# Support for XML-based publishing in OJS

ISG Innovation Programme 2018/19
June 2019

## About the project

With funding from the ISG Innovation Programme, we ran a small project to look at how the journals we support could develop an XML-based publishing workflow using existing open source tools.

XML is used widely in academic book and journal publishing. It makes scholarly content machine-readable and layout-independent, more flexible and reusable for a variety of formats (PDF, HTML, EPUB, etc.), it offers improved searchability, accessibility and preservation, and allows text mining, and content enrichment through multimedia and semantic tagging. Humanities scholars have traditionally used TEI (Text Encoding Initiative) XML, while publishers increasingly use the NISO standard JATS (Journal Article Tag Suite, formerly the NLM DTD) and its extension BITS (Book Interchange Tag Suite). XML can be introduced at different stages of the production process, with XML-first, XML-last and XML-middle workflows.

Commercial publishers typically outsource the production of XML, or use expensive specialist software in-house, neither of which is an option for small-scale Open Access journals with limited funds.

As part of the project we:

- Compiled a list of open source tools for converting scholarly documents to XML, editing XML files and rendering XML into HTML
- Contacted Editors who had previously expressed an interest in XML/HTML, and found out more about their production workflows and what they were looking to get out of XML
- Contacted other institutions that had undertaken similar projects and were willing to share their experiences and lessons learned
- Analysed existing documents (published journal articles) to identify potential problems
- Created an evaluation matrix based on the Software Sustainability Institute guidelines
- Sourced sample files for testing and produced additional material
- Reviewed, tested and evaluated the available tools
- Selected the tools that gave the most acceptable results
- Tried to combine the tools into a workflow, noting points that need attention
- Identified areas of improvement for the tools selected and highlighted tools that are currently in earlier stages of development but seem promising
- Produced an initial guide for Editors

We now have a really good understanding of open source options for XML-based publishing. We are also much better placed to advise Editors on the relative strengths and limitations of the different available solutions, and we have developed a workflow that, although imperfect and still requiring some manual corrections, could be used by Editors who want to incorporate XML into their production processes.

# Overview of open source tools for XML publishing

## Conversion to XML

The converters we reviewed included standalone tools with a terminal or a graphical interface, and tools integrated with OJS.

- DOCX2JATS and odf2jats are both terminal-interface tools. They rely on textual styles for the conversion and as such require documents to be formatted in line with specific requirements. As their names suggest, the former converts from docx documents and the latter from the open format odf. With both tools, a lot of manual cleaning is needed, because several textual elements are recognised incorrectly or not at all.
- A reworked version of DOCX2JATS, docxToJats, gives slightly better results and is available as an OJS plugin (docxConverter), which makes it a much more user-friendly option.
- GROBID extracts and parses text from PDF files and converts it to TEI XML. It is available as a standalone graphical-interface tool, and it gives impressive results when it comes to header information (title, keywords, author names and affiliations, etc.) and references, but has serious limitations when it comes to the handling of the body text. Also, the input and output formats it supports require further conversions (from Word or similar to PDF and then from TEI to JATS XML).
- meTypeset is a terminal-interface tool that coverts docx files to NLM/JATS XML. It requires the text to be formatted in a particular way and gives mixed results.
- Pandoc is a terminal-interface tool that coverts from and to a variety of formats, and it is actively and widely supported by a community of developers and users. However, it does not handle very well the structure and hierarchy of the document it converts to XML.

Of the tools reviewed, docxToJats and its OJS plugin seemed the most appropriate for our use case.

## Editing the XML

As all conversions tested are imperfect and require some degree of manual cleaning, there is need for a tool that allows Journal Editors to make corrections without having to have specialist software (e.g. Oxygen) or knowledge of XML.

- Texture is a visual XML editor that allows users to produce documents in JATS from scratch or edit existing files. It has a simple, user-friendly graphical interface and it is available as a standalone tool and as an OJS plugin. Note, however, that it supports only a strict subset of JATS elements, which some may find too restrictive – see the list of currently supported elements for details.

## Rendering into HTML in OJS

There are converters that work with JATS and produce HTML files, e.g. Pandoc, but we chose to work with existing OJS plugins that render XML into HTML online without the need for separate conversion. In this way we also keep as much of the work within OJS as possible.

- embedGalley displays the full text embedded within the article landing page. One of its limitations is that views of the page are not captured as full-text views/downloads in the usage statistics. It also appears that that it is no longer actively developed.
- JATSParser displays the full text embedded within the article landing page, but works only with one specific OJS theme.
- Lens is a well-developed tool that displays the full text in a separate window, with flexible navigation options. However, it is not optimised for display on mobile phones and tablets.

*Note that Lens and embedGalley cannot be used simultaneously – if one is enabled in OJS, the other will not work.*

## End-to-end solution in OJS

PKP were previously offering Open Typesetting Stack (OTS) both as a standalone tool and an OJS plugin. OTS allowed Journal Editors to convert documents to XML and edit them from a single location, though the conversion results were not always good. PKP have now deprecated this tool and are working on new version which is likely to use GROBID for conversion.

## Conversion to PDF

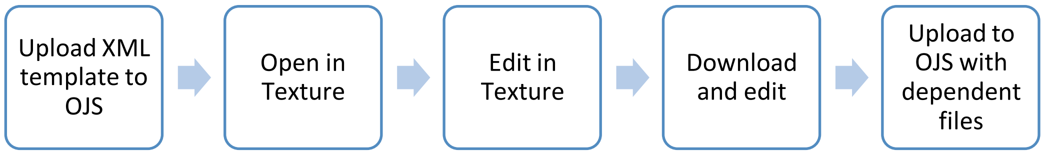
Conversion from JATS to PDF was not within the scope of the project. For an overview of potential options, please refer to the report produced by the journal eLife.

# OJS-based workflow

The options below assume that the paper has been accepted for publication, it already has a submission record in OJS, and copyediting has been completed. Please refer to the OJS user guides at https://edin.ac/oa-resources for details on how to manage those stages of the workflow.

## Option 1: Bypassing automatic conversion

This option bypasses the automatic conversion process, and gives Journal Editors more control over the process. Most of the work is completed in the visual XML editor Texture and does not require any technical expertise.

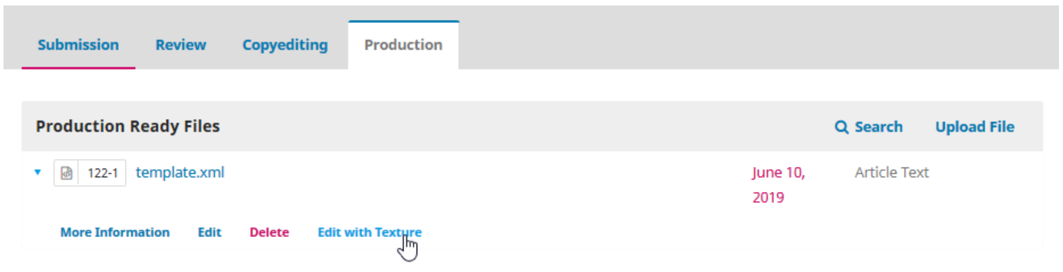

Upload XML template to OJS
Open in Texture
Edit in Texture
Download and edit
Upload to OJS with dependent files

### Requirements

- OJS 3.1.2+
- Texture and Lens plugins installed and enabled
- XML template (see associated template.xml file)
- Dependent files (e.g. images) as separate files
- Optionally, reference list exported from a reference manager like Zotero, in CSL JSON format

### Step 1: Upload empty XML file and open in Texture

1. Go to the OJS admin dashboard > Submissions > find the article you wish to convert and go to its submission record > Production tab > Production Ready Files section
2. Select 'Upload File' and upload the **template.xml** file – this will create a blank document, which you will then be able to edit in the visual XML editor
3. Click on the blue arrow next to the uploaded file and select 'Edit with Texture'

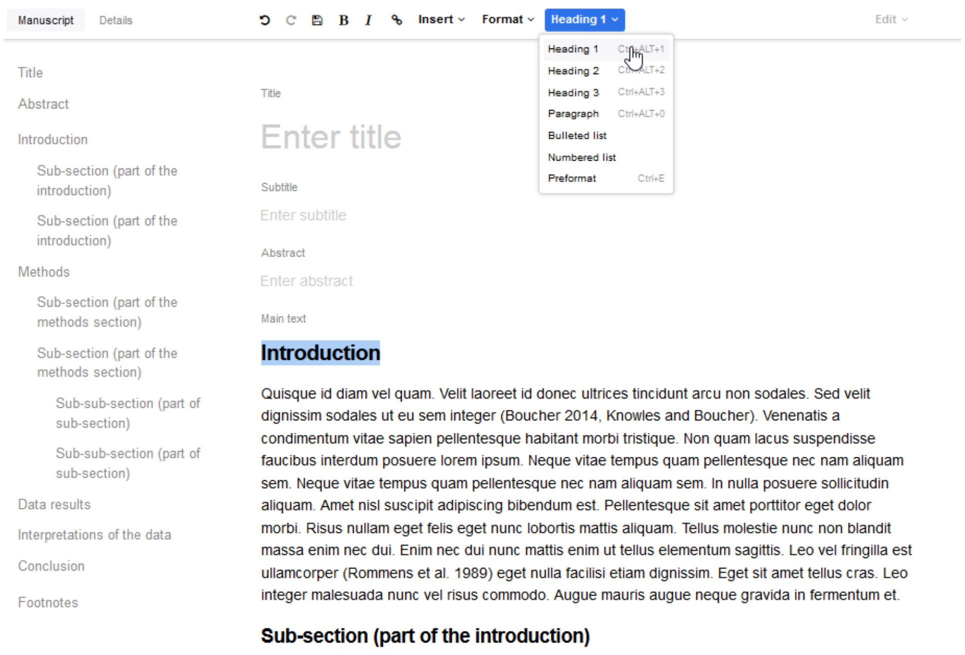


| | | |
|---|---|---|
| 122-1 template.xml | June 10, 2019 | Article Text |

### Step 2: Edit XML file with Texture

1. Paste body text from Word into the 'Main text' section of the page (Manuscript screen)
2. Edit the text as needed:
   - Apply appropriate styles to each heading and paragraph
   - Delete unnecessary spaces

- Use the Bold, Italic and Link options to make basic formatting changes and add links
- To make further formatting changes, select the text you wish to edit and use the Format menu
- Use the Insert menu to add tables, figures, etc. – the Insert menu can also be used to link to these elements from within the text
    - Tables can be pasted from Word, Excel or similar programme (Insert > Table > [empty table is inserted] > paste table into first cell of the empty table)
- Use the Undo, Redo and Edit options as needed



**Introduction**

Quisque id diam vel quam. Velit laoreet id donec ultrices tincidunt arcu non sodales. Sed velit dignissim sodales ut eu sem integer (Boucher 2014, Knowles and Boucher). Venenatis a condimentum vitae sapien pellentesque habitant morbi tristique. Non quam lacus suspendisse faucibus interdum posuere lorem ipsum. Neque vitae tempus quam pellentesque nec nam aliquam sem. Neque vitae tempus quam pellentesque nec nam aliquam sem. In nulla posuere sollicitudin aliquam. Amet nisl suscipit adipiscing bibendum est. Pellentesque sit amet porttitor eget dolor morbi. Risus nullam eget felis eget nunc lobortis mattis aliquam. Tellus molestie nunc non blandit massa enim nec dui. Enim nec dui nunc mattis enim ut tellus elementum sagittis. Leo vel fringilla est ullamcorper (Rommens et al. 1989) eget nulla facilisi etiam dignissim. Eget sit amet tellus cras. Leo integer malesuada nunc vel risus commodo. Augue mauris augue neque gravida in fermentum et.

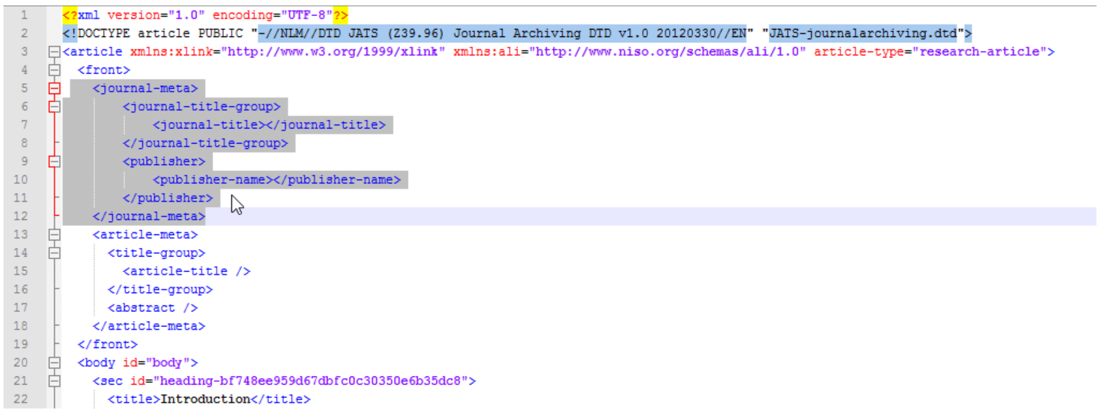
**Sub-section (part of the introduction)**

3. Optionally, go to the Details screen and add article and author information, references, footnotes etc. – references and footnotes can then be linked to from within the text (Insert > Citation or Insert > Footnote)
4. Click on the Save icon and leave Texture

## Step 3: Download and edit XML file

1. Go back to the OJS admin dashboard > Submissions > submission record > Production tab > Production Ready Files section
2. Click on the edited XML file and save it on your computer
3. Open the XML file using a plain text editor, and add the following lines under '<front>' – if you don't add these lines, may receive an error message later on

```
<journal-meta>
        <journal-title-group>
                <journal-title></journal-title>
        </journal-title-group>
        <publisher>
                <publisher-name></publisher-name>
        </publisher>
</journal-meta>
```

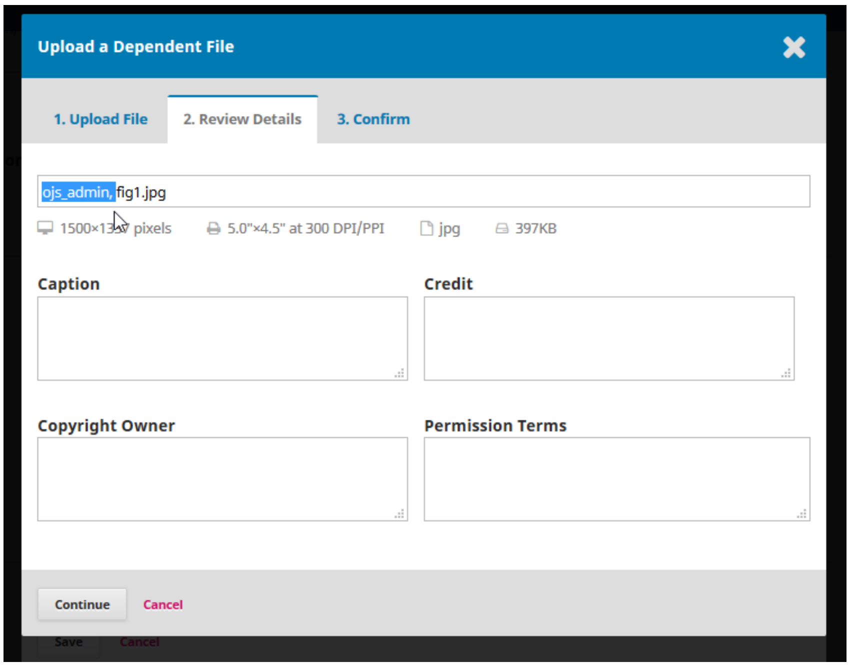


```
1  <?xml version="1.0" encoding="UTF-8"?>
2  <!DOCTYPE article PUBLIC "-//NLM//DTD JATS (Z39.96) Journal Archiving DTD v1.0 20120330//EN" "JATS-journalarchiving.dtd">
3  <article xmlns:xlink="http://www.w3.org/1999/xlink" xmlns:ali="http://www.niso.org/schemas/ali/1.0" article-type="research-article">
4    <front>
5      <journal-meta>
6          <journal-title-group>
7              <journal-title></journal-title>
8          </journal-title-group>
9          <publisher>
10             <publisher-name></publisher-name>
11         </publisher>
12     </journal-meta>
13     <article-meta>
14       <title-group>
15         <article-title />
16       </title-group>
17       <abstract />
18     </article-meta>
19   </front>
20   <body id="body">
21     <sec id="heading-bf748ee959d67dbfc0c30350e6b35dc8">
22       <title>Introduction</title>
```

4. Optionally, change value of @article-type (line 2) to “editorial”, etc. as appropriate
5. Save and close

## Step 4: Upload XML file and dependent files

1. Go back to the OJS admin dashboard > Submissions > submission record > Production tab > Galleys section
2. Select 'Add galley' and upload the XML file and all dependent files (e.g. figures) – please edit the filenames of figures to remove your username, which OJS appends automatically – the names of all figures (or any other dependent files) must be the same as the ones used when the files were uploaded to Texture in Step 2



Upload a Dependent File

1. Upload File | 2. Review Details | 3. Confirm

ojs_admin, fig1.jpg

1500×1350 pixels · 5.0"×4.5" at 300 DPI/PPI · jpg · 397KB

Caption

Credit

Copyright Owner

Permission Terms

Continue · Cancel

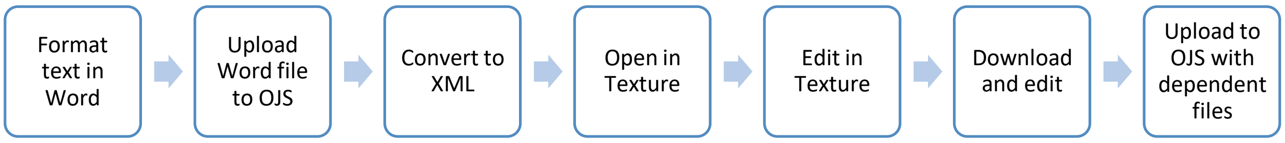
## Step 5: Check

1. Go back to the OJS admin dashboard > Submissions > submission record > Production tab > Schedule For Publication button
2. Assign the article to an issue
3. Go to the OJS admin dashboard > Issues > Future Issues

4. Select to preview the issue where the XML file has been added
5. Click to view the XML version of the article in Lens and check all is okay

## Option 2: With automatic conversion

This option lets you take an article in docx format (e.g. document produced in Word) and convert it to XML before editing in the visual XML editor Texture. As the conversion is done automatically, you may receive error messages if the document is not structured correctly.

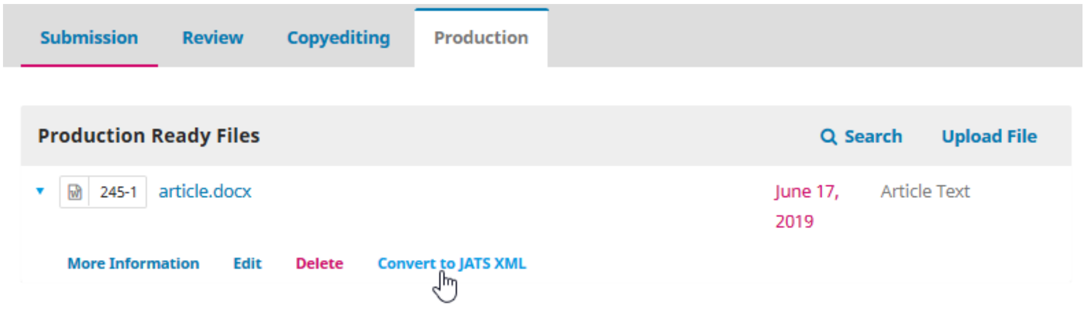


### Requirements

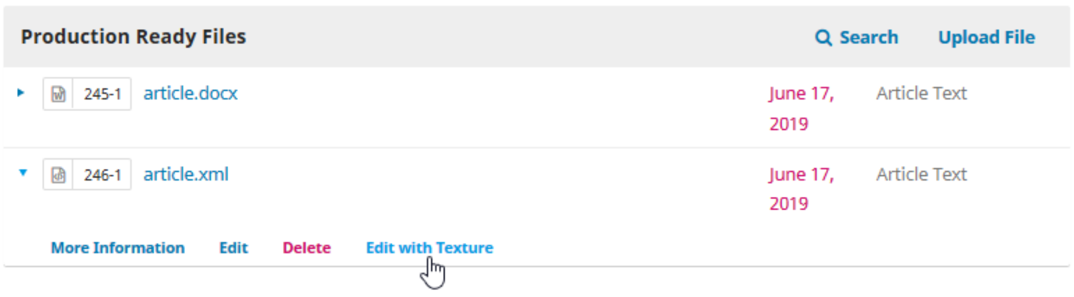
- OJS 3.1.2+
- PHP 7.2, php-xml, php-zip
- docxConverter, Texture and Lens plugins installed and enabled
- Text in Word file is formatted in line with [docxConverter requirements]()
- Dependent files (e.g. images) as separate files
- Optionally, reference list exported from a reference manager like Zotero, in CSL JSON format

### Step 1: Upload Word file, convert to XML and open in Texture

1. Go to the OJS admin dashboard > Submissions > find article and go to its submission record > Production tab > Production Ready Files section
2. Select 'Upload File' and upload the correctly-formatted Word file – no need to upload dependent files
3. Click on the blue arrow next to the uploaded file and select 'Convert to JATS XML'

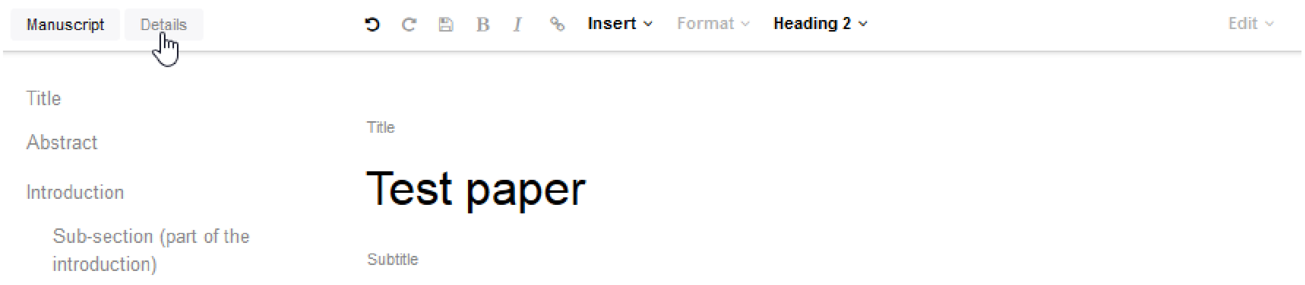


4. Once the page has refreshed, go back to the Production tab > Production Ready Files section
5. Click on the blue arrow next to the new file, ending in .xml, and select 'Edit with Texture'



## Step 2: Edit XML file with Texture

1. On the Manuscript screen (the default view when Texture opens up), edit the text as needed:
   - Apply appropriate styles to each heading and paragraph
   - Delete unnecessary spaces
   - Use the Bold, Italic and Link options to make basic formatting changes and add external links
   - To make further formatting changes, select the text you wish to edit and use the Format menu
   - Use the Insert menu to add tables, figures, etc. – the Insert menu can also be used to link to these elements from within the text
     - Tables can be pasted from Word, Excel or similar programme (Insert > Table > [empty table is inserted] > paste table into first cell of the empty table)
   - Use the Undo, Redo and Edit options as needed
2. Go to the Details screen:
   - Check the article and author information and make corrections as needed
   - Add references and footnotes if needed – these can then be linked to from within the text (Insert > Citation or Insert > Footnote)



3. Click on the Save icon and leave Texture

## Steps 3-5

Same as in Option 1:
[Step 3: Download and edit XML file](#)
[Step 4: Upload XML file and dependent files](#)
[Step 5: Check](#)

# Notes

## Texture

- Requires internet access for the duration of the editing process
- Supports only a strict subset of JATS elements, which some may find limiting – see list of supported elements for details
- Current version of Texture says it supports only Chrome – but it works fine in Firefox
- Table formatting is not possible, e.g. header cannot be made bold
- The are three ways to add structured references, listed below – alternatively, references can be added as a section of the body of the article
    - DOI lookup (supports only DataCite DOIs at the moment)
    - upload CSL JSON file, exported from a reference manager like Zotero
    - manual creation of each reference (can be time consuming)
- Supports only the Vancouver citation style at the moment
- There is a slight delay between exiting Texture and the XML file updating – it might be best to refresh the page before downloading the file (Step 3)
- The <journal-meta> tag will be removed from the XML if the file is opened in Texture and will need to be added manually (Step 3)
- At the moment, images need to be uploaded via Texture (Step 2), and then again as dependent files (Step 4). The names of the files added via Texture and those of the dependent files uploaded later must be identical.
- If the option that bypasses automatic conversion is used, author and article details (e.g. title, author names and affiliations) will need to be added manually in Texture, or they will not display at all

## Lens

- The are two versions of this plugin available at the moment
- The XML file needs to have a <journal-meta> tag (see Step 3), otherwise it may fail to import
- May not work very well on mobile phones and tablets

## docxConverter

- Text must be formatted in line with certain requirements
- The file must be in the docx format