Prequery

Extracting metadata for preprocessing from a typst document, for example image URLs for download from the web.

v0.1.0 March 19, 2024

https://typst-community.github.io/prequery/

Clemens Koza

Typst compilations are sandboxed: it is not possible for Typst packages or documents to access the "outside world". This sandboxing has good reasons, but as a consequence certain tasks require more manual work than one may like. For example, if you want to embed an image from the internet in your document, you need to download the image using its URL, save the image in your Typst project, and then show that file using the `image()` function. Prequery offers a limited interface that makes it easier to automate tasks of this kind.

Contents

I Introduction

Prequery is a Typst package and command line tool for getting external information into Typst documents. It allows you to break Typst's sandbox in a controlled way, achieving things that in TeX would be done through "shell escape".

The Prequery package allows you to prepare your documents so that they provide information to tools outside the sandbox. The `prequery` CLI tool is one such tool that can process this information to prepare results for use inside your documents.

This manual only contains generated API docs for the Typst package. For other usage information, see the online book: https://typst-community.github.io/prequery/

II Module reference

II.a `prequery`


```
prequery(
  meta: any,
  lbl: label,
  body: content,
  fallback: content,
) -> content
```

This is the fundamental function for building prequeries. It adds metadata for preprocessing to the document and conditionally shows some fallback content when `fallback` mode is enabled.

The body may be given as a function, so that errors from the body don't let compilation fail when in fallback mode.

Parameters:

`meta` (`any`) – the metadata value to provide for preprocessing

`lbl` (`label`) – the label to give the created metadata

`body` (`content`) – function): the body to display; if a function is given, that function will not be called in fallback mode

`fallback` (`content` = `none`) – the fallback content to display when in fallback mode

```
image(url: string, ..args: arguments) -> content
```

A prequery for images. Apart from the `url` parameter, the image file name is also mandatory; it is part of `args` for technical reasons. Outside fallback mode, rendering this fails when the referenced image is missing; images need to be downloaded in a preprocessing step.

This function provides a dictionary with `url` and `path` as metadata under the label `<web-resource>`. This metadata can be queried like this:

```sh
typst query --input prequery-fallback=true --field value ... '<web-resource>'
```

Fallback: renders the Unicode character “Frame with Picture” (U+1F5BC).

Parameters:

`url` (`string`) – the URL of the image to be shown

`..args` (`arguments`) – arguments to be forwarded to built-in `image`

`fallback`: `state`

A boolean state indicating whether the document should display fallback content for elements requiring results from preprocessing. For example, when using prequery images, compiling the document before download the files during preprocessing will fail. If all prequeries used in the document work even before preprocessing, it is not necessary to turn fallback mode on.

When editing the document and adding images (or other preprocessed content), it is convenient to temporarily add a line at the top to switch fallback mode on:

```typ
#fallback.update(true)
```

When querying the document for preprocessing, this can be activated using `--input`:

```sh
typst query --input prequery-fallback=true ...
```