

WebODF  
White paper  
1<sup>st</sup> ed.  
**2011-09-09**

KO GmbH  
<http://www.kogmbh.com/>  
Magdeburg

## Table of Contents

1	Preface.....	3
2	Introduction to WebODF.....	3
3	Document Types / Applications.....	3
4	Technology Overview and Architecture.....	3
	Loading and Saving.....	3
	Displaying.....	4
	Modifying.....	4
5	Use cases.....	5
6	Deployment and Integration.....	5
7	Limitations.....	5
8	Roadmap.....	5
9	Organizations using WebODF.....	5
	Zarafa.....	5
	Tiki-Wiki.....	5
10	Resources.....	6
11	The WebODF Community.....	6
	Open Source Community.....	6
	Commercial Support.....	6
12	Resources.....	6
	12.1 WebODF in AppStores.....	6
13	Conclusion.....	6
14	Postscript.....	7

## 1 Preface

This document describes the WebODF JavaScript library. The intended audience is people who want to find out what WebODF is, and decision-makers potentially interested in deploying WebODF software within their organization, either in websites, desktop applications or mobile applications.

## 2 Introduction to WebODF

WebODF is a Javascript library designed to show documents in the OpenDocument Format<sup>1</sup> (ODF) in a web browser. WebODF comprises the following components:

- a **JavaScript library** that is organized into modules that use the OpenDocument format (ODF) as its native format. The library allows use in user interfaces for desktop and handset form factors as well as deployment in websites and web-based applications such as CMS systems.
- a set of **example programs** that show how WebODF can be used. Among the example applications are an Android application, a Qt application and a web based file browser.

WebODF is built using the web standards from W3 (<http://w3.org>) commonly referred to as HTML5. This makes WebODF portable to all major desktop and mobile platforms. WebODF currently runs on Firefox, Chrome, Safari, Opera and Internet Explorer. It can be used in software components with HTML support, a common feature in modern programming environments.

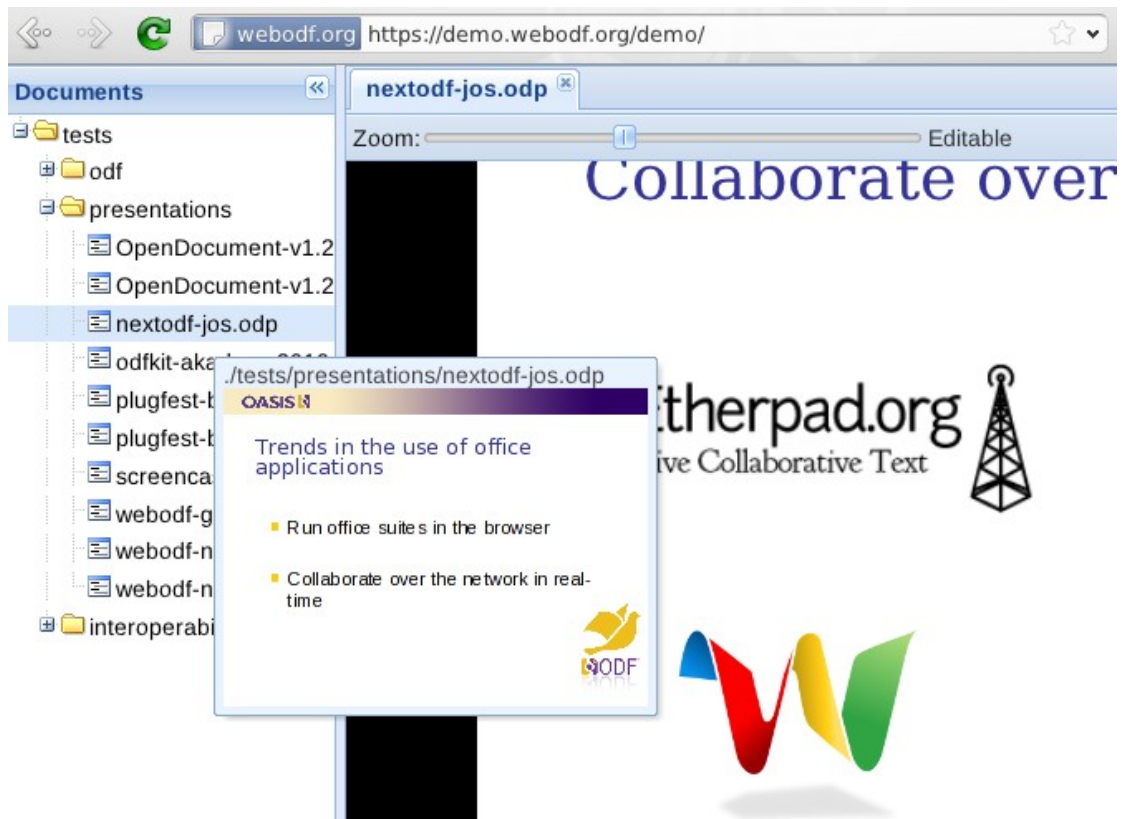


Illustration 1: WebODF showing a presentation document

<sup>1</sup> <http://en.wikipedia.org/wiki/OpenDocument>

### 3 Document Types / Applications

WebODF can handle the following OpenDocument types:

Type	Produced by (for example)
• <b>Text Documents</b>	MS Word, OpenOffice.org Writer or Calligra Words
• <b>Spreadsheets</b>	MS Excel, OpenOffice.org Calc or Calligra Tables
• <b>Presentations</b>	MS PowerPoint, OpenOffice.org Impress or Calligra Stage

OpenDocument drawings and database files are not supported by WebODF at this point. If the viewing capabilities of WebODF is combined with software for converting documents in other formats to ODF, it can also be used to view the other formats in question. One example of such an automatic conversion tool can be found in the Calligra Suite<sup>2</sup>.

### 4 Technology Overview and Architecture

This section describes the technology behind WebODF. It explains how WebODF can display ODF documents using only technology that is common in modern web browsers. There are three aspects to WebODF functioning that relate to: loading and saving, displaying and modifying the ODF documents. Each of these aspects is described in its own section below.

Keep in mind that when we mention web pages, the same reasoning also applies to desktop and mobile applications. In such applications, WebODF could run in an isolated web page inside the application through the use of an embedded browser. The WebKit toolkit is often used for this, and is integrated into most widely used UI toolkits.

#### Loading and Saving

ODF documents come in two formats. The most common format is a ZIP container that contains XML files specified in the ODF standard and additional non-XML files such as e.g. pictures. Browser technology is not typically very suited for working with binary files such as ZIP files. In WebODF we have found a way around this limitation that works in all modern browsers: The XML files can be extracted from the ZIP file using optimized Javascript code. Subsequently they are inserted into the DOM tree of the HTML page, thereby using the full rendering speed of the web browser itself.

An ODF document can also be a single XML file. This case is however rarely used since it uses more disk space than the compressed ZIP file. Handling such files is easier, though, since they can be loaded directly.

Saving the files is done via the reverse process. The default implementation loads and saves via XMLHttpRequests. This behavior can be adapted to the environment in which WebODF runs (see below about Deployment and Integration). The functions for reading and writing to storage can be easily redefined to suit every need. WebODF will automatically use the redefined functions.

#### Displaying

Just like ODF files, web pages consist of a form of XML, namely HTML. Cascading Style Sheets, CSS, are used to determine the look of web pages and are give rendering hints that are used to render the contents inside the browser. The same technology can be used to render any other XML documents.

ODF uses a different technology for styling; Styling information is stored in a separate section of ODF documents in a format specific for ODF. In WebODF, this styling information is converted to CSS and added to the same web page, using the full speed of the web browser.

<sup>2</sup> See <http://www.calligra-suite.org/> for more information. The conversion tool is called koconverter, and is an easily separated tool for batch conversion of documents to and from any supported format.

## Modifying

A document that has been loaded into WebODF can be modified using the same technologies that are used to modify any web page. Any part of the document may be modified with common JavaScript calls. After modification the styling information is updated by calling the appropriate functions of WebODF.

In addition to programmatically creating and modifying ODF documents, a controlled amount of direct editing by the user is permitted. This is done by specifying some or all of the paragraphs as editable. Thus, WebODF is suited for use in cases where only certain parts of documents should be editable.

## 5 Use cases

Even though WebODF is still a young project, the software is mature enough to be used in production environments for a number of different use cases. Here are some examples of what WebODF is already used for in different organizations:

- **Fallback viewer.** WebODF can easily be used as a cheap fallback document viewer if you lack a more advanced but expensive (in terms of resources) native viewer.
- **Mobile viewer.** WebODF is used as a document viewer for several mobile platforms. It is also packaged as a stand-alone application with a built-in browser for the Android Market.
- **Editor for limited editing.** Although still in its infancy, the area where most development is going into WebODF currently is editing. Already it can be used to edit documents if the type of editing you need to do is limited.
- **Building block in your own application, website or mobile application.** It is easy to use WebODF as a building block or library in your own application or website. See the next chapter about Deployment and Integration for more information.

## 6 Deployment and Integration

WebODF consists of a number of JavaScript files that are compiled into one file for convenience and speed. This file, *webodf.js* and an additional styling file, *webodf.css*, are all that is needed to include WebODF in your own site. The files can be used to provide viewer functionality in a website. If documents are placed on a normal web server, the only work needed is to connect the loading of documents to the interface of your web site. WebODF will then load documents via normal HTTP calls.

Commonly a website will employ a custom method for storing documents. WebODF is ready for such situations: the JavaScript methods for reading and writing documents can easily be redefined to use the custom storage method in question.

In mobile and desktop applications, the methods for reading and writing can be overridden so that WebODF has controlled access to device storage. Native UI components can be used in combination with the WebODF canvas, which is a standard web page. Every modern native environment such as Android, and iOS, supports this.

## 7 Limitations

At this point WebODF cannot handle all formatting features of ODF. Here follows a list of formatting that are not currently handled:

### General

- Vector Graphics.

### Text documents, mostly page related issues

- Pagination, i.e. splitting a text document into pages of a predefined size
- Page headers and footers
- Footnotes and endnotes
- Tracked changes cannot be shown

### Spreadsheets

- Formulas are not evaluated. Spreadsheet documents store both formulas and calculated values, and the stored values are always shown.

### Presentations

- Slide masters and slide styles; Only style elements that are stored on the slides themselves are shown.

## 8 Roadmap

Future improvements for WebODF include:

- **Better rendering.** Currently, there are a number of features of ODF that are not rendered perfectly. Most of these have to do with pagination, but there are a few others, like formula evaluation for spreadsheets that can be improved while staying in the web page paradigm.

Other features will have to use external support, for example support for generic vector images (e.g. WMF, EMF, etc) that can be transformed to formats that are supported by web browsers, like SVG. Such support on a server or as part of a native application can also be used for other purposes. Examples include converting document files from other formats than ODF, like the Microsoft binary formats doc/xls/ppt and XML formats docx/xlsx/pptx.

- **Better editing.** The editing user interface can be improved with many new features. Most formatting options of ODF are not yet supported by the editing UI, but this will improve over time.
- **General XML editor.** The current editor for ODF features could easily be extended to create a general editor for XML file formats.
- **Improved user interface.** The usability of the current user interface will be improved.

## 9 Organizations using WebODF

A number of companies and organizations are already using WebODF. Most of them use it as a document viewer in their web based application. Here we give a few examples. Many more are working already to integrate WebODF into their offerings.

### Zarafa

The email server software Zarafa has a web interface created in HTML5. This interface has been designed to be extendable via plugins. A community website is available where Zarafa and 3<sup>rd</sup> party developers offer plugins<sup>3</sup>. WebODF is available as such a plugin for viewing OpenDocument files on this website.

### Tiki-Wiki

Tiki Wiki is an enterprise collaboration suite<sup>4</sup>. It allows sharing of, and collaborating around

<sup>3</sup> <http://community.zarafa.com/>

<sup>4</sup> <http://tiki.org/>

documents. The Tiki community has developed a feature called Tiki Docs. It allows uploading, viewing and limited editing of ODF documents. Previous and current versions of these ODF documents can be shared and viewed.

## 10 The WebODF Community

The WebODF community consists of a combination of volunteers and paid developers. All the core developers are professional software developers and designers. The main organization behind WebODF is KO GmbH in Germany.

### Open Source Community

WebODF is developed and available as Open Source software. It is developed in open using the Gitorious development resources (see below about Resources). Anybody can contribute to WebODF and anybody can use it in their own applications or on their own web server.

### Commercial Support

The German-based consulting company KO GmbH is behind WebODF. It offers professional services around WebODF, the Calligra Suite and the OpenDocument Format. See <http://www.kogmbh.com/> for more information.

## 11 Resources

There are many resources for finding out more about WebODF and its community.

WebODF website: <http://www.webodf.org/>  
Developer Wiki: <http://webodf.org/redmine/projects/webodf/wiki>  
WebODF Forum: <http://www.webodf.org/redmine/projects/webodf/boards>  
WebODF bug tracker: <http://www.webodf.org/redmine/projects/webodf/issues>  
WebODF Source code: <http://gitorious.org/odfkit/webodf>  
Developer mailing list: <https://lists.opendocsociety.org/mailman/listinfo/webodf>  
IRC channel: #webodf at <irc://irc.freenode.org>

### 11.1 WebODF in AppStores

You can buy or download packaged versions of WebODF from the following appstores:

- Android Market
- Zarafa Community Hub: <http://community.zarafa.com/pg/plugins/project/656>

We expect more to come soon.

## 12 Conclusion

If you need a simple, fast and comprehensive document viewer on your website, you should take a closer look at WebODF. WebODF itself can show text documents, spreadsheets and presentations stored in ODF, the OpenDocument Format, and can also show documents stored in other formats with external support for conversion of documents.

Also, if you need a simple editor of such documents, WebODF can be for you. If you need a full featured web based editor then your needs are probably not met by WebODF at this time.

## 13 Postscript

For questions that are not answered in this white paper, contact the author Inge Wallin. If you

---

have a technical question, send a mail to the developer mailing list or talk with the developers in the IRC channel. The Team is always happy to help.