PhpWord Documentation

Release 0.9.0

The PhpWord Team

October 05, 2014

PHPWord is a library written in pure PHP that provides a set of classes to write to and read from different document file formats. The current version of PHPWord supports Microsoft Office Open XML (OOXML or OpenXML), OASIS Open Document Format for Office Applications (OpenDocument or ODF), and Rich Text Format (RTF).

CHAPTER 1

Introduction

PHPWord is a library written in pure PHP that provides a set of classes to write to and read from different document file formats. The current version of PHPWord supports Microsoft Office Open XML (OOXML or OpenXML), OASIS Open Document Format for Office Applications (OpenDocument or ODF), and Rich Text Format (RTF).

PHPWord is an open source project licensed under the terms of LGPL version 3. PHPWord is aimed to be a high quality software product by incorporating continuous integration and unit testing. You can learn more about PHPWord by reading this Developers' Documentation and the API Documentation.

1.1 Features

- Set document properties, e.g. title, subject, and creator.
- Create document sections with different settings, e.g. portrait/landscape, page size, and page numbering
- Create header and footer for each sections
- Set default font type, font size, and paragraph style
- Use UTF-8 and East Asia fonts/characters
- Define custom font styles (e.g. bold, italic, color) and paragraph styles (e.g. centered, multicolumns, spacing) either as named style or inline in text
- Insert paragraphs, either as a simple text or complex one (a text run) that contains other elements
- Insert titles (headers) and table of contents
- Insert text breaks and page breaks
- Insert and format images, either local, remote, or as page watermarks
- Insert binary OLE Objects such as Excel or Visio
- Insert and format table with customized properties for each rows (e.g. repeat as header row) and cells (e.g. background color, rowspan, colspan)
- Insert list items as bulleted, numbered, or multilevel
- Insert hyperlinks
- Insert footnotes and endnotes
- Create document from templates
- Use XSL 1.0 style sheets to transform main document part of OOXML template
- ... and many more features on progress

1.2 File formats

Below are the supported features for each file formats.

1.2.1 Writers

Features		DOCX	ODT	RTF	HTML	PDF
Document Properties	Standard					
	Custom					
Element Type	Text					
	Text Run					
	Title					
	Link					
	Preserve Text					
	Text Break					
	Page Break					
	List					
	Table					
	Image					
	Object					
	Watermark					
	Table of Contents					
	Header					
	Footer					
	Footnote					
	Endnote					
Graphs	2D basic graphs					
	2D advanced graphs					
	3D graphs					
Math	OMML support					
	MathML support					
Bonus	Encryption					
	Protection					

1.2.2 Readers

Features		DOCX	ODT	RTF	HTML
Document Properties	Standard				
	Custom				
Element Type	Text				
	Text Run				
	Title				
	Link				
	Preserve Text				
	Text Break				
	Page Break				
	List				
	Table				
	Image				
	Object				
	Watermark				
	Table of Contents				
	Header				
	Footer				
	Footnote				
	Endnote				
Graphs	2D basic graphs				
	2D advanced graphs				
	3D graphs				
Math	OMML support				
	MathML support				
Bonus	Encryption				
	Protection				

1.3 Contributing

We welcome everyone to contribute to PHPWord. Below are some of the things that you can do to contribute:

- Read our contributing guide
- Fork us and request a pull to the develop branch
- Submit bug reports or feature requests to GitHub
- Follow @PHPWord and @PHPOffice on Twitter

CHAPTER 2

Installing/configuring

2.1 Requirements

Mandatory:

- PHP 5.3+
- PHP Zip extension
- PHP XML Parser extension

Optional PHP extensions:

- GD
- XMLWriter
- XSL

2.2 Installation

There are two ways to install PHPWord, i.e. via Composer or manually by downloading the library.

2.2.1 Using Composer

To install via Composer, add the following lines to your `composer.json`:

```
{
    "require": {
       "phpoffice/phpword": "dev-master"
    }
}
```

2.2.2 Manual install

To install manually, download PHPWord package from github. Extract the package and put the contents to your machine. To use the library, include `src/PhpWord/Autoloader.php` in your script and invoke `Autoloader::register`.

```
require_once '/path/to/src/PhpWord/Autoloader.php';
\PhpOffice\PhpWord\Autoloader::register();
```

2.3 Using samples

After installation, you can browse and use the samples that we've provided, either by command line or using browser. If you can access your PHPWord library folder using browser, point your browser to the `samples` folder, e.g. `http://localhost/PhpWord/samples/`.

CHAPTER 3

General usage

3.1 Basic example

The following is a basic example of the PHPWord library. More examples are provided in the samples folder.

```
require_once 'src/PhpWord/Autoloader.php';
\PhpOffice\PhpWord\Autoloader::register();

$phpWord = new \PhpOffice\PhpWord\PhpWord();

// Every element you want to append to the word document is placed in a section.
// To create a basic section:
$section = $phpWord->addSection();

// After creating a section, you can append elements:
$section->addText('Hello world!');

// You can directly style your text by giving the addText function an array:
$section->addText('Hello world! I am formatted.',
    array('name'=>'Tahoma', 'size'=>16, 'bold'=>true));

// If you often need the same style again you can create a user defined style
// to the word document and give the addText function the name of the style:
$phpWord->addFontStyle('myOwnStyle',
    array('name'=>'Verdana', 'size'=>14, 'color'=>'1B2232'));
$section->addText('Hello world! I am formatted by a user defined style',
    'myOwnStyle');

// You can also put the appended element to local object like this:
$fontStyle = new \PhpOffice\PhpWord\Style\Font();
$fontStyle->setBold(true);
$fontStyle->setName('Verdana');
$fontStyle->setSize(22);
$myTextElement = $section->addText('Hello World!');
$myTextElement->setFontStyle($fontStyle);

// Finally, write the document:
$objWriter = \PhpOffice\PhpWord\IOFactory::createWriter($phpWord, 'Word2007');
$objWriter->save('helloWorld.docx');

$objWriter = \PhpOffice\PhpWord\IOFactory::createWriter($phpWord, 'ODText');
$objWriter->save('helloWorld.odt');
```

```
$objWriter = \PhpOffice\PhpWord\IOFactory::createWriter($phpWord, 'RTF');
$objWriter->save('helloWorld.rtf');
```

3.2 Settings

The `PhpOffice\PhpWord\Settings` class provides some options that will affect the behavior of PHPWord. Below are the options.

3.2.1 XML Writer compatibility

This option sets XMLWriter::setIndent and XMLWriter::setIndentString. The default value of this option is `true` (compatible), which is required for OpenOffice to render OOXML document correctly. You can set this option to `false` during development to make the resulting XML file easier to read.

```
\PhpOffice\PhpWord\Settings::setCompatibility(false);
```

3.2.2 Zip class

By default, PHPWord uses PHP ZipArchive to read or write ZIP compressed archive and the files inside them. If you can't have ZipArchive installed on your server, you can use pure PHP library alternative, PCLZip, which included with PHPWord.

```
\PhpOffice\PhpWord\Settings::setZipClass(\PhpOffice\PhpWord\Settings::PCLZIP);
```

3.3 Default font

By default, every text appears in Arial 10 point. You can alter the default font by using the following two functions:

```
$phpWord->setDefaultFontName('Times New Roman');
$phpWord->setDefaultFontSize(12);
```

3.4 Document properties

You can set the document properties such as title, creator, and company name. Use the following functions:

```
$properties = $phpWord->getDocumentProperties();
$properties->setCreator('My name');
$properties->setCompany('My factory');
$properties->setTitle('My title');
$properties->setDescription('My description');
$properties->setCategory('My category');
$properties->setLastModifiedBy('My name');
$properties->setCreated(mktime(0, 0, 0, 3, 12, 2014));
$properties->setModified(mktime(0, 0, 0, 3, 14, 2014));
$properties->setSubject('My subject');
$properties->setKeywords('my, key, word');
```

3.5 Measurement units

The base length unit in Open Office XML is twip. Twip means “TWentieth of an Inch Point”, i.e. 1 twip = 1/1440 inch.

You can use PHPWord helper functions to convert inches, centimeters, or points to twips.

```
// Paragraph with 6 points space after
$phpWord->addParagraphStyle('My Style', array(
    'spaceAfter' => \PhpOffice\PhpWord\Shared\Font::pointSizeToTwips(6))
);

$section = $phpWord->addSection();
$sectionStyle = $section->getSettings();
// half inch left margin
$sectionStyle->setMarginLeft(\PhpOffice\PhpWord\Shared\Font::inchSizeToTwips(.5));
// 2 cm right margin
$sectionStyle->setMarginRight(\PhpOffice\PhpWord\Shared\Font::centimeterSizeToTwips(2));
```

CHAPTER 4

Containers

Containers are objects where you can put elements (texts, lists, tables, etc). There are 3 main containers, i.e. sections, headers, and footers. There are 3 elements that can also act as containers, i.e. textruns, table cells, and footnotes.

4.1 Sections

Every visible element in word is placed inside of a section. To create a section, use the following code:

```
$section = $phpWord->addSection($sectionSettings);
```

The `$sectionSettings` is an optional associative array that sets the section. Example:

```
$sectionSettings = array(
    'orientation' => 'landscape',
    'marginTop' => 600,
    'colsNum' => 2,
);
```

4.1.1 Section settings

Below are the available settings for section:

- `orientation` Page orientation, i.e. 'portrait' (default) or 'landscape'
- `marginTop` Page margin top in twips
- `marginLeft` Page margin left in twips
- `marginRight` Page margin right in twips
- `marginBottom` Page margin bottom in twips
- `borderTopSize` Border top size in twips
- `borderTopColor` Border top color
- `borderLeftSize` Border left size in twips
- `borderLeftColor` Border left color
- `borderRightSize` Border right size in twips
- `borderRightColor` Border right color
- `borderBottomSize` Border bottom size in twips

- `borderBottomColor` Border bottom color
- `headerHeight` Spacing to top of header
- `footerHeight` Spacing to bottom of footer
- `gutter` Page gutter spacing
- `colsNum` Number of columns
- `colsSpace` Spacing between columns
- `breakType` Section break type (nextPage, nextColumn, continuous, evenPage, oddPage)

The following two settings are automatically set by the use of the `orientation` setting. You can alter them but that's not recommended.

- `pageSizeW` Page width in twips
- `pageSizeH` Page height in twips

4.1.2 Page number

You can change a section page number by using the `pageNumberingStart` property of the section.

```
// Method 1
$section = $phpWord->addSection(array('pageNumberingStart' => 1));

// Method 2
$section = $phpWord->addSection();
$section->getSettings()->setPageNumberingStart(1);
```

4.1.3 Multicolumn

You can change a section layout to multicolumn (like in a newspaper) by using the `breakType` and `colsNum` property of the section.

```
// Method 1
$section = $phpWord->addSection(array('breakType' => 'continuous', 'colsNum' => 2));

// Method 2
$section = $phpWord->addSection();
$section->getSettings()->setBreakType('continuous');
$section->getSettings()->setColsNum(2);
```

Line numbering

You can apply line numbering to a section by using the `lineNumbering` property of the section.

```
// Method 1
$section = $phpWord->addSection(array('lineNumbering' => array()));

// Method 2
$section = $phpWord->addSection();
$section->getSettings()->setLineNumbering(array());
```

Below are the properties of the line numbering style.

- `start` Line numbering starting value
- `increment` Line number increments

- `distance` Distance between text and line numbering in twip
- `restart` Line numbering restart setting continuous|newPage|newSection

4.2 Headers

Each section can have its own header reference. To create a header use the `addHeader` method:

```
$header = $section->addHeader();
```

Be sure to save the result in a local object. You can use all elements that are available for the footer. See "Footer" section for detail. Additionally, only inside of the header reference you can add watermarks or background pictures. See "Watermarks" section.

4.3 Footers

Each section can have its own footer reference. To create a footer, use the `addFooter` method:

```
$footer = $section->addFooter();
```

Be sure to save the result in a local object to add elements to a footer. You can add the following elements to footers:

- Texts `addText` and `createTextrun`
- Text breaks
- Images
- Tables
- Preserve text

See the "Elements" section for the detail of each elements.

4.4 Other containers

Textruns, table cells, and footnotes are elements that can also act as containers. See the corresponding "Elements" section for the detail of each elements.

Elements

Below are the matrix of element availability in each container. The column shows the containers while the rows lists the elements.

Num	Element	Section	Header	Footer	Cell	Text Run	Footnote
1	Text	v	v	v	v	v	v
2	Text Run	v	v	v	v	•	•
3	Link	v	v	v	v	v	v
4	Title	v	?	?	?	?	?
5	Preserve Text	?	v	v	v*	•	•
6	Text Break	v	v	v	v	v	v
7	Page Break	v	•	•	•	•	•
8	List	v	v	v	v	•	•
9	Table	v	v	v	v	•	•
10	Image	v	v	v	v	v	v
11	Watermark	•	v	•	•	•	•
12	Object	v	v	v	v	v	v
13	TOC	v	•	•	•	•	•
14	Footnote	v	•	•	v**	v**	•
15	Endnote	v	•	•	v**	v**	•
16	CheckBox	v	v	v	v	•	•
17	TextBox	v	v	v	v	•	•
18	Field	v	v	v	v	v	v
19	Line	v	v	v	v	v	v

Legend:

- `v` Available
- `v*` Available only when inside header/footer
- `v**` Available only when inside section
- `-` Not available
- `?` Should be available

5.1 Texts

Text can be added by using `addText` and `addTextRun` method. `addText` is used for creating simple paragraphs that only contain texts with the same style. `addTextRun` is used for creating complex paragraphs that contain text with different style (some bold, other italics, etc) or other elements, e.g. images or links. The syntaxes are as follow:

```
$section->addText($text, [$fontStyle], [$paragraphStyle]);
$textrun = $section->addTextRun([$paragraphStyle]);
```

5.1.1 Text styles

You can use the `$fontStyle` and `$paragraphStyle` variable to define text formatting. There are 2 options to style the inserted text elements, i.e. inline style by using array or defined style by adding style definition.

Inline style examples:

```
$fontStyle = array('name' => 'Times New Roman', 'size' => 9);
$paragraphStyle = array('align' => 'both');
$section->addText('I am simple paragraph', $fontStyle, $paragraphStyle);

$textrun = $section->addTextRun();
$textrun->addText('I am bold', array('bold' => true));
$textrun->addText('I am italic', array('italic' => true));
$textrun->addText('I am colored', array('color' => 'AACC00'));
```

Defined style examples:

```
$fontStyle = array('color' => '006699', 'size' => 18, 'bold' => true);
$phpWord->addFontStyle('fStyle', $fontStyle);
$text = $section->addText('Hello world!', 'fStyle');

$paragraphStyle = array('align' => 'center');
$phpWord->addParagraphStyle('pStyle', $paragraphStyle);
$text = $section->addText('Hello world!', 'pStyle');
```

Font style

Available font styles:

- `name` Font name, e.g. *Arial*
- `size` Font size, e.g. *20*, *22*,
- `hint` Font content type, *default*, *eastAsia*, or *cs*
- `bold` Bold, *true* or *false*
- `italic` Italic, *true* or *false*

- `superScript` Superscript, *true* or *false*
- `subScript` Subscript, *true* or *false*
- `underline` Underline, *dash*, *dotted*, etc.
- `strikethrough` Strikethrough, *true* or *false*
- `doubleStrikethrough` Double strikethrough, *true* or *false*
- `color` Font color, e.g. *FF0000*
- `fgColor` Font highlight color, e.g. *yellow*, *green*, *blue*
- `bgColor` Font background color, e.g. *FF0000*
- `smallCaps` Small caps, *true* or *false*
- `allCaps` All caps, *true* or *false*

Paragraph style

Available paragraph styles:

- `align` Paragraph alignment, *left*, *right* or *center*
- `spaceBefore` Space before paragraph
- `spaceAfter` Space after paragraph
- `indent` Indent by how much
- `hanging` Hanging by how much
- `basedOn` Parent style
- `next` Style for next paragraph
- `widowControl` Allow first/last line to display on a separate page, *true* or *false*
- `keepNext` Keep paragraph with next paragraph, *true* or *false*
- `keepLines` Keep all lines on one page, *true* or *false*
- `pageBreakBefore` Start paragraph on next page, *true* or *false*
- `lineHeight` text line height, e.g. *1.0*, *1.5*, ect...
- `tabs` Set of custom tab stops

5.1.2 Titles

If you want to structure your document or build table of contents, you need titles or headings. To add a title to the document, use the `addTitleStyle` and `addTitle` method.

```
$phpWord->addTitleStyle($depth, [$fontStyle], [$paragraphStyle]);
$section->addTitle($text, [$depth]);
```

Its necessary to add a title style to your document because otherwise the title won’t be detected as a real title.

5.1.3 Links

You can add Hyperlinks to the document by using the function addLink:

```
$section->addLink($linkSrc, [$linkName], [$fontStyle], [$paragraphStyle]);
```

- `$linkSrc` The URL of the link.
- `$linkName` Placeholder of the URL that appears in the document.
- `$fontStyle` See "Font style" section.
- `$paragraphStyle` See "Paragraph style" section.

5.1.4 Preserve texts

The `addPreserveText` method is used to add a page number or page count to headers or footers.

```
$footer->addPreserveText('Page {PAGE} of {NUMPAGES}.');
```

5.2 Breaks

5.2.1 Text breaks

Text breaks are empty new lines. To add text breaks, use the following syntax. All paramaters are optional.

```
$section->addTextBreak([$breakCount], [$fontStyle], [$paragraphStyle]);
```

- `$breakCount` How many lines
- `$fontStyle` See "Font style" section.
- `$paragraphStyle` See "Paragraph style" section.

5.2.2 Page breaks

There are two ways to insert a page breaks, using the `addPageBreak` method or using the `pageBreakBefore` style of paragraph.

:: code-block:: php

  \$section->addPageBreak();

5.3 Lists

To add a list item use the function `addListItem`.

Basic usage:

```
$section->addListItem($text, [$depth], [$fontStyle], [$listStyle], [$paragraphStyle]);
```

Parameters:

- `$text` Text that appears in the document.

- `$depth` Depth of list item.
- `$fontStyle` See “Font style” section.
- `$listStyle` List style of the current element TYPE_NUMBER, TYPE_ALPHANUM, TYPE_BULLET_FILLED, etc. See list of constants in PHPWord_Style_ListItem.
- `$paragraphStyle` See “Paragraph style” section.

Advanced usage:

You can also create your own numbering style by changing the `$listStyle` parameter with the name of your numbering style.

```
$phpWord->addNumberingStyle(
    'multilevel',
    array('type' => 'multilevel', 'levels' => array(
        array('format' => 'decimal', 'text' => '%1.', 'left' => 360, 'hanging' => 360, 'tabPos' => 3
        array('format' => 'upperLetter', 'text' => '%2.', 'left' => 720, 'hanging' => 360, 'tabPos' =
        )
     )
);
$section->addListItem('List Item I', 0, null, 'multilevel');
$section->addListItem('List Item I.a', 1, null, 'multilevel');
$section->addListItem('List Item I.b', 1, null, 'multilevel');
$section->addListItem('List Item II', 0, null, 'multilevel');
```

Level styles:

- `start` Starting value
- `format` Numbering format bullet|decimal|upperRoman|lowerRoman|upperLetter|lowerLetter
- `restart` Restart numbering level symbol
- `suffix` Content between numbering symbol and paragraph text tab|space|nothing
- `text` Numbering level text e.g. %1 for nonbullet or bullet character
- `align` Numbering symbol align left|center|right|both
- `left` See paragraph style
- `hanging` See paragraph style
- `tabPos` See paragraph style
- `font` Font name
- `hint` See font style

5.4 Tables

To add tables, rows, and cells, use the `addTable`, `addRow`, and `addCell` methods:

```
$table = $section->addTable([$tableStyle]);
$table->addRow([$height], [$rowStyle]);
$cell = $table->addCell($width, [$cellStyle]);
```

Table style can be defined with `addTableStyle`:

```
$tableStyle = array(
    'borderColor' => '006699',
    'borderSize' => 6,
    'cellMargin' => 50
);
$firstRowStyle = array('bgColor' => '66BBFF');
$phpWord->addTableStyle('myTable', $tableStyle, $firstRowStyle);
$table = $section->addTable('myTable');
```

5.4.1 Table, row, and cell styles

Table styles:

- `width` Table width in percent
- `bgColor` Background color, e.g. '9966CC'
- `border(Top|Right|Bottom|Left)Size` Border size in twips
- `border(Top|Right|Bottom|Left)Color` Border color, e.g. '9966CC'
- `cellMargin(Top|Right|Bottom|Left)` Cell margin in twips

Row styles:

- `tblHeader` Repeat table row on every new page, *true* or *false*
- `cantSplit` Table row cannot break across pages, *true* or *false*
- `exactHeight` Row height is exact or at least

Cell styles:

- `width` Cell width in twips
- `valign` Vertical alignment, *top*, *center*, *both*, *bottom*
- `textDirection` Direction of text
- `bgColor` Background color, e.g. '9966CC'
- `border(Top|Right|Bottom|Left)Size` Border size in twips
- `border(Top|Right|Bottom|Left)Color` Border color, e.g. '9966CC'
- `gridSpan` Number of columns spanned
- `vMerge` *restart* or *continue*

5.4.2 Cell span

You can span a cell on multiple columns by using `gridSpan` or multiple rows by using `vMerge`.

```
$cell = $table->addCell(200);
$cell->getStyle()->setGridSpan(5);
```

See `Sample_09_Tables.php` for more code sample.

5.5 Images

To add an image, use the `addImage` method to sections, headers, footers, textruns, or table cells.

```
$section->addImage($src, [$style]);
```

- source String path to a local image or URL of a remote image
- styles Array fo styles for the image. See below.

Examples:

```
$section = $phpWord->addSection();
$section->addImage(
    'mars.jpg',
    array(
        'width' => 100,
        'height' => 100,
        'marginTop' => -1,
        'marginLeft' => -1,
        'wrappingStyle' => 'behind'
    )
);
$footer = $section->addFooter();
$footer->addImage('http://example.com/image.php');
$textrun = $section->addTextRun();
$textrun->addImage('http://php.net/logo.jpg');
```

5.5.1 Image styles

Available image styles:

- `width` Width in pixels
- `height` Height in pixels
- `align` Image alignment, *left*, *right*, or *center*
- `marginTop` Top margin in inches, can be negative
- `marginLeft` Left margin in inches, can be negative
- `wrappingStyle` Wrapping style, *inline*, *square*, *tight*, *behind*, or *infront*

5.5.2 Watermarks

To add a watermark (or page background image), your section needs a header reference. After creating a header, you can use the `addWatermark` method to add a watermark.

```
$section = $phpWord->addSection();
$header = $section->addHeader();
$header->addWatermark('resources/_earth.jpg', array('marginTop' => 200, 'marginLeft' => 55));
```

5.6 Objects

You can add OLE embeddings, such as Excel spreadsheets or PowerPoint presentations to the document by using `addObject` method.

```
$section->addObject($src, [$style]);
```

5.7 Table of contents

To add a table of contents (TOC), you can use the `addTOC` method. Your TOC can only be generated if you have add at least one title (See "Titles").

```
$section->addTOC([$fontStyle], [$tocStyle], [$minDepth], [$maxDepth]);
```

- `$fontStyle`: See font style section
- `$tocStyle`: See available options below
- `$minDepth`: Minimum depth of header to be shown. Default 1
- `$maxDepth`: Maximum depth of header to be shown. Default 9

Options for `$tocStyle`:

- `tabLeader` Fill type between the title text and the page number. Use the defined constants in PHPWord_Style_TOC.
- `tabPos` The position of the tab where the page number appears in twips.
- `indent` The indent factor of the titles in twips.

5.8 Footnotes & endnotes

You can create footnotes with `addFootnote` and endnotes with `addEndnote` in texts or textruns, but it's recommended to use textrun to have better layout. You can use `addText`, `addLink`, `addTextBreak`, `addImage`, `addObject` on footnotes and endnotes.

On textrun:

```
$textrun = $section->addTextRun();
$textrun->addText('Lead text.');
$footnote = $textrun->addFootnote();
$footnote->addText('Footnote text can have ');
$footnote->addLink('http://test.com', 'links');
$footnote->addText('.');
$footnote->addTextBreak();
$footnote->addText('And text break.');
$textrun->addText('Trailing text.');
$endnote = $textrun->addEndnote();
$endnote->addText('Endnote put at the end');
```

On text:

```
$section->addText('Lead text.');
$footnote = $section->addFootnote();
$footnote->addText('Footnote text.');
```

The footnote reference number will be displayed with decimal number starting from 1. This number use `FooterReference` style which you can redefine by `addFontStyle` method. Default value for this style is `array('superScript' => true);`

5.9 Checkboxes

Checkbox elements can be added to sections or table cells by using `addCheckBox`.

```
$section->addCheckBox($name, $text, [$fontStyle], [$paragraphStyle])
```

- `$name` Name of the check box.
- `$text` Text following the check box
- `$fontStyle` See "Font style" section.
- `$paragraphStyle` See "Paragraph style" section.

5.10 Textboxes

To be completed

5.11 Fields

To be completed

5.12 Line

Line elements can be added to sections by using `addLine`.

```
$linestyle = array('weight' => 1, 'width' => 100, 'height' => 0, 'color' => 635552);
$section->addLine($lineStyle)
```

Available line style attributes:

- `weight` Line width in twips
- `color` Defines the color of stroke
- `dash` Line types: dash, rounddot, squaredot, dashdot, longdash, longdashdot, longdashdotdot
- `beginArrow` Start type of arrow: block, open, classic, diamond, oval
- `endArrow` End type of arrow: block, open, classic, diamond, ovel
- `width` Line-object width in pt
- `height` Line-object height in pt
- `flip` Flip the line element: true, false

CHAPTER 6

Templates

You can create a docx template with included search-patterns that can be replaced by any value you wish. Only single-line values can be replaced. To load a template file, use the `loadTemplate` method. After loading the docx template, you can use the `setValue` method to change the value of a search pattern. The search-pattern model is: `${search-pattern}`. It is not possible to add new PHPWord elements to a loaded template file.

Example:

```
$template = $phpWord->loadTemplate('Template.docx');
$template->setValue('Name', 'Somebody someone');
$template->setValue('Street', 'Coming-Undone-Street 32');
```

See `Sample_07_TemplateCloneRow.php` for example on how to create multirow from a single row in a template by using `cloneRow`.

See `Sample_23_TemplateBlock.php` for example on how to clone a block of text using `cloneBlock` and delete a block of text using `deleteBlock`.

CHAPTER 7

Writers & readers

7.1 OOXML

The package of OOXML document consists of the following files.

- _rels/
  - .rels
- docProps/
  - app.xml
  - core.xml
  - custom.xml
- word/
  - rels/
    - document.rels.xml
  - media/
  - theme/
    - theme1.xml
  - document.xml
  - fontTable.xml
  - numbering.xml
  - settings.xml
  - styles.xml
  - webSettings.xml
- [Content_Types].xml

7.2 OpenDocument

7.2.1 Package

The package of OpenDocument document consists of the following files.

- META-INF/
  - manifest.xml
- Pictures/
- content.xml
- meta.xml
- styles.xml

7.2.2 content.xml

The structure of `content.xml` is described below.

- office:document-content
  - office:font-facedecls
  - office:automatic-styles
  - office:body
    - office:text
      - draw:*
      - office:forms
      - table:table
      - text:list
      - text:numbered-paragraph
      - text:p
      - text:table-of-contents
      - text:section
    - office:chart
    - office:image
    - office:drawing

7.2.3 styles.xml

The structure of `styles.xml` is described below.

- office:document-styles
  - office:styles
  - office:automatic-styles

    – office:master-styles
        * office:master-page

7.3 RTF

To be completed.

7.4 HTML

To be completed.

7.5 PDF

To be completed.

CHAPTER 8

Recipes

8.1 Create float left image

Use absolute positioning relative to margin horizontally and to line vertically.

```
$imageStyle = array(
    'width' => 40,
    'height' => 40,
    'wrappingStyle' => 'square',
    'positioning' => 'absolute',
    'posHorizontalRel' => 'margin',
    'posVerticalRel' => 'line',
);
$textrun->addImage('resources/_earth.jpg', $imageStyle);
$textrun->addText($lipsumText);
```

8.2 Download the produced file automatically

Use `php://output` as the filename.

```
$phpWord = new \PhpOffice\PhpWord\PhpWord();
$section = $phpWord->createSection();
$section->addText('Hello World!');
$file = 'HelloWorld.docx';
header("Content-Description: File Transfer");
header('Content-Disposition: attachment; filename="' . $file . '"');
header('Content-Type: application/vnd.openxmlformats-officedocument.wordprocessingml.document');
header('Content-Transfer-Encoding: binary');
header('Cache-Control: must-revalidate, post-check=0, pre-check=0');
header('Expires: 0');
$xmlWriter = \PhpOffice\PhpWord\IOFactory::createWriter($phpWord, 'Word2007');
$xmlWriter->save("php://output");
```

8.3 Create numbered headings

Define a numbering style and title styles, and match the two styles (with `pStyle` and `numStyle`) like below.

```
$phpWord->addNumberingStyle(
    'hNum',
    array('type' => 'multilevel', 'levels' => array(
        array('pStyle' => 'Heading1', 'format' => 'decimal', 'text' => '%1'),
        array('pStyle' => 'Heading2', 'format' => 'decimal', 'text' => '%1.%2'),
        array('pStyle' => 'Heading3', 'format' => 'decimal', 'text' => '%1.%2.%3'),
        )
    )
);
$phpWord->addTitleStyle(1, array('size' => 16), array('numStyle' => 'hNum', 'numLevel' => 0));
$phpWord->addTitleStyle(2, array('size' => 14), array('numStyle' => 'hNum', 'numLevel' => 1));
$phpWord->addTitleStyle(3, array('size' => 12), array('numStyle' => 'hNum', 'numLevel' => 2));

$section->addTitle('Heading 1', 1);
$section->addTitle('Heading 2', 2);
$section->addTitle('Heading 3', 3);
```

CHAPTER 9

Frequently asked questions

9.1 Is this the same with PHPWord that I found in CodePlex?

No. This one is much better with tons of new features that you can't find in PHPWord 0.6.3. The development in CodePlex is halted and switched to GitHub to allow more participation from the crowd. The more the merrier, right?

9.2 I've been running PHPWord from CodePlex flawlessly, but I can't use the latest PHPWord from GitHub. Why?

PHPWord requires PHP 5.3+ since 0.8, while PHPWord 0.6.3 from CodePlex can run with PHP 5.2. There's a lot of new features that we can get from PHP 5.3 and it's been around since 2009! You should upgrade your PHP version to use PHPWord 0.8+.

CHAPTER 10

Credits

CHAPTER 11

References

11.1 ISO/IEC 29500, Third edition, 2012-09-01

- Part 1: Fundamentals and Markup Language Reference
- Part 2: Open Packaging Conventions
- Part 3: Markup Compatibility and Extensibility
- Part 4: Transitional Migration Features

11.2 Formal specifications

- Oasis OpenDocument Standard Version 1.2
- Rich Text Format (RTF) Specification, version 1.9.1

11.3 Other resources

- DocumentFormat.OpenXml.Wordprocessing Namespace on MSDN

CHAPTER 12

Indices and tables

- *genindex*
- *modindex*
- *search*