Translation Proxy Online Documentation

Release latest

Nov 13, 2018

Contents

1 Introduction **3**
- 1.1 Features . . . 4
- 1.2 Whitelabel . . . 5
- 1.3 Pricing . . . 5
- 1.4 Further Information . . . 5

2 Getting started **7**
- 2.1 Registration & Login . . . 7
- 2.2 Setting up Your First Project . . . 7
- 2.3 Running a Discovery . . . 10
- 2.4 Giving a Quote . . . 11
- 2.5 On the Workbench . . . 12
- 2.6 The 3-Phase Workflow . . . 12

3 The Dashboard **15**
- 3.1 Introduction . . . 15
- 3.2 Menu . . . 16
- 3.3 Cookbook . . . 74

4 The Dashboard 2.0 **111**
- 4.1 Complexity Matrix . . . 111

5 The Workbench **117**
- 5.1 Introduction . . . 117
- 5.2 Moving Around . . . 118
- 5.3 Views . . . 125
- 5.4 Translation of Segments . . . 129
- 5.5 Collaboration . . . 135
- 5.6 Cookbook . . . 137

6 Maintenance **141**
- 6.1 Project maintenance . . . 141
- 6.2 Automation possibilities . . . 141
- 6.3 Automatic pre-translation . . . 142
- 6.4 Caching And The Proxy . . . 143

7 Technical Reference **145**
- 7.1 Architecture . . . 145
- 7.2 Request Handling . . . 146

7.3 Classification of Content . . . 147
7.4 Translation Memories . . . 148
7.5 Using TMs . . . 148
7.6 Page modifiers . . . 149

8 Troubleshooting & Support **151**
8.1 Contacting Support . . . 151
8.2 Issues . . . 151

9 FAQ **159**
9.1 General . . . 159
9.2 Caches . . . 160

10 Glossary **163**
10.1 Page List Tags . . . 163


easyling

CHAPTER 1

Introduction

Easyling is a cloud-based translation proxy solution designed to make websites available in several languages. Easily.

A translation proxy is a layer placed between the browser and the original website, through which the visitor sees the original website in a different language.

There are several proxy solutions available on the market, yet the translation proxy is unique in that:

- it is a solution targeted primarily at LSPs - which also means that this is the **only** LSP-independent solution on the market
- also available as a whitelabel solution
- standard-compliant XLIFF export option ensures CAT-independence
- automated XLIFF export/import option between CAT / Translation management systems and Easyling
- in-context live preview available in certain CAT-tools at translation-time

What does that mean?

If you are a **business owner**, it can help you reach a wider customer-base through providing information to your potential customers in their native language. What's more, you can do more than just translating the text on your website, you can ***localize*** it: you can also adapt your message, the images displayed, or even your product range offered to the targeted culture. And all this without the need of heavy upfront investment in IT infrastructure and personnel, and the hassle with regular maintenance and upgrade. the translation proxy takes care of the IT part, so that you could concentrate on the content - and growing your business.

If you are a **language service provider** (LSP), you can offer cutting-edge website localization services to your customers - even under your own brand name! the translation proxy provides the technology, takes care of the IT infrastructure, leaving you to concentrate on your core business: cross-cultural communication. What's more, your translators don't need to learn using just another tool, they can keep using their own preferred CAT-tools.

Sounds good?

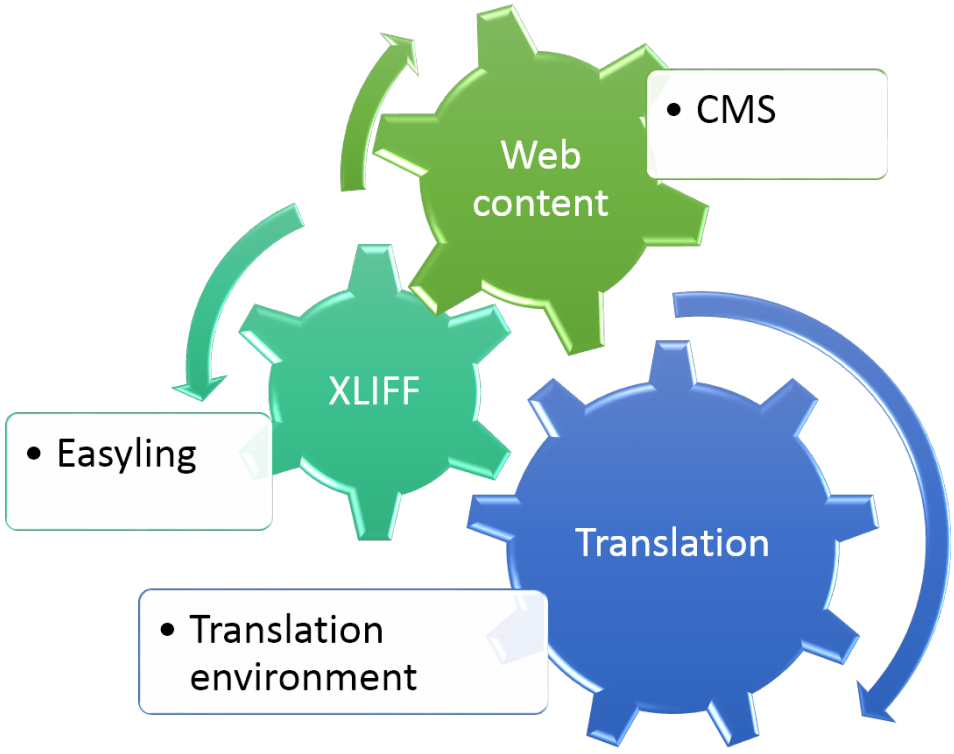
There are several challenges both business owners and language service providers face during website translation. The "ideal" workflow would be to create the content in the original language, get it translated into the desired languages, and then publish all language variants at the same time, from the website owner's own content management system (CMS) - right from the very first page on the website. But reality is different. Apart from the fact that not many CMSs are capable of handling several languages, usually website localization comes into the picture at a later stage, when there is already a huge amount of data published on the website. And, in most cases, the website owner can't extract the content for translation. If they can't extract the original, there's no easy way to load the translated content back either. Furthermore, if the website owner can't extract the content into translatable format, it is impossible to get a proper estimate for the translation costs in time and money...

Easyling can, however, discover the website by following links and grabbing translatable and localizable content - and convert it into a translatable format. This gives a realistic view of the magnitude of the translation task, and, thanks to the translation proxy, even a partially translated site can give full user experience on the website visitor's side.

Data can be extracted with a couple of clicks - and the publication of the translated site is similarly easy.

Where do we fit in the localization workflow

Easyling bridges the gap between the CMS and the translation workflow; it enables you to extract content from the website in a translation-ready format that can be used in ***any*** translation environment.



- If you are a content owner, it means that you have a technology solution that enables you to choose any LSP that suits your language requirements the best.
- If you are an LSP, it means that you can take care of the website translation requirements of your clients.

In either case, technology will no longer be the bottleneck.

Features

- process HTML, JavaScript/AJAX/JSON, XML (note: translation of Flash is not supported)
- Use the X-Proxy and other Preview mode domains to see everything in context. use Advanced Settings to translate text coming from JSON/XML sources.
- Automatically crawl static pages / HTML content only. Add extra AJAX URLs with the proper parameters.
- Fine-tune your settings to help the crawler decide what URLs to handle as same, and what to visit looking for new content.
- Translate forms, messages and dynamic content

- Translate images: replace them with their localized counterparts on a target-language basis.
- Link any external domain via multiple projects.
- Modify page behavior using customized CSS and JavaScript injection.
- Use regular expressions to filter for content.

and many more!

Whitelabel

Easyling offers a white label version that can be customized with your corporate logos and domains to create a branded version, allowing you to use & sell the translation proxy as your own product. In order to create the branded version, seven criteria must be met. See the "Whitelabel setup" section for details.

Pricing

Our pricing follows the 'pay-as-you-go' model, so you only get charged for what you use. The total cost is made up 2 types of fees: one-time fees and a monthly charge.

One-Time Fees

- **Discovery**: 1 EUR or 1.2 USD / 1000 pages for every Discovery
- **Scan**: 2 EUR or 2.4 USD / 1000 source words for each **new** word during a Scan (if no new words are found, a Scan counts as a Discovery)
- **Translation Memory**: storing your (human or machine) translation in the database costs 10 EUR or 12 USD / 1000 source words / target language

Content extraction and translation memory fees apply only the first time around, which means 102% repetitions are not counted. Once a segment is stored, subsequent scans will treat it as repetition and **no additional charges** will apply.

Monthly Fees

A 1 EUR or 1.2 USD / 1000 page views monthly fee applies to serving your tranlated content over the proxy.

In exchange, you get a guaranteed 99.99% website availability, and a capacity to handle practically unlimited traffic. You also have the option to serve the translated site from your server (in which case no proxy fee will apply), but in this case availability and traffic handling depends on your infrastructure.

Further Information

Visit us at https://easyling.com and sign up for a dedicated demo if you are new to translation over the proxy!

See https://easyling.com/pricing for pricing details. Try our Price Calculator, too!

Contact Support: support@easyling.com

Be sure to check out all the other tutorials we've compiled:

Tutorials

- https://www.youtube.com/watch?v=S47kArNiJ1o
- https://www.youtube.com/watch?v=8VsBy2bGo64
- https://gitlab.com/easyling/wikis/home
- https://drive.google.com/open?id=0Bw53oZELMrf8V1FIUnhmNEtubTA
- http://lesson101.tutorial.easyling.com/
- http://lesson102.tutorial.easyling.com/
- http://lesson103.tutorial.easyling.com/
- http://lesson105.tutorial.easyling.com/

Release notes

- https://www.easyling.com/category/blogpost/release-notes/
- https://www.easyling.com/change-logs/

CHAPTER 2

Getting started

Let us give you a quick overview of how the proxy is used. In this section, we give you a quick introduction to the Dashboard and we'll also set up a simple project.

Registration & Login

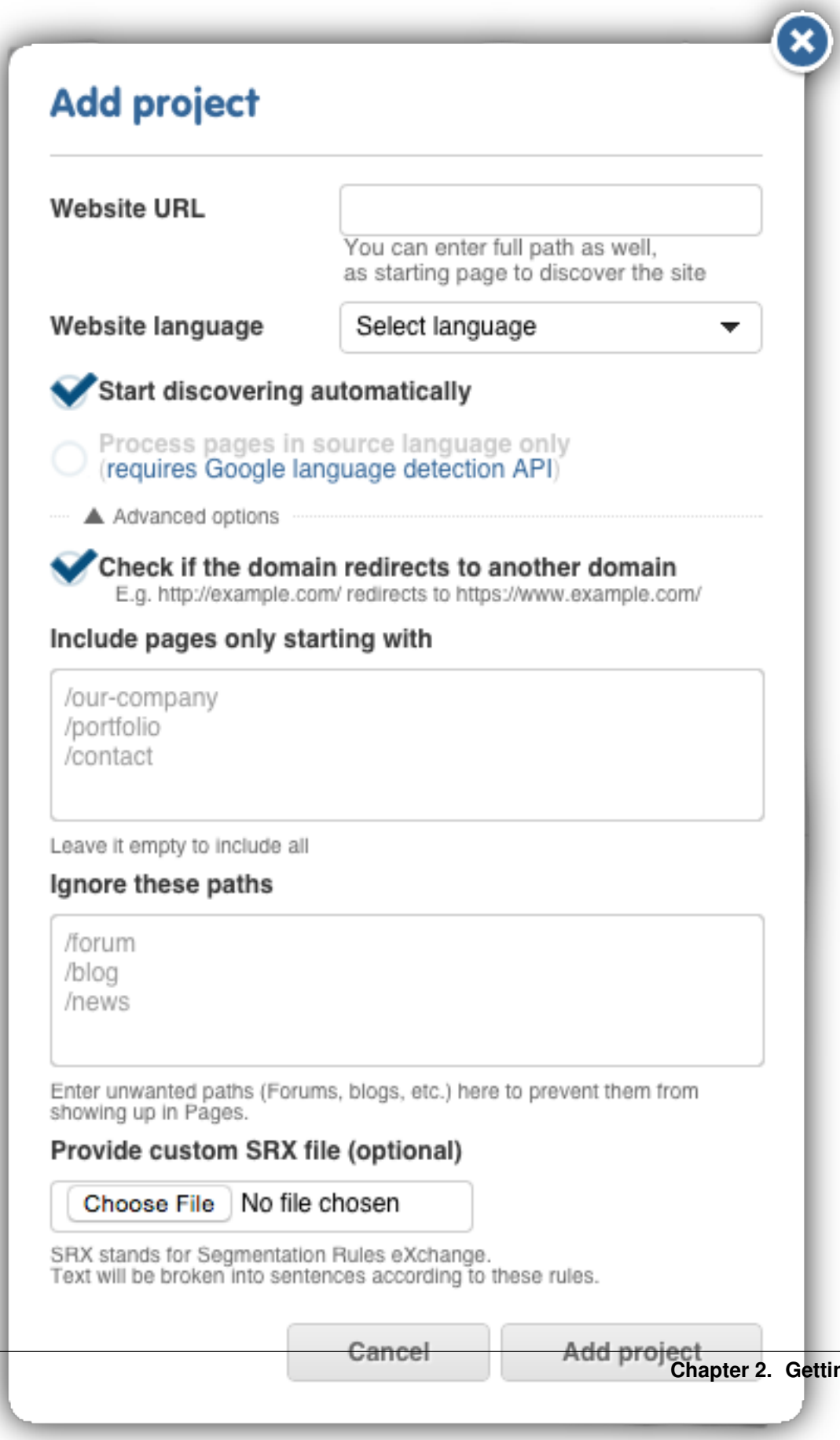
To use Easyling, you need to register and set up an account for the service at https://app.easyling.com. You can start using the service right away after registration.

Logging in, you will be taken to the Dashboard, which is the project management center, every detail of which we'll get to in later sections of this manual. There will be no existing projects at the outset, so let's try setting one up.

Setting up Your First Project

To do this, click on **Create new project** dropdown at the top and choose **Add project**.

This opens the **Add project** dialog box, where you can enter the URL of the website you would like to translate, and also select the website language; this sets the source language of the translation project. Click on Advanced Settings to access extended functionality, such as custom SRX files.


Add project
Website URL
You can enter full path as well,
as starting page to discover the site
Website language
Select language
Start discovering automatically
Process pages in source language only
(requires Google language detection API)
Advanced options
Check if the domain redirects to another domain
E.g. http://example.com/ redirects to https://www.example.com/
Include pages only starting with
/our-company
/portfolio
/contact
Leave it empty to include all
Ignore these paths
/forum
/blog
/news
Enter unwanted paths (Forums, blogs, etc.) here to prevent them from showing up in Pages.
Provide custom SRX file (optional)
Choose File
No file chosen
SRX stands for Segmentation Rules eXchange.
Text will be broken into sentences according to these rules.
Cancel
Add project

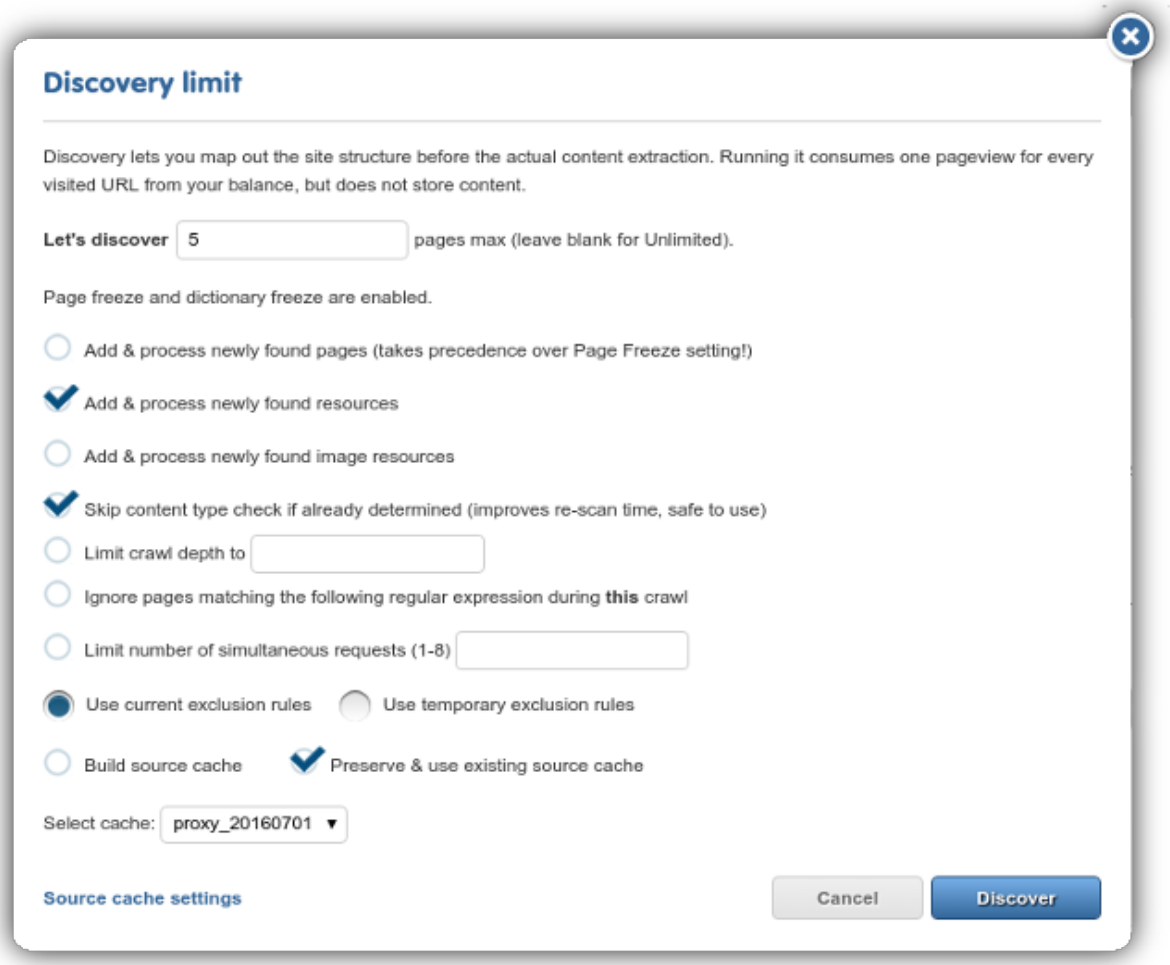
Add a Target Lanuage

You will also need to add your target language(s), so use the option on the Dashboard to add then to the project. It's not just that there is not much to do in terms of translation without a target: many crucial features (including the Preview proxy and the entire Workbench) are entirely unavailable as long as no target languages are set.



You can add any number of target languages. You can use the search function to lookup languages based on locale code or country name.

Running a Discovery



The next step is to figure out what (and how much of it) to translate and do it from a single target URL. For this, a Discovery has to be run on the site.

Discovering a website means running a `crawler` on it and allowing the proxy to 'explore' it in its entirety in order to provide Statistics for you. As you can see in the 'Add project' dialog window, the Dashboard is set up to automatically start a Discovery on a webpage after creating a project - but don't worry! After clicking on the **Add project** button, a new dialog will open where you get to set up additional details of the Discovery before really starting the process.

There are many details that potentially have to be taken into account when setting up a truly effective Discovery, but let's set those aside for the moment. For now, simply Click on the 'Add project' button and after the Discovery dialog opens, click on Discover to start a crawl on the website. **The default settings are safe.**

Depending on the size of the site, a Discovery can take quite some time to finish. A spinner on the Discovery page will indicate that the system is currently working, but there is a default page limit of 100, which means that if the Discovery finds more than one hundred pages, it will automatically exit, allowing you to fine-tune your settings.

After the process is over, you'll receive an e-mail about the results. You also have your first Statistics from the Discovery - a word count total from all Discovered pages.

Giving a Quote

You can use the results of the Discovery to give a quote (based on **unique** source words) to your clients about the estimated work-hours and expenses you forecast for a given project.

The results are an accurate indication of the translation costs associated with the text. However, with websites, it is prudent to **consider other (techincal) details** before taking the word count results of the Discovery process at face value.

Investigate the source site and consider the following:

- Is there a great deal of **dynamic content**?
- Any **Site Search** functionality? A **webshop**? A **forum**?
- Any other **web apps** that would have to be localized?
- Do the **average word lengths** of the source and target languages differ significantly?
- Is the **direction of the target language** different than that of the source language?
- Which pages are targeted for translation? Which pages need to be excluded? Ask the client if they have a specific page list.
- does the site have mixed-language content? If yes, ask the client to specify the source language(s) they need translated.
- is there an extant Translation Memory that could be used?
- is there any region-specific content? Does the site use geolocation?
- is there any content behind a secure login?
- are there any subdomains? example.com and blog.example.com require two separate projects that need to be linked.
- are there any other special requirements?
- is there any Javascript-generated content?

If you answered yes to any of those questions, that will require some deliberation, often beyond the primary focus of translators: UI fixes and a measure of fiddling with what's under the hood - take those expenses into account when you make your quote.

NOTE!: If you are unsure as to how to go about translating a part of a website, feel free to contact our **Support Centre** and we'll help you get an accurate picture of the required effort and costs.

It is also advised to negotiate the expected workflow with the client at the quoting phase. The translation of a website is, in most cases, a never-ending process, as new content is added to the original site at certain intervals.

The question is, how content added *after* the initial quote should be treated - both from a technological and fiscal viewpoint. It is a good idea to ask the client about their intentions for update cycles and new content.

Do they wish to publish at the same time in all languages? Or publish on the original site without delay, and then publish the translations later, as they become available? The different options will require different approaches when you get to the maintenance phase of the project.

As a translation proxy is practically a translation layer on top of the original site, serving translations from the datastore by replacing original content on-the-fly, new content will not be replaced, as translation is not stored for that. In practice it means that newly added content will appear in the original language on the translated site. This is called *bleedthrough*. There are 2 approaches to this phenomenon: let bleedthrough happen, to make new content available right away, even if it is in a different language, or block new content from appearing until translation is done. Both have their clear advantages and drawbacks, so you have to discuss with your client which option is more acceptable for them - and set up your project accordingly.

Sales tool for mass production

Easyling also offers a Sales Tool to help LSPs and freelancers in growing their business.

If you have a well-defined group of potential customers you'd like to offer your translation services to, like hotels or restaurants with only monolingual websites in your area, the translation proxy makes it easy for you to impress the business owners. Just collect the URL addresses, add them to the Sales Tool, and the translation proxy will automatically create a project for all webpages according to the settings you specify. Once the translation and post-editing of the translated main pages are ready, you can send a link to the business owners. If your potential clients are impressed with the translated page and the fact that no IT involvement is required on their end, you have a better chance to win the deal.

For more information see our blog or this short video

On the Workbench

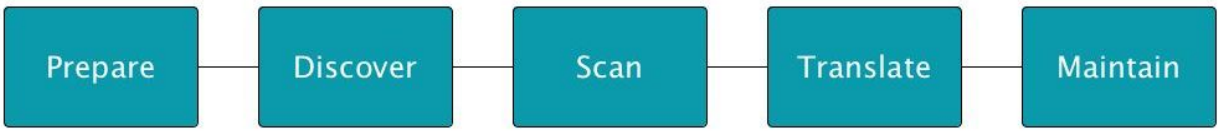
You can export all source segments, translate them in your CAT tool of choice and then reimport your results. But going through that cycle for every small change would get rapidly tedious - wouldn't it be great if you could edit & control your translations in the cloud, where it would all update in real time? You're looking for the Workbench.

In Pages View, you can hover over any page - a menu will show up right next to it - choose 'Translation in List View' and you'll be taken to the Workbench in a different tab.

If the Dashboard is the Project Management Center in Easyling, then the Workbench is the cloud CAT tool, where translation itself takes place. There are many features you can use in the Workbench to make working with websites easier - see the 'Workbench' section of this manual for the details.

The 3-Phase Workflow

Barring some detail (withheld for the sake of a convenient introduction) the above process is all that you need to get a website translation project going.



Our idea of a project's lifetime can be summarized in the 3-phase Workflow.

1. Discover & Quote

Set up a project and Discover it. Have a Unique Word Count total and a general idea of any technical issues involved. Give your quote to the client (perhaps demo/impress them via the Live Preview). Win the bid.

2. Ingest & Translate

After you are entrusted with the project, collect all text content into a database (overcoming any technical issues that may arise in the process). When you have your data, export it to your CAT tool of choice or translate in the Workbench to a selection of target languages. Reimport and edit. Use our Proofreading and Workflow features to ensure quality.

3. Publish & Maintain

After the translation is greenlit by the proofreaders, you can verify the serving domain and publish the translated website. Add a language selector to the source site. Generally, it is with publishing a website that a deadline is met.

But don't forget that a website is a living thing, with new content arriving every day - the final stage of website localization is always maintenance - making sure that new content gets translated according to schedule, all the while ensuring that visitors to the site will not be inconvenienced by bleedthrough of untranslated content.

Maintenance is the "long tail" of website translation - there are a variety of features in the proxy that make it a lot easier than it would be anyway else.

In the following pages, You will find everything there is to know about using the proxy. Keep reading!

CHAPTER 3

The Dashboard

Introduction

The Dashboard is your command center. It contains a variety of features you can use to manage your projects. In this manual, we'll take these options in the order that they appear in the menu on the left side of the screen.

When you open the Project Dashboard for the first time, the screen will display a few general settings described below.

Project Alias

This alternative name will be displayed in the Project dropdown below the URL, for easy identification of your projects. Project aliases are project-internal, they will not be displayed anywhere on the translated site.

Website Address

Exactly what it says on the tin, the website address is a property of your project that cannot be altered once declared during project creation.

There is one exception to this rule: by default, the proxy will follow redirects from the initial URL and will create the project for the address it is redirected to.

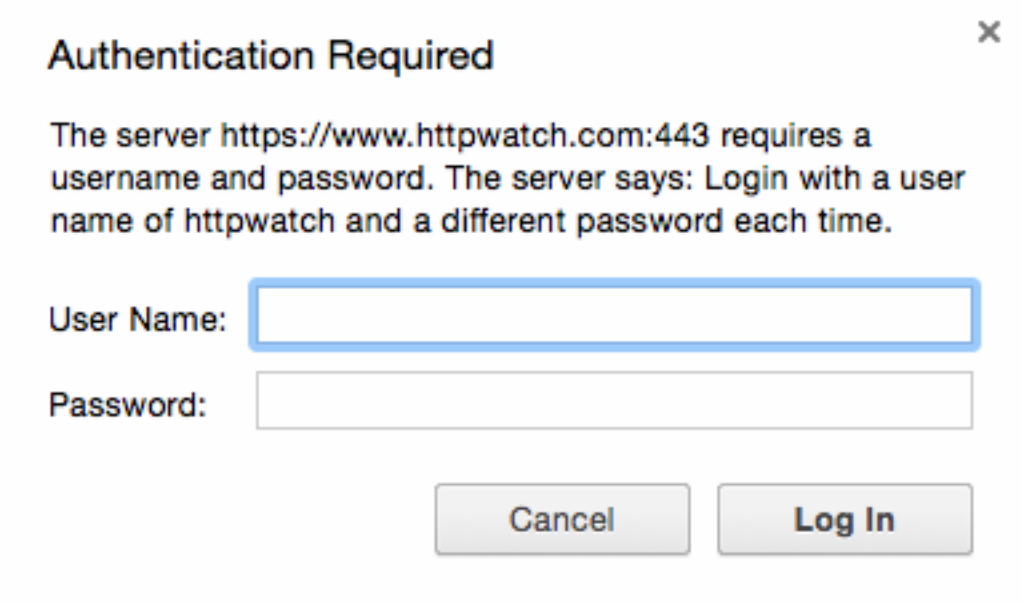
Alternative Domains

It happens sometimes that a website serves content both on the `www` subdomain and the naked domain, such as `example.com`. In these cases, it is useful to set things up over the proxy so that the different URLs are handled similarly.

After creating a project, this field is automatically filled with the complement of the Website Address. Add any further subdomains that contain identical content to this list. Separate them with commas.

Basic Authentication

Not to be confused with the project Access Control features of the proxy, the Basic Authentication username and password fields can be used for automatic authentication on the project website. Basic Authentication windows typically look like this:



If a username and password is provided on the Dashboard, the proxy will rely on this information from then on. Use this option to enable Discoveries and Scans to work properly on these sites. The various Preview proxy modes will also rely on this authentication info to get past the login screen automatically.

Project workflow

Change the number of project participants and project workflow type using this dropdown. See Collaboration.

Staging Domain

Although it is true that the project address cannot be changed after the project is created, the Staging Domain feature can still be used to change the origin server to which requests are sent.

For details, please see the Cookbook recipe on Staging domains

Language Selector

Select one or more from the available target languages to translate source language content.

Most useful translation facilities, such as the Workbench or the various instant preview features remain unavailable as long as there is no target language on a project.

You can remove or add target languages at any time. Note, however, that it is *not recommended* that you change the source language on a project that is published with translations.

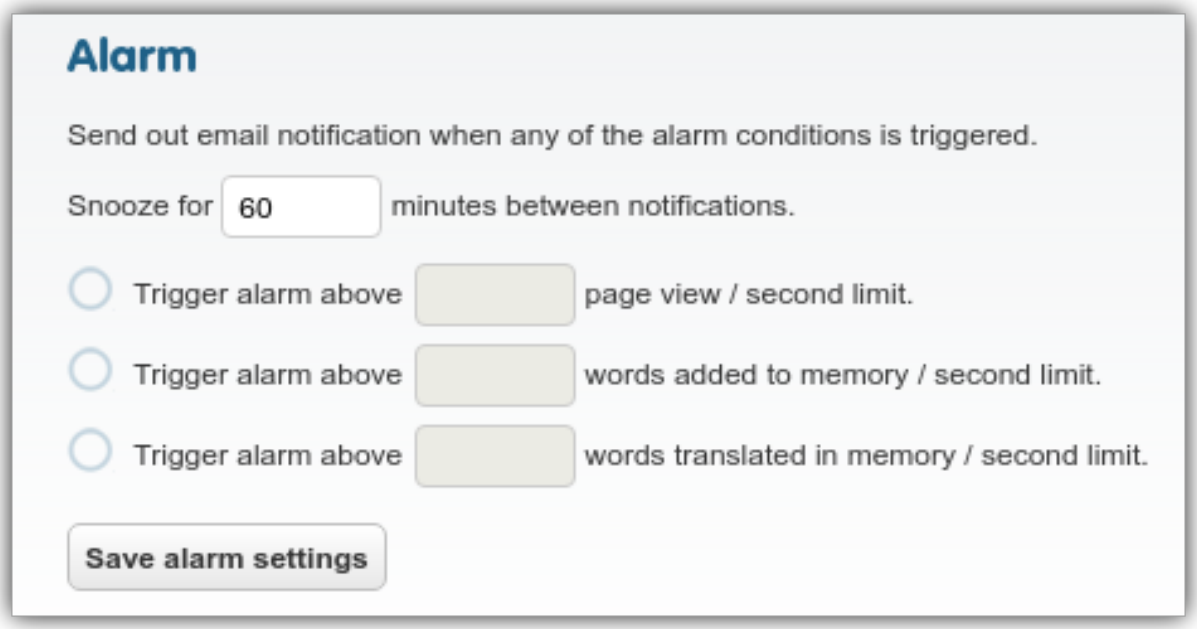
Menu

Dashboard

Project statistcs

Please see the "Statistics" section in Discovery for a detailed explanation of Project statistics.

Alarm



You can set up alarms to be notified of any actions in your project that could potentially lead to undue expenses. Enter the value for the timeout (snooze) between subsequent e-mail notifications, to prevent the Alarm from flooding you with e-mails. After that, check one of the alarm options and enter the limit above which you want to receive a notification. You can set an alarm for:

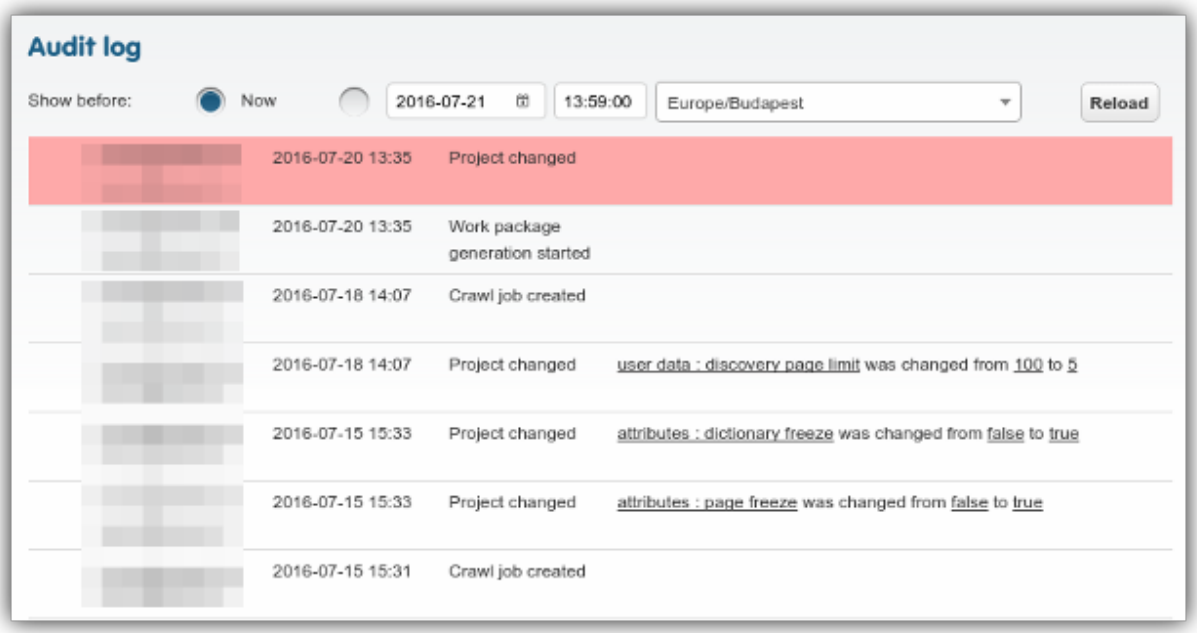
Page Views: if you wish to detect unusually large amounts of requests to the proxy. As Scans and Discoveries count as Page Views, you can also use this alarm to detect if a Scan was started.

Words added: if you wish to receive notification about new content that exceeds the set amount.

Words translated: if you wish to receive notification about Machine Translation or reimport events on the project.

The e-mails are sent to the owner by default, but you can add the 'Receive notification emails' right to any participant in Dashboard > Sharing settings.

Audit log



The audit log is the minutes-taking feature of Easyling. It records all activity on the project with a username/e-mail address attached to each entry.

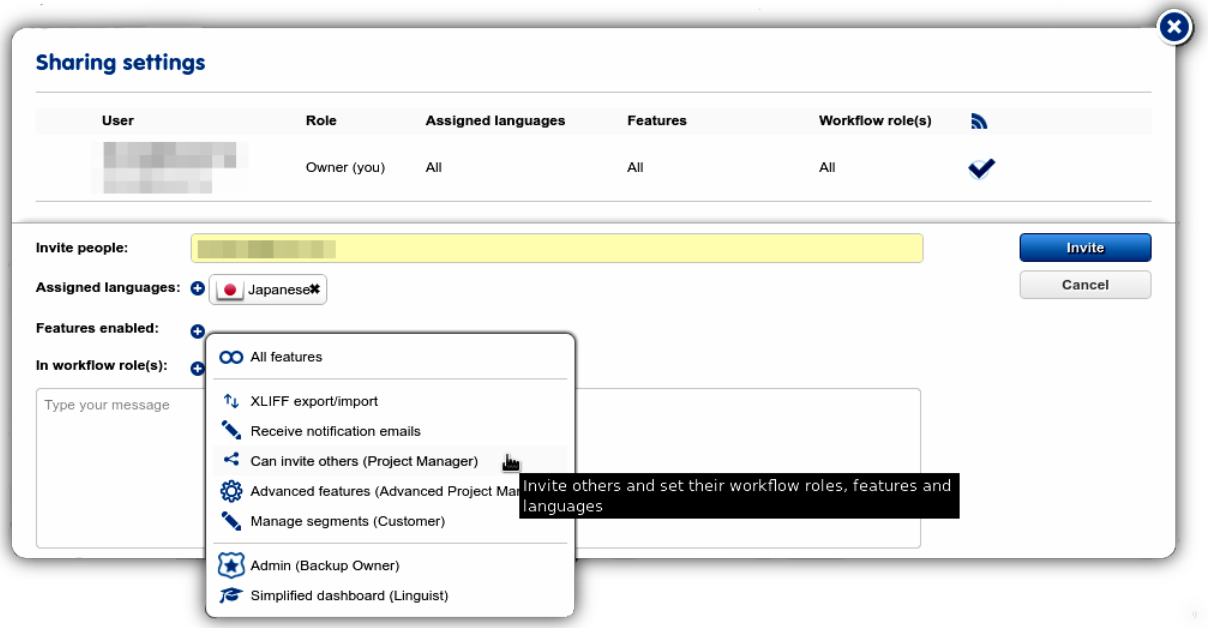
Some Audit Log entries can get quite long, such as when Page Modifier or CSS overrides are being edited and saved multiple times. For this reason, the Audit log displays onlythe first three lines of long entries by default. Use the grey bar with the downward-pointing arrow under a log entry to see the full text of that entry.

Sharing settings

In the world of website localization, it is rare for a project to be realized by a single person. As project owner or backup owner, use the Sharing settings dialog to invite people into the project and assign roles/editing roles to any one of them.

At the bottom of the Dialog, you will see the line, “Invite People:” if you click on the field, the invitation settings part of the dialog will slide up, allowing you to enter further details of the project participant to be added. You can assign languages, enable admin/access features and workflow roles for each user you invite. You can do this by clicking on the blue plus icon next to each of those options. Make your choices in the small pop-up lists. **These settings are not final, you can change them later.**

Features



The various **features**, (administrative rights) deserve to be detailed, Here they are:

All features: every feature you see below the separator in the list will be added for the user. Note that this has a potential to cause problems: If, for example, you simultaneously set **Advanced features** and **Manage segments**, it results in an editing rights conflict. There are security implications you need to consider when giving your users rights, so use this feature with care.

XLIFF export/import: enables XLIFF export/import of segments for the user. Especially important for those users who will be coordinating the translation effort throughout the project.

Receive notification e-mails: Add user to the list of recipients of Discovery/Content Scan/Alarm, etc notifications.

Can invite others: designates the user as a Project Manager, able to invite others onto the project. Take care not to include other, conflicting roles, such as **Customer**, which could re-restrict access.

Advanced features: give the user the power to edit languages as well as any entry in the project, and editing most setting, up to, and including, the URL inclusion-exclusion rules. Segmentation, publishing and certain advanced settings remain beyond the reach of the user.

Manage segments: make the user a Customer, with the ability to manage (approve or exclude) pending source segments. The user will also require a workflow role and an assigned target language to access the Workbench to manage segments. **WARNING! This right restricts the user to this role, other features will be disabled!**

NOTE Features are predominantly important from a project management viewpoint. **Workflow roles**, on the other hand, are used to manage user interaction **on the Workbench**, based on expertise (whether a user is a translator, proofreader or client will decide what sets of segments are available to them for editing - see the documentation of the Workbench for details).

After a user redeems their invitation using the link they receive in an e-mail, their address and username will appear under the owner. Their features and workflow roles can be edited in the list.

You can invite as many participants to the translation project as you'd like. You can also enable e-mail notifications for any of them by checking the notification icon in the last column.

Project Roles

Based on the project features that are available for a user, the following project ropes are available:

- **Owner**: Every project has an owner with unlimited powers over the project: the owner may add or remove anyone on the project, edit any entry in any language, including adding new languages, any change any setting, including the advanced ones. There can only be one owner on a project, but owners may renounce ownership, designating another user and setting their own privileges.
- **Backup Owners**: same as project owners. They can change any setting and entry, add or remove users, and modify language settings.
- **Project Managers**: can invite others onto the project. Other features and roles can be added as well. Take care not to include other, conflicting roles (restrictions override rights)
- **Advanced Project Managers**: they can edit languages, as well as any entry in the project. They can change most setting, up to and including the URL inclusion-exclusion rules. They cannot change segmentation rules or publishing settings. A few Advanced settings are also beyond their reach.
- **Linguists**: have access to the Simplified Dashboard and Content menu. They are only allowed to edit segments in their designated language and workflow role. They can receive notification emails about project updates, and may be given the power to import/export XLIFF files.
- **Contributors**: default users, capable of editing any entry in their selected language and workflow role, but nothing else. They may receive notification emails and project update emails, but they may not edit their features, nor invite anyone else, nor access any of the advanced settings.
- **Customer**: This role intended for a representative of the customer who excercises oversight over content. This role is read-only, meaning despite the target language added to the user, they are unable to actually make edits. They can *Approve* or *Exclude* segments.

Client Approval

Sometimes there is need for the client to view and check new segments before they enter the translation workflow.

In the **Advanced Settings** menu, default state of new segments can be set to the following:

- Pending
- Excluded
- Approved

When set to Pending or Excluded, new segments picked up will acquire that state automatically and manual approval is needed. In the Sharing Settings menu, the customer role can be assigned for this purpose.

Important: The client role is read-only. Anyone assigned this role won't be able to edit content, but approve them for translation. If you are unable to edit segments on the Workbench, please check Sharing Settings first.

Link projects

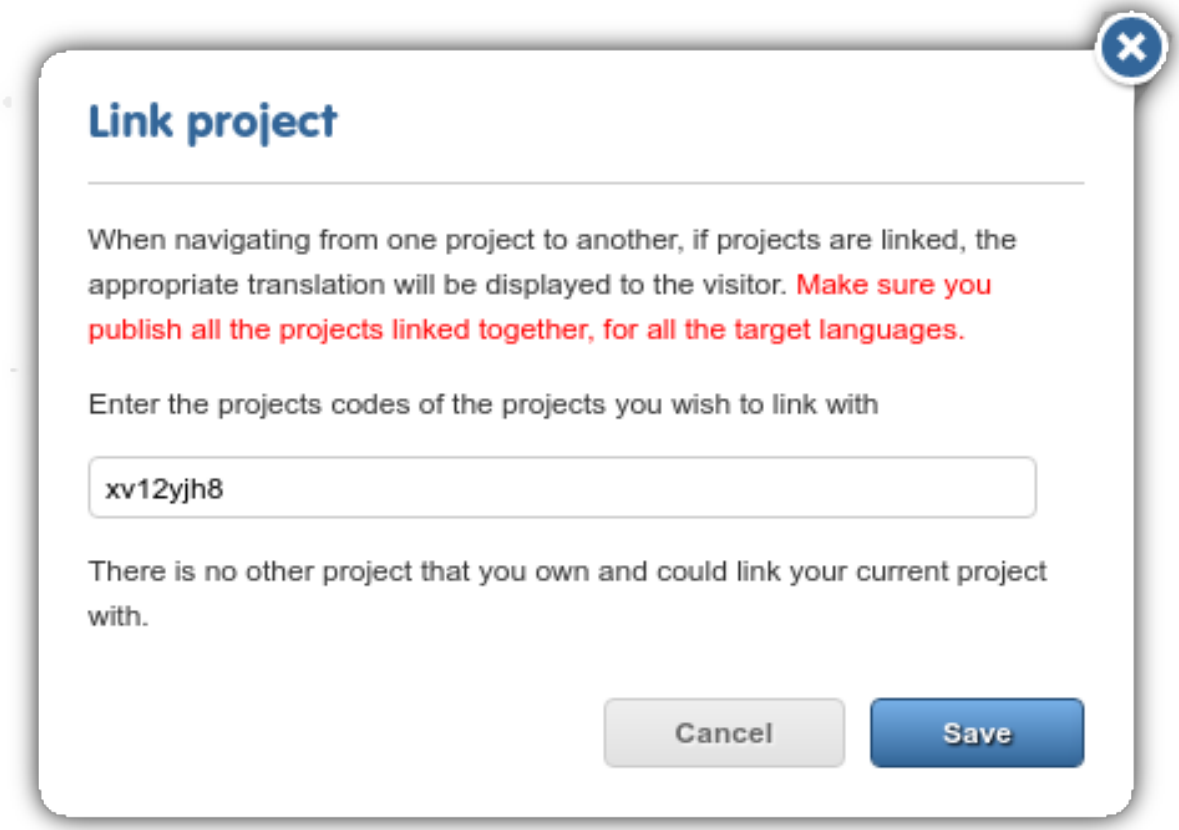
A proxy project supports translation of content on the project for which it was created, or the staging server that is set up for it. Any content coming from domains other than these has to be translated via a separate project.

`example.com` and `blog.example.com` are separate domains and separate projects are required to create an integrated translation for the whole.

Provided that each is set up appropriately, the proxy can make such related projects aware of each other using the Project Linking feature. This **enables the cross-mapping of URLs on the linked projects**. Where `example.com`

referred to `blog.example.com` before, after linking, all URLs pointing to the latter will be mapped to its published target language domain.

The dialog for linking consists of a single input field:



Copy & paste the linked project's project code into the field. Linking requires changes on each affected project, so it is recommended that you open all of them in separate tabs for this process.

Some projects are complex enough to warrant linking multiple auxiliary projects to them. You can cadd multiple projects codes to the field, separated by a colon.

A few restrictions apply to project linking:

- project linking is **two-way**: project A has to be linked on project B and then project B has to be linked on project A to complete the process.
- **refresh** the corresponding Dashboards of each project after saving in the linking dialog to ensure that each project is notified of the attempt to or completion of the linking.
- only a user who is **owner of all involved projects** can link them.
- linked projects have to be **published together** and with the **same target languages**.

Project Linking & Forms

Form solutions tend to use content from external domains. The localization of **HubSpot forms**, for example, is a frequent and somewhat challenging use case of project linking.

The topic of translating such content using proxy features only is covered in detail on this cookbook page.

Page Caches

Source Caches

SCs are your first line of defense against bleedthrough: it is a way of storing and reusing content that was received and picked up from the **remote server** (in other words, the **source site**). The goal of Source Caches is to create a *controlled snapshot* of the source content that you can safely use to translate and serve content over the proxy.

You might want to set up one Source Cache for publishing content and another to be used during translation. The content of the Source Cache being published should be highly translated, and new content should be collected in a different Source Cache to be used in Preview mode and during translation.

Benefits of Source Caches

- If you have at least one Source Cache enabled and built, you have essentially decoupled new content from what you are serving over the proxy (you are in control of untranslated content).

  Note: Source Caches can only be built by Scans. They will *never* pick up new content unless you expressly run a Content Extraction to update them.

  See Case Study #1 for an in-depth coverage of this scenario.

- Multiple Source Caches let you decouple published content from untranslated material - you are free to update any Source Caches you are not currently using for published content over the proxy. This way, you are always serving 100% translated content (using the Cache that is highly translated), and you can schedule for translation updates (on all the others you are free to update anytime with a Scan).

  See Case Study #2 for an in-depth coverage of this scenario.

When to build?

The best time to build a Source Cache for the first time is during the first Content Extraction. By this time, you should have used Discovery a couple of times to gain a general understanding of the site structure and word count, and you have all necessary exclusion rules set up.

When you are confident in your Content Settings, set up a Source Cache before you run Content Scan (see below for setup directions) on the webpage and build it. This way, you will have a Source Cache that contains all content you're basing your quote on.

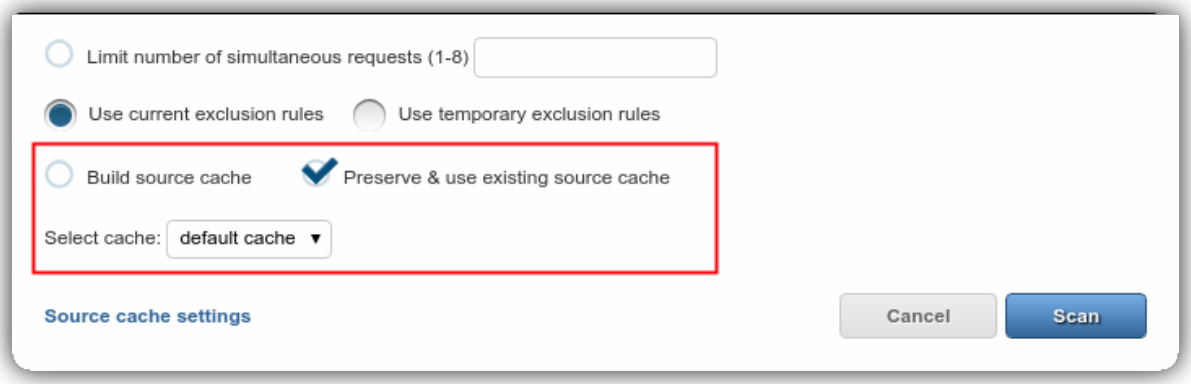
Setting up a Source Cache

You can enable and create a new Source Cache by checking **Enabled** in the *Page cache* menu in the Dashboard. Note that at this point, the SC is enabled but **unbuilt**.

WARNING!: You have to build the Source Cache before it becomes functional!

Building the SC

After enabling the Source Cache, you need to Scan for content to build it. In the Content Scan Dialog, some options become available:



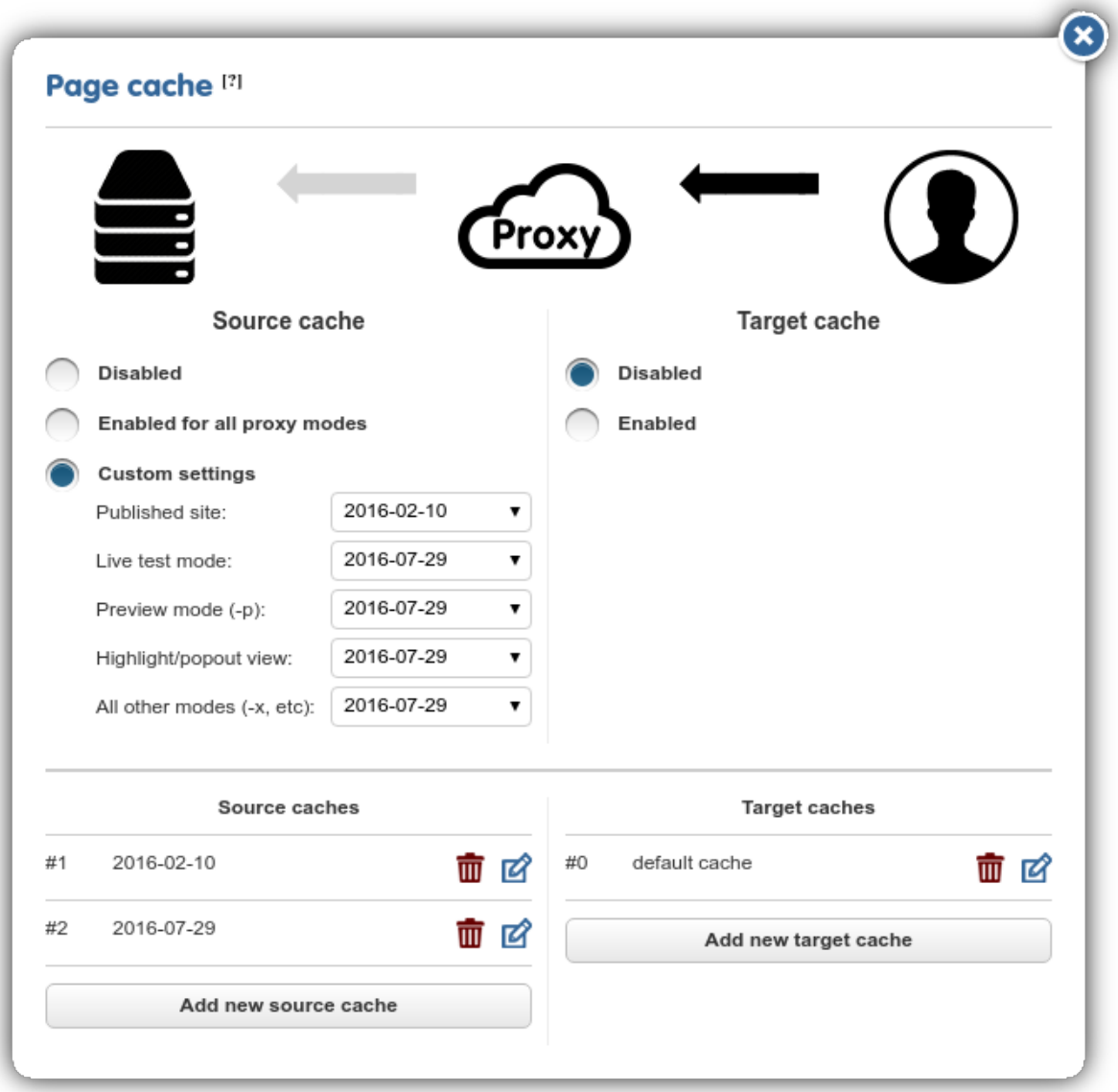
Check **Build source cache** and run the Scan. As a result of building the SC, all content that was set to be picked up by the Content Scan is added to the current SC.

If you are updating a SC, you can use the dropdown menu to select the cache you'd like to overwrite.

The '**Preserve & use existing source cache**' option ensures that a SC will not be written to during a Scan. Having both '**Build source cache**' and '**Preserve & use existing source cache**' enabled for a Scan will currently result in an inconsequential action (except for pages that are not in the current cache but referred - these will then be loaded into the cache).

Custom settings

If you have more than one Source Caches that you've built throughout multiple Content Scans, you can visit the Page Cache menu again and configure which Source Cache should be used for which proxy mode, as shown in the screenshot below:



Choose a highly translated Source Cache for publishing, so that you can be sure that visitors to the proxied site will not experience bleedthrough.

Meanwhile, you can use other, more up-to-date Source Caches for the preview/live test etc. proxy modes to work on the translations of new content.

Your Source Cache setup is complete.

Target Cache

Target cache option is used to achieve great boosts in page serving speed by enabling the proxy to skip processing the page instantly if the remote server's response matches the response that was used to generate the cache during building. By not having to insert the translations separately, page serving can be accelerated by several milliseconds, which can add up to a noticeable speedup in the case of larger pages with lots of translated content.

The Target cache is built or overwritten every time a page is loaded through the proxy, with a few notable exceptions.

The cache is not overwritten if the content served matches the content received (i.e. no processing was done on it), nor are entities larger than the hard-coded maximum entity size (960kb) saved. Furthermore, if a site changes its contents too fast (there are too many cache misses, i.e. the cached content differs from the actual), the Proxy will stop caching the given entity to prevent overusing the database. Should this happen, caches must be cleared manually to restore normal operation and reset the cache miss limit.

What is it for?

The purpose of the Target Cache is to reduce the number of requests to the **source site** and decrease response times on the proxy by skipping the content replacement process (which is basically the entire document pipeline) if the content on the **source site** or - if enabled - in the Source Cache remains *unchanged*.

Enabling Target Cache

Go into the **Page Cache** menu in the Dashboard and enable Target Cache in the dialog that opens. You can also add a maximum of **5** custom-named Target Caches.

Building the Target Cache

After you enable it, the Target Cache is filled exclusively by Page visits on the pretty (published) domain. Previews/visits to the temporary domain will have absolutely no bearing on the contents of the Target Cache.

While separate from the Source Cache, the contents of the Target Cache will nevertheless be determined by your Source Cache settings. If you have no Source Cache, then the Target Cache will be filled by whatever is returned from the source site for a request (a page visit). If you have a Source Cache, then the Target Cache will contain whatever the currently enabled Source Cache is making available to the visitor on the proxy.

The 'Keep Cache Strategy'

The Keep cache is actually a serving mode of the Target cache, used mainly to prevent the source language bleed-through effect, or Bleeding Effect in short. It operates by checking the cached content's translation ratio against the remote server's response. If the cached content is found to have a more complete translation (i.e. its translation ratio is higher), the remote response is discarded and the cached content is served instead. This does not mean an increase in page load speed, but by preventing yet-untranslated elements from appearing in the served page, the Bleeding Effect is eliminated entirely. As new translations are entered into the database, either manually or via XLIFF imports, the difference between the cached response and the actual response decreases, and the newly-translated elements are displayed automatically. Additionally, this check is run every time the source content is found.

Case Studies

Case Study 1: Freeze Site Before Translation begins

In this example, the client has declined to use a staging server, as well as declining to halt content updates for the duration of the initial translation. They do, however, insist that bleedthrough must not occur at any point in the translation process. This results in a continuously changing source that makes it nearly impossible to achieve 100% translation.

In this case, the solution is to enable the Source cache, and populate it with a Scan before the first round of translation commences. This creates what is effectively a static snapshot of the site, which remains the same regardless of the updates the client makes in the meantime, providing a stable environment for the translation and review processes.

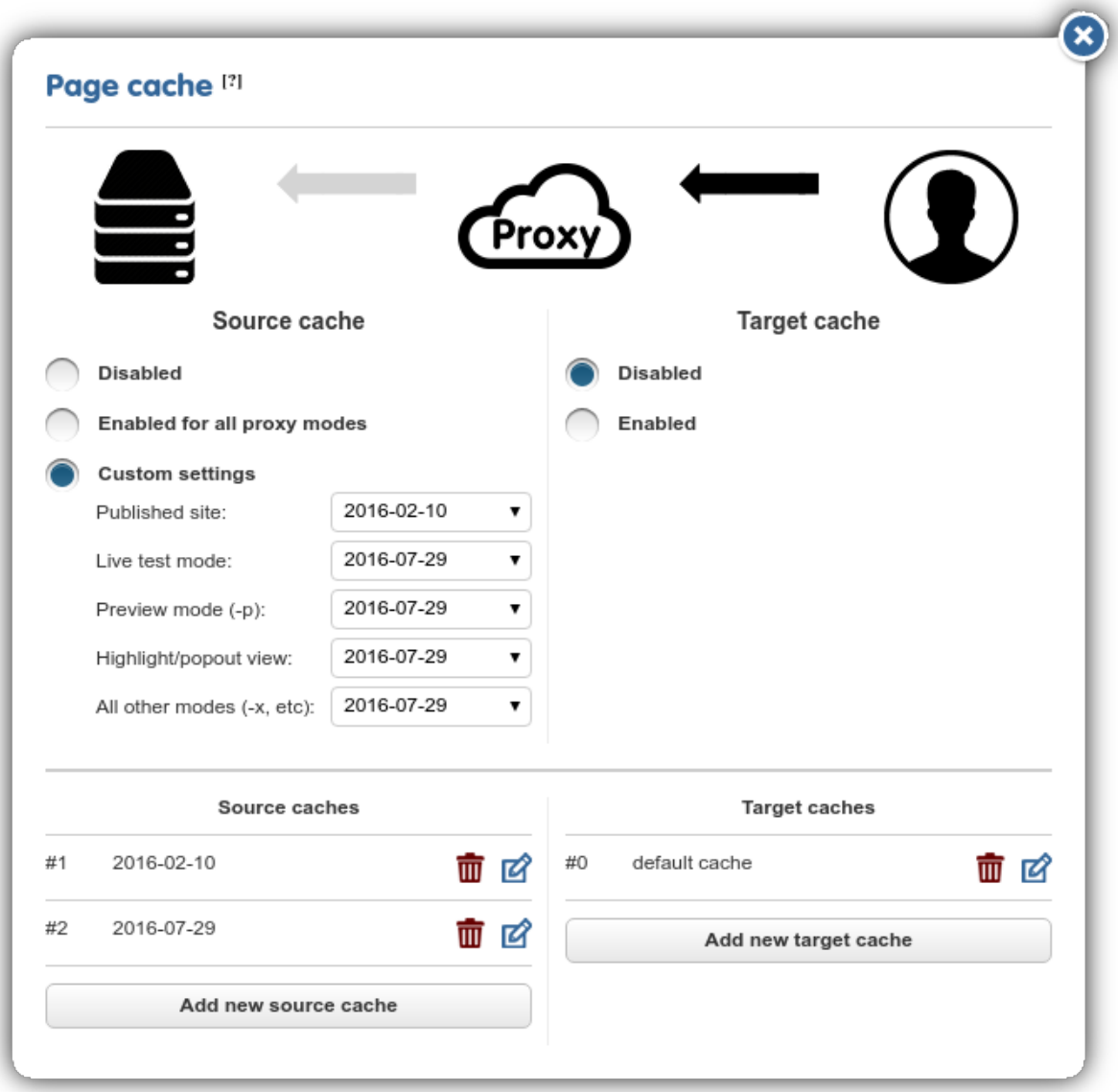
By using a Source cache to serve the translated sites, they are decoupled from the original, and content updates there will not be reflected in the translations.

However, source content will accumulate in the meantime, and once the cache is purged, bleedthrough will occur! To counter this, the second scenario can be enacted.

Case Study 2: Decoupling Content Update from Ingestion and Publishing

For this example, the client has declined to use a staging server to allow you to ingest the content ahead of publishing for translation. They did, however, agree to notify you once new content is published, and have acknowledged that this will cause the translated sites to lag behind the original until the translations (and possibly reviews) are completed. Alternatively, coupling into the previous scenario, the initial translations are in place, but the source site has moved ahead in the meantime, with the translated site being served by the initial Source cache.

Once the initial translations are completed, the caches are set up according to the image below:



By driving the published site from a separate cache entity, you gain the ability to decouple the content ingestion cycle from the translation and update cycles.

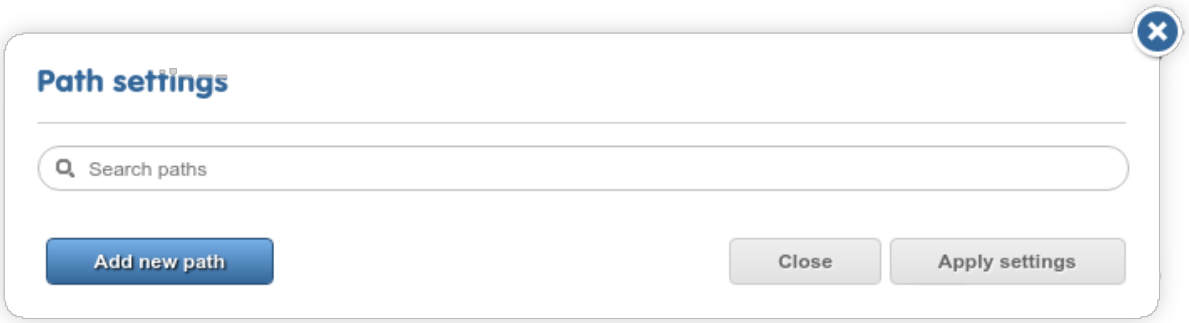
From this point on, the published site will not reflect any updates until the assigned entity is refreshed. By specifying the update cache in the Scan Limit dialog, you write the server responses into the newly created cache, which will drive the Preview and Highlight modes, allowing you to conduct the translations and ICR using the new content. Once the client signs off on the translations, you simply switch the Source cache entities being used (and purge the Target cache, if in use).

At this point, the previously-live cache entity is freed up for purging and re-building in the next update cycle. Thus, at the next crawl, the default would be selected to receive the updates, leaving the other entity untouched.

By managing which entity is being used to serve the published site and which one is being written to, it becomes possible to replicate the behavior of the staging server, albeit at the cost of increased attention to detail.

Path settings

The proxy supports the declaration and use of rules called path overrides (alternatively called subtree overrides). They can be used to apply a variety of changes to an exact (e.g. `/about-us.html`) or prefix paths (e.g. all pages under `/products`).



A project has no overrides by default. Besides the search field at the top, the "Add new path" button is available. Click on it to type or copy & paste your path/prefix. Click on the "Add path" button to the right of the row to add it to the list. The option to choose the type of the path (exact URL or prefix) is available as a dropdown.

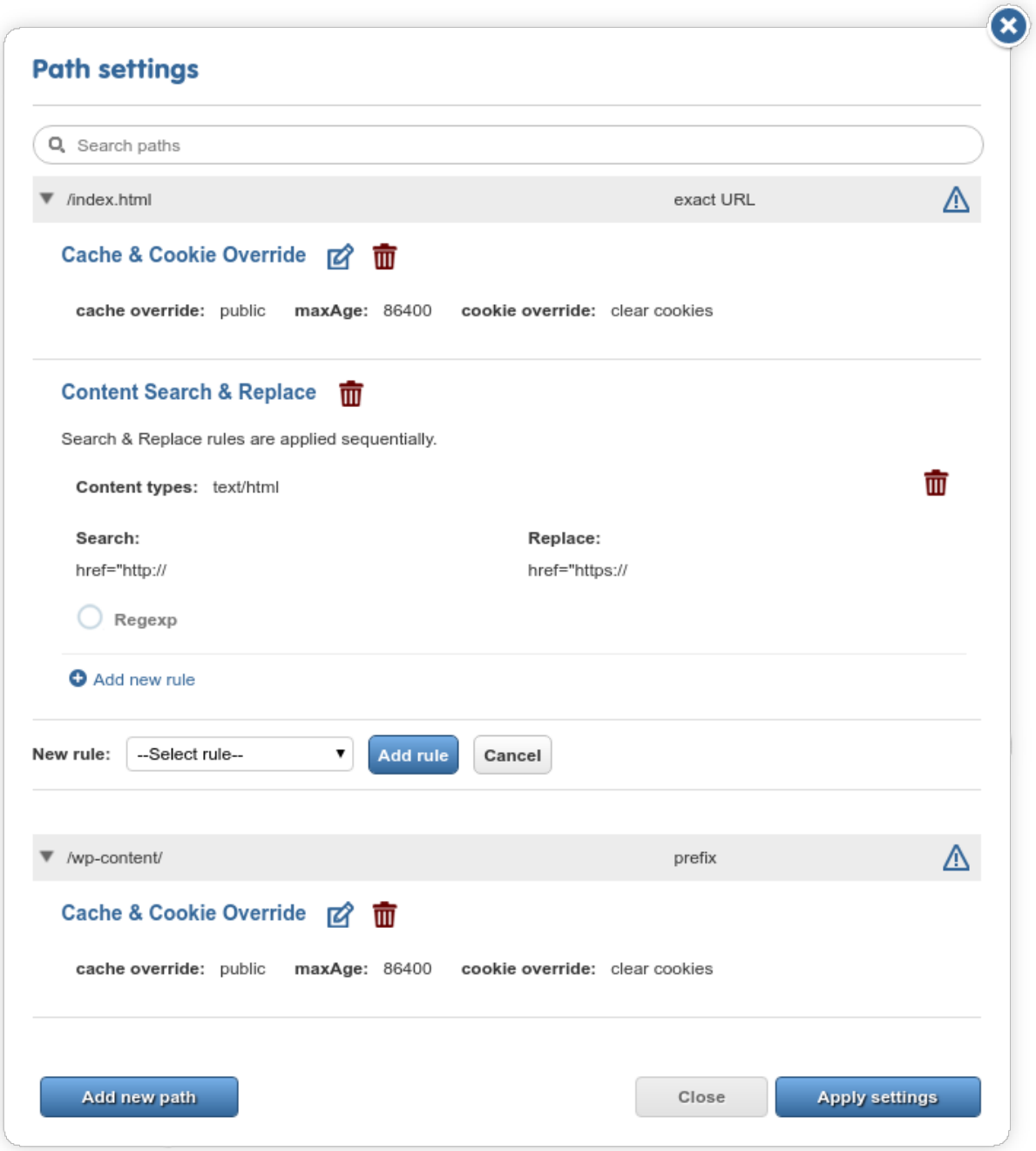
You'll notice that adding new paths and setting up corresponding overrides are done at two separate stages. Any path can have more than one type of override associated with it (but overrides differ in how many of each can be present on a given path).

There is a row of icons to the right of each path that you can use to edit the overrides. Click on the blue plus icon to add a new override for a path. Each type will also have another set of icons for editing and deletion.

Keep in mind:

- overrides are applied only on **intra-domain** paths (elsewhere called the project URL). They cannot be applied on any path referenced on an external domain.
- overrides are applied early on in the pipeline, before any translated text is inserted. They are also **not** target-language specific.
- query parameters (`?q=value` etc.) are **not** supported. Do not add them to path overrides.
- When declaring a content type, mak sure you use an *exact* description: `text/html` does not equal `text/html; charset=utf-8` or vice-versa. Be specific.
- Some of the overrides (particularly search & replace overrides) become non-editable after they are saved. You'll need to delete and re-add these rules to change them.

Important! *Add/edit multiple paths and overrides at once, but remember that all changes are **UNSAVED** until you click on "Apply settings"!* Keep an eye out for the icons on the right, as shown in the screenshot below.



Cache & Cookie Overrides

This override allows you to declare the `Cache-Control` and `max-age` headers for a prefix or URL and optionally clear the cookies.



The path seen in the screenshot above is a typical use case: it ensures that resources on the `/wp-content` prefix, associated with WordPress sites, can be cached for 24 hours.

Setting `cache-control: public,max-age=86400` on a URL/prefix in this way broadcasts to the network that the resource(s) there can be publicly cached. Depending on the location of the caching node and the pathway of the request, the content will be served from caches instead of going through the proxy pipeline.

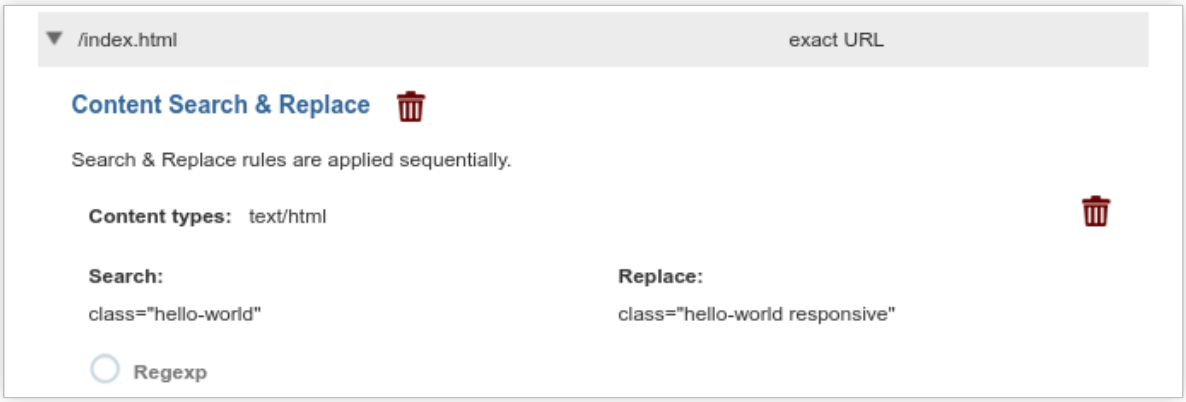
This is beneficial for both speed and cost reasons. What is otherwise tolerable server load on the original site might be unnecessary page view cost overhead over the proxy (with speed overhead not being much of a concern). We provide this capability as a useful cost optimization strategy.

Important! Do NOT add overly general paths or too large `max-age` values without considering the effects! Please read through our description of the issue before using the feature.

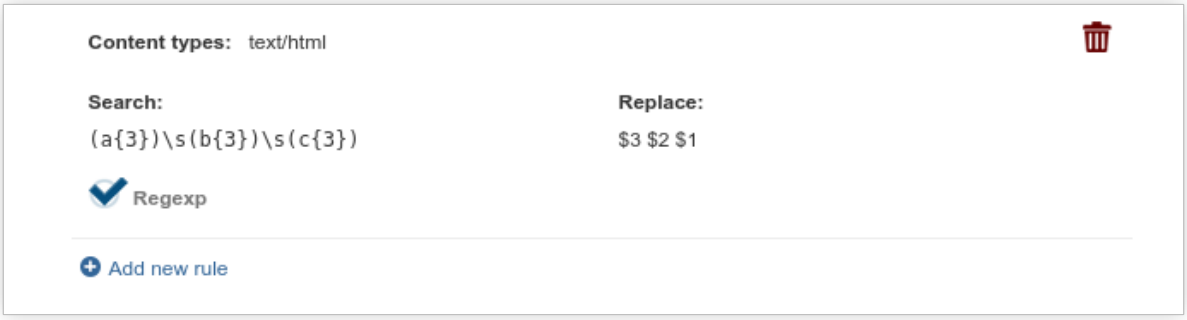
Only one cache & cookie override may be present on each path or prefix.

Search & Replace Override

Run a search & replace on the page or pages on the prefix. This lets you apply simple changes to the page source. You can choose between string or regexp replacement.



In the example above, search & replace is used to extend the list of classes for any element that possesses only the *hello-world* class.



The replacement field supports regexp backreferences via the `$n` format, as in the screenshot above. The example looks for "**aaa bbb ccc**" in the page source, stores backreferences to each letter group and reverses their order, resulting in the output "**ccc bbb aaa**".

Refer to the `java.util.regex.Pattern` documentation for the details of the supported regular expression format.

Note that search & replace is a "naive" operation both for strings and regexps: HTML is not parsed, nor JavaScript evaluated at this stage (roughly, S&R runs as if it were working on a plain text source file). It cannot be used to solve recursive tasks.

Multiple S&R overrides can be added on the same prefix, and they will be applied *sequentially*. No two replacement strings/regexps may match, the dialog will display an error if you attempt to enter the same replacement rule twice.

Default Charset Override

This override is presented as a simple text field, the contents of which will be used for `<meta charset="...">` tags. Useful in cases where the original site is declared to be in an encoding that is incompatible with one of the target languages.

Only one charset override may be present on each path or prefix.

Content-Type Override

Override the Content-Type HTTP header for a given path or prefix. Frequently used with template URLs or JS resources with mischaracterized Content-Types, it is sometimes useful to avoid encoding or character escaping troubles.

Multiple `Content-Type` overrides can be added on each path or prefix, but no two such fields may match.

Advanced settings

The Advanced settings screen is the *nitty-gritty* of the technical side of the proxy, the various features of which make complex project management and involved content extraction/management possible.

Some, such as **Freezes**, are general enough to warrant close attention from all users, others, such as the **Tweaks** checkbox list are required only in specific cases, so you can go through many, many projects without ever having to worry about them. In this section, we take a look at all options in the order that they appear on the Dashboard.

Pattern matching: there is often a great deal of text on a website that is not targeted for translation, because it is highly repetitive or numeric in nature, usernames, timestamps and prices fall into this category of content, to mention a few of many possible cases where Pattern Matching might come handy.

You can construct regular expressions and add them here, one regular expression per line.

If that regex has a capturing group, anything that is **matched by that capturing group** will be treated as **translation-invariable**.

It is important to note that if you add regexes here later into the project, your existing segments will remain in the Workbench. Pattern matching does **NOT** delete or exclude any segments.

Example: filtering usernames and dates

```
Posted by youTuTroll on Apr. 4, 13:22, 2010
Posted by joey0405 on Apr. 6, 03:29, 2010
Posted by cunning_linguist on Apr. 6, 03:45, 2010
Posted by bornToBe27 on Apr. 6, 10:01, 2010
Posted by OMEGA on Apr. 7, 11:59, 2010
```

Repetitive and inconsequential from a translation/word count viewpoint, the usernames and dates should be made translation-invariant in the content above. You can add a regex for it (such as the one below) in the textbox and the translation proxy will exclude whatever is matched within a capturing group.

```
Posted by ([\w]+) on ([\w]{3}\.\s[0-9]{1,2},\s[0-9]{2}:[0-9]{2},\s2[0-9]{3})
```

We recommend that you visit http://www.regex101.com and test your regular expressions with example snippets before setting them on a live project.

Freeze

The following are important options for when translation has already begun in earnest on a batch of content. At this point, it is a good idea to **freeze** the page list and the segments - that is, prevent *inadvertent* addition of new text or pages to the list you are currently working on.

WARNING: These settings can be overriden via explicit user action: adding Pages via the blue icon in the page list, or running Scans with the appropriate settings in place.

As you based quote on a specific Discovery/Scan (in other words, a specific state of the site at a given time), it wouldn't do to return a few days after winning the project only to find that the news section was expanded with two new items and you have 4500 new source words to deal with - indelibly added to the project, but not part of the original deal.

The following features are available:

Freeze pagelist: Prevent new pages from being added to the pagelist. If you turn this option on, you can use the various preview modes withouht having to worry about new content.

Handle unknown pages as externalized: Pages that are not already in the pagelist will be *externalized*, that is, not translated *at all*, the same as if the page was manually excluded. Be aware that on a live site, this may result in an SEO penalty (due to duplicate content being detected by the crawler)!

Freeze translation memory: No new translatable segments will be added to a project as long as this Freeze is turned on (automatically enables Page Freeze).

Group pages

Specify path rules to group pages into one entry, preventing new pages from being added, but not removing preexisting pages, and making the grouped pages share a single dictionary, necessitating translation of only one. The pages will be grouped according to the path rules specified in the textbox, one path per line.

This feature is not meant to and does not decrease the amount of pages a Discovery has to crawl, but it makes project maintenance easier, as it prevents your page lists from being overcrowded with repetitive URLs.

JavaScript translation

This field contains the capture group definitions used to extract attribute-value pairs from JavaScript files selected for translation/localization. After entering the capture parameters and re-crawling the site, the proxy will display the selected JavaScript files as translatable pages in the pagelist, from where they can be selected for translation in the List View like regular pages, and any values for the selected attributes will be made available as translatable entries, which are treated identical to regular entries.

Entering "`html`" (note that the switch is separated by a space!) after the path specification will result in the proxy applying its HTML parser to the match instead of a plaintext parser, stripping out HTML markup and only offering the actual content for translation (otherwise, should the match contain markup, the translator must take care not to alter it, or risk breaking the translated site).

If a field of the JSON being parsed contains further JSON data in a stringified form (`"key": "{\\"key\\":{\\"key\\":\\"value value value\\"}}"`), the path can be passed to a recursive JSON translator by appending "`json`" to the path, then extending the path on the next line by adding "`.json.`".

XPath Translation

The proxy can translate XML (eXtensible Markup Language) files sent by the remote server, according to the XPath standard of specifying elements of the XML structure. Similar to JavaScript translation, entering the "`html`" switch will result in the HTML parser being applied, while no switch will parse the match as plaintext.

Mark multiple Resources as Translatable

Using URL prefixes (N.B. fully qualified URL prefixes, including protocol, host, and possibly path structures!), the Proxy can enforce dictionaries over multiple resources in a single rule. This is especially useful if the site under translation contains an API (especially CREST APIs) whose responses also require translation, and each endpoint is served on a different path; in this case, entering the root of the API here will automatically capture all responses from that path without having to individually mark them as translatable from the Resources menu.

Additional Remote Request Headers

If the remote server requires certain headers to be present to serve legal responses, it will not be crawlable by default, as the crawler will not supply these. Entering the required headers here will result in them being appended to every request sent by the proxy, including the crawler requests, letting you crawl the site as required.

Ignore Classes

If a certain class of elements are not ought to be translated, they can be entered here. Elements with these classes will be treated as if they had the `<translate=no>` attribute, and will be treated as translation-invariant.

Ignore IDs

Similar to the "Ignored classes" option, this allows the treating of specific elements as translation-invariant, this time through HTML IDs.

Process Custom HTML Attributes

Some Content Management Systems may employ non-standard HTML attributes on various elements to style the page or otherwise affect aspects of their operation. If some of these attributes contain translatable text, you can enter them into the "As text" field to instruct the proxy to extract them. If they contain URLs that need to be mapped to the translated domain, you can use the "As link" field to instruct the proxy to map those non-standard link elements as well.

On request, it is also possible to activate an HTML parser for certain attributes, should they contain HTML formatting in their values.

Tweaks

In this section, you'll find checkboxes for settings that apply to very specific circumstances. For those special snowflakes and occassions. When in doubt, contact us!

- **Retaining original DOCTYPEs**: By default, the proxy generates an HTML5 standards-compliant file to send to the client. If, for some reason, this causes problems due to the site relying on HTML4 standards for operation, some of which may be deprecated by HTML5, enabling this option will cause the Proxy Application to retain the original DOCTYPE declaration of the source page.
- **Determine document type by `GET` instead of HEAD**: some servers may return different responses to the HEAD request we use to determine document type than the GET request used to download content. Enabling this option forces the proxy to use GET requests for all operations, getting the correct content type from the server (as far as server-side configurations will allow).
- **Detect JavaScript content type**: JavaScript may not be explicitly declared as such on the server, being sent to the client with misleading content types. This causes the proxy to mis-identify such streams and not offer operations reserved for JS files. Enabling this option will force a deep check on each file sent to the client, to determine if they are indeed JavaScript code, regardless of their declared content type.
- **Download images through the proxy**: this will instruct the Proxy Application to attempt to map all `<img src>` attributes to the proxied domain. This is especially useful if the images are served only after authentication.
- **Attempt to place tags according to punctuation when using TM-based pre-translation**: if using a TM-based pre-translation, the Proxy Application may encounter segments where it cannot replace the XLIFF tags automatically, due to overly large differences between the contexts (possibly because of a changed site). If this option is enabled, the translator will try to replace the XLIFF tags according to their positions relative to punctuation marks in the segment. If successful, the TM entry's confidence score will be increased by 0.1.
- **Translate excluded pages when viewing them in Preview mode (but still not in live serving mode)**: It may be necessary sometimes to view excluded pages as if they were translated, in order to assess their layout, without actually making them available on the live site. Enabling this option allows just that, by propagating translations to the excluded pages if viewed in Preview mode, but keeping them untranslated in Live serving mode.
- **Translate `javascript:` attribute**: The Proxy is capable of extracting and translating code from the onclick attribute of elements. This feature may be used when a site uses this attribute to store translatable content inlined into the attribute and requires this content to be translated. Currently we only process the onclick event's contents, all other events in the javascript attribute will be ignored.
- **Detect and handle JSON-in-string responses, like `"{\\"ResponseCode\\":\\"BadRequest\\"}`**: string-escaped JSON-format responses can be handled by activating this tweak. If active, the proxy will first attempt to string-unescape the response before passing it to the JSON parser, in order to recreate its base form.
- **Make content in `<script type="text/html"></script>` translatable as a whole (don't try to parse it as HTML)**: Script blocks may contain template data requiring translation, which is often signified by

the text/html content type (instead of the more usual application major type). In such cases, HTML parsing can be undesirable, and can be bypassed by activating this option.

- **Send a canonical link http header pointing to the original page on externalized pages**: the proxy can insert a Link header into externalized pages, in order to avoid the SEO penalty associated with duplicate content. This header will point to the original address, and have `rel=Canonical` added, to designate the relationship.
- **Perform string concatenation in JavaScript before parsing.**: the proxy will perform string concatenation and treat the result as a whole when translating/extracting it. This tweak is only useful if *no computable expression* is featured in a given concatenation.

Manual Publishing

Manual Publishing is advanced project control feature that gives project owners the ability to withhold the translations from being published on the live page (but not the preview, as it will always display the latest translation available!) until further notice.

The feature can be activated from the Advanced Settings page. Once activated, it will affect all translations going forward, but already-existing ones will not be "un-published".

Once active, a new item will appear in the Bulk Actions menu of the Workbench: "Publish". Running this action will cause all selected segments to be synchronized with their displayed translations, and once the action finishes, the markers in the status bar on the right of the entries will change to reflect this. If the action encounters an error, the server will attempt to rectify this by publishing the entire entry, after confirmation from the user. If the error is not recoverable, it will list the segments in error.

Default segment state

By default, the proxy will add new segments during a crawl as "Approved", making them available for translation immediately. However, if the user/client so desires, this behavior can be changed to adding new segments in one of the two other states, "Pending" or "Excluded". If the default setting is changed, the project owner, backup owners, or users with the Customer Role can alter segment states.

"**Pending**" segments are those that are awaiting a decision on translation. They will not be included in exports unless the relevant option is selected at export-time, and they will not appear in the Workbench for translation unless filtered for specifically. Those users able to alter segment states may change the state of these segments either by approving them for translation or excluding them altogether.

"**Excluded**" segments are those that have been deemed as not requiring translation at all. Unless the relevant option is selected, they are not included in exports and will not appear for translation unless filtered for specifically. Excluding a segment is not final, however: those users able to alter segment states may approve them for translation, making them available again.

Translation memory

Easyling's translation memories (TM) are used to store existing translations for segments. You can import translations from *.tmx files or populate the translation memory with translations you have added in the Workbench.

Type in the name of your Translation memory and click on Create. Your TM will appear right above the input field, along with the "Add target language" option. You can also use the dropdown menu to assign TMs you've created previously.

Hovering over a translation memory reveals a menu, which allows you to import/export *.tmx files or populate the TM with existing translations. Use the "Remove" button to delete the TM.

The screen is furnished with a search field you can use to look up segments in a TM.

NOTE!

The TMs you manage on this screen and the Workbench are different from the Translation Memory that is billed: they are used for concordance lookups and automatic translation, and they are completely free to create and use.

On the other hand, the proxy TM associated with a project is a database in the cloud that holds translations of segments. When you reimport your XLIFF files or run auto-pretranslate, you are writing to this project TM, which is billed the first time around.

Pre-translate

By using the Pre-translate menu, you can initiate a process which supplies untranslated segments with a preliminary translation in order to mitigate bleedthrough on the proxied pages. You can use pre-translate with a Translation Memory or a Machine Translation API of your choice. the translation proxy supports Google Translate, iTranslate4eu, Bing Translate and GeoFluent.

In order to use Machine Translation, you need to go into your Account and set up your MT account Ids/authorization keys in the "Machine Translation" menu.

Auto pre-translate

Automatize pre-translation: you can set up the project to automatically execute a pre-translate process for any new content that is found - you can hook it up with your TMs, or any Machine Translation services you subscribed to.

Pre-translation of incoming content can be done without user intervention or oversight. If new content is encountered, and at least one source is configured, a user-configurable timer starts counting down. Content is collected while this timer is running, and once it reaches zero, translation begins from the configured sources.

Content that cannot be translated (if no matches of acceptable confidence level were found in the TM or if the MT-engine returned no translation) is packaged into a Work package that you can have sent as XLIFF files to external systems, such as XTM, XTRF, or your Dropbox account. There are links at the bottom of the dialog you can use to visit the settings screens where you can set up these accounts.

Remove project



This is always displayed in red. By clicking on it, you remove the project from the datastore: translations will no longer be served and the translated domains will return an error to browsers.

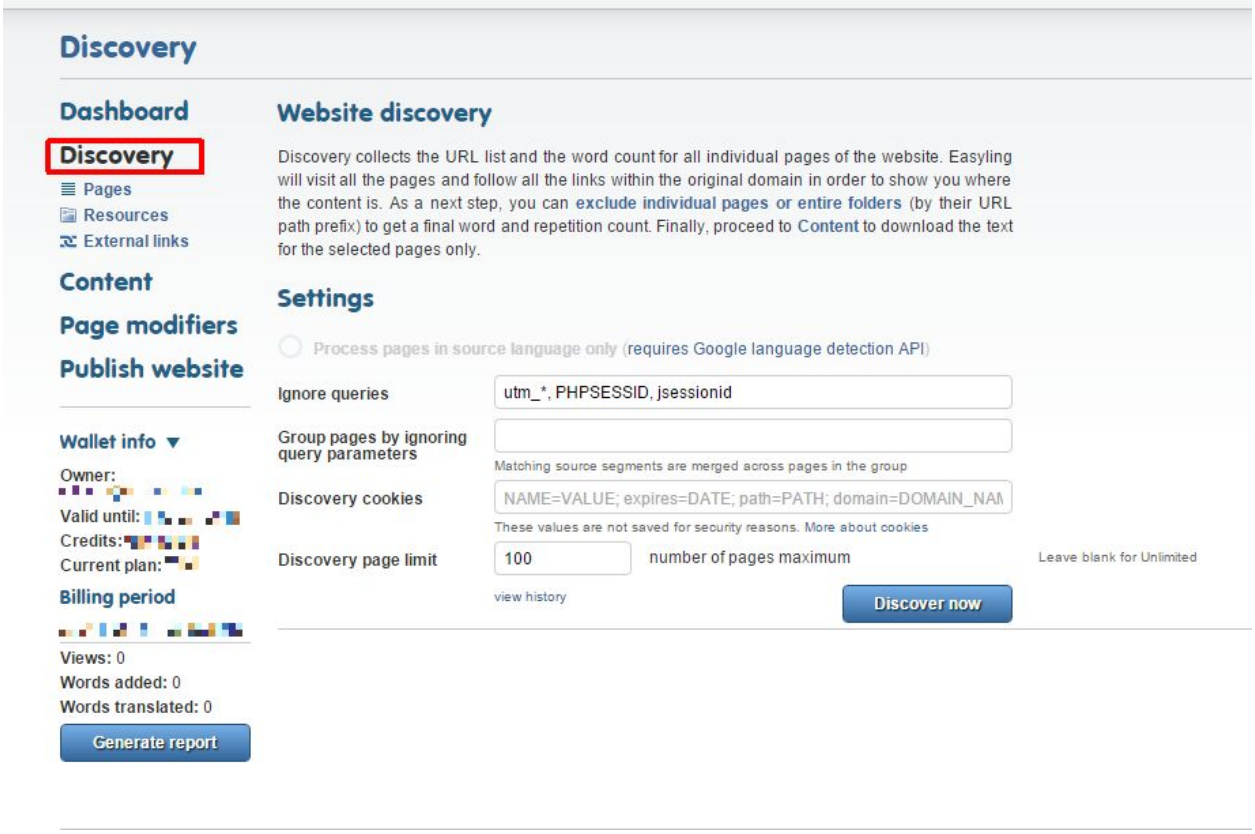
Be careful!

Discovery

General

So, what is a Discovery?

Before you can translate a website, you need to **extract** all content from it using the Scan function. A Scan, however, writes to the database, which, as you know from our pricing model, will cost you money. In that sense, it would be risky to let loose an unlimited Scan on a website you're not familiar with. To forego Content Extraction and limit the crawler's actions to finding out about the structure and word count of a given web site, a more tentative approach is needed: a Discovery.



The process maps the structure of the site by scanning each page for link elements and trying to follow these links. The content of the website is not stored, only the URL address of the visited pages, together with their status info. Any page that is verified to exist is marked as Discovered, and the ones that returned an error message (most commonly redirection (HTTP301-302) and page not found (HTTP404)) are marked Unvisited in the page list. For more information on HTTP status codes, see here. Although this process doesn't extract content, it provides a preliminary wordcount and a repetition rate as well. It has a cost of 1EUR per thousand pages.

IMPORTANT!

If you exclude pages during discovery, changing the rules only will not include them in the new discovery; you have to add them manually through the "Add pages" dialog.

Your primary choice is between limited and unlimited Discoveries.

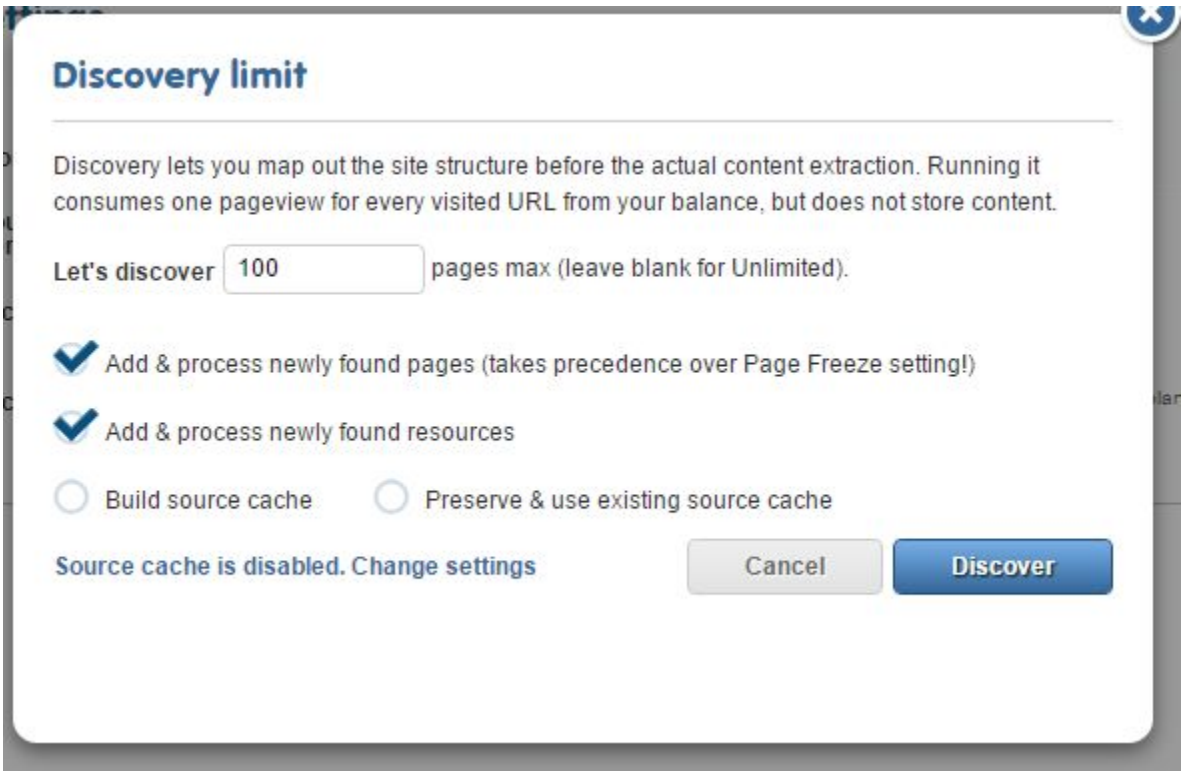
A limited Discovery is the best way to "get to know" a website - if you set a reasonable safety number, such as the default 100, you can be sure the process will go overboard when it finds an enormous forum or a structurally complicated web-shop.

Based on what a limited Discovery tells you about a site, you can set **prefix exclusion rules** in the page list or **cherry-pick** unnecessary pages for exclusion.

Wordpress sites, for example, tend to create links such as "/?page=2332" for every page, and it is obvious from the get-go that there is no real content behind these links. A limited Discovery will inform you of their presence, so that you can exclude them at.

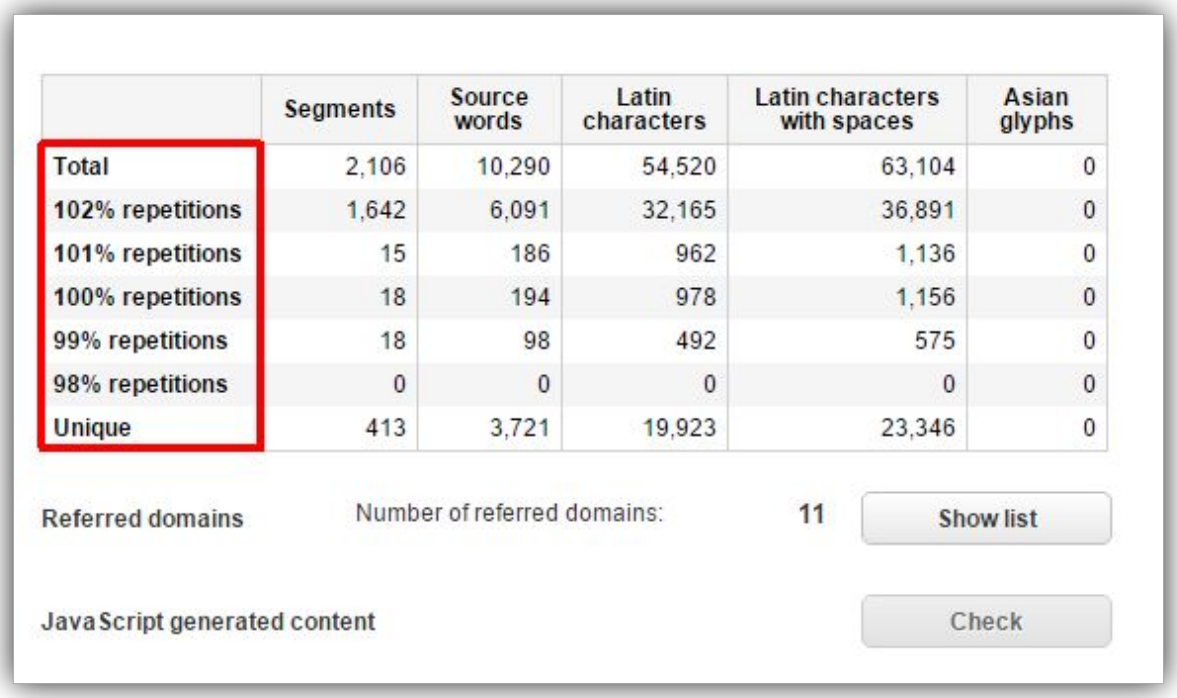
By increasing the page limit and running Discoveries in succession while adjusting your inclusion/exclusion rules, you can work your way through a site structure incrementally.

Of course, if you have a clear idea of the size of the site, you can always set up an unlimited Discovery, wait for it to finish and set up your exclusion rules afterwards.



Once the discovery is ready, you will receive an e-mail notification and the statistics will show up on the Discovery page. Based on this you can give a rough estimation for the website translation cost - both in time and money.

Statistics



	Segments	Source words	Latin characters	Latin characters with spaces	Asian glyphs
Total	2,106	10,290	54,520	63,104	0
102% repetitions	1,642	6,091	32,165	36,891	0
101% repetitions	15	186	962	1,136	0
100% repetitions	18	194	978	1,156	0
99% repetitions	18	98	492	575	0
98% repetitions	0	0	0	0	0
Unique	413	3,721	19,923	23,346	0

To understand the way statistics work in Easyling, we have to start with the most important fact: in our application, the largest unit of measurement is a block, which is usually represented by a `<p>` or a `<div>`. Blocks break down to segments, segments to words, words to letters. Since the translation proxy deals exclusively with content in webpages, HTML tags also play an important part in weighing the repetitions.

It is also important to note that the statistics in the translation proxy are different degrees of repetitions. During a crawl, the website's content is "repetitioned" against itself, simulating a translation process not unlike the Homogeneity feature in MemoQ.

Repetitions

Below is a breakdown of the various repetition percentages in the Statistics.

102% - Strong contextual repetitions: These are block repetitions. Every segment in the block is a 101% repetition, and all the tags are identical. We do not charge for these repetitions and they are propagated automatically within the project.

101% - Contextual repetitions: These repetitions are comparable to the 101% repetitions in MemoQ, or Context Matches in SDL Trados Studio. Both tags in the segment, and contexts (segments immediately before and after) repeat.

100% - Regular repetitions: This one is straightforward, and comparable to the Repetitions count in MemoQ or Trados. The segment is repeated exactly, including all tags.

99% - Strong fuzzy repetitions: In this case, a repetition is found after few transformations on the segment before comparing: tags from the ends are stripped out, words lowercased, numbers ignored.

98% - Weak fuzzy repetitions:Here, all tags are stripped out, not just the ones in the end; words lowercased, numbers ignored.

102% repetitions

102% repetitions warrant special mention. The proxy deals with *HTML block entries*, and has a very "overarching" view of them, since it strives to ensure that the same entry is never translated twice, regardless of which page it shows up on.

Consider a **navigation bar** of a website. During a Discovery, the proxy will come across the navbar *for the first time* on the landing page. It will aggregate the word count of the block elements in the navigaion bar as **unique** and moves on to analyze the next page.

Of course, navigation bars are such that they are shown on all pages of a website, so when the proxy sees the same navigation bar for the *second time* somewhere else, it doesn't count it as unique: instead, it adds the associated word count as a **102% repetition**.

So, the navigation bar is liable to be counted as many times as there are pages: so don't be alarmed if you see large numbers in the 102% repetition row – any sort of repeated content is mercillesly added there. Just keep in mind that this is work that the proxy is saving you.

Cost

102% repetitions should not be counted when creating *cost projections* based on a word count result. So, remember the formula: for any given word count result, `total minus 102% repetitions` is the maximum amount that you need to extract or translate.

Examples

You will find an illustration of the various repetitions in the table below. Hover your mouse over any of the matches to highlight the differences.

Discovery settings

Fine-tuning Discoveries and dispatching them on a site is the bread-and-butter of Easyling, and arguably the most important part of the setup-phase of a project. In this section, we take a look at the various settings, convenience features that are at your disposal. Some of the most important overarching settings will be displayed in the main screen of the Discovery screen, others will be revealed by the Discovery dialog, which opens after you click on the blue **Discover now** button. Let's take a look at each of those.

Discovery Screen

Ignore queries: Add any query parameters to this field, separated by commas, to ignore them during Discoveries.

By default, the following URLs will turn up in the Page List as two separate pages:

`http://www.example.com/examplepage.html`

`http://www.example.com/examplepage.html?querystring_1=foo&querystring_2=bar`

Query strings can influence the content that will be served, but it is also common for a page to be completely unchanged regardless of their presence or absence. In such cases, query parameters are somwehat bothersome: they cause the Discovery to revisit the same page multiple times, they overcrowd the page list with unnecessary entries and divert attention from the real page that holds the content to be translated. It is a good idea to trim the list and exclude these superfluous pages using this feature.

Group pages by ignoring query parameters: In the same vein, you can also group pages based on ignored query parameters by adding those query parameters to this field. Query parameters are bona fide part of the URL, and therefore may or may not induce changes on a case-by-case basis. For every changed query parameter, we create a new page record. If you are sure about a set of query parameters not inducing any changes in page content, you can use this function to prevent them from getting picked up.

Discovery cookies You may extract the cookies from a user session and pass them to a Discovery. This feature is most useful when you need to deal with login walls and content that is only revealed within the context of a user session.

Note that we cannot guarantee that the cookie extraction method works with all proprietary logins, as there are several solutions which can use additional security measures such as IP checking, or checking the User-Agent header. As all the requests from the proxy and the crawler come from the Google Cloud, it is necessary for the target site to not block them. The success of a Discovery via the use of session cookies is highly dependent on the implementation details of a site.

Discovery page limit: It is usually not a very good idea to unleash unlimited Discoveries right off the bat, not unless you have a 100% understanding of a site's structure. Granted, Discoveries don't really cost much, so you are in no danger of incurring any serious expenses. But a Discovery can take time - and the shorter the time span required to create a quote with high confidence levels, the better.

For this and other reasons, it is a good idea to start with a "feeler" Discovery and take it from there, adjusting query string settings, adding prefix-based exlusion rules for unnecessary subsections, fine-tuning the settings as needed - this way, you can safely shape the Discovery incrementally, until it visits only those pages that you deem necessary.

Discovery history: By running subsequent limited Discoveries using progressively refined settings, you will also get contextual clues about site state by checking the Discovery History. By clicking on the small **view history** link under the timestamp of the last Discovery, you can avail yourself of all the statistical information revealed by your previous Discoveries by using the "Show info" button in the opening dialog. You can also verify the status and end state (termination reason) of your Discovery processes here.

Referred domains Links pointing to a different domain than the project website address will be treated as external referred domains and listed here for your convenience. Note that specific instances of external links (such as links to streamed media) can be localized(replaced) over the proxy. See Content > External links for further details.

Discovery Dialog

Discovery limit Repeated from the main screen, you can limit the Discovery to a set amount of pages before exiting.

Add & process newly found pages any pages the Discovery process finds during a crawl will be added to the page list. Note that this is explicit action by the user, therefore the Page Freeze setting has no effect in this case.

Add & process newly found resources Instruct the Discovery to add newly found Resources, such as *.js files to the project. The proxy makes collecting Resources a trifle, and there is a plethora of in-depth methods to translate content embedded within them. But translation of Resourecs is also known to be one of the trickiest parts of website localization. Beware, and woe betide the unready!

Add & process newly found image resources By enabling this option, image content found on the site is added to the Resources screen, where you can add localized counterparts and instruct the proxy to serve those in place of the original.

There is often a great deal of image content on websites, so indexing all of them can take a very long time to finish; it is a good idea to turn this option off the first time around, and enable it only later, when it is clear that image localization is also going to play a part.

Skip content type check if already determined (to be added)

Limit crawl depth There is more than one way of limiting the scope of a Discovery. In scenarios where you have to Discover richly interlinked pages (a wiki, for example) you can use the crawl depth to limit how deep the Discovery should be allowed to follow links.

Ignore pages matching the following regular expression You can specify any regex to apply to all URLs and ignore the given page if there is a match. This is a user- and crawl-specific setting. As always, we recommend that you test your regular expressions on regex101.com before using them. In this case, you can export your page list from Discovery > Pages and use that list as a test string for your regex.

Limit number of simultaneous requests Control the number of requests a Discovery is allowed to send to the original site simultaneously. Use this function to prevent the Discovery from flooding a smaller server with requests.

Exclusion rules You can set Discovery-specific exclusion rules by selecting *Use temporary exclusion rules* from the menu and adding your prefixes to the field that opens - use the buttons on the right to choose whether the prefix entered is an inclusion or exclusion rule.

By default (*"Use current exclusion rules"*), a Discovery will use the exclusion rules that you've set up on the Pages list screen.

Build Source Cache Source Caches have to be built during Discovery or Content Extraction crawls. This checkbox becomes available after you have enabled Source Caches in Dashboard > Page Cache. A dropdown menu will also appear, where you can select which of the Source Caches you'd like to use for the current Discovery. Please see the Source Caches section of this manual for the details. The option to keep the current Source Cache intact (*"Preserve & use existing source cache"*) is also available - these to options override each other, you should turn the latter off when you're building a Source Cache.

After clicking on the blue "Discover" button, the Discovery starts and you will be returned to the main Discovery screen, where you can follow the word count statistics and check the status of the currently running process.

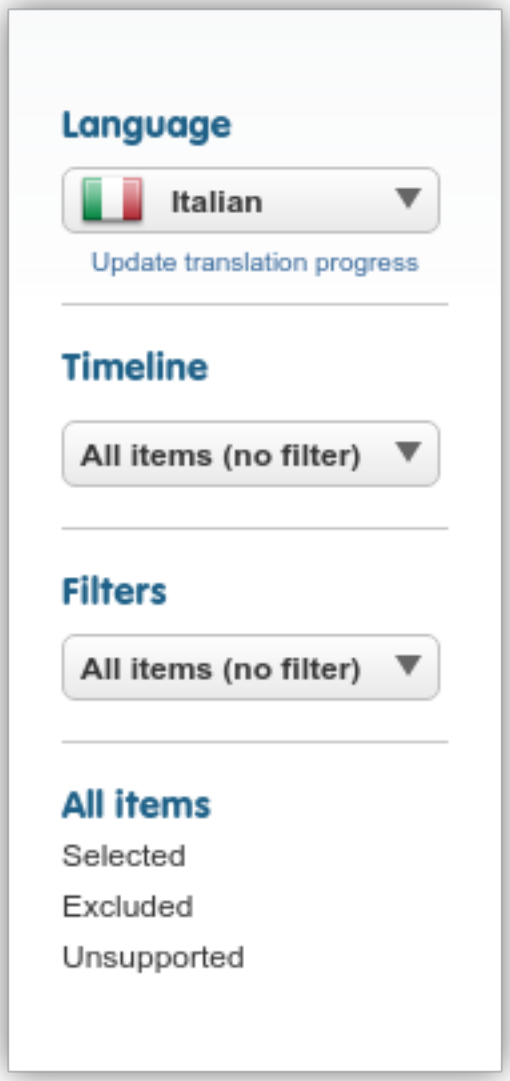
Pages

The page list lets you keep track of the various URLs that are (or were) seen by the proxy. It can be accessed from both the Discovery and Content menus (See the note on the Add Pages dialog elow for one crucial difference).

Aside from the overview of and direct access to content on the individual pages, the page list also allows you to visit the various proxy preview domains, fine-tune inclusion/exclusion settings and check the translation progress of individual pages.

Side menu

The number of pages that are displayed is influenced by the various filters available in the left-side menu:



Language Selector

Though strictly speaking not a filter at all, the language selector influences the page list in important ways. Information in the translation progress bars will refer to the selected language, and both the List and Highlight Views will open for the selected target language. The same is true for the various types of Preview links.

Ensure that you have selected a target language to gain access to many powerful features in the page list. If you do not (or there is no target language on the project at all, in which case, the source language is selected), then neither the Workbench links, nor the Previews will be available, and most of the hover menu options will be greyed out:

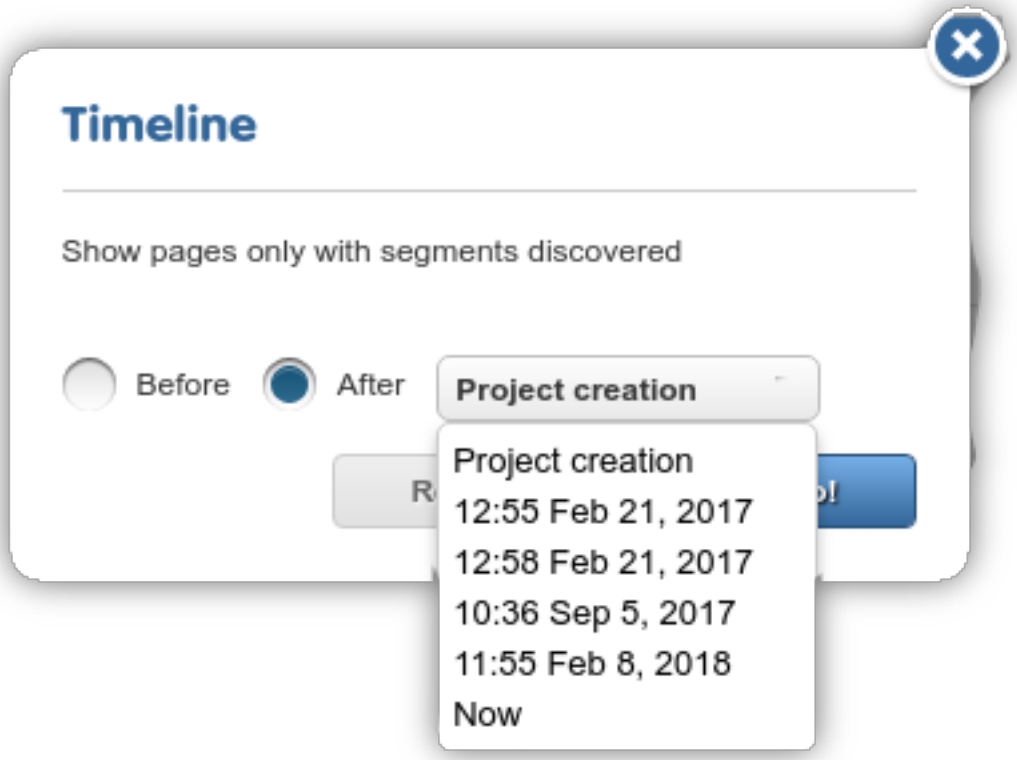


Update Translation Progress link

Click here whenever you want to receive updated information about the current translation ratio in the page progress bars. Information will refer to the selected target language. Updates are automatically run every 24 hours. Note that the process can take from a few seconds to a few minutes of time to finish, depending on the amount of content to be aggregated.

Timeline

Use the dropdown illustrated the screenshot below to filter for pages that registered new content (either via a content extraction crawl or automatic extraction in Preview) before or after a given recorded date. Set the value to "After Project creation" or click on Reset timeline to undo this filter.



Filters

This dropdown, though it may sound general, does a specific thing: lets you filter for pages that are in the selected state. The possible options are

1. **Visited (Scanned) only**: filter for pages that are already Scanned (content extracted). These pages will show a status of "NEW" or a translation progress bar, indicating that they are in-progress.
2. **Unvisited only**: filter for pages that are referenced on some other visited page and their URLs were added to the page list, but neither a word count, nor a content extraction crawl has had a chance to process them yet.

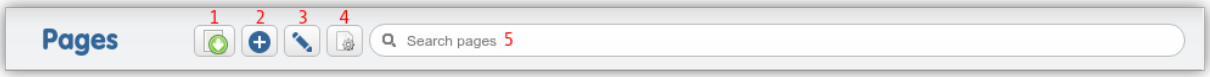
3. **Discovered only**: filter for pages that were word counted before. Note that the "Discovered" note to the right of a page does not tell you the details of *which crawl Discovered this page*, so you can't automatically assume that *all* Discovery statistics contain the word count from it.
4. **All items**: disable this filter.

Exclusion State

A project can keep tabs on many distinct URLs. For example, query parameters, especially in combination tend to increase the number of URLs, potentially crowding the page list and resulting in much unnecessary scrolling. Use the exclusion filters at the bottom of the side menu (and in combination with the search field on top) to get a clear view of your currently selected or excluded set of URLs (see below for the details of exclusion/inclusion rules).

Note that you can use the search field to search for *any part* of a URL, not just the beginning.

Top Menu



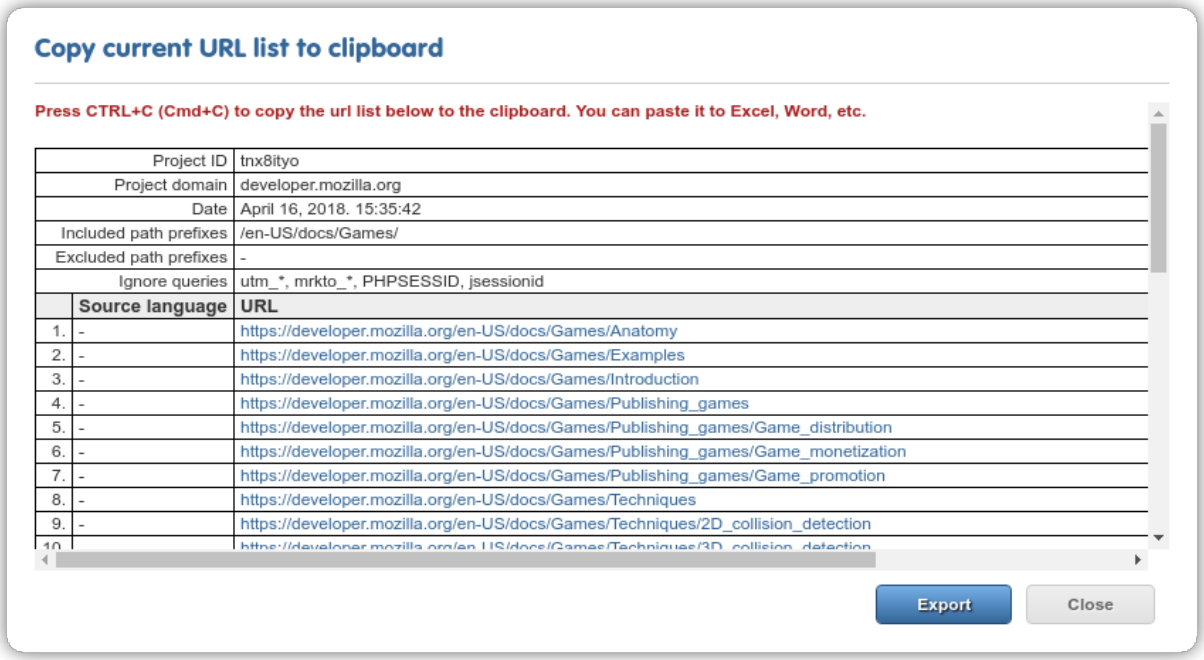
The following features are available in the top menu:

1. Export page list
2. Add New Pages
3. Mass Exclusion of URLs
4. Include/Exclude
5. Search

See below for the details of each feature.

Page list export

Clicking on this icon presents you with your current selection of pages in an easy to copy & paste format (autoselected).



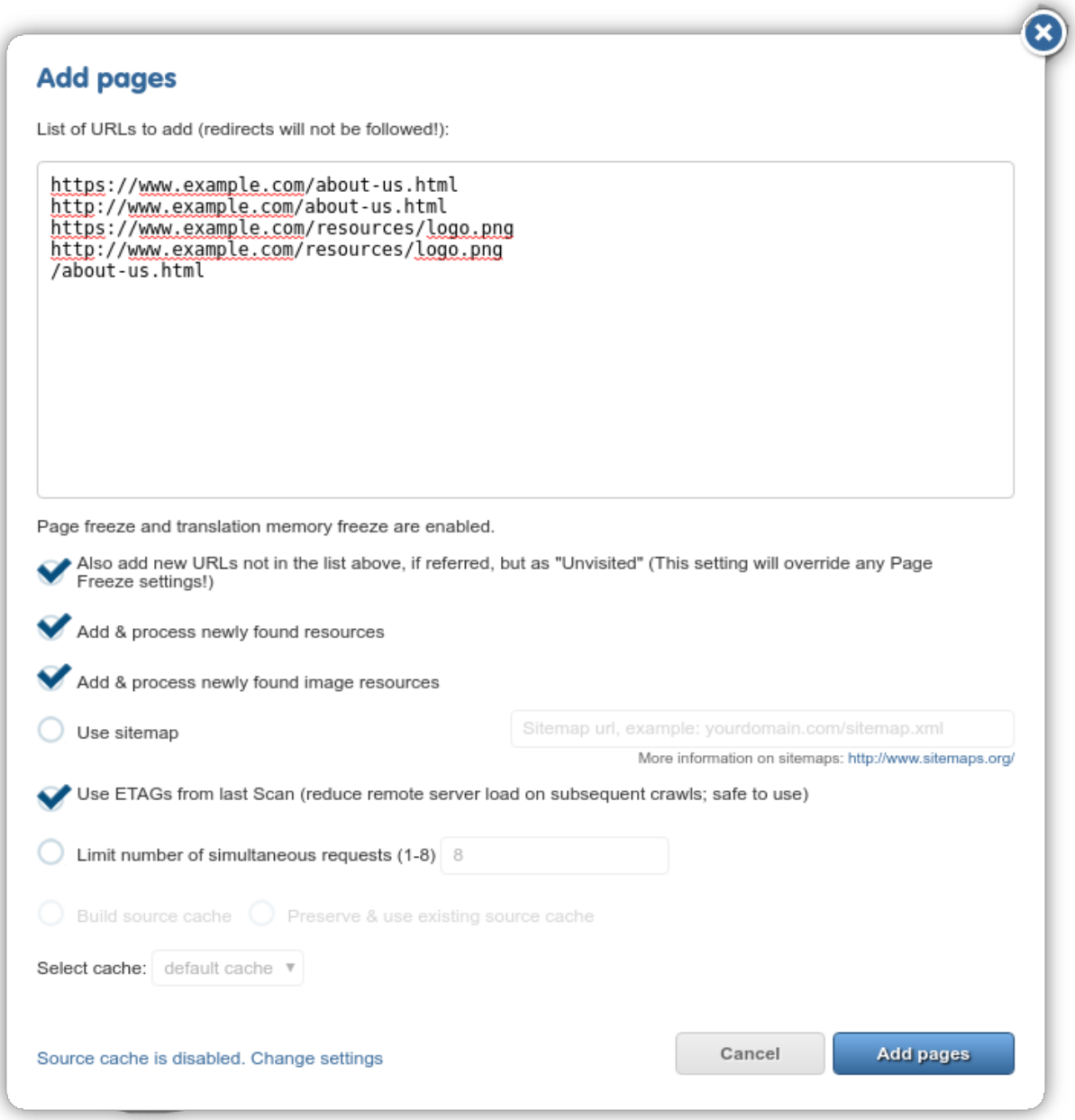
Above a certain reasonable threshold, the Dashboard will not render the whole page list in the dialog, and you will receive a message that the page list is too big. The option to export as XLS remains avaialble in both cases.

Add New Pages

The Add pages dialog is a powerful feature: it lets you start *page-specific crawls*. Use it to add new pages or refresh already extracted pages in a "surgical" fashion.

You might find yourself in a situation where a specific set of pages would benefit from a content extraction, but you don't want to start a full crawl for this purpose alone. Perhaps you'd like to add a few images for localization (it is frequent that only a few of the images on a site require translation, so this use case tends to come up a lot).

The Add Pages lets you do all these things without the overhead of a full-blown crawl.



Note the examples in the screenshot: HTTPS URLs are added using fully qualified URLs. If you specify the pathname only, such as `/about-us.html` in the screenshot above, the proxy will default to requesting the page over HTTP. The same applies to any resource that is available via both HTTP and HTTPS: add both URLs to ensure that you have all possible versions of the resource.

Features that limit the crawl's scope (e.g. page limit) are not present in this dialog as the list of URLs effectively tells the crawler where to go. Source cache-related settings are available, however, you have the option to add a sitemap link for the proxy to crawl.

IMPORTANT! There is an important difference between the Add Pages dialogs of the two page lists: If the Add Pages dialog is opened from Discovery > Pages, it will start a word count Discovery; opening it from the Content > Pages menu will result in a content extraction Scan!

Mass Exclusion

We mentioned previously that the proxy can exclude pages in two ways: by applying a prefix rule or by manual exclusion. But sometimes, a prefix-based approach is "too inclusive". At the same time, exclusion of large numbers of URLs by hand is tedious. Click on this icon and copy & paste URLs in the opening dialog to apply manual exclusion to those URLs *en-masse*.

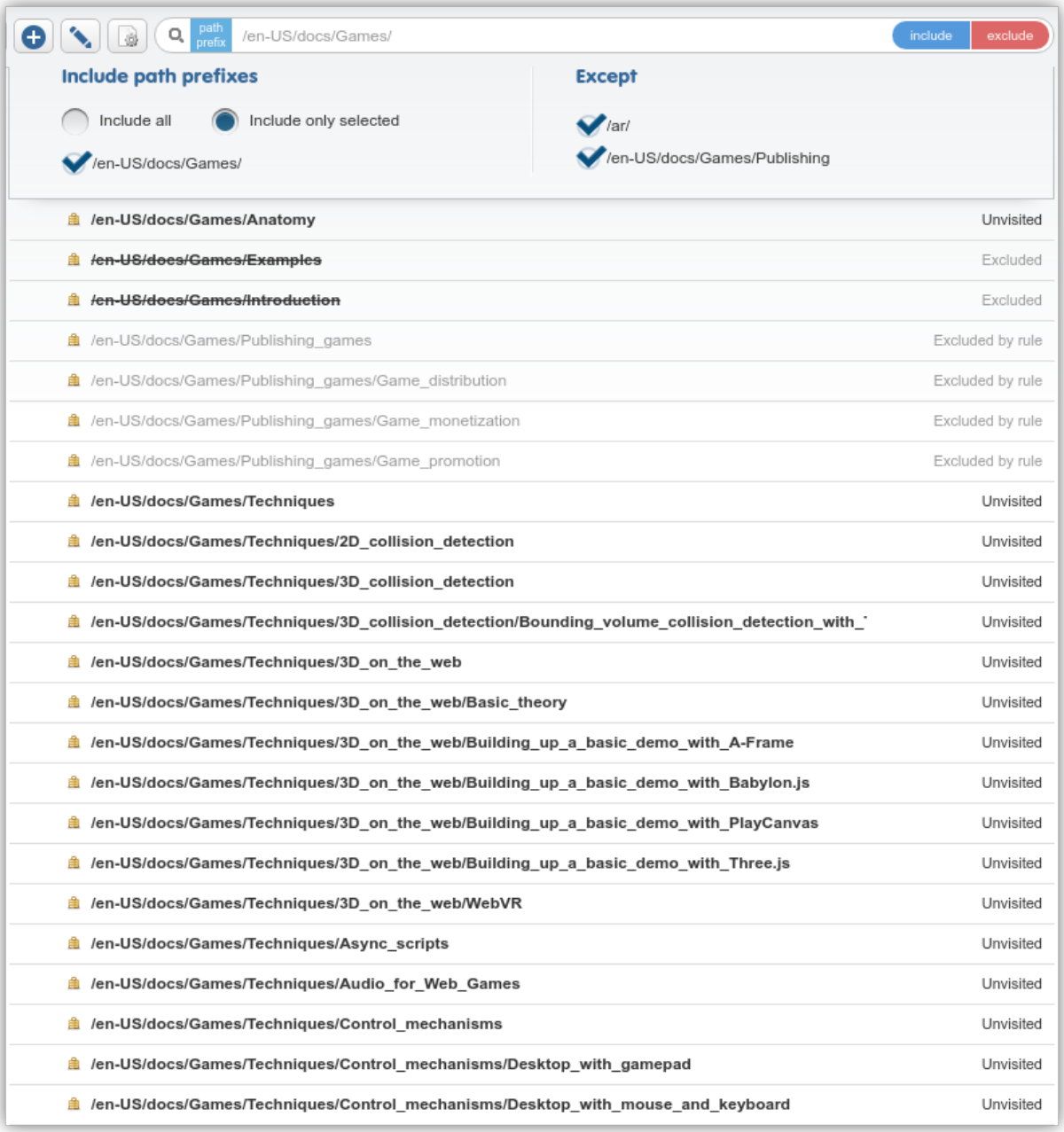
It pays to be careful, however: keep in mind that if you lose the list, your only option will be to re-include the pages is one-by-one manual action (unless, of course, you can re-generate the list of URLs via the page list filters).

In general, inclusion/exclusion *rules* are a better approach if your scope can be specified with them.

Inclusion/exclusion rules

Inclusion rules let you set the **scope of translation**, that is, the list of pages that the crawler is allowed to visit. Inclusions fall into two main categories, prefix and manual. We discuss the prefix-based rules in this subsection.

When you click on the "Sheet & Gear" icon, the top menu and the search bar is transformed into the inclusion / exclusion rule editor as shown in the screenshot below:



The Dashboard has a number of features that support inclusion/exclusion prefixes, such as auto-pretranslation or work packages. But the rules you specify here are special and powerful: they have influence over the entire project.

An excluded page will stay *untranslated* over any proxy domains (both Preview and Live), and excluded pages are ignored by the crawler.

Type or copy & paste an URL into the search field when the rule editor is active, and click on either “Include” or “Exclude” to add it to the appropriate column. If you add a fully-qualified URL via copy & paste, the scheme/domain part is automatically stripped.

Include path prefixes

A set of radio buttons in the inclusion column lets you toggle inclusion rules on your project.

Include All disables any inclusion limits on the project. No rule in the *include* column will be applied. but exceptions listed in the "Except" column on the right will be adhered to.

Include only selected enables inclusion-based limitations. E.g., if you have a single inclusion prefix `/en/` in the left side list and enable "Include only selected", the crawler will only process pages that can be found in the `/en/` directory on the server (to put it more accurately, it will process only those URLS in which the pathname part matches the string `/en/` from the beginning).

If you have multiple inclusion rules listed, you can selectively enable/disable any one of them using the checkboxes to the left of each rule. If you hover over a rule, a small delete icon will appear to the right, use this to remove a rule.

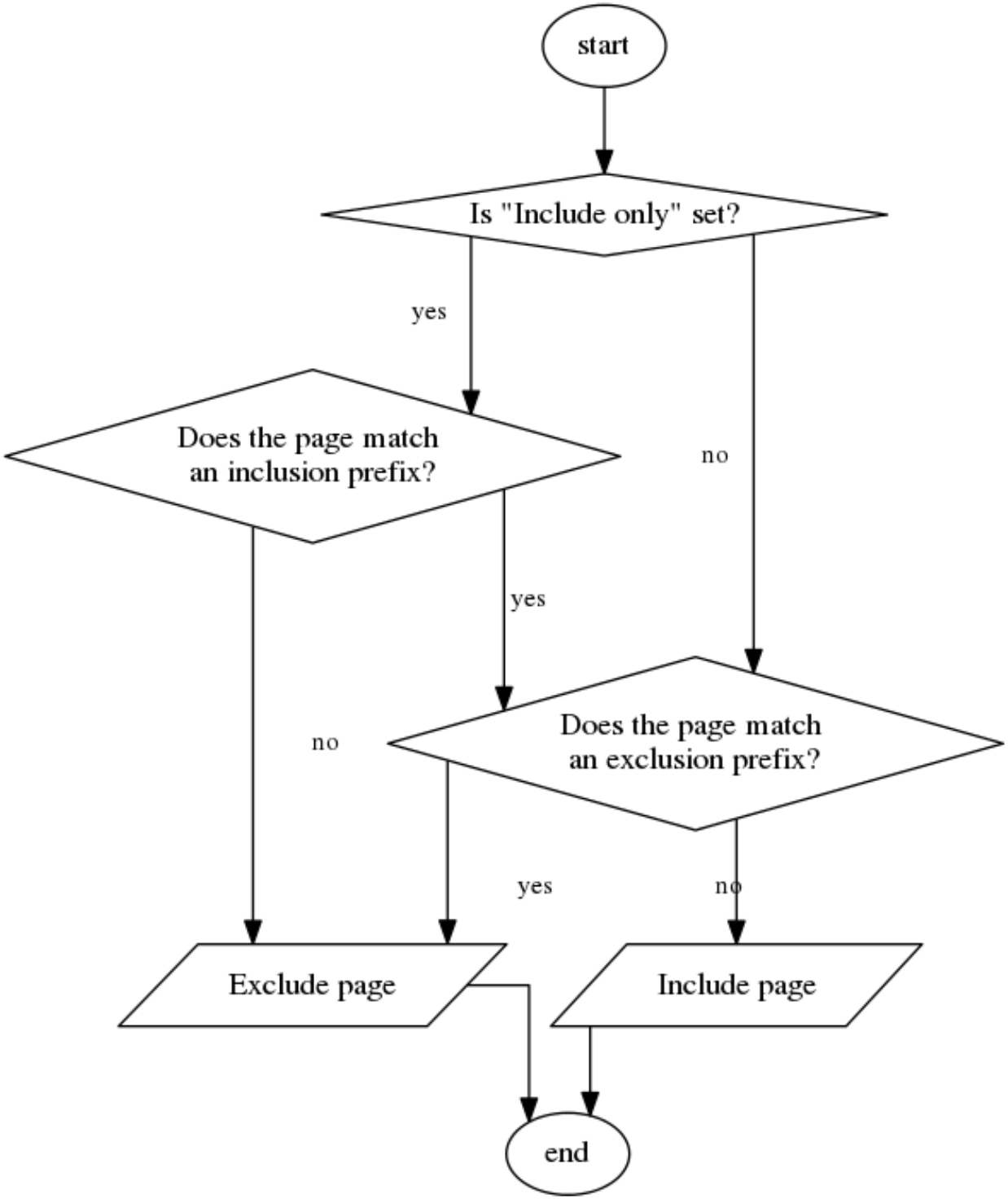
Unless you specify further inclusion rules and fine-tune your scope with exclusion rules, the proxy will ignore all pages outside of this directory.

Except

Exception rules don't need to be enabled "generally" in the way that inclusion rules do: if you add an URL prefix as an exclusion and leave the checkbox to its left enabled, it will *always* be applied.

Rule Application

Whenever the proxy comes across a page in an affected context (crawling, serving, translating, etc.), it will evaluate it according to the rules you provide. This process is summarized in the flowchart below:



Inclusion/Exclusion Evaluation

A few points of note concerning inclusion/exclusion rules:

- pathnames are first checked for inclusion, then exclusion.
- a pathname has to match only one from the set of enabled rules. Each such rule is applied to the path in sequence until a match is found or there are no more rules.

- the rules are **strings** and matched from the beginning of the pathname. The proxy does not analyze them in detail or produce complex internal representations.
- query parameters are supported (but be careful, you can't really count on a query parameter to have a set position!).
- if a path falls outside of the scope of your inclusion rules or an exclusion rule applies to it, it will be greyed out in the page list and the text "Excluded by rule" will be visible next to it.
- Rule-excluded paths cannot be un-excluded directly or via the Mass exclusion dialog, and the hover menu button you can ordinarly use for the purpose will be missing. You need to edit your rules if they gobble a path that you want.

NOTE: If all pages are excluded on a project, crawls *cannot be started*, even if the currently active rules would allow for the inclusion of *some, as-of-yet undiscovered page*. In this case, crawls will exit after 0 pages visited. You have to ensure an entry point: that at least one of the known pages is in an included state. Otherwise, the crawler can't set its foot in the door.

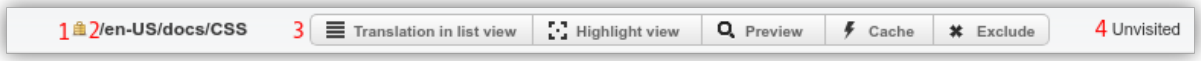
Though it may seem nonsensical to exclude every single URL on a project, we note this unusal case because it can come about from inadvertent use of *inclusion rules*.

Consider, for example, that if you set `/en/` as the sole "Include only" rule on your project, but no page starting with `/en/` is in the page list, then not a single valid entry point is provided to the crawler.

Page List

The page list contains all pages that the proxy has seen and collected previously, either via crawls or via user visits (to prevent the page list from being automatically extended in this latter fashion, enable "Page Freeze" in Advanced settings).

Parts of a page list entry



1. HTTPS icon
2. pathname
3. hover menu
4. translation state

The proxy handles HTTP and HTTPS pages *separately* on the assumption that content between the two can potentially differ. You will see a *yellow padlock* icon next to HTTPS pages. Click on the pathname of a page list item to visit the original page.

Hover menu

Contains a set of useful proxy features that can be triggered in connection with the page in question.

Translate in List View & Highlight View

Click on either of these two buttons to open the Cloud Workbench on the given page. Very useful when you want to do some targeted editing.

Preview

The proxy has a number of proxy modes available, and two of them are exposed here for each page. Click on the "Preview" button to open the preview domain in the currently selected language to see how the page looks in translation (but outside the Workbench, as a visitor would see it).

Hold the `Ctrl` button and click on "Preview" to open the so-called X-proxy, which you can use for debugging purposes (such as detecting unusual sources of translatable content, JSON response hunting, etc.).

Cache

Depending on your Source and Target cache settings, what you see over the various proxy domains might be different from what you expect. Use the Cache dialog to check the dates of the various caches and optionally void/flush them for the page (see Dashboard > Page Cache for a full cache flush).

Exclude

Click on this button to manually exclude the page. Has the same behavior as the Mass page exclusion dialog accessible on top. This form of exclusion is the most "targeted" approach you can take.

Manually excluded pages are shown as crossed out.

Translation State

There are a number of states that a page can be in throughout the lifetime of a project (we have already alluded to this fact above in connection with filtering options). The current state of a page is shown in the rightmost column of a page list item.

1. **Unvisited**: indicated pages that were already seen in the source of another page, added to the list, but for one reason or another, not visited by a crawler. For example, if you run a Discovery with a page limit of 8 and allow it to collect new URLs at the same time, it will likely visit the landing page, "/". The landing page is likely contain many references to various pages on the same domain. THe Discovery will exit after 8 pages, but also add the rest as "Unvisited".
2. **Discovered**: a page that was included in a word count previously is said to be Discovered. The content on it is accounted for in at least one of the word count statistics, but the translatable content **has not been extracted for translation** yet.
3. **NEW**: the "new" in this part refers to the fact that the page is content extracted and in a "pristine" state: no entries on it have been translated yet. It is usual to find pages in "NEW" state right after the first content extraction crawl (when a project has all the content ready for translation, but no translations have been provided yet).
4. **Progress bar**: will be shown when a non-zero percentage of the entries that the given page refers to have been translated. Note that due to automatic propagation, translating one entry can influence the progress bars of multiple pages). Hovering over the progress bar will show more specific numbers in the tooltip, but progress is also color-coded as follows (end value decimal places are omitted):

(a) `progress < 30%` - red

(b) `progress < 60%` - yellow

(c) `progress < 100%` - green

(d) `progress = 100%` - blue

Last Seen Status Code

The proxy also provides information on the status code with which the individual pages last responded with.

Only the most important subset of status codes is reported. `301 Moved Permanently` and `302 Found` redirects are shown in green, `404 Not Found` and `500 Internal Server Error` status codes are displayed in red.

Refer to Google or the very thorough RFC 7231 for a detailed description of HTTP status codes.

Page deletion

A project, especially one with few inclusion/exclusion rules or one that uses URLs with many query parameters can over time accure many pages in the page list. Page deletion is not available on the Dashboard for reasons of content safety and simplicity at this time.

If you'd like to have some subset of the pages deleted from your project, feel free to contact support and we'll help you with project cleanup. In the meantime, stay tuned until page deletion is introduced as a user feature.

Resources

Resources are binary content, such as images, PDFs, CSS and textual sources such JS files, etc. It is important to remember that the content of these resources is not extracted for translation, so you have to translate / edit them separately.

When you open this screen, a subset of all Resources collected during a previous crawl will be listed. Scroll down to load more, or use the search field for lookup (as some websites are quite Resource-heavy, the search function is recommended).

Hover and click over any Resource to display a localization dialog for it. You can click on the green "plus" icon to mark the Resource as translatable. After doing so, a few fields will be added to the dialog: a source and a target language text box you can use to transcribe the Resrouce, and a target URL field which will contain the the URL of the localized Resource on the proxy. Use the "Choose File" button to upload a localized Resource.

Note that you can only localize Resources *one at a time* and on a *strictly target language basis* (make sure you have the appropriate target language selected on the left side of the screen).

You can click on the flags on the lower left side of the frame to check the localized and original versions of a Resource (especially images, which will be displayed within the dialog). Document browsing, however, is not supported on the Resources screen.

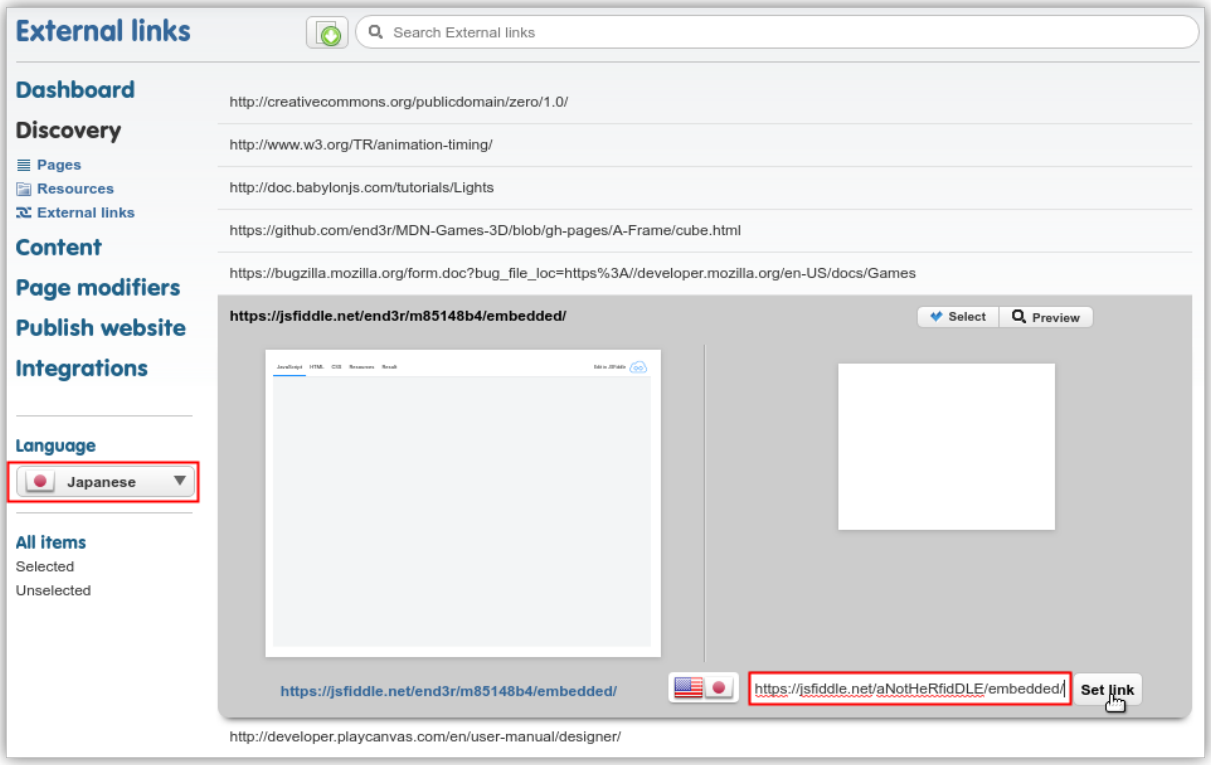
External links

External links point to domains different from the project domain, including any of its subdomains. If you are working on a project for `www.example.com`, `blog.example.com` is considered *external* to it (in addition to any other domain). Content seen as an external link will be collected and listed (and all external domains seen will be listed on the Discovery screen) in this menu.

Each of these references can be localized, which means changing it to point to a different resource for the target language (which must be provided externally in the same format as the original). The proxy will replace the given external link automatically in each language where an alternative is provided.

You can start from either Discovery > External links or Content > External links.

1. select a language in the left side dropdown into which to localize.
2. click on "Select" in the hover menu of a link to start the localization process. Alternatively, if you have already clicked on the link and the localization frame is active, click on the "Mark this external link as localizable".
3. input the external link that you want to use.
4. click on "Set link".



Embedded videos are frequent targets of such external link localization.

This feature is **not** meant to create gateways between two separately translated projects. To integrate references between projects in a consistent and seamless manner, use project linking.

Content

General

The main difference between Discoveries and Scans is that Discoveries do not extract or store content. They are used for statistical purposes, such as providing a word count or mapping a site's URL structure.

A Content Scan, however, writes into the database. It extracts and stores the source content to be translated. New source words added to the database are billed, therefore care must be taken when starting Scans. If you are not really sure about a site, stick to Discovery until you gain better understanding of its structure! Unlimited Scans especially require attention: they will relentlessly add all content to the database according to the specification you set for them.

Settings you used to experiment with Discoveries can also be used to initiate Scans - for a detailed explanation of the different settings, please take a look at the Discovery chapter of this section.

There is button on the main screen that you will not find in the Discovery menu, that is "Update word count", which you can use to refresh the statistics window during or after an extended Scan.

Pages & Resources & External links

Currently, these three submenus are duplicated between the Discovery and Content menus. See the pertaining Discovery section for details.

IMPORTANT! There is an important difference between the Add Pages dialogs of the two page lists: If the Add Pages dialog is opened from Discovery > Pages, it will start a word count Discovery; opening it from the Content > Pages menu will result in a content extraction Scan!

Manage segments

By clicking here, you will be taken to the Workbench on a new tab. The Workbench is the working end of the proxy service - please visit the Workbench section of this manual for the details of how to use it.

Work packages

Work package generation is the recommended method of dealing with exports/new content. Along with the general import and export features, work package generation becomes available after the first round of Scanning (Content Extraction) has finished and translatable content (segments) has been placed in the database.

Initially, the screen will be empty, but if you click on the blue "plus" icon, a dialog will pop up where you can generate new work packages using a variety of settings at your disposal. Let's take a look at them one by one.

Work package generation

Work package name You can name your work package any way you like. As a suggestion, it is recommended that you give it a name that accurately reflects its contents, so as to make project management using work packages easier. So, for example, if you generate a work package for Arabic as a target language on April 10th, 2016 for all new content, you would name it something like *ar-SA-untranslatedContent-2016-04-10* and so on.

Source language The source language for the project is displayed in this field (settable from the Dashboard main screen).

Work package language You can add multiple target languages for the Generation Process, but remember that a separate Work Package will be created for each target language.

Timeline You can specify which Content Extraction cycle you'd like to see in the Work Package that is to be generated. Click on "change" and use the timeline window to specify the time period, and the generator will limit its attention to just that timeframe. Select any two timestamped buttons to define the start and end of the timeframe.

Filter pages You can use your current selection (set on the Pages list screen), or define specific selected pages (both paths and full URLS are allowed) or folders using path prefixes. When you use a predefined list of pages (selected pages) the additional option of including both the HTTP and HTTPS versions of pages will be displayed as a checkbox.

Options Miscellanious options you can use to fine-tune the properties of your Work Package.

Split your package at preset words to create easily manageable chunks to assign to translators. Disable it to put everything in one big file (Note, however, that Work packages will be automatically split at 30.000 words)

You can choose to create separate work packages for hidden elements, and set the Work package generation to automatically take care of exporting for you. As with the general import function, you may elect to export only those entries that have no translation yet.

If you configured XTM, XTRF or Dropbox to integrate with the proxy, you may instruct the software to automatically send the exported XLIFF files to either one of those services. Use the dropdown menu to make your choice (only available if the checkbox is checked).

Work package list

You will receive e-mail notification about your generated work packages and they will appear in the list. You can use the refresh button to get an up-to-date view of your work packages.

If you hover over a work package, a floating menu will appear to the right with two options: XLIFF export and Progress report generation.

Import translation

When your XLIFF files have made a round in your favourite CAT tool, it is time to reimport them, which is a straightforward process - click on "Import translation" and the Import dialog will pop up, where you can click on "Choose file" to browse for your translated XLIFF files. Click on "Import translation" and wait for the import process to finish.

You may optionally set the workflow status of imported segments to Translated-CONFIRMED, and you can use the second tab in the dialog window to reimport translations from your Dropbox account.

NOTE! Importing is not instantaneous, it may take some time to finish depending on the size of your XLIFF!

Export translation

By clicking on this link, the general Export dialog will open. Most of the options will be familiar from the Work package generation dialog. Choose a file format of your preference (with XLIFF being the *de facto* industry standard, it is recommended that you go with this option, as it is more structured than CSV).

Languages to export You can choose to export more than one target languages in one go, or choose to export them all.

Export range Click on "change" and use the timeline window to specify the time period, and the export function will be limited to that timeframe. Select any two timestamped buttons to define the start and end of the time frame.

Export You can choose between exporting only those entries that lack a translation, or include all entries.

Unique segments only Check this option to ensure that identical segments are not duplicated in your exported XLIFF file.

Skip exlcuded and pending segments from export Export only those segments that have been approved for translation on the Workbench.

Trim export to contain as few tags and whitespaces as possible Use this option to clean the source segments of tags for the purposes of translation.

Split enter the number of segments that you'd like to have in a single file, and the export will split your segments into separate files accordingly.

Copy source to target where empty Some CAT-tools require a target entry on import. Activating this option will copy the source content to the target entry during the export, so the tool can be used to edit it.

Send XLIFF files Choose among your XTM, XRTF or Dropbox accounts to send your exported files to. Each of these accounts has a link at the bottom of the dialog that will take you to the given account's setting screen.

Click on Export to execute. Wait for the process to finish - you will receive an e-mail when your files are ready.

Previous exports

Although the notification e-mail does contain the link to your exported file, there is no reason to track it from your mailbox. This dialog will list and organize all previous exports for your viewing with the appropriate timestamp, target language and word count displayed right next to an Export list entry.

Currently running processes are also displayed, with a spinner indicating that the Export is still underway.

Page Modifiers

General

The proxy makes translation of websites relatively easy, but the web is an admittedly complex environment that can surprise in countless ways: changes to the page language can cause various layout issues to surface, especially if the developers of the site did not anticipate that the site might get translated.

word length differences between the source and target languages might cause the menu to crowd.

Differences between the lengths of different text blocks can cause otherwise well-designed CSS rules to behave in unexpected ways.

The plethora of plugins that run on a modern website - there is a big pool of possible glitches right there.

Fortunately, changing the site content over the proxy is relatively easy. The Page Modifiers function is made available for this reason: to empower you to add your own CSS rules and JavaScript snippets to influence the way a given proxied site looks to the user.

Because the datastream must pass through the proxy to have the translation embedded, the proxy can insert JavaScript modifiers, modify style sheets, even embed entire pages that do not exist on the original site.

Note that CSS rules and JavaScript are injected into each page that is served over the proxy, including page content overrides defined in Dashboard > Content override.

CSS editor

The proxy can be used to insert locale-specific CSS rules into the site being served. The rules are inserted as the last element of the head on every page served through the proxy. A very common use case for this feature is RTL conversion of a website: almost always necessary when one of the target languages is Arabic.

There are a couple of caveats:

It is good practice to make each CSS rule language specific (Easyling will always insert the appropriate locale code into the HTML):

```
html[lang="de-DE"] ul.one-selector li a {
  float: left;
}

html[lang="fr-FR"] .another-selector h2 {
  height: auto;
}
```

If you omit the `html[lang="fr-FR"]`, the CSS rule will be applied in all target languages, which might not be the behavior you expect.

Javascript editor

An editor with syntax highlighting support is available on the Dashboard. You can add JavaScript code here, which will be injected into the `head` tag of each page. By default, it contains placeholder code that demonstrates page modifier use. The following snippet also demonstrates a plausible page modifier:

```
(function() {

    "use strict";

    document.addEventListener("readystatechange", function() {
        if (document.readyState === "interactive") {
            resizeTextBoxOnContactPage();
        }
    }, false);

    window.addEventListener("resize", function() {
        resizeTextBoxOnContactPage();
    }, false);

    // make function page-specific. Altenatively, add check to eventListener above
    function resizeTextBoxOnContactPage() {
        if (document.location.pathname !== "/contact-us") { return };
        // implementation details here
    }

    // more function definitions here

})();
```

IIFEs (Immediately Invoked Functional Expressions), such as in the script above, are a generally good pattern to use with a page modifier. In this way, a programmer can continue to work within the confines of their local scope, and the function and variable definitions will not intrude on the global namespace - when it comes to page modifiers, it is very important that the modification has as little chance to clash with site code as possible (unless in a considered move).

The structure below can serve as an even simpler skeleton for a page modification:

```
(function () {

    "use strict"

    $(document).ready(modifyPage); // if jQuery is present

    function modifyPage () {
        console.log("modifications go here");
    }

})()
```

Of course, the possibilities of code injection are endless. See our tutorial for site search integration for a comparatively detailed example to help you get a glimpse of the possibilities of injected JavaScript.

The code injection feature puts the power of all client-side coding at your fingertips, and a truly in-depth discussion of the core web technoliges is not possible in this documentation. The excellent Mozilla Developers Network contains all the details you could ever need, and the W3mschools website contains useful tutorials on various web-related topics if you are just starting out.

Development Tip: a userscript extension, such as Tampermonkey can be used to develop page modifiers locally. While the Dashboard editor is adequate for basic editing, it is not meant to replace the many powerful editors available.

Tampermonkey can `@require` your script from the filesystem if it is granted `file://` access (supported by Google Chrome), in which case you can use your editor of choice and retain preview-on-refresh capability.

But keep security in mind!

Page Content Overrides

The proxy relies on the original site for substance: both translatable content and server-side services are provided by the original. The proxy relays and processes request-responses between that server and the visitor.

PCOs, as they are usually abbreviated, are an exception to this rule. They are "virtual" pages that are either created completely from scratch (and thus may not even have a counterpart on the original) or they override URLs that are accessible on the original server with wholly custom content over the proxy.

If a visitor requests an URL over the proxy that has an associated PCO, then the corresponding request is not sent to the origin (even if the URL is made unique by query parameters) and the PCO is served instead. A PCO, in a similar fashion to responses from the original site, will go through the translation pipeline. This bears repetition: **the contents of a PCO can be extracted and translated**.

Headers

Content Types

The source need not be HTML: any custom content-type can be entered (as long as it is a textual type), such as `text/xml` or `application/javascript` along with customized cache headers and status codes.

Of the many content types that can be used, the behavior of `text/html` is special in an important way: CSS and JS page modifiers on the project will be **injected** into such PCOs. The corresponding `<style>` and `<script>` tags are injected even if the PCO is a simple HTML snippet (that is, it lacks an `<html>` or `<head>` tag).

Syntax highlighting is automatically activated for a subset of content types after the Header field is filled out.

Cache Control & Pragma

Loading overrides over the proxy also count as page requests, so it is recommended that you add an optimized `max-age` value to each PCO and make them publicly cacheable. Placeholders in each field indicate reasonable defaults. At the same time, you'll want to set `max-age` to a smaller value *during testing*, in order to avoid having incomplete or work-in-progress PCOs cached for a long time over the network.

Response Codes & Location

The 300-family of status codes requires the Location header to be defined as well, to specify the target of redirection. Note also that only those HTTP status codes can be used that are permitted by the Java Servlet class. Detailed documentation can be found here.

Search result pages

The fact that URLs associated with a PCO are never requested from the origin server can be beneficial when implementing site search. If a distinct result page is used to display results, an empty version of that original can be copied verbatim into a PCO. This is particularly useful in cases where site search load is considerable over the proxy and the origin server would benefit from not having to handle search requests that it is not equipped to answer.

Scripts/Libraries

The PCO feature lets you add JS libraries or CSS stylesheets over the proxy as PCO with the appropriate Content-Type. A script declared in this fashion can be included in the `<head>` tag by going to Page modifiers > Javascript editor and **adding a reference to the PCO URL** at the top.

This approach is not recommended for jQuery or similar standard libraries. Inclusion of a script accessible via CDNs should be done using those external URLs, if not for any other reason, then because such external domain requests do not cost money over the proxy.

If, however, a standard library has to be changed in some small way over the proxy, PCOs can come in very handy.

Placeholders / Redirection

PCOs can be used to create "Under construction" pages or to ensure that the user is steered away from a specific URL via a redirection. But keep in mind that there is no prefix support for PCOs: they are always **exact URLs** and can only be applied as overrides in a targeted manner.

Binary Content

Binary content PCOs (including images and webfonts) are not supported. Use the resource localization features to replace images over the proxy with their translated counterparts.

Publish Website

General

In this menu, you are provided with the means of publishing your website on a pretty domain after translation tasks have finished.

The publishing interface has two options to make the site available to the web at large: serving domain mode and subdirectory publishing mode.

Domain settings

This screen opens by default after clicking on "Publish website". The following settings and fields are available:

Target language Choose which target language you would like to configure a serving domain for.

Temporary domain This is the preview proxy you can use to preview your translations live (this is immutable).

Access control you can add basic HTTP authentication to the proxied site in order to restrict access to it. Click on *Configure access control...*. Select the proxy modes that you'd like to limit access to, enter your username and password. You can add multiple users to the list.

Serving domain mode (de.example.com)

In the serving domain mode, the translation proxy will publish the translated site either on a subdomain of the original (the default behavior, such as `de.example.com`), or on a completely separate naked domain (such as `example.com`). In order to use this mode, you (or the client) will have to modify the DNS settings corresponding to the original domain - the three to five records (three for subdomains, five for new naked domains) that need to be

inserted in your DNS settings are found below the two input fields, and will change as you enter the desired serving domain.

CNAME settings

There are three CNAME settings that are required on your domain to enable publishing of your website. Each of the lines in the table that is displayed has a specific function:

CNAME 1 allows mapping of the subdomain in Google AppEngine, enabling us to alter the datastream and translate the site. This row is computed from the serving domain entered in the second input field, and needs to be added once per serving domain.

CNAME 2 The second line determines where the currently selected target language will be published. This defaults to the language code, but you are not obligated to keep it that way. This has to be entered separately for every target language you publish.

CNAME 3 The third line verifies ownership of the original domain. This is computed from the user's ID who is currently looking at the publishing page (i.e. different users will see different values, so one person should communicate this to the client and hit the Verify button), and needs to be added only once per serving domain as well. You can set the subdomain and domain where the translated site will appear.

After all the settings have been entered into the DNS records, there is short time while the changes propagate and are replicated across the world. This can vary wildly with the DNS and hosting providers, taking anywhere between one and twenty-four hours. It is recommended to wait out the twenty-four hours, as the translation proxy will not be able to save the published domain until all checks are passed.

HTTPS & SSL certificates

Additionally, if the original site was HTTPS, or the translated sites will be served over a secure channel, an appropriate certificate and its associated key will also be needed. Ideally a wildcard certificate (one certificate for all subdomain), but Extended Validation certificates can also be used, although they require more setup work.

Our support team can also provide you with the *Certificate Signing Requests* necessary for the generation of these certificates - this will also have the added benefit of not having to send the private keys over the wider web for recording in AppEngine.

Additionally, we have a *Managed Certificate* program, where the proxy handles SSL certification automatically for published websites. The Managed Certificate program has a cost of 50EUR (60USD)/proxied domain/year or 100EUR (120USD)/proxied domain/**three** years.

See the pertaining section of the documentation here.

Subdirectory publishing (example.com/de)

The alternative to subdomain-based publishing is to retain your own domain and publish the site as a subdirectory. I.e. the translated pages will appear under separate paths under the same domain as the one the project was created for (the original domain).

Due to the way the proxy works, this requires a reverse proxy configuration to be placed in front of the webserver. A variety of load balancer/reverse proxy solutions are available on the market, with `nginx`, `CloudFlare` and `AWS CloudFront` being three of the most well-known solutions available. See the vendor documentation for the details of setting up a reverse proxy (do note that nowadays, reverse proxies are monumentally powerful network solutions, and discussion of all their features is beyond the scope of this introductory description).

If you go to the Dashboard and go to the Publish Website menu, you'll see that the *Subdirectory Publishing* option displays an example `nginx` configuration. Below is a list of the absolute minimum configurations required to achieve a

workable reverse proxy using the three most popular webserver systems (Nginx, Apache httpd, and Microsoft Internet Information Systems)

Nginx

```
location ~* ^/(de) {
    resolver 8.8.8.8;

    set $xhost de-de-gereblye.app.proxytranslation.com;

    proxy_set_header X-TranslationProxy-Cache-Info   disable;
    proxy_set_header Server $xhost;
    proxy_ssl_name $xhost;
    proxy_set_header X-TranslationProxy-EnableDeepRoot true;
    proxy_set_header X-TranslationProxy-AllowRobots true;
    proxy_set_header X-TranslationProxy-ServingDomain $host;
    proxy_set_header Host $xhost;

    //old nginx
    //  proxy_pass $scheme://$xhost;
    //new nginx:
    proxy_pass $scheme://ghs.googlehosted.com;
}
```

Apache httpd

```
<VirtualHost *:80>
    Define domain example.com
    Define previewDomain http://de-de-gereblye.app.proxytranslation.com
    ServerName ${domain}

    <LocationMatch "/de/(.*)">
        RequestHeader set X-TranslationProxy-ServingDomain "${domain}"
        RequestHeader set X-TranslationProxy-Cache-Info "disable"
        RequestHeader set X-TranslationProxy-EnableDeepRoot "true"
        RequestHeader set X-TranslationProxy-AllowRobots "true"
        RequestHeader set Host "${previewDomain}"

        ProxyPass "${previewDomain}"
    </LocationMatch>
</VirtualHost>
```

Microsoft IIS

Please note that for IIS, server variables need to be whitelisted before this configuration is applied correctly!

```
<?xml version="1.0" encoding="UTF-8"?>
<configuration>
    <system.webServer>
        <rewrite>
            <rules>
                <remove name="ReverseProxyInboundRule1" />
                <rule name="ReverseProxyInboundRule1" stopProcessing="true">
```

```
                    <match url="^/de(.*)" negate="false" />
                    <action type="Rewrite" url="http://de-de-gereblye.app.
↪proxytranslation.com{R:1}" />
                    <serverVariables>
                        <set name="X-TranslationProxy-EnableDeepRoot" value="true" />
                        <set name="X-TranslationProxy-ServingDomain" value="example.
↪com" />
                        <set name="X-TranslationProxy-Cache-Info" value="disable" />
                        <set name="X-TranslationProxy-AllowRobots" value="true" />
                    </serverVariables>
                </rule>
            </rules>
        </rewrite>
    </system.webServer>
</configuration>
```

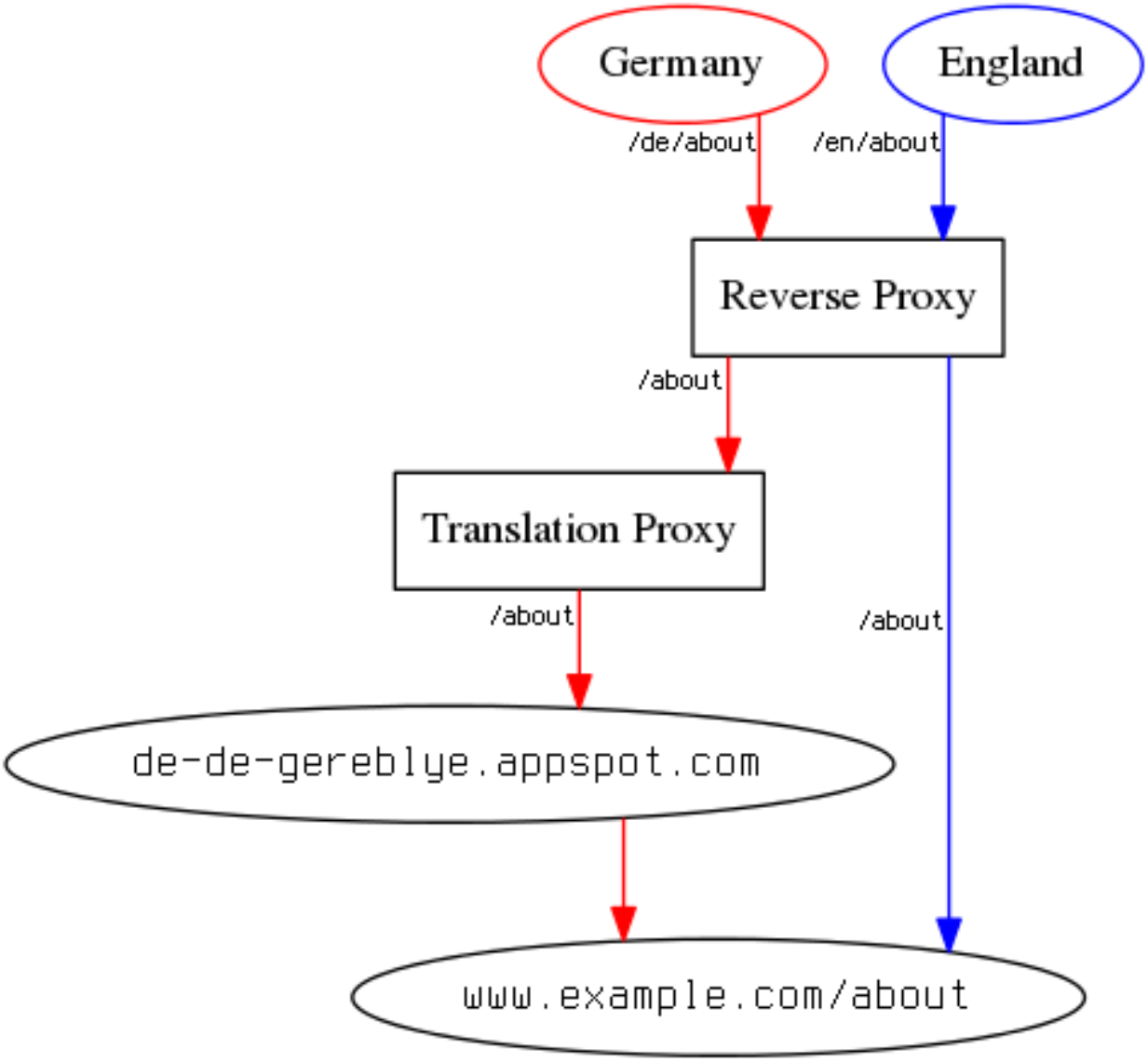
The goal of any reverse proxy configuration interoperating with a translation proxy domain requires that its configuration reflect the intent of the configuration example above.

Let's clarify this pithy snippet with a more general, high-level explanation of a reverse proxy in operation.

Example:

Suppose that we know the following about a translation project about to be published using a reverse proxy.

1. The origin server domain is `www.example.com`.
2. Source language is English.
3. Translation exists for German
4. German serving domain: `de-de-gereblye.app.proxytranslation.com`



The task of the reverse proxy, standing right at the beginning of the pipeline that will serve a client request, is to decide which target language is requested by the user and to ensure that the request is relayed to the appropriate domain that can respond with content in the appropriate language.

In our example scenario, the reverse proxy has to make one decision: is the user requesting a resource in English or in German?

Translation proxy

From our perspective, The most interesting case is when a user from Germany requests `www.example.com/de/about`, the reverse proxy decides that the target language should be served via Translation proxy. It relays the request to the Google Cloud, where it is resolved to what we call the **temporary serving domain**, defined as `de-de-gereblye.app.proxytranslation.com`. In the serving subdomain mode, this domain is hidden from the user by the DNS settings added. In subdirectory publishing, the reverse proxy hides the temporary domain.

You can see that `de-de-gereblye.app.proxytranslation.com` will – same as with subdomain publishing

– relay the request and all necessary request headers to the origin server, which will respond accordingly with source language content that the Proxy then processes on the way back and sends long to the client in a translated form.

Setting aside the exact details of that cloud translation pipeline, that's about it.

Source language request

Requesting the original content is a relatively straightforward process that we should nevertheless describe in brief for the sake of completeness.

If a user in England requests `www.example.com/en/about`. The reverse proxy strips the `/en` prefix, decides on the language to be served and relays the request to the origin server.

In this case, there is no more proxy mediation (that is, no Translation Proxy) between the origin server and the requesting client, so the server response is returned and the user can peruse the webpage in the original language.

Note that on the origin server, the `/en` prefix does not exist as part of the directory structure - it is a virtual prefix used by the reverse proxy to dispatch to different domains based on the target language.

If the origin server is capable of providing content in more than one target language, the reverse proxy should presumably do the same thing for each of those target languages. If a request for `www.example.com/de/about` can be fullfilled by the origin server alone, the reverse proxy will relay that request straight to the origin server (where it is assumed that the server backend will make the decision based on the HTTP request headers received).

Language Selector

As translation of a website is nearing completion, it is usually necessary to prepare a language selector to make the various languages more accessible. There are two options:

```
1. develop a custom selector in-house
2. use one of the proxy default selectors.
```

In all cases, the addition of a language selector will need to be done on the original server and then propagated to the translated domains through the proxy.

Proxy Solutions

To spare you the expense and time of developing a custom language selector, we have prepared two default solutions you can use. Each of these require the minimum possible intervention on the original site for deployment. Note, however, that seamless visual integration of the selector requires some fiddling with the CSS rules. The following two types of language selectors are available:

Sidebar

The sidebar is an easy-to-deploy-and-use language selector (with limited customization options) that doesn't require any changes on the original site beyond the addition of a single `<script>` tag to the source code. The language selector is a hovering sidebar which displays the flags of the selected target languages.

The sidebar is an easy-to-deploy language selector that displays flags for each specified target locale on the left or right side of the website. The horizontal orientation and display names are settable.

Set-up steps

On the Dashboard:

1. Click on the Publish Website menu
2. Select "Sidebar"
3. Select "Left Side" or "Right Side" according to your preference
4. Specify the Target languages by enabling the last checkbox on the screen

Note: You may also declare custom language names to be displayed by using the dropdown selector or choose "Language" to display only the language, but not the Country.

On the original site:

1. Finally: copy the embed code from the textbox and add it to the end of `<head>` tag on the original site. With the tag added to the original site, a reload should reveal the language selector.

Dropdown

The Dropdown is the more customizable option. One bit of additional adjustment in the original source is necessary, however: besides adding the `<script>` tag, a dedicated `<div>` tag is also necessary – since the dropdown is more closely integrated with the site structure (that is, it has to be positioned in relation to other elements), this `<div>` tag will be used to locate the designated location of the dropdown.

Setup steps:

On the Dashboard:

1. Click on Publish Website menu
2. Select "Dropdown" (default)
3. Declare a unique class name for the `<div>` that will contain the language selector. The default is `el-dropdown` but you may specify any class name that does not clash with any other classnames in the original HTML. For the purposes of this tutorial, the classname will be `el-dropdown`
4. Select a display template. You may choose between "Language (Country)", "Language" or entirely custom names according to your preference.
5. Specify the Target languages by enabling the last checkbox on the screen.
6. Copy the embed code from the textbox.

On the original site:

1. Locate the position in the markup that you wish to insert the Dropdown
2. Extend the HTML markup with a `<div>` tag. Furnish it with the previously specified class. In this case, it would look something like this: `<div class="el-dropdown"></div>`
3. In addition, Add the embed code copied from the Dashboard to the `<head>` tag of your page.

Post-setup:

At this point, the language selector is displayed after reloading the page in your browser (the JavaScript referenced in the `<script>` should take care of downloading and rendering it). The language selector contains a menu button, a container and target language entries you can customize via CSS rules. The language selector is self-contained for convenience, so it uses in-line styling. Postfix your CSS rules with "!important" to make sure that they are applied.

RYO or Existing Selector

It is possible that site undergoing translation is also running (or was previously running) a different localization solution, in which case a language selector is usually already present. It is relatively straightforward to extend an existing language selector with the proxy-published target languages.

Since that language selector is also proxied, it is important to protect the link pointing to the original language site from being remapped in the process.

In order to do this, the `__ptNoRemap` special class can be used to indicate to the proxy that it should not change any links in the given tag.

A brief example of how the original site link can be protected from remapping with `__ptNoRemap`:

```
<div class="selector-container">
    <ul id="selector">
        <!-- PREVENT REMAPPING OF ORIGINAL LINK - this requires a full domain-->
        <li id="en-US" class="language selected"><a href="//www.example.com" class="__
↪ptNoRemap">English</a></li>
        <ul class="dropdown">
            <li id="de-DE" class="language"><a href="//de.example.com">Deutsch</a></
↪li>
            <li id="fr-FR" class="language"><a href="//fr.example.com">Francais</a></
↪li>
        </ul>
    </ul>
</div>
```

You may also wish to use the translate=”no” attribute in places where you want the proxy to ignore the textual content of the HTML element.

Example

Depending on how involved it is, a language selector may have to react to user input/animate dropdowns/start XHR requests, etc. Its appearance and behavior depend on the number of languages, publishing method, site layout & design etc, which is beyond the scope of this manual. But to get you started with a suggestion, a commented minimal example in vanilla JS/CSS follows below, which is also available in a slightly modified form as a JSFiddle here.

```
(function () {
    "use strict";
    /* The target locale code, a two-letter language code plus a two
       letter country code, separated by a dash, is added
       automatically over the proxy. Either this value or the HTML
       markup might have to be adjusted to be applicable on the
       original site */
    var lang = document.querySelector("html").getAttribute("lang") || "en-US";
    /* The usual "readystatechange" event listener works if
       you need to wait

       document.addEventListener("readystatechange", function () {
            if (document.readyState === "complete") init();
       }))
    */

    init();

    function init () {
```

```
        selectDefault();
        setEventHandlers();
    }

    function setEventHandlers () {
        var items = document.querySelectorAll("div.selector-container li.language a");
        for (var i = 0; i < items.length; i++) {
            items[i].addEventListener("click", function (e) {
                selectLanguage(e.target.parentNode);
            })
        }
    }

    function selectDefault () {
        /* See if we can select anything in the language selector
           based on our locale code. */
        var base = document.querySelector("div.selector-container li#" + lang);
        /* ...and try to default to anything we've found */
        if (base !== "undefined") selectLanguage(base);
    }

    function selectLanguage (target) {
        /* 'click' event was triggered on any of the language selector
           entries. */
        var selected = document.querySelector("li.selected");
        var dropdown = document.querySelector(".dropdown");

        /* don't do anything if we would be switching to the same
           language */
        if (target === selected) return;

        /* SWITCH LANGUAGE.

           ...but we are only "swapping elements" in the menu (the
           code doesn't navigate in actuality). A target language
           domain or a CST query parameter could result in a page
           load.

           The "dropdown" approach showcased here is also an
           illustration of a frequent issue: when selectLanguage() is
           called, the selected target language is replacing the
           current one in the dropdown list. So, if you are particular
           about the target languages being listed in a set order, a
           hide/show approach might work better than DOM element
           swapping. */
        target.setAttribute("class", "language selected");
        selected.setAttribute("class", "language");

        selected.parentNode.replaceChild(target, selected);
        dropdown.insertBefore(selected, dropdown.firstChild);
    }
})()
```

Below are the associated CSS rules.

```
div.selector-container ul {
  box-shadow: 2px 2px 8px rgba(0, 0, 0, 0.5) !important;
  list-style: none;
```

```
  margin: 0;
  padding: 0;
}

ul#selector {
  list-style-type: none;
  text-align: center;
  background-color: #f5f5f5;
  max-width: 100px;
}

ul#selector li {
  list-style-type: none;
  list-style: none;
  padding-top: 10px;
  padding-bottom: 10px;
}

ul#selector a {
  color: black;
  text-decoration: none;
}

ul#selector li.selected:hover a {
  color: grey;
}

ul#selector li.selected:hover + ul.dropdown {
  display: block;
}

ul#selector ul.dropdown:hover {
  display: block;
}

ul.dropdown {
  display: none;
}

ul.dropdown li.language {
  padding-top: 10px;
  padding-bottom: 10px;
}

ul.dropdown li.language:hover {
  background-color: chartreuse;
}
```

RYO selector + Client-Side Translation

If a project is published using Client-side translation, either an existing language selector needs to be used, or a new one developed to trigger translations after selection.

CST translation is activated when the target locale code is stored in the browser with the help of the `__ptLanguage` query parameter.

In order to trigger language selection/translation in-browser, add this query parameter with the appropriate published

locale codes as values. Note that in order for the method to work, URL navigation has to happen once, which means a full reload is required in order to change languages.

Let's modify the exampe above to suit the needs of CST:

```
<ul id="languageSelector">
    <li class="language selected">English</li>
    <ul>
        <li><a href="/?__ptLanguage=en-GB">English</a></li>
        <li><a href="/?__ptLanguage=de-DE">Deutsch</a></li>
        <li><a href="/?__ptLanguage=fr-FR">Francais</a></li>
    </ul>
</ul>
```

The value of `__ptLanguage` will persist across sessions, the JS file will rely on it until the user changes their mind about their preferred language.

Visiting `www.example.com/?__ptLanguage=ja-JP` will change target languages into Japanese and store the setting. Visiting `www.example.com` the next time around will result in the site being translated into Japanese automatically.

The default browser locale can be detected successfully to be a published target language, CST will attempt a locale-specific translation without having had a query parameter provided to it.

Although that is about it, it is useful to keep in mind a few things when setting up a language selector to work with CST.

Somewhat different from the proxy method, CST requires that we provide all locale codes explicitly, especially the **original** language. Since CST *stores* the user's selection in the browser's local storage, the query parameter is necessary in order to allow the user return to the original language version.

CST publishing does not require any proxy traffic, but the Workbench In-Context editing screen remains available. This can sometimes result in the Client-Side Translation getting mixed up with the proxy-based translations, so if you are using the proxy modes for your translation work while publishing with Client-Side translation, it is generally a good idea to ignore the language selector in Preview.

Wordpress Plugin

Clicking on this menu will take you to the Wordpress site where you can download the translation proxy Wordpress plugin.

The proxy provides an option to install the Translation Proxy directly to the Wordpress CMS. In this case no Translation Proxy fee will be charged, but the foreign traffic shall be handled by the content owner's infrastructure. For more information see our blog or the WordPress.org.

Client-Side Translation

CST is currently in a live-tested beta phase.

What is CST?

To summarize briefly, Client-side Translation is a **publishing method** that lets you use your translations on the original domain.

As opposed to the proxy method, which involves standing between the user request and the original site and translating responses on-the-fly, Client-Side translation does not require the proxy pipeline to serve content.

That is, instead of being processed in the Cloud, translation of the page will happen in the user's browser, using a JavaScript based dictionary file that can be exported from your project and embedded right in the original site's HTML source.

Pros & Cons

Client-side translation is especially suited to lightweight, streamlined sites with comparatively few pages, or sites that are not publicly accessible, such as intranets.

It does not require the proxy pipeline to serve content, which means that there is no monthly cost involved.

Image translation is supported.

On the other hand, given the nature of the translation process, the target language content cannot be indexed by search engines.

Additionally, automatic change detection is not an option with a JS-based publishing method, so the only way to detect changes is by setting up a recurring crawl.

Publishing process

The publishing method is in open beta phase and is available for testing and live use with a bit of help from our end. If you are interested in taking it for test drive, please contact our support team and we'll help you set it up.

Client-side publishing consists of two steps:

Add stub.js reference to original site

Like embedding a video, including your published translations involves adding a single `<script>` tag to the original HTML source. The script tag looks like the following:

```
<script type="application/javascript" src="http://${PROXY_APP_DOMAIN}/client/$
↪{PROJECT_CODE}/0/stub.js"></script>
```

Once the script is placed on the original site, a dictionary file of published translations will be downloaded by the site automatically. Besides selecting the language, no further change on the original site is necessary.

Dashboard - Publish Your Translations

This step currently requires a "toggle" on our end, but keep an eye on our release logs until the publishing method becomes available as an automatized step!

Integrations

General

Easyling is capable of integrating with various external systems, TMSes, storage systems, and CAT tools. Currently, we can connect to the following systems

TMS

XTRF

- Export into XTRF projects

CAT tools

XTM

- Export into XTM projects
- Automatic import from XTM projects on completion of the workflow

MemoQ

- Live preview of translations using MemoQ Desktop
- Integration with MemoQ Web

Storage drivers

Dropbox

- Export into a designated folder, split by projects; import translations directly from uploads after email notifications

MT engines

GeoFluent

- Automatic language detection
- Use in pre-translation

Google Translate (Dictionary-based)

- Traditional machine translation on Google's infrastructure
- Use in pre-translation

Google Translate (Neural net-based)

- Machine translation using Google's neural net-based approach
- Use in pre-translation

Microsoft Cognitive Services - Translation API

- Microsoft Translation system
- Use in pre-translation

Itranslate4EU

- Use in pre-translation

Cookbook

Whitelabel Setup

Note: This article will use variable values, that will change for everyone. These variables are written here in the UNIX style of `${VARIABLE_NAME}`. When providing us information, simply replace the entire construct (not just the name itself) with the data in mind. There is also "global" variable to keep in mind: `${APP_DOMAIN}` refers to the domain chosen in point #3 to serve the translation proxy on.

1. A €200 EUR topup / month (recurring): the translation proxy under your own brand name is special service, offered to customers who cater not only to one or two clients, but put their weight behind the punch and open up whole new markets with our proxy solution.
2. A one-time setup fee of €200.
3. A custom domain name: you will need a place to serve the translation proxy (as well as any previews) on. Generally, our clients settle on `app.${yourdomain}.com`, but we can use practically anything that comes to your mind - the only limitation is that we are unable to serve the proxy on a naked domain (for instance, `yourproxy.com`). Just keep in mind that once you settle on something, and we set up your branded Easyling, it becomes fixed, so your decision is final.
4. Two logos: one goes on the Dashboard, the other goes on the Workbench. They should be transparent PNGs, ideally, but we can use other file formats as well. However, their dimensions are fixed: the Dashboard logo is set at 200x62px, while the Workbench logo needs to be 109x44px.
5. An SSL certificate: the translation proxy uses encrypted channels to communicate on, and for that, we require an SSL certificate to be made out for the domain name of your choice, and any subdomains it may have - the translation proxy uses your "app domain" to serve previews until they're published, so your certificate must be a so-called "wildcard certificate". This is a type of SSL certificate that is valid not only for `app.yourdomain.com`, but also `*.app.yourdomain.com`, needed in order to ensure SSL coverage of all project-specific preview subdomains (the names of which are created by combining an arbitrary locale code and the *randomly generated* project code). Certificate issuers are likely to request a Certificate Signing Request (CSR) for the certificate, which we will have to provide.

In order to generate a CSR for you, you'll need to provide a few pieces of data related to your company, which need to be incorporated into the certificate. Please provide the following by replacing the fields (this should look fairly familiar to your IT department) with the appropriate data.

```
1. `countryName_default = ${COUNTRY}`

2. `localityName_default = ${CITY}`

3. `streetAddress_default = ${ADDRESS}`

4. `postalCode_default = ${ZIP}`
```

```
5. `0.organizationName_default = ${COMPANY_NAME}`

6. `organizationalUnitName_default = ${ORG_UNIT}`
```

1. The final step is to configure your DNS servers; and if you use Google Apps for `yourdomain.com`, the setup process will require someone with Google Apps admin rights as well. You will need to add the following CNAME records to enable the translation proxy on your domain:
   (a) `${APPENGINE_KEY}.${APP_DOMAIN}` CNAME `${APPENGINE_HASH}` - these values will be provided to you.
   (b) `${APP_DOMAIN}` CNAME `ghs.domainverify.net`
   (c) `*.${APP_DOMAIN}` CNAME `ghs.domainverify.net`
2. In order for the translation proxy to be able to send emails from under your domain, you will need to provide authorization to the email service. This is done by adding specialized DNS records.
   (a) `mandrill._domainkey.${APP_DOMAIN}` TXT `v=DKIM1; k=rsa; p=MIGfMA0GCSqGSIb3DQEBAQUAA4GNADCBiQKBgQCrLHiExVd55zd/IQ/J/mRwSRMAocV/hMB3jXwaHH36c`
   (b) `${SELECTOR}._domainkey.${APP_DOMAIN}` TXT `${DKIM_KEY}` - these values will be provided to you as they are domain-dependent.
   (c) `v=spf1 include:spf.mandrillapp.com include:sparkpostmail.com ?all;` or if you're already using an SPF record, add `include:spf.mandrillapp.com include:sparkpostmail.com` just before the last operator.
3. Finally, if you want, you can specify the following information to customize the white label experience (this is completely voluntary and can be changed at any time):
   (a) name: Name of your branded product
   (b) greeter: Person signing the greeting emails for new users
   (c) team: Team name
   (d) greeter address: Email address of the person sending the greeting emails
   (e) greeter display name: Name of the person sending the greeting emails
   (f) noreply address: Email address used for automated emails
   (g) noreply name: Display name used for automated emails
   (h) quote and wordcount signer: Person signing the quote and word count emails

Once all seven main points are settled, and we have the logos and the SSL certificate with us, we'll set up the white label version for you, and you'll be ready to start cracking your target market wide open.

JS/JSON/XML Translation

In this section, we describe the process of translating content in JavaScript files and dynamic (JSON or XML) responses.

General

Translation of HTML is mostly automatized over the proxy. But websites rely on many additional resources besides the document itself, such as JS libraries, CSS stylesheets, webfonts, dynamic requests and images. Not all of these

resource types require translation, but JSON and XML responses frequently do. Such responses can also be very inhospitable to the translator and proxy specialist.

One of the problems is detection: proxy crawls/analyses do not operate in a browser-like environment. There is no headless browser or VM running in which a page load could be initiated or JavaScript evaluated for content detection purposes. Inherent complexity is another issue: the enormous diversity (to put it charitably) of web technologies in use nowadays prevents reliable automation of such a process.

But, though JS/dynamic content can slip under the radar at first, the proxy can easily translate it with some help.

Finding Content

An early investigation will reveal content that is unavailable to the crawler by default, and will save you trouble (of having to deal with untranslated content as late as in the review phase, for instance).

X-proxy

To check how a website is doing over the proxy, open it in the **X-proxy** mode, a specialized Preview mode available through the Dashboard page list. Click on Preview in the hover menu while holding `Ctrl` to open it.

Note that you need to *add at least one target language* and *select it in the left side menu* to access the preview.

The X-proxy replaces text it recognizes with x-es. Though not too impressive visually, it is an excellent research tool. It lets you *home in on undetected content*. To utilize it most effectively, combine it with your browser's DevTool.

Major browsers such as Firefox and Chrome allow you to do full-text search on (almost) all resources/requests used during and after page load. In Chrome, for example, you can press `Ctrl` + `Shift` + `F` to start a full text search in the DevTool.

The following screenshot demonstrates how the X-proxy can make untranslated/undetected content obvious:



Having removed all known text from the equation, you are free to concentrate on the "untranslated" parts. Findings will naturally be site-specific, but there are some familiar, recurring scenarios:

1. **content is in `<script>` tags**: this is the simple case, as there isn't even a distinct resource to mark as translatable. `text/html` pages are translated by default, and JS content in them can be annotated right away.
2. **content is a string in a JS file**: aside from the necessary annotation process, you'll also need to ensure that the resource in question is marked as translatable.
3. **content is requested dynamically**: dynamic content can be tough to dig up. Many DevTools don't support full text search in XHR response bodies. If content is in plain sight on a webpage but the DevTool is not reporting any of it, then the content could have arrived via an XHR request. Check the "XHR" section in the Network tab after a reload. Aside from the inconvenience of locating them, dynamic request endpoints can be annotated and

marked in the same way as JS files.

4. **content is on an external domain**: this scenario requires some work. External domains require separate, but linked projects (add to this that you also have to ensure that URL references are mapped well, which can be difficult in a JS file), and the resources have to be marked and annotated on those projects to be translated.
5. **content is in an image**: though not strictly connected with the topic of JS translation, an "honorable" mention goes to natural language content *as image data*, also frequently revealed by the x-proxy.

There are many ingenious ways in which webpages encode content, and the proxy has various levels of support for all these schemes (usually involving a combination of features). When in doubt, feel free to contact support for advice!

Marking Resources

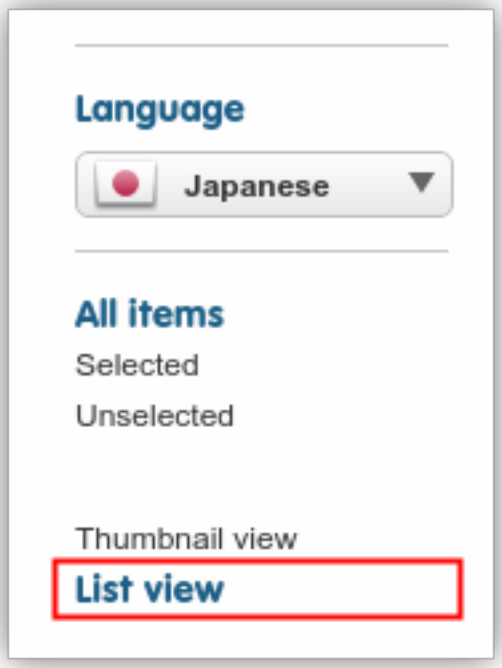
You can mark a resource as translatable manually on the Resource screen or using a prefix in Advanced Settings.

Manual

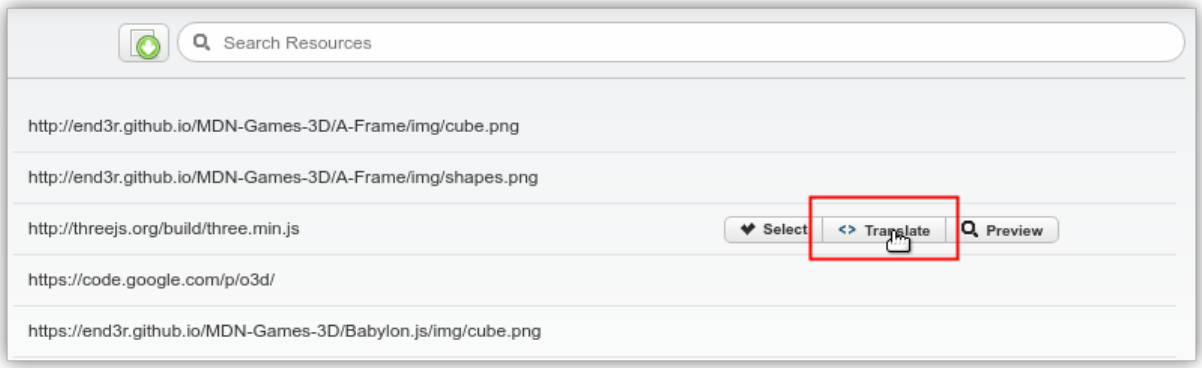
All collected resources are listed in Discovery > Resources and Content > Resources. All "pages" and files that are not of the `text/html` content type will go here.

What sets resources apart from pages is that by default, they have no associated *source entries* or translations. Marking a resource as translatable means declaring that it does have translatable text that can be stored as source entries (and accessed via the Workbench).

So, to mark a resource as translatable:



1. switch from thumbnail view to list view
2. click on the "Translate" button in the hover menu of a resource



The resource is moved to the page list and from that point onward, it can be opened on the Workbench. Note, however, that we have not yet told the proxy what and how to translate in it.

Prefix

You can do the marking via prefixes. Go to Advanced settings to find the "Mark Multiple Resources as Translatable" text field. Copy & paste the prefixes of your choice and click on Save to apply your changes.



Note that in the screenshot above, HTTP and HTTPS prefixes are handled separately, a recommended practice for sites that support both kinds of traffic. Prefixes are treated as simple strings by the proxy when matching them against a pathname. You are free to add as many of them as you like.

This feature is made available because cherry-picking resources for translation is not always feasible. For instance, versioned URLs are liable to create new resources on a project whenever a file is updated on the original site (the proxy keeps these URLs separate by default), but the new resources are not marked automatically.

You will recognize those cases where you want to apply the exact same translation rules and process to a set of URLs that differ in minimal ways. A resource prefix will let you do this without having to mark things one-by-one as they come.

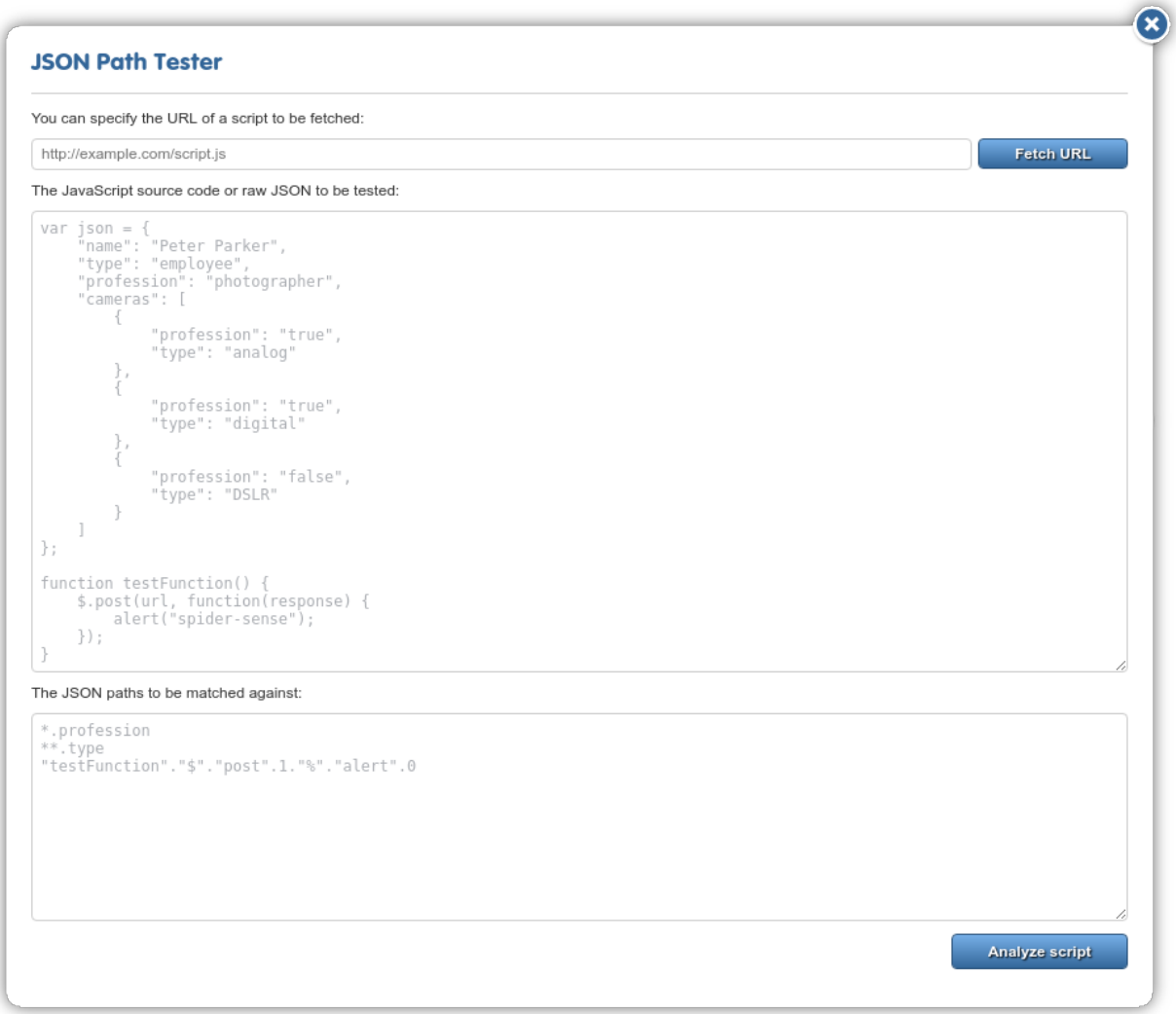
Annotating JS/JSON

Picking up JSON/JS/XML content *wholesale* would be both costly and unwieldy. When you have successfully identified the source of content and earmarked it for translation, the last major task is to annotate those parts of it that really want to translate. **JS/JSON paths** and **Xpaths** can be used for this purpose.

JS/JSON paths

JSON Path tester

Go to Advanced settings > JavaScript translation options, and click on the "JSON Path tester tool" link right below text field to open the tester dialog. It looks like this:



We'll use the following JavaScript snippet in the remainder of this section. It illustrates many use cases for JS translation:

```
(function () {

  var exampleVar = "Hello World!";

  var exampleUrl = "https://shadowcat.skawa.hu";

  var exampleHtmlString = "<p>Hello World!</p>";

  var exampleObject = {
    "sentence01": "Hello World!",
    "sentence02": "Hello Again!",
    "nestedObject": {
```

```
      "sentence03": "Hello World!",
      "sentence04": "Hello Again!"
    },
    "exampleArray": [{ "value": "foo" },
                     { "value": "bar" },
                     { "value": "baz" }],
    "exampleNestedJS": "var nestedVar = { nestedKey: \"Nested sentence\"}",
    "exampleNestedHTMLinJS": "var nestedHTML = \"<p>Hello world!</p>\""
  };
})();
```

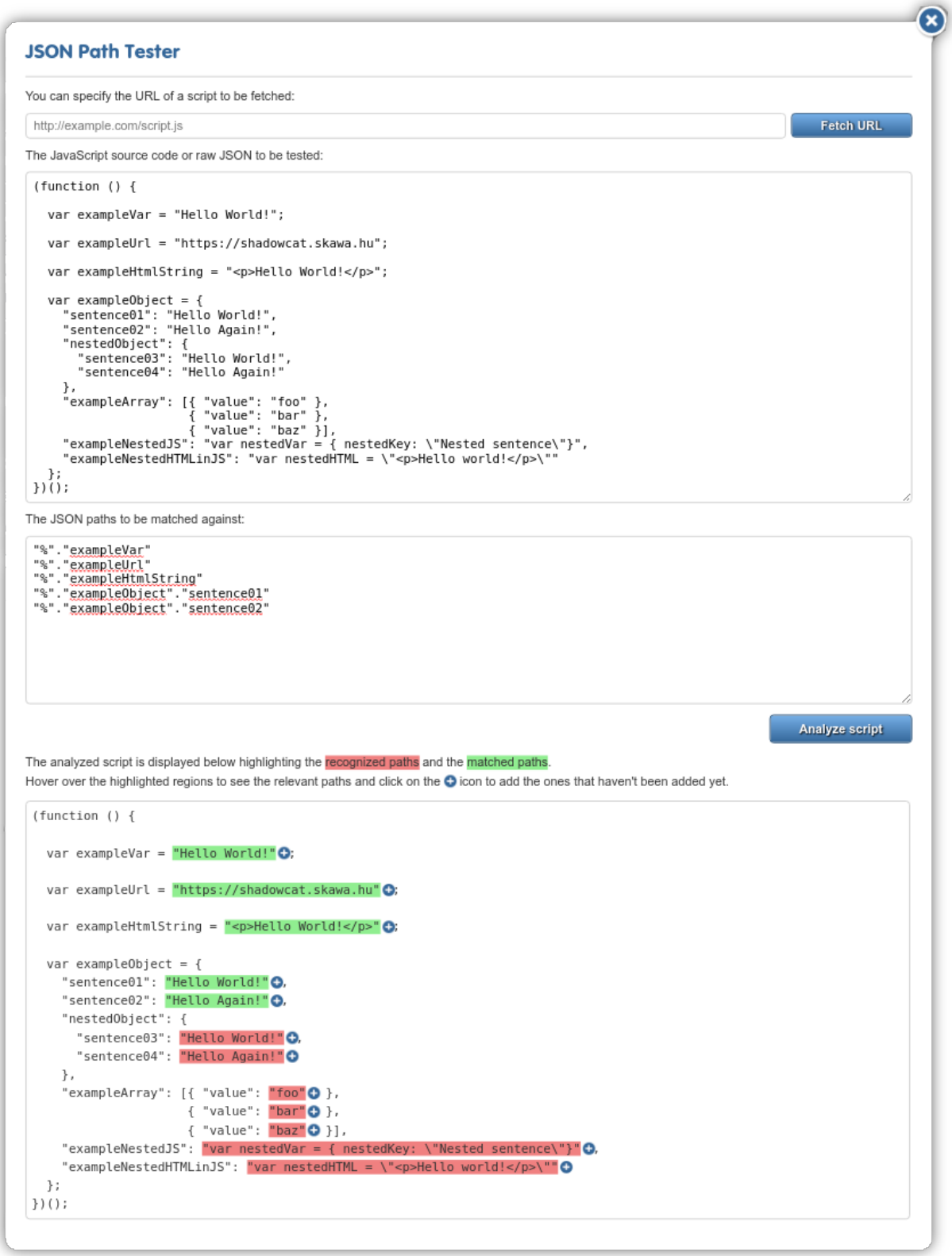
You can copy & paste code into the upper field (or fetch the entire file via the field & button on top if you have the URL) and click on "Analyze script". The file/text will be requested/sent for analysis in the cloud, when it's finished, you should get a highlighted representation of the same code in the dialog.

Click on any of the blue icons to generate a **JS path** for the string in question. If you generate paths for all available strings in the example , the list of paths in the upper text field should look like this:

```
"%"."exampleVar"
"%"."exampleUrl"
"%"."exampleHtmlString"
"%"."exampleObject"."sentence01"
"%"."exampleObject"."sentence02"
"%"."exampleObject"."nestedObject"."sentence03"
"%"."exampleObject"."nestedObject"."sentence04"
"%"."exampleObject"."exampleArray".0."value"
"%"."exampleObject"."exampleArray".1."value"
"%"."exampleObject"."exampleArray".2."value"
"%"."exampleObject"."exampleNestedJS"
"%"."exampleObject"."exampleNestedHTMLinJS"
```

Some of these paths require adjustment before they'll behave correctly.

Supported strings are highlighted in **red**, and those that are already covered by a listed JS path are be highlighted **green**. Your results should look



When you have all the JS paths you need, copy & paste them into the main JS translation text field in Advanced settings. Click on “Save” to apply your changes.

Keys / Variables

Translatable elements are specified by a dot-separated list of words, each optionally double quoted and constituting either a.) a valid JS variable/JSON key name or b.) a token specifying one or more hierarchical levels (anonymous function, array index or globbing mark).

```
var exampleVar = "Hello World!";
```

The simplest possible case would be `"exampleVar"` to mark the value of the top-level element `exampleVar` as translatable. Anonymous function calls are denoted with `"%"`, and since the entire block of variables is wrapped by an anonymous function `(function () { ... })()`, this leading ampersand shows up in each case. Paths for dynamic JSON responses should be prefixed with `"json"`.

Globs

Use an asterisk (or Kleene-star) to collapse a single hierarchical level. E.g., the value of`"exampleArray"` is an array of objects. To include every index in the array, you can roll three rules into one:

```
"%"."exampleObject"."exampleArray".*."value"
```

Double asterisks are even more inclusive: they recursively glob all child nodes. Exact specification can be restarted by following `**` with a double-quoted form. That is, the rule

```
`%`.**."value"
```

marks any variable or property called `value` it finds *at any hierarchical level* within an anonymous function call. If a JS path *ends* with the `**`, then the entire subtree is marked as translatable. Incautious use of this construct is not recommended.

Processing Modes

Nodes are processed as plain text by default, but you can enable specific processing modes with whitespace-separated postfixes. The available processing modes are `url`, `html` and `javascript`.

URL

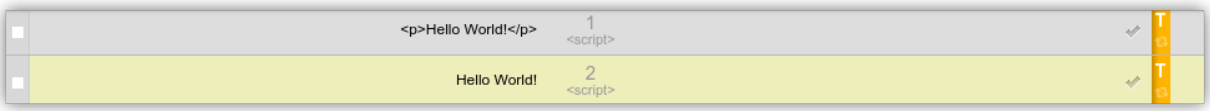
Variables can contain either *the project URL* or some other important location (such as that of a *linked project*) that you would prefer to have **remapped** over the proxy. Don't give in to the temptation to localize URLs in JS as plain text! Instead, use the `url` postfix to map them:

```
"%"."exampleUrl" url
```

HTML

`exampleHtmlString` demonstrates the fact that JS variables frequently hold markup (for better or worse). The `html` postfix lets you process these strings as HTML.

```
"%"."exampleHtmlString" html[@process]
```



The screenshot above demonstrates the difference HTML-processing makes. Picking up HTML-markup explicitly as text is generally considered error-prone and disadvantageous from a localization viewpoint, and isn't recommended.

`[@process]` is optional. By adding it, you instruct the proxy to apply the translation-invariable regular expressions currently set on the project.

Nested Javascript

Although JS paths are mostly specified in a single line, the `javascript` postfix bends this rule. It tells the proxy to apply the rule in the *next line* to the value of the postfixed JSON path. One level of nesting is supported. It is rarely needed, but invaluable when it is called for.

Plain text:

```
"%"."exampleObject"."exampleNestedJS" javascript
"%"."exampleObject"."exampleNestedJS"."nestedVar"."nestedKey"
```

HTML:

```
"%"."exampleObject"."exampleNestedHTMLinJS" javascript
"%"."exampleObject"."exampleNestedHTMLinJS"."nestedHTML" html
```

Note that the JSON Path tester tool is not equipped to display the nested use case.

Xpaths

Xpaths for XML AJAX responses in a similar way as JS/JSON paths do for their respective content types. For example, `/ajax-response/component[1]/text() html` assumes that the first `<component>` node contains translatable HTML markup.

Due to space constraints, we decline to reproduce a full Xpath tutorial in this documentation, and direct the reader's attention to the many tutorial resources available online. The W3Schools summary of Xpath syntax serves as a good starting point.

Limitations

Content extraction

A simple content extraction crawl takes care of JS content in the source of an HTML document. But in many cases, such content is requested **as part of a page load** or **via user action**. The same limitation still applies, however: the page is not available to the crawler in a way that would allow for such "interactive" requests to start.

The solution is to extract content **via Preview**. Open the page in Preview mode and go through the required user actions to trigger all necessary events. The proxy will take care of extracting content from the affected dynamic responses (provided that your setup is correct).

Note the influence of **TM Freeze** on this approach: you need to disable it temporarily for Preview-based content extraction to work.

String Concatenation & Computed Values

JS translation cannot be used with values such as `var string = "You have " + getCartItems().length + " items in your basked"`. In these cases, you either have to forego translation or change the content so that no *computed expression* is present among the concatenated elements.

This implies that those instances of string concatenation where *no* token of the expression is computable are supported. This is indeed so, and provided that the appropriate tweak is enabled in Advanced settings, the proxy can perform string concatenation *upfront* and handle the resulting string as a whole.

Resource Translation

Resources are binary content found on the sites, suchg as images, PDFs, CSS and JS files, etc. Please note that the content of these resources is not extracted for translation, so you have to translate / edit them separately.

Replacing Images with Localized Versions

You have the option to replace images, downloadable files and other resources with their localized version.

1. Please make sure that you see the appropriate target language selected on the left side menu.
2. Navigate to the original image in the Resources view of the Content (or the Discovery) section.
3. Hover over the thumbnail image. A green '+' icon should appear.
4. Click this icon. It marks the image as 'localizable', i.e. a candidate for replacement.
5. Select the replacement image and upload it. It will immediately replace the original image, and the new image will show up on te translated site.
6. Check the Preview to see if the image was replaced properly.
7. Repeat these steps for all images and all target languages that need localized versions.

Please note that you can replace only 1 resource, for 1 target language at one time.

Extraction Issues with `data-image` Attribute

Images might be inside a "data-image" attribute in the source code. In such cases you have to set the "data-image" attribute up to be translated as a link in the Advanced Settings screen. The `<img src>` attributes are recognized by default as URLs, while data-* attributes are not, as they can contain any sort of data, and must be configured manually to tailor the behavior to the current project's needs.

The image might also be drawn from a subdomain. In this case you have to create a second project for the subdomain, mutually link the two projects together, and add the resource in the subdomain's project. In this case you have to publish the two projects together, in order to preserve the mapping.

HubSpot Forms

The proxy supports translation of HubSpot (or similar) forms via a combination of project linking and JS translation.

Method #1 marshals a combination of advanced proxy features. It is entirely **hands-off** from the site maintainer's perspective, no change on the original server is necessary (which is a rather frequent constraint).

Method #2 relies on injected JS and HubSpot to provide separate, localized forms for each target language. Compared to **#1**, it is a clean and simple approach.

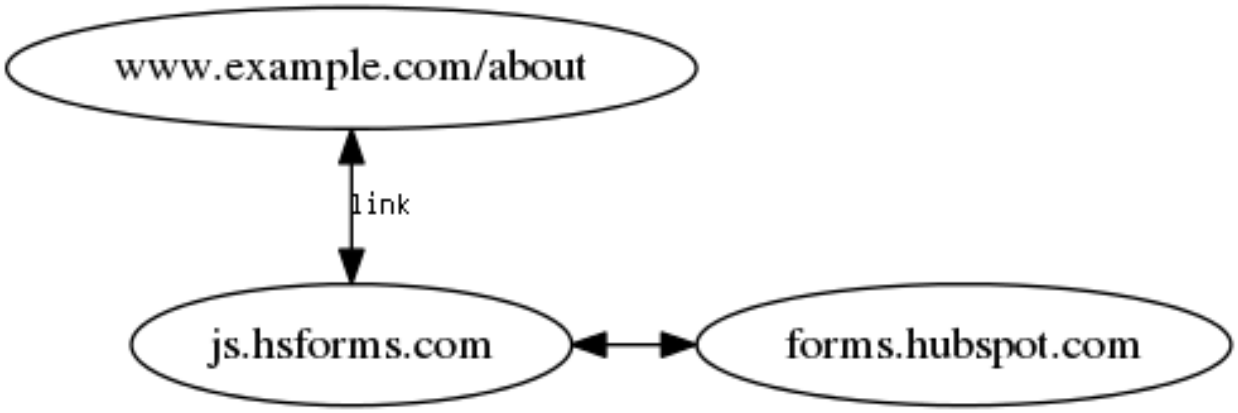
Method #1: Proxy

The proxy approach traces the structure of the main and form domains via linked projects. Affected JS resources/endpoints are overridden and the responses marked as translatable.

Project Creation & Setup

HubSpot uses several external domains to drive a form. You will see https://js.hsforms.net/forms/v2.js referenced in the page source. This file itself references https://forms.hubspot.com, which is where the translatable form contents are coming from.

Domains used by a HubSpot form are related to the main project and each other in the following manner:



Assuming that `example.com` is already set up, at most two additional projects are required:

1. `js.hsforms.com`: creation of this project is *optional*, though not complicated. At the time of writing, visiting the landing page results in a `403 Forbidden` page (but this is not a problem).
2. `forms.hubspot.com`: this URL redirects you to `https://developers.hubspot.com/`. To create a project for it, **disable Redirect checking** in the Add Project dialog. Click on **Advanced settings** to reveal the option and uncheck the checkbox:



Don't forget to add every target language of the main project to each project you create.

Link Projects

Open each project in a separate tab and link each project according to the section on Project Linking. The result should be a chain of projects leading from `example.com` to `forms.hubspot.com` with `js.hsforms.net` as an intermediary.

Alternative: Search & Replace

The `js.hsforms.net` project is not, strictly speaking, necessary. Its true purpose is merely to expose a slightly modified version of the `/forms/v2.js` script. If its URL is referred to in a way that makes it possible, you can *sidestep* the domain using a combination of Search & Replace and a page content override. The setup steps for this are as follows (done on the main project):

1. **create a path override** for the exact URL where the form is present (the diagram above shows `/contact` as an illustration).
2. **add a Search & Replace rule**: replace `https?://js.hsforms.net` (a regex matching both HTTP and HTTPS versions of the same URL) with the empty string. This turns the reference to `/forms/v2.js` into a relative URL, pointing it toward a page content override that is to be created in a moment).

Overriding `v2.js`

This resource contains a crucial variable called `urlRoot`, which has to be **remappable** over the proxy. However, it is set via a *computed expression*, which is unsupported by the proxy for reasons discussed in the section on JS translation, so an override and a small change is unavoidable (regardless of the presence/absence of the intermediate project). Follow the steps below to create the override:

1. visit `https://js.hsforms.net/forms/v2.js` and **copy & paste** the contents of the JS file.
2. use the DevTool or an online pretty printer before pasting the code. Though optional, it is highly recommended that you do this (such minified code is cumbersome to work with as it is).
3. create a PCO for the `/forms/v2.js` pathname in Page modifiers > Content Override. The response code default is 200, and the `Content-Type` header is `application/javascript; charset=utf-8`. We'll return to `Cache-Control` and `Pragma` later, after setup is complete.
4. Add the following line to the top of the PCO:

```
var HUBSPOT_URL_ROOT = "https://forms.hubspot.com";
```

1. Search for `this.urlRoot`. It is set in a line similar to the one below:

```
 o ? this.urlRoot = "https://f.hsforms" + e + ".net/fallback" : null != a ? this.
↪urlRoot = "" + a : this.urlRoot = "https://forms.hubspot" + e + ".com";
```

1. Add the following line after it to make it use the "accessible" value:

```
this.urlRoot = HUBSPOT_URL_ROOT
```

1. Use the *Mark multiple resources as Translatable* text field in Advanced settings. Simply add the pathname prefix of the PCO to the list:

```
/forms/v2.js
```

1. URL Translation & HTTP/HTTPS Finally, add the following JS path to the list of translatable paths in Advanced settings:

```
"HUBSPOT_URL_ROOT" url
```

Open the PCO link over any one of the proxy preview domains to test it. If all projects are correctly linked and you followed the setup steps correctly, the `HUBSPOT_URL_ROOT` variable will hold an appropriate proxy-mapped domain (and consequently `this.urlRoot` will be set to the same value).

Form Contents

Set up the HubSpot content endpoint as translatable on the project for `forms.hubspot.com` according to the JS translation section. In summary:

1. locate the form request using the DevTool and add it to the "Mark multiple resources as translatable" list of prefixes. For any given HubSpot form, translatable content will usually be associated with a prefix similar to the one below (it will also have a `callback` query parameter).

```
/embed/v3/form/{numericId}/{formId}
```

1. use the JSON path tester tool in Advanced settings to process the response. HubSpot forms come in a response format called JSONP or padded JSON (a function call such as `hs_request_0` with the form data passed to it as argument). It is not necessary to prefix JS paths with `"json"` in this case.
2. use the x-proxy to test your JS paths, and use Preview for content extraction.

Publishing & Caching

All projects need to be published together in all target languages. Note that you don't need to publish on a subdomain of the original server: you are free to proxy the German version of `forms.hubspot.com` through `hs-de.mydomain.com`, for example.

Once setup, translation and publishing is complete, you are free to set an appropriate Cache Header on your page content overrides (either on the PCO itself or on a prefix-basis) to reduce page request costs.

Method #2: HubSpot & Injected JS

You can rely on HubSpot to localize the form property names after cloning your form for each target language, and rely on a little JavaScript zo drive the forms on the client-side for each target language. This approach is cleaner than the one described above, as long as you don't mind having a separate form for each target language.

The proxy sets the `lang` attribute of the `<html>` tag to the appropriate locale code on each target language domain, which you can use for branching. The code below demonstrates one example of how such code could look in practice:

```
var lang = document.querySelector("html").getAttribute("lang");

var HSFormId = {
  "en-US": "English9-bb4c-45b4-8e32-21cdeaa3a7f0",
  "fr-FR": "Frenche9-bb4c-45b4-8e32-21cdeaa3a7f0",
  "de-DE": "Germane9-bb4c-45b4-8e32-21cdeaa3a7f0"
};

// refer to the HubSpot documentation for further customization at
// https://developers.hubspot.com/docs/methods/forms/advanced_form_options
hbspt.forms.create({
  portalId: "portalId",
  formId: HubSpotFormId[lang]
})
```

Page requests, CDNs and Caching

The number of page requests sent when a page loads predicts monthly costs on a project. The goal of this page is to provide an in-depth description (and an upfront summary) of the matter and offer suggestions for cost reduction.

Summary

Page requests (in the strict technical sense of an **HTTP request**) are a primary concern over the proxy as they form the basis of monthly costs. Since each HTTP request that hits the proxy is billed, it is useful to consider various ways of reducing the number of hits.

The three methods discussed here, **URL remap prevention**, **use of CDNs** and **public caching** have the potential to dramatically reduce monthly costs (and can be combined according to need).

Page Views vs. HTTP requests

The difference between a *page view* as generally seen and a *page request* as understood as part of pricing is crucial to keep in mind.

Views

Website owners/maintainers usually focus on tracking the number of *page views* (via Google Analytics, for example). This is a useful metric to gain insight about the visitors to a site and make various predictions/business decisions based on user traffic.

Such analytics count *end user visits*, where we generally expect one additional page view to show up in our Google Analytics View each time a user loads a page on the site.

Requests from bots of search engines and request where Javascript is not run (e.g. a JS-disabled browser, `HTTtrack` or command-line tools such as `cURL` and `wget`) do not result in a "visit".

The proxy pricing system **does not** refer to this understanding of a page request.

HTTP Requests

Refreshing the page with a devtool's Network tab openend, or using the Content Breakdown section of www.webpagetest.org for a page shows you that a modern webpage heavily relies on multiple resources (HTML, CSS, JS files, images, fonts etc.) over several domains to construct the unified whole shown to the user. By *page requests*, then, we mean the number of distinct **HTTP requests** for such resources.

It is this sense that the pricing system uses the term. The proxy is a technical solution to process and translate HTTP requests between the visitor and the original site, and any HTTP request that has to be relayed between the user and the server is counted as 1 page request.

This is regardless of the type of content: whether HTML, an XHR/AJAX request or a static resource, it will be counted as a page request if the proxy has to process it.

It is easy to see now why understanding the number of requests going into a page load is important – they act as a multiplier on the number of **page visits** and become an important predictor of monthly project costs.

The ideal case, of course, is 1 page request per 1 user visit (meaning that only the HTML document has to be translated and served). Although this ideal case might not be attainable in all cases, it very often is.

See the next section for a suggested manual approach of evaluating a page load.

In this section, we'll go over the various ways you can *reduce the number of requests* and consequently, project costs.

Possible approaches

Optimization means preventing HTTP requests from reaching the proxy. In practice, there are three general approaches:

1. prevent URL remapping for non-localized resources
2. ensure that "auxiliary" content such as CSS and JS files and images is served from a CDN
3. caching intra-domain resources

Remapping

The proxy billing system is only concerned with those requests that are forced to go through it, so the simplest way of preventing an HTTP request from going through the proxy is to prevent it from being re-mapped from the original site to the translated domain entirely.

If *www.example.com* is being translated into German and published on *de.example.com*, any URLs in the source that point to the original server (that is, are intra-domain) will be **remapped** to refer to the translated domain over the proxy.

But useful exceptions should be made. For example, images that are not localized do not get mapped by default, which means that the image will be downloaded from the original *www.example.com* site instead of the TL domain, naturally preventing an HTTP request from going through the proxy. Altough a tweak exists in Advanced settings to force images through the proxy, you should consider the cost implications of this tweak before turning it on (and look into the subsection on caching to offset increased costs).

The `__ptNoRemap` HTML class is handled specially by the proxy. If this class name is detected, the `href` or `src` attribute of the given element will not be mapped to the proxy (avoiding the request cost).

For example, on a project for `www.example.com`, an achor tag such as

```
<script src="https://www.example.com/client.min.js"></script>
```

would be remapped by default, and it would look like this if published on `de.example.com`:

```
<script src="https://de.example.com/client.min.js"></script>
```

The `__ptNoRemap` class disables this default action, so the script `src` is not remapped even if the page is opened over the proxy:

```
<script src="https://www.example.com/client.min.js" class="__ptNoRemap"></script>
```

Used in a systematic manner, this change has to be applied on the origin server, which is a potential downside if you don't have source/admin access. It bears mentioning that the class is reported to have solved one-off problems by being search & replaced into targeted spots in the page source.

CDNs

The simplest way to prevent an HTTP request to the project domain is to offload it from the original server, too. Content Delivery Networks are servers that are capable of providing source (non-localized) content in a reliable way across the globe.

Public Caching

(*Note: not to be confused with Source and Target caches!*)

Many static resources need no processing whatsoever by the translation proxy, and become cost overhead if funneled through it. But it is also often the case that it impossible to avoid having these request go through. What to do in these cases? Enter public caching.

HTTP supports what is called a `Cache-Control` header that can be added by the server to a response. The content of this header instructs public caching nodes on the network on how to store a static copy of the resource for a time.

A cache (usually affecting a given geographical area or specific network pathway) will serve the resource to visitors until the "Time-To-Live" of the cached resource expires. This TTL is defined by the value of `max-age` in the `Cache-Control` header, and its current value is tracked in the `Age` header.

Until such time as this time is up, requests coming in from a place served by the cache will not, in fact, reach the proxy. After `max-age` expires, the cache will re-request the original for another `max-age` term - during that time, however, a cache can serve a multitude of requests without having to burden the original server (& proxy for it).

Declaring a `max-age` of 86,400 on the image `/about-us/logo.jpg`, for example, broadcasts on the network that for the duration of one day, any public caching node should feel free to cache the resource – it's up to them, but if possible, they should not re-request it until then.

This way, the caching/serving/trafficking burden evens out on the network, and many of the repeating requests can be avoided (i.e. an intently browsing user might load the same resource over and over again, but most of those requests will be served from a cache).

Keep the following important points in mind:

1. Caching naturally introduces **update delays**. A user getting cached content will have to wait for it expire before seeing a new version.
2. Frequently updated HTML pages (such as those with a newsstream) should not be targeted for caching, and URLs serving **dynamic resources should never be cached!**
3. Public caching works best with **static resources** (JS, CSS and images) and **versioned URLs**.
4. It is **impossible** to reach a cached resource coming in from the server side. Neither the original site, nor the proxy can tell a node to throw a cached version away. Only expiry will do that. This is architectural, the HTTP protocol does not provide a method to send cache invalidation notifications to arbitrary nodes. The upshot is that while it might seem sensible (from a cost-reduction perspective) to add as huge `max-age` values as possible, this is highly recommended **against**. Unless the URL of the given resource is versioned to allow for updates anytime, users may end up "walled out" by a cache storing an image for weeks, for example.

A measure of carefulness is advised!

Cache-Header Setting

If you find that very sensible defaults are coming in from the original server, that is very good news, but it is also a fact that such is not always the case.

To adjust `Cache-Control` on the proxy-side, go to Dashboard > Path settings to override headers on a URL or prefix-basis. See the Path settings documentation to learn more about overriding/fine-tuning Cache-Control headers.

References

See HTTP caching in Google's Web Fundamentals on public caching.

See RFC 7234 for the full technical detail on HTTP `Cache-Control`.

Page Request Evaluation

In this recipe, we go over one possible method of evaluating a page for the number of *HTTP requests* that it sends to the project domain when it loads.

Opening the DevTool

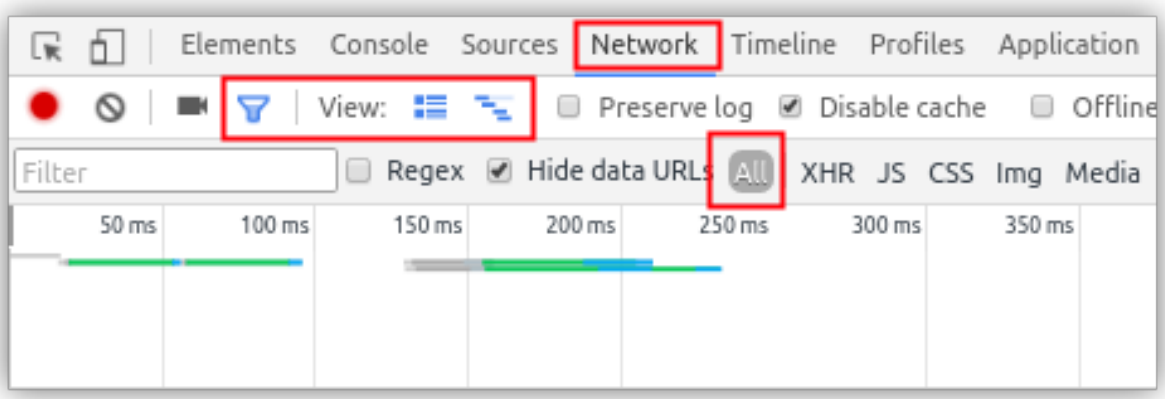
Investigation of the number of requests can be done on the original site. With the site open and selected in a tab, press **F12** to open the DevTool for that specific URL. By default, the DevTool will be docked within the browser screen, but you can also unlock it into a separate window.

```
1. Click on the Network tab.

2. Select "All" to track all requests.

3. Enable "Use large request rows" (to the right of "View" in the toolbar).

4. You can leave the "Hide data URLs" option on
```



At this point, since you have opened the DevTool after the site has already loaded, the Network tab will be completely empty.

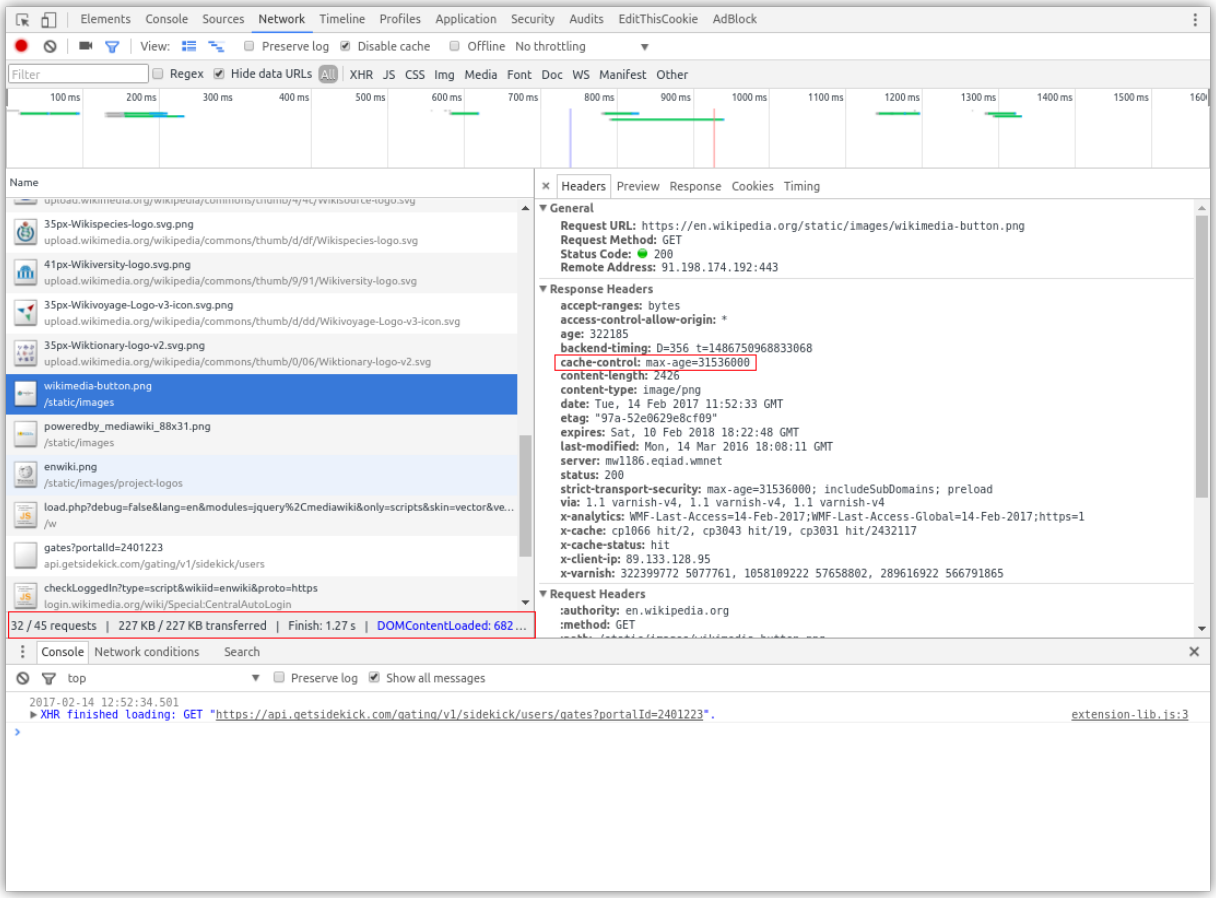
```
5. Refresh the site to start logging requests.
```

You will see a flurry of activity as the site reloads. After it has finished, you can start analyzing the various requests in the list.

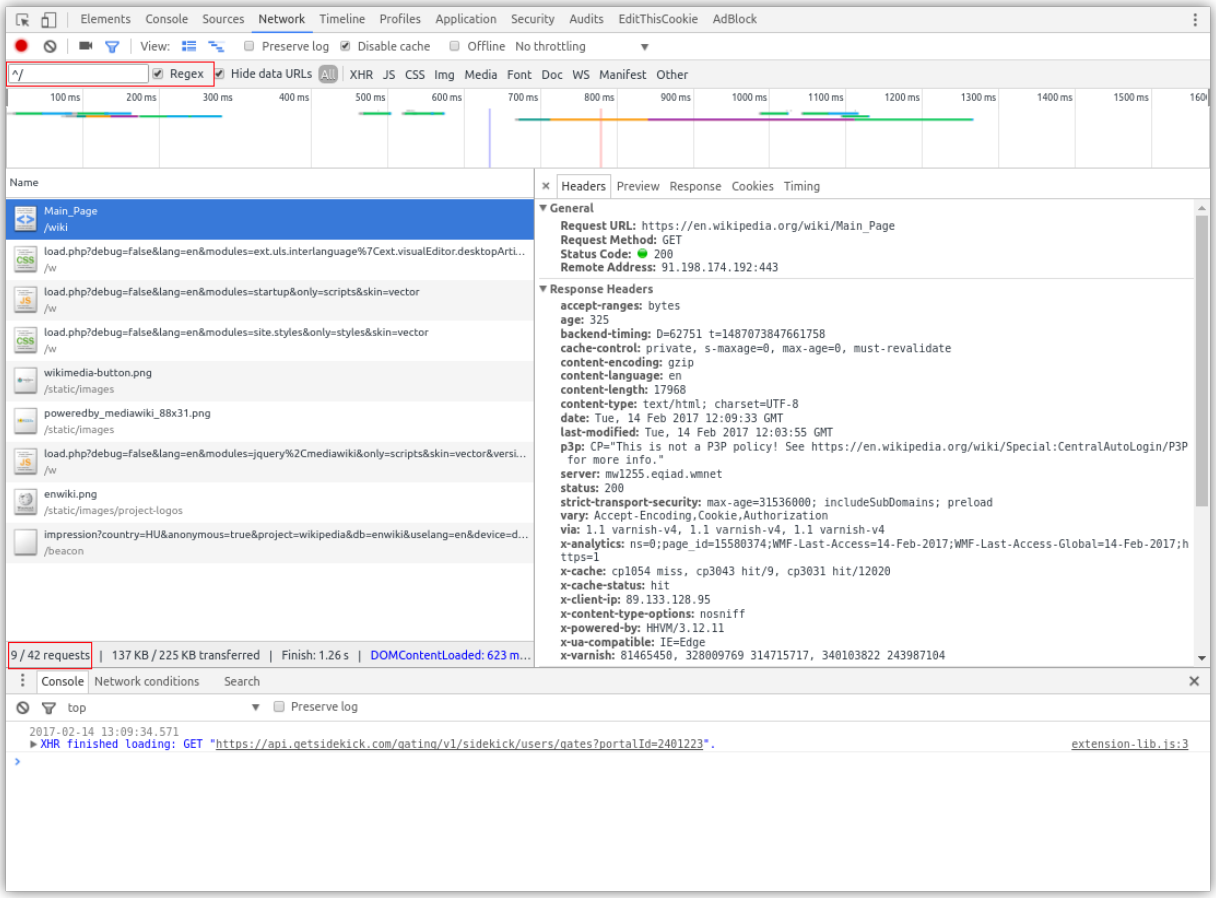
Filter Requests on the same Domain

```
6. Enable "Regex" for filtering
```

You are free to ignore all requests that go to external domains that will not be part of any linked projects. If you have enabled the “large request rows” option, the DevTool will helpfully list the paths below the resource names.



The DevTool provides detailed information, but not all of it is needed in this case. As soon as you have the full list in view, you can remove from view those requests that go to external domains. The easiest way to do this is to add the "^/" regex to the filtering options on top.



At this point, the first relevant statistic becomes available at the bottom of the request list - the number of currently displayed requests / total number of requests will be a good first indication of the number of requests that *potentially* have to go through the proxy when the page is served.

Considerations

What Cache Headers are present?

If you click on a request in the list, a new sidebar will appear with information about that particular request. For evaluation purposes, the `Header` tab, and the `cache-control` directive is especially important.

`cache-control: private` or `cache-control: no-cache` indicate a request that the original server expressly states to be non-cacheable - usually, it is not a good idea to change this haphazardly. It is better to count these requests as necessary for the construction of the page.

Is the resource static?

PNG/JPG images, JS and CSS files are those resources that tend not to change rapidly. You can override the Cache Headers of such Resources in Dashboard > Path settings - with the effective result being that the burden of serving that content is offloaded to independent caching nodes on the global network.

Re-caching happens for each cached entity after the duration of the `max-age` directive passes. Using `max-age`, you declare a time-frame during which you will enlist the help of the network to serve the content in unchanged form. It

can be used to fine-tune the time that you'll allow a specific cached instance of a resource to persist.

NOTE While Cache Header overrides work in the overwhelming majority of cases, there is no "law" to force caching nodes to respect them: consequently, the pace at which various Resources are cached/re-cached on the global network is, to a degree, arbitrary.

While technically possible, making Document resources cacheable requires careful consideration, but as little as 10 to 60 minutes can be very useful. Consider that in most cases, the landing page receives the most page requests. Consequently, allowing it to be cached with a controlled `max-age` value means considerable savings on the proxy (with the caveat that any changes will take at most the time declared for `max-age` to propagate across the network).

Does the requested resource change often/constantly according to context?

Dynamic Resource

XHRs/AJAX calls/dynamic content cannot be cached without rapidly running into many problems on the published site. It is better to say that they simply can not be cached. This also applies to those requests that are sent throughout the user session on the page after loading it (and in lieu of hard data, it is very difficult to forecast the number of dynamic requests a given user will start).

Salient examples are search field handler scripts, web-shop endpoints, PHP scripts, backend endpoints and other similar sources that give wildly varying responses based on the parameters sent in the requests.

Frequently Iterated Resource

If a site is undergoing development, for example, it is usually not a good idea to add Cache Header overrides (certainly not overly long ones). This would in effect delay propagation of any changes by the value of `max-age`, resulting in syncing problems between the original site and its translated counterparts.

Overriding Cache Headers

Evaluation of the number of requests is most useful when estimating the monthly cost associated with serving a site. For overriding cache headers on the Dashboard, see the Path Settings section of this documentation. Enable the public caching tweak in Dashboard > Advanced settings to facilitate further speedups on the Easyling end.

Example Scenario & Conclusions

If the following information is available:

```
* The original site has 50,000 monthly user visits
* A single target language sub-domain is expected to receive about half of that, 25,
↪000
* Each page uses between 50-70 requests to build
```

Using the consideration points above, it is determined that most of those requests can be counted out and the rest cached for 24 hours. The site will require 3 non-cacheable requests from a user coming in from a location where there is no caching node on the way.

From this you will be able to conclude the following:

```
* 2 requests are necessary, so 25,000 * 3 = 75.000 expected page requests
* BUT: consider also the number and visit frequency of revisitors - each time a␣
↪cached entity's max-age (24 hours in this hypothetical scenario) expires, that␣
↪resource has to be re-cached, which will increase the number of requests going␣
↪through the proxy with a certain amount (although this amount is usually␣
↪negligible).
```

With the appropriate Cache Headers, Google's geographically specific Edge Cache, the public internet, the various ISP caching nodes, and at the other end of the process, the user's browser cache will participate in offloading the page request from the proxy's translation pipeline.

Conclusion

Armed with this knowledge of an overarching view of requests that will have go through the proxy, you will be able to provide accurate estimates for the monthly costs of the proxy

Staging Domains

Use Staging Domains to change the origin server to use in Preview or crawls.

Website maintainers like to test any changes they make on a development or staging server before unleashing them on the Live site. The same staging server that is in place on the site can be used over the proxy as a testing ground for any translatable updates. Add a Staging Domain option to extract and use data from that domain staging server in the various proxy modes.

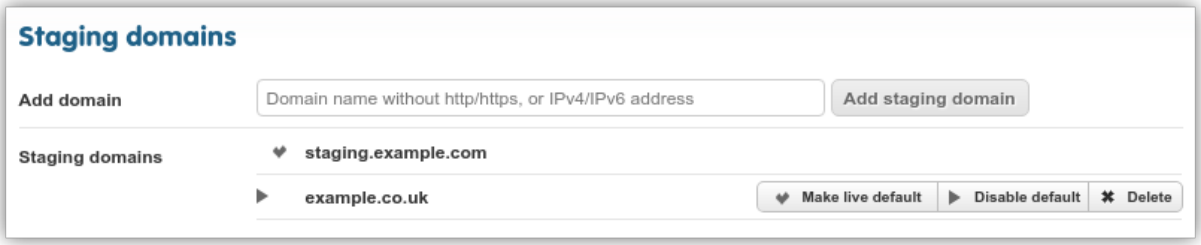
If the project URL is `example.com`, and a staging server exists at `dev.example.com`, you can enter that URL into the Staging Domain field and click on "Add Staging Domain". The domain is added to the menu below the text field automatically and enabled as default.

All requests going through the proxy (regardless of initiator, such as a user session in Preview or a content extraction Scan) will be mapped to that domain, regardless of the original project URL - the project domain might be `example.com`, but the Preview will be displaying content from `dev.example.com`.

Very important: Translations are propagated after extraction as usual, but this comes with an important warning: *previous translations* will only show up appropriately if the path structures of the original site and the staging server match 100%. If this is not the case, then a new project page is created in the project.

Default and Live Default Staging domains

Hover over the staging domain name to reveal three options. *Live default*, *Default* and *Delete*.



Click on "Make Live default" to enable the staging domain for the live published site. A tick icon will be displayed next to the staging domain when this option is enabled.

The same goes for *Default*, which applies to all other proxy modes (such as the Highlight View in the Workbench, the Preview domain or the X-proxy). Click on *Delete* to remove a Staging Domain.

Naturally, only one staging domain can be designated as Live or Preview default at one time, but it is possible to enable one domain to be both.

When to Use Staging Domains?

Content Decoupling

The staging domain feature of the proxy is beneficial when updates to the original site are regularly tested on the staging server first.

Staging domains let you make that same content available through the proxy for translation work without disturbing the translation quality over the published domain in the meantime.

The idea is to extract content from the staging domain and begin translation work early – by the time the changes on the original site are moved from the staging domain to the main one, all target language entries will have their translations ready.

Domain Name Changes

An alternative use of the staging domain feature is when the original site changes domains. As you know, project addresses cannot be changed once a project is created. However, migrating an entire project due to a simple name change might not be a very optimal solution.

In these cases, you can use the Staging Domain option to set up the new address as a staging server on the project. By following up on the name change in the Publishing settings, you can transform both the origin and the published domains to the new address (but note that the project address will remain the same, however).

Left-Right conversion

It depends on the site...

1. Best case scenario:

```
html {
    direction:rtl;
}
```

```
<html dir="rtl">
```

If it looks mostly OK, then all is left just some minor CSS fixes:

- flipping images (in carousel / slider)
- list elements' bullet is defined using a background image
- If text's align is defined explicitly (e.g. using WP's text editor) inline, then each and every element must be overridden using `!important`

1. Bootstrap and framework alike:

- some framework might provide a dedicated RTL css file bootstrap-rtl.css

1. If this is not case, each and every element must be positioned individually

Mixed content within text

- When it comes to actually render the text (numbers), the direction is determined by a couple of rules. Please read this for the details.

- ***As a rule of thumb: during the translation (from LTR language to RTL), don't use your CAT tool to change the order of the numbers and text where the translation is with latin characters. Phone numbers like 1-800-123-1234 should be left in this order.***
- To make sure numbers are rendered properly at the end, a Left-to-Right (LRM) must be inserted before every number. The dash between numbers split them, so LRM must be inserted after them again. Click to read more on how to insert these LRM characters.
- The same holds for parenthesis, etc. Make sure you understand the rules; sometime LRM must be inserted *after* the closing parenthesis

```
<p style="direction: rtl;">
<span>(TTY/TDD) 711 </span>
</p>
```

```
<p style="direction: rtl;">
<span>&lrm;(TTY/TDD) 711</span>
</p>
```

```
<p style="direction: rtl;">
<span>&#8234;(TTY/TDD) 711&#8236;</span>
</p>
```

- As an alternative, LRE character can be inserted *before* the sequence, terminated by a PDF mark.
- To edit the XML (XLIFF) directly, use Sublime, available on Windows, OS X and Linux as well. It works very well with Regular Expressions. It's a robust way to insert these marks using their unicode symbol, such as `‎`

Further readings:

- http://dotancohen.com/howto/rtl_right_to_left.html

Site Search

Transparent multilingual search is a frequently required (though often belatedly acknowledged) part of website translation. We recommend a client-side approach to search (the approach to which the proxy translation model lends itself best), a comparatively involved use case of code injection. Given its pervasiveness and importance, it warrants detailed treatment in our documentation.

Part I of this recipe provides a general description of the site search issue and an introduction to the recommended solution. **Part II** contains a simple example implementation.

This latter section also serves as an in-depth **page modifier tutorial** as well, and as such, it assumes basic familiarity with core web technologies such as HTML, Javascript and CSS.

Part I - General

Issue

Visitors expect to be able to search for things in the language of the website – to input any term into the search fields and receive appropriate results immediately.

Let's say that a visitor uses the search field of `www.example.com` to search for the word "**product**". Upon pressing enter, they are navigated to `https://www.example.com/search?q=product`. The server will detect and

use the value of the `q` parameter and runs whatever server-side search mechanism it uses to search for this term, assemble a list of results and construct a result page to send to the browser.

On a proxied site, however, we run up against a problem: the original does not know about the translations. Requests are automatically relayed to the original server. If a visitor were to type "**produkt**" on the German domain (resulting in an URL navigation to `https://de.example.com/search?q=produkt`), that is the same as relaying a search query with German language content to the original site.

Not possessing a *German language index*, the response is certain to contain 0 results. The fact that the proxy is CMS-agnostic and that it generally doesn't require that translated content be shared with the original server also means that this same content will not be available to the indexing/search software that is running on the original.

Recommended Solution

The way out of this conundrum is that proxied pages themselves are *publicly available for indexing by search engine bots*. A search engine that supports site-specific queries can provide localized search via client-side AJAX requests.

There are two aspects to this kind of solution that you should consider:

On the one hand, such a site search has to be coded in JavaScript in the form of an override (an example with a detailed explanation follows below), and depending on how ornate/feature-laden the original's search functionality is, complexity of the override implementation can vary from the relatively simple to the astonishingly complex.

On the other hand, familiarity with the various indexing-related conditions of your chosen vendor is also important (we provide pointers to documentation for Bing, since it is the vendor used in the example). In our experience, it is not at all unusual for search engines to be leisurely in their pace. Remember that a site can only be indexed after it is published over the proxy.

On the same note, the publishing method you use can also have a bearing on the way search & indexing will work, so a case-by-case analysis is a must. Note that the Client-Side Translation publishing method is *not compatible* with a site search integration of the type described here.

Third-party Integrations

Bing

Bing's Web Search API can be used for site search integration purposes over the proxy. In order to access Bing's API, you will need to purchase an API key. See the details on Microsoft Azure on procuring one, and see the pricing page for a detailed description of the various service tiers.

Consult the Webmaster Tools Documentation concerning general usage-related matters. You can submit a domain for indexing in Bing here. Submission of specific URLs is possible here, but this feature is limited to root domains at the time of writing: until such time as this limitation is lifted, you might only be able to use this targeted indexing feature with subdirectory publishing.

Bing: Verification of subdomains

If you want to track the indexing of your subdomains in Webmaster Tools and do SEO tracking, you'll need to verify ownership of the target language subdomains. The available methods are described here.

The XML-based approach is the simplest to implement over the proxy (it can be done without having to apply any changes on the origin): create a temporary Page Content Override with the contents of `BingSiteAuth.xml` (content type should be `text/xml`). This exposes the authorization XML over the proxy domains. From then on, you are free to add the target language subdomains in Webmaster Tools and verify them one-by-one.

Part II: Example

In the rest of this documentation page, let us detail how a simple site search integration could be implemented as injected JavaScript over the proxy using v7 of the Microsoft Web Search API.

Page Modifiers

Customized JavaScript can be injected into the `<head>` tag of each page over the proxy. This capability is the entryway for a search override. To add injected JavaScript you'll find an editor in Page Modifiers > Javascript editor, where you can type or copy & paste Javascript code. After saving the modifier, it will show up in the page source over the proxy after a refresh (but note that cache settings might get in the way of an instantaneous update on the live domain!).

Site search functionality is frequently displayed to the visitor in the form of an input field and a button. We'll go with this familiar scenario and use the following minimal webpage for this tutorial:

```
<html>
  <head>
    <script src="https://code.jquery.com/jquery-2.2.4.min.js"></script>
  </head>
  <body>
    <script>
     function showResults(){
       document.querySelector("#results")
               .setAttribute("style", "display:block;");
     }
    </script>
    <div id="search">
      <form id="form" action="javascript:showResults();">
        <label for="#input">Search:</label>
        <input id="input" type="text"/>
        <button id="button" action="submit">Submit</button>
      </form>
      <div id="results" style="display:none;">
        <div class="result-item">
          <h2><a href="#">Result title</a></h2>
          <p>Result summary</p>
        </div>
      </div>
    </div>
  </body>
</html>
```

First steps

`showResults()` simulates the original search functionality of a site. The goal of our search integration is to prevent the original from running and replace it with a client-side dynamic request. A site search integration is no different from any other client-side code, and the skeleton that we described previously on the Page Modifiers documentation page is a safe starting point:

```
(function (){

  "use strict";

  $(document).ready(overrideSearch);
```

```
  function overrideSearch () {
    ...
  }

})();
```

We use jQuery's `$(document).ready()` to wait until the DOM is ready to be manipulated. The IIFE wrapper isolates our modifier from the global namespace, and we never leave the house without `use strict`.

We could define a callback on `document.onreadystatechange`, but this approach suffers from potential problems: we'd be setting our modifier on a publicly accessible property of `document`, exposing us to the possibility that some other script might inadvertently redefine it (or we could be the ones doing the redefining).

We would also end up having to pool different page modifiers in one function definition, which goes against the principle of separation of concerns. For these and similar reasons, vanilla JavaScript's `document.addEventListeners` or jQuery are a better fit.

With basic scaffolding in place, we move on to the nitty-gritty of search. The integration needs to do three things:

1. override search elements on the page
2. send search request based on user input and handle response
3. build the results page based on the response

We take each of these responsibilities in turn to see how they can be implemented.

Overriding Elements

Having prepared this basic scaffolding, we consider the contents of `overrideSearch()`. We would like to attach our event handlers to all search-related elements:

```
function overrideSearch () {
  $("#form").on("keydown", overrideInput);
  $("#button").on("click", overrideClick);
}
```

Next, we implement the callbacks. For the input field, we ensure that the user is not disturbed by the modifier needlessly, which only steps in if the Return key is pressed:

```
function overrideInput () {
  if (event.keyCode === 13) {
    event.preventDefault();
    sendRequest($("#input").val(), renderResult);
  }
}
```

The button is straightforward to override. The use of `this.previousSibling` is featured here as a suggested alternative to selectors.

```
function overrideClick(){
  event.preventDefault();
  sendRequest(this.previousSibling.value, renderResult);
}
```

We are done with the override part. The error message `Uncaught ReferenceError: sendRequest is not defined` should appear in the console if we try to search at this point. The default event might have been

a navigation or a dynamic request of the original site's own, but we prevent it from executing and call the function `sendRequest` instead, which we'll implement next.

Sending the Request

We assemble a `GET` request using the search term passed to `sendRequest` and send it to the Bing endpoint. For example:

```
function sendRequest (term, callback) {
  $.ajax(createRequest(term)).success(function (resp) {
    callback(resp);
  })
}
```

`createRequest` is an important part of this process, where various API-related parameters to construct and return the request are sent. Using Bing, it could look something like this:

```
function createRequest (term, offset) {
  return {
    beforeSend: function (xhr) {
      xhr.setRequestHeader("Ocp-Apim-Subscription-Key", config.API_KEY);
    },
    error: function (xhr, error, thrown) {
      console.log("Error during request!");
    }
    url: config.API_URL,
    type: "get",
    data: {
      q: "site:" + config.DOMAIN + "/ " + term,
      count: 10,
      offset: offset || 1
    }
  };
};
```

`beforeSend` sets the **Ocp-Apim-Subscription-Key** header, which contains the API key to authenticate the request. We introduce Bing's **site-specific search** feature into the request by prefixing the value of the `q` property with "site:"(which works the same way as on bing.com).

It is also at this point that the code should handle the publishing method you use. If you publish your project in a subdirectory of the original domain, then in addition to the "site:" prefix, you would also have to add the target-language directory prefix (e.g "/ja/" or "/de/"). If you are publishing in multiple languages, you can use the `lang` attribute of the `html` tag (the value of which is always the current locale) to make this part of your code target-language specific.

A variable called `config` was also introduced to hold search-related config parameters in one place. This variable can go to the top of the modifier:

```
var settings = {
  API_KEY: "nmtcxylkj56lkjmnnj3mg782nmvf23gz",   // example only!
  API_URL: "https://api.cognitive.microsoft.com/bing/v7.0/search",
  DOMAIN: location.host
}
```

If the API key is valid and version-compatible with the endpoint, Bing responds with a JSON to each request, details of which can be found here. The `webPages.value` property of this response is particularly important, a JSON array containing the first batch of search results.

Note that `settings.DOMAIN` is set to `location.host`. Encoded in this fact is the assumption that `location.host` (the site in which the code is running) is to be used for searches and it is already indexed.

During development and testing, however, no proper search result might be available yet. For dev purposes, it can be useful to change `DOMAIN` to e.g `en.wikipedia.org` temporarily to receive proper search results. The use of a mock responses is also an option.

Search engine APIs have many features that you can provide in your implementation beyond this example, but we will not discuss all the possibilities in this tutorial.

We pass the response to the final component of our override, the DOM handler, `renderResult`. We already mentioned it in passing to the `sendRequest` calls.

Building the Results

Astute readers will notice that there is hardly any site-specific information in the code above (practically none, except the element selectors). We'd be inclined to think that plugging a `form` and `button` selector into the functions and changing the API_KEY will get us on our way on any similar site – and to the extent of element overrides and request handling, that might very well be the case. `renderResult`, however, does not enjoy the benefit of generality.

The DOM handler is responsible for *displaying* search results on the proxied site in a way that seamlessly integrates with the original appearance, making this part of a site search integration heavily dependent on the specific context of the website that we are dealing with.

In order to work out the imlementation for our example webpage, we turn our attention from `#search` to `#results` and research the structure of a search result item.

In the same breath, we can add the basic structure of `renderResult` right away. The bare minimum functionality is to clear the result list and add the one that was received and passed as argument.

```
function renderResult(response) {
  $("#results").empty();
  if (typeof response.webPages.value !== "undefined") {
    for (var hit in data.webPages.value)
      $("#results").append(createResultItem(data.webPages.value[r]))
  }
}
```

`createResultItem` constructs an element for a result item using information provided by Bing, but copying the element structure from the original, which has a huge benefit: existing CSS styles will apply to the search integrated results automatically. Inspecting the search results produced by the original server, we learn that a search result item looks like this:

```
<div class="result-item">
  <h2><a href="#">Result title</a></h2>
  <p>Result summary</p>
</div>
```

Knowing this, we can implement `createResultItem` in the following simple fashion:

```
function createResultItem (result) {
  return $("<div>").attr("class", "result-item")
    .append($("<h2>")
      .append($("<a>").attr("href", result.url).text(result.name)))
    .append($("<p>").text(result.snippet))
};
```

Among the available options for creating DOM elements (`document.createElement`, string contatenation, etc.), the jQuery-based approach is clear and concise. Many bells and whistles can be added to a DOM handler, such as a pager, for example, which would require that we handle offsets (only hinted at by `createRequest` in this tutorial) and add Previous/Next buttons, etc.

Our tutorial, however, does not extend to those details and we conclude it here.

Notes

Some frequent, but non-essential complexity of client-side search is omitted from our example:

- offsets (a Bing-specific term) are only alluded to. Any search API will support requests for the next batch of search results for the same search query. This is usually exposed to the end user in the form of a pager that, when clicked, sends the same request but increments the `offset` by 10. Such a feature causes both `renderResult` to manage the concept of a pager and `sendRequest`/`createRequest` to have to keep tabs on tne current `offset`.
- a search field is also generally available on all pages on a site, not just the search results page. This means that an integration needs to ensure that the user is redirected to the search results page and that query parameters are appropriately handled. Besides using the values of the input field (a user-driven query after page load), a search integration usually has to be able to extract `produkt` from `https://de.example.com/search?q=produkt` to start a default search on the search results page when it loads for the first time. This is usually not difficult, but as we've said previously, much depends on specific circumstance.
- a search integration also has to be prepared to display a "No results." search result page if the search engine returns no hits. Such natural language has to exposed in a way that can be translated using the proxy.

Code

We repeat the code from the discussion above in its entirety. In summary, when injected into the example webpage, it will override both the search field and the button to request search results from Bing via an AJAX call and then displays those search results in-place.

```
(function (){

  "use strict";

  var config = {
    API_KEY: "nmtcxylkj56lkjmnnj3mg782nmvf23gz",    // example only!
    API_URL: "https://api.cognitive.microsoft.com/bing/v7.0/search",
    DOMAIN: location.host
  };

  $(document).ready(overrideSearch);

  // OVERRIDE
  function overrideSearch () {
    $("#form").on("keydown", overrideInput);
    $("#button").on("click", overrideInput);
  };

  function overrideInput () {
    if (event.keyCode === 13) {
      event.preventDefault();
      sendRequest($("#input").val(), renderResult);
```

```
    }
  };

  function overrideClick(){
    event.preventDefault();
    sendRequest(this.previousSibling.value, renderResult);
  };

  // REQUEST
  function sendRequest (term, callback) {
    $.ajax(createRequest(term)).success(function (resp) {
      callback(resp);
    })
  };

  function createRequest (term, offset) {
    return {
      beforeSend: function (xhr) {
        xhr.setRequestHeader("Ocp-Apim-Subscription-Key", config.API_KEY);
      },
      error: function (xhr, error, thrown) {
        console.log("Error during request!");
      },
      url: config.API_URL,
      type: "get",
      data: {
        q: "site:" + config.DOMAIN + "/ " + term,
        count: 10,
        offset: offset || 1
      }
    };
  };

  // DOM
  function renderResult(response) {
    $("#results").show().empty();
    if (typeof response.webPages.value !== "undefined") {
      for (var hit in response.webPages.value)
        $("#results").append(createResultItem(response.webPages.value[hit]))
    }
  };

  function createResultItem (result) {
    return $("<div>").attr("class", "result-item")
      .append($("<h2>")
              .append($("<a>").attr("href", result.url).text(result.name)))
      .append($("<p>").text(result.snippet))
  };

})();
```

SSL Certificates

Easyling has the ability to proxy HTTPS pages, but to do so, it must be provided a certificate and private key matching the URL. Otherwise, the proxy will be unable to identify itself as a valid server, and the browser will abort the connection for security reasons.

Easyling support can assist in deploying an HTTPS site by providing a CSR (Certificate Signing Request) to generate the appropriate certificate if the required information is provided, or the client can prepare the certificate themselves.

Additionally, a certificate is required to provide a branded Easyling instance. For more information, you can check the Whitelabel article of the FAQ.

The protocol

HTTPS (also called HTTP over TLS, HTTP over SSL, and HTTP Secure) is a protocol for secure communication over a computer network which is widely used on the Internet. HTTPS consists of communication over Hypertext Transfer Protocol (HTTP) within a connection encrypted by Transport Layer Security or its predecessor, Secure Sockets Layer. The main motivation for HTTPS is authentication of the visited website and protection of the privacy and integrity of the exchanged data.

In its popular deployment on the internet, HTTPS provides authentication of the website and associated web server with which one is communicating, which protects against man-in-the-middle attacks. Additionally, it provides bidirectional encryption of communications between a client and server, which protects against eavesdropping and tampering with and/or forging the contents of the communication. In practice, this provides a reasonable guarantee that one is communicating with precisely the website that one intended to communicate with (as opposed to an impostor), as well as ensuring that the contents of communications between the user and site cannot be read or forged by any third party.

Historically, HTTPS connections were primarily used for payment transactions on the World Wide Web, e-mail and for sensitive transactions in corporate information systems. In the late 2000s and early 2010s, HTTPS began to see widespread use for protecting page authenticity on all types of websites, securing accounts and keeping user communications, identity and web browsing private. (Courtesy of Wikipedia)

Issuing an SSL certificate

Issuing universally accepted certificates is restricted to the so-called “Root Certificate Authorities”. However, root authorities will generally delegate their powers to “Intermediate Authorities”, who will sign certificates as requested by the end user (provided ownership of the domain can be verified). When requesting a certificate for your white-label installation or HTTPS site, you will most likely interact with an intermediate authority, by providing them an encrypted configuration file (the Certificate Signing Request), which the provider consumes to produce a cryptographically signed certificate.

Using a CSR has several distinct advantages over generating your own certificate at the issuer: since the the translation proxy support crew is in control of the final product, we can tailor the request to generate the certificate we need from you; and since the private key remains safe with us, you do not need to take special precautions when sending it. However, we require some information to be provided to us beforehand:

1. `countryName_default = ${COUNTRY}`
2. `localityName_default = ${CITY}`
3. `streetAddress_default = ${ADDRESS}`
4. `postalCode_default = ${ZIP}`
5. `0.organizationName_default = ${COMPANY_NAME}`
6. `organizationalUnitName_default = ${ORG_UNIT}`

Issuing a certificate consists of several predetermined steps. First, a cryptographic key pair is generated, one half public, the other private. Then, a file indicating the domain the certificate is to be issued for, as well as various data about the entity holding the domain (generally, you or your client) is created. This file is then combined with the public half of the key, and signed with the private half. The resulting encrypted file is then handed to the issuer, who verifies the information contained within, and if successful, encodes the information into a public certificate, signing it with its own private key. This is the file that needs to be provided to Easyling support.

Providing us with a certificate

Following the previous phase, you are now in possession of a cryptographic certificate, and possibly its private key (if you elected to create your own certificate instead of requesting a CSR from us).

If we provided you with a CSR file, you need to send only the certificate. On its own, the certificate is not viable - it requires the private key to be useful, thus, it can be sent via email safely. The private key remains safe with us, and we will use it to upload the certificate to Google AppEngine, from where it will be available to authenticate the proxied site to the browser, and enable HTTPS for the translations.

If you elected to forgo the CSR and generate your own certificate, we will need the associated private key as well (and any passwords used to lock the private key). In this case, however, care must be taken to prevent the certificate falling into the wrong hands: the certificate and its keyfile, or the keyfile and its password must never travel together - if the email carrying both is intercepted, the malicious third party can use it to impersonate your site! Either use two separate emails, or even two different channels (email and Skype, for instance) to provide us the key and password and the certificate. Once we receive the files, we will again upload them to AppEngine after decrypting the keyfile, at which point it will be available for use with the proxied site.

N.B.: When selecting an SSL provider, bear in mind that Google only accepts certificates that are either self-signed or signed by a publicly trusted root Certificate Authority. One important point that needs to be highlighted is that the root CA for CloudFlare's Origin certificate system is *not* publicly trusted. Thus, we are unable to make use of Origin certificates generated via CloudFlare's system - in this case, please contact us for a `CSR` file for use with a provider of your choice.

SSL Manipulation Commands

Converting private keys to RSA

When uploading the keys into AppEngine, the file must be in RSA format. To verify, the beginning of the file should be

`-----BEGIN RSA PRIVATE KEY-----`.

If you read

`-----BEGIN PRIVATE KEY-----`,

you need to convert it, using the following command:

```
openssl rsa -in key -out key.rsa.key
```

Extracting Certificate and Private Key Files from a .pfx / PKCS#12 File (includes both the certificate and the private key)

- export the private key: `openssl pkcs12 -in certname.pfx -nocerts -out key.pem -nodes`
- export the certificate: `openssl pkcs12 -in certname.pfx -nokeys -out cert.pem`
- create RSA key / remove passphrase from the key: `openssl rsa -in key.pem -out server.key`

Check if a given key matches a certificate (or CSR)

By running these commands on the keyfile and certificate, you can verify that the key used to generate the certificate matches the one you have on hand. If the two outputs match, so do the keyfiles. But if not, your client may have used

their own private keys to create the certificate, which you will have to obtain before forwarding it to us.

- `openssl rsa -noout -modulus -in privateKey.key | openssl md5`
- `openssl x509 -noout -modulus -in certificate.crt | openssl md5`

Alternatively, if you have the CSR as well, you can use the following command to obtain the checksum of the CSR's key and verify that the CSR you have on hand was used to generate the certificate.

- `openssl req -noout -modulus -in CSR.csr | openssl md5`

Proxy modes - X, P, live

X-proxy (testing, JS bugs, JS fix domains , CORS)

The X-proxy is great for testing. You can spot content that does not get picked up by default, and make your configurations to your project, and check for success.

There are a couple of situations, when the X-proxy comes in handy:

- Testing regular expressions, for example on e-commerce sites.
- Testing JSON (JavaScript) and XML translation.
- Just browsing through a site, for evaluation purposes.

An example X-proxy URL: `https://de-de-{project_code}-x.app.easyling.com`

The X-proxy can be accessed from the pages list under Content (or Discovery) by clicking on the Preview button in the hover toolbar, while holding down the Ctrl/Cmd button, or you can just replace the -p for a -x in the normal preview's URL for the same effect.

P - Preview

The standard proxy mode to view the translated website before publishing. However, the preview can be used for a couple of other things:

- Cookie header extraction to get behind logins
- Visiting pages manually, to ingest content

An example Preview-proxy URL: `https://de-de-{project_code}-p.app.easyling.com`

C - Channel

Live serving mode

After publishing the website, the proxy serves content on the chosen domain.

HTTP Headers

What are HTTP Headers?

HTTP header fields are components of the header section of request and response messages in the Hypertext Transfer Protocol (HTTP). They define the operating parameters of an HTTP transaction. The header fields are transmitted after the request or response line, which is the first line of a message. Header fields are colon-separated name-value pairs in clear-text string format, terminated by a carriage return (CR) and line feed (LF) character sequence. The end

of the header section is indicated by an empty field, resulting in the transmission of two consecutive CR-LF pairs. (Wikipedia)

Headers and the proxy

The Proxy strives to be as transparent when it comes to headers as possible. Therefore, we forward the majority of headers added to any incoming request, with a few exceptions where the presence of said header could cause undesirable operation in the original server.

Additionally, The Proxy also adds a few specialized headers both to requests to the remote server and responses to the client. The presence of these headers SHOULD NOT cause erroneous behavior in the server.Request headers contain additional information on the client viewing the site and the language being served. This can be used to provide customized content. Some Proxy-specific request headers are:

- `X-TranslationProxy-Translating-To: ja-JP`: This gives the language of the translated version the client is browsing.
- `X-TranslationProxy-Translating-Host: jp.eveonline.com`: This header contains the domain under which the Proxy is serving the translated site.
- `X-TranslationProxy-Originator-IP: 192.168.168.192`: If enabled, this header contains the IP address of the requester, which may be hidden by CDNs and other proxies.
- Future headers of the `X-TranslationProxy` containing other metadata
- `User-Agent`: This header is somewhat special, in the sense that it's not specific to The Proxy, rather, it is sent by almost all browsers to identify themselves. The reason it finds a place in this article is that it can be used to identify Proxy-served requests. Google AppEngine modifies this header when sending requests, in a way that ensure no further tampering before the header reaches the original server, by adding `AppEngine-Google; (+http://code.google.com/appengine; appid: u~skawa-easyling)` - this can be used to whitelist the Proxy.

HTTP Headers and Security

Due to its nature as a proxy, and the fact that Google URLFetch uses a diverse range of addresses allocated randomly, it is easy to see that proxied requests may be caught by Web Application Firewalls, Anti-DDOS software, or even security provider companies. When a project is launched, it is often a good idea to contact the client and have them notify any contracted security providers, or make the necessary changes to firewalls and block-lists.

The easiest way to identify proxied requests is to read the `User-Agent` header, and locate the above-mentioned pattern - the application ID is added by Google at the last possible moment, thus, it can be a trusted indicator (along with the other headers, if needed) that the request was initiated by the Proxy. Security providers can be advised that the appearance of these headers is normal and should not be construed as an attack/phishing attempt.

CHAPTER 4

The Dashboard 2.0

Complexity Matrix

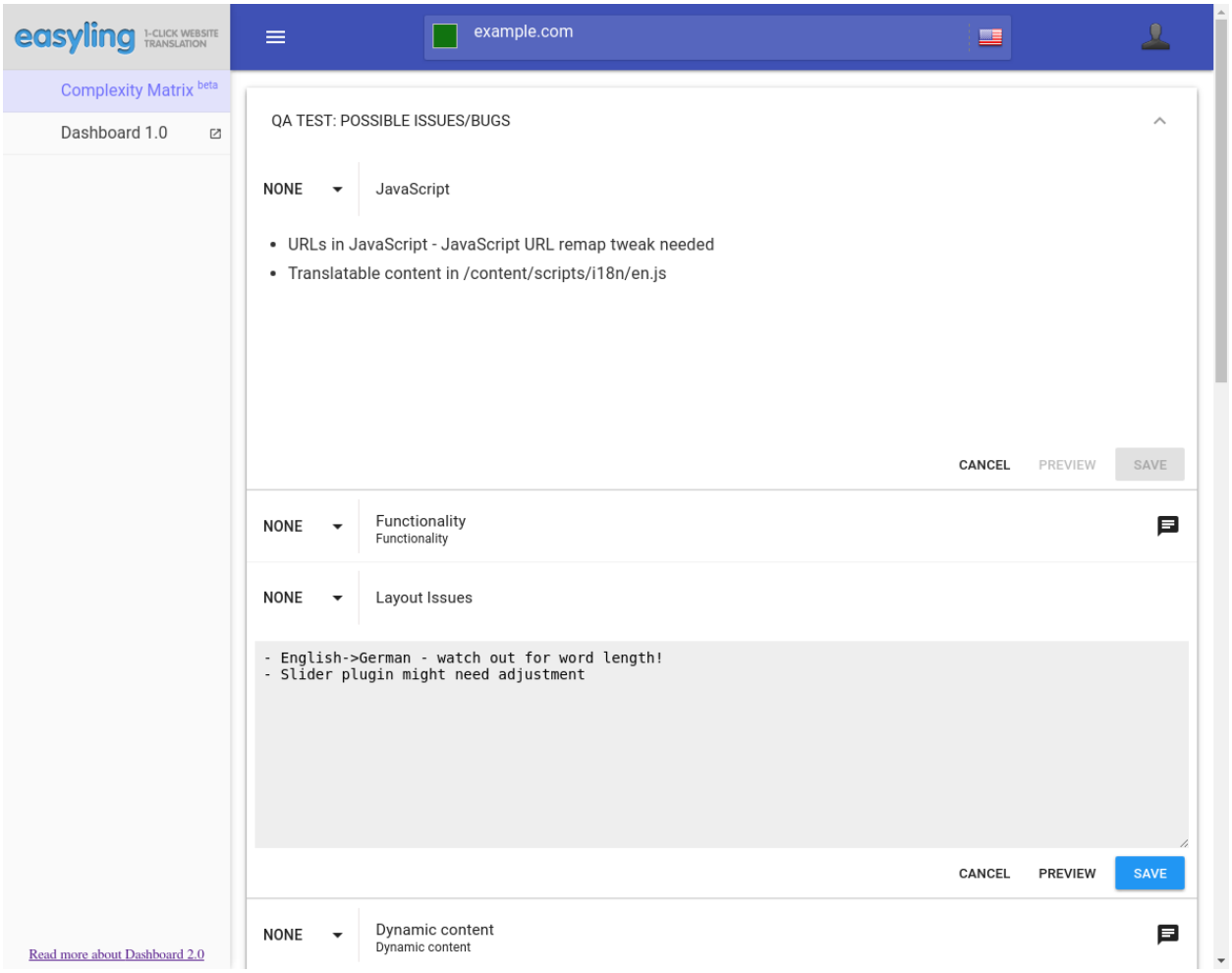
The Complexity Matrix (CM) is two things: first, it is a **note-taking application** where you can get your knowledge of a project domain and various potential issues in order. Second, it is the first feature to showcase the look & feel of the **Dashboard 2**. Feedback is welcome!

You'll recall from the introduction that there are a number of questions to ask when you embark on a new project, so as to create better forecasts for your project expenses and the required amount of work to get the project going. The CM is meant to make this easier via a screen that you can use to track issues from within the project itself.

Accessing & Using the CM

While incremental development of the Dashboard 2.0 is ongoing, the old interface remains available. From there, go into the Dashboard menu and click on **Complexity Matrix** to access the new Dashboard and the CM screen.

Navigating the CM



Expect the left-side menu to have additional items added as development of the new interface progresses. For now, it contains the menu entries for the Complexity Matrix itself and a link to what we have now retroactively dubbed *Dashboard 1.0*, the old user interface. The left-side menu itself can be hidden using the hamburger menu icon on the left-side of the header.

The header also contains a search field and drop-down that you can use to search in your projects. Click on any of them to open their associated Complexity Matrix.

In the CM, a project consists of four distinct aspects you can use to categorize various issues:

1. **QA Test: Possible Issues/Bugs**
2. **Code**
3. **Layout**
4. **User Experience**

You can open and close each entry with a single click. When opened, they will reveal further, more granular categorization of potential issues that warrant attention (discussion of which follows below).

Each categorization has an associated dropdown on the left that you can use to mark it as present. You can mark an issue as causing "**MINOR**", "**MAJOR**" or "**SEVERE**" problems over the proxy. Otherwise, you can leave the dropdown in its original position of "**NONE**".

Each of the categories can be clicked to reveal an editing field that you can use to keep track of the various issues that you detect on the site. The text fields support markdown/basic HTML.

In the following sections, we'll discuss the various aspects of a project's potential complexity using the various available categories in the Complexity Matrix, and list a few examples that are worth documenting if you come across them.

QA Tests

QA tests should be considered as part of the project setup phase. In the remainder of this documentation page, You will find a non-exhaustive list of possible issues that warrant careful consideration.

JavaScript

Content in JavaScript

The proxy will find all translatable content in the HTML source, but such is not the case with JS: the appropriate JS Translation paths need to be worked out and added to the Advanced settings. Use the x-proxy after Discovering/Scanning a site to reveal any content that was not found by the crawl.

If you notice that specific parts, such as the navigation bar or a user account page are not x-ed out in the x-proxy, it is likely that that content is in JS in `<script>` tags, or discrete JavaScript Resources that are requested as the page is loading.

Investigating the original site and the proxy preview for content in JavaScript increases project setup time.

Things to Note

- *Watch the **domain of origin** for JS resources. Translatable content on external domains takes longer to set up.*
- *string concatenation (especially that of HTML markup) is difficult to overcome. The example below is a **sign of trouble**!*

```
var exampleVar = "world"
var htmlContent = "<div><p>Hello " + exampleVar + "</p></div>"
```

URLs in JavaScript

JS code often contains URLs or pathnames that are referred to elsewhere when making requests (both for intra and extra-domain Resources). Since the proxy will remap the project domain according to its own preview and publishing settings, it is possible that the JS URLs will point to the wrong place.

Things to Note

- *Fully qualified, non-concatenated URLs can be remapped via a tweak available in Advanced settings.*

Functionality

HTTP and HTTPS

If you notice that a site uses HTTPS, pay attention. Since the proxy preview domains are certified themselves, you can go a long way without realizing that the Live target language sub-domains will also require certification. Procuring an appropriate SSL certificate takes time, and it can become blocking in the Publishing phase.

Things to Note

- *An SSL certificate for the published sub-domains is necessary.*

Site Search

Most large-scale websites nowadays have a site search functionality that requires additional work on the proxy end of things to intergrate appropriately.

Things to Note

- *Evaluation of the Site Search functionality increases project setup time.*
- *The proxy supports Google's Custom Search Engine. The integration has to happen via JavaScript page modifiers to maintain the original look and feel of the site - this, however, means coding work.*

Language Selector

Many sites already have some sort of localization solution in place. Investigate the current language selection mechanism for potential issues.

Things to Note

- *If a language selector is already present, it might require some tweaking on the original server to better handle the **link mapping** that happens over the proxy. The `___pTNoremap` class needs to be added to `href` attributes that should not be remapped.*
- *If there is no selector, it will have to be implemented,or one of the default proxy selectors used. This requires action from someone who has **access to the source** on the original server - these changes can increase the time it takes for a project to go live.*

Plugins

External i18n Modules

Closely related to the Language Selector, the presence of i18n modules can life easier or more difficult, depending on what you can do with them. Consider whether these modules can be reused in the context of the proxy or if new translations will have to be provided.

Things to Note

- *Is the i18n module part of minified code? Is it dynamic content? These are usually extremely difficult to reliably reuse.*

User Sessions & Logins

Sites often have sections that require login credentials. The proxy can handle these sections if it is passed the appropriate **session cookies**.

Things to Note

- *Extracting session cookies is an additional step that needs to be executed for each Scan/Content Ingestion Cycle.*

Query Parameters

Sometimes, the project page list will contain many instances of the same path with different query parameters after a Discovery. The thing about query parameters is that they do not necessarily imply unique content for the specific settings they engender. For example,

https://example.com/products?sort=asc

https://example.com/products?sort=desc

could plausibly serve exactly the same content, only in different order. Conversely, it could also be requesting an entirely different set of data to use in the browser. This requires careful combing of the site to make sure that all necessary content is extracted, but no superfluous pages are kept.

Things to Note

- *The proxy can ignore or group specified query parameters, but figuring out which ones to keep and which ones to throw away will add to project setup time.*
- *Consider also that independently made changes to the original site might instantly deprecate your current project settings - this also has implications for the maintenance phase of a project.*

Dynamic Content

XHR, AJAX, XML and JSON

Things to Note

- *Dynamic content has to be extracted manually* ***via the Preview***.

Webshops

Single-Page Applications (Angular, etc.)

Layout & CSS

Word lengths

English and German, for example, differ considerably in their average word lenght. While not normally an issue in the case of paragraphs, this can become an issue if some part of the site layout is too tight (for example, a navigation header with widths declared in pixels). If too many assumptions are ingrained in it about the amount of content that has to fit in a given element, that might need some tweaks.

Other times, enclosing elements (such as border boxes) will change their size based on the amount of text they contain. This potentially requires some target-language specific tweaking of the layout via CSS/JS Page modifiers.

Things to Note

- *Some things can be made to fit via CSS tricks and Page Modifiers. In other cases, only rewording will help if extensive reworking of the site layout is not an option.*
- *The other case may also happen: Chinese terminology, for example, can be considerably shorter than its German counterpart. Too much room is less often a problem than too little, but it is useful to keep in mind nevertheless.*

Text Direction

Some target languages, such as Hebrew or Arabic require that the site be redesigned from Left-To-Right to Right-To-Left via CSS rules/page modifiers/framework-specific RTL libraries. This usually means an extensive reworking of the site structure using development tools.

Not only does the CSS/JS parts of RTL redesign (which has to be done before the project is published) take a long time, there are some parts of a site that require even more detail work. Images with specific directionality (a left-pointing arrow, for)

Things to Note

- *Keep track of UI elements that have explicit directions (such as images).
- *Interactive elements and vendor plugins usually mean many hours of work to get everything just so. Be extra careful!*

Responsive Layout

Responsive site design is a compounding factor for all extant layout issues. Websites are expected to work with a wide variety of screen sizes and browser settings, you have to take it into account. Use the Chrome DevTool's device mode to gain understanding of site behavior over the proxy (and in general).

Code Issues

The list of potential issues is not meant to be exhaustive. Nevertheless, it is heavily **work-in-progress**. Suggestions and feedback are welcome!

<To-Be-Written>

Translatable Content in JavaScript

External Content, Project Linking, Forms, iframes

Hubspot & Marketo

Layout

Responsive Layout

Proxy Compatibility

User Experience

Wording, Layout

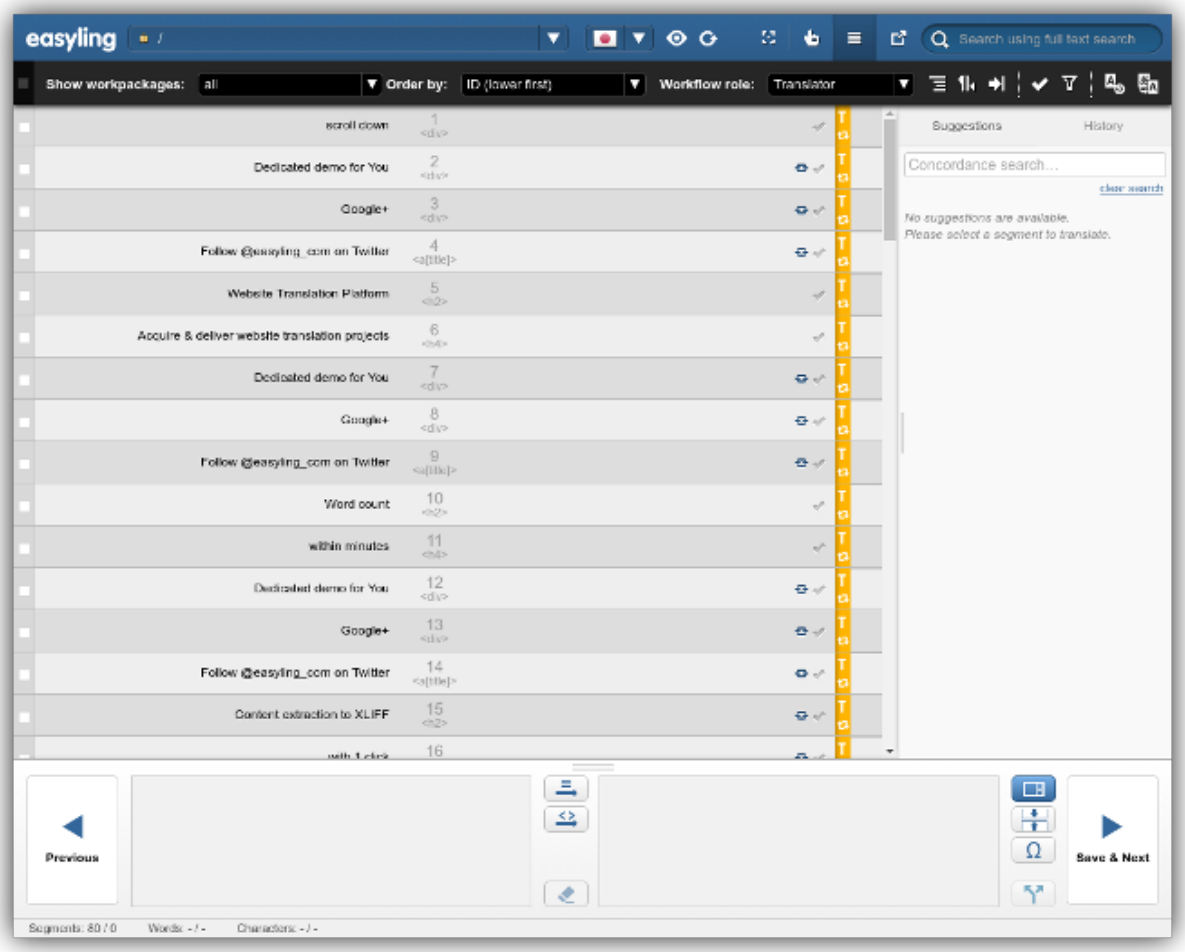
Content Access

Consistency

CHAPTER 5

The Workbench

Introduction



The Workbench is a CAT tool in the cloud.

If the Dashboard is our take on the project management side of webiste localization, then the Workbench is the analogous CAT tool you can use without ever leaving the browser window.

In this chapter, we take a close look at all the various features available on the Workbench. The explanations pertaining to the various elements and functionality of the Workbench screen are grouped into thematically linked subsections. Use the sidebar to navigate to the subsection of your choice.

Opening the Workbench

You can access the Workbench anytime after adding at least one target language.

There are a few ways to enter it:

1. In the Dashboard menu, see the *Languages* part. If you hover over any of the target languages, a menu bar will be displayed to the right. Click on "Translate in List View"..
2. Click on "Manage segments" in Content menu...
3. Enter the Page list and hover over any entry to display a menu to the right. Click on "Translate in List View"...

.. and the Workbench will open in a new tab.

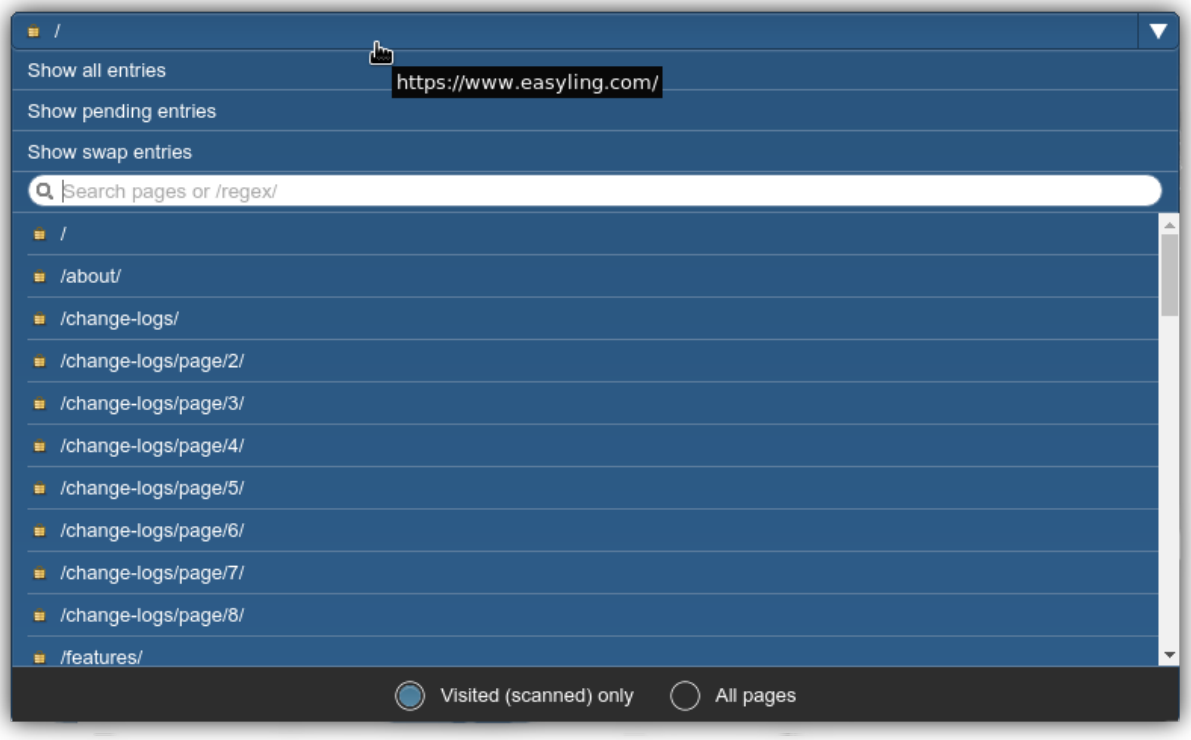
Moving Around

The Workbench has a single viewport, so every feature you'd need to navigate between pages and segments is always just a couple of clicks away.

You may switch between pages, search for text, filter for segments based on a variety of metadata, such as Approval state, containing block element, translation source and so forth. This section deals with the various ways to do that.

Workbench Page List

A segment is tied to the specific page it was found on. You can use the Page list dropdown right next to the logo to get a list of all pages currently within the scope of translation.



You can click on any page entry in the list to visit that page and get an overview of all segments associated with that page. Websites can get very large with a huge list of pages - use the search field embedded in this dropdown to locate a specific page (you may use regular expressions with the usual format of including them between two slashes like this: `/[regular expression]/`).

There are three options at the top of the dropdown that bear special mention:

Show All Entries

Most of the time, you will want segments displayed for a specific page, but you may also use this option to get an overview of all segments across all pages.

WARNING! Only **List View** is available in this view, all other View buttons will be unavailable!

The All Entries list doesn't flood your browser with every last segment all at once: segments will be loaded in batches of 500. The Workbench will automatically fetch a new batch of entries as you scroll down.

Show Pending Entries

By default, Scans will pick up new entries in "Approved" state, which in this case means "Approved for Translation", immediately available for translation. You can change this default behavior, by going into the Dashboard and changing it in Advanced settings to either Pending or Excluded.

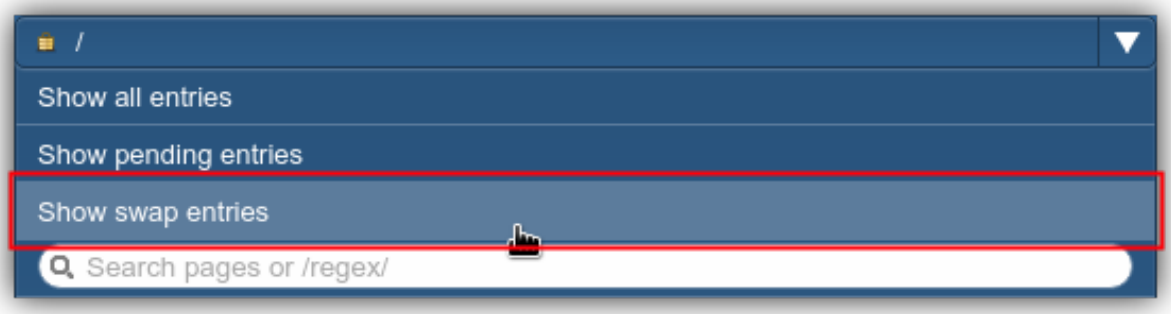
By clicking on "Show pending entries" in the page list, the list of all entries that are currently waiting for Approval is displayed. In their current state, They will not be included into exports unless the relevant option is selected at export-time, and will not appear for translation unless filtered for specifically.

Project or backup owners, or users with the Customer role can move these into either one of the other two states, by **approving** them or **excluding** them entirely from the scope of translation.

Show Swap Entries

Swap entries are those segments that have had the "EL_swap" class added to their enclosing tags on the source site.

They are special in that they are added to the Workbench without processing their tags. They will be displayed verbatim, allowing you to edit the source content markup directly. That content will be sent as-is by the proxy for each request.



Take caution when editing swap entries. All responsibility of rendering them successfully *and* safely is delegated to the requesting browser.

Filters

There is a comprehensive assortment of filters available. Click on the Filters icon in the toolbar to get a an overview of all available filters:



Use the checkboxes to define your filtering settings in the dialog and click on "Set Filters".

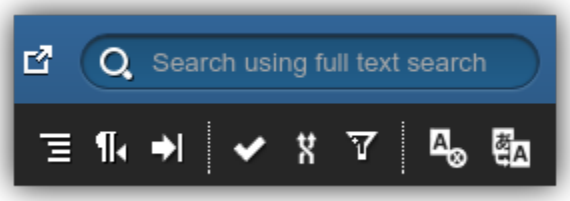


The dialog will close and a new element will appear in the toolbar, indicating that user-defined Filtering settings are currently active, and the segment list is updated accordingly. Click on the "X" to disable filtering. You may also click anywhere else on the toolbar indicator to open the Filters dialog again and fine-tune your settings.

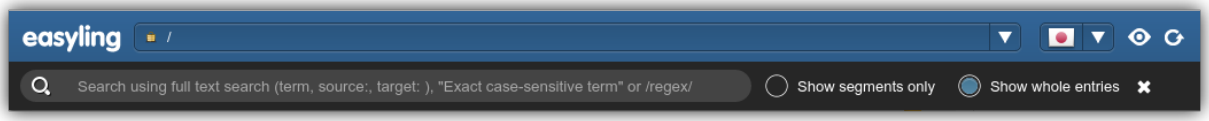


These filters work with the various types of metadata associated with an Entry (such as currently assigned workflow role, enclosing block element in source, approval state or source of translation), **not** content - use the **Search** functionality to filter for source or target text.

Searching for segments



Use the full-text Search field in the upper-right hand corner of the screen to search for segments. When you click on the Search field, the Workflow part of the black menu bar will be superimposed with a search field right above the Viewport.



Normal full-text search and regular expressions are supported. The normal search can be a bit misleading, as it is a forward-from-word-boundary search. If, for example, you'd like to find `"translation"` in a source segment, a search query for `"translatio"` will turn up all segments containing any words that start with that exact set of strings (and likely end with `n`,`ns`, or `nese`, perhaps). Searching for `"ranslation"` only, however, will return zero matches.

Use regular expressions to search between word boundaries: to extend the previous example, a query for `/.ranslation/` will show all segments that contain any character (except newline) exactly once, followed by the literal string `ranslation`. The same as with the Page list, a string is interpreted as a regular expression if you enclose it between slashes.

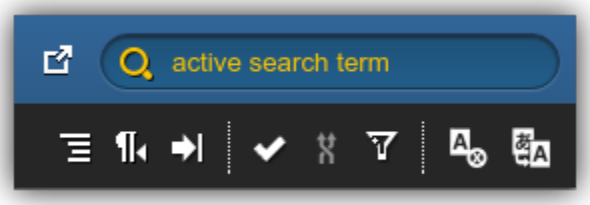
You may select between displaying segments or whole entries using the radio buttons next to the Search field.

Closing the Search Field

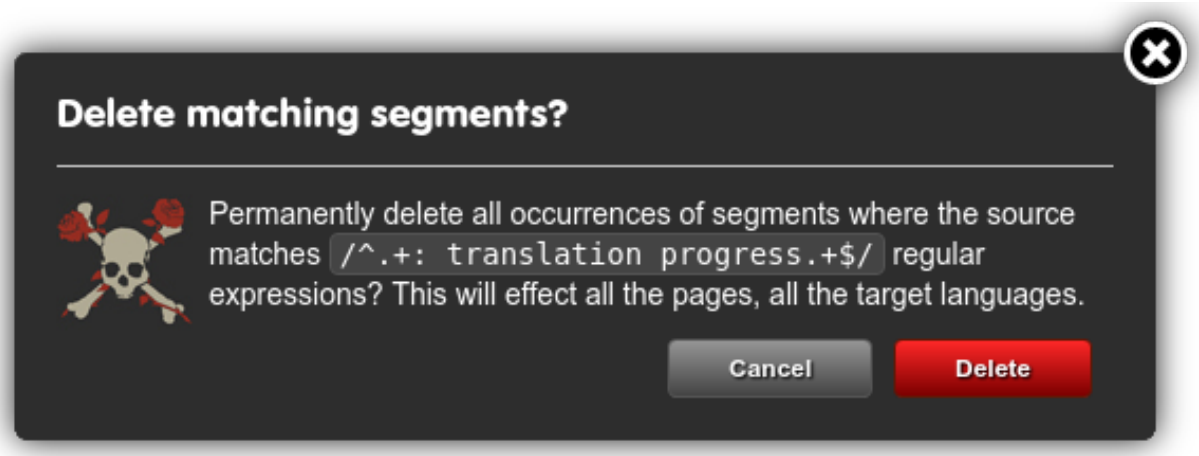
It is important to remember that the search field is also a **filter**: as long as it is active, segments will be filtered based on its contents regardless of being in All entries view or on a specific page.

If you wish to restore full view of segments, clear the search field and send an empty search.Use the close button next to the Search display options to close the search field.

If a search string is still present, it will be preserved and displayed in the upper right corner Search field in orange, like this:



Deletion of Segments



You might be wondering what this has to do with navigation, but executing a regex search reveals another feature of the Workbench you might ordinarily look for someplace else: that of deleting segments. If you click on the Magnifying glass icon while the regex search is active (a bottom-facing triangle will indicate availability), the dialog above will open.

The non-discoverability of this feature is premeditated. Skulls & Bones warnings would generally apply to **any** situation where the words "delete" and "regex" are found in the same sentence. That being said, to give you a measure of peace of mind, deletion of segments is not as final as we seem to make it out to be: when TM Freeze is **disabled**, you can re-add segments anytime by Scanning or visiting in Preview the page that contains the deleted segments.

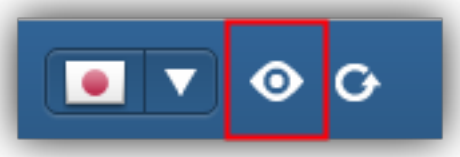
But that is new words added each time. So, if for no other reason, be careful about deleting segments in order to avoid unnecessary expenses. Buyer, beware!



Preview

By clicking on the “Eye” icon, you can visit the temporary domain to check your translations in their original context. In the “All Entries” view, this function is disabled if no segment is selected - without picking a segment, the Workbench has no information on which page to show you. Otherwise, if a Page view is open, the selected page will be loaded.

If you select a segment in All Entries view, the Preview proxy will open on the page where it was seen by the crawler for the first time.

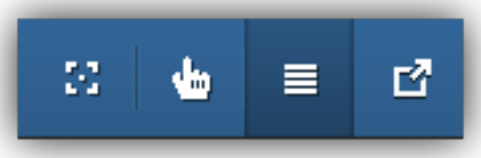


The icon on the Workbench will take you to the Preview mode, but there are a few different Proxy modes available besides that. See here for details.

There is more than one way of looking at content on the Workbench, and the default, the List View is only one of them. In this section, we go over the various in-context ‘Views’ you can access from the Workbench and use to edit your translation while making sure that it is behaving exactly as it should in the original context.

Views

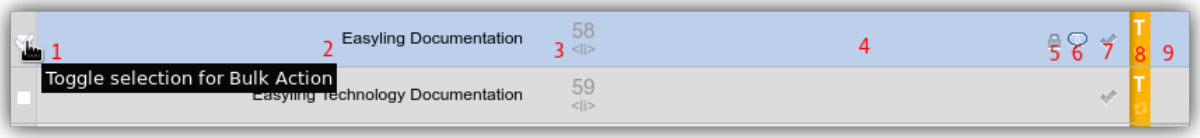
List View



While not necessarily the most impressive, the List View is certainly one of the most useful views on the Workbench. You can use it to go over each segment being translated and edit, filter and search for any subset.

The List view provides various features to use with each entry. The currently selected Entry will be highlighted in yellow, hovering will causes the Entry under the cursor to highlight in blue. The presentation of Entries is clear and simple. yet there are a variety of features you can use with each. Let's take a detailed look at an entry and see what each part of a line does.

Anatomy of an Entry

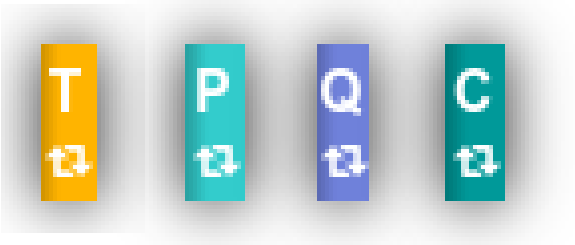


1. **Select Checkbox**: check this box to select the entry or segment in question. You may use the "Bulk Actions" icon to batch process selected entries (i.e. confirm, exclude or approve them)
2. **Source Entry**: contains the text that was found within a given block element on the source site. You may set the text direction using the "Align source text" icon in the toolbar. Otherwise, the Source Entry is not editable.
3. **Entry No. + Containing Block Element**: use the "Go to Segment" icon to jump to a segment with a given number. This number is assigned to entries in order of arrival (new entries can be found at the bottom of the list). Additionally, the tag that contained the text on the original site is displayed below it (useful when you want to identify a segment in the HTML source).
4. **Target entry**: The translation, provided by a variety of sources: Manual Editing, Translation Imports, Machine Translation or Translation Memory.
5. **Lock Segment**: Prevent any changes from influencing the current content of the Target Entry. Especially useful when you want to run batch processes on your segments, but you want to exclude an entry from the scope.
6. **Comment on Segment**: Use this icon to add comments to an Entry either as a note-to-self, or as part of a collaborative translation effort. All comments have a checkbox next to them, allowing you to mark them as settled.
7. **Chain link**: indicates that the segment is repeated verbatim (102% match) in the current view. Click on the icon to jump to the next repetition.
8. **Confirm Tick** click on the tick to Confirm the segment for the current workflow and send it to the next stage. Confirmed entries remain editable as long as they are unedited by the next workflow role.

9. **Workflow Status Indicator**: Display the current workflow state of the segment. Note that this might differ depending on which user in which workflow role is currently looking at the segment.
10. **Flag**: Display current translation source.

Workflow Tags

Depending on what Workflow role you're currently in, the following Workflow tags will be displayed for each segment, influencing their availability for editing:



T - Translator

P - Proofreader

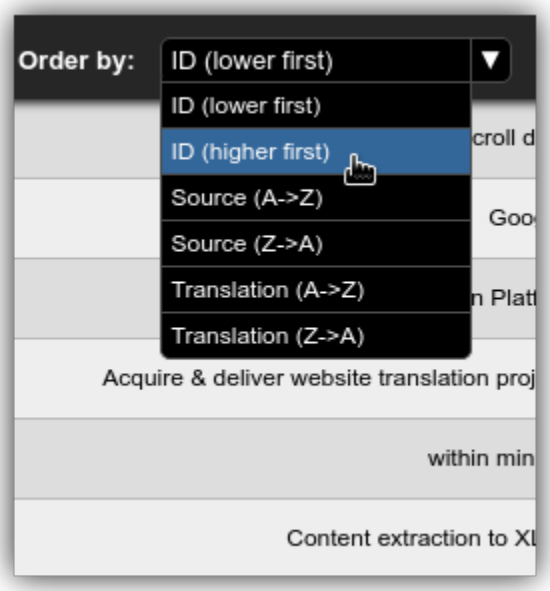
Q - Proofreader 2 (**Q**uality Check)

C - Customer (or **C**lient)

See the section on Workflow roles for a detailed description of the various workflow roles.

Flags

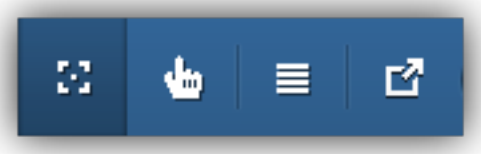
Ordering Segments in List View



Use the "Order by" dropdown to alphabetize the target or source entries or reverse the order of segments based on their Workbench ID. The following Ordering methods are available from the dropdown:

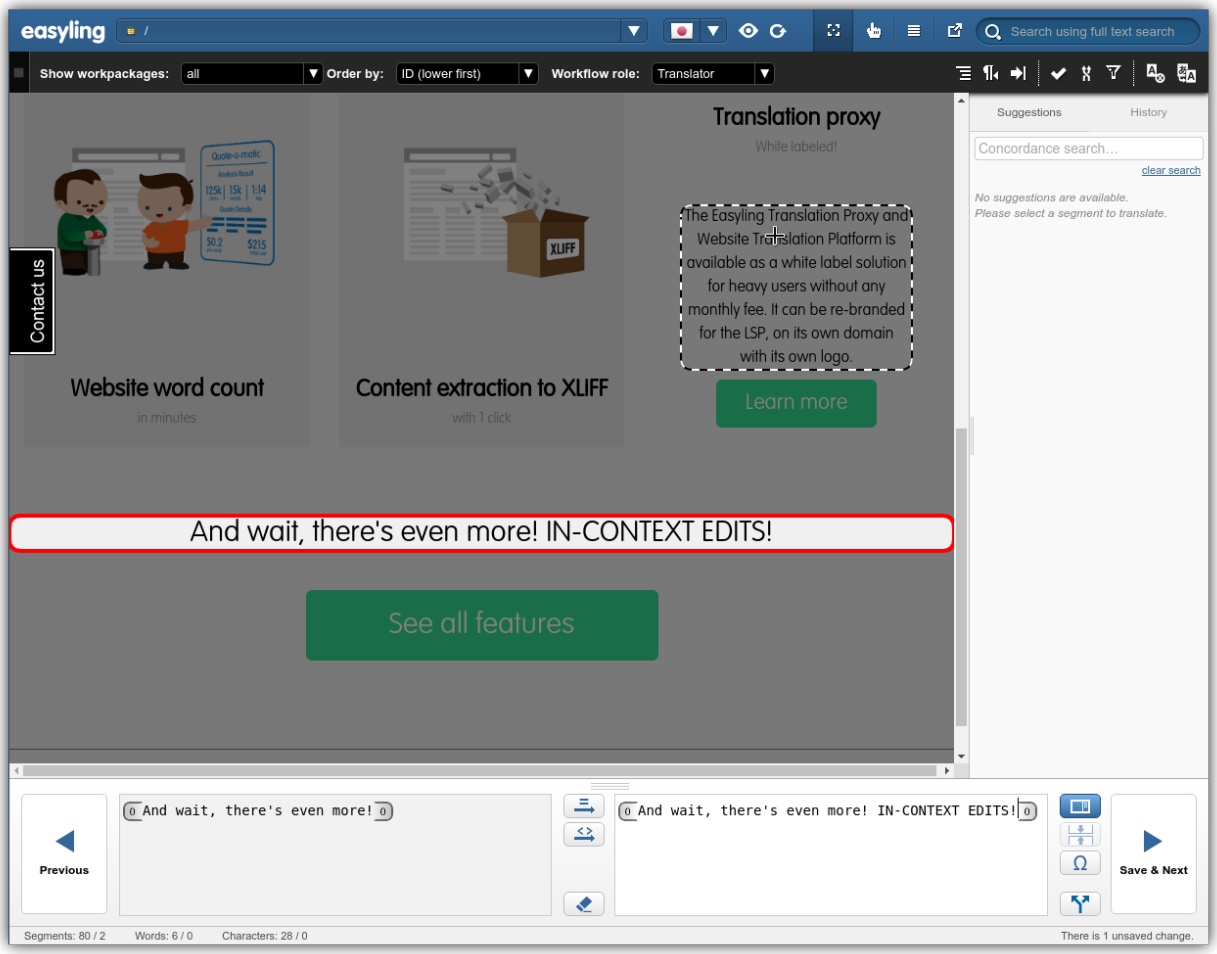
- **ID, lower first** (default)
- **ID, higher first** (use to quickly navigate to new segments)
- **Source A → Z**
- **Source Z → A**
- **Translation A → Z**
- **Translation Z → A**

Highlight View



The highlight view is a true in-context editing view that makes the Workbench popular with Translators, and *the* solution to the problem of adequate context during website localization.

By selecting an Entry in List view and clicking on the Highlight View on the Workbench, you will be shown the text on the original webpage



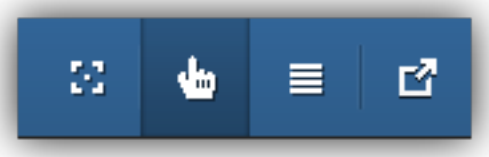
You may click on any part of a website and have a highlighting frame appear around that segment. At the same time, the editing box below will jump to the segment in question, where you can add/edit your translations in-place.

Really, the Highlight View is simplicity itself with very little in the way of hidden gotchas. Select a page, point & click, and translate away!

But keep in mind that while you are using the Highlight View with a page, links will be unclickable - use Free-click View to navigate on the original site from within the Workbench.

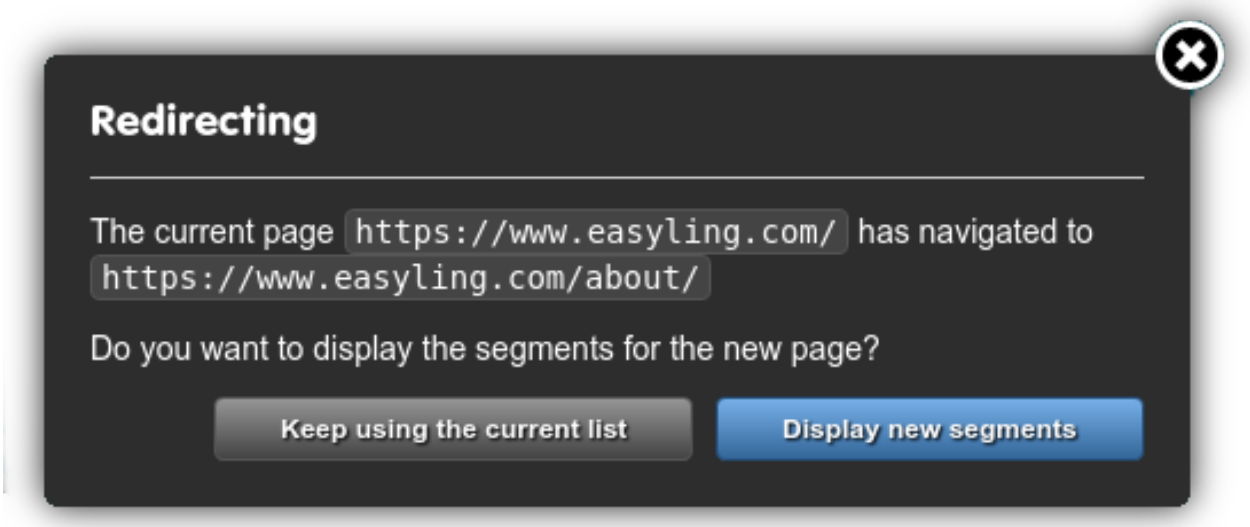
NOTE The Highlight view is a wonderful tool we are very proud of, but don't forget that much of the textual content of a website is not clickable. Check the other modes and the Preview to make sure that everything is covered!

Free-Click View



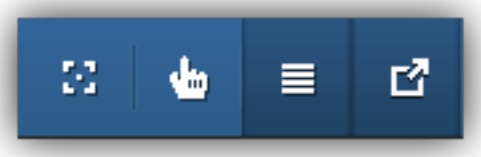
The Free-Click View is much like the Highlight View, but it allows you to use the links on a site to visit other parts of the website. Alternate between Free-click view and Highlight View to translate segments as you explore the website.

Free-click view will offer to reload your segment list based on which page you are on. The following dialog will be displayed.



Highlight View will only work if the segment list is that of the current page.

Pop-out View



The Workbench is a single-viewport application, and the Pop-out View is a feature meant to allow you to have the Highlight View and the List View on your screen **at the same time**. Select an Entry from the list and click on the Pop-out view icon to open a Highlight View in a separate pop-up window.

Modern browsers block popup windows by default, so you will most likely have to **enable popups** for the Workbench for this feature to work.

Translation of Segments

We are getting to the primary purpose of the Workbench, which is of course, translation of text. In this section, we'll talk about the different ways of doing that.

It bears mentioning right off the bat that the Workbench is not meant to selfishly yank you out of your CAT tool of choice. As the proxy supports exporting segments in the industry-standard XLIFF format, you can always employ any tool of you preference, SDL Trados and MemoQ, for example, being two major players in the field. The choice is yours.

Nevertheless, you will find that the Workbench is closer to where the website really is - up there in the cloud, and therefore immensely useful during the editing process. You'll see it truly begins to shine when your XLIFF files have done their first round-trip to and from external CAT tools.

There are multiple ways of translating content on the Workbench:

- manual Translation using the Target editor
- using Translation Memories
- using Machine Translation
- importing translated XLIFF files

The following sections provide an overview of the translation methods listed above.

Editing segments

Editing of segments happens in the Target editor located on the bottom of the Workbench:



You can use the Highlight View or the List View to select a segment, and the editor will reflect your selection. The Source entry is displayed to the left, the translation (Target) to the right. Only the target is editable.

Editing Window

Editing translations on the Web means going around/avoiding/leaving intact the forest of HTML tags that the text os usually embedded on. The Workbench abstracts away these markup details to ease working with text (the first and foremost task of translators), but doesn't, and will never attempt to hide the fact of their existence.

Tags are represented as numbered grey widgets around certain parts of the text, which you can use as a yardstick to place your translations in the appropriate tag context without having to worry about what those tags actually do.

Two things:

- **You can Drag & Drop the Numbered Tags**
- **You can NOT delete them**

Adding translations is otherwise straightforward text input. Untranslated entries will contain the Source text as a placeholder until such time as a change is made to the contents of the Target, at which point both the List View and the Segment contents are updated.

There are a few smaller buttons in various parts of the Target editor. In the middle, you'll see these three buttons:



These contain editing functions that relate the contents of the Source and the Target in different ways. Use the top "**Equal**" button to copy the Source contents to the Target, rendering them identical. You can use the middle, "**Tag**" button to copy the tag structure of the Source to the Target. The lower "**Eraser**" button will delete the translation and restore the placeholder.

There is another set of four buttons to the right:



They all deal with the Target. In descending order, these are:

1. **Toggle sidebar**: use this to hide the Suggestions & History sidebar above the button.
2. **Preserve Whitespace**: prevent trimming of whitespaces in the segment
3. **Insert non-printing characters** There are a number of non-printing characters you might need during a translation project, especially if you are dealing with languages that have a Right-to-Left writing direction, such as Arabic. See the section on RTL conversion for further details.
4. **Split segment from group**

Saving changes

Click on "Save & Next" to save your work on the given Entry and jump to the next segment. You can also use Ctrl+Up or Ctrl+Down to do this. This is an explicit, although not strictly necessary action, as any edits are **automatically**

saved upon leaving a page, or otherwise after 60 seconds of inactivity on the Workbench. Any navigation generated by the Workbench itself will also trigger a flush of all unsaved segments.

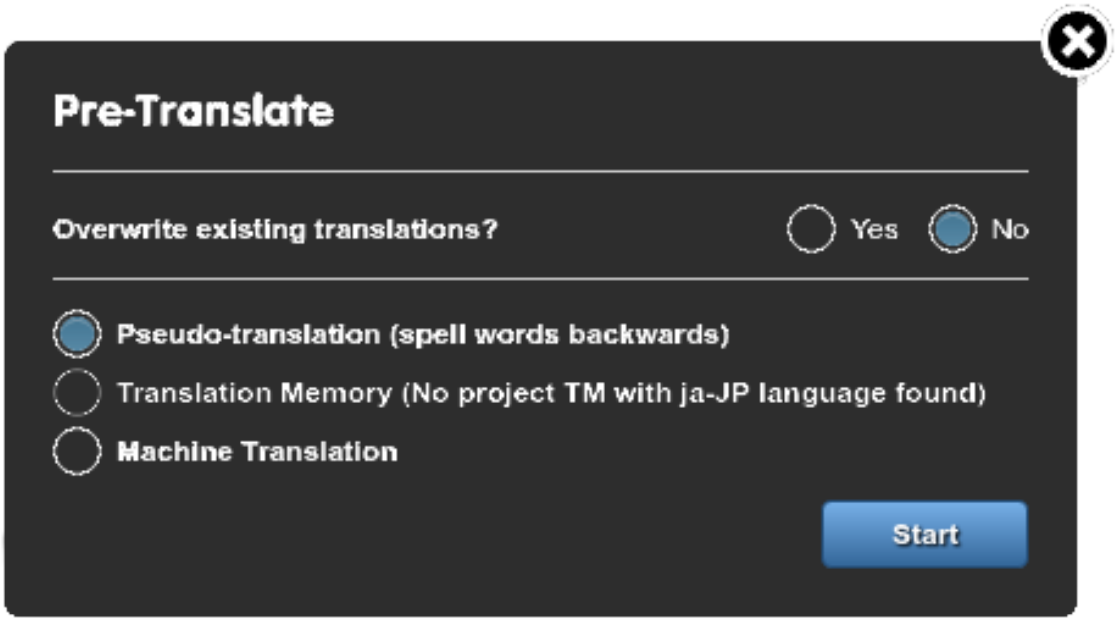
Automatic Translation

There are a number of Auto-translation features that you can employ to ensure quality on the proxy.

The option to run pre-translation or set it up to run automatically is also available on the Dashboard. Here, we discuss the options available after clicking on the "Pre-translate" icon in the toolbar:



The following dialog is displayed:



Pseudo-translation

This function is admittedly not translation in a meaningful sense, but string-reversing each and every word of every segment on the spot is very effective during a demo. Use it, then go to Preview to demonstrate to the client that website localization is as painless as using CAT tools in tandem with the Proxy. Good for the wow!-factor. You can also use pseudo-translation to test the various editing features available on the Dashboard and the Workbench.

Translation Memory

If you have a populated Translation Memory on a project, you may use its contents to translate segments. Use this feature to translate your content with a preset match threshold.

Machine Translation

You can choose to Machine Translate the currently selected batch of entries using one of the available MT APIs (Google Translate, Bing Translate, iTranslate4u and GeoFluent).

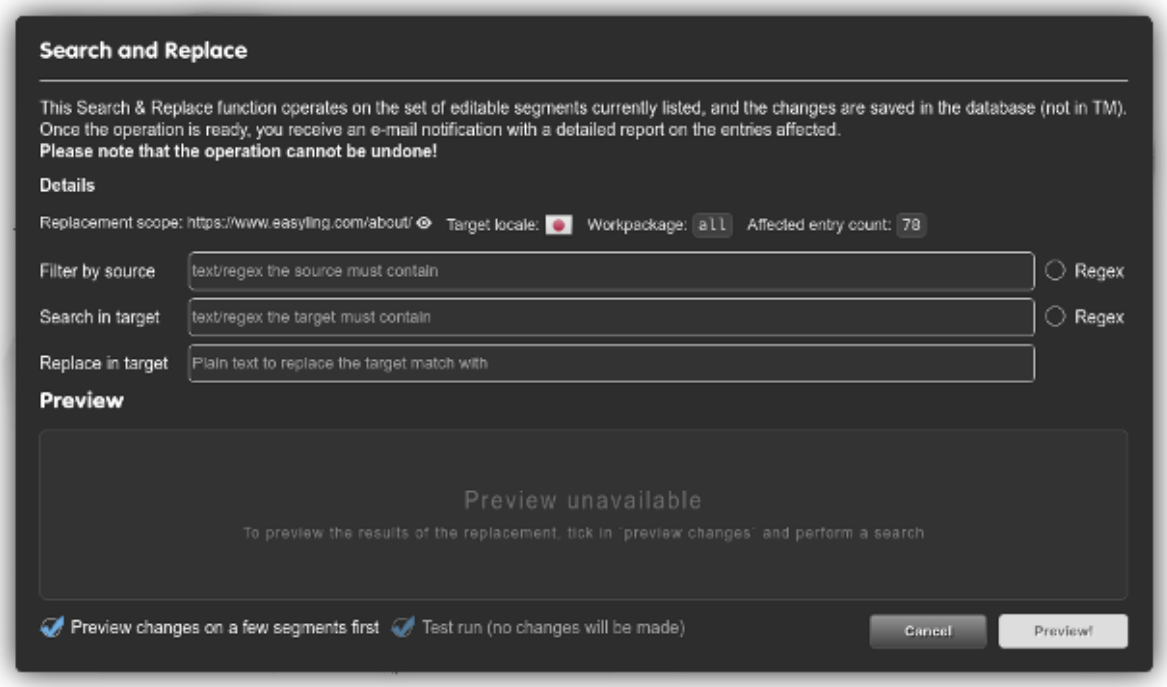
Translation Memory and Machine pre-translation are both reproductions of the options accessible on the Dashboard, with the added functionality of being able to control which specific group of segments should be targeted by the process.

Search & Replace

The Workbench has a Search & Replace feature you can access by clicking on this icon in List View:



It always operates on the currently listed set (or subset) of editable segments. The dialog that opens will look like this:



Both 'Search in target' and 'Replace in target' values are required, while 'Filter by source' is optional. You can use either simple strings or regular expressions as search terms (simple strings are the default, check the 'Regex' option next to the field to use a regular expression). The function is case-sensitive.

Choose 'Test run (no changes will be made)' to check how the operation would affect your translation. To see the entries affected by replacement, check the 'Preview changes on a few segments first'; the segments will appear in the Preview area (Preview can't be used without Test run). You can also start replacement right away, without preliminary checking.

Once the operation is ready, you will receive an e-mail notification with a detailed report on the entries affected.

Please note that the replacement is done in the database, not in the TM. If you want your TM to reflect these changes, you need to run 'Populate TM' from the Dashboard.

Keep in mind that **the operation cannot be undone!**

History

The proxy keeps tabs on what happens to each Entry in the project timeline, and the Workbench displays these tabs in the sidebar. You can use these to access previous editing states of an Entry. Here follows a short description of the History functionality.

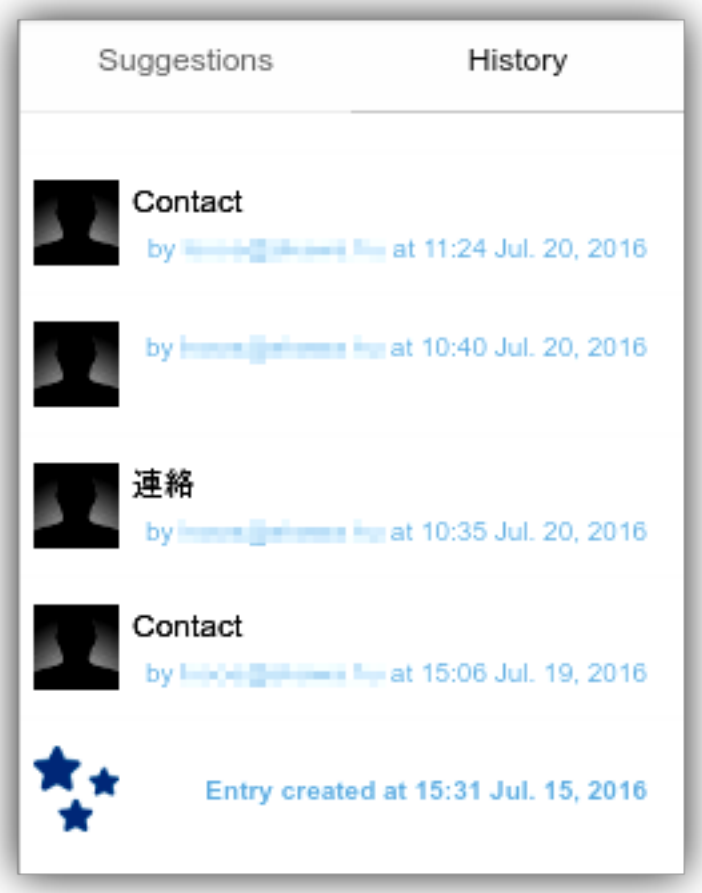
Translation Memories

If you select a segment, TM suggestions will be displayed for it in the sidebar tab labeled "Suggestions". Click on any one of them to add it as a translation for the given Entry.

A Search field is also provided that you can use for concordance lookups.

Segment History

Whether a result of a manual or an automatic edit, each saved state of an Entry will be saved with a username and a timestamp in the Entry history. You can access it in the sidebar tab labeled "History".



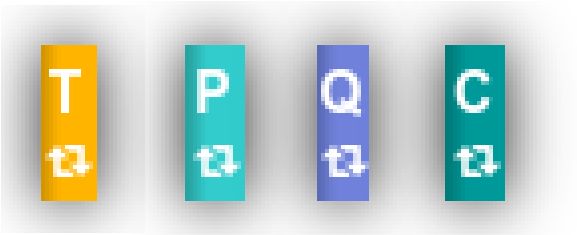
This means you don't have to worry about ever losing translated content as a result of manual edits - you can always restore a previous state of an Entry by selecting the Entry in List View or Highlight View, going to History and copy & pasting a previous state of your choosing.

Collaboration

Collaboration are a must with website localization. If you have used the Dashboard's Sharing settings to invite other people into the project, granted them the appropriate editing rights, the list of segments in the Workbench will look a bit different to each user.

Workflow Roles

As mentioned previously, oles are predominantly a project management feature associated with work on the Workbench. To reiterate, there are four different roles:



T - Translator (default)

P - Proofreader

Q - Proofreader 2 (**Q**uality Check)

C - Customer (or **C**lient)

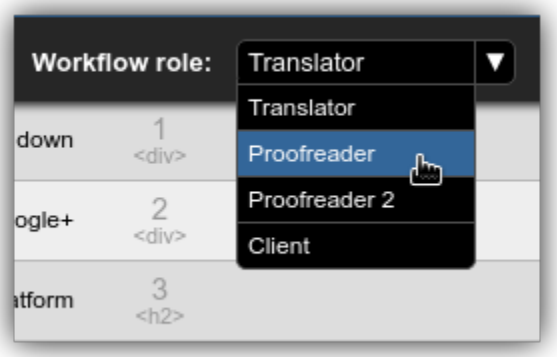
There are four different workflows on the proxy you may employ. You may set these on the Dashboard.

- Simple Translation Workflow (**T**)
- Translation + Proofreading (**TP**)
- Translation + Proofreading + Client Approval (**TPC**)
- Translation +Proofreading + Quality Check + Client Approval (**TPQC**)

Each setting will activate the necessary roles, which the Owner or Backup Owner may assign to any project participant. By default, only the Translator role is required. Owners have access to all workflow roles.

Workflow Roles in Action

Use the Workflow role dropdown in the toolbar to switch between the available Workflow roles:



Take TPQC, the workflow with the most participants, for example.

1. Each **approved** segment is assigned to the Translator role.
2. When finished with a segment (either through manual edits, automatic translation or via XLIFF importing), the translator clicks on the Confirm tick to declare that segment cleared for that phase and send it to the next role, the *proofreader*.

3. The *proofreader* (and much everyone else) may use Filters to display only those segments that are assigned to that role. He takes the segments sent by the Translator, edits them, changes their wording as required. When finished with an entry, the *proofreader* clicks on the Confirm tick, sending the segment along to *Quality Check*.

And so on. This cycle is then repeated until a segment (more to the point, *all* segments) reach the final workflow role, that of the Customer, who approves translated entries.

A few things to keep in mind:

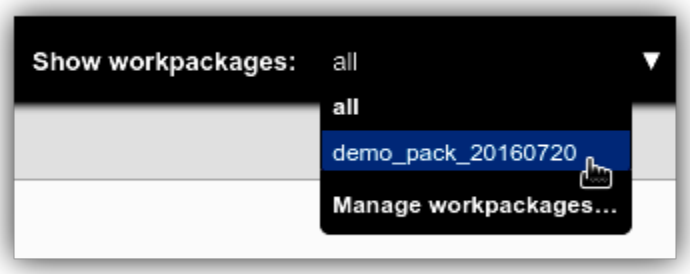
- Each Role has access to the lists of **upstream** roles.
- Only Owners, Backup Owners and Project Managers have access to **all** roles.
- Entries/segments belonging to another role are **greyed out**.
- A segment remains available for editing after Confirming it just as long as it is not touched by the next Workflow role. If you ever mistakenly Confirm an entry, you may, so to speak, **reclaim** it for some more work before the next role can get to it.

And that's about it!

Work Packages

If Workflow Roles is a method of grouping your users, then Work Packages are a method of grouping segments. See the Dashboard chapter for the details of how to generate them.

Use the Work Package dropdown to select a Work package, and the Workbench will display only those Entries that belong to that Work Package (note that an Entry may belong to more than one Work Package!). The dropdown looks like this.



The only default entry in the dropdown is "All", which means disabling Work Package based filtering on the Workbench. As new Work packages are generated, this list will automatically update after refreshing the window. The dropdown will always contain the names of the 100 latest Work Packages.

Clicking on "Manage workpackages..." will take you to the Dashboard where you can tend to your existing Work Packages or generate new ones.

Cookbook

Seach & Replace

Tutorial 1: Fix Spelling (String Replacement)

A simple use case of the *search & replace* feature is the old chestnut: the differences between British and American spelling rules.

This could come up whenever you are working on both the `en-US` and `en-GB` locales, or if two translators, each on a different side of the pond, forgot to coordinate their spelling.

Let's say you have the following targets with German as a source language:

```
The world's No.1 donut
Vanilla donut
Chocolate-chip donut
Doughnut miss it!
```

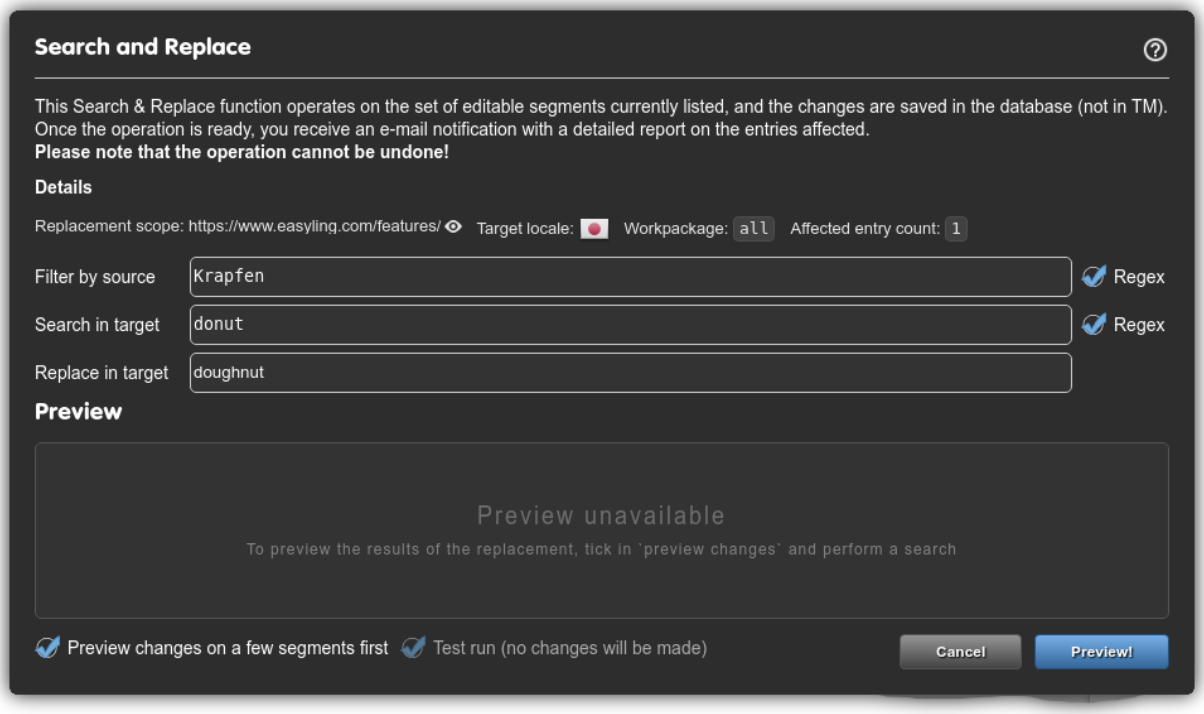
and so on. Replace all instances of the word `donut` with the `doughnut` variant by following the steps below.

1. Click on the Search & Replace icon



1. Fill in the `Filter by source` field to work exclusively on those entries that contain the given string in the source language. This example is about the spelling of "donut", so you would enter the German original, `Krapfen`, to limit the search. Rest assured: source entries are **never** changed.
2. Enter the word that you'd like to replace in the `Search in target` field, in this case, `donut`.
3. Enter the replacement, `doughnut`, in the `Replace in target` field.

That's all there is to setting things up, the rest is about making sure your changes will not cause any problems.



1. Click on **Preview!** to see your changes applied to a subset of segments.

2. If contents of the Preview area look good, uncheck the Preview checkbox.
3. (*Optionally*)do a Test run by clicking on "Go¡'
4. Uncheck the `Test run` checkbox and click on "Go!" to really apply the replacement.

Depending on the number of segments, the process can take some time to finish. You will receive an e-mail for both the Test and Live modes, containing a list of proposed (Test) or applied (Live) changes.

Tutorial 2: to-do (Regex Replace)

CHAPTER 6

Maintenance

Project maintenance

The translation project of a website is a continuous task, as new content is regularly added to the site. So uploading the initial translation is not the end of the project, but the beginning of a new phase. This phase is practically a repetitive cycle of the following activities, many of which can be automated:

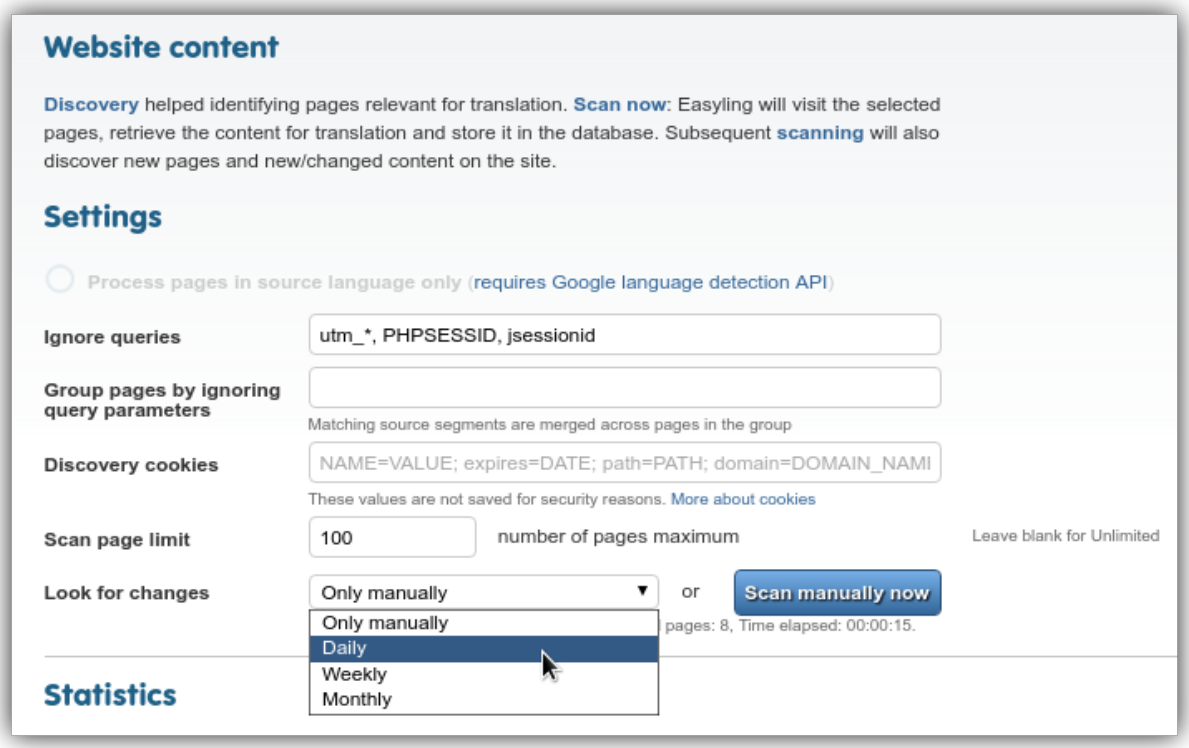
- checking the site for new content
- extracting new content for translation
- translation of the new content
- uploading the new translation
- and, of course, error fixing, if needed

Automation possibilities

The first two activities can be set to be carried out automatically at daily, weekly or monthly intervals, depending on how frequently content is updated on the website. This is called **scheduled crawl**. If you turn on this feature, changes will be checked and retrieved for translation at the specified intervals, and e-mail notification will be sent to all project participants. The new content is available right away for translation in the online editor interface, and you can also download an XLIFF for translation.

Please note that this check is technically content extraction, so it has an associated cost of EUR 2 per 1000 words.

You can enable this option in **Content > Settings > Look for changes**. Of course, you can set it to **Only manually**, if the site does not update frequently.

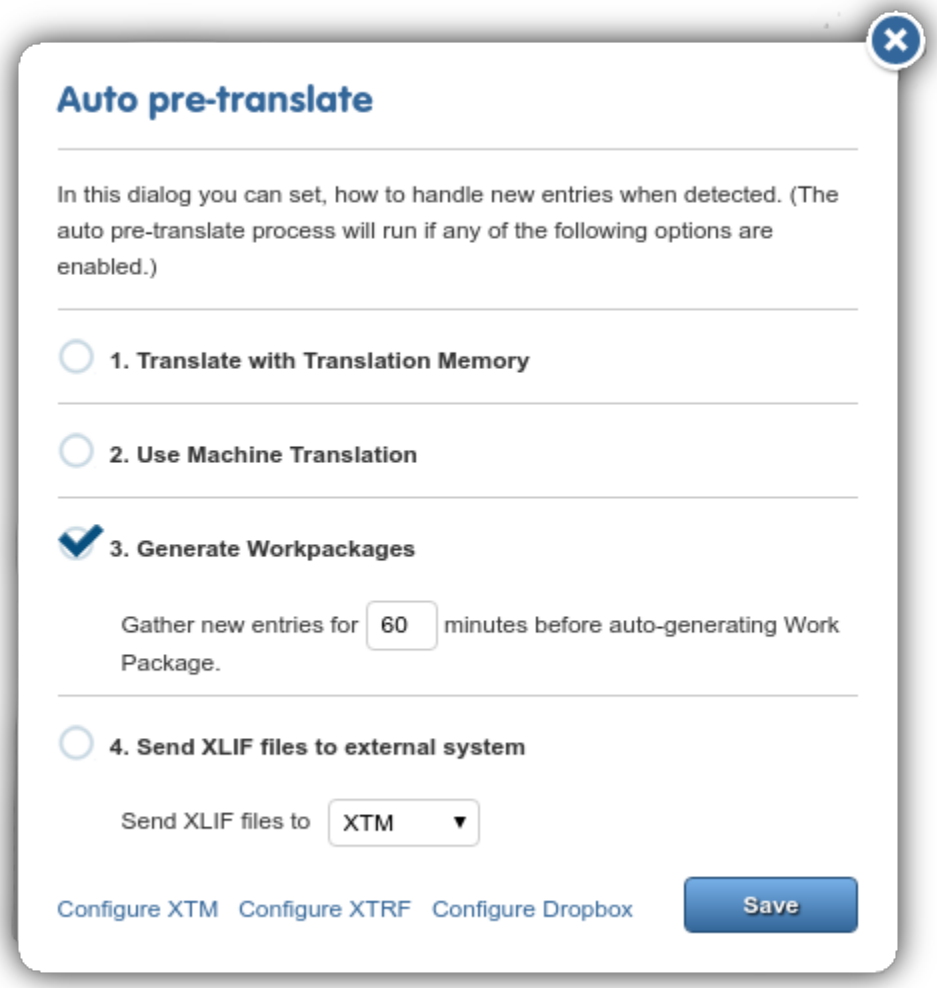


The process can automatically extract new content, apply the associated translation memories and machine translation services, prepare a work package and the XLIFF export. This is particularly useful for fast-moving sites where content arrives quickly and time spent untranslated needs to be minimized (possibly at the expense of real-time human oversight).

Automatic pre-translation

Easyling can automatically pre-translate incoming content without user intervention or oversight, feeding its translation engine from saved translations (from an the translation proxy Translation Memory) above a certain confidence level or using configured machine translation engines (Google Translate, Bing Translate, iTranslate4EU, and GeoFluent, currently).

If new content is encountered, and at least one source is configured, a user-configurable timer starts counting down. Content is collected during this timer, and automatically translated using the configured sources. At the end of the configured window, any content that cannot be translated with the assigned sources (no matches of the desired confidence are found in the TM or the MT-engine returns no translation) is packaged into an auto-generated Work Package and exported into an XLIFF file (being pushed directly to XTM, if an integration is configured). The resulting export can then be translated in any external system and imported back normally.



Caching And The Proxy

Overview

Nearly all browsers today implement local caches to accelerate page loading and prevent unnecessary requests from being sent out to the network. However, the operation of these caches is tied to the presence of certain headers on the page, such as `Pragma` and `Cache-Control` - based on their presence and the values communicated in these headers, the browser (and various systems, such as CDNs) may make a decision to intercept the request and serve up certain content without requesting it anew from the server.

Normally, the Proxy simply forwards these headers, much the same way it does with any other. The option to override their presence and values exists (see the Path Settings option on the Dashboard), but by default, they are left unmodified, in the spirit of minimum invasion. This is not always desirable, however, as a site without such cache headers will remain uncached in the visitors' devices, and each visit to the page will result in another request that is billed.

Inspecting Cache Headers

You can investigate how well a site may be cached using the Developer Tools in most major browsers. In Chrome, for instance, the DevTool can be summoned using by pressing the F12 key (or Alt+Cmd+I on under Mac OS), and after refreshing the page, the Network tab can be used to browse traffic associated with the tab. By selecting any entry in the list, you can view its details, in particular, the request and response headers. To tell whether or not a given resource will be requested again, you need to look at the "Response Headers" section, and look for the keys `Cache-Control` and `Pragma`.

If you see `Pragma:private` and/or `Cache-control:no-cache`, it is safe to say that the given resource will not be cached and each visitor will result in another hit. Files like this will likely prove resource drains if the site receives large amounts of traffic. On the other hand, `Pragma:public` and `Cache-Control:public,max-age=\d+` (where `\d+` means at least one digit, or more) are good signs in that these files will be stored on the client's device after the first request, and will not be requested until `max-age` seconds have elapsed since the last load, and will save resources in the long run. Of course, this also means that visitors may be seeing an "outdated" version of the resource for a limited time before their caches expire and are reloaded.

There is a bit of a gray area when seeing `Cache-Control:must-revalidate`: this directive allows the cache to make the final decision based on its own algorithm, and the response **may** be stored, but not necessarily. When seeing this it is good to prepare for potential increased traffic, as the browser cache may or may not retain these responses.

What to take away

If you're experiencing consistently large numbers of page views on a given project, it is often a good idea to suspect caches (as opposed to search bots, which cause transient spikes). In such cases, you should inspect a few pages using the DevTool, and determine whether or not the site is set up to take advantage of browser caches.

If it turns out be the case that the site is not set up to use caches, the best course of action is to notify the owners - perhaps they have a good reason for it. In any case, these headers should be added to the source, so that they provide consistent values across all languages. Alternatively, you can use the aforementioned Path Settings dialog to force the Proxy to override the cache headers and make the site cacheable, at the risk of diverging from the original, even if only for a short time.

CHAPTER 7

Technical Reference

Architecture

Modularity

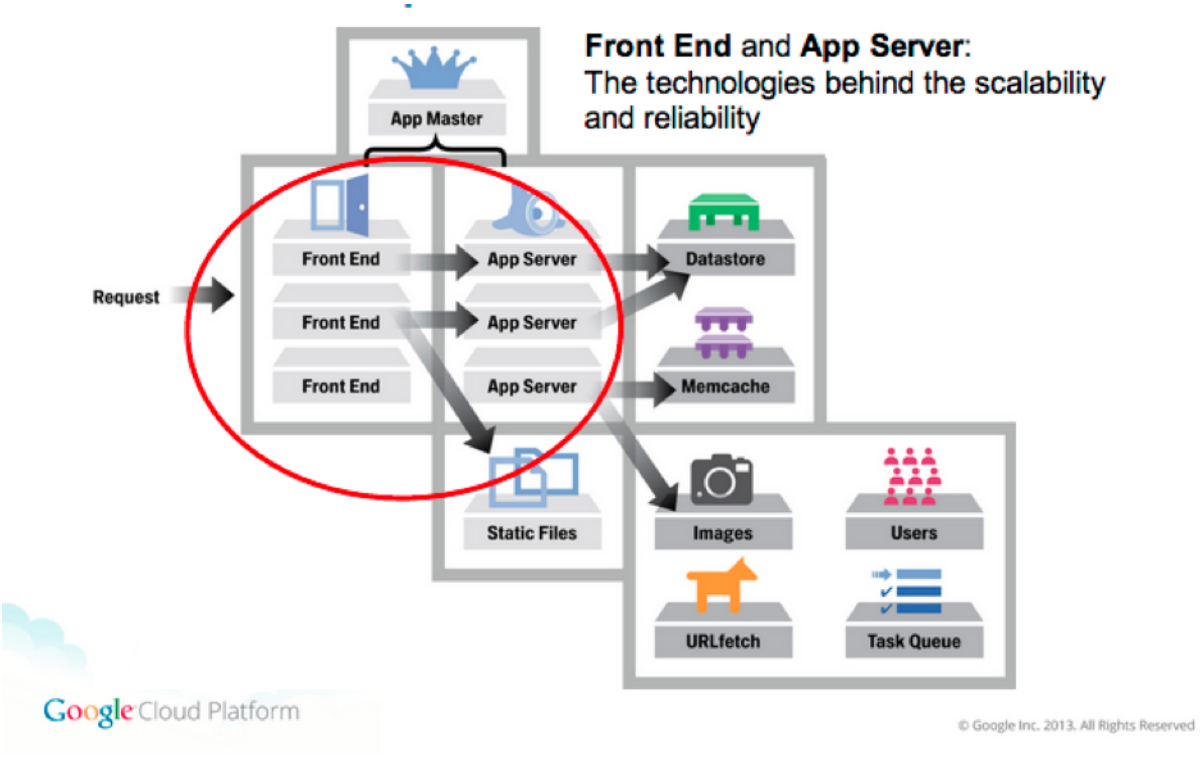
The Translation Proxy is based over Google's AppEngine infrastructure, split into frontend and backend modules. Each module encompasses variable numbers of instances, scaling automatically in response to demand. Modules are versioned and deployed separately, and can be switched independently, if needed.

Frontend instances serve requests from visitors to translated pages (in addition to serving the Dashboard and providing user-facing functionality).Requests are routed to the Proxy Application via the CNAME records created during the publishing process.

Backend modules are responsible for billing, statistics aggregation, and handling potentially long-running tasks, like XML import-export. Backend instances are not directly addressable, and provide no user-facing interaction.

Underlying technologies

Immediately underlying the Proxy Application is the AppEngine infrastructure, responsible for rapidly scaling the deployed application. AppEngine also handles communication with the Google Cloud Datastore, a high-replication NoSQL database acting as the main persistent storage; as well as the Google Cloud Storage system, an also-distributed long-term storage. Logging is provided by Google Cloud Logging, while BigQuery provides rapid search ability in the saved logs on request.



Encompassing the entire application is the Google EdgeCache network, proactively caching content in various data centers located regionally to the request originator. Any content bearing the appropriate headers (`Cache-control:public; max-age=/\d+/` and `Pragma:public` - both are required) is cached by the EdgeCache for as long as needed, for requests originating in the same geographic area.

The current instance of the Proxy Application is hosted in the US central region, as a multi-tenant application (serving multiple users, but dedicated to proxying). However, single-tenant deployments (dedicated to a single user), special deployments to other regions (EU or Asia), or internal systems fulfilling the AppScale system requirements can be discussed on a per-request basis.

Request Handling

In the Translation Proxy, frontend instances are responsible for serving translated pages. Thanks to AppEngine's quick-reaction scaling, the number of frontend instances aggressively follows (and somewhat predicts) demand, keeping latency low. The general life cycle of a proxy request can be described as follows.

- The incoming requests, based on the domain name, reach the Google Cloud (rerouted via DNS record CNAME, pointing to `ghs.domainverify.net`.
- Based on the domain name and the deployed Proxy application, AppEngine decides that this specific request should be routed to the Proxy AppEngine deployment.
- The request reaches the Proxy Application internally; the application does a lookup against the domain for the associated project. There are special domain names, and the final serving domain, for which caching is activated.
- Based on the URL, the Proxy application determines the matching Page in the Proxy database. The database has a list of segments, pointing to our internal Translation Memory (TM). We retrieve all these existing Database entries, including the translations for the given target language.
- The Proxy application processes the incoming URL request, and transforms it to point back to the original site's domain. Then, the source content of the translation is sourced, according to cache settings in effect on the

project.

  - If source caching is *disabled*, the application issues a request, and retrieves the result from the original web server, which is hosting the original website language.
  - If source caching is *enabled*, a local copy (a previously stored version of the source HTML) is used, instead of issuing a request to the original web server.

- Depending on the `Content-type` of the response, the appropriate `Translator` is selected, and the response is passed to an instance of the `Translator` as a document. The behavior of the `Translator` can be affected by cache settings as well.
  - If binary caching is *disabled*, the application then builds the Document Object Model (DOM) tree of the result, finally iterates through all the block level elements, and matches them against the segments loaded from the database. If there's a match, we replace the text with the translation. If not, we 'report' it as a missing translation.
  - If binary caching is *enabled* and the hash of the source HTML *matches* the one stored in the cache, a previously prepared and stored translated HTML is served.
  - If binary caching and keep cache are *both enabled*, and the hash of the source HTML *doesn't match* the one stored in the cache, the proxy translates the page using the TM. If the number of translated segments is higher than the previously prepared and stored translated HTML, the new version is served; otherwise the old one. (Keep cache can be thought of as a "poor man's staging server").
- Hyperlinks in the translated content are altered by the `LinkMapper` to point to the proxied domain instead of the original. This affects all `href` or `src` attributes in the document equally, unless the element is given the `__ptNoRemap` class. At this point, resources may be replaced by their localized counterparts on a string-replacement basis.
- The application serializes the translated DOM tree, and writes it to the response object.
- Before final transmission takes place, the Proxy may rewrite or add any additional headers, such as `Cache-control` or `Pragma`.
- Finally, the Proxy serves the document as an HTML5 stream, as a response to the original request. AppEngine *must* close the connection once the response is transmitted, so proxying streaming services is not possible in this fashion!

Classification of Content

The Translation Proxy distinguishes two main types of content: text content and resources. The key difference is that text content may be translated, while resource content is treated opaquely, and only references can be replaced as resource localization. It is possible to reclassify entities from Resource to Text, but not the other way around.

During proxying, resources are enumerated, and any already-localized references are replaced, while text content is passed to an applicable `Translator` implementation for segmented translation.

Text content

By default, the Proxy only handles responses with `Content-Type:text/html` as translatable. To process HTML content, the source response's content is parsed into a Document, then text content is extracted from the DOM-nodes. Additionally, various attribute values are processed (without additional configuration, `title` and `alt`).

The content is then transformed into `SourceEntry` entities server-side. Each block element comprises one source entry, with a globally unique key. If segmentation is enabled on the project, the appropriate rules are loaded (either using the default segmentation or by loading the custom SRX file attached to the project), and the content is segmented

accordingly, with the resulting token bounds stored in the `SourceEntry`.Along with the `SourceEntry` entities, the corresponding `TargetEntry` and `SourceEntryTarget` entities are created. `TargetEntry` entities, as the name suggests, hold the translations; `SourceEntryTarget`s act as the bridge between the two, and hold the segment status indicators for both.

The content of source entries is analyzed in the context of the project, and statistics are computed. These statistics include the amount of repeated content at different confidence levels based on the similarity of the segment - The Translation Proxy differentiates five levels of similarity:

1. 102%: Strong contextual matches: every segment in the block level element (~paragraph) is a 101% match, where all the tags are identical. These matches do not result in the creation of new `SourceEntry` entities, thus changes in one place are propagated instantly to all occurrences.
2. 101%: Contextual matches: both tags in the segment, and contexts (segments immediately before and after) match.
3. 100%: Regular matches: the segment is repeated exactly, including all tags.
4. 99%: Strong fuzzy matches: tags from the ends are stripped out, words lowercased, numbers ignored.
5. 98%: Weak fuzzy matches: all tags are stripped out (may have to be adjusted manually afterwards!), words lowercased, numbers ignored. If the Proxy cannot match the tags between the translation and the source due to excessive differences, all tags are placed at the end of the segment, requiring manual review!

These classifications are reused during memory-powered pre-translation in order to select the best applicable translation or to propagate existing translations.

Resource content

By default, any content with content types other than `text/html` are treated as a resource, and is not a candidate for translation, only replacement en bloc. This mainly includes `application/javascript`, `text/xml`, and various `image/*` content types. Every resource can be given different replacements per target language, and if required, certain resources (`application/javascript` and `text/xml`) can be made translatable after pre-configuration is done. In this case, instead of references being replaced, the appropriate `Translator` will be instantiated and the content passed to it. This can enable partial or complete translation of dynamic content transmitted as `JSON` or `XML`.

Translation Memories

The Translation Proxy can be configured to maintain and leverage internal translation memories. These memories can contain more than one target locale allowing leveraging them for any pair of locales contained within.

As opposed to project dictionaries, translation memories are keyed to the *user* creating them, and can be assigned to any project the user has Backup Owner privileges or higher.Any project can contain an arbitrary number of memories, but one must always be designated the default: only this memory will be utilized when segments are being written; while pre-translation and suggestions are fed from *all* memories assigned to the project with applicable locale configurations.

Using TMs

Translation memories are initialized empty, and must be first configured with locales. After the target locales are defined, the memory can be populated. There are three ways a segment can be injected into the memory:

- TMX-import: The Proxy can digest a standard `TMX` (Translation Memory eXchange) file and populate a designated memory based on its contents. The memory must be configured with at least one of the target locales of the `TMX` file. Duplicate segments are either merged (if for different locales) or discarded during import.

- Project population: The Proxy can populate the memory from the project it is currently assigned to. The memory must be configured with at least one of the *project's* target locales for this to work. If there are several locales assigned to the memory, the UI will treat them as a set, and offer the *intersection* of the memory and the project's locales as the default. This set can be further restricted by removing locales from the population task before committing it. This action is logged in the project's Audit Log.
- Individual Injection: If a memory is assigned to the project with at least one locale present on both, it will be available on the Workbench for use. Confirming one or more segments will trigger the `saveToMemory` action, injecting the segment in its current form into the memory.

Memories are used for two tasks on the UI:

- Pre-translation tasks can leverage any memories assigned to the project, provided the memory is configured with the correct locale. This applies to user-triggered Pre-translation, as well as Automatic Pre-translation triggered by new content. Only content with confidence levels above the user-configured threshold will be used, matches with lower percentages are discarded.
- The Workbench automatically leverages any memories with the appropriate locales on segment selection. Matches are displayed in the Suggestions tab of the sidebar, along with their match percentages. Additionally, all memories on the project with the applicable target segments can be queried at will by entering a search term.

Confidence levels

The Proxy differentiates five levels of similarity between individual segments/entries (see here). Memory application yields the best results between 101% and 99% - 98% matches disregard tagging, and may need manual adjustment. However, searching below 98% is also possible, using the Google Search API, but these matches should be used with caution, as there is no guarantee regarding their accuracy due to the Search API's word stemming.

Page modifiers

Due to the way the Proxy Application operates, it becomes fairly easy to modify the pages as they are being served. Because the datastream must pass through the proxy to have the translation embedded, the Proxy Application can insert JavaScript modifiers, modify style sheets, and even embed entire pages that do not exist on the original.

- CSS Editor: the Proxy Application can be used to insert locale-specific CSS rules into the site being served. The rules are inserted as the last element of the `head` on every page served through the proxy. The most common use of this feature is to alter the writing direction for non-Latin scripts, such as Arabic.
- JavaScript Editor: the JavaScript edited here is inserted into the `head` element of every page being served through the Proxy Application. As the last element of the `head`, it has access to any global variables added by scripts before it.
- Content Override: the Proxy Application can create a "virtual" page in the site or override an existing one with custom code. For any requests to an overridden page, the corresponding remote server request is not sent, and the override contents are used as the basis of the translation. The source is not required to be HTML, custom content-types can be entered, along with customized cache headers, and status codes (HTTP status codes are restricted to those permitted by the Java Servlet class!) - note that the 300-family of status codes requires the `Location` header to be defined as well.

Both the CSS and JavaScript injectors can use already-existing files for injection instead of copied content. The injected files must be handled by the project in some way (either by being in the project domain, or in the domain of a linked project), or be created by a content override. The order of definition for these entries matters, as they will be inserted into the document in the order they are displayed on the UI, which may cause dependency or concurrency issues!

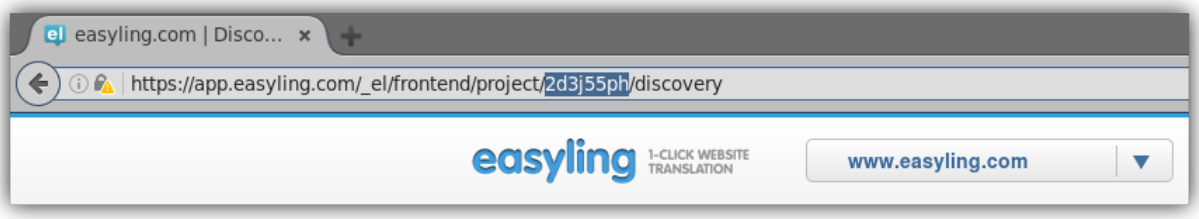
CHAPTER 8

Troubleshooting & Support

Contacting Support

Website translation can be a complex task, even with the help of a piece of software like the proxy. Finding the root of the problem can be equally complex for those of us on support duty. Therefore, we would like to give you a few pointers on what information to supply if you decide to contact us.

This first thing we need is the **project code**. This eight-character unique string uniquely identifies the project. As you can see in the screenshot below, the project code is located in the address bar of your browser - you can copy the entire URL for us, but the project code by itself should suffice.



We also need a **thorough description of the problem**. Screenshots are tremendously helpful, especially if you have layout issues on the translated site. If you're running into translation issues, please give an example segment along with the page link it can be found on.

If you have issues with importing XLIFF or TMX files, please attach them so that we can take a look. If you have questions about statistics, reports, or crawl logs, attaching or linking them in your query will considerably speed things up for us.

The information you provide will help us uncover the root cause of the problem. This often requires a bit of "detective work" on the original site, so we ask for of your patience while we figure things out. Someone from the support team will respond shortly with a solution, a request for more information, or simply an update.

Issues

Scanning Content Behind Secure Login

To scan content behind secure login, please follow this procedure:

1. Open your project and navigate to the **Content** menu.

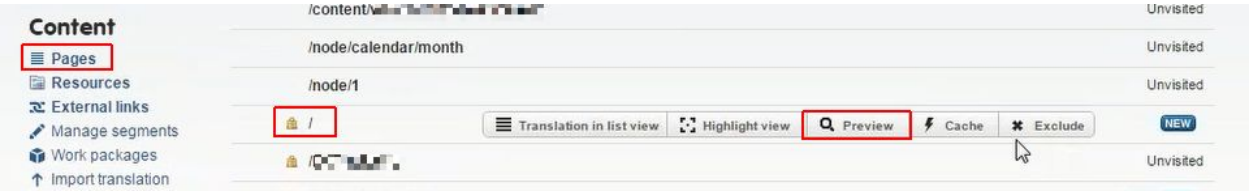
2. Open the **Pages** list.
3. Visit the page with the login, if it is listed, and click **Preview**.

   OR

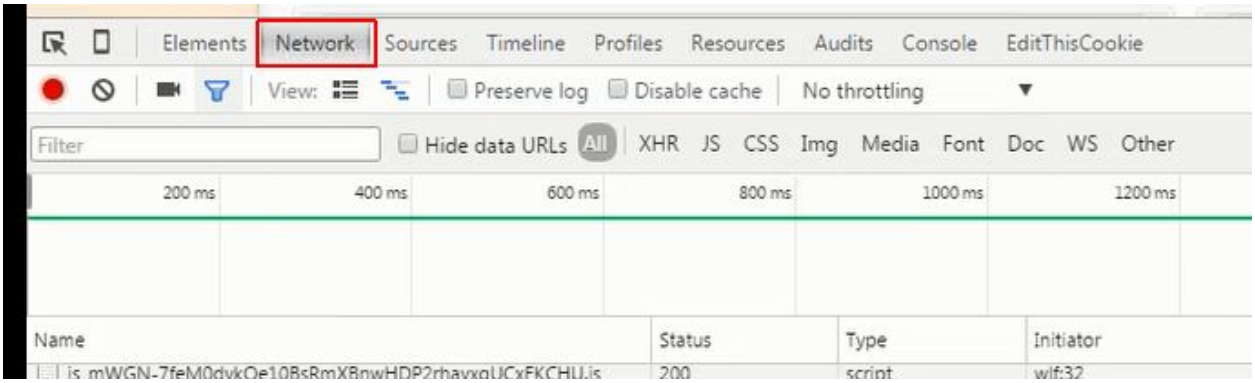
   Go to the **Preview** of the front page (the "/", the first one on the Pages list). It will give you the front page through the proxy.

   OR

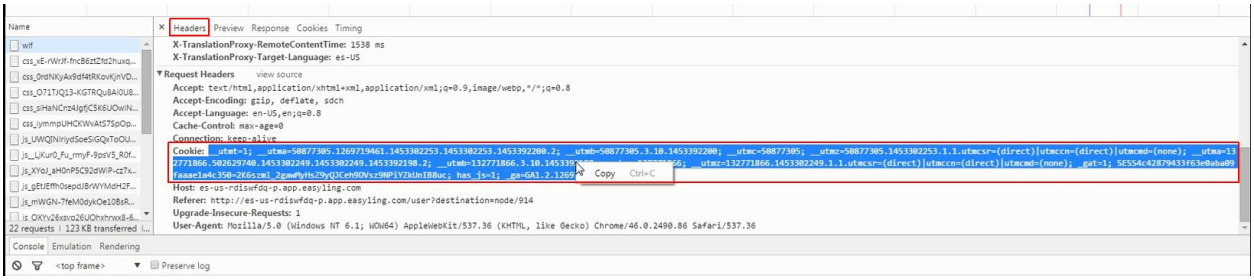
   Go to the address bar and type in the URL of the login-protected page.



1. Enter your login details.
2. Open your browser's DevTools from the Menu (F12 on Windows).
3. Go to** Network** and reload the page.



1. Scroll up to the first item and click on it.
2. Under headers scroll to the cookie header (among request headers), and copy the entire header.



1. Pass it to the Proxy: go back to your project and click on **Content**. Paste the entire content in the **Scanning cookies** field.



1. Click on **Scan manually** and specify the required scanning settings. You will receive an e-mail notification once scanning is ready and new content is available for translation.

XLIFF import error

When you import your translated XLIFF file back to easyling, you will receive an e-mail notification when the process is ready. This mail contains the URL of the import log, and an overview of the log entries:

- Error:
- Warning:
- Info:

If you see other than '**Error: 0**' in your notification mail, the XLIFF file needs fixing. Usually these are tag placement errors that can be easily fixed in a text editor like Notepad++ or Sublime (the ones that have syntax highlighting, to make this fixing job easier), yet they do need attention, as the corresponding translation will not show up on the website.

- Open the XLIFF in your text editor
- Open the log file and check the error message(s)
- Do the necessary corrections in your XLIFF (see 'Troubleshooting')
- Save & upload the corrected XLIFF

In very serious cases the import might fail completely, but this is very rare. These cases include: attempt to upload an XLIFF related to another project, XLIFF with target language that doesn't exist in the project, and fully invalid XML in the file. In most cases the file is imported, only the faulty entries are omitted.

Please note that you need XLIFF files. Ideally, the export format of the CAT-tool should be the same as the import format, and as you import an XLIFF file for translation, the output should also be a standard XLIFF file. However, some versions of Studio tend to create an **SDLXLIFF** file upon exporting the translation. In this case, simply use the "**Finalize**" batch task or open the document in the **Editor**, press **SHIFT+F12** and select the target file location. This will create the XLIFF file for you (instead of SDLXLIFF).

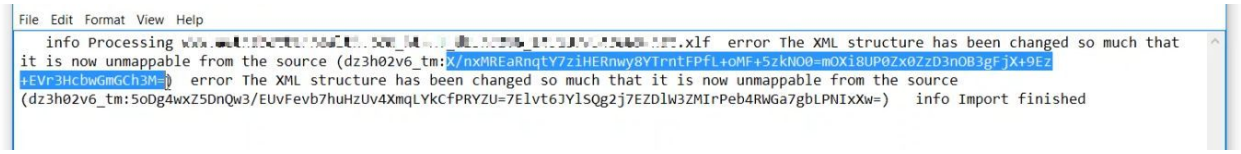
You might also need to disable segment info storage in Studio (***Options -> File Types -> XLIFF -> Settings -> 'Do not store segmentation information in the translated file'*** should be checked). This may require creating a new project.

Troubleshooting

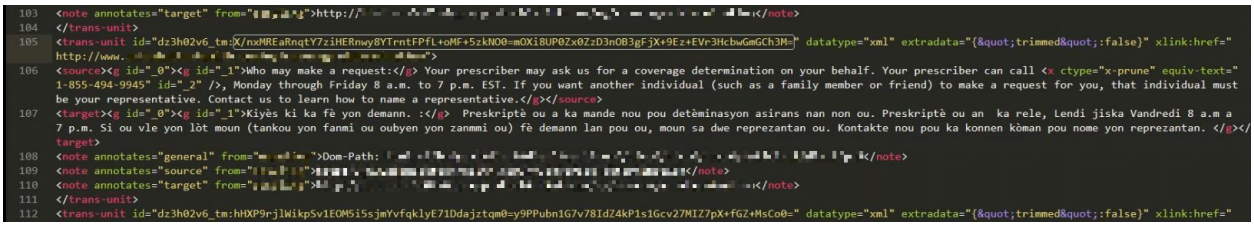
Error: *The xml structure has been changed so much that it is now unmappable from the source*

Fix:

1. Open both the **XLIFF file** and the **error log** in a text editor.
2. Select & copy the ***TM-key*** of the faulty entry, the part after '(trans-unit id="xxxxx)_tm:' in parenthesis right after the error message



1. Search for this key in the **XLIFF file** by pasting it into the 'Find' field. Only 1 translation unit will match.



1. Compare the tags in the source and target languages, and fix the mismatch by editing the **target text**. (You can also use an online text comparison tool for this task: copy-paste `<source> ... </source>`in one pane and `<target> ... </target>` in the other one.)
2. Save the corrected XLIFF file and upload again. It should give no error message now.

OR, alternatively,

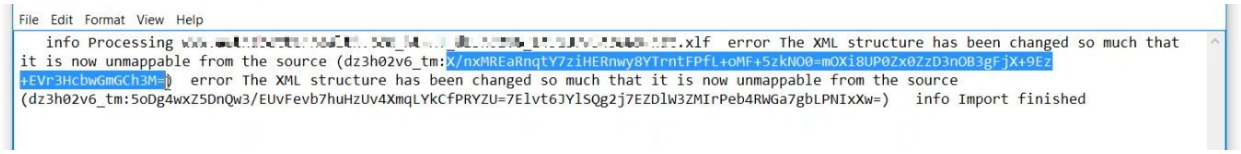
1. Go back to your CAT tool, where you did the translation and open the faulty file for editing
2. Run QA. It will list you all the tag mismatches
3. Navigate to the faulty segment(s) and fix the tags
4. Export the corrected file and upload it. It should give no error message now.

Error: *Content found outside of outermost element*

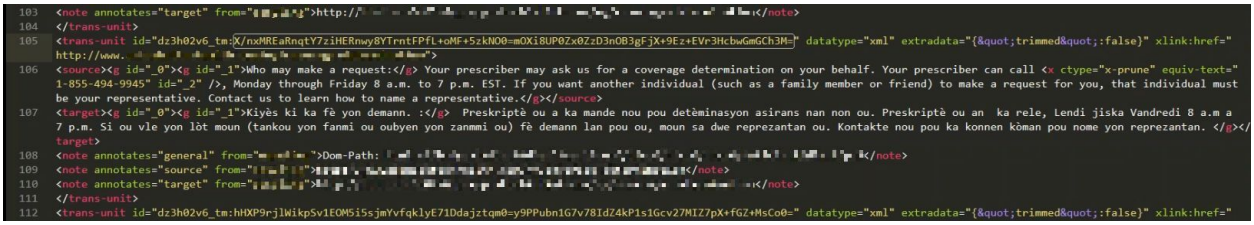
Fix:

Practically this means that there is an extra space before the starting `<target><g*>`or after the closing `</g></target>`.

1. Open both the **XLIFF file** and the **error log** in a text editor
2. Select & copy the ***TM-key*** of the faulty entry, the part after '(trans-unit id="xxxxx)_tm:' in parenthesis right after the error message



1. Search for this key in the **XLIFF file** by pasting it into the 'Find' field. Only 1 translation unit will match.



1. Delete the extra space around the tags
2. Save the corrected XLIFF file and upload again. It should give no error message now.

OR, alternatively,

1. Go back to your CAT tool, where you did the translation and open the faulty file for editing
2. Run QA. It will list you all the tag mismatches
3. Navigate to the faulty segment(s) and fix the tags
4. Export the corrected file and upload it. It should give no error message now.

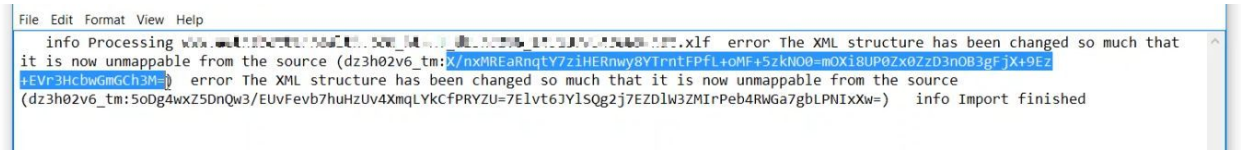
IMPORTANT! Most of these issues can be avoided if the QA parameters of your CAT tool are set up properly, and you run QA before exporting your XLIFF files. Please make sure to check your translation for tag consistency and extra spaces; these are critical errors in website translation that can spoil the code.**

Error: *Illegal character*

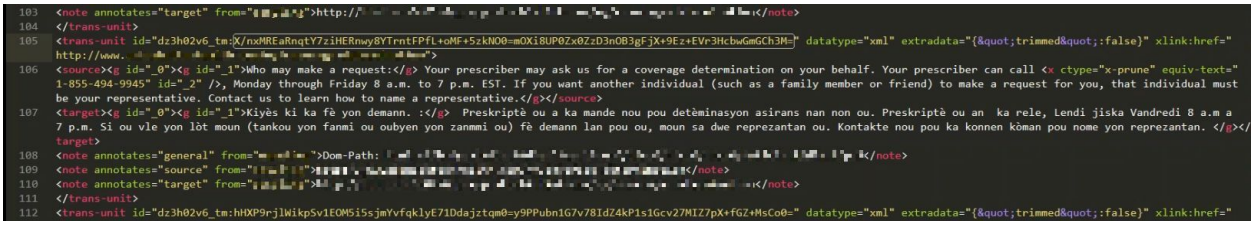
Fix:

The reason for this error is usually a coding mismatch: all our XLIFF files are exported using the world-standard UTF-8 codepage. However, your CAT tool may save the file using another codepage, depending on the language, which may cause certain characters to appear as invalid.

1. Open both the **XLIFF file** and the **error log** in a text editor
2. Select & copy the ***TM-key*** of the faulty entry, the part after '(trans-unit id="xxxxx)_tm:' in parenthesis right after the error message



1. Search for this key in the **XLIFF file** by pasting it into the 'Find' field. Only 1 translation unit will match.



1. Check if you see any strange characters, like squares or other meaningless characters.
2. Go back to your CAT tool and change the export options to use UTF-8 encoding. (As UTF-8 is a universal standard, it should be available.)
3. Re-export the XLIFF with UTF-8 encoding and upload it. It should give no error message now.

Publishing issues

Translated page doesn't show up

Issue: *I've just uploaded the translation of some new pages, but they don't show up on the translated site / they still appear in the source language.*

Fix:

You experience this problem because Target cache is enabled, and you need to clear the Cache to update content to be served. The very reason for using a Target cache is to mask unfinished translation and avoid bleed-through. The Target cache shows the last fully translated version of the pages - so if content changes on the original page, it remains hidden on the translated site until a fully translated version is available.

To fix the issue you need to explicitly delete the page from the Cache, so that the updated content could be loaded in upon the very first viewing of the page.

If you have several languages, it might be more convenient to clear the entire Target cache by clicking the Trash icon.

Translated page is not listed in Google search results

Issue: *The pages served from the Proxy are not listed in Google search results.*

Fix:

Google has never indexed the site in the first place, most likely due to the fact that there are no "hreflang" links on any of the pages, so the Googlebot has no idea there are other pages to look for. More information on the element and how it affects Google rankings may be found at https://support.google.com/webmasters/answer/189077?hl=en

Additionally, creating and submitting a sitemap (more information at https://support.google.com/webmasters/answer/2620865) to Google in order to force an indexing of the pages can also help. Even so, without the hreflang attributes, it may mean that some penalty is applied to the rankings, due to perceived duplicate content.

Using the `hreflang` Element

Google's CrawlerBot will eventually find your translated page if there are any links to it. However, if the content there is not marked appropriately, it will not be given the same SEO scores as your main content. In fact, it may even be treated as duplicate content, and a scoring penalty may be applied.

To prevent this from happening, you need to provide the GoogleBot with information on how the translated sites relate to the original. The easiest way to do this is the `<link rel="alternate" hreflang="" href="">` element.

These elements have to be placed in the page head (i.e. before the HTML body), and have two rules that must be satisfied in order for the GoogleBot to consider them:

- `hreflang` elements must be reciprocal: if a link points to a translated site, the translated site must point back to the original as well.
- `hreflang` elements must be circular: each language must also refer to itself with a link.

Consider the following HTML snippet from an imaginary site at `http://example.com`:

```
<html>
<head>
<title>Title Here</title>
<link rel="alternate" hreflang="en" href="http://example.com" />
<link rel="alternate" hreflang="jp" href="http://jp.example.com" />
```

```
</head>
[...]
```

and its translated counterpart at `http://jp.example.com`:

```
<html lang="ja-JP">
<head>
<title>Title Here</title>
<link rel="alternate" hreflang="en" href="http://example.com" />
<link rel="alternate" hreflang="jp" href="http://jp.example.com" />
</head>
[...]
```

This snippet will provide proper SEO, since it satisfies both criteria: the references the English and the Japanese site are reciprocal (one refers to the other and vice versa) and the references are also circular (both languages also refer to themselves as well as their counterparts). This provides all the information GoogleBot needs to index each site in its rightful place and apply SEO scores across both domains.

For more information on this topic, see this article from Google: https://support.google.com/webmasters/answer/189077?hl=en

WPEngine issues

Redirections during crawling

Due to WPEngine's *caching system* and *Redirect bots* settings you might experience any of the following issues on your **WPEngine hosted sites**:

1. Scan extracts outdated content from WPEngine cache
2. Scan returns **301: Moved permanently** error message for existing pages
3. Translated page is redirected from HTTPS to HTTP - which results in an error due to mixed content

WPEngine caching uses different so-called *buckets* based on request type, and they have one for bots. If the request comes from `Google` or other listed user-agents, and/or the URL has `?ver=` followed by a number, the\ redirect bots settings take effect.

The above issues can be resolved on the WPEngine site by turning off the *redirect bots*.

Intermittent `HTTP403` on proxied pages

WPEngine automatically blocks traffic from "problem" IP addresses, typically those that generate large amounts of traffic in a short time. Due to the nature of the proxy, requests from several users may appear to have come through one IP, leading to WPEngine blocking that node due to their perception of "increased traffic".

If that happens, and you note random pages of the proxied site not loading intermittently, call or chat WPEngine Customer Service with the following:

> In relation to issue #874002, please enable proxy access on our installation.

According to an agreement with WPEngine, they will enable an alternate method of IP resolution that should no longer prevent access to the translated pages.

Captchas

Captcha doesn't work on the translated site

This issue is most likely results from page caching. Certain captcha solutions, like WP plugins are hardcoding the image URL into the HTML, instead of sending it asynchronously. During the crawl that builds up the Source cache for the project one hash is saved, so it beco\ mes static, while it should be changing on each occasion. As a result, the server rejects the request because of the outdated verification image.

Fix:

- disable caching altogether to make captcha work
- use caching without captcha
- use another CAPTCHA system that uses async requests to retrieve the verification image, like Google's Re-Captcha solution

Another possible cause can be the CORS-header. If the proxied page is not listed as an allowed origin, the browser blocks the page when it tries to load the image.

CHAPTER 9

FAQ

General

Where are my translations published?

Instead of "where?", a better question to ask is "how?".

Imagine the proxy as standing between the original site and a visitor's browser. Publishing the Japanese translation of example.com on the jp.example.com subdomain means mapping jp.example.com (presumably owned by the owner of example.com) to point to the proxy.

Visiting jp.example.com/contact.html results in that request being caught by the proxy and relayed to example.com/contact.html - the origin server. The contact.html page is served as a response, which is caught on the way back, translated on-the-fly at blazing speeds and served to the visitor.

This requires that jp.example.com be mapped to the cloud proxy application in the owner's DNS settings.

Does the proxy host a copy of my site?

No. The proxy does not store any copies of the original site pages, it only stores translations, which it uses to process responses served by the original site to visitor's queries.

There is one exception to this principle: if a source cache is built and enabled for a proxy mode, that cached version of the page will be used in place of the origin server's response.

Some parts of a site are on a subdomain. How will the crawler pick them up?

The sites www.company.com and blog.company.com are treated as separate domains by the crawler. From the vantage point of a crawler running on www.company.com, a path on blog.company.com is an external and will be treated as such. The solution is to create two separate projects and link those with each other.

The Discovery went beyond the limit I set. Why?

A crawl will finish the current round and visit the redirects and links on the last page. If it took the limit too literally, that could potentially result in trailing links being thrown out.

Can I get page-specific repetition statistics?

Repetition statistics make the most sense in a site-wide context. The problem with controlling calculations on a per-page basis is that it is not true to life to call a segment on a given page a "canonical instance". Take a navigation bar or a footer, for example. It will be "repeated" on all pages, but it cannot be said to "belong" to any one of them. the translation proxy stores the first instance it comes across and then propagates its translation to all other instances.

The page I'm trying to translate has prices. What can I do to handle local currencies?

The prices themselves can be made translation-invariable, but real-time price handling for different currencies will have to be implemented by the client on the source site, making it possible for the proxied site to access the locale-specific information. Pricing of products and services also has legal / market implications that are beyond the tasks of LSPs. Of course, once currency-specific information is accessible from the original site, we are happy to help with integrating any backend calls / ajax requests on the proxy.

How do I enable automated scan on my project?

To enable automated content extraction on your site, please go to Content, and choose either of the daily, weekly or monthly options in the drop-down next to the Look for changes option.

Is it possible to set up automated scanning behind secure login?

No, scanning can't be automated behind secure login. For such processes you need to extract cookies with you browser's dev. tools and pass them on to the proxy. Some cookies get invalidated over time, and we don't store cookies either.

What do the various tags mean next to each page in the page list?

See the Glossary for a detailing of the various tags encountered in the page list.

Caches

Can I preview newer content on the workbench without causing bleedthrough on the published site?

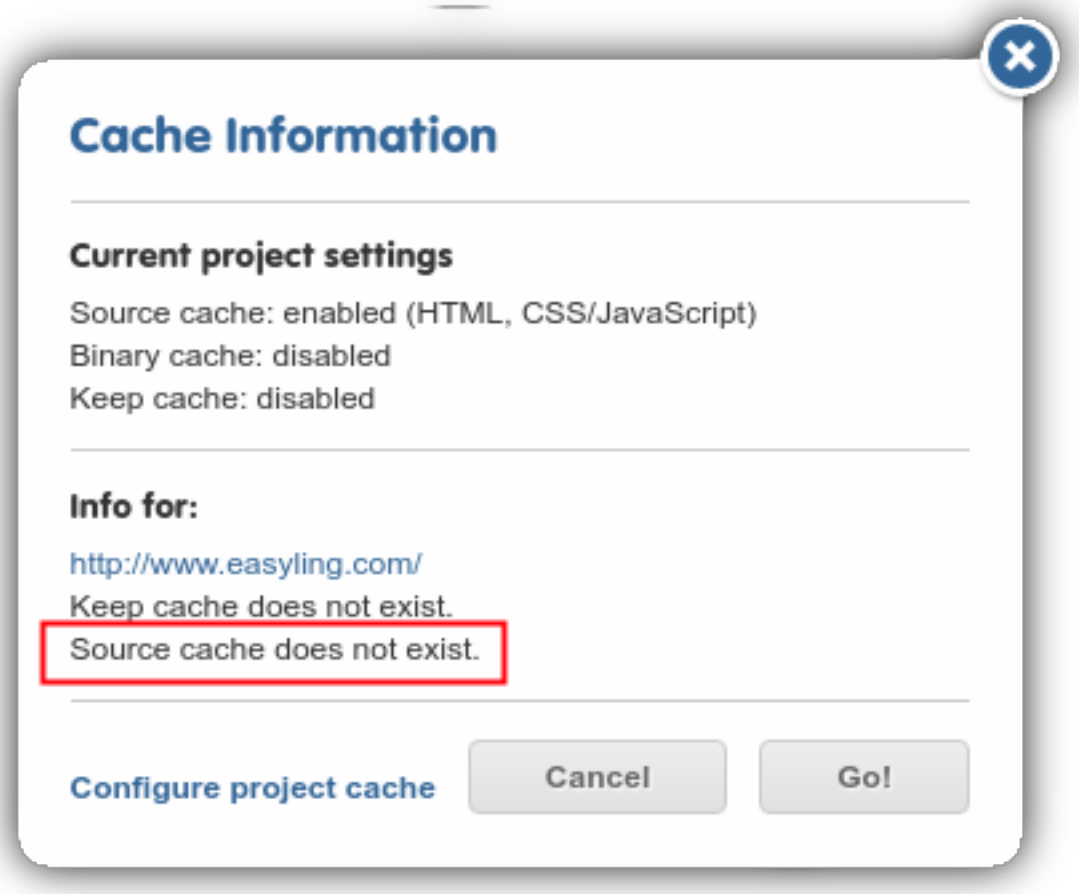
You can customize which Source Cache to use on which proxy mode - go into Page Cache, choose custom settings and select Disabled from the dropdown menu. The preview mode will display all new content. It is recommended that you keep TM Freeze turned on while exploring the new content, otherwise everything will be automatically added to the Workbench.

Does building a Source Cache cost any money?

You can use Content Scan with the appropriate options checked to build your Source Cache. As long as there is no new content to pick up, Scan costs the same as a Discovery.

How can I check if a page uses the Source Cache?

Go into **Pages view** in the Dashboard. If you hover with the mouse over a page in the list, you will see a **Cache** button. Click on it to verify Source Cache information for that page. If there is no Source Cache for that page, you will see the following screen:



Does building a Target Cache cost any money?

Setting aside the inherent cost of the Page Visits you have to accrue to build them, Target Caches are free of charge.

CHAPTER 10

Glossary

We use a set of recurring terminology in this manual and also in our Support Channels - we collect them here for your reference.

Page List Tags