Technical specification for the feature: Testing of documents

This is the technical specification for the Testing of documents feature. The feature was created as part of an EU-funded project called We4Authors Cluster. All features, their technical specification and their video documentation are published on www.accessibilitycluster.com.

If you have any questions or comments, please do not hesitate to reach out to research@funka.com.

The technical specification contains several perspectives:

- the web interface of the feature, based on the experience of the end user or, when it comes to the testing-features, the web author;
- the code of the feature ensuring accessibility by default;
- a description highlighting the key elements of the code;
- reference to a video documentation;
- recommendations for implementation.

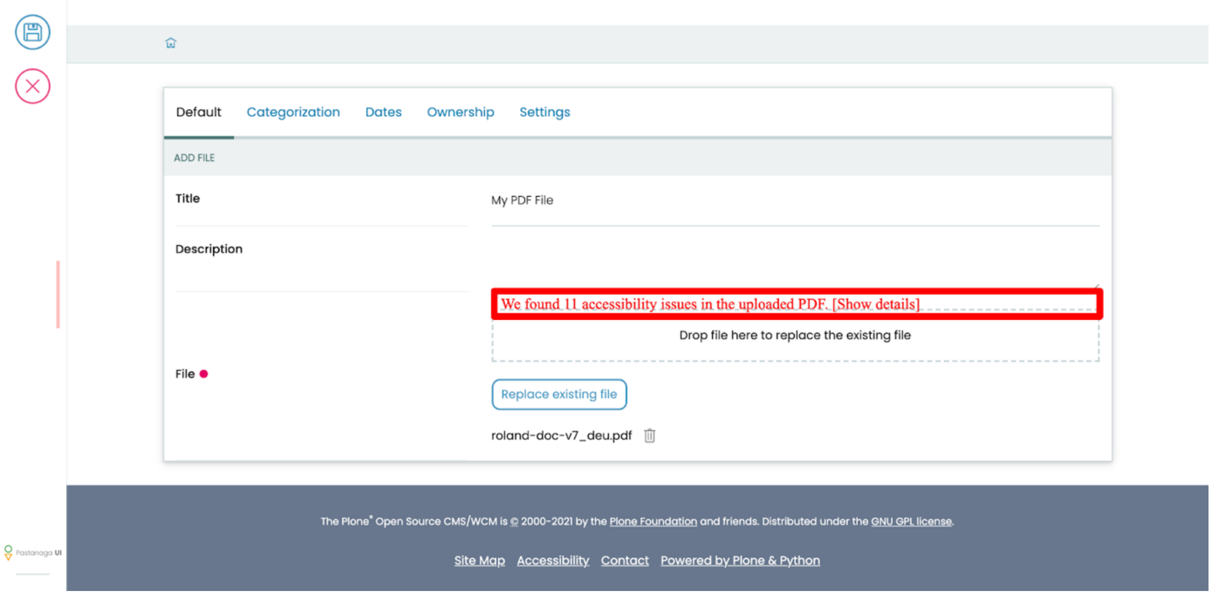
Specifications of the feature: Testing of documents

A feature testing documents before publishing them makes it easier for the web author to handle documents.

Automatic testing of document accessibility already exists on the market, both as external commercial products and built into the PDF-making software Adobe Acrobat DC. As with any automatic accessibility testing, they are not able to find all accessibility issues, but the automated tests do cover the key issues that may for example exclude screen reader users from the content.

The Adobe accessibility checker could potentially be included in the authoring tool by creating an API-based service that can be used to test the document before publishing it. This way, the author would be made aware about issues that can be automatically tested, such as tagging of the document, which is important for assistive technology to navigate the content. The automatic checker can also point to potential issues that have to be controlled manually, like colour contrast and reading order.

Web interface: web author view



ADD FILE

Title: My PDF File

Description

File ●

We found 11 accessibility issues in the uploaded PDF. [Show details]

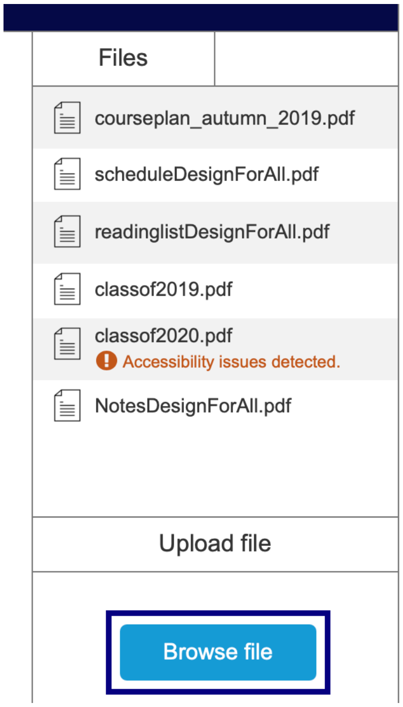
Drop file here to replace the existing file

roland-doc-v7_deu.pdf

The Plone® Open Source CMS/WCM is © 2000-2021 by the Plone Foundation and friends. Distributed under the GNU GPL license.

Site Map Accessibility Contact Powered by Plone & Python

Figure 1: Web author view - PDF checker



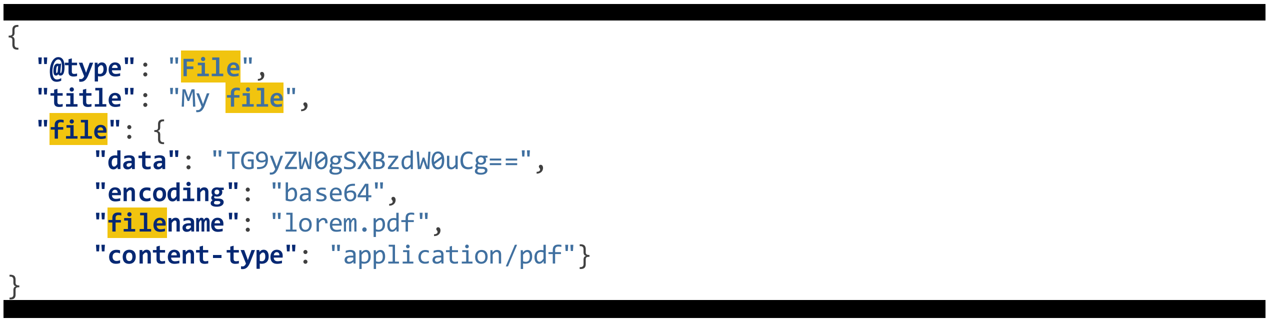
- courseplan_autumn_2019.pdf
- scheduleDesignForAll.pdf
- readinglistDesignForAll.pdf
- classof2019.pdf
- classof2020.pdf
  ! Accessibility issues detected.
- NotesDesignForAll.pdf

Figure 2: Web author view - warnings in the file library when detecting accessibility issues with a file

Guide for implementation

When a PDF file is uploaded it is asynchronously checked for accessibility issues. The system returns an accessibility status and a list of accessibility issues. This could happen via a separate REST response or with the REST response that uploaded the file.

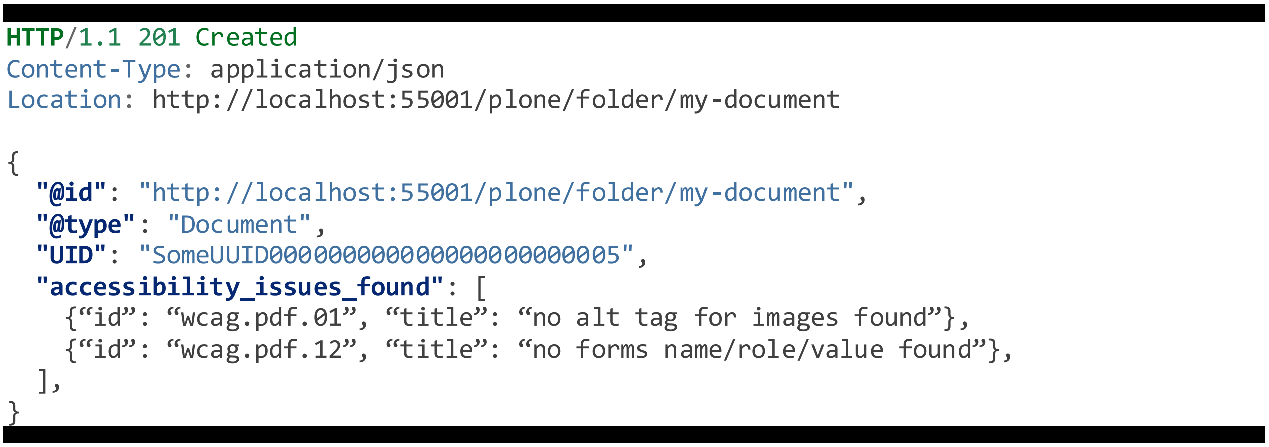
The client/browser sends a POST request to upload a file to the CMS:



```
{
  "@type": "File",
  "title": "My file",
  "file": {
      "data": "TG9yZW0gSXBzdW0uCg==",
      "encoding": "base64",
      "filename": "lorem.pdf",
      "content-type": "application/pdf"}
}
```

The actual request might differ depending on the authoring tool used. Most traditional authoring tools would most likely send a post request with x-www-form-urlencoded and not base64 encode it.

The authoring tool would reply to the file upload with a response that not only contains the regular response code (e.g. HTTP “201 Created” for a successful upload) but also a list of possible accessibility issues:



```
HTTP/1.1 201 Created
Content-Type: application/json
Location: http://localhost:55001/plone/folder/my-document

{
  "@id": "http://localhost:55001/plone/folder/my-document",
  "@type": "Document",
  "UID": "SomeUUID000000000000000000000005",
  "accessibility_issues_found": [
    {“id”: “wcag.pdf.01”, “title”: “no alt tag for images found”},
    {“id”: “wcag.pdf.12”, “title”: “no forms name/role/value found”},
  ],
}
```

Reference to an example of a library that could be used to implement such a backend:
https://gitlab.tingtun.no/eiii_source/pdfwam/-/blob/master/pdfwcag.py

Reference to a discussion on possible implementation options in Plone:
https://github.com/plone/volto/issues/1506

Video documentation

This technical specification has been developed in the We4Author Cluster project to reflect recommendations for accessibility features that can be implemented in any authoring tool. The specifications are complemented by a video documentation, covering a live description of:

- web author challenges, and
- feature solutions.

Recommendations for implementation

To make sure the implementation of the features is not causing accessibility problems for web authors with disabilities:

- avoid drag-and-drop for choosing template, and/or always have a keyboard alternative;
- always provide one pointer alternative without specific gestures;
- always support keyboard navigation;
- consider keyboard shortcuts;
- do not rely on sensory characteristics as the sole indicator for understanding and operating content;
- do not indicate important information using colour alone;
- make sure there is enough contrast between text objects and its background colour.