：

[存取控制](#)

### 2.16.3 AUDITLOG\_EXPIRY

在 3.6 版本新加入。

Weblate 應該將審計日誌保存多少天，審計日誌包括了賬戶活動的信息。

默認 180 天。

### 2.16.4 AUTH\_LOCK\_ATTEMPTS

在 2.14 版本新加入。

在 rate 限制應用前，授權嘗試失敗的最多次數。

當前，這應用在後面的位置：

- Sign in. Deletes the account password, preventing the user from signing in without requesting a new password.
- Password reset. Prevents new e-mails from being sent, avoiding spamming users with too many password reset attempts.

Defaults to 10.

也參考:

頻次限制

## 2.16.5 AUTO\_UPDATE

在 3.2 版本新加入.

在 3.11 版本變更: 更改原來的開關選項, 來區分接受哪個字符串。

以每天的頻率更新所有倉庫。

---

**提示:** 在不使用通知勾 來自動更新 Weblate 倉庫的情況下有用。

---

---

**備註:** 除了字符串選項還存在開關選項, 用於向後兼容。

---

選項有:

**"none"**

No daily updates.

**"remote" also False**

Only update remotes.

**"full" 也 True**

更新遠程, 符合工作副本。

---

**備註:** 這需要使用 *Celery* 的後台任務 工作, 在重啟後生效。

---

## 2.16.6 AVATAR\_URL\_PREFIX

構成頭像 URL 的前綴: `${AVATAR_URL_PREFIX}/avatar/${MAIL_HASH}?${PARAMS}`。已知後面的服務工作:

**Gravatar (默認), 根據 <https://gravatar.com/>**

```
AVATAR_URL_PREFIX = 'https://www.gravatar.com/'
```

**Libravatar, as per <https://www.libravatar.org/>**

```
AVATAR_URL_PREFIX = 'https://www.libravatar.org/'
```

也參考:

個人頭像快取, *ENABLE\_AVATARS*, *Avatars*

## 2.16.7 AUTH\_TOKEN\_VALID

在 2.14 版本新加入.

身份驗證令牌和密碼重置電子郵件中臨時密碼的有效時間。以秒單位, 默認 172800 (2 天)。

## 2.16.8 AUTH\_PASSWORD\_DAYS

在 2.15 版本新加入。

How many days will Weblate reject reusing previously used password for an user.

The checking is based on the audit log, `AUDITLOG_EXPIRY` needs to be at least same as this.

---

**備註：** 自動修復很大，但可能導致損壞；寫本的時候要小心。

---

默認 180 天。

## 2.16.9 AUTOFIX\_LIST

當存儲字符串時應用自動修復列表。

---

**備註：** 應用自動修復見面的 Python 類提供完全合規的路徑。

---

可用的修復：

**`weblate.trans.autofixes.whitespace.SameBookendingWhitespace`**

將字符串開始與結尾的空白字符與元字符串匹配。

**`weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis`**

替連續的點 (...), 如果源字符串有一個對應的省略號 (...).

**`weblate.trans.autofixes.chars.RemoveZeroSpace`**

去掉零寬度字符，如果源字符串不包含的話。

**`weblate.trans.autofixes.chars.RemoveControlChars`**

去掉控制字符，如果源字符串不包含的話。

**`weblate.trans.autofixes.html.BleachHTML`**

從標記 `safe-html` 的字符串中去掉不安全的 HTML 標記（請參見不安全的 *HTML*）。

可以選擇使用哪一個：

```
AUTOFIX_LIST = (
    "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
    "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
)
```

**也參考：**

自動修正, *Custom automatic fixups*

## 2.16.10 BACKGROUND\_TASKS

在 4.5.2 版本新加入。

定義應組件觸發冗長的維護任務的頻率。

Right now this controls:

- 自動翻譯 add-on
- 檢查和修復 recalculation

Possible choices:

- “每月”（這是默認值）
- weekly

- daily
- never

---

備 F: 當 Weblate 包含數千個組件時，不建議不要增加頻率。

---

### 2.16.11 BASIC\_LANGUAGES

在 4.4 版本新加入。

提供給用 F 開始新的翻譯的語言列表。當未指定時，使用 F 建列表，其中包括所有常用的語言，但 F 有特定國家/地區的變體。

這只是限制了非特權用 F 將不想要的語言添加進來。項目管理員仍然被給出 Weblate 中定義的所有語言選擇。

---

備 F: 這對 Weblate F 不定義新的語言，它只在數據庫中篩選了現有的那些。

---

示例：

```
BASIC_LANGUAGES = {"cs", "it", "ja", "en"}
```

也參考：

語言定義

### 2.16.12 BORG\_EXTRA\_ARGS

在 4.9 版本新加入。

You can pass additional arguments to **borg create** when built-in backups are triggered.

示例：

```
BORG_EXTRA_ARGS = ["--exclude", "vcs/"]
```

也參考：

備份和移動 Weblate, borg create

### 2.16.13 csp\_script\_src, csp\_img\_src, csp\_connect\_src, csp\_style\_src, csp\_font\_src

F Weblate 定制 Content-Security-Policy 標頭。根據允許集成的第三方服務（Matomo、Google Analytics、Sentry ……）來自動生成標頭。

這些默認 F 空列表。

示例：

```
# Enable Cloudflare Javascript optimizations
CSP_SCRIPT_SRC = ["ajax.cloudflare.com"]
```

也參考：

: ref: csp, ‘[F 容安全策略 \(csp\)](https://developer.mozilla.org/en-us/docs/web/http/csp)’ <<https://developer.mozilla.org/en-us/docs/web/http/csp>>‘\_

### 2.16.14 CHECK\_LIST

翻譯時執行的質量檢查列表。

備：實施檢查界面的 Python 類提供完全合規的路徑。

調整檢查列表，來包括與你相關的那些檢查。

所有建的質量檢查 默認都打開，可以從那更改設置。它們在配置的例子 中被默認釋掉，從而使用默認值。然後每個新的 Weblate 版本執行新的檢查。

可以關閉所有檢查：

```
CHECK_LIST = ()
```

可以只打開一部分檢查：

```
CHECK_LIST = (
    "weblate.checks.chars.BeginNewlineCheck",
    "weblate.checks.chars.EndNewlineCheck",
    "weblate.checks.chars.MaxLengthCheck",
)
```

備：更改這些設置只影響新更改的翻譯，現存的檢查仍然存儲在數據庫中。了將更改同樣應用到存儲的翻譯中，運行 `updatechecks`。

也參考：

質量檢查, 使用標自定義行

### 2.16.15 COMMENT\_CLEANUP\_DAYS

在 3.6 版本新加入。

在一定天數後除釋。默認 None，意味著不除。

### 2.16.16 COMMIT\_PENDING\_HOURS

在 2.10 版本新加入。

通過後台任務方式提交待定的更改之間的小時數。

也參考：

組件配置, 更動後提交的經過時間, *Running maintenance tasks*, `commit_pending`

### 2.16.17 CONTACT\_FORM

在 4.6 版本新加入。

配置如何發送來自聯人表單的電子郵件。選擇與郵件服務器配置匹配的配置。

" 回覆到 "

發件人用於：MailHeader: “回复”，這是默認行。

" 來自 "

發件人:mailheader:From, 你的电子邮箱服務器需要允許發送此類郵件。

### 2.16.18 DATA\_DIR

Weblate 文件夾將所有數據存儲其中。它包含到版本控制系統（VCS）倉庫的鏈接，全文索引和外部工具的各种文件。

後面的子目通常存在：

**home**

Home 目用於調用本。

**ssh**

SSH 密鑰和配置。

**static**

態 Django 文件的默認位置，由 `django:STATIC_ROOT` 指定。見：[ref:static-files](#)。

Docker 容器這個使用了一個單獨的，請參 [Docker container volumes](#)。

**media**

Django 媒體文件的默認位置，由 `django:MEDIA_ROOT` 指定。包含已上傳的截屏，見：[ref:screenshots](#)。

**vcs**

Version control repositories for translations.

**backups**

每日備份數據，細節請參考下載的數據用於備份。

**celery**

Celery 調度程序數據，見使用 *Celery* 的後台任務。

**fonts:**

User-uploaded fonts, see [管理字型](#)。

---

**備：** 這個目必須由 Weblate 寫入。運行作 `uWSGI` 意味著 `www-data` 用應該對它具有寫入權限。實現這個的最簡單方式是使用作目的的所有者：

```
sudo chown www-data:www-data -R $DATA_DIR
```

---

Defaults to `/home/weblate/data`, but it is expected to be configured.

**也參考：**

[文件系統權限, 備份和移動 Weblate](#)

### 2.16.19 DATABASE\_BACKUP

在 3.1 版本新加入。

數據庫備份應該存儲純文本，壓縮還是跳過。授權值：

- "plain"
- "compressed"
- "none"

**也參考：**

[備份和移動 Weblate](#)




### 2.16.20 DEFAULT\_ACCESS\_CONTROL

在 3.3 版本新加入。

新項目的默認訪問控制設置：

0	<i>Public</i>
1	<i>Protected</i>
100	<i>Private</i>
200	<i>Custom</i>



如果手動管理 ACL 則使用 *Custom*，這意味著不依賴於 Weblate  部管理。

**也參考：**

[專案存取控制](#), [存取控制](#)

### 2.16.21 DEFAULT\_AUTO\_WATCH

在 4.5 版本新加入。

配置是否：Guilabel：“自動觀看貢獻的項目，應該打開新用 。默認  “true”。

**也參考：**

[通知](#)

### 2.16.22 DEFAULT\_RESTRICTED\_COMPONENT


在 4.1 版本新加入。

組件限制的默認值。

**也參考：**

[受限制的訪問](#), [Scope of teams](#)

### 2.16.23 default\_add\_message, default\_addon\_message, default\_commit\_message, default\_delete\_message, default\_merge\_message

不同操作的默認執行信息，細節請查  組件配置。

**也參考：**

[模板標記](#), [組件配置](#), [Commit](#), [add](#), [delete](#), [merge](#), [add-on](#), and [merge request messages](#)

## 2.16.24 DEFAULT\_ADDONS

Default add-ons to install on every created component.

---

備 F: 此設置只影響新創建的組件。

---

例:

```
DEFAULT_ADDONS = {
    # Add-on with no parameters
    "weblate.flags.target_edit": {},
    # Add-on with parameters
    "weblate.autotranslate.autotranslate": {
        "mode": "suggest",
        "filter_type": "todo",
        "auto_source": "mt",
        "component": "",
        "engines": ["weblate-translation-memory"],
        "threshold": "80",
    },
}
```

也參考:

*install\_addon*, 附加元件, *WEBLATE\_ADDONS*

## 2.16.25 DEFAULT\_COMMITER\_EMAIL

在 2.4 版本新加入.

提交者電子郵件地址默認 F noreply@weblate.org。

也參考:

*DEFAULT\_COMMITER\_NAME*

## 2.16.26 DEFAULT\_COMMITER\_NAME

在 2.4 版本新加入.

提交者姓名默認 F Weblate。

也參考:

*DEFAULT\_COMMITER\_EMAIL*

## 2.16.27 DEFAULT\_LANGUAGE

在 4.3.2 版本新加入.

例如在:ref:‘component-source\_language’中使用的默認源語言。

默認 F en。匹配的語言對象需要存在於數據庫中。

也參考:

語言定義, 來源語言

### 2.16.28 DEFAULT\_MERGE\_STYLE

在 3.4 版本新加入。

任何新組件的合 風格。

- *rebase* - default
- *merge*

**也參考：**

組件配置, 合 類型

### 2.16.29 DEFAULT\_SHARED\_TM

在 3.2 版本新加入。

配置默認值： `ref: project-use_shared_tm` 和 `ref: project-contribute_shared_tm`。

### 2.16.30 DEFAULT\_TRANSLATION\_PROPAGATION

在 2.5 版本新加入。

翻譯傳播的默認設置，默認 `True`。

**也參考：**

組件配置, 允許翻譯再用

### 2.16.31 DEFAULT\_PULL\_MESSAGE

Configures the default title and message for pull requests.

### 2.16.32 ENABLE\_AVATARS

是否 用 打開基於 Gravatar 的頭像。默認打開。

頭像取出 緩存在從服務器中，降低了 漏個人信息的風險，加速了用 體驗。

**也參考：**

個人頭像快取, `AVATAR_URL_PREFIX`, *Avatars*

### 2.16.33 ENABLE\_HOOKS

是否允許匿名遠程 子。

**也參考：**

通知 勾

### 2.16.34 ENABLE\_HTTPS

將鏈接作 HTTPS 還是 HTTP 發送給 Weblate。這個設置影響發送電子郵件，生對 URL。

在默認配置中，這還用於與 HTTPS 相關的幾個 Django 設置—用安全 cookie、切 HSTS 或允許重定向到 HTTPS URL。

在一些情況下 HTTPS 重定向會有問題，您可以在使用反向代理進行 SSL 終端連接的情況下碰到無限重定向的問題，使用反向代理進行 SSL 終端連接不能將協議標頭正確地傳遞給 Django。請調整自己的反向代理配置，去掉 X-Forwarded-Proto 或 Forwarded 標頭，或者配置 `SECURE_PROXY_SSL_HEADER`，讓 Django 正確地檢測 SSL 狀態。

**也參考：**

：設置： `django: session_cookie_secure`，：設置： `django: csrf_cookie_secure`，：設置： `django: secure_sl_redirect`

### 2.16.35 ENABLE\_SHARING

打開/關閉:guilabel: *Share* 菜單，使用能將翻譯過程分享到社交網絡上。

### 2.16.36 EXTRA\_HTML\_HEAD

在 4.15 版本新加入。

Insert additional markup into HTML header. Can be used for verification of site ownership, for example:

```
EXTRA_HTML_HEAD = '<link href="https://fosstodon.org/@weblate" rel="me">'
```

**警告：** No sanitization is performed on the string, it is inserted as is into the HTML header.

### 2.16.37 GET\_HELP\_URL

在 4.5.2 版本新加入。

可以找到你的 Weblate 實例支持的 URL。

### 2.16.38 GITEA\_CREDENTIALS

在 4.12 版本新加入。

List for credentials for Gitea servers.

```
GITEA_CREDENTIALS = {
    "try.gitea.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitea.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

**也參考：**

`WEBLATE_GITEA_USERNAME`, `WEBLATE_GITEA_TOKEN`, `WEBLATE_GITEA_HOST`

### 2.16.39 GITEA\_USERNAME

在 4.14.2 版本開始<sup>[1]</sup>用: This configuration is insecure, it is recommended to switch to *GITEA\_CREDENTIALS*.  
在 4.12 版本新加入.

Gitea username used to send pull requests for translation updates.

**也參考:**

*GITEA\_CREDENTIALS*, *Gitea pull requests*

### 2.16.40 GITEA\_TOKEN

在 4.14.2 版本開始<sup>[1]</sup>用: This configuration is insecure, it is recommended to switch to *GITEA\_CREDENTIALS*.  
在 4.12 版本新加入.

Gitea personal access token used to make API calls to send pull requests for translation updates.

**也參考:**

*GITEA\_CREDENTIALS*, *Gitea pull requests*, *Creating a Gitea personal access token*

### 2.16.41 GITLAB\_CREDENTIALS

在 4.3 版本新加入.

用於 GitLab 服務器的證明列表。

```
GITLAB_CREDENTIALS = {
    "gitlab.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "gitlab.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

**也參考:**

*WEBLATE\_GITLAB\_USERNAME*, *WEBLATE\_GITLAB\_TOKEN*, *WEBLATE\_GITLAB\_HOST*

### 2.16.42 GITLAB\_USERNAME

在 4.14.2 版本開始<sup>[1]</sup>用: This configuration is insecure, it is recommended to switch to *GITLAB\_CREDENTIALS*.  
GitLab 用<sup>[1]</sup>名，用於發送翻譯更新的和<sup>[1]</sup>請求。

**也參考:**

*GITLAB\_CREDENTIALS*, *GitLab merge requests*

### 2.16.43 GITLAB\_TOKEN

在 4.14.2 版本開始 F 用: This configuration is insecure, it is recommended to switch to *GITLAB\_CREDENTIALS*.  
在 4.3 版本新加入.

GitLab personal access token used to make API calls to send merge requests for translation updates.

也參考:

: 設置: *gitlab\_credentials*, : ref: *vcs-gitlab*, 'gitlab: 個人訪問令牌  
<[https://docs.gitlab.com/ee/user/profile/personal\\_access\\_tokens.html](https://docs.gitlab.com/ee/user/profile/personal_access_tokens.html)>‘\_

### 2.16.44 GITHUB\_CREDENTIALS

在 4.3 版本新加入.

用於 GitHub 服務器的證明列表。

```
GITHUB_CREDENTIALS = {
    "api.github.com": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "github.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

也參考:

*WEBLATE\_GITHUB\_USERNAME*, *WEBLATE\_GITHUB\_TOKEN*, *WEBLATE\_GITHUB\_HOST*

### 2.16.45 GITHUB\_USERNAME

在 4.14.2 版本開始 F 用: This configuration is insecure, it is recommended to switch to *GITHUB\_CREDENTIALS*.

GitHub 用 F 名，用於發送翻譯更新的拉取請求。

也參考:

*GITHUB\_CREDENTIALS*, *GitHub pull requests*

### 2.16.46 GITHUB\_TOKEN

在 4.14.2 版本開始 F 用: This configuration is insecure, it is recommended to switch to *GITHUB\_CREDENTIALS*.

在 4.3 版本新加入.

GitHub 個人訪問令牌，用於進行 API 調用，來發送翻譯更新的拉取請求。

也參考:

*GITHUB\_CREDENTIALS*, *GitHub pull requests*, *Creating a GitHub personal access token*

### 2.16.47 GOOGLE\_ANALYTICS\_ID

谷歌分析 ID，開 使用谷歌分析對 Weblate 的監控。

### 2.16.48 HIDE\_REPO\_CREDENTIALS

隱藏倉庫憑據避免出現在 web 界面中。在倉庫 URL 帶有用 名和密碼的情況下，Weblate 會在相關信息顯示給用 時將其隱藏起來。

例如除了 `https://user:password@git.example.com/repo.git` 會只顯示 `https://git.example.com/repo.git`。它也試圖以相似的方式清除版本控制系統（VCS）錯誤信息。

---

備：默認這是打開的。

---

### 2.16.49 HIDE\_VERSION

在 4.3.1 版本新加入。

從未進行身份驗證的用 那 的隱藏版本信息。這樣將所有文檔的連接點連接到最新的版本，而不是與當前安裝的版本匹配文檔的版本。

一些公司推薦在安全實踐上推薦隱藏版本，但不能防止攻擊者通過試探性 來找出版本。

---

備：This is turned off by default.

---

### 2.16.50 INTERLEDGER\_PAYMENT\_POINTERS

在 4.12.1 版本新加入。

List of Interledger Payment Pointers (ILPs) for Web Monetization.

If multiple are specified, probabilistic revenue sharing is achieved by selecting one randomly.

Please check <<https://webmonetization.org/>> for more details.

---

提示：The default value lets users fund Weblate itself.

---

### 2.16.51 IP\_BEHIND\_REVERSE\_PROXY

在 2.14 版本新加入。

指示 Weblate 是否在反向代理後面運行。

如果設置 True，Weblate 會從:setting:'IP\_PROXY\_HEADER'定義的標頭中得到 IP 地址。

**警告：**確保真正使用反向代理， 且設置了這個標頭，否則用 將能 假冒 IP 地址。

---

備：默認這不是打開的。

---

**也參考:**

在反向代理後面運行, 頻次限制, `IP_PROXY_HEADER`, `IP_PROXY_OFFSET`

## 2.16.52 IP\_PROXY\_HEADER

在 2.14 版本新加入.

指示當 `IP_BEHIND_REVERSE_PROXY` 打開時, Weblate 應該從那個標頭得到 IP 地址。

預設 `HTTP_X_FORWARDED_FOR`。

**也參考:**

: 參考: “反向代理”, : 參考: “速率限制”, : 設置: “Django: secure\_proxy\_sl\_header”, : 設置: `ip_behind_reverse_proxy`, : 設置: `Ip_proxy_offset`

## 2.16.53 IP\_PROXY\_OFFSET

在 2.14 版本新加入.

指示 `IP_PROXY_HEADER` 的哪部分用作客端 IP 地址。

依賴於你的設置, 這個標頭會包括幾個 IP 地址, (例如 “X-Forwarded-For: a, b, client-ip”), 且可以配置標頭的哪個地址在這用作客端 IP 地址。

**警告:** 設置這個會影響你安裝的安全性, 應該只配置它來使用信任的代理來確定 IP 地址。

Defaults to 0.

**也參考:**

: 參考: “反向代理”, : REF: “速率限制”, : 設置: “Django: secure\_proxy\_ssl\_header”, : 設置: `ip_behind_reverse_proxy`, : 設置: “`ip_proxy_header`”

## 2.16.54 LEGAL\_TOS\_DATE

在 4.15 版本新加入.

---

備: You need 法律 installed to make this work.

---

Date of last update of terms of service documents. Whenever the date changes, users are required to agree with the terms of service.

```
from datetime import date

LEGAL_TOS_DATE = date(2022, 2, 2)
```



### 2.16.55 LEGAL\_URL

在 3.5 版本新加入。

你的 Weblate 事例顯示其法律文件的 URL。

---

**提示：** 在將你的法律文件保存在 Weblate 之外，而將其嵌入 Weblate 的情 F 下有用。細節請查看法律。

---

例：

```
LEGAL_URL = "https://weblate.org/terms/"
```

**也參考：**

*PRIVACY\_URL*

### 2.16.56 LICENSE\_EXTRA

包括在許可選擇中的其他許可。

---

**備 F：** 每個許可的定義應該是其短名稱、長名稱和 URL 的元組。

---

例如：

```
LICENSE_EXTRA = [
    (
        "AGPL-3.0",
        "GNU Affero General Public License v3.0",
        "https://www.gnu.org/licenses/agpl-3.0-standalone.html",
    ),
]
```

### 2.16.57 LICENSE\_FILTER

在 4.3 版本變更：將其設置 F 空值現在關閉了許可提醒。

要顯示的被篩選的許可列表。在設置 F 空時還關閉了許可提醒。

---

**備 F：** 這個過濾器使用了短許可名稱。

---

例如：

```
LICENSE_FILTER = {"AGPL-3.0", "GPL-3.0-or-later"}
```

後面關閉了許可提醒：

```
LICENSE_FILTER = set ()
```

**也參考：**

alerts

## 2.16.58 LICENSE\_REQUIRED

定義了是否需要組件配置 中的許可屬性。

---

備：默認這是關閉的。

---

## 2.16.59 LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH

Whether the length of a given translation should be limited. The restriction is the length of the source string × 10 characters.

---

提示： Set this to `False` to allow longer translations (up to 10,000 characters) irrespective of source string length.

---

---

備：預設 `True`.

---

## 2.16.60 LOCALIZE\_CDN\_URL 和 LOCALIZE\_CDN\_PATH

These settings configure the *JavaScript 在地化 CDN* add-on. `LOCALIZE_CDN_URL` defines root URL where the localization CDN is available and `LOCALIZE_CDN_PATH` defines path where Weblate should store generated files which will be served at the `LOCALIZE_CDN_URL`.

---

提示： 在 hosted Weblate 中，這使用 `https://weblate-cdn.com/`。

---

也參考：

*JavaScript 在地化 CDN*

## 2.16.61 LOGIN\_REQUIRED\_URLS

A list of URLs you want to require signing in. (Besides the standard rules built into Weblate).

---

提示： 這允許密碼保護整個安裝，通過使用：

```
LOGIN_REQUIRED_URLS = (r"/(.*)$",)
REST_FRAMEWORK["DEFAULT_PERMISSION_CLASSES"] = [
    "rest_framework.permissions.IsAuthenticated"
]
```

---

---

提示： 同樣想要鎖住 API 訪問，如上面的例子所示。

---

也參考：

`REQUIRE_LOGIN`

### 2.16.62 LOGIN\_REQUIRED\_URLS\_EXCEPTIONS

用於 `LOGIN_REQUIRED_URLS` 的例外列表。如果未指定，則允許用 訪問登 頁。

你可能想要包括的一些例外：

```
LOGIN_REQUIRED_URLS_EXCEPTIONS = (
    r"/accounts/(.*)$", # Required for sign in
    r"/static/(.*)$", # Required for development mode
    r"/widgets/(.*)$", # Allowing public access to widgets
    r"/data/(.*)$", # Allowing public access to data exports
    r"/hooks/(.*)$", # Allowing public access to notification hooks
    r"/api/(.*)$", # Allowing access to API
    r"/js/i18n/$", # JavaScript localization
)
```

### 2.16.63 MATOMO\_SITE\_ID

你想要跟踪的 Matomo（以前的 Piwik）中的网站 ID。

---

**備：** 這個集成不支持 Matomo Tag Manager。

---

**也參考：**

`MATOMO_URL`

### 2.16.64 MATOMO\_URL

你想要跟踪的 Matomo（以前的 Piwik）安裝的全 URL（包括反斜杠）。更多細節請查 <<https://matomo.org/>>。

---

**提示：** 這個集成不支持 Matomo Tag Manager。

---

例如：

```
MATOMO_SITE_ID = 1
MATOMO_URL = "https://example.matomo.cloud/"
```

**也參考：**

`MATOMO_SITE_ID`

### 2.16.65 NEARBY\_MESSAGES

在查看當前翻譯字符串時顯示多少字符串。這只是默認值，用 可以在 `:ref:~user-profile~` 中調整。

### 2.16.66 DEFAULT\_PAGE\_LIMIT

在 4.7 版本新加入.

Default number of elements to display when pagination is active.

### 2.16.67 PAGURE\_CREDENTIALS

在 4.3.2 版本新加入.

Pagure 服務器憑據列表。

```
PAGURE_CREDENTIALS = {
    "pagure.io": {
        "username": "weblate",
        "token": "your-api-token",
    },
    "pagure.example.com": {
        "username": "weblate",
        "token": "another-api-token",
    },
}
```

**也參考:**

*WEBLATE\_PAGURE\_USERNAME, WEBLATE\_PAGURE\_TOKEN, WEBLATE\_PAGURE\_HOST*

### 2.16.68 PAGURE\_USERNAME

在 4.14.2 版本開始<sup>F</sup>用: This configuration is insecure, it is recommended to switch to *PAGURE\_CREDENTIALS*.

在 4.3.2 版本新加入.

Pagure username used to send merge requests for translation updates.

**也參考:**

*PAGURE\_CREDENTIALS, Pagure merge requests*

### 2.16.69 PAGURE\_TOKEN

在 4.14.2 版本開始<sup>F</sup>用: This configuration is insecure, it is recommended to switch to *PAGURE\_CREDENTIALS*.

在 4.3.2 版本新加入.

Pagure personal access token used to make API calls to send merge requests for translation updates.

**也參考:**

: 設置: *pagure\_credentials*, : ref: *vcs-pagure*, *pagure api*

### 2.16.70 PRIVACY\_URL

在 4.8.1 版本新加入.

URL where your Weblate instance shows its privacy policy.

---

**提示:** 在將你的法律文件保存在 Weblate 之外，而將其嵌入 Weblate 的情<sup>F</sup>下有用。細節請查看法律。

---

例:

```
PRIVACY_URL = "https://weblate.org/terms/"
```

**也參考:**

*LEGAL\_URL*

### 2.16.71 PRIVATE\_COMMIT\_EMAIL\_OPT\_IN

在 4.15 版本新加入.

Configures whether the private commit e-mail is opt-in or opt-out (by default it is opt-out).

**也參考:**

個人檔案, *PRIVATE\_COMMIT\_EMAIL\_TEMPLATE*

### 2.16.72 PRIVATE\_COMMIT\_EMAIL\_TEMPLATE

在 4.15 版本新加入.

Template to generate private commit e-mail for an user. Defaults to "{username}@users.noreply.{site\_domain}".

Set to blank string to disable.

---

**備 <sup>F</sup>:** Using different commit e-mail is opt-in for users unless configured by *PRIVATE\_COMMIT\_EMAIL\_OPT\_IN*. Users can configure commit e-mail in the 個人檔案.

---

### 2.16.73 PROJECT\_BACKUP\_KEEP\_COUNT

在 4.14 版本新加入.

Defines how many backups per project are kept on the server. It defaults to 3.

**也參考:**

*Project level backups*

### 2.16.74 PROJECT\_BACKUP\_KEEP\_DAYS

在 4.14 版本新加入。

Defines how long the project backups will be kept on the server. Defaults to 30 days.

**也參考:**

*Project level backups*

### 2.16.75 PROJECT\_NAME\_RESTRICT\_RE

在 4.15 版本新加入。

Defines a regular expression to restrict project naming. Any matching names will be rejected.

**也參考:**

專案名懲

### 2.16.76 PROJECT\_WEB\_RESTRICT\_RE

在 4.15 版本新加入。

Defines a regular expression to restrict project websites. Any matching URLs will be rejected.

**也參考:**

專案網站

### 2.16.77 RATELIMIT\_ATTEMPTS

在 3.2 版本新加入。

在應用次數限制前，身份認證嘗試的最多次數。

Defaults to 5.

**也參考:**

頻次限制, *RATELIMIT\_WINDOW*, *RATELIMIT\_LOCKOUT*

### 2.16.78 RATELIMIT\_WINDOW

在 3.2 版本新加入。

應用次數限制後，可接受多少次身份認證。

秒數默認 F 300 (5 分鐘)。

**也參考:**

頻次限制, *RATELIMIT\_ATTEMPTS*, *RATELIMIT\_LOCKOUT*

### 2.16.79 RATELIMIT\_LOCKOUT

在 3.2 版本新加入。

在應用次數限制後，身份認證鎖定多久。

秒數默認 600（10 分鐘）。

**也參考：**

頻次限制, `RATELIMIT_ATTEMPTS`, `RATELIMIT_WINDOW`

### 2.16.80 REGISTRATION\_ALLOW\_BACKENDS

在 4.1 版本新加入。

身份驗證後端的列表，允許從中選擇。這只限制新的，用可以使用配置的身份驗證後端，來進行身份驗證和添加身份驗證。

當限制後端時，推薦保持 `REGISTRATION_OPEN` 開狀態，否則將能，但 Weblate 不會在用界面顯示的鏈接。

例：

```
REGISTRATION_ALLOW_BACKENDS = ["azuread-oauth2", "azuread-tenant-oauth2"]
```

**提示：** 與身份驗證 URL 中使用的名稱匹配的後端名稱。

**也參考：**

`REGISTRATION_OPEN`, 身份核對

### 2.16.81 REGISTRATION\_CAPTCHA

True 或 False 的值指示新賬戶是否被 CAPTCHA 保護。這個設置是可選的，默認的 True 是假定不提供。

如果打開，CAPTCHA 會添加到用輸入其電子郵箱地址的所有頁面中：

- New account registration.
- 找回密碼。
- 將電子郵箱地址添加到賬戶中。
- 供未登用使用的聯表格。

### 2.16.82 REGISTRATION\_EMAIL\_MATCH

在 2.17 版本新加入。

允許你篩選哪個電子郵箱地址可以。

默認 `.*`，允許使用任何電子郵箱地址。

可以用它限制到單一的電子郵箱域名：

```
REGISTRATION_EMAIL_MATCH = r"^.*@weblate\.org$"
```

### 2.16.83 REGISTRATION\_OPEN

是否新賬在當前是允許的。這個可選設置可以保持默認 `True`，或更改 `False`。

這個設置影響了建的通過電子郵件地址或通過 `Python Social Auth` 的身份驗證（可以使用 `REGISTRATION_ALLOW_BACKENDS` 適當的後端建立白名單）。

---

備：如果使用第三方身份驗證方法，如 `LDAP` 身份驗證，只是隱藏表格，而新用仍然能登建立賬。

---

#### 也參考：

`REGISTRATION_ALLOW_BACKENDS`, `REGISTRATION_EMAIL_MATCH`, 身份核對

### 2.16.84 REPOSITORY\_ALERT\_THRESHOLD

在 4.0.2 版本新加入。

觸發倉庫過期警告的 值，或者包含了太多更改的 值。默認 25。

#### 也參考：

alerts

### 2.16.85 REQUIRE\_LOGIN

在 4.1 版本新加入。

這會用 `LOGIN_REQUIRED_URLS` 配置 REST 框架來對所有 API 端點都要求登。

---

備：這實現在 `sample-configuration` 中。對於 Docker，使用 `envvar:WEBLATE_REQUIRE_LOGIN`。

---

### 2.16.86 SENTRY\_DSN

在 3.9 版本新加入。

Sentry DSN 用於 *Collecting error reports*。

#### 也參考：

Sentry 的 Django 集成

### 2.16.87 SESSION\_COOKIE\_AGE\_AUTHENTICATED

在 4.3 版本新加入。

對身份驗證的用設置會話過期。這補充了用於有身份驗證用的 `SESSION_COOKIE_AGE`。

#### 也參考：

`SESSION_COOKIE_AGE`



## 2.16.88 SIMPLIFY\_LANGUAGES

對於默認語言/國家組合使用簡單語言代碼。例如，`fr_FR` 翻譯將使用 `fr` 語言代碼。這通常是受歡迎的特性，因它簡化了這些默認組合列出的語言。

如果對每種不同的變化想要不同的翻譯，那請將其關閉。

## 2.16.89 SITE\_DOMAIN

配置網站域名。這在很多領域產生正確的對鏈接是必要的（例如激活電子郵件、通知或 RSS 推送）。

在 Weblate 運行在非標準端口時，這同樣要包括它。

範例：

```
# Production site with domain name
SITE_DOMAIN = "weblate.example.com"

# Local development with IP address and port
SITE_DOMAIN = "127.0.0.1:8000"
```

備：這個設置應該只包含域名。對於配置協議，（允許或制 HTTPS）使用 `ENABLE_HTTPS` and for changing URL, use `URL_PREFIX`。

提示：關於 Docker 容器，網站域名通過 `WEBLATE_ALLOWED_HOSTS` 來配置。

也參考：

：參考：“生現場”，：參考：“生 - 主機”，：REF：“生-SSL”：envvar:” WebLate\_Site\_Domain “，：設置:” enable\_https“

## 2.16.90 SITE\_TITLE

用於網站和發送電子郵件的網站名稱。

## 2.16.91 SPECIAL\_CHARS

屏幕鍵盤中包括的另外的字符，*Visual keyboard*。

默認值：

```
SPECIAL_CHARS = ("\\t", "\\n", "\\u00a0", " ... ")
```

## 2.16.92 SINGLE\_PROJECT

在 3.8 版本新加入。

將用直接重定向到項目或組件，而不是顯示控制面板。可以將其設置 `True`，在這種情況下，只在 Weblate 實際只有單一的項目時有用。另外可以設置項目標識串，將無條件地重定向到這個項目。

在 3.11 版本變更：設置只接受項目標識串，來制顯示那個單一項目。

例：

```
SINGLE_PROJECT = "test"
```

## 2.16.93 SSH\_EXTRA\_ARGS

在 4.9 版本新加入。

Allows to add custom parameters when Weblate is invoking SSH. This is useful when connecting to servers using legacy encryption or other non-standard features.

For example when SSH connection in Weblate fails with *Unable to negotiate with legacyhost: no matching key exchange method found. Their offer: diffie-hellman-group1-sha1*, you can enable that using:

```
SSH_EXTRA_ARGS = "-oKexAlgorithms=+diffie-hellman-group1-sha1"
```

---

**提示：** The string is evaluated by shell, so make sure to quote any whitespace and special characters.

---

**也參考：**

[OpenSSH Legacy Options](#)

## 2.16.94 STATUS\_URL

Weblate 事例報告其狀態的 URL。

## 2.16.95 SUGGESTION\_CLEANUP\_DAYS

在 3.2.1 版本新加入。

指示當 `IP_BEHIND_REVERSE_PROXY` 打開時，Weblate 應該從那個標頭得到 IP 地址。

## 2.16.96 UPDATE\_LANGUAGES

在 4.3.2 版本新加入。

控制在運行數據庫遷移時是否應該更新語言數據庫在默認情況下。這個設置對 `djadmin: setuplang` 的調用有影響。

**警告：** The languages display might become inconsistent with this. Weblate language definitions extend over time and it will not display language code for the defined languages.

**也參考：**

[Built-in language definitions](#)

## 2.16.97 URL\_PREFIX

這個設置允許在一些路徑下運行 Weblate（否則它依賴於從 web 服務器根目運行）。

---

**備註：** 了使用這個設置，還需要配置服務器來去掉這個前綴。例如 WSGI，可以通過設置 `WSGIScriptAlias` 來實現。

---

---

**提示：** 前綴應該以 / 開始。

---

例：

```
URL_PREFIX = "/translations"
```

**備註：** 這個設置在 Django's 建服務器中不起作用，必須調整 `urls.py` 來包含這個前綴。

## 2.16.98 VCS\_API\_DELAY

在 4.15.1 版本新加入。

Configures minimal delay in seconds between third-party API calls in *GitHub pull requests*, *GitLab merge requests*, *Gitea pull requests*, and *Pagure merge requests*.

This rate-limits API calls from Weblate to these services to avoid overloading them.

If you are being limited by secondary rate limiter at GitHub, increasing this might help.

The default value is 10.

## 2.16.99 VCS\_BACKENDS

可用的版本控制系統（VCS）後端的配置。

**備註：** Weblate 嘗試使用你所有工具支持的後端。

**提示：** 可以使用這個來限制選擇或添加定版本控制系統（VCS）後端。

```
VCS_BACKENDS = ("weblate.vcs.git.GitRepository",)
```

**也參考：**

版本控制整合

## 2.16.100 VCS\_CLONE\_DEPTH

在 3.10.2 版本新加入。

配置 Weblate 應該對倉庫進行深度多少的克隆。

**備註：** 當前這只在 *Git* 中支持。默認 Weblate 進行倉庫的淺克隆，使克隆更快節省磁盤空間。根據應用（例如當使用定制的附加元件時），你可能想要增加深度，或通過設置 0 來完全關閉淺克隆。

**提示：** 在從 Weblate 推送而得到 `fatal: protocol error: expected old/new/ref, got 'shallow <commit hash>'` 錯誤的情況下，通過設置來完全關閉淺克隆：

```
VCS_CLONE_DEPTH = 0
```

### 2.16.101 WEBLATE\_ADDONS

List of add-ons available for use. To use them, they have to be enabled for a given translation component. By default this includes all built-in add-ons, when extending the list you will probably want to keep existing ones enabled, for example:

```
WEBLATE_ADDONS = (  
    # Built-in add-ons  
    "weblate.addons.gettext.GenerateMoAddon",  
    "weblate.addons.gettext.UpdateLinguasAddon",  
    "weblate.addons.gettext.UpdateConfigureAddon",  
    "weblate.addons.gettext.MsgmergeAddon",  
    "weblate.addons.gettext.GettextCustomizeAddon",  
    "weblate.addons.gettext.GettextAuthorComments",  
    "weblate.addons.cleanup.CleanupAddon",  
    "weblate.addons.consistency.LangugeConsistencyAddon",  
    "weblate.addons.discovery.DiscoveryAddon",  
    "weblate.addons.flags.SourceEditAddon",  
    "weblate.addons.flags.TargetEditAddon",  
    "weblate.addons.flags.SameEditAddon",  
    "weblate.addons.flags.BulkEditAddon",  
    "weblate.addons.generate.GenerateFileAddon",  
    "weblate.addons.json.JSONCustomizeAddon",  
    "weblate.addons.xml.XMLCustomizeAddon",  
    "weblate.addons.properties.PropertiesSortAddon",  
    "weblate.addons.git.GitSquashAddon",  
    "weblate.addons.removal.RemoveComments",  
    "weblate.addons.removal.RemoveSuggestions",  
    "weblate.addons.resx.ResxUpdateAddon",  
    "weblate.addons.autotranslate.AutoTranslateAddon",  
    "weblate.addons.yaml.YAMLCustomizeAddon",  
    "weblate.addons.cdn.CDNJSAddon",  
    # Add-on you want to include  
    "weblate.addons.example.ExampleAddon",  
)
```

---

**備註：** Removing the add-on from the list does not uninstall it from the components. Weblate will crash in that case. Please uninstall add-on from all components prior to removing it from this list.

---

#### 也參考：

附加元件, `DEFAULT_ADDONS`

### 2.16.102 WEBLATE\_EXPORTERS

在 4.2 版本新加入。

可用的導出程序列表，提供下載各種文件格式的翻譯或詞表。

#### 也參考：

支持的文件格式

### 2.16.103 WEBLATE\_FORMATS

在 3.0 版本新加入。

可供使用的文件格式列表。

---

備註：默認列表已經具有了常見格式。

---

**也參考：**

[支持的文件格式](#)

### 2.16.104 WEBLATE\_MACHINERY

在 4.13 版本新加入。

List of machinery services available for use.

**也參考：**

[配置自動建議](#)

### 2.16.105 WEBLATE\_GPG\_IDENTITY

在 3.1 版本新加入。

Weblate 使用的身份，用於登 Git Commits，例如：

```
WEBLATE_GPG_IDENTITY = "Weblate <weblate@example.com>"
```

搜索 Weblate GPG 鑰匙鏈（home/.gnupg，在 *DATA\_DIR* 之下）。如果沒有找到，則生成密鑰，更多細節請查詢簽署 *GnuPG* 的 *Git* 承諾。

**也參考：**

[簽署 \*GnuPG\* 的 \*Git\* 承諾](#)

### 2.16.106 WEBSITE\_REQUIRED

定義是否：REF：創建項目時必須指定 *Project-Web*。默認情況下打開，因為這適合公共服務器設置。

## 2.17 配置的例子

後面的例子作 `weblate/settings_example.py` 與 Weblate 一起上市：

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

import os
import platform
from logging.handlers import SysLogHandler

# Title of site to use
SITE_TITLE = "Weblate"

# Site domain
```

(繼續下一頁)

(繼續上一頁)

```

SITE_DOMAIN = ""

# Whether site uses https
ENABLE_HTTPS = False

#
# Django settings for Weblate project.
#

DEBUG = True

ADMINS = (
    # ("Your Name", "your_email@example.com"),
)

MANAGERS = ADMINS

DATABASES = {
    "default": {
        # Use "postgresql" or "mysql".
        "ENGINE": "django.db.backends.postgresql",
        # Database name.
        "NAME": "weblate",
        # Database user.
        "USER": "weblate",
        # Name of role to alter to set parameters in PostgreSQL,
        # use in case role name is different than user used for authentication.
        # "ALTER_ROLE": "weblate",
        # Database password.
        "PASSWORD": "",
        # Set to empty string for localhost.
        "HOST": "127.0.0.1",
        # Set to empty string for default.
        "PORT": "",
        # Customizations for databases.
        "OPTIONS": {
            # In case of using an older MySQL server,
            # which has MyISAM as a default storage
            # "init_command": "SET storage_engine=INNODB",
            # Uncomment for MySQL older than 5.7:
            # "init_command": "SET sql_mode='STRICT_TRANS_TABLES'",
            # Set emoji capable charset for MySQL:
            # "charset": "utf8mb4",
            # Change connection timeout in case you get MySQL gone away error:
            # "connect_timeout": 28800,
        },
        # Persistent connections
        "CONN_MAX_AGE": 0,
        # Disable server-side cursors, might be needed with pgbouncer
        "DISABLE_SERVER_SIDE_CURSORS": False,
    }
}

# Data directory, you can use following for the development purposes:
# os.path.join(os.path.dirname(os.path.dirname(os.path.abspath(__file__))), "data")
DATA_DIR = "/home/weblate/data"

# Local time zone for this installation. Choices can be found here:
# http://en.wikipedia.org/wiki/List_of_tz_zones_by_name
# although not all choices may be available on all operating systems.
# In a Windows environment this must be set to your system time zone.

```

(繼續下一頁)

(繼續上一頁)

```

TIME_ZONE = "UTC"

# Language code for this installation. All choices can be found here:
# http://www.i18nguy.com/unicode/language-identifiers.html
LANGUAGE_CODE = "en-us"

LANGUAGES = (
    ("ar", "العربية"),
    ("az", "Azərbaycan"),
    ("be", "Беларуская"),
    ("be@latin", "Biełaruskaja"),
    ("bg", "Български"),
    ("br", "Brezhoneg"),
    ("ca", "Català"),
    ("cs", "Čeština"),
    ("da", "Dansk"),
    ("de", "Deutsch"),
    ("en", "English"),
    ("el", "Ελληνικά"),
    ("en-gb", "English (United Kingdom)"),
    ("es", "Español"),
    ("fi", "Suomi"),
    ("fr", "Français"),
    ("gl", "Galego"),
    ("he", "עברית"),
    ("hu", "Magyar"),
    ("hr", "Hrvatski"),
    ("id", "Indonesia"),
    ("is", "Íslenska"),
    ("it", "Italiano"),
    ("ja", "日本語"),
    ("kab", "Taqbaylit"),
    ("kk", "Қазақ тілі"),
    ("ko", "한국어"),
    ("nb", "Norsk bokmål"),
    ("nl", "Nederlands"),
    ("pl", "Polski"),
    ("pt", "Português"),
    ("pt-br", "Português brasileiro"),
    ("ro", "Română"),
    ("ru", "Русский"),
    ("sk", "Slovenčina"),
    ("sl", "Slovenščina"),
    ("sq", "Shqip"),
    ("sr", "Српски"),
    ("sr-latn", "Srpski"),
    ("sv", "Svenska"),
    ("th", "ไทย"),
    ("tr", "Türkçe"),
    ("uk", "Українська"),
    ("zh-hans", "简体中文"),
    ("zh-hant", "正體中文"),
)

SITE_ID = 1

# If you set this to False, Django will make some optimizations so as not
# to load the internationalization machinery.
USE_I18N = True

# If you set this to False, Django will not format dates, numbers and

```

(繼續下一頁)

(繼續上一頁)

```

# calendars according to the current locale.
USE_L10N = True

# If you set this to False, Django will not use timezone-aware datetimes.
USE_TZ = True

# Type of automatic primary key, introduced in Django 3.2
DEFAULT_AUTO_FIELD = "django.db.models.AutoField"

# URL prefix to use, please see documentation for more details
URL_PREFIX = ""

# Absolute filesystem path to the directory that will hold user-uploaded files.
MEDIA_ROOT = os.path.join(DATA_DIR, "media")

# URL that handles the media served from MEDIA_ROOT. Make sure to use a
# trailing slash.
MEDIA_URL = f"{URL_PREFIX}/media/"

# Absolute path to the directory static files should be collected to.
# Don't put anything in this directory yourself; store your static files
# in apps' "static/" subdirectories and in STATICFILES_DIRS.
STATIC_ROOT = os.path.join(DATA_DIR, "static")

# URL prefix for static files.
STATIC_URL = f"{URL_PREFIX}/static/"

# Additional locations of static files
STATICFILES_DIRS = (
    # Put strings here, like "/home/html/static" or "C:/www/django/static".
    # Always use forward slashes, even on Windows.
    # Don't forget to use absolute paths, not relative paths.
)

# List of finder classes that know how to find static files in
# various locations.
STATICFILES_FINDERS = (
    "django.contrib.staticfiles.finders.FileSystemFinder",
    "django.contrib.staticfiles.finders.AppDirectoriesFinder",
    "compressor.finders.CompressorFinder",
)

# Make this unique, and don't share it with anybody.
# You can generate it using weblate-generate-secret-key
SECRET_KEY = ""

TEMPLATES = [
    {
        "BACKEND": "django.template.backends.django.DjangoTemplates",
        "OPTIONS": {
            "context_processors": [
                "django.contrib.auth.context_processors.auth",
                "django.template.context_processors.debug",
                "django.template.context_processors.i18n",
                "django.template.context_processors.request",
                "django.template.context_processors.csrf",
                "django.contrib.messages.context_processors.messages",
                "weblate.trans.context_processors.weblate_context",
            ],
        },
        "APP_DIRS": True,
    },

```

(繼續下一頁)



(繼續上一頁)

```

    }
]

# GitHub username and token for sending pull requests.
# Please see the documentation for more details.
GITHUB_CREDENTIALS = {}

# GitLab username and token for sending merge requests.
# Please see the documentation for more details.
GITLAB_CREDENTIALS = {}

# Authentication configuration
AUTHENTICATION_BACKENDS = (
    "social_core.backends.email.EmailAuth",
    # "social_core.backends.google.GoogleOAuth2",
    # "social_core.backends.github.GithubOAuth2",
    # "social_core.backends.bitbucket.BitbucketOAuth2",
    # "social_core.backends.suse.OpenSUSEOpenId",
    # "social_core.backends.ubuntu.UbuntuOpenId",
    # "social_core.backends.fedora.FedoraOpenId",
    # "social_core.backends.facebook.FacebookOAuth2",
    "weblate.accounts.auth.WeblateUserBackend",
)

# Custom user model
AUTH_USER_MODEL = "weblate_auth.User"

# Social auth backends setup
SOCIAL_AUTH_GITHUB_KEY = ""
SOCIAL_AUTH_GITHUB_SECRET = ""
SOCIAL_AUTH_GITHUB_SCOPE = ["user:email"]

SOCIAL_AUTH_GITHUB_ORG_KEY = ""
SOCIAL_AUTH_GITHUB_ORG_SECRET = ""
SOCIAL_AUTH_GITHUB_ORG_NAME = ""

SOCIAL_AUTH_GITHUB_TEAM_KEY = ""
SOCIAL_AUTH_GITHUB_TEAM_SECRET = ""
SOCIAL_AUTH_GITHUB_TEAM_ID = ""

SOCIAL_AUTH_BITBUCKET_OAUTH2_KEY = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_SECRET = ""
SOCIAL_AUTH_BITBUCKET_OAUTH2_VERIFIED_EMAILS_ONLY = True

SOCIAL_AUTH_FACEBOOK_KEY = ""
SOCIAL_AUTH_FACEBOOK_SECRET = ""
SOCIAL_AUTH_FACEBOOK_SCOPE = ["email", "public_profile"]
SOCIAL_AUTH_FACEBOOK_PROFILE_EXTRA_PARAMS = {"fields": "id,name,email"}

SOCIAL_AUTH_GOOGLE_OAUTH2_KEY = ""
SOCIAL_AUTH_GOOGLE_OAUTH2_SECRET = ""

# Social auth settings
SOCIAL_AUTH_PIPELINE = (
    "social_core.pipeline.social_auth.social_details",
    "social_core.pipeline.social_auth.social_uid",
    "social_core.pipeline.social_auth.auth_allowed",
    "social_core.pipeline.social_auth.social_user",
    "weblate.accounts.pipeline.store_params",
    "weblate.accounts.pipeline.verify_open",

```

(繼續下一頁)

(繼續上一頁)

```

"social_core.pipeline.user.get_username",
"weblate.accounts.pipeline.require_email",
"social_core.pipeline.mail.mail_validation",
"weblate.accounts.pipeline.revoke_mail_code",
"weblate.accounts.pipeline.ensure_valid",
"weblate.accounts.pipeline.remove_account",
"social_core.pipeline.social_auth.associate_by_email",
"weblate.accounts.pipeline.reauthenticate",
"weblate.accounts.pipeline.verify_username",
"social_core.pipeline.user.create_user",
"social_core.pipeline.social_auth.associate_user",
"social_core.pipeline.social_auth.load_extra_data",
"weblate.accounts.pipeline.cleanup_next",
"weblate.accounts.pipeline.user_full_name",
"weblate.accounts.pipeline.store_email",
"weblate.accounts.pipeline.notify_connect",
"weblate.accounts.pipeline.password_reset",
)
SOCIAL_AUTH_DISCONNECT_PIPELINE = (
    "social_core.pipeline.disconnect.allowed_to_disconnect",
    "social_core.pipeline.disconnect.get_entries",
    "social_core.pipeline.disconnect.revoke_tokens",
    "weblate.accounts.pipeline.cycle_session",
    "weblate.accounts.pipeline.adjust_primary_mail",
    "weblate.accounts.pipeline.notify_disconnect",
    "social_core.pipeline.disconnect.disconnect",
    "weblate.accounts.pipeline.cleanup_next",
)

# Custom authentication strategy
SOCIAL_AUTH_STRATEGY = "weblate.accounts.strategy.WeblateStrategy"

# Raise exceptions so that we can handle them later
SOCIAL_AUTH_RAISE_EXCEPTIONS = True

SOCIAL_AUTH_EMAIL_VALIDATION_FUNCTION = "weblate.accounts.pipeline.send_validation"
SOCIAL_AUTH_EMAIL_VALIDATION_URL = f"{URL_PREFIX}/accounts/email-sent/"
SOCIAL_AUTH_LOGIN_ERROR_URL = f"{URL_PREFIX}/accounts/login/"
SOCIAL_AUTH_EMAIL_FORM_URL = f"{URL_PREFIX}/accounts/email/"
SOCIAL_AUTH_NEW_ASSOCIATION_REDIRECT_URL = f"{URL_PREFIX}/accounts/profile/#account
↪"
SOCIAL_AUTH_PROTECTED_USER_FIELDS = ("email",)
SOCIAL_AUTH_SLUGIFY_USERNAMES = True
SOCIAL_AUTH_SLUGIFY_FUNCTION = "weblate.accounts.pipeline.slugify_username"

# Password validation configuration
AUTH_PASSWORD_VALIDATORS = [
    {
        "NAME": "django.contrib.auth.password_validation.
↪UserAttributeSimilarityValidator" # noqa: E501, pylint: disable=line-too-long
    },
    {
        "NAME": "django.contrib.auth.password_validation.MinimumLengthValidator",
        "OPTIONS": {"min_length": 10},
    },
    {"NAME": "django.contrib.auth.password_validation.CommonPasswordValidator"},
    {"NAME": "django.contrib.auth.password_validation.NumericPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.CharsPasswordValidator"},
    {"NAME": "weblate.accounts.password_validation.PastPasswordsValidator"},
    # Optional password strength validation by django-zxcvbn-password
    # {

```

(繼續下一頁)

(繼續上一頁)

```

#     "NAME": "zxcvbn_password.ZXCVBNValidator",
#     "OPTIONS": {
#         "min_score": 3,
#         "user_attributes": ("username", "email", "full_name")
#     }
# },
]

# Password hashing (prefer Argon)
PASSWORD_HASHERS = [
    "django.contrib.auth.hashers.Argon2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2PasswordHasher",
    "django.contrib.auth.hashers.PBKDF2SHA1PasswordHasher",
    "django.contrib.auth.hashers.BCryptSHA256PasswordHasher",
]

# Allow new user registrations
REGISTRATION_OPEN = True

# Shortcut for login required setting
REQUIRE_LOGIN = False

# Middleware
MIDDLEWARE = [
    "weblate.middleware.RedirectMiddleware",
    "weblate.middleware.ProxyMiddleware",
    "django.middleware.security.SecurityMiddleware",
    "django.contrib.sessions.middleware.SessionMiddleware",
    "django.middleware.csrf.CsrfViewMiddleware",
    "weblate.accounts.middleware.AuthenticationMiddleware",
    "django.contrib.messages.middleware.MessageMiddleware",
    "django.middleware.clickjacking.XFrameOptionsMiddleware",
    "social_django.middleware.SocialAuthExceptionMiddleware",
    "weblate.accounts.middleware.RequireLoginMiddleware",
    "weblate.api.middleware.ThrottlingMiddleware",
    "weblate.middleware.SecurityMiddleware",
    "weblate.wladmin.middleware.ManageMiddleware",
]

ROOT_URLCONF = "weblate.urls"

# Django and Weblate apps
INSTALLED_APPS = [
    # Weblate apps on top to override Django locales and templates
    "weblate.addons",
    "weblate.auth",
    "weblate.checks",
    "weblate.formats",
    "weblate.glossary",
    "weblate.machinery",
    "weblate.trans",
    "weblate.lang",
    "weblate_language_data",
    "weblate.memory",
    "weblate.screenshots",
    "weblate.fonts",
    "weblate.accounts",
    "weblate.configuration",
    "weblate.utils",
    "weblate.vcs",
    "weblate.wladmin",

```

(繼續下一頁)

(繼續上一頁)

```

    "weblate.metrics",
    "weblate",
    # Optional: Git exporter
    "weblate.gitexport",
    # Standard Django modules
    "django.contrib.auth",
    "django.contrib.contenttypes",
    "django.contrib.sessions",
    "django.contrib.messages",
    "django.contrib.staticfiles",
    "django.contrib.admin.apps.SimpleAdminConfig",
    "django.contrib.admindocs",
    "django.contrib.sitemaps",
    "django.contrib.humanize",
    # Third party Django modules
    "social_django",
    "crispy_forms",
    "compressor",
    "rest_framework",
    "rest_framework.authtoken",
    "django_filters",
]

# Custom exception reporter to include some details
DEFAULT_EXCEPTION_REPORTER_FILTER = "weblate.trans.debug.
↪WeblateExceptionReporterFilter"

# Default logging of Weblate messages
# - to syslog in production (if available)
# - otherwise to console
# - you can also choose "logfile" to log into separate file
#   after configuring it below

# Detect if we can connect to syslog
HAVE_SYSLOG = False
if platform.system() != "Windows":
    try:
        handler = SysLogHandler(address="/dev/log", facility=SysLogHandler.LOG_
↪LOCAL2)
        handler.close()
        HAVE_SYSLOG = True
    except OSError:
        HAVE_SYSLOG = False

if DEBUG or not HAVE_SYSLOG:
    DEFAULT_LOG = "console"
else:
    DEFAULT_LOG = "syslog"
DEFAULT_LOGLEVEL = "DEBUG" if DEBUG else "INFO"

# A sample logging configuration. The only tangible logging
# performed by this configuration is to send an email to
# the site admins on every HTTP 500 error when DEBUG=False.
# See http://docs.djangoproject.com/en/stable/topics/logging for
# more details on how to customize your logging configuration.
LOGGING = {
    "version": 1,
    "disable_existing_loggers": True,
    "filters": {"require_debug_false": {"()": "django.utils.log.RequireDebugFalse"}
↪},
    "formatters": {

```

(繼續下一頁)

(繼續上一頁)

```

"syslog": {"format": "weblate[% (process)s]: % (levelname)s % (message)s"},
"simple": {"format": "[% (asctime)s: % (levelname)s/% (process)s] % (message)s"},
↪ },
"logfile": {"format": "[% (asctime)s % (levelname)s % (message)s"],
"django.server": {
    "()": "django.utils.log.ServerFormatter",
    "format": "[% (server_time)s] % (message)s",
},
},
"handlers": {
    "mail_admins": {
        "level": "ERROR",
        "filters": ["require_debug_false"],
        "class": "django.utils.log.AdminEmailHandler",
        "include_html": True,
    },
    "console": {
        "level": "DEBUG",
        "class": "logging.StreamHandler",
        "formatter": "simple",
    },
    "django.server": {
        "level": "INFO",
        "class": "logging.StreamHandler",
        "formatter": "django.server",
    },
    "syslog": {
        "level": "DEBUG",
        "class": "logging.handlers.SysLogHandler",
        "formatter": "syslog",
        "address": "/dev/log",
        "facility": SysLogHandler.LOG_LOCAL2,
    },
    # Logging to a file
    # "logfile": {
    #     "level": "DEBUG",
    #     "class": "logging.handlers.RotatingFileHandler",
    #     "filename": "/var/log/weblate/weblate.log",
    #     "maxBytes": 100000,
    #     "backupCount": 3,
    #     "formatter": "logfile",
    # },
},
"loggers": {
    "django.request": {
        "handlers": ["mail_admins", DEFAULT_LOG],
        "level": "ERROR",
        "propagate": True,
    },
    "django.server": {
        "handlers": ["django.server"],
        "level": "INFO",
        "propagate": False,
    },
    # Logging database queries
    # "django.db.backends": {
    #     "handlers": [DEFAULT_LOG],
    #     "level": "DEBUG",
    # },
    "weblate": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Logging VCS operations

```

(繼續下一頁)

(繼續上一頁)

```

    "weblate.vcs": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Python Social Auth
    "social": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # Django Authentication Using LDAP
    "django_auth_ldap": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
    # SAML IdP
    "djangosaml2idp": {"handlers": [DEFAULT_LOG], "level": DEFAULT_LOGLEVEL},
},
}

# Remove syslog setup if it's not present
if not HAVE_SYSLOG:
    del LOGGING["handlers"]["syslog"]

# List of machine translations
MT_SERVICES = (
    # "weblate.machinery.apertium.ApertiumAPYTranslation",
    # "weblate.machinery.baidu.BaiduTranslation",
    # "weblate.machinery.deepl.DeepLTranslation",
    # "weblate.machinery.glosbe.GlosbeTranslation",
    # "weblate.machinery.google.GoogleTranslation",
    # "weblate.machinery.googlev3.GoogleV3Translation",
    # "weblate.machinery.libretranslate.LibreTranslateTranslation",
    # "weblate.machinery.microsoft.MicrosoftCognitiveTranslation",
    # "weblate.machinery.microsoftterminology.MicrosoftTerminologyService",
    # "weblate.machinery.modernmt.ModernMTTranslation",
    # "weblate.machinery.mymemory.MyMemoryTranslation",
    # "weblate.machinery.netease.NeteaseSightTranslation",
    # "weblate.machinery.tmserver.AmagamaTranslation",
    # "weblate.machinery.tmserver.TMServerTranslation",
    # "weblate.machinery.yandex.YandexTranslation",
    # "weblate.machinery.saptranslationhub.SAPTranslationHub",
    # "weblate.machinery.youdao.YoudaoTranslation",
    "weblate.machinery.weblatetm.WeblateTranslation",
    "weblate.memory.machine.WeblateMemory",
)

# Machine translation API keys

# URL of the Apertium APY server
MT_APERTIUM_APY = None

# DeepL API key
MT_DEEPL_KEY = None

# LibreTranslate
MT_LIBRETRANSLATE_API_URL = None
MT_LIBRETRANSLATE_KEY = None

# Microsoft Cognitive Services Translator API, register at
# https://portal.azure.com/
MT_MICROSOFT_COGNITIVE_KEY = None
MT_MICROSOFT_REGION = None

# ModernMT
MT_MODERNMT_KEY = None

# MyMemory identification email, see
# https://mymemory.translated.net/doc/spec.php
MT_MYMEMORY_EMAIL = None

```

(繼續下一頁)

(繼續上一頁)

```

# Optional MyMemory credentials to access private translation memory
MT_MYMEMORY_USER = None
MT_MYMEMORY_KEY = None

# Google API key for Google Translate API v2
MT_GOOGLE_KEY = None

# Google Translate API3 credentials and project id
MT_GOOGLE_CREDENTIALS = None
MT_GOOGLE_PROJECT = None

# Baidu app key and secret
MT_BAIDU_ID = None
MT_BAIDU_SECRET = None

# Youdao Zhiyun app key and secret
MT_YOUDAO_ID = None
MT_YOUDAO_SECRET = None

# Netease Sight (Jianwai) app key and secret
MT_NETEASE_KEY = None
MT_NETEASE_SECRET = None

# API key for Yandex Translate API
MT_YANDEX_KEY = None

# tmserver URL
MT_TMSERVER = None

# SAP Translation Hub
MT_SAP_BASE_URL = None
MT_SAP_SANDBOX_APIKEY = None
MT_SAP_USERNAME = None
MT_SAP_PASSWORD = None
MT_SAP_USE_MT = True

# Use HTTPS when creating redirect URLs for social authentication, see
# documentation for more details:
# https://python-social-auth-docs.readthedocs.io/en/latest/configuration/settings.
# ↪html#processing-redirects-and-urlopen
SOCIAL_AUTH_REDIRECT_IS_HTTPS = ENABLE_HTTPS

# Make CSRF cookie HttpOnly, see documentation for more details:
# https://docs.djangoproject.com/en/1.11/ref/settings/#csrf-cookie-httponly
CSRF_COOKIE_HTTPONLY = True
CSRF_COOKIE_SECURE = ENABLE_HTTPS
# Store CSRF token in session
CSRF_USE_SESSIONS = True
# Customize CSRF failure view
CSRF_FAILURE_VIEW = "weblate.trans.views.error.csrf_failure"
SESSION_COOKIE_SECURE = ENABLE_HTTPS
SESSION_COOKIE_HTTPONLY = True
# SSL redirect
SECURE_SSL_REDIRECT = ENABLE_HTTPS
SECURE_SSL_HOST = SITE_DOMAIN
# Sent referrer only for same origin links
SECURE_REFERRER_POLICY = "same-origin"
# SSL redirect URL exemption list
SECURE_REDIRECT_EXEMPT = (r"healthz/$",) # Allowing HTTP access to health check
# Session cookie age (in seconds)
SESSION_COOKIE_AGE = 1000

```

(繼續下一頁)

(繼續上一頁)

```

SESSION_COOKIE_AGE_AUTHENTICATED = 1209600
SESSION_COOKIE_SAMESITE = "Lax"
# Increase allowed upload size
DATA_UPLOAD_MAX_MEMORY_SIZE = 50000000
# Allow more fields for case with a lot of subscriptions in profile
DATA_UPLOAD_MAX_NUMBER_FIELDS = 2000

# Apply session cookie settings to language cookie as well
LANGUAGE_COOKIE_SECURE = SESSION_COOKIE_SECURE
LANGUAGE_COOKIE_HTTPONLY = SESSION_COOKIE_HTTPONLY
LANGUAGE_COOKIE_AGE = SESSION_COOKIE_AGE_AUTHENTICATED * 10
LANGUAGE_COOKIE_SAMESITE = SESSION_COOKIE_SAMESITE

# Some security headers
SECURE_BROWSER_XSS_FILTER = True
X_FRAME_OPTIONS = "DENY"
SECURE_CONTENT_TYPE_NOSNIFF = True

# Optionally enable HSTS
SECURE_HSTS_SECONDS = 31536000 if ENABLE_HTTPS else 0
SECURE_HSTS_PRELOAD = ENABLE_HTTPS
SECURE_HSTS_INCLUDE_SUBDOMAINS = ENABLE_HTTPS

# HTTPS detection behind reverse proxy
SECURE_PROXY_SSL_HEADER = None

# URL of login
LOGIN_URL = f"{URL_PREFIX}/accounts/login/"

# URL of logout
LOGOUT_URL = f"{URL_PREFIX}/accounts/logout/"

# Default location for login
LOGIN_REDIRECT_URL = f"{URL_PREFIX}/"

# Anonymous user name
ANONYMOUS_USER_NAME = "anonymous"

# Reverse proxy settings
IP_PROXY_HEADER = "HTTP_X_FORWARDED_FOR"
IP_BEHIND_REVERSE_PROXY = False
IP_PROXY_OFFSET = 0

# Sending HTML in mails
EMAIL_SEND_HTML = True

# Subject of emails includes site title
EMAIL_SUBJECT_PREFIX = f"[{SITE_TITLE}] "

# Enable remote hooks
ENABLE_HOOKS = True

# By default the length of a given translation is limited to the length of
# the source string * 10 characters. Set this option to False to allow longer
# translations (up to 10.000 characters)
LIMIT_TRANSLATION_LENGTH_BY_SOURCE_LENGTH = True

# Use simple language codes for default language/country combinations
SIMPLIFY_LANGUAGES = True

# Render forms using bootstrap

```

(繼續下一頁)



(繼續上一頁)

```

CRISPY_TEMPLATE_PACK = "bootstrap3"

# List of quality checks
# CHECK_LIST = (
#     "weblate.checks.same.SameCheck",
#     "weblate.checks.chars.BeginNewlineCheck",
#     "weblate.checks.chars.EndNewlineCheck",
#     "weblate.checks.chars.BeginSpaceCheck",
#     "weblate.checks.chars.EndSpaceCheck",
#     "weblate.checks.chars.DoubleSpaceCheck",
#     "weblate.checks.chars.EndStopCheck",
#     "weblate.checks.chars.EndColonCheck",
#     "weblate.checks.chars.EndQuestionCheck",
#     "weblate.checks.chars.EndExclamationCheck",
#     "weblate.checks.chars.EndEllipsisCheck",
#     "weblate.checks.chars.EndSemicolonCheck",
#     "weblate.checks.chars.MaxLengthCheck",
#     "weblate.checks.chars.KashidaCheck",
#     "weblate.checks.chars.PunctuationSpacingCheck",
#     "weblate.checks.format.PythonFormatCheck",
#     "weblate.checks.format.PythonBraceFormatCheck",
#     "weblate.checks.format.PHPFormatCheck",
#     "weblate.checks.format.CFormatCheck",
#     "weblate.checks.format.PerlFormatCheck",
#     "weblate.checks.format.JavaScriptFormatCheck",
#     "weblate.checks.format.LuaFormatCheck",
#     "weblate.checks.format.ObjectPascalFormatCheck",
#     "weblate.checks.format.SchemeFormatCheck",
#     "weblate.checks.format.CSharpFormatCheck",
#     "weblate.checks.format.JavaFormatCheck",
#     "weblate.checks.format.JavaMessageFormatCheck",
#     "weblate.checks.format.PercentPlaceholdersCheck",
#     "weblate.checks.format.VueFormattingCheck",
#     "weblate.checks.format.I18NextInterpolationCheck",
#     "weblate.checks.format.ESTemplateLiteralsCheck",
#     "weblate.checks.angularjs.AngularJSInterpolationCheck",
#     "weblate.checks.icu.ICUMessageFormatCheck",
#     "weblate.checks.icu.ICUSourceCheck",
#     "weblate.checks.qt.QtFormatCheck",
#     "weblate.checks.qt.QtPluralCheck",
#     "weblate.checks.ruby.RubyFormatCheck",
#     "weblate.checks.consistency.PluralsCheck",
#     "weblate.checks.consistency.SamePluralsCheck",
#     "weblate.checks.consistency.ConsistencyCheck",
#     "weblate.checks.consistency.TranslatedCheck",
#     "weblate.checks.chars.EscapedNewlineCountingCheck",
#     "weblate.checks.chars.NewLineCountCheck",
#     "weblate.checks.markup.BBCodeCheck",
#     "weblate.checks.chars.ZeroWidthSpaceCheck",
#     "weblate.checks.render.MaxSizeCheck",
#     "weblate.checks.markup.XMLValidityCheck",
#     "weblate.checks.markup.XMLTagsCheck",
#     "weblate.checks.markup.MarkdownRefLinkCheck",
#     "weblate.checks.markup.MarkdownLinkCheck",
#     "weblate.checks.markup.MarkdownSyntaxCheck",
#     "weblate.checks.markup.URLCheck",
#     "weblate.checks.markup.SafeHTMLCheck",
#     "weblate.checks.placeholders.PlaceholderCheck",
#     "weblate.checks.placeholders.RegexCheck",
#     "weblate.checks.duplicate.DuplicateCheck",
#     "weblate.checks.source.OptionalPluralCheck",

```

(繼續下一頁)

(繼續上一頁)

```

#     "weblate.checks.source.EllipsisCheck",
#     "weblate.checks.source.MultipleFailingCheck",
#     "weblate.checks.source.LongUntranslatedCheck",
#     "weblate.checks.format.MultipleUnnamedFormatsCheck",
#     "weblate.checks.glossary.GlossaryCheck",
# )

# List of automatic fixups
# AUTOFIX_LIST = (
#     "weblate.trans.autofixes.whitespace.SameBookendingWhitespace",
#     "weblate.trans.autofixes.chars.ReplaceTrailingDotsWithEllipsis",
#     "weblate.trans.autofixes.chars.RemoveZeroSpace",
#     "weblate.trans.autofixes.chars.RemoveControlChars",
# )

# List of enabled addons
# WEBLATE_ADDONS = (
#     "weblate.addons.gettext.GenerateMoAddon",
#     "weblate.addons.gettext.UpdateLinguasAddon",
#     "weblate.addons.gettext.UpdateConfigureAddon",
#     "weblate.addons.gettext.MsgmergeAddon",
#     "weblate.addons.gettext.GettextCustomizeAddon",
#     "weblate.addons.gettext.GettextAuthorComments",
#     "weblate.addons.cleanup.CleanupAddon",
#     "weblate.addons.cleanup.RemoveBlankAddon",
#     "weblate.addons.consistency.LanguaugeConsistencyAddon",
#     "weblate.addons.discovery.DiscoveryAddon",
#     "weblate.addons.autotranslate.AutoTranslateAddon",
#     "weblate.addons.flags.SourceEditAddon",
#     "weblate.addons.flags.TargetEditAddon",
#     "weblate.addons.flags.SameEditAddon",
#     "weblate.addons.flags.BulkEditAddon",
#     "weblate.addons.generate.GenerateFileAddon",
#     "weblate.addons.generate.PseudolocaleAddon",
#     "weblate.addons.generate.PrefillAddon",
#     "weblate.addons.json.JSONCustomizeAddon",
#     "weblate.addons.xml.XMLCustomizeAddon",
#     "weblate.addons.properties.PropertiesSortAddon",
#     "weblate.addons.git.GitSquashAddon",
#     "weblate.addons.removal.RemoveComments",
#     "weblate.addons.removal.RemoveSuggestions",
#     "weblate.addons.resx.ResxUpdateAddon",
#     "weblate.addons.yaml.YAMLCustomizeAddon",
#     "weblate.addons.cdn.CDNJSAddon",
# )

# E-mail address that error messages come from.
SERVER_EMAIL = "noreply@example.com"

# Default email address to use for various automated correspondence from
# the site managers. Used for registration emails.
DEFAULT_FROM_EMAIL = "noreply@example.com"

# List of URLs your site is supposed to serve
ALLOWED_HOSTS = ["*"]

# Configuration for caching
CACHES = {
    "default": {
        "BACKEND": "django_redis.cache.RedisCache",
        "LOCATION": "redis://127.0.0.1:6379/1",
    }
}

```

(繼續下一頁)

(繼續上一頁)

```

# If redis is running on same host as Weblate, you might
# want to use unix sockets instead:
# "LOCATION": "unix:///var/run/redis/redis.sock?db=1",
"OPTIONS": {
    "CLIENT_CLASS": "django_redis.client.DefaultClient",
    "PARSER_CLASS": "redis.connection.HiredisParser",
    # If you set password here, adjust CELERY_BROKER_URL as well
    "PASSWORD": None,
    "CONNECTION_POOL_KWARGS": {},
},
"KEY_PREFIX": "weblate",
"TIMEOUT": 3600,
},
"avatar": {
    "BACKEND": "django.core.cache.backends.filebased.FileBasedCache",
    "LOCATION": os.path.join(DATA_DIR, "avatar-cache"),
    "TIMEOUT": 86400,
    "OPTIONS": {"MAX_ENTRIES": 1000},
},
}

# Store sessions in cache
SESSION_ENGINE = "django.contrib.sessions.backends.cache"
# Store messages in session
MESSAGE_STORAGE = "django.contrib.messages.storage.session.SessionStorage"

# REST framework settings for API
REST_FRAMEWORK = {
    # Use Django's standard `django.contrib.auth` permissions,
    # or allow read-only access for unauthenticated users.
    "DEFAULT_PERMISSION_CLASSES": [
        # Require authentication for login required sites
        "rest_framework.permissions.IsAuthenticated"
        if REQUIRE_LOGIN
        else "rest_framework.permissions.IsAuthenticatedOrReadOnly"
    ],
    "DEFAULT_AUTHENTICATION_CLASSES": (
        "rest_framework.authentication.TokenAuthentication",
        "weblate.api.authentication.BearerAuthentication",
        "rest_framework.authentication.SessionAuthentication",
    ),
    "DEFAULT_THROTTLE_CLASSES": (
        "weblate.api.throttling.UserRateThrottle",
        "weblate.api.throttling.AnonRateThrottle",
    ),
    "DEFAULT_THROTTLE_RATES": {"anon": "100/day", "user": "5000/hour"},
    "DEFAULT_PAGINATION_CLASS": "weblate.api.pagination.StandardPagination",
    "PAGE_SIZE": 50,
    "VIEW_DESCRIPTION_FUNCTION": "weblate.api.views.get_view_description",
    "UNAUTHENTICATED_USER": "weblate.auth.models.get_anonymous",
}

# Fonts CDN URL
FONTS_CDN_URL = None

# Django compressor offline mode
COMPRESS_OFFLINE = False
COMPRESS_OFFLINE_CONTEXT = [
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI": ↵
↵ True},
    {"fonts_cdn_url": FONTS_CDN_URL, "STATIC_URL": STATIC_URL, "LANGUAGE_BIDI": ↵

```

(繼續下一頁)

(繼續上一頁)

```

↪False},
]

# Require login for all URLs
if REQUIRE_LOGIN:
    LOGIN_REQUIRED_URLS = (r"/(.*)$",)

# In such case you will want to include some of the exceptions
# LOGIN_REQUIRED_URLS_EXCEPTIONS = (
#     rf"{URL_PREFIX}/accounts/(.*)$", # Required for login
#     rf"{URL_PREFIX}/admin/login/(.*)$", # Required for admin login
#     rf"{URL_PREFIX}/static/(.*)$", # Required for development mode
#     rf"{URL_PREFIX}/widgets/(.*)$", # Allowing public access to widgets
#     rf"{URL_PREFIX}/data/(.*)$", # Allowing public access to data exports
#     rf"{URL_PREFIX}/hooks/(.*)$", # Allowing public access to notification hooks
#     rf"{URL_PREFIX}/healthz/$", # Allowing public access to health check
#     rf"{URL_PREFIX}/api/(.*)$", # Allowing access to API
#     rf"{URL_PREFIX}/js/i18n/$", # JavaScript localization
#     rf"{URL_PREFIX}/contact/$", # Optional for contact form
#     rf"{URL_PREFIX}/legal/(.*)$", # Optional for legal app
#     rf"{URL_PREFIX}/avatar/(.*)$", # Optional for avatars
# )

# Silence some of the Django system checks
SILENCED_SYSTEM_CHECKS = [
    # We have modified django.contrib.auth.middleware.AuthenticationMiddleware
    # as weblate.accounts.middleware.AuthenticationMiddleware
    "admin.E408"
]

# Celery worker configuration for testing
# CELERY_TASK_ALWAYS_EAGER = True
# CELERY_BROKER_URL = "memory://"
# CELERY_TASK_EAGER_PROPAGATES = True
# Celery worker configuration for production
CELERY_TASK_ALWAYS_EAGER = False
CELERY_BROKER_URL = "redis://localhost:6379"
CELERY_RESULT_BACKEND = CELERY_BROKER_URL

# Celery settings, it is not recommended to change these
CELERY_WORKER_MAX_MEMORY_PER_CHILD = 200000
CELERY_BEAT_SCHEDULE_FILENAME = os.path.join(DATA_DIR, "celery", "beat-schedule")
CELERY_TASK_ROUTES = {
    "weblate.trans.tasks.auto_translate*": {"queue": "translate"},
    "weblate.accounts.tasks.notify*": {"queue": "notify"},
    "weblate.accounts.tasks.send_mails": {"queue": "notify"},
    "weblate.utils.tasks.settings_backup": {"queue": "backup"},
    "weblate.utils.tasks.database_backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup": {"queue": "backup"},
    "weblate.wladmin.tasks.backup_service": {"queue": "backup"},
    "weblate.memory.tasks.*": {"queue": "memory"},
}

# Enable plain database backups
DATABASE_BACKUP = "plain"

# Enable auto updating
AUTO_UPDATE = False

# GPG commits signing
WEBLATE_GPG_IDENTITY = None

```

(繼續下一頁)

(繼續上一頁)

```
# Third party services integration
MATOMO_SITE_ID = None
MATOMO_URL = None
GOOGLE_ANALYTICS_ID = None
SENTRY_DSN = None
SENTRY_ENVIRONMENT = SITE_DOMAIN
AKISMET_API_KEY = None
```

## 2.18 管理命令

**備註：**不同用戶下運行管理命令而不是一人運行您的 web 服務器，可以導致文件得到錯誤的權限，更多細節請查看[文件系統權限](#)。

您會找到基本的管理命令（作 Django 源中的 `file:./manage.py` 來獲得它，或者作可安裝在 Weblate 頂層的腳本調用 `command:weblate` 中的擴展組來獲得它）。

### 2.18.1 Invoking management commands

如上面所提到的，以用依賴於您如何安裝 Weblate。

如果使用 Virtualenv 來運行 Weblate，那麼您可以或者 `weblate` 指定全路徑，或者在調用前激活 `virtualenv`：

```
# Direct invocation
~/weblate-env/bin/weblate

# Activating virtualenv adds it to search path
. ~/weblate-env/bin/activate
weblate
```

如果您直接使用源代碼（來源於 tarball 或 Git checkout），管理腳本可以在 Weblate 源文件的 `./manage.py` 中獲得。要運行它：

```
python ./manage.py list_versions
```

If you've installed Weblate using the pip installer, or by using the `./setup.py` script, the `weblate` is installed to your path (or virtualenv path), from where you can use it to control Weblate:

```
weblate list_versions
```

對於 Docker 映像，腳本向上面一樣安裝，您可以使用 `docker exec` 來運行：

```
docker exec --user weblate <container> weblate list_versions
```

對於 `docker-compose`，過程是相似的，您只是必須使用 `docker-compose exec`：

```
docker-compose exec --user weblate weblate weblate list_versions
```

在您需要向它傳遞文件的情況下，您可以臨時添加卷：

```
docker-compose exec --user weblate /tmp:/tmp weblate weblate importusers /tmp/
↪users.json
```

**也參考:**

: Doc: 安裝/*docker*, : doc: westal / Venv-debian', : doc: westall / Venv-suse'

**2.18.2 add\_suggestions**

**weblate add\_suggestions** <project> <component> <language> <file>

在 2.5 版本新加入。

從文件導入翻譯，來作 給定翻譯的建議來使用。它跳過了 翻譯的步驟；只會添加不同的 容。

**--author** USER@EXAMPLE.COM

建議的作者電子郵箱地址。這個用 必須在導入前存在（您可以根據需要在管理界面建立一個）。

例：

```
weblate --author michal@cihar.com add_suggestions weblate application cs /tmp/
↪ suggestions-cs.po
```

**2.18.3 auto\_translate**

**weblate auto\_translate** <project> <component> <language>

在 2.5 版本新加入。

在 4.6 版本變更: Added parameter for translation mode.

根據其他組件翻譯進行自動翻譯。

**--source** PROJECT/COMPONENT

指定組件，用作可獲得翻譯的來源。如果不指定，將使用項目中的所有組件。

**--user** USERNAME

指定列出的用 名，作 翻譯的作者。如果不指定那 使用 “匿名用 ”。

**--overwrite**

是否去覆蓋現有的翻譯。

**--inconsistent**

是否去覆蓋現有的不一致的翻譯（請參見 不一致 ）。

**--add**

如果給定的翻譯不存在，自動添加語言。

**--mt** MT

實用機器翻譯而不是其他組件作 機器翻譯。

**--threshold** THRESHOLD

用於機器翻譯的相似性 值，默認 80。

**--mode** MODE

Specify translation mode, default is translate but fuzzy or suggest can be used.

例：

```
weblate auto_translate --user nijel --inconsistent --source weblate/application_
↪ weblate website cs
```

**也參考:**

自動翻譯

## 2.18.4 celery\_queues

**weblate celery\_queues**

在 3.7 版本新加入。

顯示 Celery 任務隊列的長度。

## 2.18.5 checkgit

**weblate checkgit <project|project/component>**

打印後端 Git 倉庫的當前狀態。

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

## 2.18.6 commitgit

**weblate commitgit <project|project/component>**

將任何可能待定的更改提交給後端 Git 倉庫。

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

## 2.18.7 commit\_pending

**weblate commit\_pending <project|project/component>**

提交早於給定時間段的待定更改。

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

**--age** HOURS

時間段以小時為單位。如果不指定，則使用在組件配置中配置的值。

---

**備註：** 這由 Weblate 在後台自動執行，所以實際不需要手動調用，除了要限制早於組件配置指定的執行。

---

**也參考：**

*Running maintenance tasks, COMMIT\_PENDING\_HOURS*

## 2.18.8 cleanuptrans

**weblate cleanuptrans**

清理無主的檢查和翻譯建議。這通常不需要手動運行，因為清理在後台自動運行。

**也參考：**

*Running maintenance tasks*

### 2.18.9 cleanup\_ssh\_keys

#### **weblate cleanup\_ssh\_keys**

在 4.9.1 版本新加入。

Performs cleanup of stored SSH host keys:

- Removes deprecated RSA keys for GitHub which might cause issues connecting to GitHub.
- Removes duplicate entries in host keys.

**也參考:**

[SSH 倉庫](#)

### 2.18.10 createadmin

#### **weblate createadmin**

除非指定，否則用隨機密碼建立 admin 賬 F。

**--password** PASSWORD

在命令行提供密碼，而不要生成隨機的。

**--no-password**

不要設置密碼，這對 ‘update’ 可能有用。

**--username** USERNAME

使用給定的姓名而不是 admin。

**--email** USER@EXAMPLE.COM

指定 admin 的電子郵箱地址。

**--name**

指定 admin 的姓名（可見的）。

**--update**

更新現有的用 F（您可以用這個來更改密碼）。

在 2.9 版本變更: 添加參數 --username、--email、--name 和 --update。

### 2.18.11 dump\_memory

#### **weblate dump\_memory**

在 2.20 版本新加入。

將包含 Weblate 翻譯記憶庫 F 容的 JSON 文件導出。

**也參考:**

翻譯記憶, [Weblate Translation Memory Schema](#)



### 2.18.12 dumpuserdata

**weblate dumpuserdata <file.json>**

Dumps userdata to a file for later use by *importuserdata*.

---

**提示：** 這在遷移或合併 Weblate 事例是會很方便。

---

### 2.18.13 import\_demo

**weblate import\_demo**

在 4.1 版本新加入。

Creates a demo project with components based on <<https://github.com/WeblateOrg/demo>>. Make sure the celery tasks are running before running this command.

這在開發 Weblate 時會有用。

### 2.18.14 import\_json

**weblate import\_json <json-file>**

在 2.7 版本新加入。

根據 JSON 數據批量導入組件。

導入的 JSON 文件結構非常符合組件對象（請參見 *GET /api/components/(string:project)/(string:component)/*）。您必須包括 `name` 和 `filemask` 字段。

**--project** PROJECT

指定從哪裏導入組件。

**--main-component** COMPONENT

對所有的使用來自這個組件的給定版本控制系統（VCS）倉庫。

**--ignore**

跳過（已經）導入的組件。

**--update**

更新（已經）導入的組件。

在 2.9 版本變更：那裏的參數 `--ignore` 和 `--update` 用於處理已經導入的組件。

JSON 文件的例子：

```
[
  {
    "slug": "po",
    "name": "Gettext PO",
    "file_format": "po",
    "filemask": "po/*.po",
    "new_lang": "none"
  },
  {
    "name": "Android",
    "filemask": "android/values-*/strings.xml",
    "template": "android/values/strings.xml",
    "repo": "weblate://test/test",
    "file_format": "aresource"
  }
]
```

(繼續下一頁)

```
}
]
```

也參考:

*import\_memory*

## 2.18.15 import\_memory

**weblate import\_memory <file>**

在 2.20 版本新加入。

將 TMX 或 JSON 文件導入 Weblate 翻譯記憶庫。

**--language-map** LANGMAP

允許將 TMX 的語言映射到 Weblate 翻譯記憶庫。語言代碼通常在 Weblate 進行規範化之後映射。

例如 `--language-map en_US:en` 將所有 `en_US` 字符串作 `en` 字符串來導入。

在您的 TMX 文件地區恰好與您在 Weblate 使用地區不同的情況下，這會有用。

也參考:

翻譯記憶, *Weblate Translation Memory Schema*

## 2.18.16 import\_project

**weblate import\_project <project> <gitrepo> <branch> <filemask>**

在 3.0 版本變更: `import_project` 命令現在基於組件探索 插件，導致一些行爲的更改，接受一些參數。

Batch imports components into project based on the file mask.

`<project>` 將已存在的項目命名，組件將導入其中。

`<gitrepo>` 確定了要使用的 Git 倉庫的 URL，而 `<branch>` 明了 Git 分支。了從現有的 Weblate 組件導入另外的翻譯組件，使用 `<gitrepo>` 的 `weblate://<project>/<component>` URL。

`<filemask>` 倉庫定義了文件發現。或者可以使用通配符來使它簡單，或者可以使用正則表達式的全部功能。

簡單的匹配對組件名稱使用 `**`，對語言使用 `*`，例如: `**/*.po`

正則表達式必須包含組命名的 *component* 和 *language*。例如: `(?P<language>[^\/*])/(?P<component>[^\/*]*)\.po`

根據文件，導入與現有的組件匹配，且添加不存在的那些。它不更改已經存在的那些。

**--name-template** TEMPLATE

使用 Django 模板語法來定制組件的名稱。

例如: `Documentation: {{ component }}`

**--base-file-template** TEMPLATE

單語言翻譯定制譯文模板文件。

例如: `{{ component }}/res/values/string.xml`

**--new-base-template** TEMPLATE

另外新的翻譯定制譯文模板文件。

例如: `{{ component }}/ts/en.ts`

**--file-format** FORMAT

您還可以使用的文件格式（請參見：[支持的文件格式](#)），默認自動檢測。

**--language-regex** REGEX

您可以使用這個參數指定語言過濾器（請參見：[組件配置](#)）。它必須是合法的正則表達式。

**--main-component**

您可以指定選擇哪個組件作主要的一個——即真正包含版本控制系統（VCS）倉庫的那個。

**--license** NAME

指定整體、項目或組件翻譯的許可。

**--license-url** URL

指定翻譯許可所在的 URL。

**--vcs** NAME

在需要指定使用哪個版本的輕質系統的情況下，您可以在這進行。默認版本控制是 Git。

了給出一些例子，讓我們導入兩個項目。

第一個是 Debian 手冊翻譯，那的每種語言都有各自的文件夾，面有每個章節的翻譯：

```
weblate import_project \
  debian-handbook \
  git://anonscm.debian.org/debian-handbook/debian-handbook.git \
  squeeze/master \
  '*/**.po'
```

然後 Tangaguru 工具，那需要指定文件格式和譯文模板文件，且指定所有組件和翻譯如何位於單一一個文件夾中：

```
weblate import_project \
  --file-format=properties \
  --base-file-template=web-app/tgol-web-app/src/main/resources/i18n/%s-I18N.
→properties \
  tanaguru \
  https://github.com/Tanaguru/Tanaguru \
  master \
  web-app/tgol-web-app/src/main/resources/i18n/**-I18N_*.properties
```

更雜的例子是關於解析文件名而從文件名中得到正確的組件和語言，像 `src/security/Numerous_security_holes_in_0.10.1.de.po`：

```
weblate import_project \
  tails \
  git://git.tails.boum.org/tails master \
  'wiki/src/security/(?P<component>.*).\.(?P<language>[^.]*)\.po$'
```

篩選出指定的語言的翻譯：

```
./manage import_project \
  --language-regex '^(cs|sk)$' \
  weblate \
  https://github.com/WeblateOrg/weblate.git \
  'weblate/locale/*/LC_MESSAGES/**/*.po'
```

導入 Sphinx 文檔，分成多個文件：

```
$ weblate import_project --name-template 'Documentation: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/**/*.po'
```

導入 Sphinx 文檔，分成多個文件和文件夾：

```
$ weblate import_project --name-template 'Directory 1: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/dir1/**/*.po'
$ weblate import_project --name-template 'Directory 2: %s' \
  --file-format po \
  project https://github.com/project/docs.git master \
  'docs/locale/*/LC_MESSAGES/dir2/**/*.po'
```

**也參考:**

更多具體的例子可以在 `starting` 章節找到，另外您會想要使用 `import_json`。

## 2.18.17 importuserdata

**weblate importuserdata <file.json>**

從 `dumpuserdata` 建立的文件中導入用 `JSON` 數據

## 2.18.18 importusers

**weblate importusers --check <file.json>**

從 Django `auth_users` 數據庫的 JSON 轉儲中導入用 `JSON`。

**--check**

使用這個選項可以檢查給定文件是否可以被導入，`--check` 且報告用 `JSON` 名或電子郵件地址可能導致的衝突。

可以從現有的 Django 安裝中導出用 `JSON`，這需要使用：

```
weblate dumpdata auth.User > users.json
```

## 2.18.19 install\_addon

在 3.2 版本新加入。

**weblate install\_addon --addon ADDON <project|project/component>**

將附加組件安裝到一組組件中。

**--addon ADDON**

要安裝的附加組件名稱。例如 `weblate.gettext.customize`。

**--configuration CONFIG**

JSON encoded configuration of an add-on.

**--update**

Update the existing add-on configuration.

可以或者定義將附加組件安裝到哪個項目或組件中（例如 `weblate/application`），或者使用 `--all` 來包括所有現有的組件。

`JSON` 所有的組件安裝自訂 `gettext` 輸出：

```
weblate install_addon --addon weblate.gettext.customize --config '{"width": -1}' --
  ↳ update --all
```

**也參考:**

附加元件

## 2.18.20 list\_languages

**weblate list\_languages <locale>**

列出 MediaWiki 標記中支持的語言——語言代碼、英語名稱和本地化名稱。

這用來生成 <[https://wiki.l10n.cz/Slovn%C3%ADk\\_s\\_n%C3%A1zvy\\_jazyk%C5%AF](https://wiki.l10n.cz/Slovn%C3%ADk_s_n%C3%A1zvy_jazyk%C5%AF)>.

## 2.18.21 list\_translators

**weblate list\_translators <project|project/component>**

對給定的項目列出這種語言做出貢獻的譯者：

```
[French]
Jean Dupont <jean.dupont@example.com>
[English]
John Doe <jd@example.com>
```

**--language-code**

用語言代碼而不是語言來列出名稱。

可以或者定義使用哪個項目或組件（例如 weblate/application），或者使用 **--all** 從所有現存的組件中列出翻譯者。

## 2.18.22 list\_versions

**weblate list\_versions**

列出所有 Weblate 依賴及其版本。

## 2.18.23 loadpo

**weblate loadpo <project|project/component>**

從磁盤重新加載翻譯（例如您在版本控制系統（VCS）倉庫完成一些更新的情況下）。

**--force**

強制更新，即使文件應該是更新的。

**--lang LANGUAGE**

將處理限制為單一語言。

您可以確定或者哪個項目或組件要更新（例如 weblate/application），或者使用 **--all** 來更新所有現有組件。

---

**備註：** 極少調用這個，Weblate 將對每一次版本控制系統（VCS）更新自動加載更改的文件。在您手動更改下層 Weblate 版本控制系統（VCS）倉庫的情況下，或在更新後的一些特殊情況下才需要這個。

---

### 2.18.24 lock\_translation

**weblate lock\_translation** <project|project/component>

防止進一步翻譯組件。

---

**提示：** 在您想要對下層倉庫進行一些維護時有用。

---

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

**也參考：**

`unlock_translation`

### 2.18.25 move\_language

**weblate move\_language** source target

在 3.0 版本新加入。

允許您合語言容。當更新到新版本，這個新版本對使用 (*generated*) 前綴建立的之前未知的語言包含名時，這會有用。它將所有容從 *source* 語言移動到 *target* 語言。

例：

```
weblate move_language cze cs
```

移動容後，您應該檢查是否有什落下了（這是因有人在同時更新倉庫而導致的競情），而且要除 (*generated*) 語言。

### 2.18.26 pushgit

**weblate pushgit** <project|project/component>

將執行的更改推送到上游版本控制系統（VCS）倉庫。

**--force-commit**

在推送前，制執行任何待定的更改。

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

---

**備：** 如果開了組件配置中的 `ref:component-push_on_commit`，Weblate 會自動推送更改，這是默認的。

---

### 2.18.27 unlock\_translation

**weblate unlock\_translation** <project|project/component>

將給定的組件解鎖，使它能被翻譯。

---

**提示：** 在您想要對下層倉庫進行一些維護時有用。

---



您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

也參考:

*lock\_translation*

## 2.18.28 setupgroups

**weblate setupgroups**

配置默認的組，可選地將所有用指定到那個默認組中。

**--no-privs-update**

關閉對現有組的自動更新（只添加新的）。

**--no-projects-update**

防止對現有項目的組的自動更新。這允許將新添加的組加入到現有項目中，請參見專案存取控制。

也參考:

特殊權限列表與建角色

## 2.18.29 setuplang

**weblate setuplang**

將 Weblate 中的確定語言的列表更新。

**--no-update**

關閉現有語言的自動更新（只添加新的）。

## 2.18.30 updatechecks

**weblate updatechecks <project|project/component>**

對所有字符串更新所有檢查。

---

**提示：** 對檢查進行主要更改的更新是有用的。

---

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

## 2.18.31 updategit

**weblate updategit <project|project/component>**

取回遠程版本控制系統（VCS）倉庫更新部緩存。

您可以確定或者哪個項目或組件要更新（例如 `weblate/application`），或者使用 `--all` 來更新所有現有組件。

---

**備：** 通常最好在倉庫中配置子，來觸發通知勾，而不是常規的通過 *updategit* 來投票。

---

## 2.19 公告

在 4.0 版本變更: 在此前的發布版本中，這個特性被稱爲白板消息。

通過張貼網站範圍的，每個項目、組件或語言的公告，向翻譯者提供信息。

宣布翻譯的目的、截止日期、狀態或特定目標。

用它可以接收到關注項目公告的通知（除非用選擇關閉）。

這會用於從發網站的目的到指定翻譯的目標的各種事情。

可以使用 *Post announcement*，在 *Manage* 菜單的每一張貼公告：

Webate Dashboard Projects Languages Checks

WebateOrg translated 90%

Translations will be used only if they reach 60%.

Components Languages Info Search Insights Files Tools Manage Share Not watching

Post announcement

**Message**

You can use Markdown and mention users by @username.

**Category**

Info (light blue)

Category defines color used for the message.

**Expiry date**

mm/dd/yyyy

The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

☒ **Notify users**

The message is shown for all translations within the project, until its given expiry, or permanently until it is deleted.

Add

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

還可以使用管理界面添加：



Weblate administration
WELCOME, **WEBLATE TEST**. [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

Home · Weblate translations · Announcements · Add Announcement




### Add Announcement




Required fields are marked in bold.




**Message:**


Translations will be used only if they reach 60%


You can use Markdown and mention users by @username.

**Project:** WeblateOrg   

**Component:** -----   

**Language:** -----   

**Category:** Info (light blue)   
Category defines color used for the message.

**Expiry date:** Today   
The message will be not shown after this date. Use it to announce string freeze and translation deadline for next release.

☒ Notify users

Save and add another
Save and continue editing
SAVE

然後根據特定的語境顯示公告：

無指定的語境

顯示在控制面版上（登 F 頁面）。

特定項目

項目 F 顯示，包括其所有的組件和翻譯。

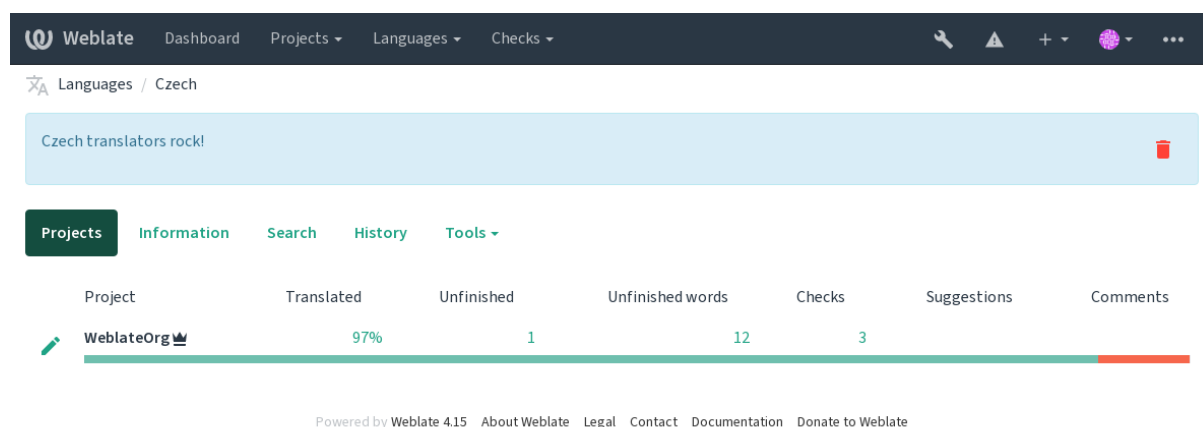
特定組件

對於給定的組件機器翻譯來顯示。

特定語言

顯示語言的全景和該語言的全部翻譯。

這就是語言全景頁面上的樣子：



## 2.20 組件列表

指定多個列表的組件，出現在用 F 控制面板上作 F 選項，從用用 F 可以選擇一個作 F 默認視圖。請參見 [控制面板](#) 來了解更多信息。

在 2.20 版本變更: 控制面板上出現的每個組件列表都會顯示狀態。

可以在管理界面的 *Component lists* 部分指定組件列表的名稱和 F 容。每種組件列表必須名稱來顯示給用 F，F 具有標識串將其顯示在 URL 中。

在 2.13 版本變更: 從管理界面檢查匿名用 F 的控制面板設置，修改掉控制面本顯示給未授權用 F 的 F 容。

### 2.20.1 Automatic component lists

在 2.13 版本新加入。

通過建立 *Automatic component list assignment* 規則，根據其標識串自動將組件添加到列表中。

- 對於維護大型安裝的逐漸列表有用，或者如果你希望在 Weblate 安裝中有一個包含所有組件的組件列表。

---

**提示:** F 作組件列表，包含自己的 Weblate 安裝時的所有組件。

---

1. Define *Automatic component list assignment* with `^.*$` as regular expression in both the project and the component fields, as shown on this image:

Weblate administration
WELCOME, **WEBLATE TEST**. [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

Home · Weblate translations · Component lists · Add Component list

## Add Component list

Required fields are marked in bold.

**Component list name:**   
Display name

**URL slug:**   
Name used in URLs and filenames.

☒ Show on dashboard  
When enabled this component list will be shown as a tab on the dashboard

Components:

Available components ⓘ

- WeblateOrg/Django
- WeblateOrg/Language names
- WeblateOrg/WeblateOrg

Choose all ⓘ

Chosen components ⓘ

+

Remove all

Hold down "Control", or "Command" on a Mac, to select more than one.

AUTOMATIC COMPONENT LIST ASSIGNMENTS

PROJECT REGULAR EXPRESSION ⓘ	COMPONENT REGULAR EXPRESSION ⓘ	DELETE? ⓘ
<input type="text" value="^.*\$"/>	<input type="text" value="^.*\$"/>	<input type="button" value="✕"/>

+ Add another Automatic component list assignment

Save and add another

Save and continue editing

SAVE

## 2.21 Optional Weblate modules

可以獲得幾個可選的模塊來配置您的設置。

### 2.21.1 Git 導出器

在 2.10 版本新加入。

使用 HTTP(S) 您提供對底層 Git 倉庫的只讀訪問。

#### 安裝

1. 將 `weblate.gitexport` 添加到 `settings.py` 中安裝的 `apps` 中：

```
INSTALLED_APPS += ("weblate.gitexport",)
```

2. 通過安裝後遷移數據庫，將現有的倉庫導出：

```
weblate migrate
```

#### Usage

模塊自動加入 Weblate，且在組件配置中設置導出倉庫 URL。倉庫在 Weblate URL 的 `/git/` 部分下是可以訪問的，例如 `https://example.org/git/weblate/main/`。

公共可用項目的倉庫可以被克隆而無需認證：

```
git clone 'https://example.org/git/weblate/main/'
```

對存儲庫的受限制的訪問（使用 ‘Private’ *access control* 或在 `:setting:REQUIRE_LOGIN` 處於啟用狀態時）需要一個 API 令牌，獲取位置在你的 `:ref: user-profile <user-profile>`：

```
git clone 'https://user:KEY@example.org/git/weblate/main/'
```

---

**提示：** 成員或 *Users* 用組與匿名用默認通過 *Access repository* 和 *Power user* 角色訪問公開項目。

---

### 2.21.2 帳單

在 2.4 版本新加入。

這用在 [Hosted Weblate](#) 上來確定付費套餐，跟踪收據和使用限制。

#### 安裝

1. Add `weblate.billing` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.billing",)
```

2. 運行數據庫遷移，來可選地模塊安裝另外的數據庫結構：

```
weblate migrate
```

## Usage

安裝後您可以在管理界面控制賬單。允許了賬單的用 戶將在他們的使用者個人檔案 中得到新的 *Billing* 標 識。

賬單模塊額外允許項目管理員不是超級用 戶的情 況下去新建新的項目和組件（請參見添加翻譯項目和組件）。當後面的條件滿足是這是可能的：

- 賬單在其配置的限制下（任何過度的使用都會阻止新建項目/組件）， 戶且被支付（如果價格 戶非零值的話）
- 用 戶是現有的帶有賬單項目的管理員，或者用 戶是賬單的所有者（當 戶用 戶新建新的賬單，而允許導入新的項目時，後者是必要的）。

在新建項目時，用 戶在訪問多個賬單的情 況下，能 戶選擇將項目記在哪個賬單上。

### 2.21.3 法律

在 2.15 版本新加入。

這用在 [Hosted Weblate](#) 上，來提供所需的法律文檔。它開始時提供空白文檔，會希望您填充文檔中後面的模板：

**legal/documents/tos.html**  
服務條款文檔

**legal/documents/privacy.html**  
Privacy policy document

**legal/documents/summary.html**  
服務條款與隱私政策的簡短概 括

On changing the terms of service documents, please adjust `LEGAL_TOS_DATE` so that users are forced to agree with the updated documents.

---

**備 註：** 可以在這個 [Git 倉庫 <https://github.com/WeblateOrg/wllegal/tree/main/wllegal/templates/legal/documents>](https://github.com/WeblateOrg/wllegal/tree/main/wllegal/templates/legal/documents) 中獲取 Hosted Weblate 的法律文檔。

這些很可能對您 戶有直接的用處，但如果調整來滿足您的需求時，以此作 戶起點可能會比較方便。

---

## 安裝

1. Add `weblate.legal` to installed apps in `settings.py`:

```
INSTALLED_APPS += ("weblate.legal",)

# Optional:

# Social auth pipeline to confirm TOS upon registration/subsequent sign in
SOCIAL_AUTH_PIPELINE += ("weblate.legal.pipeline.tos_confirm",)

# Middleware to enforce TOS confirmation of signed in users
MIDDLEWARE += [
    "weblate.legal.middleware.RequireTOSMiddleware",
]
```

2. 運行數據庫遷移，來可選地 戶模塊安裝另外的數據庫結構：

```
weblate migrate
```

3. 編輯 `weblate/legal/templates/legal/` 文件夾中的而法律文檔，與您的服務匹配。

## Usage

安裝後編輯後，法律文檔顯示在 Weblate UI 中。

### 2.21.4 Avatars

頭像在服務器端下載並緩存，來減少對默認服務的網站的暴露。通過其配置的電子郵件地址來取回頭像的构建支持，可以使用 `ENABLE_AVATARS` 來關閉。

Weblate currently supports:

- Gravatar
- Libravatar

也參考:

個人頭像快取, `AVATAR_URL_PREFIX`, `ENABLE_AVATARS`

### 2.21.5 針對垃圾郵件的保護

你可以免受用發送垃圾郵件的侵擾，方法是使用 Akismet 服務。

1. 安裝 ‘akismet’ python 模塊（這已包含在官方 Docker Image 中）。
2. 獲取 Akismet API 密鑰。
3. 將其存儲：設置：`akismet_api_key`或：`envvar: 'weblate_akismet_api_key` in docker。

以下內容被發送到 AkisMet 進行檢查：

- Suggestions from unauthenticated users
- Project and component descriptions and links

---

備：這（除了其它事情以外）依賴於客戶端的 IP 地址，適當的配置請參見：在反向代理後面運行。

---

也參考:

在反向代理後面運行, `AKISMET_API_KEY`, `WEBLATE_AKISMET_API_KEY`

### 2.21.6 簽署 GnuPG 的 Git 承諾

在 3.1 版本新加入。

所有的承諾可以由 Weblate 時間的 GnuPG 密鑰簽署。

1. Turn on `WEBLATE_GPG_IDENTITY`. (Weblate will generate a GnuPG key when needed and will use it to sign all translation commits.)

這個特性需要安裝 GnuPG 2.1 或更新版。

您可以在 `DATA_DIR` 中找到密鑰，而公鑰顯示在“關於”頁面上：

[Dashboard](#)
[Projects](#)
[Languages](#)
[Checks](#)
[Register](#)
[Sign in](#)

[About Weblate](#) / [Weblate keys](#)

[About Weblate](#)
[Statistics](#)
[Keys](#)

### SSH key

Weblate uses SSH key to access remote repositories. The corresponding public key is found below, you can use it to grant Weblate access to a repository.

```
ssh-rsa
AAAAB3NzAC1yc2EAAAADAQABAAQAD0nFKdhteduFr+6x1UDLz0fAmXf/NMcUhhq/YBQN4z0M8tXiDS6MTrrJJjA+bvYpavSU9vHPuCOYd8DnXoIS5Eqshi+nC3tJypNK+1:
Weblate
```

### Commit signing

All commits made with Weblate are signed with the GPG key 8FB46C635F3B30B7D5751EABFC21A7391BE232FB, for which the corresponding public key is found below.

```
-----BEGIN PGP PUBLIC KEY BLOCK-----

mQGNBGOcNG0BDADrVjK3mM78AV1b+QO9iARuU21/ZxuQhDO0zecz/4mhHKxx22
SI2AethXGZHfBN/uPwf6E9Iqzh1TmOPg1VSyb+03hl1+VCq90EUOPZSNgTw0nhUS
n6aYgUm8cUP1EA4oftT5uEHW5a5FjrxkCv877CcEpPDhcHncizcWUNDdd158jqU
LnW6m0G4WwIsLj1SyWHcNMUAIDRURTTbYwgqEvz1ZgxQ5no8PLUb7o+n0fRpfqwp
sZ0vfVMn6FYQhZJGA/9o4vrXvhVwb1I2sNUeL2Pf+L7WSuSZr31VX5RI1KQxx8cn
RBEjIzZaQHf3HCchn5vWMtarfkqWBSHaKyHd3hwIzOu8Jc/UN2GFj+XshaY7y79u
nqPeXpFMOZBHWsj56basasE3EI80M+oHw21mOQ7BOQ/0vPWsJiVb3TDvC8Gec7RI
SMi435utuyHcnyYjvbrKZKSww+d7nYqwCNF+Rra/USLBtchRBN4eKxuqlj3gbEYq
X73d40YGz47KXfMAEQEAAbQdV2VibGF0ZSA8d2VibGF0ZUBleGFtcGxlLmNvbT6J
Ac4EEwEKADgWlQSPtGxjXzswt9V1Hqv8Iac5G+ly+wUCY5w0bQlBawULCQgHAgVv
CgklCwIEFgIDAQIeAQIXgAAKCRD8Iac5G+ly+4hQC/oD9dH58OQBZuLcJqf84p4q
4nnXjDA5Am+dZ1QY4rzN8KQROwXfV7bqml+ewjeEC1a+0DjKy/Ki+IA1iezmru
-----
```

Powered by Weblate 4.15

[About Weblate](#)
[Legal](#)
[Contact](#)
[Documentation](#)
[Donate to Weblate](#)

2. Alternatively you can also import existing keys into Weblate, just set `HOME=$DATA_DIR/home` when invoking `gpg`.

也參考:

`WEBLATE_GPG_IDENTITY`

## 2.21.7 頻次限制

在 3.2 版本變更: 頻次限制先擇接受更細粒度的配置。

在 4.6 版本變更: 速率限制不再適用於超級用。

Weblate 的一些操作受到頻次限制。在 `RATELIMIT_WINDOW` 的秒數最多允許 `RATELIMIT_ATTEMPTS` 次數的嘗試。然後阻止用 `RATELIMIT_LOCKOUT` 時間。還有指定範圍的設置，例如 `RATELIMIT_CONTACT_ATTEMPTS` 或 `RATELIMIT_TRANSLATE_ATTEMPTS`。下面的表格是可用範圍的完整列表。

後面的操作受到頻次限制:

名徵	範圍	允許的嘗試	頻次限制窗口	鎖定時間
<code>FF</code>	REGISTRATION	5	300	600
將消息發送給管理員	MESSAGE	2	300	600
Password authentication on sign in	LOGIN	5	300	600
網站範圍的搜索	SEARCH	6	60	60
Translating	TRANSLATE	30	60	600
Adding to glossary	GLOSSARY	30	60	600
Starting translation into a new language	LANGUAGE	2	300	600
Creating new project	PROJECT	5	600	600

If a user fails to sign in `AUTH_LOCK_ATTEMPTS` times, password authentication will be turned off on the account until having gone through the process of having its password reset.

這個設定也可被使用在 Docker 容器中，透過添加 `WEBLATE_` 前綴字在設定名稱前。如：`RATELIMIT_ATTEMPTS` 可使用 `WEBLATE_RATELIMIT_ATTEMPTS`。

API 具有另外的速率限制設置，請參見 [API 頻次限制](#)。

也參考：

頻次限制, 在反向代理後面運行, [API 頻次限制](#)

## 2.21.8 Fedora Messaging integration

Fedora Messaging 是基於 AMQP 的發布者，用於 Weblate 中發生的所有變化。您可以在使用此操作中正在發生的更改的其他服務。

Fedora Messaging Integration 可作單獨的 Python 模塊 “weblate-fedora-messaging”。請參見 [https://github.com/weblateorg/fedora\\_messaging/](https://github.com/weblateorg/fedora_messaging/) 用於設置。

## 2.22 定制 Weblate

使用 Django 和 Python 擴展與定制。將你的更改貢獻給上游，使每人都能受益。這降低了你的維護成本；Weblate 中的代碼對更改界面或重構編碼時的情況。

**警告：** 界面與模板都不被認為是穩定的 API。請對每次升級都查自己的定制，接口或其語義可能未經通知就進行更改。

也參考：

[Weblate 做貢獻](#)

### 2.22.1 建立 Python 模塊

如果不熟悉 Python，你可以查看 [Python For Beginners](#)，它解釋了其基本內容指向了高級教程。

To write a file with custom Python code (called a module), a place to store it is needed, either in the system path (usually something like `/usr/lib/python3.9/site-packages/`) or in the Weblate directory, which is also added to the interpreter search path.

在 3.8-5 版本新加入：When *using Docker*, you can place Python modules in `/app/data/python/` (see [Docker container volumes](#)), so they can be loaded by Weblate, for example from a *settings override file*.

更好地是，將你的定制化轉變為適當的 Python 包：

1. 在你的包建立文件夾（我們會使用 `weblate_customization`）。
2. 在包內新建 `setup.py` 文件來描述包：

```
from setuptools import setup

setup(
    name="weblate_customization",
    version="0.0.1",
    author="Your name",
    author_email="yourname@example.com",
    description="Sample Custom check for Weblate.",
    license="GPLv3+",
```

(繼續下一頁)



(繼續上一頁)

```
keywords="Weblate check example",
packages=["weblate_customization"],
)
```

3. 建立定制代碼的 Python 模塊（也被成 `weblate_customization`）的文件夾。
4. 在該面建立 `__init__.py` 文件來確認 Python 可以導入模塊。
5. 現在可以使用 `pip install -e` 安裝這個包。更多信息可以在 [Editable installs](#) 中找到。
6. 模塊一旦安裝，就可以用在 Weblate 配置中（例如 `weblate_customization.checks.FooCheck`）。

Your package structure should look like this:

```
weblate_customization
├── setup.py
└── weblate_customization
    ├── __init__.py
    ├── addons.py
    └── checks.py
```

可以在 <https://github.com/WeblateOrg/customize-example> 找到定制 Weblate 的例子，它涵蓋了下面描述的所有題目。

### 2.22.2 Changing the logo

1. 建立簡單的 Django app 來包含想要覆蓋的圖態文件（請參見[建立 Python 模塊](#)）。

品牌出現在後面的文件中：

**icons/weblate.svg**

導航條中顯示的 Logo。

**logo-\*.png**

根據屏幕分辨率和 web 瀏覽器的 Web 圖標。

**favicon.ico**

傳統瀏覽器使用的 Web 圖標。

**weblate-\*.png**

機器人或匿名用戶使用的頭像。一些 Web 瀏覽器使用這些作快捷圖標。

**email-logo.png**

在通知電子郵件中使用。

2. Add it to `INSTALLED_APPS`:

```
INSTALLED_APPS = (
    # Add your customization as first
    "weblate_customization",
    # Weblate apps are here...
)
```

3. 運行 `weblate collectstatic --noinput`，來收集提供給客戶端的圖態文件。

也參考：

[How to manage static files \(e.g. images, JavaScript, CSS\)](#), [Serving static files](#)

### 2.22.3 Custom quality checks, add-ons and auto-fixes

要在 Weblate 中安裝你的 *Custom automatic fixups*, *Writing own checks* 或 *Writing add-on* 代碼:

1. 將文件放進你的包含 Weblate 定制的 Python 模塊中 (請參見 [建立 Python 模塊](#))。
2. 在專用設置 (`WEBLATE_ADDONS`、`CHECK_LIST` 或 `AUTOFIX_LIST`) 中將其完全合法的路徑添加到 Python 類中:

```
# Checks
CHECK_LIST += ("weblate_customization.checks.FooCheck",)

# Autofixes
AUTOFIX_LIST += ("weblate_customization.autofix.FooFixer",)

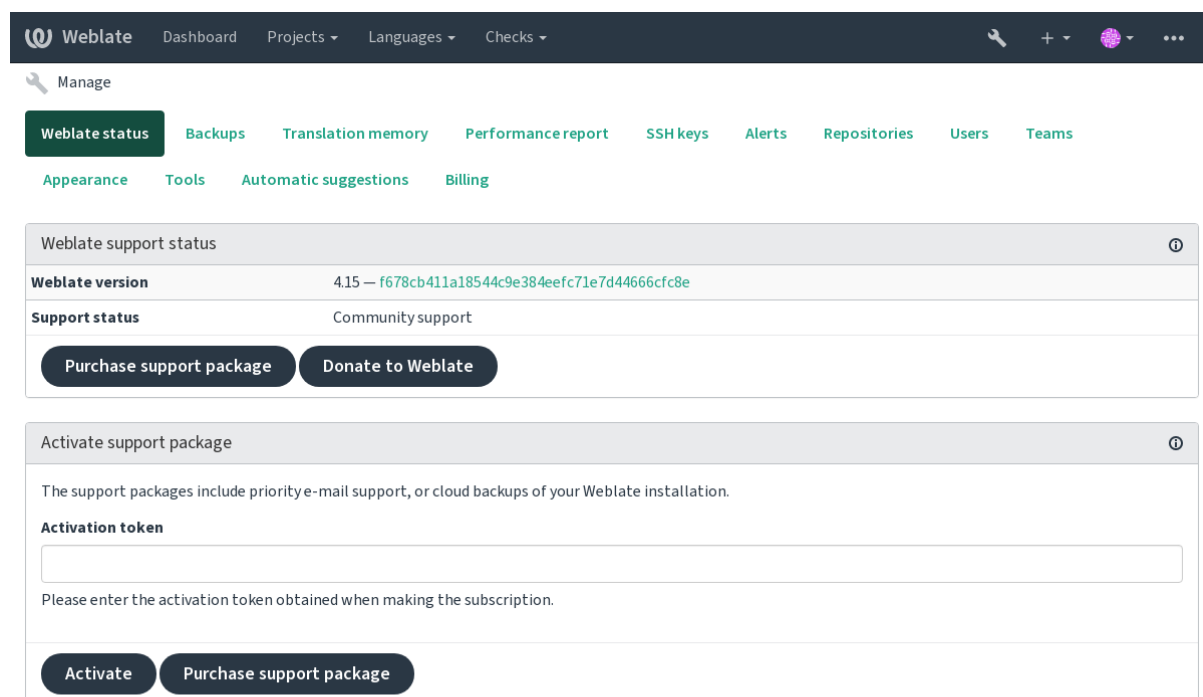
# Add-ons
WEBLATE_ADDONS += ("weblate_customization.addons.ExamplePreAddon",)
```

也參考:

*Custom automatic fixups*, *Writing own checks*, *Writing add-on*, *Executing scripts from add-on*

## 2.23 管理介面

管理界面在 `file:///manage/` 下面提供管理設置。它對於具有管理特權的登用是可行的，通過使用右上角的扳手圖標來訪問:



Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

它包括您 Weblate 的基本視圖:

- 支援狀態, 參閱從 [Weblate](#) 獲得支持
- 備份, 請參見 [備份和移動 Weblate](#)
- Shared translation memory, see [翻譯記憶](#)
- 性能報告, 來查 Weblate 的健康狀況和 Celery 隊列的長度

- SSH keys management, see [SSH 倉庫](#)
- 所有組件的警報概述，見 [alerts](#)

### 2.23.1 Django 管理界面

**警告：** 將來會刪除，因其應用困難——多數特性可以直接在 Weblate 中管理。

可以在這管理數據庫中存儲的對象，如用、翻譯和其他設置：

Webplate administration

WELCOME, WEPLATE TEST · RETURN TO WEPLATE / DOCUMENTATION / CHANGE PASSWORD / SIGN OUT

Site administration

REPORTS

Webplate support status

Status of repositories

SSH keys

Performance report

Translation memory

ACCOUNTS

Audit log entries

+ Add

Change

User profiles

+ Add

Change

Verified e-mails

+ Add

Change

AUTH TOKEN

Tokens

+ Add

Change

AUTHENTICATION

Groups

+ Add

Change

Roles

+ Add

Change

Users

+ Add

Change

BILLING

Billing plans

+ Add

Change

Customer billings

+ Add

Change

Invoices

+ Add

Change

FONTS

Font groups

+ Add

Change

Fonts

+ Add

Change

LEGAL

TOS agreements

+ Add

Change

PYTHON SOCIAL AUTH

Associations

+ Add

Change

Nonces

+ Add

Change

User social auths

+ Add

Change

SCREENSHOTS

Screenshots

+ Add

Change

TRANSLATION MEMORY

Translation memory entries

+ Add

Change

WEPLATE CONFIGURATION

Settings

+ Add

Change

WEPLATE LANGUAGES

Languages

+ Add

Change

WEPLATE TRANSLATIONS

Announcements

+ Add

Change

Component lists

+ Add

Change

Components

+ Add

Change

Contributor agreements

+ Add

Change

Projects

+ Add

Change

Recent actions

My actions

None available

在 *Reports* 部分，可以檢查網站的狀態，[生成設置](#) 進行調整，或者管理用於訪問的 SSH 密鑰[訪問存儲庫](#)。

管理任意部分下的數據庫對象。最有趣的也許是 *Weblate translations*，你可以在這[管理可翻譯的項目](#)，請參見[項目配置](#) 和 [組件配置](#)。

*Weblate languages* 保持語言定義，在[語言定義](#) 中進一步解釋。

## 加入一個專案

添加項目作[所有組件](#)的容器。通常可以[一部分軟件或圖書](#)（各自參數的信息請參見[項目配置](#)）來建立一個項目：

Weblate administration

WELCOME, **WEBLATE TEST**. [RETURN TO WEBLATE](#) / [DOCUMENTATION](#) / [CHANGE PASSWORD](#) / [SIGN OUT](#)

[Home](#) · [Weblate translations](#) · [Projects](#) · [Add Project](#)

Add Project

Required fields are marked in bold.

Project name:

WebplateOrg

Display name

URL slug:

weblateorg

Name used in URLs and filenames.

Project website:

https://weblate.org/

Main website of translated project.

Translation instructions:

https://weblate.org/contribute/

You can use Markdown and mention users by @username.

☒ Set "Language-Team" header

Lets Weblate update the "Language-Team" file header of your project.

☒ Use shared translation memory

Uses the pool of shared translations between projects.

☒ Contribute to shared translation memory

Contributes to the pool of shared translations between projects.

Access control:

Protected

How to restrict access to this project is detailed in the documentation.

☐ Enable reviews

Requires dedicated reviewers to approve translations.

☐ Enable source reviews

Requires dedicated reviewers to approve source strings.

☒ Enable hooks

Whether to allow updating this repository by remote hooks.

Language aliases:

Comma-separated list of language code mappings, for example: en\_GB:en,en\_US:en

Machinery settings:

Save and add another

Save and continue editing

SAVE

也參考:  
項目配置

398

Chapter 2. 管理員文檔

## 雙語言組件

一旦添加了一個項目，就可以添加翻譯組件了。（關於各自參數的信息，請參見[組件配置](#)）：

[illegible]



**也參考:**

[組件配置](#), [雙語](#)和[單語格式](#)

**單語言組件**

為了使這些翻譯更容易，提供了模板文件，包含了各自源語言的對應信息 ID（通常 英語）。（對於各自參數的信息，請參見[組件配置](#)）：

Webiate administration

HELLO! WEBLATE TEST RETURN TO WEBLATE, DOCUMENTATION, CHANGE PASSWORD, LOG OUT

Home

Recent translations

Components

100 components

Add Component

CREATE NEW TRANSLATION COMPONENT

Required fields are marked as bold.

Component name

Android

Display name

URL slug

android

Name used in URLs and filenames

Project

Webiate.org

▼

Version control system

Git

▼

Repository control system used to manage your repository containing translations. You can also choose additional integration with third party providers to address merge requests.

Source code repository

webiate/webiate-language-servers

URL of a repository your webiate component should clone in with other components

Repository push URL

URL of a push repository pointing to source of a merge

Repository browser

Link to repository browser, use [branch] for branch, [filename] and [diff] as filename and file placeholders. You might want to strip leading directory by using [filename:parent].

Repository (pull)

URL of repository where others can fetch changes from Webiate

Source stringing bug reporting address

Email address for reports on errors in source strings. Leave empty for no emails.

Repository branch

Repository branch to translate

Push branch

Branch for pushing changes, leave empty to use repository branch

File mask

app/src/main/res/values-\*/strings.xml

Part of files to translate relative to repository root, use \* instead of language code, for example post\* or locale/\*LC\_MESSAGES/strings.xml

Metadata based language file

app/src/main/res/values-\*/strings.xml

Metadata of translation base file, containing all strings and their sources. It is recommended for metadata based translation formats.

Get base file

☒

Source users will be able to edit the base file for metadata based translations.

Intermediate language file

Metadata of intermediate translation file. In most cases this is a translation file provided by developers and is used when creating actual source strings.

Template for new translations

Metadata of this used for creating new translations. For getting choose per file.

File format

Android String Resource

▼

☐ Locked

Locked components will not get any translation updates.

☒ Allow translation propagation

Whether translation updates in other components will cause automatic translation in this one.

☒ Use on our suggestions

Enable to allow translation suggestions at all.

☐ Suggestion voting

Users can vote on suggestions and mark them as translations.

Automatic suggestions

☒

Automatically suggest suggestions with this number of votes, use 0 to turn it off.

Translation flags

Additional comment separator flag to influence Webiate behavior

Enhanced checks

☒

Use of checks which can not be passed

Translation license

MIT License

▼

Contributor agreement

User agreement which needs to be approved before a user can translate this component.

Add new translation

Create new language file

▼

Press to create new translation for existing component.

Language code style

Default based on the file format

▼

Customize language code used to generate the filename for translations created by Webiate.

☐ Manage strings

Manage strings and allowing strings change from Webiate. If your strings are extracted from the source code or managed externally you probably want to keep it disabled.

Merge style

Rollup

▼

Define whether Webiate should merge the upstream repository or release changes into it.

Control message when translating

Translated using Webiate ({{ language\_name }})

Currently translated at {{ state.translated\_percent }}% ({{ state.translated }} of {{ state.all }} strings)

Translation ({{ project\_name }}) ({{ component\_name }})

Translatable ({{ project\_name }}) ({{ component\_name }})

Translatable ({{ project\_name }}) ({{ component\_name }})

You can use translation language for various info, please consult the documentation for more details.

Control message when adding translation

Add translation using Webiate ({{ language\_name }})

You can use translation language for various info, please consult the documentation for more details.

Control message when merging translation

Delete translation using Webiate ({{ language\_name }})

You can use translation language for various info, please consult the documentation for more details.

Control message when merging translation

Merge branch ({{ component\_name\_branch }}) into Webiate

You can use translation language for various info, please consult the documentation for more details.

Control message when add-on makes a change

Update translation files

Update by {{ author\_name }} took in Webiate.

Translation ({{ project\_name }}) ({{ component\_name }})

Translatable ({{ project\_name }}) ({{ component\_name }})

You can use translation language for various info, please consult the documentation for more details.

Merge request message

Translations update from ({{ url\_repo }})

Translations update from ({{ url\_repo }}) ({{ url\_repo }}) for ({{ project\_name }}) ({{ component\_name }}) ({{ url\_repo }})

For ({{ component\_name }}) ({{ url\_repo }})

For ({{ component\_name }}) ({{ url\_repo }})

For ({{ component\_name }}) ({{ url\_repo }})

For ({{ component\_name }}) ({{ url\_repo }})

You can use translation language for various info, please consult the documentation for more details.

☒ Push on commit

Push on commit: The repository should be pushed to GitHub on every commit.

Age of changes to commit

24

▼

Time in hours after which any pending changes will be committed to the VCS.

☒ Push on save

Push on save: The repository should be pushed on repository save.

Source language

English

▼

Language used for source strings in all components.

Language filter

\*/-/\*

Regular expression used to filter translation files when searching for the mask.

Variants regular expression

Regular expression used to determine variants of a string.

Priority

Medium

▼

Components with higher priority are affected first in translations.

☐ Restricted component

Restrict access to this component to only those explicitly given permission.

Show in projects

Webiate.org

▼

Choose additional projects where this component will be listed. Add one "Test" or "Training" or a file, to reflect more than one.

☐ Use as a gallery

Glancey color

Silver

▼

Remote version

Local version

Save and add source

Save and update strings

Save

也參考：

組件配置, 雙語和單語格式

## 2.24 從 Weblate 獲得支持

Weblate 擁有非盈利版權的開源軟件，由社區支持。訂購者接受優先支持而無需額外的費用。預付方式可以使軟件包被每個人獲得。您可以在 <https://weblate.org/support/>，找到關於當前提供的支持的更多信息。

### 2.24.1 集成支持

在 3.8 版本新加入。

購買的支持包可以可選地集成在您的 Weblate 的 ‘subscription management <https://weblate.org/user/>’ 界面中，您可以在那找到它的鏈接。關於您的安裝的基本事例的細節，也會以這種方式反饋給 Weblate。

Manage

Weblate status Backups Translation memory Performance report SSH keys Alerts Repositories Users Teams

Appearance Tools Automatic suggestions Billing

Weblate support status ⓘ

Weblate version 4.15 — f678cb411a18544c9e384eefc71e7d44666cfc8e

Support status Community support

Purchase support package Donate to Weblate

Activate support package ⓘ

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

Activation token

Please enter the activation token obtained when making the subscription.

Activate Purchase support package

Powered by Weblate 4.15 About Weblate Legal Contact Documentation Donate to Weblate

### 2.24.2 提交給 Weblate 的數據

- 配置 Weblate 事件的 URL
- Your site title
- 您運行的 Weblate 版本
- 您 Weblate 數據庫中一些對象的記號（項目、組件、語言、源字符串和用）
- 您事件的 SSH 公鑰

此外，何時：參考：“發現-Weblate”已開：

- 公共項目列表（名稱，URL 和網站）

有提交其它數據。

### 2.24.3 集成服務

- 查看您的支持包是否合法
- *Weblate provisioned backup storage*
- 探索 *Weblate*

**提示：**購買的支持包在購買時已經激活，且不必集成它們就可以使用。

### 2.24.4 探索 Weblate

在 4.5.2 版本新加入。

**備註：** This feature is currently in early beta.

Discover Web2Rate 是一個選項服務，使用更容易查找 WebLare 服務器和社區。用可以在 <https://weblate.org/discover/> 上覽服務，查找有項目貢獻。

### Getting listed

**提示：**參與發現 WeBlare 使 WeBlare 提交一些有關您服務器的信息，請參閱：REF：“支持數據”。

要使用活動支持訂列出您的服務器（請參閱：REF：“激活 - 支持”）在“發現 WeBlare”中，您需要執行此操作，請在“管理”面板中：

The screenshot shows the Weblate management interface. The top navigation bar includes 'Weblate', 'Dashboard', 'Projects', 'Languages', and 'Checks'. Below this is a 'Manage' section with various tabs: 'Weblate status', 'Backups', 'Translation memory', 'Performance report', 'SSH keys', 'Alerts', 'Repositories', 'Users', 'Teams', 'Appearance', 'Tools', 'Automatic suggestions', and 'Billing'. The 'Weblate status' tab is active, displaying the following information:

- Weblate support status** (with an info icon)
- Weblate version**: 4.15 — f678cb411a18544c9e384eefc71e7d44666cfc8e
- Support status**: Community support
- Discover Weblate**: Your Weblate is not listed on weblate.org. A button 'Enable discovery' is present, and a link 'Browse discovery' is available.
- At the bottom, there are three buttons: 'Manage support package', 'Purchase support package', and 'Donate to Weblate'.

Below the 'Weblate status' section, there is another section titled 'Activate support package' (with an info icon). It contains the text: 'The support packages include priority e-mail support, or cloud backups of your Weblate installation.' Underneath, there is a section for 'Activation token' with a text input field and a note: 'Please enter the activation token obtained when making the subscription.' At the bottom of this section are two buttons: 'Activate' and 'Purchase support package'.

列出您的服務器，有 Socation WebBlate 中的支持訂：

1. Register yourself at <<https://weblate.org/user/>>
2. Register your Weblate server in the discovery database at <<https://weblate.org/subscription/discovery/>>
3. 確認 WebBlate 中的服務激活，然後使用以下方式在 Web20 管理頁面中的發現列表：Guilabel: “用發現” 按鈕：

Webblate status

**Weblate version** 4.15 — f678cb411a18544c9e384eefc71e7d44666cfc8e

**Support status** Community support

**Discover Weblate** Your Weblate is not listed on weblate.org [Browse discovery](#)

[Enable discovery](#)

[Manage support package](#) [Purchase support package](#) [Donate to Weblate](#)

**Activate support package**

The support packages include priority e-mail support, or cloud backups of your Weblate installation.

**Activation token**

Please enter the activation token obtained when making the subscription.

[Activate](#) [Purchase support package](#)

Powered by Weblate 4.15 [About Weblate](#) [Legal](#) [Contact](#) [Documentation](#) [Donate to Weblate](#)

## Customizing listing

您可以通過在 <<https://weblate.org/user/>> 上提供文本和圖像（570 x 260 像素）來自定義列表。

## 2.25 Legal documents

備：在此，您將找到各種法律信息，您可能需要在某些法律司法管轄區中運營 Weblate。它作指導手段提供，有任何準確性或正確性的保證。最終是您有責任確保您使用 Weblate 符合所有適用的法律法規。

### 2.25.1 Licensing compliance

Weblate comes with [REUSE 3.0](#) compliant license specification.

### 2.25.2 itar 和其他出口管制

Weblate 可以在您自己的數據中心或擬私有云中運行。因此，它可用於存儲 ITAR 或其他導出控制信息，但是，最終用負責確保這種合規性。

管的 Weblate 服務尚未遵守符合 ITAR 或其他出口管制，目前尚未提供限制國家/地區訪問權限的能力。

### 2.25.3 US encryption controls

Weblate 不包含任何加密代碼，但可能是使用加密，數據 Integrity 和 Confidentiality 使用加密的第三方組件的主題導出控件。

Weblate 最可能將被分類 ECCN 5D002 或 5D992，且作公開可用的 Libre 軟件，它不應該受到耳朵（請參“加密項目不受耳朵的影響 <<https://www.bis.doc.gov/index.php/策略-指導/加密/1-encryption-item-not-project-the-ear>>”）。

Weblate 使用的軟件組件（僅列出與加密功能相關的組件）：

#### Python

See [https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is\\_Python\\_subject\\_to\\_export\\_laws.3F](https://wiki.python.org/moin/PythonSoftwareFoundationLicenseFaq#Is_Python_subject_to_export_laws.3F)

#### GnuPG

Optionally used by Weblate

#### Git

Optionally used by Weblate

#### curl

Used by Git

#### OpenSSL

用 python 和 curl 使用

加密密鑰的度取於 Weblate 和 In Interact 的第三方組件的配置，但在任何體面設置中它將包括所有導出受限加密功能：

- 對稱算法超過 56 位
- 非對稱算法超過 512 位的整數的分解
- 用於非對稱算法的大於 512 比特的有限場的乘法群中的離散對數的計算
- 用於非對稱算法的組中的分立對數超過 112 比特的組中的離散對數

Weblate 有任何加密激活功能，但它可以不涉及加密代碼的方式配置。加密功能包括：

- 使用安全傳輸協議（HTTPS）訪問遠程服務器
- 代碼的提交生成簽名（PGP）

也參考：

“出口控制（耳朵）開源軟件 <<https://www.magicplat.com/blog/ear/>>”

## 3.1 Weblate 做貢獻

有幾十種方法來改進 Weblate。你可以選擇一個你覺得舒服的方式，它可以是編碼、圖形設計、文檔、贊助，或一個想法：

- 在 *Weblate* 中匯報問題
- 開始  *Weblate* 貢獻代碼
- *Contributing to Weblate modules*
- 翻譯 *Weblate*
- *Contribute to Weblate documentation*
- *Weblate discussions*
-  *Weblate* 開發提供資金

### 3.1.1 翻譯 Weblate

Weblate 持續使用 Weblate 自身進行 ‘translated <<https://hosted.weblate.org/>>’。盡你所能，使 Weblate 能以盡可能多的人類語言提供。它讓 Weblate 更接近它的用 ！

如果您在源字符串中找到可能的錯誤，則可以將其標記  Web2 編輯器中的  釋。這樣，可以討論和糾正它。如果您確定了，您也可以單擊以下鏈接：Guilabel: “源字符串位置” 部分， 以修正提交 PR。

### 3.1.2 Contribute to Weblate documentation

歡迎您來改進您選擇的文檔頁面。點擊符號： `guilabel`：‘編輯在頁面的右上角的 `github`’按鈕上輕鬆進行。  
請在寫作時尊重這些指導方針：

1. 如果有效，請勿刪除部分文檔。
2. 使用清晰易懂的語言。你正在寫科技作家，而不是一首詩。非所有 Docs 讀者都是母語人士，很沉思。
3. Don't be afraid to ask if you are not certain. If you have to ask about some feature while editing, don't change its docs before you have the answer. This means: You change or ask. Don't do both at the same time.
4. 通過在遵循文檔時執行描述的操作來驗證您的更改。
5. 將 PR 與小塊的更改發送，以便更容易，更快地查看和合。
6. 如果要重寫更改大文章的結構，請分兩步執行以下操作：
  1. Rewrite
  2. 一旦重寫，審核，光和合，改變另一條公關的段落的結構。

---

**提示：** 您可以使用 docs <https://hosted.weblate.org/projects/weblate/documentation/>’\_ 翻譯。

---

### 3.1.3 Extending built-in language definitions

語言定義是在 ‘webleate-language-data 存儲庫中 <https://github.com/weblateorg/language-data/>’\_。

You are welcome to add missing language definitions to `languages.csv`, other files are generated from that file.

### 3.1.4 Weblate discussions

如果您有一個想法，不確定它是否適合問題，請不要擔心。您可以加入 “Github 討論 <https://github.com/weblateorg/weblate/discussions>’\_ 中的社區 >’\_。

### 3.1.5 Weblate 開發提供資金

您可以在 `donate page`’\_ 上助推 Weblate 的開發。收集到的資金用於免費管自由軟件項目和 Weblate 的進一步開發。籌款目標和資助者的回報等選項，請查看 ‘donate page’。

#### Supporters who have funded Weblate

Weblate 支持者列表：

- Yashiro Ccs
- Cheng-χ at 僧
- Timon Reinhard
- Cassidy James
- Loic Dachary
- Marozed
- <https://freedombox.org/>
- GNU Solidario (GNU Health)



- [BallotReady](#)
- [Richard Nespithal](#)
- [MyExpenses.Mobi](#)
- [Michael Totschnig](#)

想要加入列表中嗎？請查看 [Donate to Weblate](#) 上的選項。

## 3.2 開始 F Weblate 貢獻代碼

要理解 Weblate 源代碼，請查看 [Weblate 源代碼](#)、[Weblate 前端](#) 和 [Weblate F 部](#)。

### 3.2.1 Starting with the codebase

要讓你自己熟悉 Weblate 代碼庫，那 F 請查看標記 F [good first issue](#) 的那些 bug。

You are welcome to start working on these issues without asking. Just announce that in the issue, so that it's clear that somebody is working on that issue.

### 3.2.2 本地運行 Weblate

開始 Weblate 開發的最舒適的方法是按照 [Installing from sources](#)。它將給您一個帶有可編輯的 Weblate 源代碼的 F 擬環境。

1. Clone the Weblate source code:

```
git clone https://github.com/WeblateOrg/weblate.git
cd weblate
```

2. Create a virtualenv:

```
virtualenv .venv
.venv/bin/activate
```

3. 安裝 Weblate（這需要一些系統依賴，見 [Installing from sources](#)）:

```
pip install -e .
```

3. 安裝用於開發的所有依賴性:

```
pip install -r requirements-dev.txt
```

4. Start a development server:

```
weblate runserver
```

5. 取 F 於配置，你也許想 F 動 Celery workers:

```
./weblate/examples/celery start
```

6. 要運行測試（更多細節見 [本地測試](#)）:

```
. scripts/test-database
./manage.py test
```

也參考:

[Installing from sources](#)

### 3.2.3 在 Docker 中本地運行 Weblate

如果你已經安裝了 Docker 和 Docker -compose，你就可以動開發環境，只需運行：

```
./rundev.sh
```

它將新建 Docker 影響動它。Weblate 運行在 `<http://127.0.0.1:8080/>` 上，且你可以以 `admin` 用名，`admin` 密碼來登。新的安裝是空的，所以您會想要以添加翻譯項目和組件來繼續。

對此，`Dockerfile` 和 `docker-compose.yml` 位於本地的 `dev-docker` 目中。

The script also accepts some parameters, to execute tests, run it with the `test` parameter and then specify any `test` parameters, for example running only tests in the `weblate.machine` module:

```
./rundev.sh test --failfast weblate.machine
```

---

**備：** 小心在運行測試前您的 Docker 容易活動運行。您可以通過運行 `docker ps` 命令來檢查。

---

To display the logs:

```
./rundev.sh logs
```

了停止後台容器，運行：

```
./rundev.sh stop
```

運行有任何參數的本將重建 Docker 容器重動它。

---

**備：** 這不是用於生的合適設置，因它包括幾個不安全的小技巧，但它們會使開發更容易。

---

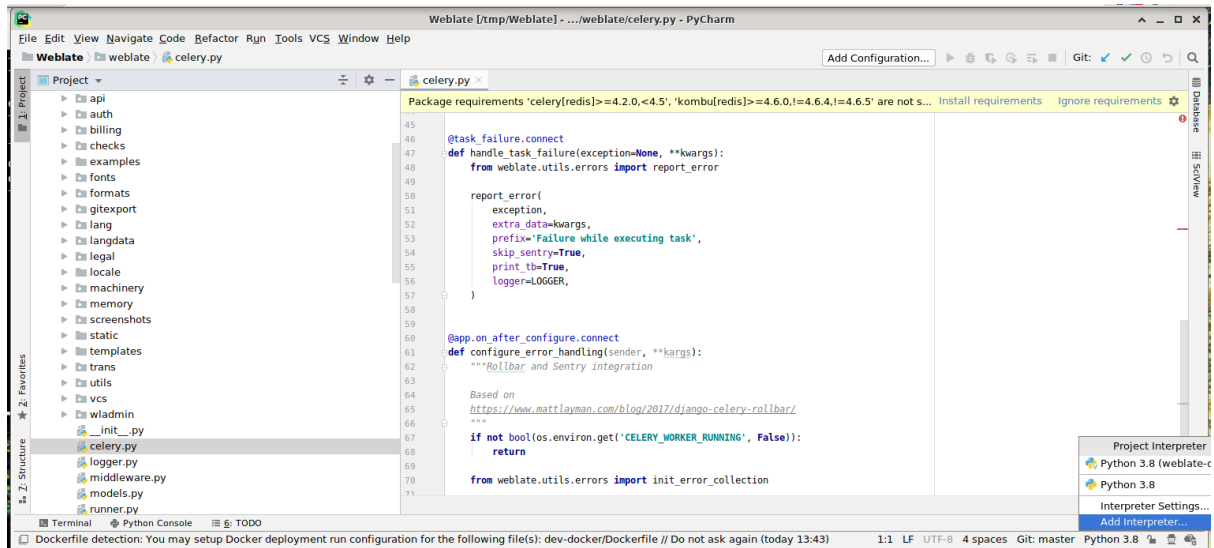
### 3.2.4 Bootstrapping your devel instance

您會想要使用 `import_demo` 來新建演示翻譯，且使用 `createadmin` 來創建一名管理用。

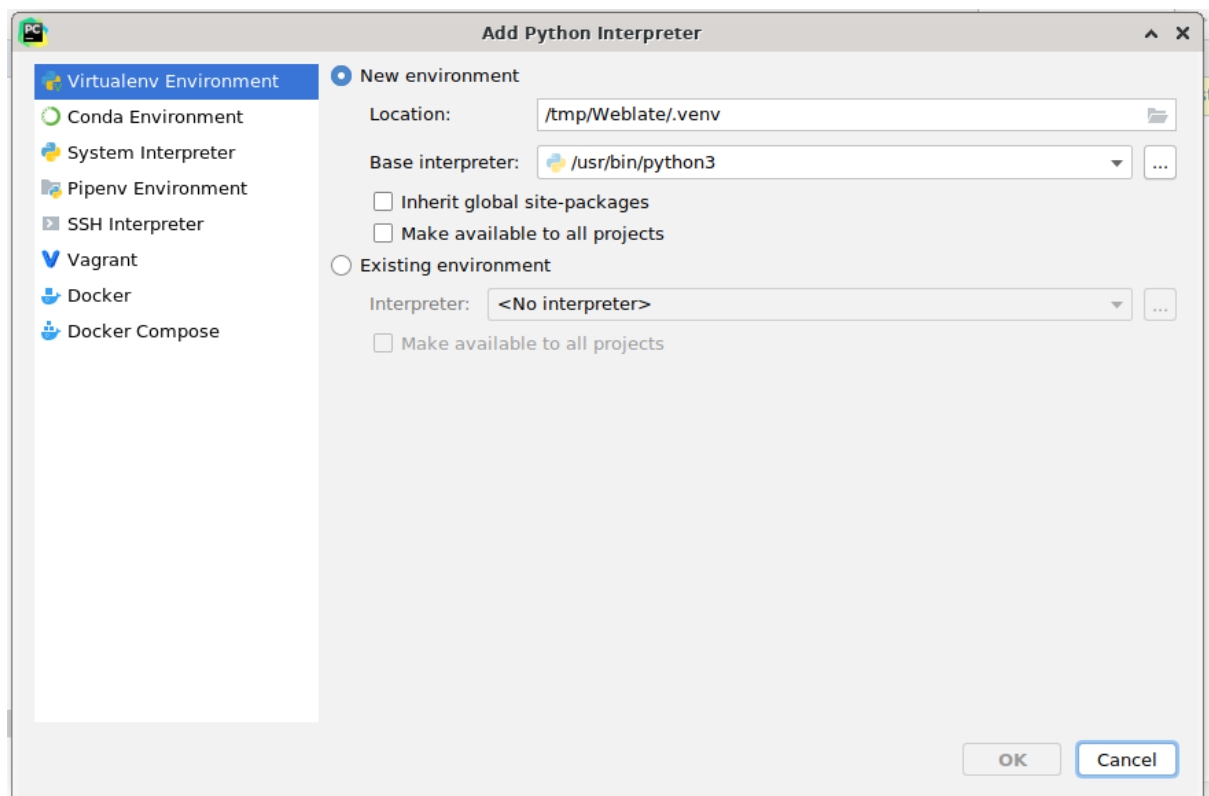
### 3.2.5 使用 PyCharm Weblate 編寫代碼

PyCharm 是 Python 的著名 IDE，這有一些幫助你在其中建立你的 Weblate 項目的指南。

考慮到你剛剛將 GitHub 存儲庫到一個文件夾中，只需使用 PyCharm 打開它。一旦 IDE 打開，第一步要做的是指定你想要使用的解釋器：

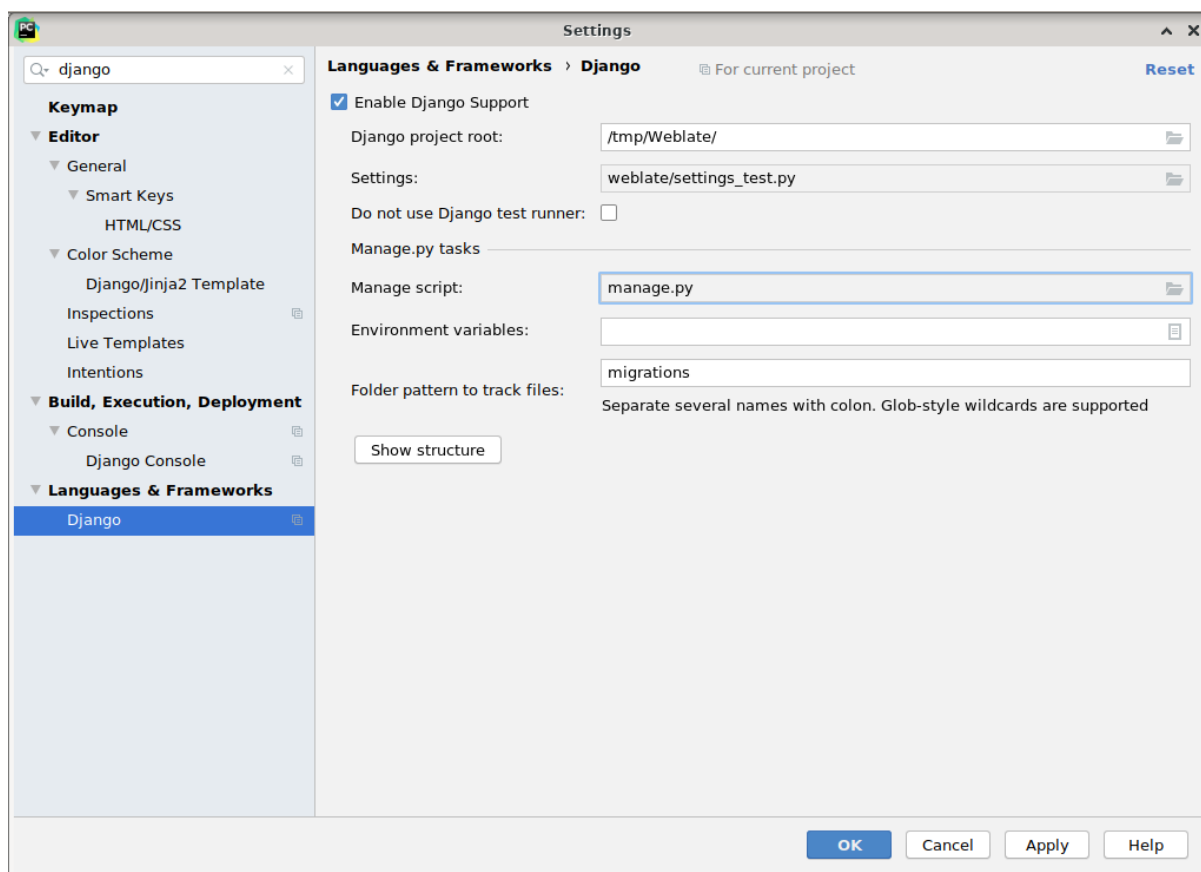


可以或者選擇讓 PyCharm 為您創建 virtualenv（擬環境），或者選擇已經存在的：



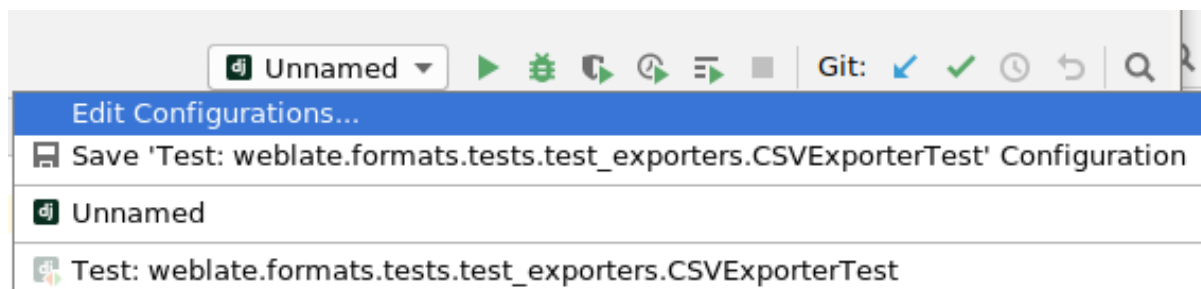
一旦設置了解釋程序不要忘記安裝依賴：要么通過控制台（IDE 的控制台默認情況下會直接使用你的 virtualenv），或者當你得到一個關於缺少依賴項的警告時通過接口。

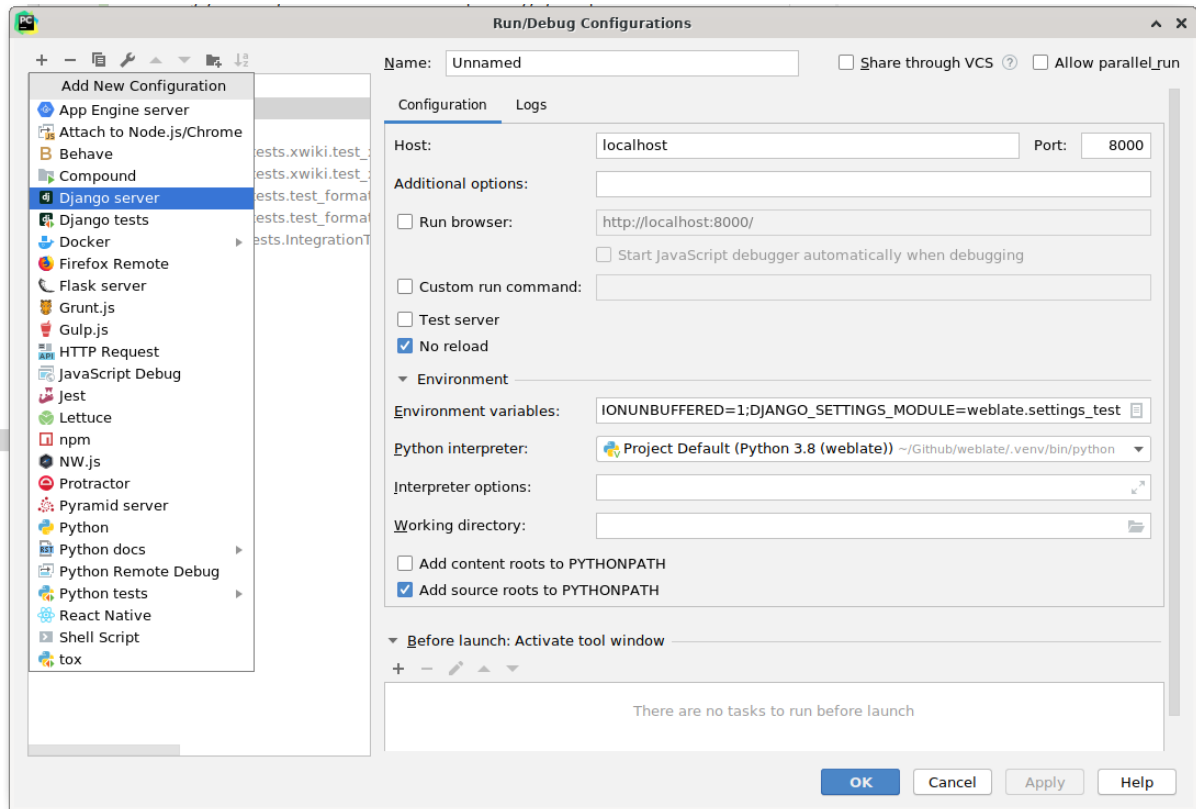
第二步是設置正確的信息來在 PyCharm 中原生使用 Django：理念是能立即觸發 IDE 中的單元測試。此，你需要指定該 Django 項目的根路徑及其設置路徑：



要小心，*Django project root*（*Dejango* 項目的根）是代碼庫的實際根目<sup>[F]</sup>，而不是 *Weblate* 子目<sup>[F]</sup>。關於設置，你可以使用來自代碼庫的 `file:'weblate/settings_test.py'`，你也可以創建自己的設置<sup>[F]</sup>在那<sup>[F]</sup>設置它。

最後一步是運行服務器，<sup>[F]</sup>將斷點放置在代碼中而能<sup>[F]</sup>調試它。這通過新建新的 *Django Server* 配置來完成：





提示：小心被稱`guiabel.No reload`的屬性：如果你修改文件，它會阻止服務器被實時重新加載。這允許保留現有的調試器斷點，而它們通常會在重新加載服務器時被`FF`。

## 3.3 Weblate 源代碼

Weblate 在 [GitHub](#) 上開發。歡迎您將代碼分叉`FF`打開拉去請求。同樣歡迎任何形式的補丁。

也參考：

查詢 [Weblate](#) `FF` 部 看看 Weblate 從`FF`部看是什`FF`樣子的。

### 3.3.1 Coding guidelines

編寫 Weblate 的任何代碼應該時刻記得‘Security by Design Principles’\_（由設計原理來提供安全性）。

Any code should come with documentation explaining the behavior. Don’ t forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize [PEP 484](#) type hints. We’ re not checking this in our CI yet as existing code does not yet include them.

### 3.3.2 Coding standard and linting the code

代碼應該符合 PEP-8 變成指南，[F](#)且應該使用 **black** 代碼格式化程序來格式化。

[F](#)了檢查代碼質量，可以使用 **flake8**，推薦的插件列在 `.pre-commit-config.yaml` 中，而其配置放置在 `setup.cfg` 中。

The easiest approach to enforce all this is to install [pre-commit](#). The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

還能[F](#)手動觸發檢查，來檢查所有文件的運行：

```
pre-commit run --all
```

## 3.4 調試 Weblate

Bugs can behave as application crashes or as various misbehavior. You are welcome to collect info on any such issue and submit it to the [issue tracker](#).

### 3.4.1 除錯模式

Turning on debug mode will make the exceptions show in the web browser. This is useful to debug issues in the web interface, but not suitable for a production environment because it has performance consequences and might leak private data.

In a production environment, use [ADMINS](#) to receive e-mails containing error reports, or configure error collection using a third-party service.

**也參考：**

*[Disable debug mode](#), [Properly configure admins](#), [Collecting error reports](#)*

### 3.4.2 Weblate [F](#)

Weblate can produce detailed logs of what is going on in the background. In the default configuration it uses syslog and that makes the log appear either in `/var/log/messages` or `/var/log/syslog` (depending on your syslog daemon configuration).

The Celery process (see [使用 Celery 的後台任務](#)) usually produces its own logs as well. The example system-wide setups logs to several files under `/var/log/celery/`.

Docker containers log to their output (as per usual in the Docker world), so you can look at the logs using `docker-compose logs`.

**也參考：**

配置的例子 包含 [LOGGING](#) 配置。

### 3.4.3 Not processing background tasks

A lot of things are done in the background by Celery workers. If things like sending out e-mails or component removal does not work, there might be a related issue.

Things to check in that case:

- Check that the Celery process is running, see [使用 Celery 的後台任務](#)
- Check the Celery queue status, either in [管理介面](#), or using `celery_queues`
- Look in the Celery logs for errors (see [Weblate 日誌](#))

### 3.4.4 不接收來自 Weblate 的電子郵件

You can verify whether outgoing e-mail is working correctly by using the `sendtestemail` management command (see [Invoking management commands](#) for instructions on how to invoke it in different environments) or by using [管理介面](#) under the *Tools* tab.

These send e-mails directly, so this verifies that your SMTP configuration is correct (see [Configuring outgoing e-mail](#)). Most of the e-mails from Weblate are however sent in the background and there might be some issues with Celery involved as well, please see [Not processing background tasks](#) for debugging that.

### 3.4.5 分析應用的崩潰

In case the application crashes, it is useful to collect as much info about the crash as possible. This can be achieved by using third-party services which can collect such info automatically. You can find info on how to set this up in [Collecting error reports](#).

### 3.4.6 無報告的故障

很多任務寫在 Celery 進行後台處理。故障不顯示在 [管理介面](#) 上，但出現在 Celery 的 [日誌](#) 中。配置 [Collecting error reports](#) 會幫助您更容易地 [注意](#) 到這樣的故障。

### 3.4.7 性能問題

In case Weblate performs badly in some scenario, please collect the relevant logs showing the issue, and anything that might help figuring out where the code might be improved.

如果有些請求在 [管理介面](#) 有任何提示的情況下花費了很長時間，你可能想要安裝 `dogslow` <<https://pypi.org/project/dogslow/>>，附加參數 `ref:collecting-errors` 在錯誤收集工具中獲取精確和詳細的回溯信息。

In case the slow performance is linked to the database, you can also enable logging of all database queries using following configuration after enabling `DEBUG`:

```
LOGGING["loggers"]["django.db.backends"] = {"handlers": ["console"], "level":
↪ "DEBUG" }
```

## 3.5 Weblate 部

---

**備：** 這一章將給出 Weblate 部的基本概。

---

Weblate 從 Django 得到其多數代碼架構，基於它。

### 3.5.1 目結構

Weblate 主倉庫目結構的速覽：

**docs**

本文檔的源碼，可使用 Sphinx 來構建。

**dev-docker**

運行開發服務器的 Docker 代碼，請參見在 Docker 中本地運行 Weblate。

**weblate**

Weblate Django 應用的的源代碼作，請參見 Weblate 部。

**weblate/static**

客端文件（CSS、Javascript 和圖片），請參見 Weblate 前端。

### 3.5.2 Modules

Weblate 包括幾個 Django 應用（一些是可選的，請參見 *Optional Weblate modules*）：

accounts

用賬、簡介和通知。

addons

微調 Weblate 行的附加組件，請參見:ref:“addons”。

api

基於 Django REST framework 的 API。

auth

認證和權限。

billing

可選的帳單 模塊。

checks

翻譯字符串質量檢查 模塊。

fonts

字體提供檢查模塊。

formats

基於 translate-toolkit 的文件格式抽象層。

gitexport

可選的Git 導出器 模塊。

lang

定義語言和復數模型的模塊。



legal

可選的法律模塊。

machinery

機器翻譯服務的集成。

memory

Built-in translation memory, see 翻譯記憶.

screenshots

屏幕截圖管理與 OCR 模塊。

trans

處理翻譯的主模塊。

utils

各種幫助功能。

vcs

版本控制系統抽象概念。

wladmin

Django 管理界面定制化。

## 3.6 Developing add-ons

附加元件 是在 Weblate 中自定義本地化工作流的方法。

**class** weblate.addons.base.BaseAddon (*storage=None*)

Weblate 附加組件的基礎類。

**classmethod** can\_install (*component, user*)

Check whether add-on is compatible with given component.

**configure** (*settings*)

Save configuration.

**daily** (*component*)

每日觸發 F 子。

**classmethod** get\_add\_form (*user, component, \*\*kwargs*)

Return configuration form for adding new add-on.

**get\_settings\_form** (*user, \*\*kwargs*)

Return configuration form for this add-on.

**post\_add** (*translation*)

Hook triggered after new translation is added.

**post\_commit** (*component*)

Hook triggered after changes are committed to the repository.

**post\_push** (*component*)

Hook triggered after repository is pushed upstream.

**post\_update** (*component, previous\_head: str, skip\_push: bool*)

在倉庫從上游更新之後觸發子。

#### 參數

- **previous\_head** (*str*) – 倉庫更新前的 HEAD，在初始克隆時可以是空白的。
- **skip\_push** (*bool*) – Whether the add-on operation should skip pushing changes upstream. Usually you can pass this to underlying methods as `commit_and_push` or `commit_pending`.

**pre\_commit** (*translation, author*)

Hook triggered before changes are committed to the repository.

**pre\_push** (*component*)

在倉庫推送上游之前觸發子。

**pre\_update** (*component*)

在倉庫從上游更新之前觸發子。

**save\_state** ()

Save add-on state information.

**store\_post\_load** (*translation, store*)

Hook triggered after a file is parsed.

它接收文件格式類的事件作參數。

這對修復該文件格式類參數有用，例如，調整文件如何存儲。

**unit\_pre\_create** (*unit*)

創建新的單元前觸發的子。

Here is an example add-on:

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

from django.utils.translation import gettext_lazy as _

from weblate.addons.base import BaseAddon
from weblate.addons.events import EVENT_PRE_COMMIT

class ExampleAddon(BaseAddon):
    # Filter for compatible components, every key is
    # matched against property of component
    compat = {"file_format": {"po", "po-mono"}}
    # List of events add-on should receive
    events = (EVENT_PRE_COMMIT,)
    # Add-on unique identifier
    name = "weblate.example.example"
    # Verbose name shown in the user interface
    verbose = _("Example add-on")
    # Detailed add-on description
    description = _("This add-on does nothing it is just an example.")

    # Callback to implement custom behavior
    def pre_commit(self, translation, author):
        return
```

## 3.7 Weblate 前端

前端當前使用 Bootstrap、jQuery 和一些第三方庫來構建。

### 3.7.1 Supported browsers

Weblate 支持所有主要瀏覽器和平臺的最新的、穩定的版本。

不明確支持使用最新版的 WebKit、Blink 或 Gecko 的替換瀏覽器，無論是否直接還是通過平台的 web 視圖 API。然而，Weblate 應該（在多數情況下）頁在這些瀏覽器中正常顯示工作。

其它瀏覽器也能工作，但一些特性會受到限制。

### 3.7.2 依賴性管理

The yarn package manager is used to update third party libraries. The configuration lives in `scripts/yarn` and there is a wrapper script `scripts/yarn-update` to upgrade the libraries, build them and copy to correct locations in `weblate/static/vendor`, where all third partly frontend code is located. The Weblate specific code should be placed directly in `weblate/static` or feature specific subdirectories (for example `weblate/static/editor`).

添加了新的第三方庫，典型包括：

```
# Add a yarn package
yarn --cwd scripts/yarn add PACKAGE
# Edit the script to copy package to the static folder
edit scripts/yarn-update
# Run the update script
./scripts/yarn-update
# Add files to git
git add .
```

### 3.7.3 編碼風格

Weblate 依賴於 Prettier 來進行 JavaScript 和 CSS 文件的代碼格式化。

我們還是用 ESLint 來檢查 JavaScript 代碼。

### 3.7.4 在地化

如果在前端代碼中需要任何用戶可見的文本，那麼應該將其本地化。在多數情況下，所有需要的是將文本打包到 `gettext` 函數內部，但也有更複雜的特性來使用：

```
document.write(gettext('this is to be translated'));

var object_count = 1 // or 0, or 2, or 3, ...
s = gettext('literal for the singular case',
            'literal for the plural case', object_count);

fmts = gettext('There is %s object. Remaining: %s',
               'There are %s objects. Remaining: %s', 11);
s = interpolate(fmts, [11, 20]);
// s is 'There are 11 objects. Remaining: 20'
```

也參考：

： doc: *django* 文檔中的翻譯主題 <django: 主題/i18n/translation>

### 3.7.5 Icons

Weblate 目前使用 material design 圖標。如果你想找新的符號，檢查 [Material Design Icons](#) 或 [Material Design Resources](#)。

此外，有 `scripts/optimize-svg` 來縮小 SVG 的大小，因為多數圖標嵌入在 HTML 中，而使路徑有風格。

## 3.8 在 Weblate 中匯報問題

Weblate [issue tracker](#) 位於 Github。

歡迎報告任何問題，或提出改進建議。我們準備了各種模板，可以輕鬆地指導你完成問題報告。

如果你在 Weblate 中發現了安全問題，請查看下方的 `:ref:security` 部分。

如果您不確定您的錯誤報告或功能請求，您可以嘗試：參考：“討論”。

### 3.8.1 安全性問題

為了給予社區實踐來響應升級，我們強烈敦促您私下匯報所有的安全問題。[HackerOne](#) 用於處理安全問題，而且而可以在 [HackerOne](#) 直接匯報。一旦你在那提交它，社區有有限但足夠的時間來解決事件。

另外，可以匯報給 [security@weblate.org](mailto:security@weblate.org)，最後也會到 [HackerOne](#) 上。

If you don't want to use [HackerOne](#), for whatever reason, you can send the report by e-mail to [michal@weblate.org](mailto:michal@weblate.org). You can choose to encrypt it using this PGP key `3CB 1DF1 EF12 CF2A C0EE 5A32 9C27 B313 42B7 511D`. You can also get the PGP key from [Keybase](#).

---

**備註：** Weblate 在很多事情上依賴於第三方組件。如果你發現一個影響這些組件的漏洞，請直接報告給相應的項目。

這些中的一些是：

- [Django](#)
  - [Django REST 框架](#)
  - [Python Social Auth](#)
- 

## 3.9 Weblate 測試套件與連續集成

測試套件存在於多數當前代碼，通過任何新的功能添加測試來擴大覆蓋範圍，並確認其正常工作。

### 3.9.1 連續集成

當前的測試結果可以在 [‘GitHub Actions <https://github.com/WeblateOrg/weblate/actions>’](https://github.com/WeblateOrg/weblate/actions) 上找到，並且覆蓋範圍在 [‘Codecov <https://codecov.io/github/WeblateOrg/weblate>’](https://codecov.io/github/WeblateOrg/weblate) 上報告。

有幾項工作來確認不同的方面：

- 單元測試
- 文檔構建與外部鏈接
- 來自所有支持的發布版本的合並測試
- 代碼整理

- 設置確認（確保生成的 dist 文件不失去任何內容，可以測試）

CI 的配置在 `.github/workflows` 目錄中。它重度使用了 `ci` 目錄中的幫助本。本還可以手動執行，但它們需要一些環境變量，多數用來定義 Django 設置要使用的文件和數據庫連接。它的示例定義在 `scripts/test-database` 中：

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

簡單的執行看起來可以像這樣：

```
. scripts/test-database
./ci/run-migrate
./ci/run-test
./ci/run-docs
```

### 3.9.2 本地測試

如果本地運行測試套件，要使用：

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test
```

**提示：** 您會需要使用數據庫（PostgreSQL）服務器來檢測。Django 默認新建另外的數據庫，以 `test_` 前綴來運行測試，因此在您的設置配置使用 `weblate` 的情況下，測試會使用 `test_weblate` 數據庫。設置的指示請參見 [Weblate 的數據庫設置](#)。

`weblate/settings_test.py` 也用在 CI 環境中（請參見[連續集成](#)），且可以使用環境變量調整：

```
# Copyright © Michal Čihař <michal@weblate.org>
#
# SPDX-License-Identifier: GPL-3.0-or-later

# Simple way to configure test database from environment

# Database backend to use postgresql / mysql / mariadb
export CI_DATABASE=${1:-postgresql}

# Database server configuration
export CI_DB_USER=weblate
export CI_DB_PASSWORD=weblate
export CI_DB_HOST=127.0.0.1

# Django settings module to use
export DJANGO_SETTINGS_MODULE=weblate.settings_test
```

運行測試前，應該收集狀態文件，因一些測試在出現時依賴於它們：

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py collectstatic
```

您也可以指定要運行的各自測試：

```
DJANGO_SETTINGS_MODULE=weblate.settings_test ./manage.py test weblate.gitexport
```

提示：測試也可以在開發者 docker 容器執行，請參見在 [Docker](#) 中本地運行 [Weblate](#)。

也參考：  
運行 Django 寫測試的更多信息請參見 [Testing in Django](#)。

### 3.10 數據架構

Weblate 使用 [JSON Schema](#) 來定義外部 JSON 文件的輸入。

#### 3.10.1 Weblate Translation Memory Schema

<a href="https://weblate.org/schemas/weblate-memory.schema.json">https://weblate.org/schemas/weblate-memory.schema.json</a>		
type	array	
items	The Translation Memory Item	
	type	object
	properties	
	• category	The String Category
		1 是全局的，2 是共享的，10000000+ 是項目特定的，20000000+ 使用特定的
	type	integer
	examples	1
	minimum	0
	default	1
	• origin	字符串源頭
		Filename or component name
	type	string
	examples	test.tmx
		project/component
	default	
	• source	The Source String
	type	string
	examples	Hello
	minLength	1
	default	
	• source_language	The Source Language
		ISO 639-1 / ISO 639-2 / IETF BCP 47
	type	string
	examples	en
	pattern	^[^ ]+\$
	default	
	• target	The Target String
	type	string
	examples	Ahoj
	minLength	1

繼續下一頁

表格 1 – 繼續上一頁

		default	
	• <b>target_language</b>	<i>The Target Language</i>	
		ISO 639-1 / ISO 639-2 / IETF BCP 47	
		type	<i>string</i>
		examples	cs
		pattern	^[^ ]+\$
		default	
	additionalProperties	False	
definitions			

也參考:

翻譯記憶, *dump\_memory*, *import\_memory*

### 3.10.2 Weblate user data export

https://weblate.org/schemas/weblate-userdata.schema.json			
type	object		
properties			
• 基本	Basic		
	type	object	
	properties		
	• username	Username	
		type	string
		examples	管理
		default	
	• full_name	Full name	
		type	string
		examples	Weblate Admin
		default	
	• email	E-mail	
		type	string
		examples	noreply@example.com
		default	
		format	電子信箱
	• date_joined	Date joined	
		type	string
		examples	2019-11-18T18:53:54.862Z
		default	
		format	date-time
	additionalProperties	False	
• 個人檔案	Profile		
	type	object	
	properties		
	• language	Language	
		type	string
		examples	cs
		pattern	^[^ ]*\$
		default	
	• suggested	Number of suggested strings	
		type	integer
		examples	1

繼續下一頁


表格 2 - 繼續上一頁

		default	0
	• <b>translated</b>	<i>Number of translated strings</i>	
		type	<i>integer</i>
		examples	24
		default	0
	• <b>uploaded</b>	<i>Number of uploaded screenshots</i>	
		type	<i>integer</i>
		examples	1
		default	0
	• <b>hide_completed</b>	<i>Hide completed translations on the dashboard</i>	
		type	<i>boolean</i>
		examples	False
		default	True
	• <b>secondary_in_zen</b>	在連續模式中顯示第二語言翻譯	
		type	<i>boolean</i>
		examples	True
		default	True
	• <b>hide_source_secondary</b>	<i>Hide source if a secondary translation exists</i>	
		type	<i>boolean</i>
		examples	False
		default	True
	• <b>editor_link</b>	<i>Editor link</i>	
		type	<i>string</i>
		examples	
		pattern	<i>^.*\$</i>
		default	
	• <b>translate_mode</b>	<i>Translation editor mode</i>	
		type	<i>integer</i>
		examples	0
		default	0
	• <b>zen_mode</b>	[F]心編輯模式	
		type	<i>integer</i>
		examples	0
		default	0
	• <b>special_chars</b>	<i>Special characters</i>	
		type	<i>string</i>
		examples	
		pattern	<i>^.*\$</i>
		default	
	• <b>dashboard_view</b>	<i>Default dashboard view</i>	
		type	<i>integer</i>
		examples	1
		default	0
	• <b>dashboard_component_list</b>	<i>Default component list</i>	
		default	null
		anyOf	type <i>null</i>
		type	<i>integer</i>
	• <b>語言</b>	<i>Translated languages</i>	
		type	<i>array</i>
		default	
		items	<i>Language code</i>
		type	<i>string</i>
		examples	cs
		pattern	<i>^.*\$</i>
		default	

繼續下一頁



表格 2 – 繼續上一頁

	• <b>sec- ondary_languages</b>	Secondary languages			
		type		array	
		default			
		items	Language code		
			type	string	
			examples	sk	
			pattern	^.*\$	
			default		
		• <b>已關注</b>	Watched projects		
			type		array
	default				
	items		Project slug		
			type	string	
			examples	weblate	
			pattern	^.*\$	
	default				
	additionalProperties	False			
• <b>稽核紀</b> 	Audit log				
	type		array		
	default				
	items	Items			
		type		object	
		properties			
		• <b>address</b>	IP address		
			type	string	
			examples	127.0.0.1	
			pattern	^.*\$	
			default		
		• <b>user_agent</b>	User agent		
			type	string	
			examples	PC / Linux / Firefox 70.0	
			pattern	^.*\$	
			default		
		• <b>時間標記</b>	Timestamp		
			type	string	
			examples	2019-11-18T18:58:30.845Z	
			default		
			format	date-time	
		• <b>activity</b>	Activity		
			type	string	
			examples	login	
			pattern	^.*\$	
			default		
	additionalProperties	False			
definitions					

也參考:

使用者個人檔案, *dumpuserdata*

## 3.11 發布 Weblate

### 3.11.1 發布日程

Weblate 有兩個月的發布期，版本 (x.y)。

主要版本的更改指示了升級過程不能跳過這個版本——在升級到更高版本的版本 x.y 之前總是必須升級到版本 x.0。

也參考：

升級 *Weblate*

### 3.11.2 發布計劃

到來的版本的特性使用 Github 里程碑來收集，可以在 <<https://github.com/WeblateOrg/weblate/milestones>> 看到路圖。

### 3.11.3 發布過程

發布前要檢查的事情：

1. 由 `./scripts/list-translated-languages` 來檢查新翻譯的語言。
2. 由 `./scripts/prepare-release` 來設置最終版本。
3. Make sure screenshots are up to date `make -j 12 -C docs update-screenshots`.
4. 合任何可能起的翻譯：命令：`wlc` 推動;git 遠程更新;git 合起源/*web blate*

執行發布：

5. 創建一個發布 `./scripts/create-release --tag`（要求請參見下面）。

張貼發布手動步驟：

6. 更新 Docker 圖像。
7. 管理 GitHub 里程碑。
8. 一旦檢測到 Docker 圖像，則添加標推送。
9. 將 Helm 圖表更新到新的版本。
10. 在 `.github/workflows/migrations.yml` 中包含新的版本，從而在合監測中覆蓋它。
11. 增加網站下載鏈接中的版本。
12. 由 `./scripts/set-version` 在倉庫中增加版本。
13. Check that readthedocs.org did build all translations of the documentation using `./scripts/rtd-projects`.

要使用 `./scripts/create-release` 本來創建標記的話，需要後面的：

- 帶有私鑰的 GnuPG 用於發布簽名
- 推送訪問 Weblate git 倉庫（它推送標記）
- 配置的 **hub** 工具和訪問，在 Weblate repo 上創建發布版本
- SSH 訪問 Weblate 下載服務器（Weblate 下載到那）

## 3.12 Security and privacy

---

**小訣竅：** Weblate 以安全為根本，努力維護重視使用者隱私的環境。

---

Development of Weblate adheres to the [Best Practices of the Linux Foundation's Core Infrastructure Initiative](#).

**也參考：**

[安全性問題](#)

### 3.12.1 Tracking dependencies for vulnerabilities

Security issues in our dependencies are monitored using [Dependabot](#). This covers the Python and JavaScript libraries, and the latest stable release has its dependencies updated to avoid vulnerabilities.

---

**提示：** 在第三方庫中可能存在漏洞，不會影響 Weblate，因此不會通過釋放釋放的 Weblate 的錯誤文件來解開這些內容。

---

### 3.12.2 Docker 容器安全

The Docker containers are regularly scanned using [Anchore](#) and [Trivy](#) security scanners.

這使我們能早期檢測漏洞并快速釋放改進。

您可以在 [GitHub](#) 上獲取這些掃描的結果 - 它們以 SARIF 格式（二態分析結果交二格式）存儲在我們的 CI 上的二影。

**也參考：**

[連續集成](#)

## 3.13 Contributing to Weblate modules

Besides the main repository, Weblate consists of several Python modules. All these follow same structure and this documentation covers them all.

For example, this covers:

- [wlc](#), Python client library, see [Weblate 客戶端](#)
- [translation-finder](#), used to discover translatable files in the repository
- [language-data](#), language definitions for Weblate, see [語言定義](#)

### 3.13.1 Coding guidelines

編寫 Weblate 的任何代碼應該時刻記得‘Security by Design Principles’（由設計原理來提供安全性）。

Any code should come with documentation explaining the behavior. Don't forget documenting methods, complex code blocks, or user visible features.

Any new code should utilize [PEP 484](#) type hints. We're not checking this in our CI yet as existing code does not yet include them.

### 3.13.2 Running tests

The tests are executed using `py.test`. First you need to install test requirements:

```
pip install -r requirements-test.txt
```

You can then execute the testsuite in the repository checkout:

```
py.test
```

也參考:

The CI integration is very similar to [Weblate 測試套件與連續集成](#).

### 3.13.3 Coding standard and linting the code

代碼應該符合 PEP-8 變成指南，[F](#)且應該使用 `black` 代碼格式化程序來格式化。

[F](#)了檢查代碼質量，可以使用 `flake8`，推薦的插件列在 `.pre-commit-config.yaml` 中，而其配置放置在 `setup.cfg` 中。

The easiest approach to enforce all this is to install `pre-commit`. The repository contains configuration for it to verify the committed files are sane. After installing it (it is already included in the `requirements-lint.txt`) turn it on by running `pre-commit install` in Weblate checkout. This way all your changes will be automatically checked.

還能[F](#)手動觸發檢查，來檢查所有文件的運行：

```
pre-commit run --all
```

也參考:

[Weblate 源代碼](#)

## 3.14 關於 Weblate

### 3.14.1 項目目標

Web-based continuous localization tool with tight [版本控制整合](#) supporting a wide range of *file formats*, making it easy for translators to contribute.

### 3.14.2 專案名稱

「Weblate」由單詞「web」和「translate」融合而來。

### 3.14.3 專案網站

The landing page is <https://weblate.org> and there is a cloud-hosted service at <https://hosted.weblate.org>. The documentation can be read at <https://docs.weblate.org>.

### 3.14.4 項目標識

The project logos and other graphics are available in <https://github.com/WeblateOrg/graphics>.

### 3.14.5 Leadership

This project is maintained by Michal Čihař, who can be reached at [michal@weblate.org](mailto:michal@weblate.org).

### 3.14.6 Authors

Weblate was started by Michal Čihař. Since its inception in 2012, thousands of people have contributed.

## 3.15 授權

More detailed licensing information is available in the Weblate source code and follows [REUSE 3.0 specification](#).

Copyright © Michal Čihař [michal@weblate.org](mailto:michal@weblate.org)

本程序是自由軟件：您可以在自由軟件基金會發布的 GNU 一般公共許可的第三版許可，或（您選擇的）更新版本的條款之下，將其重新發布且/或者修改。

發布本程序希望它有用，但不具有任何質保；甚至沒有應用可銷售性或適於特定目的的質保。更多細節請參見 GNU 一般公共許可。

您應該與本程序一起收到 GNU 一般公共許可的副本。如果有的話，請參見 <https://www.gnu.org/licenses/>。

### 4.1 Weblate 4.15.2

Released on January 25th 2023.

- Enabled gotext JSON and i18next v4 formats in the default configuration.
- Fixed crash on uploading corrupted files.
- Show stale directories in Git repository status.

[All changes in detail.](#)

### 4.2 Weblate 4.15.1

Released on January 19th 2023.

- Fixed suggestions from automatic translation.
- Fixed add-on page crash in some corner cases.
- Fixed untranslating template for new translations in some cases.
- Documented licensing using [REUSE 3.0](#).
- Fixed users pagination on team management.
- Improved performance of project creation and saving.
- Added support for gotext JSON files.
- Added support for i18next v4 files.
- Pagination in the API is now customizable.

[All changes in detail.](#)

## 4.3 Weblate 4.15

Released on December 16th 2022.

- Added support for browsing changes for a individual string.
- Fixed plurals handling in automatic translation from other components.
- Added keyboard shortcut Alt+Enter to submit string as a suggestion.
- Added support for placeables in the Fluent format.
- Improved performance of translation memory.
- Autogenerate repoweb browsing links for well known code hosting services.
- Improved performance of several views.
- Improved listing of strings with plurals.
- Added support for adding custom markup to HTML head.
- Fixed generation of MO files in the add-on to include only translated files.
- Fixed rendering of regular expression flags.
- Improved placeholders check behavior with plurals.
- Added support for translation files naming suitable for Google Play.
- Added support for labels in API.
- Added support for choosing different e-mail for commits than for notifications.
- The Docker image no longer enables debug mode by default.
- Order glossary terms based on the glossary component priority.
- Added team administrators who can add or remove members of the team.
- Added a popup confirmation before deleting users.
- Added add-on to customize XML output.

[All changes in detail.](#)

## 4.4 Weblate 4.14.2

Released on November 5th 2022.

- Added support for removing entries from translation memory.
- Improved analysis on the duplicate language alert.
- Improved accuracy of the consecutive duplicated words check.
- Improved scaling of sending many notifications.
- Improved string state handling for subtitle translation.
- Deprecated insecure configuration of VCS service API keys via `_TOKEN/_USERNAME` configuration instead of `_CREDENTIALS` list.
- Fixed processing of some uploaded CSV files.
- Improved whitespace changes handling in diff display.
- Added automatic suggestions management link to management pages.
- Track comment removal/resolving in history.
- Fixed restoring project backups with linked components.

- Fixed captcha entering on unsuccessful registration.
- Improved languages support in DeepL.
- Improved webhooks compatibility with authenticated repositories.
- Added support for Python 3.11.

[All changes in detail.](#)

## 4.5 Weblate 4.14.1

Released on September 15th 2022.

- 修復了在某些情況下生成項目備份的問題。
- 改進了文件上傳的錯誤報告。
- 在身份驗證期間從 GitHub 獲取所有使用者驗證的電子郵件。
- 避免在上下文或鍵上匹配詞表術語。
- 添加了刪除字符串的通知。
- 改進了詞表中不可翻譯術語的管理。
- 在團隊管理頁面上列出團隊成員的數量。
- 增加群組管理介面。
- 啟用評價后，始終顯示評價統計資訊。
- Added searching support in units API.
- 修復進度條在預覽流程中唯讀字串的顯示。
- Improved Burmese punctuation check.
- Fixed garbage collecting of metrics data.

所有變化的詳情見 [此處](#)。

## 4.6 Weblate 4.14

於 2022 年 8 月 22 日發布。

- 跟蹤歷史記錄中的附加更改。
- Fixed parsing translation from Windows RC, HTML and text files.
- Extended language code style configuration options.
- Added support for plurals updated in the recent CLDR releases.
- Reduced memory usage while updating components with a lot of translations.
- Added support for translation domain in SAP Translation Hub.
- Allow absolute links in source string locations.
- Improved operation behind some reverse proxies.
- Extended API to cover translation memory.
- Improved document translation workflow.
- Improved reliability of HTML and text files translation.
- Added support for project level backups.



- Improved performance and memory usage of translation memory lookups.

所有變化的詳情見。

## 4.7 Weblate 4.13.1


於 2022 年 7 月 1 日發布。

- Fixed tracking suggestions in history.
- Fixed parsing reverse proxy info from Cloudflare.
- Make parse error lock a component from translating.
- Fixed configuring intermediate file in the discovery add-on.
- Fixed DeepL translations behavior with placeholders.
- Fixed untranslating strings via API.
- Added support for removing user from a group via API.
- Fixed audit log for user invitation e-mails.
- Fixed flag names for Java formatting strings.

[All changes in detail.](#)

## 4.8 Weblate 4.13

於 2022 年 6 月 15 日發布。

- Changed behavior of updating language names.
- Added pagination to projects listing.
- API for creating new units now returns information about newly created unit.
- Component discovery now supports configuring an intermediate language.
- Added fixed encoding variants to CSV formats.
- Changed handling of context and location for some formats to better fit underlying implementation.
- Added support for ResourceDictionary format.
- Improved progress bar colors for color blind.
- Fixed variants cleanup on string removal.
- Compatibility with Django 4.1.
- Added support for storing escaped XML elements in XLIFF.
- Improved formatting of placeholder check errors.
- Redirect `/.well-known/change-password` to `/accounts/password/`.
- Machine translation services are now configurable per project.
- 新增分開的權限在解  的評論與賦予 *Review strings* 角色。
- Added support for storing alternative translations in the CSV file.
- The placeholders check can now be case-insensitive as well.

[All changes in detail.](#)

## 4.9 Weblate 4.12.2

於 2022 年 5 月 11 日發布。

- Fixed rebuilding project translation memory for some components.
- Fixed sorting components by untranslated strings.
- Fixed possible loss of translations while adding new language.
- Ensure Weblate SSH key is generated during migrations.

[All changes in detail.](#)

## 4.10 Weblate 4.12.1

於 2022 年 4 月 29 日發布。

- Fixed pull request message title.
- Improved syntax error handling in Fluent format.
- Fixed avatar display in notification e-mails.
- Add support for web monetization.
- Fixed removal of stale source strings when removing translations.

[All changes in detail.](#)

## 4.11 Weblate 4.12

於 2022 年 4 月 20 日發布。

- Added support for Amharic in 句號不相符.
- Added support for Burmese in 問號不相符.
- Extended options of the F 語系 F 生 add-on.
- Added `ignore-all-checks` flag to ignore all quality checks on a string.
- Avoid F 語系 F 生 add-on to trigger failing checks.
- Added support for *Gitea pull requests*.
- Added Linux style language code to 語言代碼類型.
- Added support for rebuilding project translation memory.
- Improved API for creating components from a file.
- Add copy and clone buttons to other translations.
- Make merge request message configurable at component level.
- Improved maximal length restriction behavior with XML tags.
- Fixed loading Fluent files with additional comments.

[All changes in detail.](#)

## 4.12 Weblate 4.11.2

於 2022 年 3 月 4 日發布。

- Fixed corrupted MO files in the binary release.

[All changes in detail.](#)

## 4.13 Weblate 4.11.1

於 2022 年 3 月 4 日發布。

- Fixed missing sanitizing of arguments to Git and Mercurial - CVE-2022-23915, see [GHSA-3872-f48p-pxqj](#) for more details.
- Fixed loading fuzzy strings from CSV files.
- Added support for creating teams using the API.
- Fixed user mention suggestions display.
- The project tokens access can now be customized.

[All changes in detail.](#)

## 4.14 Weblate 4.11

於 2022 年 2 月 25 日發布。

- Fixes stored XSS - CVE-2022-24710, see [GHSA-6jp6-9rf9-gc66](#) for more details.
- Fixed add-on installation using API.
- Renamed *Strings needing action* to *Unfinished strings*.
- Fixed false positives from *ICU MessageFormat* 語法.
- Indicate lock and contributor agreement on other occurrences listing.
- Fixed updating PO files with obsolete strings or missing plurals.
- Improved squash add-on compatibility with Gerrit.
- Automatically initialize user languages based on the [Accept-Language](#) header.
- Improved error handling on string removal.
- Weblate now requires Python 3.7 or newer.
- Fixed some write operations with project token authentication.
- Fixed string state tracking when the strings changes in the repository.
- Track string changes from the repository.
- Sticky header on translations listing to improve navigation.
- Fixed untranslating strings in *Java* 屬性.
- Fixed Git operation with non-ascii branch names.
- New add-on 將原文預先填充進翻譯.
- Added *Merge without fast-forward* 合 F 類型.
- Fixed 自動翻譯 add-on trigger on newly added strings.
- Improved punctuation checks for Burmese.

- Added support for defining custom teams at project level to grant users access, see [管理單一專案的存取控制](#).
- Added documentation links to alerts.
- Docker container automatically enables TLS/SSL for outgoing e-mail when needed.
- Added support for searching for resolved comments.
- Added support for borgbackup 1.2.
- Fixed applying of *Automatically translated* label.

[All changes in detail.](#)

## 4.15 Weblate 4.10.1

於 2021 年 12 月 22 日發布。

- Documented changes introduced by upgrading to Django 4.0.
- Fixed displaying of *Automatically translated* label.
- Fixed API display of branch in components with a shared repository.
- Improved analysis on the failed push alert.
- Fixed manually editing page when browsing changes.
- Improved accuracy of [使用 Kashida letter](#).
- The Weblate Docker container now uses Python 3.10.

[All changes in detail.](#)

## 4.16 Weblate 4.10

於 2021 年 12 月 16 日發布。

- Added support for formality and placeholders with DeepL.
- Bulk edit and search and replace are now available on project and language level.
- Added filtering to search and replace.
- Fixed: 「Perform automatic translation」 privilege is no longer part of the *Languages* group.
- 「Perform automatic translation」 is in the *Administration* and the new *Automatic translation* group.
- Fixed generating XLSX files with special chars.
- Added ability to the GitHub authentication backend to check if the user belongs to a specific GitHub organization or team.
- Improved feedback on invalid parameters passed to API.
- Added support for project scoped access tokens for API.
- Fixed string removal in some cases.
- Fixed translating newly added strings.
- Label automatically translated strings to ease their filtering.

[All changes in detail.](#)

## 4.17 Weblate 4.9.1

於 2021 年 11 月 19 日發布。

- Fixed upload of monolingual files after changing template.
- Improved handling of whitespace in flags.
- Add support for filtering in download API.
- Fixed statistics display when adding new translations.
- Mitigate issues with GitHub SSH key change.

[All changes in detail.](#)

## 4.18 Weblate 4.9

於 2021 年 11 月 10 日發布。

- Provide more details for events in history.
- Improved rendering of history.
- Improved performance of the translation pages.
- Added support for restricting translation file downloads.
- The `safe-html` can now understand Markdown when used with `md-text`.
- The `max-length` tag now ignores XML markup when used with `xml-text`.
- Fixed dimensions of rendered texts in 翻譯的最大長度.
- Lowered app store title length to 30 to assist with upcoming Google policy changes.
- Added support for customizing SSH invocation via `SSH_EXTRA_ARGS`.
- Added checks for ICU MessageFormat.
- Improved error condition handling in machine translation backends.
- Highlight unusual whitespace characters in the strings.
- Added option to stay on translated string while editing.
- Added support for customizing Borg invocation via `BORG_EXTRA_ARGS`.
- Fixed generating of MO files for monolingual translations.
- Added API endpoint to download all component translations as a ZIP file.
- Added support for Python 3.10.
- Added support for resending e-mail invitation from the management interface.

[All changes in detail.](#)

## 4.19 Weblate 4.8.1

於 2021 年 9 月 10 日發布。

- Fixed user removal in Django admin interface.
- Document add-on parameters in greater detail.
- Fixed JavaScript error in glossary.
- Add limit to number of matches in consistency check.
- Improve handling of placeholders in machine translations.
- Fixed creating add-ons using API.
- Added `PRIVACY_URL` setting to add privacy policy link to the footer.
- Hide member e-mail addresses from project admins.
- Improved gettext PO merging in case of conflicts.
- Improved glossary highlighting.
- Improved `safe-html` flag behavior with XML checks.
- Fixed commit messages for linked components.

[All changes in detail.](#)

## 4.20 Weblate 4.8

於 2021 年 8 月 21 日發布。

- Added support for Apple stringsdict format.
- The exact search operator is now case-sensitive with PostgreSQL.
- Fixed saving glossary explanations in some cases.
- 文檔改進。
- 性能改進。
- Improved squash add-on compatibility with Gerrit.
- Fixed adding strings to monolingual glossary components.
- Improved performance in handling variants.
- Fixed squash add-on sometimes skipping parsing upstream changes.
- Preserve file extension for downloads.
- Added support for the Fluent format.
- Added support for using tabs to indent JSON formats.

[All changes in detail.](#)

## 4.21 Weblate 4.7.2

於 2021 年 7 月 15 日發布。

- Support more language aliases to be configured on a project.
- Fixed search string validation in API.
- Fixed Git exporter URLs after a domain change.
- Fixed cleanup add-on for Windows RC files.
- Fixed possible crash in XLIFF updating.

[All changes in detail.](#)

## 4.22 Weblate 4.7.1

於 2021 年 6 月 30 日發布。

- Improved popup for adding terms to glossary.
- Added support for LibreTranslate machine translation service.
- Added rate limiting on creating new projects.
- Improved performance of file updates.

[All changes in detail.](#)

## 4.23 Weblate 4.7

於 2021 年 6 月 17 日發布。

- Improved configuration health check.
- Added support for `object-pascal-format` used in gettext PO, see *Object Pascal* 格式.
- 更名 F: 圭 F: “附近的鑰匙” 到: 圭 F: “類似的鑰匙” 以更好地描述目的。
- Added support for *mi18n lang files*.
- Improved SAML authentication integration.
- Fixed *Gerrit* integration to better handle corner cases.
- Weblate now requires Django 3.2.
- Fixed inviting users when e-mail authentication is disabled.
- Improved language definitions.
- Added support for blocking users from contributing to a project.
- Fixed automatic creation of glossary languages.
- Extended documentation about add-ons.
- Performance improvements for components with linked repositories.
- Added support for free DeepL API.
- The user management no longer needs Django admin interface.

所有變化的詳情見。

## 4.24 Weblate 4.6.2

於 2021 年 5 月 8 日發布。

- Fixed crash after moving shared component between projects.
- Fixed adding new strings to empty properties files.
- 修復了 RTL 語言中的副本圖標對齊。
- Extended string statistics on the Info tab.
- 修復了 Git 中忽略的翻譯文件的處理。
- Improved metrics performance.
- Fixed possible bug in saving glossaries.
- 修復了具有不同規則的語言的一致性檢查行。

所有變化的詳情見。

## 4.25 Weblate 4.6.1

發行於 2021 年 05 月 02 日。

- 除了過時的垃圾郵件防護代碼。
- Improve source plural check accuracy.
- 更新 Docker 中的用界面語言列表。
- Improved error messages when creating pull requests.
- 修復了在 Pagure 上創建合請求的問題。
- Fixed triggering automatically installed add-ons.
- Fixed possible caching issues on upgrade.
- Fixed adding new units to monolingual translations using upload.

所有變化的詳情見。

## 4.26 Weblate 4.6

發行於 2020 年 04 月 19 日。

- Auto\_Translate Management 命令現在具有用於指定翻譯模式的參數。
- Added support for 文字檔。
- 增加了所有對象的趨勢和指標。
- Added support for directly copying text from secondary languages.
- 新增覽變化時日期過濾。
- 改進了活動圖表。
- 現在可以配置聯人表格電子郵件的發件人。
- Improved parameters validation in component creation API.
- 速率限制不再適用於超級用。
- Improved automatic translation add-on performance and reliability.



- 現在可以在 Docker 容器中自定義速率限制。
- 用於創建組件的 API 現在自動使用：REF: “[內部 URL](#)”。
- 簡化了列出字符串時的狀態提示。
- 密碼散列現在默認使用 Argon2。
- Simplified progress bars indicating translation status.
- 重命名：REF: “addon-weblate.consistency.languages”以澄清目的。
- Fixed saving string state to XLIFF.
- Added language-wide search.
- 初步支持：REF: “[Docker-Scaling](#)” Docker 部署。

所有變化的詳情見。

## 4.27 Weblate 4.5.3

發佈於 2020 年 04 月 01 日。

- 修復了指標收集的問題。
- Fixed possible crash when adding strings.
- Improved search query examples.
- 修正了替`UTF-8`上傳時可能會`UTF-8`失新添加的字符串的問題。

## 4.28 Weblate 4.5.2

發佈於 2021 年 03 月 26 日。

- Configurable schedule for automatic translation.
- Added Lua format check.
- 忽略以下格式字符串：ref: “`check-duplicate`”檢查。
- 允許從翻譯頁面上傳截圖。
- 向存儲庫維護添加了`UTF-8`文件同步。
- 修復了代碼較長的語言的自動建議。
- Improved performance when adding new strings.
- 修復了幾處質量檢查中的錯誤。
- 幾處性能改進。
- Added integration with 探索 [Weblate](#).
- 修復了對只讀字符串的檢查行`UTF-8`。

所有變化的詳情見。

## 4.29 Weblate 4.5.1

發 於 2021 年 03 月 05 日。

- 修正了在某些特殊情 下編輯術語表標 的問題。
- 擴展指標用法以提高幾頁的性能。
- 以 TMX 文件存儲正確的源語言。
- Better handling for uploads of monolingual PO using API.
- Improved alerts behavior on glossary components.
- Improved Markdown link checks.
- 在 包屑導航中指示詞 表和源語言。
- Paginated component listing of huge projects.
- Improved performance of translation, component or project removal.
- 改進了大量編輯的性能。
- 固定保留 “需要編輯” 和 “批准” 狀態對 ODF 文件。
- Improved interface for customizing translation-file downloads

所有變化的詳情見。

## 4.30 Weblate 4.5

發 於 2021 年 02 月 19 日。

- Added support for `lua-format` used in gettext PO.
- Added support for sharing a component between projects.
- 用多種格式標記修復了多個未命名變量檢查行 。
- 去掉了項目的郵件列表字段， 翻譯者提供通用 。
- Added pseudolocale generation add-on.
- Added support for TermBase eXchange files.
- Added support for manually defining string variants using a flag.
- Improved performance of consistency checks.
- Improved performance of translation memory for long strings.
- Added support for searching in explanations.
- 現在可以在雙語格式中添加和 除字符串。
- 擴展亞馬遜機器翻譯支持的語言。
- Automatically enable Java MessageFormat checks for Java Properties.
- Added a new upload method to add new strings to a translation.
- Added a simple interface to browse translation.
- Glossaries are now stored as regular components.
- 除了作 組件 API 的詞 表的特定 API。
- 添加了切 一些 flag 的簡化界面。
- Added support for non-translatable or forbidden terms in the glossary.

- Added support for defining terminology in a glossary.
- 移動文本方向切 F 以獲取可視鍵盤的更多空間。
- Added option to automatically watch projects user-contributed to.
- 添加了檢查翻譯是否與詞 F 表匹配的功能。
- Added support for customizing navigation text color.

所有變化的詳情見。

## 4.31 Weblate 4.4.2

發 F 於 2021 年 1 月 14 日。

- 修復了一個發布的 MO 文件的崩潰問題。

## 4.32 Weblate 4.4.1

發 F 於 2021 年 1 月 13 日。

- Fixed reverting plural changes.
- 修復了展示項目設置幫助。
- Improved administration of users.
- Improved handling of context in monolingual PO files.
- 修復了 HTML、ODF、IDML 和 Windows RC 格式的清理附加組件的行 F。
- Fixed parsing of location from CSV files.
- 下載文件時使用 F 容壓縮。
- 改進了從 ZIP 文件導入的用 F 體驗。
- Improved detection of file format for uploads.
- 避免在 Pagure 上 F F 拉取請求。
- Improved performance when displaying ghost translations.
- 重新實現翻譯編輯器，使用原生 F 覽器文本區。
- Fixed cleanup add-on breaking adding new strings.
- Added API for add-ons.

所有變化詳情見。

## 4.33 Weblate 4.4

發 F 於 2020 年 12 月 15 日。

- Improved validation when creating a component.
- Weblate 現在需要 Django 3.1。
- Added support for appearance customization in the management interface.
- 修復了批量編輯時只讀狀態的處理。
- Improved CodeMirror integration.

- Added add-on to remove blank strings from translation files.
- The CodeMirror editor is now used for translations.
- XML、HTML、Markdown 和 reStructuredText 在翻譯編輯器中語法高亮。
- Highlight placeables in translation editor.
- 改進了對非標準語言代碼的支持。
- Added alert when using ambiguous language codes.
- 添加新的翻譯時，用會看到過濾後的語言列表。
- 擴展了更改歷史的搜索能力。
- Improved billing detail pages and Libre hosting workflow.
- Extended translation statistics API.
- Improved 「other translations」 tab while translating.
- 添加了任務 API。
- Improved performance of file upload.
- 改進了用定義的特殊字符的顯示。
- Improved performance of auto-translation.
- 幾處用小改進。
- 改進了 ZIP 文件下載的命名。
- Added option for getting notifications on unwatched projects.

所有變化的詳情見。

## 4.34 Weblate 4.3.2

發於 2020 年 11 月 4 日。

- Fixed crash on certain component file masks.
- Improved accuracy of the consecutive duplicated words check.
- Added support for Pagure pull requests.
- Improved error messages for failed registrations.
- 恢復了將開發者釋解析 Markdown 格式。
- 簡化了默認分支非「master」的 Git 倉庫的安裝設置。
- 新建的部倉庫現在使用主干作默認分支。
- 降低了翻譯重構文本時未更改譯文的誤報率。
- 修復了一些情下的 Codemirror 顯示問題。
- 將模板組重命名 “源”，來明確其意義。
- 修復了路徑較長代碼倉庫的 GitLab 拉取請求。

所有變化的詳情見。

## 4.35 Weblate 4.3.1

發 F 於 2020 年 10 月 21 日。

- Improved auto-translation performance.
- 修復了授權用 F 會話到期問題。
- Add support for hiding version information.
- 改進了 F 子與 Bitbucket 服務器的兼容性。
- Improved performance of translation memory updates.
- F 少了 F 存的使用。
- Improved performance of Matrix view.
- Added confirmation before removing a user from a project.

所有變化的詳情見。

## 4.36 Weblate 4.3

發 F 於 2020 年 10 月 15 日。

- 包括了 API 中的用 F 統計數據。
- 修復了分頁的頁面上訂購的組件。
- F 詞 F 表確定了語言。
- 重寫了對 GitHub 和 GitLab 拉取請求的支持。
- Fixed stats counts after removing suggestion.
- Extended public user profile.
- 修復了 F 制檢查的配置。
- 改進了 F 建備份的文檔。
- 將源語言屬性從項目移動到組件。
- Add Vue I18n formatting check.
- 一般 F 位符的檢查現在支持了正則表達式。
- Improved look of Matrix mode.
- Machinery is now called automatic suggestions.
- 增加了與多個 GitLab 或 GitHub 實例交互的支持。
- 擴展了 API 以覆蓋項目更新、單元更新與 F 除，以及詞 F 表。
- 單元 API 現在能正常處理多個字符串。
- 組件的新建現在能 F 處理上傳的 ZIP 文件或文檔。
- 鞏固了 API 相應狀態代碼。
- Support Markdown in contributor agreement.
- Improved source strings tracking.
- 改進了 JSON、YAML 和 CSV 格式兼容性。
- Added support for removing strings.
- 改進了文件下載的性能。

- Improved repository management view.
- 安卓自動 java 格式。
- Added support for localized screenshots.
- Added support for Python 3.9.
- 修復了某些條件下翻譯 HTML 文件。

所有變化的詳情見。

## 4.37 Weblate 4.2.2

發於 2020 年 09 月 02 日。

- Fixed matching of source strings for JSON formats.
- 修復了一些驗證配置的登重定向。
- 修復了使用組同步的 LDAP 身份驗證。
- Fixed crash in reporting automatic translation progress.
- 修復了預告時的 Git 提交變形。
- 修復了使用 API 創建本地版本控制系統（VCS）組件。

## 4.38 Weblate 4.2.1

發於 2020 年 08 月 21 日。

- 修復了在安裝資源中一些區域設置存儲數。
- Fixed crash in the cleanup add-on for some XLIFF files.
- 允許在 Docker 映像中設置本地化 CDN。

## 4.39 Weblate 4.2

發於 2020 年 8 月 18 日。

- 改進了用頁面添加了用列表。
- 去掉了從 3.x 版本遷移的支持，從 4.0 或 4.1 遷移。
- 添加了幾種單語言格式的導出。
- 改進了活動圖表。
- 可以配置字符串附近顯示的數字。
- Added support for locking components experiencing repository errors.
- 簡化了主導航（用圖表替按鈕）。
- 改進了 Google Translate 集成中的語言代碼處理。
- Git 變形附加組件可以生成“Co-authored-by:”預告。
- 改進了查詢搜索解析。
- 改進了格式字符串檢查的用反饋。
- 改進了大量的狀態更改的性能。

- 添加了項目或組件重命名後重定向的兼容性。
- 字符串的統一、組件的鎖定和許可的更改添加了通知。
- ModernMT 添加了支持。
- 允許在文件上傳時避免覆蓋已同意的翻譯。
- 去掉了一些對兼容 URL 重定向的支持。
- Added check for ECMAScript template literals.
- Added option to watch a component.
- 去掉了來自 JSON 單元密鑰的前導的點。
- Removed separate Celery queue for translation memory.
- Allow translating all components a language at once.
- 允許配置 Content-Security-Policy HTTP 標頭。
- 在項目層語言名添加支持。
- 幫助 HTML 或 JavaScript 本地化的新附加組件，請參見 *JavaScript 在地化 CDN*。
- Weblate 域現在在設置中配置，請參見 *SITE\_DOMAIN*。
- Add support for searching by component and project.

## 4.40 Weblate 4.1.1

發於 2020 年 06 月 19 日。

- 修復了 Docker 中更改自動修復或附加組件配置。
- 修復了在“關於”頁面中可能的崩潰。
- 改進了字節編譯的區域設置文件的安裝。
- 修復了向詞表添加單詞。
- 修復了機器翻譯的鍵盤快捷鍵。
- 除了一些設置中導致失事件的調試輸出。
- 修復了在項目列表中所定指示。
- 修復了一些設置中列出 GPG 密鑰。
- 需要使用的 DeepL API 版本添加了選項。
- 作 SAML 服務提供商添加了支持，請參見 *SAML 身份驗證*。

## 4.41 Weblate 4.1

發於 2020 年 06 月 15 日。

- 使用包含的國家代碼新建新的翻譯添加了支持。
- 增加了對用截圖搜索源字符串的支持。
- 擴展了統計數據洞察中可用的信息。
- 改進了在“翻譯”頁面上的搜索編輯。
- 改進了發倉庫更新的處理。
- 在項目新建表格中包括了源語言。

- 包括了信用的更改計數。
- 修復了一些情況下的 UI 語言選擇。
- 允許關閉時的白名單方法。
- 改進了詞表中相關術語的查找。
- 改進了翻譯記憶庫匹配。
- 將相同的機器翻譯結果分組。
- 編輯翻譯頁面的屏幕截圖添加了直接鏈接。
- 改進了除確認對話。
- 在 ZIP 下載中包括了模板。
- 聲明中的標記和通知配置添加了支持。
- 擴展了檢查列表的細節。
- 新的文件格式：*Laravel PHP 字串*, *HTML files*, *OpenDocument Format*, *IDML Format*, *Windows RC files*, *INI 翻譯*, *Inno Setup INI translations*, *GWT 屬性*, *go-i18n JSON files*, *:ref:arb* 添加了支持。
- 一致地使用了放作放檢查的狀態。
- Add support for configuring default add-ons to enable.
- 修復了編輯器對放檢查的鍵盤快捷鍵。
- 改進了帶有位符的字符串的機器翻譯。
- 顯示了用語言的幽靈翻譯，使之易於動。
- 改進了語言代碼解析。
- 顯示了列表中的第一個用語言的翻譯。
- 重命名來塑造更一般的名稱變量。
- 添加了新的質量檢查：*多個未命名變數*, *長期未翻譯*, *連續重單字*。
- 擦除翻譯記憶庫重新引入了支持。
- 修復了忽略檢查源的選項。
- 配置不同分支來解析更改添加了支持。
- API 現在在 HTTP 標頭重報告速率限制狀態。
- 對 Google Translate V3 API（高級版）添加了支持。
- 添加了對組件層訪問限制的能力。
- 翻譯標記中的空白字符和其它特殊字符添加了支持，請參見使用標自定義行。
- 總是顯示受到的文本檢查，如果動的話。
- API 現在支持對更改的篩選。
- 項目之間分享詞表添加了支持。



## 4.42 Weblate 4.0.4

發行於 2020 年 05 月 7 日。

- 修復了測試套件在一些 Python 3.8 環境下的執行。
- 文檔中筆誤的修復。
- 修復了一些情況下使用 API 新建組件的問題。
- 修復了移動導航中爆發的 JavaScript 錯誤。
- 修復了顯示一些檢查時的崩潰。
- 修復了屏幕截圖列表。
- 修復了每月摘要通知。
- 修復了使用翻譯中不存在的單元的中間翻譯行。

## 4.43 Weblate 4.0.3

發行於 2020 年 05 月 2 日。

- 修復了報告中可能的崩潰。
- 用在釋中的提及現在不區分大小寫。
- 修復了非超級用的 PostgreSQL 遷移。
- 修復了新建組件時更改倉庫 URL。
- 修復了上游倉庫失時的崩潰。

## 4.44 Weblate 4.0.2

發行於 2020 年 04 月 27 日。

- 改進了翻譯統計數據的性能。
- 改進了更改標的性能。
- 改進了大量編輯的性能。
- 改進了翻譯記憶庫的性能。
- 修復了組件除時可能的崩潰。
- 修復了一些極端情況下顯示翻譯更改的問題。
- 改進了 celery 隊列過長的警告。
- 改進了一致性檢查中的誤報。
- 修復了更改連接的組件倉庫時的死鎖。
- 包括了更改列表和 CSV 與報告中的編輯距離。
- 避免了對加拿大法語進行符號間隔檢查時的誤報。
- 修復了用位符導出 XLIFF。
- 修復了零寬度檢查的誤報。
- 改進了配置錯誤的報告。
- 改進了雙語言源上傳。

- 機器翻譯自動檢測支持的語言。
- 修復了一些極端情下的進度條顯示。
- 修復了非翻譯字符串出發的一些檢查。

## 4.45 Weblate 4.0.1

發於 2020 年 04 月 16 日。

- 修復了來自 Pypi 的軟件包安裝。

## 4.46 Weblate 4.0

發於 2020 年 04 月 16 日。

- Weblate 現在需要 Python 3.6 或更新版本。
- 添加了組件提醒的管理概述。
- 添加了斷裂的倉庫瀏覽器 URLs 的組件提醒。
- 改進了登陸和頁面。
- 項目訪問控制與工作流程配置集成在項目設置中。
- i18next 插值和嵌套添加了檢查和高亮標記。
- 百分號位符添加檢查和高亮標記。
- Display suggestions failing checks.
- 在歷史中記源字符串更改。
- 將 Microsoft Translator 升級版本 3 的 API。
- 重新應用翻譯記憶庫後端。
- 在搜索中查找幾個 is: 添加了支持。
- 允許未更動的翻譯避免部黑名單。
- 改進了從單語言 po 文件中提取釋。
- 重新命名要宣布的白板消息。
- 修復了郵件偶爾出現的問題。
- 改進了 LINGUAS 更新附加組件來處理更多的語法變量。
- 修復了編輯單語言 XLIFF 源文件。
- 搜索中的準確匹配添加了支持。
- 擴展了 API 覆蓋屏幕截圖、用、用組、組件列表，擴展了新建項目。
- 雙語言翻譯上傳的源添加支持。
- 開發者的中間語言添加支持。
- 新增支持來源字符串的審核。
- 擴展了平台範圍的翻譯記憶庫的下載選項。

## 4.47 Weblate 3.x 系列

### 4.47.1 Weblate 3.11.3

發 於 2020 年 03 月 11 日。

- 修復了以某種優先性來搜索字段。
- 修復了近期添加的字符串的預定義隊列。
- 修復了搜索返回重 的匹配。
- 修復 Gmail 中提供的通知。
- 修復了從歷史中還原更改。
- 添加了到摘要通知中的事件的鏈接。
- 修復了賬 除確認的電子郵件。
- 添加了對 Docker 容器中 Slack 身份認證的支持。
- 避免發送未訂 語言的通知。
- 在性能概 中包括了 Celery 隊列。
- Fixed documentation links for add-ons.
- 降低了 更改翻譯檢查的誤報。
- 提高了處理 CVE-2020-6802 的 bleach 依賴性。
- 修復了在歷史中列出項目層的更改。
- 修復了一些極端情 下的統計數據被驗證 非法。
- 修復了搜索某個字符串狀態。
- 改進了格式字符串檢查行 失百分比的問題。
- 修復了使用第三方提供商的身份驗證。

### 4.47.2 Weblate 3.11.2

發 於 2020 年 02 月 22 日。

- 修復了提出建議的問題。
- 修復了一些字符串被錯誤地報告 有單詞的問題。

### 4.47.3 Weblate 3.11.1

發 於 2020 年 02 月 20 日。

- 存檔的 Celery 設置更改。
- 改進了新建組件時文件名的驗證。
- 修復了一些依賴性的最低版本。
- 修復了以某個 Django 版本添加群組。
- 修復了手動推送到上游倉庫。
- 改進了詞 表匹配。

#### 4.47.4 Weblate 3.11

發 於 2020 年 02 月 17 日。

- 允許通過 API 新建組件的過程中使用版本控制系統（VCS）推送 URL。
- 寬度檢查現在以渲染來顯示圖像。
- 修復了通知電子郵件中的鏈接。
- 改進了純文本電子郵件的外觀。
- 顯示了忽略的檢查 且允許使它們再次激活。
- 在單語言翻譯上顯示附件的鍵。
- Added support for grouping string shapings.
- 在系統檢查時推薦更新 新的 Weblate 版本。
- 對重 的語言提醒提供了更具體的分析。
- 在項目頁面上包括更具體的版本信息。
- 如果需要的話自動非淺 本地 件。
- 修復了需要動作的字符串下載。
- New alert to warn about using the same file mask twice.
- 改進了 XML 代碼塊提取。
- `SINGLE_PROJECT` 現在可以 制重定向來選擇項目。
- 添加了選項來解 釋。
- 添加了標記的大量編輯。
- labels 添加了支持。
- Added bulk edit add-on.
- 制檢查 添加了選項。
- 增加了確認鏈接的默認驗證。
- 改進了 Matomo 集成。
- 修復了已經翻譯過 來正確低處理源字符串更改。
- 擴展了通過 `AUTO_UPDATE` 的自動更新配置。
- LINGUAS 附加組件現在在 Weblate 中完全同步翻譯。

#### 4.47.5 Weblate 3.10.3

發 於 2020 年 01 月 18 日。

- 支持 translate-toolkit 2.5.0。

#### 4.47.6 Weblate 3.10.2

發 於 2020 年 01 月 18 日。

- 項目添加了鎖定指示。
- 修復了在一些 web 瀏覽器中促使閃退的 CSS 缺陷。
- 修復了以非英語地區設置在系統中搜索。
- 改進了對 Github 和 Bitbucket 子倉庫。
- 修復了一些 Python 2.7 安裝中的數據的遷移。
- 允許 Git 錢克隆的配置。
- 改進了後台通知處理。
- 修復了在 web 瀏覽器導航回去時表格提交中斷。
- 配置 YAML 格式化的新附加組件。
- 修復了單數形式語言中有進行相同的數檢查。
- 修復了相同字段的 regex 搜索。

#### 4.47.7 Weblate 3.10.1

發 於 2020 年 01 月 09 日。

- 擴展了新建翻譯 API。
- 修復了數據遷移的幾個極端情。
- 與 Django 3.0 兼容。
- Improved data clean-up performance.
- 定制的 security.txt 添加了支持。
- 改進了更改日的包屑 (breadcrumbs)。
- 改進了面板上的翻譯列表。
- 改進了 Webhooks 的 HTTP 響應。
- 在 Docker 容器中添加了 GitLab 和請求。

#### 4.47.8 Weblate 3.10

發 於 2019 年 12 月 20 日。

- 改進了應用用界面。
- 添加了雙空格檢查。
- 修復了新建新的語言。
- 避免了向除的电子邮箱發送審計日。
- Added support for read-only strings.
- 釋中的標記添加了支持。
- 允許在項目信息中放置翻譯指示文本。
- 第二語言添加了到剪貼板。
- 改進了對 Mercurial 的支持。
- 改進了 Git 倉庫取回性能。

- 添加了搜索字符串時間的查找。
- 所有翻譯顯示源語言。
- 顯示字符串附近的語境。
- 倉庫操作的通知添加了支持。
- 改進了翻譯列表。
- 擴展了搜索能力。
- 標記編輯的自動翻譯字符串添加了支持。
- 對連接的組件提醒避免發送重發的通知。
- 改進了默認的合請求消息。
- 更好地指示了在模式下的字符串狀態。
- Yandex 翻譯中的更多語言添加了支持。
- 改進了通知電子郵件的外觀。
- 提供了翻譯許可的選擇。

#### 4.47.9 Weblate 3.9.1

發於 2019 年 10 月 28 日。

- 從備份中除了一些不需要的文件。
- 修復了報告的在崩潰。
- 修復了數據庫交叉遷移故障。
- 對制推送 Git 倉庫添加了支持。
- 降低了令牌非法的風險。
- 修復了賬除點擊率的限制。
- 添加了基於優先性的搜索。
- 修復了向 JSON 文件添加字符串可能的崩潰。
- 安全 HTML 檢查於修復現在接受源字符串標記。
- 避免向邀請的和除的用發送通知。
- 修復了在 Docker 容器中 Celery 的到 redis 的 SSL 連接。

#### 4.47.10 Weblate 3.9

發於 2019 年 10 月 15 日。

- 在下載的文件中包括了 Weblate 元數據。
- 改進了失敗檢查的 UI。
- 在格式檢查中指示了失的字符串。
- 將法語標點間隔檢查分開。
- 修復一些質量檢查錯誤添加了支持。
- 添加了分開的權限來新建新的項目。
- 擴展了字符計數的統計數據。
- 改進了對 Java 風格語言代碼的支持。

- 新的位符提供的通用檢查。
- WebExtension JSON 位符添加了支持。
- 純 XML 格式添加了支持。
- 擴展了 API 的項目、組件和翻譯的除與新建。
- Gitea 和 Gitee webhooks 添加了支持。
- 添加了新的定制的基於正則表達式的檢查。
- 允許配置來分享的翻譯記憶庫做出貢獻。
- 添加了更多翻譯文件的 ZIP 下載。
- 使 XLIFF 標準兼容最大寬度和字體的解析。
- 傳輸 web 應用的安全 HTML 標記添加了新的檢查進行了修復。
- 對未支持的配置添加了組件提醒。
- Added automatic translation add-on to bootstrap translations.
- 擴展了自動翻譯來添加建議。
- 在概覽上顯示附加組件參數。
- 現在通過當代的 Sentry SDK 而不是 Raven 支持 Sentry。
- 更改了示例設置，更好地適配生產環境。
- 添加了使用 BorgBackup 的自動備份。
- 分離了 RESX 清理附加組件，來避免不想要的文件更新。
- 添加了高級搜索功能。
- 允許下載他們自己的報告。
- 添加了本地化導來配置組件。
- 增加了對 GitLab Merge Request 的支持。
- 改進了倉庫狀態的顯示。
- 在後台執行自動翻譯。

#### 4.47.11 Weblate 3.8

發於 2019 年 08 月 15 日。

- 簡化創建相似的組件提供了支持。
- 從基於 XML 的文件格式分析翻譯標記添加了支持。
- 將意外記入 Celery 日誌。
- Improve performance of repository scoped add-ons.
- 改進了通知電子郵件的外觀。
- 修復了密碼重置行。
- 改進了多數翻譯頁面的性能。
- 修復了 Weblate 未知的語言列表。
- Add support for cloning add-ons to discovered components.
- 用上傳來替文件容添加了支持。
- 翻譯非基於版本控制系統（VCS）的容添加了支持。

- 添加了 OpenGraph widget 圖像而在社交網絡上使用。
- 動畫屏幕截圖添加了支持。
- 改進了單語言 XLIFF 文件的處理。
- 避免單個事件發送多個通知。
- 篩選更改添加了支持。
- 擴展了報告的預定義期。
- Azure Repos 添加了 webhook 支持。
- 待定建議或未翻譯字符串的新的選擇使用通知。
- 添加了通知電子郵件的一鍵退訂。
- 修復了已翻譯檢查的誤報。
- 管理員的新管理界面。
- 字符串優先性現在可以使用標記來指定。
- 添加了語言管理視圖。
- 添加了 Qt 庫和 Ruby 格式字符串的檢查。
- 添加了配置來更好地符合單一項目安裝。
- 在單語言翻譯中通過源字符串更改中的新字符串。
- 添加帶有搜索功能的另外的翻譯記憶庫視圖。

#### 4.47.12 Weblate 3.7.1

發於 2019 年 06 月 28 日。

- 文檔更新。
- 修復了一些要求限制。
- 更新了語言數據庫。
- 本地化更新。
- 各種用界面調整。
- 改進了對不支持但發現的翻譯文件的處理。
- 更詳細地報告失文件格式要求。

#### 4.47.13 Weblate 3.7

發於 2019 年 06 月 21 日。

- 添加了用於通知的另外的 Celery 隊列。
- 對於 API 覽使用了與應用一致的外觀。
- 在報告中包括了同意的統計數據。
- 當更新翻譯組件時的報告過程。
- 允許終止運行後台組件的更新。
- 文件名操作擴展了模板語言。
- 對於編輯器鏈接和倉庫覽器 URL 使用模板。
- 當編輯翻譯時指示最大長度和當前字符計數。



- 改進了未更改翻譯檢查中的縮寫處理。
- 刷新了新貢獻者的登入頁面。
- Add support for configuring msgmerge add-on.
- 當發送通知時延遲打開 SMTP 連接。
- 改進了錯誤日誌。
- 允許 MO 生成附加組件中的自定義位置。
- 添加了清理舊的建議或解釋的附加組件。
- 添加了選項來啟動編輯器的水平模式。
- 提高了許多被鏈接組件的導入性能。
- 修復了一些情況下的示例安裝。
- 改進了更改中提供提醒。
- 添加了新水平統計數據 widget。
- 改進了數值的格式字符串檢查。
- 添加了字體管理工具。
- 提供文本尺寸的新檢查。
- 字幕格式添加了支持。
- 包括了語言的全部完成的統計數據。
- 添加了在項目或全體範圍內的報告。
- 改進了顯示翻譯狀態時的用戶界面。
- 新的 Weblate 標誌和配色方案。
- 位圖徽章的新外觀。

#### 4.47.14 Weblate 3.6.1

發行於 2019 年 04 月 26 日。

- 改進了單語言 XLIFF 文件的處理。
- 修復了一些極端情況下的摘要通知。
- 修復了附加組件本錯誤警告的問題。
- 修復了單語言 PO 文件生成 MO 文件。
- 修復了未安裝檢查的顯示。
- 指示項目列表中管理的項目。
- 允許更新而從丟失的版本控制系統（VCS）倉庫恢復。

### 4.47.15 Weblate 3.6

發 於 2019 年 04 月 20 日。

- 下載用 數據添加了支持。
- 附加組件在安裝時會被自動觸發。
- 改進了解 合 衝突的指示。
- 清理附加組件現在與 app 商店元數據翻譯兼容。
- 當添加新語言時可配置的語言代碼語法。
- 警告 Python 2 的使用計劃於 2020 年 04 月終止。
- 視覺鍵盤從源字符串中提取特定字符。
- 擴展了貢獻者統計數據，來反映源計數和目標計數。
- 管理員和一致性附加組件現在可以添加翻譯，即使對用 禁用。
- 修復了禁止 Language-Team 標頭操作切 的描述。
- 通知 釋中提到的用 。
- 從組件設置中 除了文件格式自動檢查。
- 修復了 單語言 PO 文件生成 MO 文件。
- 添加了摘要通知。
- 組件通知 音添加了支持。
- 新的提醒、白板消息或組件添加了通知。
- 現在可以配置管理項目的通知。
- 改進了三字母語言代碼的處理。

### 4.47.16 Weblate 3.5.1

發 於 2019 年 03 月 10 日。

- 修復了 Celery systemd 單元例子。
- 修復了帶有登 的 HTTP 倉庫的通知。
- 修復了單語言翻譯中編輯源字符串的競態條件。
- 在日 中包括附加組件執行失敗的輸出。
- 改進了添加新語言選擇的驗證。
- 在組件設置中允許編輯文件格式。
- 更新安裝指示 首選 Python 3。
- 裝入翻譯時的性能與一致性改進。
- 使 Microsoft Terminology 服務與當前的 Zeep 發布版本兼容。
- 本地化更新。

### 4.47.17 Weblate 3.5

發 於 2019 年 03 月 03 日。

- 改進了 建翻譯記憶庫的性能。
- 添加了管理全體翻譯記憶庫的界面。
- 改進了壞的組件狀態的提醒。
- 添加了管理白板消息的用 界面。
- 現在可以配置附加組件提交消息。
- 少更新上游倉庫時的提交數量。
- 修復了在項目之間移動組件時可能的元數據失。
- 改進了纏模式的導航。
- 添加了幾個新的質量檢查（相關的標記和 URL）。
- 對 app 商店元數據文件添加了支持。
- 切 GitHub 或 Gerrit 集成添加了支持。
- 卡什達字母添加了檢查。
- 添加選項來根據作者壓縮提交。
- 改進了對 XLSX 文件格式的支持。
- 與 Tesseract 4.0 兼容。
- 賬單附加組件現在在未支付賬單 45 天后除項目。

### 4.47.18 Weblate 3.4

發 於 2019 年 01 月 22 日。

- XLIFF 位符添加了支持。
- Celery 現在可以使用多個項目隊列。
- 項目與組件的重命名和移動添加了支持。
- 在報告中包括字符計數。
- 添加了帶有翻譯文件自動檢測的翻譯組件添加導。
- 可定制的 Git 合提交消息。
- 在導航中添加了組件提醒的視覺指示。
- 改進了裝入翻譯文件的性能。
- New add-on to squash commits prior to push.
- 改進了翻譯更改的顯示。
- 更改了默認合風格來 rebase 且使之可配置。
- 更好地處理語言代碼中的私用子標。
- 改進了全文索引更新的性能。
- 擴展了文件上傳 API 來支持更多的參數。

### 4.47.19 Weblate 3.3

發 於 2018 年 11 月 30 日。

- 組件和項目 除添加了支持。
- 改進了一些單語言翻譯的性能。
- 添加了翻譯組件提醒，使翻譯的問題高亮。
- 當可用時暴露 XLIFF 字符串 `resname` 作 語境。
- XLIFF 狀態添加支持。
- DATA\_DIR 中的非可寫入文件添加檢查。
- 改進了更改的 CSV 導出。

### 4.47.20 Weblate 3.2.2

發 於 2018 年 10 月 20 日。

- 除了不再需要的 Babel 依賴性。
- 更新了語言定義。
- 改進了附加組件、LDAP 和 Celery 的文檔。
- 修復了 動新的 `dos-eol` 和 `auto-java-messageformat` 標記。
- 修復了從 PyPI 軟件包運行 `setup.py` 測試。
- 改進了 數處理。
- 修復了一些極端情 下失敗的翻譯上傳 API。
- 修復了手動更改情 下更新 Git 配置。

### 4.47.21 Weblate 3.2.1

發 於 2018 年 10 月 18 日。

- Python 2.7 的 `backports.csv` 的文檔依賴性。
- 修復根下運行測試。
- 改進了 `gitexport` 模塊中的錯誤處理。
- 修復了報告新添加語言的過程。
- 正確地向 Sentry 報告 Celery worker 錯誤。
- 修復了用 Qt linguist 新建新的翻譯。
- 修復了偶發的全文索引更新失敗。
- 改進了新建新組件時的驗證。
- 清理舊的建議添加了支持。

### 4.47.22 Weblate 3.2

發 於 2018 年 10 月 06 日。

- 自動化的附加組件安裝添加 `install_addon` 管理命令。
- 允許更細粒度的速率限制設置。
- Excel 文件的導出和導入添加了支持。
- 改進了多組件發現附加組件情 下的組件清理。
- 重寫了 Microsoft Terminology 機器翻譯後端。
- Weblate 現在使用 Celery 來卸載一些處理。
- 改進了搜索功能， 添加了正則表達式搜索。
- 有道智雲機器翻譯添加了支持。
- 百度 API 機器翻譯添加了支持。
- 集成了使用 Celery 的維護和清理任務。
- 改進了裝入翻譯的性能幾乎 25%。
- 除了對上傳時合 標頭的支持。
- 除了對客 提交消息的支持。
- 可配置的編輯模式（ 模式/全模式）。
- 向 Sentry 報告錯誤添加了支持。
- 每天自動更新倉庫添加了支持。
- 用 新建項目和組件添加了支持。
- Built-in translation memory now automatically stores translations done.
- 用 和項目現在可以導入他們現有的翻譯記憶庫。
- 更好地管理屏幕截圖的相關字符串。
- 檢查 Java MessageFormat 添加了支持。

所提到問題的具體列表請參見 [3.2 milestone on GitHub](#)。

### 4.47.23 Weblate 3.1.1

發 於 2018 年 07 月 27 日。

- 修復了一些設置中失敗的測試套件。

### 4.47.24 Weblate 3.1

發 於 2018 年 07 月 27 日。

- 不再支持從早於 3.0.1 的更舊版本的升級。
- 允許覆蓋從設置中默認提交的消息。
- 改進 webhooks 與自主環境的兼容性。
- 添加了對 Amazon Translate 的支持。
- 與 Django 2.1 兼容。
- Django system 系統檢查現在用於診斷安裝問題。
- 除了將很快關閉的 libavatar 服務的支持。

- New add-on to mark unchanged translations as needing edit.
- 翻譯時跳到特定位置添加支持。
- 現在可以定制下載翻譯。
- 改進了翻譯記憶庫匹配中的字符串相似性的計算。
- 添加了由 GnuPG 簽發 Git 提交的支持。

#### 4.47.25 Weblate 3.0.1

發 於 2018 年 06 月 10 日。

- 修復了自 2.20 依賴可能的合 問題。
- 本地化更新。
- 除了過時的 子的例子。
- 改進了緩存文檔。
- 改進了管理文檔的顯示。
- 改進了長語言名稱的處理。

#### 4.47.26 Weblate 3.0

發 於 2018 年 06 月 01 日。

- 重寫了訪問控制。
- 導致移動或重命名模塊的一些代碼清理。
- 自動組件發現的新附加組件。
- `import_project` 管理命令現在有一些略微不同的參數。
- Windows RC 文件添加了基本支持。
- 新的附加組件，將貢獻者姓名存儲在 PO 文件標頭中。
- 除了每組件 子本，轉而使用附加組件替代。
- 收集貢獻者協議添加支持。
- 現在在歷史中跟踪訪問控制更改。
- New add-on to ensure all components in a project have same translations.
- 在提交消息模板中支持更多變量。
- 提供另外的文本語境添加支持。

### 4.48 Weblate 2.x 系列

#### 4.48.1 Weblate 2.20

發 於 2018 年 04 月 04 日。

- 改進了 subversion 倉庫克隆的速度。
- 更改了倉庫鎖定來使用第三方庫。
- 下載只需要動作的字符串添加了支持。
- 立刻在幾種語言中搜索添加了支持。

- 新附加組件來配置 gettext 輸出行。
- 新附加組件來配置 JSON 格式。
- 使用 RFC 6750 兼容的 Bearer 認證添加了 API 中認證的支持。
- 未使用機器翻譯服務自動翻譯添加了支持。
- 白板消息中的 HTML 標記添加了支持。
- 大量更改字符串的狀態添加了支持。
- 現在至少需要 2.3.0 版本的 Translate-toolkit，較老的版本不再支持。
- Added built-in translation memory.
- 面板添加了組件概和每個組件列表概的頁面。
- DeepL 機器翻譯服務添加了支持。
- 機器翻譯結果現在緩存在 Weblate 。
- Added support for reordering committed changes.

#### 4.48.2 Weblate 2.19.1

發於 2018 年 02 月 20 日。

- 修復了從 2.18 升級的合問題。
- 改進了文件上傳 API 驗證。

#### 4.48.3 Weblate 2.19

發於 2018 年 02 月 15 日。

- 修復了跨一些文件格式的導入。
- 在審查日中顯示用友好的瀏覽器信息。
- 文件添加了 TMX 導出程序。
- 裝入翻譯文件的各種性能改進。
- 添加了選項在 Weblate 中禁止訪問管理，有利於 Django。
- 改進了大字符串的詞表查詢速度。
- 與 django\_auth\_ldap 1.3.0 兼容。
- 配置錯誤現在被存儲被持久地報告。
- 在空白字符自動修復中 Honor 忽略標記。
- 改進了一些 Subversion 設置的兼容性。
- Improved built-in machine translation service.
- 對 SAP Translation Hub 服務添加了支持。
- Microsoft Terminology 服務添加了支持。
- 除了對通知電子郵件中的廣告的支持。
- 改進了語言層次的翻譯過程報告。
- 改進了不同數形式的支持。
- 添加了對不使用 stdlayout 的 Subversion 倉庫的支持。
- Added add-ons to customize translation workflows.

#### 4.48.4 Weblate 2.18

發 於 2017 年 12 月 15 日。

- 擴展了貢獻者的統計數據。
- 改進了特殊字符可視鍵盤的配置。
- DTD 文件格式添加了支持。
- 更改了鍵盤快捷鍵，較少地於瀏覽器/系統的快捷鍵衝突。
- 改進了對 XLIFF 文件中的同意標記的支持。
- 添加了對 gettext PO 文件中不行的長字符串的支持。
- 添加了按鈕當前翻譯永久鏈接。
- 去掉了對 Django 1.10 的支持，而添加了對 Django 2.0 的支持。
- 除了當翻譯時對翻譯的鎖定。
- 對於單語言翻譯添加新的字符串添加了支持。
- 對專門復核人員的翻譯流程添加了支持。

#### 4.48.5 Weblate 2.17.1

發 於 2017 年 10 月 13 日。

- 修復了在一些特定情況下運行測試套件的問題。
- 區域設置更新。

#### 4.48.6 Weblate 2.17

發 於 2017 年 10 月 13 日。

- 現在 Weblate 默認進行 Git 淺克隆。
- 改進了更新大的翻譯文件時的性能。
- 阻擋的某個電子郵件地址添加了支持。
- 用現在可以除自己的釋。
- 搜索和替特性添加了預覽步驟。
- 搜索和上傳表格時客戶端一側設置的持久性。
- 擴展了搜索能力。
- 每個項目的更細粒度的 ACL 配置。
- BASE\_DIR 的默認值已經更改。
- 添加了兩步驟除，來防止意外除。
- 現在可以編輯項目訪問控制設置。
- 添加了使用 Akismet 建議的可選垃圾郵件保護。



### 4.48.7 Weblate 2.16

發 於 2017 年 08 月 11 日。

- 更重新能改進。
- 嵌套 JSON 格式添加了支持。
- WebExtension JSON 格式添加了支持。
- 修復了 git 導出認證。
- 改進了某些情況下的 CSV 導入。
- 改進了其它翻譯 widget 的外觀。
- 對於表格中的文本長度現在限制進行最大長度檢測。
- 每個組件進行 commit\_pending 時間配置。
- 各種用界面清理。
- 修復了組件/項目/網站範圍的翻譯搜索。

### 4.48.8 Weblate 2.15

發 於 2017 年 06 月 30 日。

- 在其它翻譯中顯示了更多相關的翻譯。
- 添加了選項來查看當前字符串到其它語言的翻譯。
- 對立陶宛語默認使用 4 種數形式。
- 修復了不同格式單語言文件的上傳。
- 改進了認證失敗的錯誤消息。
- 當從詞表除單詞時保持頁面狀態。
- 添加了直接鏈接來編輯第二語言翻譯。
- 添加了 Perl 格式質量檢查。
- 了拒再次使用的密碼添加了支持。
- 編輯 RTL 語言擴展了工具條。

### 4.48.9 Weblate 2.14.1

發 於 2017 年 05 月 24 日。

- 修復了將搜索結果分頁時可能的錯誤。
- 修復了一些極端情況下從較老的版本合的問題。
- 修復了項目有人或無人監守時可能的 CSRF。
- 密碼重置不再認證用。
- 修復了忘記密碼是可能的略過 CAPTCHA。

#### 4.48.10 Weblate 2.14

發 於 2017 年 05 月 17 日。

- 使用 AJAX 添加詞表條目。
- 登出現在使用 POST 來避免 CSRF。
- API 密鑰令牌重置現在使用 POST 來避免 CSRF。
- Weblate 默認設置 Content-Security-Policy。
- 驗證本地編輯器 URL 來避免 self-XSS。
- 現在默認相對於一般缺陷來驗證密碼。
- 向用 通知他們賬 的重要動作，如密碼變更。
- CSV 導出現在會避免可能的公式。
- 安全上的各種小改進。
- 現在限制了認證嘗試的次數。
- 建議的 容存儲在歷史中。
- 在審計日 中存儲重要的賬行 。
- 當 除賬 或添加新的團體時，詢問密碼確認。
- 當做出建議時顯示時間。
- 新的質量檢查來跟踪分號。
- 確保搜索鏈接可以被分享。
- 在 API 中包括了源字符串信息和屏幕截圖。
- 允許通過 API 上傳覆蓋翻譯。

#### 4.48.11 Weblate 2.13.1

發 於 2017 年 04 月 12 日。

- 修復了個人資料中管理的項目的列表。
- 修復了在一些許可 失的情 下的合 問題。
- 修復了翻譯下載中當前文件的列表。
- 當用 有特權而嘗試訪問項目時返回 HTTP 404。

#### 4.48.12 Weblate 2.13

發 於 2017 年 04 月 12 日。

- 修復了翻譯模板的質量檢查。
- 添加了質量檢查，在 失翻譯時觸發。
- 添加選項來查看用 待定的建議。
- 添加選項來自動建立組件列表。
- 未認證用 可以配置的默認面板。
- 新增選項來隨機 覽 25 個字串於審查。
- 歷史現在指示字符串的更改。
- 添加新的翻譯時更好的錯誤報告。

- 在項目中添加了每種語言的搜索。
- Group（群組）ACLs 現在可以限制某個權限。
- The per project ACLs are now implemented using Group ACL.
- 添加了更精細顆粒的特權控制。
- 各種小的 UI 改進。

#### 4.48.13 Weblate 2.12

發於 2017 年 03 月 03 日。

- 改進了群組的管理界面。
- Yandex Translate API 添加了支持。
- 改進了網站範圍的搜索速度。
- 添加了項目和組件範圍的搜索。
- 添加了項目和組件範圍的搜索與替。
- 改進了將不一致翻譯待定。
- 在本地編輯器中添加了開源文件的支持。
- 配置帶有特殊字符的鍵盤添加了支持。
- 改進了帶有 OCR 支持的屏幕截圖管理來匹配源字符串。
- 默認提交消息現在包括翻譯信息和 URL。
- Joomla 翻譯格式添加了支持。
- 改進了重要的跨文件格式的可靠性。

#### 4.48.14 Weblate 2.11

發於 2017 年 01 月 31 日。

- 在語言頁包括了具體語言信息。
- Mercurial 後端改進。
- 添加了選項來定翻譯組件優先性。
- 更一致地使用 Group ACLs，即使具有較少使用的許可。
- 子本添加了 WL\_BRANCH 變量。
- 改進了開發者文檔。
- 與 Git 導出器附加組件中多個 Git 版本的更佳兼容性。
- 包括每個項目的和組件的統計數據。
- 添加語言代碼映射，來更好地支持 Microsoft Translate API。
- 將全文清理移動後台工作，使去除翻譯更快速。
- 修復了具有單個數形式的語言的數源顯示。
- 改進了 import\_project 中的錯誤處理。
- 更重新能改進。

#### 4.48.15 Weblate 2.10.1

發 於 2017 年 01 月 20 日。

- 不 露密碼重置表格上存在賬 (CVE-2017-5537)。

#### 4.48.16 Weblate 2.10

發 於 2016 年 12 月 15 日。

- 添加質量檢查，來檢查是否 數被翻譯出來。
- 對帶有驗證的倉庫修復了 GitHub 子。
- 添加了可選的 Git 導出模塊。
- 支持 Microsoft Cognitive Services Translator API。
- 簡化了項目和組件用 界面。
- 添加了自動修復來去除控 字符。
- 項目添加了每種語言的概 。
- CSV 導出添加了支持。
- 統計數據添加了 CSV 下載。
- 增加了矩陣視圖，以快速概述所有翻譯。
- 更改和字符串添加了基本 API。
- Apertium APy 服務器添加了支持，用於機器翻譯。

#### 4.48.17 Weblate 2.9

發 於 2016 年 11 月 04 日。

- 擴展了 createadmin 管理命令的參數。
- 擴展了 import\_json，而能 處理現有組件。
- YAML 文件添加了支持。
- 項目所有者現在可以配置翻譯組件和項目細節。
- 使用「Watched」（關注的）而不是「Subscribed」（訂 的）項目。
- 可以從項目頁面直接關注項目。
- 添加多語言狀態 widget。
- 如果 有顯示源的話，將第二語言高亮。
- 將建議 除記 在歷史中。
- 改進個人資料中語言選擇的 UX。
- 修復了顯示組件的白板信息。
- 存儲後保持偏好標 被選擇。
- 更突出地顯示源字符串 釋。
- 從 Git 倉庫自動安裝 Gettext PO 合 驅動。
- 添加搜索和替 特性。
- 上傳可視 容（屏幕截圖）用於翻譯添加了支持。

### 4.48.18 Weblate 2.8

發 於 2016 年 08 月 31 日。

- 文檔改進。
- Translations.
- Updated bundled JavaScript libraries.
- 添加了 list\_translators 管理命令。
- 不再支持 Django 1.8。
- 修復了與 Django 1.10 的兼容性。
- 添加了 Subversion 支持。
- 從 XML 合法性檢查中分出不匹配的 XML 標 。
- 修復了 API 來接受 HIDE\_REPO\_CREDENTIALS 設置。
- 在 模式中顯示源更改。
- Alt+PageUp/PageDown/Home/End 現在也在 模式中工作良好。
- 添加提示來顯示更改的準確時間。
- 添加選項從翻譯頁面中選擇篩選程序和搜索。
- 添加了翻譯 除的 UI。
- 改進了插入代碼塊時的行 。
- 修復了 模式的自動鎖定問題。

### 4.48.19 Weblate 2.7

發 於 2016 年 07 月 10 日。

- 去除了 Google web translation 機器翻譯。
- 改進了添加翻譯時的提交消息。
- 修復了希伯來語的 Google Translate API。
- 與 Mercurial 3.8 的兼容性。
- 添加了 import\_json 管理命令。
- 糾正了列出翻譯的順序。
- 顯示全部建議文本，而只是差 。
- 擴展 API（具體的倉庫狀態、統計數據、...）。
- 測試套件不再需要網絡範文來測試倉庫。

### 4.48.20 Weblate 2.6

發 於 2016 年 04 月 28 日。

- 修復了語言篩選程序的驗證組件。
- 改進了對 XLIFF 文件的支持。
- 修復了非英語源的機器翻譯。
- 添加了 REST API。
- Django 1.10 的兼容性。
- 白板纖細添加了分類。

### 4.48.21 Weblate 2.5

發 於 2016 年 03 月 10 日。

- 修復了項目所有者的自動翻譯。
- 改進了提交和推送操作的性能。
- 新的管理命令，從命令行添加建議。
- 文件上傳時的合 釋添加了支持。
- C printf 格式的一些 GNU 後綴添加了支持。
- 文檔改進。
- 生成翻譯者信用添加了支持。
- 生成貢獻者統計數據添加了支持。
- 網站範圍的搜索可以只在一種語言中搜索。
- 改進了亞美尼亞語的質量檢查。
- 開始 有現有翻譯的翻譯組件添加支持。
- 支持在 Qt TS 中添加新的翻譯。
- 改進了翻譯 PHP 文件的支持。
- 質量檢查的性能提高。
- Fixed site wide search for failing checks.
- 添加選型來指定源語言。
- 改進了對 XLIFF 文件的支持。
- 擴展了 import\_project 選項的列表。
- 改進了白板消息的目標。
- 支持跨項目的自動翻譯。
- 最佳化全文搜尋索引。
- 自動翻譯添加了管理命令。
- 添加了代碼塊高亮。
- 代碼塊、檢查和機器翻譯添加了鍵盤快捷方式。
- 改進了翻譯鎖定。
- AngularJS 插值添加了質量檢查。
- 添加了廣泛的基於群組的 ACLs。

- 闡明了需要編輯的字符串的術語（以前被稱 fuzzy，即模糊）。
- Clarified terminology on strings needing action and untranslated strings.
- 支持 Python 3。
- 去掉了對 Django 1.7 的支持。
- 去掉了用於新建新的 gettext PO 文件的 msginit 依賴性。
- 添加了可配置的面板視圖。
- 改進了解析錯誤的通知。
- 添加了選項將帶有復制名稱的組件導入 import\_project。
- 改進了翻譯 PHP 文件的支持。
- 添加了字典的 XLIFF 導出。
- 所有翻譯添加了 XLIFF 和 gettext PO 導出。
- 文檔改進。
- 可配置的自動群組指定添加了支持。
- 改進了新翻譯的添加。

#### 4.48.22 Weblate 2.4

發於 2015 年 09 月 20 日。

- 改進了對 PHP 文件的支持。
- 匿名用添加 ACL 的能力。
- 改進了 import\_project 命令的配置。
- 添加了歷史的 CSV 轉儲。
- 避免/粘貼空白字符的錯誤。
- Bitbucket webhooks 添加了支持。
- 在翻譯上傳時對模糊字符串更嚴格的控制。
- 幾個 URLs 已經更改，可能必須要更新您的書。
- 子本以 VCS root 作當前目錄執行。
- 子本以描述當前組件的環境變量來執行。
- 添加了管理命令來優化全文索引。
- 動態條上報告錯誤添加了支持。
- 項目現在可以有多個所有者。
- 項目所有者可以管理自己。
- gettext PO 中使用的 javascript-format 添加了支持。
- 在 XLIFF 中添加新翻譯的支持。
- 改進了文件格式自動檢測。
- 擴展了鍵盤快捷鍵。
- 改進了幾種語言的字典匹配。
- 改進了多數頁面的局。
- 支持在翻譯時將單詞添加入字典。

- 篩選語言由 Weblate 管理添加了支持。
- 翻譯導入 CSV 文件添加了支持。
- 重寫了態文件的處理。
- 第三放服務的直接登入鏈接，如果是唯一一個的話。
- 賬除時提交待定的更改。
- 添加管理命令來更改網站名稱。
- 添加選項來配置默認提交者。
- 在添加新翻譯後添加子。
- 添加選項來指定多個文件來添加提交。

### 4.48.23 Weblate 2.3

發佈於 2015 年 05 月 22 日。

- 去掉對 Django 1.6 和 South migrations 的支持。
- 支持使用 Java 屬性文件時添加新翻譯。
- 允許接受建議而不編輯。
- 改進了對 Google OAuth 2.0 的支持。
- 添加了對 Microsoft .resx 文件的支持。
- 微調了默認的 robots.txt 文件而不允許翻譯特緩慢地進行。
- 簡化了接受建議的工作流程。
- 添加了總是接收重要通知的項目所有者。
- 允許禁止編輯單語言模板。
- 更具體的倉庫狀態視圖。
- 更改翻譯時編輯模板的直接鏈接。
- 允許項目所有者添加更多的權限。
- 允許在模式下顯示第二語言。
- 支持隱藏源字符串而有利於第二語言。

### 4.48.24 Weblate 2.2

發佈於 2015 年 02 月 19 日。

- 性能改進。
- 在所在地和釋位置的全文檢索。
- New SVG/JavaScript-based activity charts.
- 支持 Django 1.8。
- 支持除釋。
- 添加自己的 SVG 徽章。
- Google Analytics 添加支持。
- 改進了翻譯文件名的處理。
- 單語言 JSON 翻譯添加了支持。



- 在歷史中記組件鎖定。
- 支持單語言翻譯編輯源（模板）語言。
- Gerrti 添加了基本支持。

#### 4.48.25 Weblate 2.1

發於 2014 年 12 月 05 日。

- Mercurial 倉庫添加了支持。
- 由 Awesome 替了 Glyphicon 字體。
- 添加了社交驗證服務的圖標。
- 更一致的按鈕色和圖標。
- 文檔改進。
- 各種缺陷修復。
- 對於小屏幕自動隱藏翻譯列表中的列。
- 更改了文件系統路徑的配置。
- 改進了 SSH 密鑰處理與存儲。
- 改進了倉庫鎖定。
- 每個源字符串可定制的質量檢查。
- 允許從面板上隱藏完成的翻譯。

#### 4.48.26 Weblate 2.0

發於 2014 年 11 月 06 日。

- 使用 Bootstrap 的新的倉庫 UI。
- 重寫了 VCS 後端。
- 文檔改進。
- 網站範圍的消息添加了白板。
- 可配置的字符串優先性。
- JSON 文件格式添加了支持。
- 修復了某些情況下生成 mo 文件。
- GitLab 通知添加了支持。
- 添加了支持來禁止翻譯建議。
- Django 1.7 支持。
- ACL 項目現在具有用管理。
- 擴展了搜索可能性。
- 翻譯者給出數的更多提示。
- 修復了 Git 倉庫鎖定。
- 與舊的 Git 版本的兼容性。
- 改進了 ACL 支持。
- 每種語言引用和其它特殊字符添加了按鈕。

- 支持導出統計數據作 JSONP。

## 4.49 Weblate 1.x 系列

### 4.49.1 Weblate 1.9

發於 2014 年 05 月 06 日。

- Django 1.6 兼容性。
- 不在維護與 Django 1.4 的兼容性。
- 用於鎖定/解鎖翻譯的管理命令。
- 改進了對 Qt TS 文件的支持。
- 用現在可以除他們的賬。
- 頭像可以被禁止。
- 合第一個和最後一個名稱屬性。
- 頭像現在可以取回緩存在服務器一側。
- shields.io 徽章添加了支持。

### 4.49.2 Weblate 1.8

發於 2013 年 11 月 07 日。

- 關於升級的指示，請查看手。
- 更好地列出了項目概。
- 用於分享的更好的可視化選型。
- 對匿名用特權的更多控制。
- 支持使用第三方服務登，更多細節請查看手。
- 用可以通過電子郵件地址而不是用名登。
- 文檔改進。
- 改進了來源字串審查。
- 跨所有字串的搜索。
- 更好地跟踪源字符串。
- 的 Captcha 保護。

### 4.49.3 Weblate 1.7

發於 2013 年 10 月 07 日。

- 關於升級的指示，請查看手。
- 支持檢查 Python brace 格式字符傳。
- 每個組件的質量檢查定制。
- 具體的每個翻譯的統計數據。
- 更改了將建議、檢查與釋連接到字符串的方式。
- 用現在可以添加文本來提交消息。

- 支持新語言請求的訂
- 支持添加新的翻譯。
- 現在使用 Pillow 而不是 Pango+ Cairo 來提供 Widget 和圖表。
- 添加狀態徽章 Widget。
- 去掉了不合法的文本方向檢查。
- 字典中的更改現在在歷史中記入日。
- Performance improvements for translation view.

#### 4.49.4 Weblate 1.6

發於 2013 年 07 月 25 日。

- 時更好的錯誤處理。
- 更改的覽。
- 修復了機器翻譯建議的排序。
- 改進了對 MyMemory 機器翻譯的支持。
- Amagam 機器翻譯添加了支持。
- 頻繁使用的頁面上的各種優化。
- 在搜索結果中將搜索到的短語高亮。
- 支持存儲消息時的自動修復。
- 跟踪翻譯歷史和選項來還原。
- Google Translate API 添加了翻譯。
- 管理 SSH 主機密鑰添加了支持。
- 各種表格驗證改進。
- 各種質量檢查改進。
- 導入的性能改進。
- 建議投票添加了支持。
- 清理管理員界面。

#### 4.49.5 Weblate 1.5

發於 2013 年 04 月 16 日。

- 關於升級的指示，請查看手。
- 添加了公共用頁面。
- 更好地命名數形式。
- 詞表的 TBX 導出添加了支持。
- Bitbucket 通知添加了支持。
- 行圖標現在可用於每個翻譯、語言或用。
- 擴展了 import\_project admin 命令的選項。
- 與 Django 1.5 兼容。
- 現在使用 libavatar 顯示頭像。

- pretty print JSON 導出添加了可能性。
- 更重新能改進。
- 對項目或同樣對語言，在進度條中指示失敗的檢查或模糊字符串。
- 客預提交投資與提交另外的文件添加了支持。
- 重寫了搜索實現更好的性能與用體驗。
- 機器翻譯的新界面。
- 添加了對單語言 po 文件的支持。
- 擴展了緩存的元數據的量，來改進各種搜索的速度。
- 現在也顯示單詞計數。

#### 4.49.6 Weblate 1.4

發於 2013 年 01 月 23 日。

- 修復了在字符串除時除檢查/釋。
- 添加了選項來禁止自動傳播翻譯。
- 添加了選項合失敗而訂。
- 正確地導入需要客 ttkit 導入程序的項目。
- 添加了網站地圖允許爬蟲更容易地訪問。
- 在通知電子郵件或 feed 中提供到字符串的直接鏈接。
- 對管理員界面的各種改進。
- 在管理員界面中生設置提供提示。
- 添加了每種語言的 widgets 和接合頁面。
- 改進了翻譯鎖定處理。
- widgets 以更多變量顯示代碼段。
- 在進度挑中指示失敗的檢測或模糊字符串。
- 用於將提交信息格式化的更多選項。
- 修復了機器翻譯服務的錯誤處理。
- 改進了自動翻譯鎖定行。
- 支持顯示不同於之前的源字符串的更改。
- 字符串搜索添加了支持。
- 各種質量檢查改進。
- 支持每個項目的 ACL。
- 基本代碼由單元測試來覆蓋。

### 4.49.7 Weblate 1.3

發 於 2012 年 11 月 16 日。

- 與 PostgreSQL 數據庫後端兼容。
- 將上游 git 倉庫中 的語言也 。
- 改進了質量檢查處理。
- 添加了新的檢查（BB 代碼、XML 標記和新的行）。
- 支持可選地 rebasing 而不是合 。
- 重新定位 Weblate 的可能性（例如在 /weblate 路徑下運行）。
- 支持在自動檢測失敗的情 下手動選擇文件類型。
- 更好地支持 Android 源。
- 支持從 web 界面生成 SSH 密鑰。
- 更多的可視化數據導出。
- 新的按鈕來輸入一些特殊字符。
- 支持導出字典。
- 支持鎖定整個 Weblate 安裝。
- 檢查與支持來源字串的審查。
- 支持翻譯和源字符串的用 釋。
- 更好地更改日 跟踪。
- 現在更改可以使用 RSS 監測。
- 改進了對 RTL 語言的支持。

### 4.49.8 Weblate 1.2

發 於 2012 年 08 月 14 日。

- Weblate 現在 數據庫遷移使用 South，如果升級的話請檢查升級的指示。
- 修復了連接 git repos 的小問題。
- 現在將人員與翻譯建立聯 的簡介頁面使用 Weblate。
- 添加了小程序，可以用於推動翻譯項目。
- 添加了選項將倉庫重置 原始狀態（對於有特權的用 ）。。
- 現在可以 翻譯鎖定項目和組件。
- 禁止一些翻譯的可能性。
- 添加新翻譯的可配置選項。
- 每個項目的 git 提交的配置。
- 簡單的防垃圾郵件保護。
- 主頁面更好的 局。
- 支持每次提交時自動推送更改。
- 支持譯者電子郵件通知。
- 在配置編輯器中只列出使用的語言。
- 改進了當導入項目時對未知語言的處理。

- 支持翻譯者鎖定翻譯。
- 可選地維護 po 文件中的 Language-Team 標頭。
- 在 about 頁面中包括一些統計數據。
- 支持（且需要）django-registration 0.8。
- Caching counts of strings with failing checks.
- 在設置中檢查要求。
- 文檔改進。

#### 4.49.9 Weblate 1.1

發 於 2012 年 07 月 04 日。

- 改進了幾個翻譯。
- 新建組件時更好的驗證。
- 在組件之間分享 git 倉庫添加了支持。
- 不必每次嘗試都提交來拉取遠程 repo。
- 卸載索引添加了支持。

#### 4.49.10 Weblate 1.0

發 於 2012 年 05 月 10 日。

- 改進了添加/存儲組件時的驗證。
- 試驗支持安卓組件文件（需要打補丁的 ttkit）。
- 來自子的更新在後台運行。
- 改進了安裝指示。
- 改進了字典中的導航。

### 4.50 Weblate 0.x 系列

#### 4.50.1 Weblate 0.9

發 於 2012 年 04 月 18 日。

- 修復了未知語言的導入。
- 改進了附近消息的列表。
- 改進了幾項檢查。
- 文檔更新。
- 更多的幾種語言添加了定義。
- 各種編碼清理。
- 文檔改進。
- 更改了文件局。
- 將幫助本更新 Django 1.4。
- 改進了翻譯時的導航。

- 更好地處理 po 文件的重命名。
- 新建組件時更好的驗證。
- 將完全設置集成到 syncdb 中。
- 所有翻譯頁面添加了最近更改的列表。
- Check for untranslated strings ignores format string only messages.

### 4.50.2 Weblate 0.8

發行於 2012 年 04 月 03 日。

- 用 Whoosh 代替自己的全文搜索。
- 對檢查的各種修改和改進。
- 新的命令更新檢查。
- 很多翻譯更新。
- 添加了字典，來存儲最常用的術語。
- 倉庫狀態的概覽添加了 /admin/report/。
- 機器翻譯頁面不再阻礙裝入頁面。
- 管理界面現在也包含有用的行來更新數據。
- 記錄所做更改的日誌。
- 推至提交給 Git 的能力，來產生來自單一用的更少的提交。
- 瀏覽失敗檢查的可能性。
- 使用已經翻譯的字符串來自動翻譯。
- 新的顯示使用的版本的頁面。
- Django 1.4 的兼容性。
- 從 web 界面將更改推送給遠程 repo 的能力。
- 添加了對其他人完成的翻譯進行檢查。

### 4.50.3 Weblate 0.7

發行於 2012 年 02 月 16 日。

- 對 GitHub 通知的直接支持。
- 添加了對清潔孤立的檢查與翻譯的支持。
- 翻譯時顯示附近的字符串。
- 翻譯時顯示相似的字符串。
- 改進了字符串的搜索。

#### 4.50.4 Weblate 0.6

發 於 2012 年 02 月 14 日。

- 添加了對翻譯的消息的各種檢查。
- 可調整的訪問控制。
- 改進了對新一行翻譯的處理。
- 添加了客 端一側表格的排序。
- 在升級時請檢查升級指示。

#### 4.50.5 Weblate 0.5

發 於 2012 年 02 月 12 日。

- 通過使用後面的服務對機器翻譯的支持：
  - Apertium
  - 微軟的 Translator
  - MyMemory
- 幾個新的翻譯。
- 改進了上游更改的合 。
- 更好地處理 發的 git 拉取與翻譯。
- 傳播那些同樣用於模糊更改的工作。
- 傳播那些同樣用於文件上傳的工作。
- 修復了使用 FastCGI（也可能是其它）時的文件下載。

#### 4.50.6 Weblate 0.4

發 於 2012 年 02 月 08 日。

- 將使用 導添加到文檔中。
- 修復了不需要 CSRF 保護的 API 子。

#### 4.50.7 Weblate 0.3

發 於 2012 年 02 月 08 日。

- 對於復數翻譯更海底顯示源。
- Sphinx 格式的新文檔。
- 翻譯時顯示第二語言。
- 改進了錯誤頁面，來給出現有項目的列表。
- 新的每種語言的統計數據。



### 4.50.8 Weblate 0.2

發佈於 2012 年 02 月 07 日。

- 改進了幾種形式的驗證。
- 在個人資料升級時警告用戶。
- 記住 URL 用於登入。
- 輸入代碼形式時對文本區域的命名。
- 自動擴大翻譯區域。

### 4.50.9 Weblate 0.1

發佈於 2012 年 02 月 06 日。

- 初始發布。

## W

wlc, [155](#)

wlc.config, [156](#)

wlc.main, [156](#)

---

## HTTP Routing Table

---

<b>/</b>	GET /api/components/(string:project)/(string:comp
	126
<b>/api</b>	GET /api/components/(string:project)/(string:comp
	128
<b>/api/addons</b>	GET /api/components/(string:project)/(string:comp
	129
<b>/api/addons/</b>	GET /api/components/(string:project)/(string:comp
	127
<b>/api/addons/(int:id)/</b>	GET /api/components/(string:project)/(string:comp
	126
<b>/api/addons/(int:id)/</b>	GET /api/components/(string:project)/(string:comp
	130
<b>/api/addons/(int:id)/</b>	GET /api/components/(string:project)/(string:comp
	129
<b>/api/addons/(int:id)/</b>	POST /api/components/(string:project)/(string:comp
	143
<b>/api/addons/(int:id)/</b>	POST /api/components/(string:project)/(string:comp
	131
<b>/api/addons/(int:id)/</b>	POST /api/components/(string:project)/(string:comp
	127
<b>/api/addons/(int:id)/</b>	POST /api/components/(string:project)/(string:comp
	128
<b>/api/addons/(int:id)/</b>	POST /api/components/(string:project)/(string:comp
	129
<b>/api/addons/(int:id)/</b>	PUT /api/components/(string:project)/(string:comp
	125
<b>/api/addons/(int:id)/</b>	DELETE /api/components/(string:project)/(string:c
	125
<b>/api/addons/(int:id)/</b>	DELETE /api/components/(string:project)/(string:c
	131
<b>/api/addons/(int:id)/</b>	PATCH /api/components/(string:project)/(string:co
	124
<b>/api/components</b>	<b>/api/groups</b>
	GET /api/groups/, 109
<b>/api/components/(string:project)/(string:comp</b>	GET /api/groups/(int:id)/, 109
	121
<b>/api/components/(string:project)/(string:comp</b>	POST /api/groups/, 109
	112
<b>/api/components/(string:project)/(string:comp</b>	POST /api/groups//changes/,
	112
<b>/api/components/(string:project)/(string:comp</b>	POST /api/groups//file-/id/components/,
	111
<b>/api/components/(string:project)/(string:comp</b>	POST /api/groups//links-/id/languages/,
	111
<b>/api/components/(string:project)/(string:comp</b>	
	130

POST /api/groups/(int:id)/projects/, 111  
 DELETE /api/projects/(string:project)/, 116  
 POST /api/groups/(int:id)/roles/, 111  
 PATCH /api/projects/(string:project)/, 116  
 PUT /api/groups/(int:id)/, 110  
 DELETE /api/groups/(int:id)/, 110  
 DELETE /api/groups/(int:id)/componentlist/, 112  
 GET /api/roles/, 112  
 DELETE /api/groups/(int:id)/components/, 111  
 GET /api/roles/(int:id)/, 112  
 DELETE /api/groups/(int:id)/languages/, 111  
 POST /api/roles/, 112  
 PUT /api/language/(int:id)/, 113  
 DELETE /api/roles/(int:id)/, 113  
 DELETE /api/groups/(int:id)/projects/(int:project\_id)/, 111  
 PATCH /api/projects/(int:id)/, 113  
 PATCH /api/groups/(int:id)/, 110

## /api/languages

GET /api/languages/, 113  
 GET /api/languages/(string:language)/, 113  
 GET /api/languages/(string:language)/string/, 115  
 POST /api/languages/, 113  
 POST /api/screenshots/(int:id)/file/, 141  
 PUT /api/languages/(string:language)/, 114  
 POST /api/screenshots/(int:id)/units/, 141  
 DELETE /api/languages/(string:language)/, 115  
 PUT /api/screenshots/(int:id)/, 142  
 PATCH /api/languages/(string:language)/, 114  
 DELETE /api/screenshots/(int:id)/, 142  
 DELETE /api/screenshots/(int:id)/units/(int:unit\_id)/, 142  
 PATCH /api/screenshots/(int:id)/, 142

## /api/memory

GET /api/memory/, 137  
 DELETE /api/memory/(int:memory\_object\_id)/, 137

## /api/metrics

GET /api/metrics/, 145

## /api/projects

GET /api/projects/, 115  
 GET /api/projects/(string:project)/, 115  
 GET /api/projects/(string:project)/changes/, 117  
 GET /api/projects/(string:project)/component/, 118  
 GET /api/projects/(string:project)/languages/, 120  
 GET /api/projects/(string:project)/repositories/, 117  
 GET /api/projects/(string:project)/statistics/, 121  
 POST /api/projects/, 115  
 POST /api/projects/(string:project)/components/, 118  
 POST /api/projects/(string:project)/repositories/, 117  
 PUT /api/projects/(string:project)/, 116

## /api/roles

## /api/screenshots

GET /api/screenshots/, 140  
 GET /api/screenshots/(int:id)/, 140  
 GET /api/screenshots/(int:id)/file/, 140  
 POST /api/screenshots/, 141  
 POST /api/screenshots/(int:id)/file/, 141  
 POST /api/screenshots/(int:id)/units/, 141  
 PUT /api/screenshots/(int:id)/, 142  
 DELETE /api/screenshots/(int:id)/, 142  
 DELETE /api/screenshots/(int:id)/units/(int:unit\_id)/, 142  
 PATCH /api/screenshots/(int:id)/, 142

## /api/tasks

GET /api/tasks/, 145  
 GET /api/tasks/(str:uuid)/, 145

## /api/translations

GET /api/translations/, 131  
 GET /api/translations/(string:project)/(string:component)/, 131  
 GET /api/translations/(string:project)/(string:component)/, 134  
 GET /api/translations/(string:project)/(string:component)/, 135  
 GET /api/translations/(string:project)/(string:component)/, 136  
 GET /api/translations/(string:project)/(string:component)/, 136  
 GET /api/translations/(string:project)/(string:component)/, 134  
 POST /api/translations/(string:project)/(string:component)/, 134  
 POST /api/translations/(string:project)/(string:component)/, 135  
 POST /api/translations/(string:project)/(string:component)/, 136  
 POST /api/translations/(string:project)/(string:component)/, 134  
 DELETE /api/translations/(string:project)/(string:component)/, 133

## /api/units

```
GET /api/units/, 137
GET /api/units/(int:id)/, 137
PUT /api/units/(int:id)/, 139
DELETE /api/units/(int:id)/, 139
PATCH /api/units/(int:id)/, 138
```

## /api/users

```
GET /api/users/, 106
GET /api/users/(str:username)/, 106
GET /api/users/(str:username)/notifications/,
108
GET /api/users/(str:username)/notifications/(int:subscription_id)/,
108
GET /api/users/(str:username)/statistics/,
108
POST /api/users/, 106
POST /api/users/(str:username)/groups/,
107
POST /api/users/(str:username)/notifications/,
108
PUT /api/users/(str:username)/, 106
PUT /api/users/(str:username)/notifications/(int:subscription_id)/,
108
DELETE /api/users/(str:username)/, 107
DELETE /api/users/(str:username)/groups/,
107
DELETE /api/users/(str:username)/notifications/(int:subscription_id)/,
109
PATCH /api/users/(str:username)/, 107
PATCH /api/users/(str:username)/notifications/(int:subscription_id)/,
108
```

## /exports

```
GET /exports/rss/, 150
GET /exports/rss/(string:project)/,
149
GET /exports/rss/(string:project)/(string:component)/,
149
GET /exports/rss/(string:project)/(string:component)/(string:language)/,
149
GET /exports/rss/language/(string:language)/,
149
GET /exports/stats/(string:project)/(string:component)/,
148
```

## /hooks

```
GET /hooks/update/(string:project)/,
146
GET /hooks/update/(string:project)/(string:component)/,
146
POST /hooks/azure/, 147
POST /hooks/bitbucket/, 147
POST /hooks/gitea/, 147
POST /hooks/gitee/, 148
POST /hooks/github/, 146
POST /hooks/gitlab/, 146
POST /hooks/pagure/, 147
```

## 符號

.XML resource file  
    file format, 87  
--add  
    auto\_translate 命令列選項, 374  
--addon  
    install\_addon 命令列選項, 380  
--age  
    commit\_pending 命令列選項, 375  
--author  
    add\_suggestions 命令列選項, 374  
--author-email  
    wlc 命令列選項, 153  
--author-name  
    wlc 命令列選項, 153  
--base-file-template  
    import\_project 命令列選項, 378  
--check  
    importusers 命令列選項, 380  
--config  
    wlc 命令列選項, 151  
--config-section  
    wlc 命令列選項, 151  
--configuration  
    install\_addon 命令列選項, 380  
--convert  
    wlc 命令列選項, 153  
--email  
    createadmin 命令列選項, 376  
--file-format  
    import\_project 命令列選項, 378  
--force  
    loadpo 命令列選項, 381  
--force-commit  
    pushgit 命令列選項, 382  
--format  
    wlc 命令列選項, 151  
--fuzzy  
    wlc 命令列選項, 153  
--ignore  
    import\_json 命令列選項, 377  
--inconsistent  
    auto\_translate 命令列選項, 374  
--input  
    wlc 命令列選項, 153  
--key  
    wlc 命令列選項, 151  
--lang  
    loadpo 命令列選項, 381  
--language-code  
    list\_translators 命令列選項, 381  
--language-map  
    import\_memory 命令列選項, 378  
--language-regex  
    import\_project 命令列選項, 379  
--license  
    import\_project 命令列選項, 379  
--license-url  
    import\_project 命令列選項, 379  
--main-component  
    import\_json 命令列選項, 377  
    import\_project 命令列選項, 379  
--method  
    wlc 命令列選項, 153  
--mode  
    auto\_translate 命令列選項, 374  
--mt  
    auto\_translate 命令列選項, 374  
--name  
    createadmin 命令列選項, 376  
--name-template  
    import\_project 命令列選項, 378  
--new-base-template  
    import\_project 命令列選項, 378  
--no-password  
    createadmin 命令列選項, 376  
--no-privs-update  
    setupgroups 命令列選項, 383  
--no-projects-update  
    setupgroups 命令列選項, 383  
--no-update  
    setuplang 命令列選項, 383  
--output  
    wlc 命令列選項, 153  
--overwrite  
    auto\_translate 命令列選項, 374  
    wlc 命令列選項, 153

--password  
     createadmin 命令列選項, 376  
 --project  
     import\_json 命令列選項, 377  
 --source  
     auto\_translate 命令列選項, 374  
 --threshold  
     auto\_translate 命令列選項, 374  
 --update  
     createadmin 命令列選項, 376  
     import\_json 命令列選項, 377  
     install\_addon 命令列選項, 380  
 --url  
     wlc 命令列選項, 151  
 --user  
     auto\_translate 命令列選項, 374  
 --username  
     createadmin 命令列選項, 376  
 --vcs  
     import\_project 命令列選項, 379

## A

add\_suggestions  
     weblate admin command, 374  
 add\_suggestions 命令列選項  
     --author, 374  
 ADMINS  
     setting, 204  
 AKISMET\_API\_KEY  
     setting, 331  
 ALLOWED\_HOSTS  
     setting, 204  
 Android  
     file format, 82  
 ANONYMOUS\_USER\_NAME  
     setting, 331  
 API, 102, 150, 155  
 Apple strings  
     file format, 83  
 ARB  
     file format, 86  
 AUDITLOG\_EXPIRY  
     setting, 331  
 AUTH\_LOCK\_ATTEMPTS  
     setting, 331  
 AUTH\_TOKEN\_VALID  
     setting, 332  
 auto\_translate  
     weblate admin command, 374  
 auto\_translate 命令列選項  
     --add, 374  
     --inconsistent, 374  
     --mode, 374  
     --mt, 374  
     --overwrite, 374  
     --source, 374  
     --threshold, 374  
     --user, 374

AUTO\_UPDATE  
     setting, 332  
 AUTOFIX\_LIST  
     setting, 333  
 AVATAR\_URL\_PREFIX  
     setting, 332

## B

BACKGROUND\_TASKS  
     setting, 333  
 BaseAddon (*weblate.addons.base* 中的類), 417  
 BASIC\_LANGUAGES  
     setting, 334  
 bilingual  
     translation, 74  
 BORGE\_EXTRA\_ARGS  
     setting, 334

## C

can\_install() (*weblate.addons.base.BaseAddon* 的類方法), 417  
 CELERY\_BACKUP\_OPTIONS, 162, 178  
 CELERY\_BEAT\_OPTIONS, 162, 178  
 CELERY\_MAIN\_OPTIONS, 162, 178  
 CELERY\_MEMORY\_OPTIONS, 162, 178  
 CELERY\_NOTIFY\_OPTIONS, 162, 178  
 celery\_queues  
     weblate admin command, 375  
 CELERY\_TRANSLATE\_OPTIONS, 162, 178  
 changes  
     wlc 命令列選項, 152  
 CHECK\_LIST  
     setting, 334  
 checkgit  
     weblate admin command, 375  
 cleanup  
     wlc 命令列選項, 152  
 cleanup\_ssh\_keys  
     weblate admin command, 376  
 cleanuptrans  
     weblate admin command, 375  
 Comma separated values  
     file format, 88  
 Command (*wlc.main* 中的類), 156  
 COMMENT\_CLEANUP\_DAYS  
     setting, 335  
 commit  
     wlc 命令列選項, 152  
 commit\_pending  
     weblate admin command, 375  
 commit\_pending 命令列選項  
     --age, 375  
 COMMIT\_PENDING\_HOURS  
     setting, 335  
 commitgit  
     weblate admin command, 375  
 configure() (*weblate.addons.base.BaseAddon* 的方法), 417

CONTACT\_FORM  
    setting, 335  
createadmin  
    weblate admin command, 376  
createadmin 命令列選項  
    --email, 376  
    --name, 376  
    --no-password, 376  
    --password, 376  
    --update, 376  
    --username, 376  
CSP\_CONNECT\_SRC  
    setting, 334  
CSP\_FONT\_SRC  
    setting, 334  
CSP\_IMG\_SRC  
    setting, 334  
CSP\_SCRIPT\_SRC  
    setting, 334  
CSP\_STYLE\_SRC  
    setting, 334  
CSV  
    file format, 88

## D

daily() (*weblate.addons.base.BaseAddon* 的方法), 417  
DATA\_DIR  
    setting, 335  
DATABASE\_BACKUP  
    setting, 336  
DATABASES  
    setting, 204  
DEBUG  
    setting, 204  
DEFAULT\_ACCESS\_CONTROL  
    setting, 336  
DEFAULT\_ADD\_MESSAGE  
    setting, 337  
DEFAULT\_ADDON\_MESSAGE  
    setting, 337  
DEFAULT\_ADDONS  
    setting, 337  
DEFAULT\_AUTO\_WATCH  
    setting, 337  
DEFAULT\_COMMIT\_MESSAGE  
    setting, 337  
DEFAULT\_COMMITTER\_EMAIL  
    setting, 338  
DEFAULT\_COMMITTER\_NAME  
    setting, 338  
DEFAULT\_DELETE\_MESSAGE  
    setting, 337  
DEFAULT\_FROM\_EMAIL  
    setting, 204  
DEFAULT\_LANGUAGE  
    setting, 338  
DEFAULT\_MERGE\_MESSAGE

    setting, 337  
DEFAULT\_MERGE\_STYLE  
    setting, 338  
DEFAULT\_PAGE\_LIMIT  
    setting, 347  
DEFAULT\_PULL\_MESSAGE  
    setting, 339  
DEFAULT\_RESTRICTED\_COMPONENT  
    setting, 337  
DEFAULT\_SHARED\_TM  
    setting, 339  
DEFAULT\_TRANSLATION\_PROPAGATION  
    setting, 339  
download  
    wlc 命令列選項, 153  
DTD  
    file format, 90  
dump\_memory  
    weblate admin command, 376  
dumpuserdata  
    weblate admin command, 377

## E

ENABLE\_AVATARS  
    setting, 339  
ENABLE\_HOOKS  
    setting, 339  
ENABLE\_HTTPS  
    setting, 339  
ENABLE\_SHARING  
    setting, 340  
EXTRA\_HTML\_HEAD  
    setting, 340

## F

file format  
    .XML resource file, 87  
Android, 82  
Apple strings, 83  
ARB, 86  
Comma separated values, 88  
CSV, 88  
DTD, 90  
gettext, 76  
go-i18n, 85  
gotext, 86  
GWT properties, 80  
i18next, 85  
INI translations, 80, 81  
Java properties, 79  
Joomla translations, 81  
JSON, 84  
mi18n lang, 80  
PHP strings, 83  
PO, 76  
Qt, 81  
RC, 91  
ResourceDictionary, 88



RESX, 87  
 Ruby YAML, 89  
 Ruby YAML Ain't Markup Language, 89  
 string resources, 82  
 TS, 81  
 WPF, 88  
 XLIFF, 77  
 XML, 90  
 YAML, 89  
 YAML Ain't Markup Language, 89

## G

`get()` (*wlc.Weblate* 的方法), 155  
`get_add_form()` (*weblate.addons.base.BaseAddon* 的類方法), 417  
`GET_HELP_URL`  
   setting, 340  
`get_settings_form()` (*weblate.addons.base.BaseAddon* 的方法), 417  
`gettext`  
   file format, 76  
`GITEA_CREDENTIALS`  
   setting, 340  
`GITEA_TOKEN`  
   setting, 341  
`GITEA_USERNAME`  
   setting, 340  
`GITHUB_CREDENTIALS`  
   setting, 342  
`GITHUB_TOKEN`  
   setting, 342  
`GITHUB_USERNAME`  
   setting, 342  
`GITLAB_CREDENTIALS`  
   setting, 341  
`GITLAB_TOKEN`  
   setting, 341  
`GITLAB_USERNAME`  
   setting, 341  
`go-i18n`  
   file format, 85  
`GOOGLE_ANALYTICS_ID`  
   setting, 342  
`gotext`  
   file format, 86  
`GWT properties`  
   file format, 80

## H

`HIDE_REPO_CREDENTIALS`  
   setting, 343  
`HIDE_VERSION`  
   setting, 343

## I

`i18next`  
   file format, 85

`import_demo`  
   weblate admin command, 377  
`import_json`  
   weblate admin command, 377  
`import_json` 命令列選項  
   --ignore, 377  
   --main-component, 377  
   --project, 377  
   --update, 377  
`import_memory`  
   weblate admin command, 378  
`import_memory` 命令列選項  
   --language-map, 378  
`import_project`  
   weblate admin command, 378  
`import_project` 命令列選項  
   --base-file-template, 378  
   --file-format, 378  
   --language-regex, 379  
   --license, 379  
   --license-url, 379  
   --main-component, 379  
   --name-template, 378  
   --new-base-template, 378  
   --vcs, 379  
`importuserdata`  
   weblate admin command, 380  
`importusers`  
   weblate admin command, 380  
`importusers` 命令列選項  
   --check, 380  
`INI translations`  
   file format, 80, 81  
`install_addon`  
   weblate admin command, 380  
`install_addon` 命令列選項  
   --addon, 380  
   --configuration, 380  
   --update, 380  
`INTERLEDGER_PAYMENT_POINTERS`  
   setting, 343  
`iOS`  
   translation, 83  
`IP_BEHIND_REVERSE_PROXY`  
   setting, 343  
`IP_PROXY_HEADER`  
   setting, 344  
`IP_PROXY_OFFSET`  
   setting, 344

## J

`Java properties`  
   file format, 79  
`Joomla translations`  
   file format, 81  
`JSON`  
   file format, 84

## L

LEGAL\_TOS\_DATE  
setting, 344

LEGAL\_URL  
setting, 344

LICENSE\_EXTRA  
setting, 345

LICENSE\_FILTER  
setting, 345

LICENSE\_REQUIRED  
setting, 345

LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH  
setting, 346

list\_languages  
weblate admin command, 381

list\_translators  
weblate admin command, 381

list\_translators 命令列選項  
--language-code, 381

list\_versions  
weblate admin command, 381

list-components  
wlc 命令列選項, 152

list-languages  
wlc 命令列選項, 152

list-projects  
wlc 命令列選項, 152

list-translations  
wlc 命令列選項, 152

load() (*wlc.config.WeblateConfig* 的方法), 156

loadpo  
weblate admin command, 381

loadpo 命令列選項  
--force, 381  
--lang, 381

LOCALIZE\_CDN\_PATH  
setting, 346

LOCALIZE\_CDN\_URL  
setting, 346

lock  
wlc 命令列選項, 152

lock\_translation  
weblate admin command, 382

lock-status  
wlc 命令列選項, 152

LOGIN\_REQUIRED\_URLS  
setting, 346

LOGIN\_REQUIRED\_URLS\_EXCEPTIONS  
setting, 346

ls  
wlc 命令列選項, 152

## M

main() (於 *wlc.main* 模組中), 156

MATOMO\_SITE\_ID  
setting, 347

MATOMO\_URL  
setting, 347

mi18n lang  
file format, 80

monolingual  
translation, 74

move\_language  
weblate admin command, 382

## N

NEARBY\_MESSAGES  
setting, 347

## P

PAGURE\_CREDENTIALS  
setting, 348

PAGURE\_TOKEN  
setting, 348

PAGURE\_USERNAME  
setting, 348

PHP strings  
file format, 83

PIWIK\_SITE\_ID  
setting, 347

PIWIK\_URL  
setting, 347

PO  
file format, 76

post() (*wlc.Weblate* 的方法), 155

post\_add() (*weblate.addons.base.BaseAddon* 的方法), 417

post\_commit() (*weblate.addons.base.BaseAddon* 的方法), 417

post\_push() (*weblate.addons.base.BaseAddon* 的方法), 417

post\_update() (*weblate.addons.base.BaseAddon* 的方法), 417

pre\_commit() (*weblate.addons.base.BaseAddon* 的方法), 418

pre\_push() (*weblate.addons.base.BaseAddon* 的方法), 418

pre\_update() (*weblate.addons.base.BaseAddon* 的方法), 418

PRIVACY\_URL  
setting, 348

PRIVATE\_COMMIT\_EMAIL\_OPT\_IN  
setting, 349

PRIVATE\_COMMIT\_EMAIL\_TEMPLATE  
setting, 349

PROJECT\_BACKUP\_KEEP\_COUNT  
setting, 349

PROJECT\_BACKUP\_KEEP\_DAYS  
setting, 349

PROJECT\_NAME\_RESTRICT\_RE  
setting, 350

PROJECT\_WEB\_RESTRICT\_RE  
setting, 350

pull  
wlc 命令列選項, 152

push

wlc 命令列選項, 152  
 pushgit  
   weblate admin command, 382  
 pushgit 命令列選項  
   --force-commit, 382  
 Python, 155  
 Python Enhancement Proposals  
   PEP 484, 413, 427

## Q

Qt  
   file format, 81

## R

RATELIMIT\_ATTEMPTS  
   setting, 350  
 RATELIMIT\_LOCKOUT  
   setting, 350  
 RATELIMIT\_WINDOW  
   setting, 350  
 RC  
   file format, 91  
 REDIS\_PASSWORD, 175  
 register\_command() (於 *wlc.main* 模組中), 156  
 REGISTRATION\_ALLOW\_BACKENDS  
   setting, 351  
 REGISTRATION\_CAPTCHA  
   setting, 351  
 REGISTRATION\_EMAIL\_MATCH  
   setting, 351  
 REGISTRATION\_OPEN  
   setting, 351  
 repo  
   wlc 命令列選項, 152  
 REPOSITORY\_ALERT\_THRESHOLD  
   setting, 352  
 REQUIRE\_LOGIN  
   setting, 352  
 reset  
   wlc 命令列選項, 152  
 ResourceDictionary  
   file format, 88  
 REST, 102  
 RESX  
   file format, 87  
 RFC  
   RFC 5646, 73  
 Ruby YAML  
   file format, 89  
 Ruby YAML Ain't Markup Language  
   file format, 89

## S

save\_state() (*weblate.addons.base.BaseAddon* 的  
   方法), 418  
 SECRET\_KEY  
   setting, 204  
 SENTRY\_DSN

setting, 352  
 SERVER\_EMAIL  
   setting, 205  
 SESSION\_COOKIE\_AGE\_AUTHENTICATED  
   setting, 352  
 SESSION\_ENGINE  
   setting, 204  
 setting  
   ADMINS, 204  
   AKISMET\_API\_KEY, 331  
   ALLOWED\_HOSTS, 204  
   ANONYMOUS\_USER\_NAME, 331  
   AUDITLOG\_EXPIRY, 331  
   AUTH\_LOCK\_ATTEMPTS, 331  
   AUTH\_TOKEN\_VALID, 332  
   AUTO\_UPDATE, 332  
   AUTOFIX\_LIST, 333  
   AVATAR\_URL\_PREFIX, 332  
   BACKGROUND\_TASKS, 333  
   BASIC\_LANGUAGES, 334  
   BORG\_EXTRA\_ARGS, 334  
   CHECK\_LIST, 334  
   COMMENT\_CLEANUP\_DAYS, 335  
   COMMIT\_PENDING\_HOURS, 335  
   CONTACT\_FORM, 335  
   CSP\_CONNECT\_SRC, 334  
   CSP\_FONT\_SRC, 334  
   CSP\_IMG\_SRC, 334  
   CSP\_SCRIPT\_SRC, 334  
   CSP\_STYLE\_SRC, 334  
   DATA\_DIR, 335  
   DATABASE\_BACKUP, 336  
   DATABASES, 204  
   DEBUG, 204  
   DEFAULT\_ACCESS\_CONTROL, 336  
   DEFAULT\_ADD\_MESSAGE, 337  
   DEFAULT\_ADDON\_MESSAGE, 337  
   DEFAULT\_ADDONS, 337  
   DEFAULT\_AUTO\_WATCH, 337  
   DEFAULT\_COMMIT\_MESSAGE, 337  
   DEFAULT\_COMMITTER\_EMAIL, 338  
   DEFAULT\_COMMITTER\_NAME, 338  
   DEFAULT\_DELETE\_MESSAGE, 337  
   DEFAULT\_FROM\_EMAIL, 204  
   DEFAULT\_LANGUAGE, 338  
   DEFAULT\_MERGE\_MESSAGE, 337  
   DEFAULT\_MERGE\_STYLE, 338  
   DEFAULT\_PAGE\_LIMIT, 347  
   DEFAULT\_PULL\_MESSAGE, 339  
   DEFAULT\_RESTRICTED\_COMPONENT, 337  
   DEFAULT\_SHARED\_TM, 339  
   DEFAULT\_TRANSLATION\_PROPAGATION,  
     339  
   ENABLE\_AVATARS, 339  
   ENABLE\_HOOKS, 339  
   ENABLE\_HTTPS, 339  
   ENABLE\_SHARING, 340  
   EXTRA\_HTML\_HEAD, 340

- GET\_HELP\_URL, 340
  - GITEA\_CREDENTIALS, 340
  - GITEA\_TOKEN, 341
  - GITEA\_USERNAME, 340
  - GITHUB\_CREDENTIALS, 342
  - GITHUB\_TOKEN, 342
  - GITHUB\_USERNAME, 342
  - GITLAB\_CREDENTIALS, 341
  - GITLAB\_TOKEN, 341
  - GITLAB\_USERNAME, 341
  - GOOGLE\_ANALYTICS\_ID, 342
  - HIDE\_REPO\_CREDENTIALS, 343
  - HIDE\_VERSION, 343
  - INTERLEDGER\_PAYMENT\_POINTERS, 343
  - IP\_BEHIND\_REVERSE\_PROXY, 343
  - IP\_PROXY\_HEADER, 344
  - IP\_PROXY\_OFFSET, 344
  - LEGAL\_TOS\_DATE, 344
  - LEGAL\_URL, 344
  - LICENSE\_EXTRA, 345
  - LICENSE\_FILTER, 345
  - LICENSE\_REQUIRED, 345
  - LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH, 346
  - LOCALIZE\_CDN\_PATH, 346
  - LOCALIZE\_CDN\_URL, 346
  - LOGIN\_REQUIRED\_URLS, 346
  - LOGIN\_REQUIRED\_URLS\_EXCEPTIONS, 346
  - MATOMO\_SITE\_ID, 347
  - MATOMO\_URL, 347
  - NEARBY\_MESSAGES, 347
  - PAGURE\_CREDENTIALS, 348
  - PAGURE\_TOKEN, 348
  - PAGURE\_USERNAME, 348
  - PIWIK\_SITE\_ID, 347
  - PIWIK\_URL, 347
  - PRIVACY\_URL, 348
  - PRIVATE\_COMMIT\_EMAIL\_OPT\_IN, 349
  - PRIVATE\_COMMIT\_EMAIL\_TEMPLATE, 349
  - PROJECT\_BACKUP\_KEEP\_COUNT, 349
  - PROJECT\_BACKUP\_KEEP\_DAYS, 349
  - PROJECT\_NAME\_RESTRICT\_RE, 350
  - PROJECT\_WEB\_RESTRICT\_RE, 350
  - RATELIMIT\_ATTEMPTS, 350
  - RATELIMIT\_LOCKOUT, 350
  - RATELIMIT\_WINDOW, 350
  - REGISTRATION\_ALLOW\_BACKENDS, 351
  - REGISTRATION\_CAPTCHA, 351
  - REGISTRATION\_EMAIL\_MATCH, 351
  - REGISTRATION\_OPEN, 351
  - REPOSITORY\_ALERT\_THRESHOLD, 352
  - REQUIRE\_LOGIN, 352
  - SECRET\_KEY, 204
  - SENTRY\_DSN, 352
  - SERVER\_EMAIL, 205
  - SESSION\_COOKIE\_AGE\_AUTHENTICATED, 352
  - SESSION\_ENGINE, 204
  - SIMPLIFY\_LANGUAGES, 352
  - SINGLE\_PROJECT, 353
  - SITE\_DOMAIN, 353
  - SITE\_TITLE, 353
  - SPECIAL\_CHARS, 353
  - SSH\_EXTRA\_ARGS, 353
  - STATUS\_URL, 354
  - SUGGESTION\_CLEANUP\_DAYS, 354
  - UPDATE\_LANGUAGES, 354
  - URL\_PREFIX, 354
  - VCS\_API\_DELAY, 355
  - VCS\_BACKENDS, 355
  - VCS\_CLONE\_DEPTH, 355
  - WEBLATE\_ADDONS, 355
  - WEBLATE\_EXPORTERS, 356
  - WEBLATE\_FORMATS, 356
  - WEBLATE\_GPG\_IDENTITY, 357
  - WEBLATE\_MACHINERY, 357
  - WEBSITE\_REQUIRED, 357
  - setupgroups
    - weblate admin command, 383
  - setupgroups 命令列選項
    - no-privs-update, 383
    - no-projects-update, 383
  - setuplang
    - weblate admin command, 383
  - setuplang 命令列選項
    - no-update, 383
  - show
    - wlc 命令列選項, 152
  - SIMPLIFY\_LANGUAGES
    - setting, 352
  - SINGLE\_PROJECT
    - setting, 353
  - SITE\_DOMAIN
    - setting, 353
  - SITE\_TITLE
    - setting, 353
  - SPECIAL\_CHARS
    - setting, 353
  - SSH\_EXTRA\_ARGS
    - setting, 353
  - stats
    - wlc 命令列選項, 152
  - STATUS\_URL
    - setting, 354
  - store\_post\_load() (weblate.addons.base.BaseAddon 的方法), 418
  - string resources
    - file format, 82
  - SUGGESTION\_CLEANUP\_DAYS
    - setting, 354
- ## T
- translation
    - bilingual, 74
    - iOS, 83

- monolingual, 74
- TS
  - file format, 81
- U
  - unit\_pre\_create() (weblate.addons.base.BaseAddon 的方法), 418
  - unlock
    - wlc 命令列選項, 152
  - unlock\_translation
    - weblate admin command, 382
  - UPDATE\_LANGUAGES
    - setting, 354
  - updatechecks
    - weblate admin command, 383
  - updategit
    - weblate admin command, 383
  - upload
    - wlc 命令列選項, 153
  - URL\_PREFIX
    - setting, 354
- V
  - VCS\_API\_DELAY
    - setting, 355
  - VCS\_BACKENDS
    - setting, 355
  - VCS\_CLONE\_DEPTH
    - setting, 355
  - version
    - wlc 命令列選項, 152
- W
  - WEB\_WORKERS, 162, 178
  - Weblate (wlc 中的類 F), 155
  - weblate admin command
    - add\_suggestions, 374
    - auto\_translate, 374
    - celery\_queues, 375
    - checkgit, 375
    - cleanup\_ssh\_keys, 376
    - cleanuptrans, 375
    - commit\_pending, 375
    - commitgit, 375
    - createadmin, 376
    - dump\_memory, 376
    - dumpuserdata, 377
    - import\_demo, 377
    - import\_json, 377
    - import\_memory, 378
    - import\_project, 378
    - importuserdata, 380
    - importusers, 380
    - install\_addon, 380
    - list\_languages, 381
    - list\_translators, 381
    - list\_versions, 381
    - loadpo, 381
    - lock\_translation, 382
    - move\_language, 382
    - pushgit, 382
    - setupgroups, 383
    - setuplang, 383
    - unlock\_translation, 382
    - updatechecks, 383
    - updategit, 383
  - WEBLATE\_ADDONS
    - setting, 355
  - WEBLATE\_ADMIN\_EMAIL, 162, 163
  - WEBLATE\_ADMIN\_NAME, 162, 163
  - WEBLATE\_ADMIN\_PASSWORD, 158, 162–164
  - WEBLATE\_AKISMET\_API\_KEY, 390
  - WEBLATE\_ALLOWED\_HOSTS, 204, 208, 353
  - WEBLATE\_API\_RATELIMIT\_ANON, 105
  - WEBLATE\_API\_RATELIMIT\_USER, 105
  - WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD, 169
  - WEBLATE\_DEBUG, 228
  - WEBLATE\_EMAIL\_HOST\_PASSWORD, 176
  - WEBLATE\_EMAIL\_PORT, 175, 176
  - WEBLATE\_EMAIL\_USE\_SSL, 175, 176
  - WEBLATE\_EMAIL\_USE\_TLS, 175, 176
  - WEBLATE\_ENABLE\_HTTPS, 238
  - WEBLATE\_EXPORTERS
    - setting, 356
  - WEBLATE\_FORMATS
    - setting, 356
  - WEBLATE\_GITEA\_HOST, 166, 340
  - WEBLATE\_GITEA\_TOKEN, 340
  - WEBLATE\_GITEA\_USERNAME, 340
  - WEBLATE\_GITHUB\_HOST, 166, 227, 342
  - WEBLATE\_GITHUB\_TOKEN, 342
  - WEBLATE\_GITHUB\_USERNAME, 342
  - WEBLATE\_GITLAB\_HOST, 166, 341
  - WEBLATE\_GITLAB\_TOKEN, 341
  - WEBLATE\_GITLAB\_USERNAME, 341
  - WEBLATE\_GPG\_IDENTITY
    - setting, 357
  - WEBLATE\_LOCALIZE\_CDN\_PATH, 177
  - WEBLATE\_MACHINERY
    - setting, 357
  - WEBLATE\_PAGURE\_HOST, 166, 348
  - WEBLATE\_PAGURE\_TOKEN, 348
  - WEBLATE\_PAGURE\_USERNAME, 348
  - WEBLATE\_RATELIMIT\_ATTEMPTS, 392
  - WEBLATE\_SECURE\_PROXY\_SSL\_HEADER, 165
  - WEBLATE\_WORKERS, 162, 178
  - WeblateConfig (wlc.config 中的類 F), 156
  - WeblateException, 155
  - WEBSITE\_REQUIRED
    - setting, 357
  - wlc, 150
    - 模組, 155
  - wlc 命令列選項
    - author-email, 153
    - author-name, 153

- config, 151
- config-section, 151
- convert, 153
- format, 151
- fuzzy, 153
- input, 153
- key, 151
- method, 153
- output, 153
- overwrite, 153
- url, 151
- changes, 152
- cleanup, 152
- commit, 152
- download, 153
- list-components, 152
- list-languages, 152
- list-projects, 152
- list-translations, 152
- lock, 152
- lock-status, 152
- ls, 152
- pull, 152
- push, 152
- repo, 152
- reset, 152
- show, 152
- stats, 152
- unlock, 152
- upload, 153
- version, 152
- wlc.config
  - 模組, 156
- wlc.main
  - 模組, 156
- WPF
  - file format, 88

## X

- XLIFF
  - file format, 77
- XML
  - file format, 90

## Y

- YAML
  - file format, 89
- YAML Ain't Markup Language
  - file format, 89



模組

- wlc, 155
- wlc.config, 156
- wlc.main, 156



環境變數

- CELERY\_BACKUP\_OPTIONS, 162, 178
- CELERY\_BEAT\_OPTIONS, 162, 178
- CELERY\_MAIN\_OPTIONS, 162, 178
- CELERY\_MEMORY\_OPTIONS, 162, 178
- CELERY\_NOTIFY\_OPTIONS, 162, 178
- CELERY\_TRANSLATE\_OPTIONS, 162, 178
- POSTGRES\_ALTER\_ROLE, 174
- POSTGRES\_CONN\_MAX\_AGE, 174
- POSTGRES\_DATABASE, 173
- POSTGRES\_DISABLE\_SERVER\_SIDE\_CURSORS, 174
- POSTGRES\_HOST, 173
- POSTGRES\_PASSWORD, 173
- POSTGRES\_PASSWORD\_FILE, 173
- POSTGRES\_PORT, 173
- POSTGRES\_SSL\_MODE, 174
- POSTGRES\_USER, 173
- REDIS\_DB, 174
- REDIS\_HOST, 174
- REDIS\_PASSWORD, 175
- REDIS\_PASSWORD\_FILE, 175
- REDIS\_PORT, 174
- REDIS\_TLS, 175
- REDIS\_VERIFY\_SSL, 175
- ROLLBAR\_ENVIRONMENT, 177
- ROLLBAR\_KEY, 177
- SENTRY\_DSN, 177
- SENTRY\_ENVIRONMENT, 177
- SOCIAL\_AUTH\_SLACK\_SECRET, 172
- WEB\_WORKERS, 162, 178
- WEBLATE\_ADD\_ADDONS, 178
- WEBLATE\_ADD\_APPS, 177
- WEBLATE\_ADD\_AUTOFIX, 177
- WEBLATE\_ADD\_CHECK, 177
- WEBLATE\_ADD\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS, 166
- WEBLATE\_ADMIN\_EMAIL, 162, 163
- WEBLATE\_ADMIN\_NAME, 162, 163
- WEBLATE\_ADMIN\_PASSWORD, 158, 162–164
- WEBLATE\_ADMIN\_PASSWORD\_FILE, 163
- WEBLATE\_AKISMET\_API\_KEY, 167, 390
- WEBLATE\_ALLOWED\_HOSTS, 164, 204, 208, 353
- WEBLATE\_API\_RATELIMIT\_ANON, 105, 168
- WEBLATE\_API\_RATELIMIT\_USER, 105, 168
- WEBLATE\_AUTH\_LDAP\_BIND\_DN, 169
- WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD, 169
- WEBLATE\_AUTH\_LDAP\_BIND\_PASSWORD\_FILE, 169
- WEBLATE\_AUTH\_LDAP\_CONNECTION\_OPTION\_REFERRALS, 169
- WEBLATE\_AUTH\_LDAP\_SERVER\_URI, 169
- WEBLATE\_AUTH\_LDAP\_USER\_ATTR\_MAP, 169
- WEBLATE\_AUTH\_LDAP\_USER\_DN\_TEMPLATE, 169
- WEBLATE\_AUTH\_LDAP\_USER\_SEARCH, 169

- WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_FILTER, 169
- WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_UNION, 169
- WEBLATE\_AUTH\_LDAP\_USER\_SEARCH\_UNION\_DEFAULT, 169
- WEBLATE\_AUTO\_UPDATE, 176
- WEBLATE\_AVATAR\_URL\_PREFIX, 168
- WEBLATE\_BASIC\_LANGUAGES, 168
- WEBLATE\_BORG\_EXTRA\_ARGS, 168
- WEBLATE\_CONTACT\_FORM, 164
- WEBLATE\_CSP\_CONNECT\_SRC, 167
- WEBLATE\_CSP\_FONT\_SRC, 167
- WEBLATE\_CSP\_IMG\_SRC, 167
- WEBLATE\_CSP\_SCRIPT\_SRC, 167
- WEBLATE\_CSP\_STYLE\_SRC, 167
- WEBLATE\_DATABASE\_BACKUP, 174
- WEBLATE\_DEBUG, 163, 228
- WEBLATE\_DEFAULT\_ACCESS\_CONTROL, 167
- WEBLATE\_DEFAULT\_AUTO\_WATCH, 168
- WEBLATE\_DEFAULT\_COMMITTER\_EMAIL, 167
- WEBLATE\_DEFAULT\_COMMITTER\_NAME, 167
- WEBLATE\_DEFAULT\_FROM\_EMAIL, 164
- WEBLATE\_DEFAULT\_PULL\_MESSAGE, 167
- WEBLATE\_DEFAULT\_RESTRICTED\_COMPONENT, 167
- WEBLATE\_DEFAULT\_SHARED\_TM, 167
- WEBLATE\_DEFAULT\_TRANSLATION\_PROPAGATION, 167
- WEBLATE\_EMAIL\_BACKEND, 176
- WEBLATE\_EMAIL\_HOST, 175
- WEBLATE\_EMAIL\_HOST\_PASSWORD, 175, 176
- WEBLATE\_EMAIL\_HOST\_PASSWORD\_FILE, 176
- WEBLATE\_EMAIL\_HOST\_USER, 175
- WEBLATE\_EMAIL\_PORT, 175, 176
- WEBLATE\_EMAIL\_USE\_SSL, 175, 176
- WEBLATE\_EMAIL\_USE\_TLS, 175, 176
- WEBLATE\_ENABLE\_AVATARS, 168
- WEBLATE\_ENABLE\_HOOKS, 168
- WEBLATE\_ENABLE\_HTTPS, 164, 238
- WEBLATE\_ENABLE\_SHARING, 169
- WEBLATE\_EXTRA\_HTML\_HEAD, 169
- WEBLATE\_GET\_HELP\_URL, 176
- WEBLATE\_GITEA\_HOST, 166, 340
- WEBLATE\_GITEA\_TOKEN, 166, 340
- WEBLATE\_GITEA\_USERNAME, 166, 340
- WEBLATE\_GITHUB\_HOST, 166, 227, 342
- WEBLATE\_GITHUB\_TOKEN, 166, 342
- WEBLATE\_GITHUB\_USERNAME, 166, 342
- WEBLATE\_GITLAB\_HOST, 166, 341
- WEBLATE\_GITLAB\_TOKEN, 166, 341
- WEBLATE\_GITLAB\_USERNAME, 166, 341
- WEBLATE\_GOOGLE\_ANALYTICS\_ID, 166
- WEBLATE\_GPG\_IDENTITY, 167
- WEBLATE\_HIDE\_VERSION, 168
- WEBLATE\_INTERLEDGER\_PAYMENT\_POINTERS, 165
- WEBLATE\_IP\_PROXY\_HEADER, 165
- WEBLATE\_LEGAL\_URL, 176
- WEBLATE\_LICENSE\_FILTER, 167
- WEBLATE\_LICENSE\_REQUIRED, 167
- WEBLATE\_LIMIT\_TRANSLATION\_LENGTH\_BY\_SOURCE\_LENGTH, 168
- WEBLATE\_LOCALIZE\_CDN\_PATH, 177
- WEBLATE\_LOCALIZE\_CDN\_URL, 177
- WEBLATE\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS, 166
- WEBLATE\_LOGLEVEL, 163
- WEBLATE\_LOGLEVEL\_DATABASE, 163
- WEBLATE\_NO\_EMAIL\_AUTH, 173
- WEBLATE\_PAGURE\_HOST, 166, 348
- WEBLATE\_PAGURE\_TOKEN, 166, 348
- WEBLATE\_PAGURE\_USERNAME, 166, 348
- WEBLATE\_PRIVACY\_URL, 176
- WEBLATE\_PRIVATE\_COMMIT\_EMAIL\_OPT\_IN, 169
- WEBLATE\_PRIVATE\_COMMIT\_EMAIL\_TEMPLATE, 169
- WEBLATE\_RATELIMIT\_ATTEMPTS, 168, 392
- WEBLATE\_RATELIMIT\_LOCKOUT, 168
- WEBLATE\_RATELIMIT\_WINDOW, 168
- WEBLATE\_REGISTRATION\_ALLOW\_BACKENDS, 164
- WEBLATE\_REGISTRATION\_OPEN, 164
- WEBLATE\_REMOVE\_ADDONS, 178
- WEBLATE\_REMOVE\_APPS, 177
- WEBLATE\_REMOVE\_AUTOFIX, 177
- WEBLATE\_REMOVE\_CHECK, 177
- WEBLATE\_REMOVE\_LOGIN\_REQUIRED\_URLS\_EXCEPTIONS, 166
- WEBLATE\_REQUIRE\_LOGIN, 165
- WEBLATE\_SAML\_IDP\_ENTITY\_ID, 173
- WEBLATE\_SAML\_IDP\_IMAGE, 173
- WEBLATE\_SAML\_IDP\_TITLE, 173
- WEBLATE\_SAML\_IDP\_URL, 173
- WEBLATE\_SAML\_IDP\_X509CERT, 173
- WEBLATE\_SECURE\_PROXY\_SSL\_HEADER, 165
- WEBLATE\_SERVER\_EMAIL, 164
- WEBLATE\_SERVICE, 178
- WEBLATE\_SILENCED\_SYSTEM\_CHECKS, 167
- WEBLATE\_SIMPLIFY\_LANGUAGES, 167
- WEBLATE\_SITE\_DOMAIN, 163
- WEBLATE\_SITE\_TITLE, 163
- WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_OAUTH2\_KEY, 172
- WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_OAUTH2\_SECRET, 172
- WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_KEY, 172
- WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_SECRET, 172
- WEBLATE\_SOCIAL\_AUTH\_AZUREAD\_TENANT\_OAUTH2\_TENANT\_ID, 172



WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_OAUTH2\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_OAUTH2\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_BITBUCKET\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_FACEBOOK\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_FACEBOOK\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_FEDORA, 172  
 WEBLATE\_SOCIAL\_AUTH\_GITEA\_API\_URL, 171  
 WEBLATE\_SOCIAL\_AUTH\_GITEA\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_GITEA\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_KEY, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_KEY, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_NAME, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_ORG\_SECRET, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_SECRET, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_ID, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_KEY, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITHUB\_TEAM\_SECRET, 170  
 WEBLATE\_SOCIAL\_AUTH\_GITLAB\_API\_URL, 171  
 WEBLATE\_SOCIAL\_AUTH\_GITLAB\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_GITLAB\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_KEY, 171  
 WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_SECRET, 171  
 WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_WHITELISTED\_DOMAINS, 171  
 WEBLATE\_SOCIAL\_AUTH\_GOOGLE\_OAUTH2\_WHITELISTED\_EMAILS, 171  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_ACCESS\_TOKEN\_URL, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_ALGORITHM, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_AUTHORIZATION\_URL, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_IMAGE, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_KEY, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_PUBLIC\_KEY, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_SECRET, 172  
 WEBLATE\_SOCIAL\_AUTH\_KEYCLOAK\_TITLE, 172  
 WEBLATE\_SOCIAL\_AUTH\_OIDC\_KEY, 173  
 WEBLATE\_SOCIAL\_AUTH\_OIDC\_OIDC\_ENDPOINT, 173  
 WEBLATE\_SOCIAL\_AUTH\_OIDC\_SECRET, 173  
 WEBLATE\_SOCIAL\_AUTH\_OIDC\_USERNAME\_KEY, 173  
 WEBLATE\_SOCIAL\_AUTH\_OPENSUSE, 172  
 WEBLATE\_SOCIAL\_AUTH\_SLACK\_KEY, 172  
 WEBLATE\_SOCIAL\_AUTH\_UBUNTU, 172  
 WEBLATE\_SSH\_EXTRA\_ARGS, 168  
 WEBLATE\_STATUS\_URL, 176  
 WEBLATE\_TIME\_ZONE, 164  
 WEBLATE\_URL\_PREFIX, 167  
 WEBLATE\_WEBSITE\_REQUIRED, 167  
 WEBLATE\_WORKERS, 162, 178  
 WL\_BRANCH, 328  
 WL\_COMPONENT\_NAME, 328  
 WL\_COMPONENT\_SLUG, 328  
 WL\_COMPONENT\_URL, 328  
 WL\_ENGAGE\_URL, 328  
 WL\_FILE\_FORMAT, 328  
 WL\_FILEMASK, 328  
 WL\_LANGUAGE, 328  
 WL\_NEW\_BASE, 328  
 WL\_PATH, 328  
 WL\_PREVIOUS\_HEAD, 328  
 WL\_PROJECT\_NAME, 328  
 WL\_PROJECT\_SLUG, 328  
 WL\_REPO, 327  
 WL\_TEMPLATE, 328  
 WL\_VCS, 327