dynadocx

DYNADOCX V1.1.0
DOCUMENTATION

www.dynadocx.com

CONTENTS

GETTING STARTED

A brief guide to work with **dynadocx**.

INTRODUCTION

dynadocx is a comprehensive tool to generate and modify **Word (DOCX) documents** with HTML, CSS and JSON. Using these standards allow to generate almost any kind of document in seconds from scratch or using a DOCX as a template.

Compatible with **Windows, Linux** and **macOS**, **dynadocx** can be run as a command or be integrated in any coding language.

DOWNLOAD

The Try page has a **trial version** of **dynadocx** available for download for the following operating systems: Windows (64-bit and 32-bit), Linux (64-bit and ARM64) and macOS (64-bit). The trial version allows to work with all the **dynadocx** functionalities, but it adds a watermark at the end of every section of the generated documents.

Purchasing a license, be it of a **Subscription** or a **Perpetual** type, **allows to download** its package, which is available in the user's dashboard. Paid licenses do not add watermarks on the documents.

INSTALLATION

After downloading the **dynadocx** package, be it a trial version or a paid one from the dashboard, you have to **extract it** with a decompression tool for ZIP files such as WinZIP, WinRAR, 7-ZIP, unzip, or any other.

Once the files are unzipped you can **start working** with **dynadocx**.

To execute **dynadocx** it is required to assign **execution permissions** to the bin/dynadocx folder. However, if you want to execute it from anywhere without a reference to the absolute path it is advisable to add it to a directory of the PATH of the operating system.

By default, macOS **does not allow** working with apps from third parties, not downloaded from the Apple App Store or non-verified developers, unless stated otherwise on the preferences. macOS shows an alert regarding this when attempting the task.

Given that case, to work with the **dynadocx** package it is necessary to grant it permission. Go to **System Preferences > Security and privacy > Allow apps downloaded from** to allow macOs to access the package. You can set back the security permissions after installing it succesfully.

First Document

How to generate a document with **dynadocx**.

Adding contents

Create a file or URL with the **HTML/CSS content** to be added to the document:

```
<style>
p {
  font-size: 12px;
  font-family: Arial;
}
</style>
<p>Hello world</p>
```

Generate a DOCX adding its **HTML/CSS as input**:

```
dynadocx -i content.html -o document_output.docx
```

Apply settings

Create a file or URL with **JSON format** with the settings to be applied:

```
{
  "design": {
    "page": {
      "color": "DEEAF6"
    }
  }
}
```

Generate a DOCX, adding **JSON as data**:

```
dynadocx -i content.html -d design.json -o document_output.docx
```

USING A TEMPLATE

Create a new DOCX using MS Word, LibreOffice, Google Docs or any other DOCX editor, with **one or more placeholders** (variables). The format to define a placeholder (variable) in a DOCX is ${VAR}.

Create a file or URL with JSON format **with the replacements** to be done:

```
{
  "template": {
    "replace_texts": [
      {
        "placeholders": [
          "VAR_1",
          "VAR_2",
        ],
        "values": [
          "New value 1",
          "New value 2",
        ]
      }
    ]
  }
}
```

Generate a DOCX with **JSON as data** and stating the template to be used:

```
dynadocx -d data.json -t template.docx -o document_output.docx
```

Arguments -i and -d allow using from 1 to N values.

AVAILABLE OPTIONS

List of arguments available when running **dynadocx**.

ARGUMENTS

A set of options are available to work with the documents:

Option	Content	Default	Description
`--action` `-a`	string	default	Optional. Supported values: • default: default value, used to work with documents. • get_variables: returns a JSON with the existing placeholders (variables) in a DOCX. • get_styles: returns a JSON with the existing styles in a DOCX. • transform: converts DOCX to PDF and other document formats. Only a single action can be used.
`--data` `-d`	file or URL (JSON)		Optional. Setting and design properties, and template contents. Allows adding multiple values.
`--input` `-i`	file or URL (HTML and CSS)		Optional. Contents and styles to be added. Allows adding multiple values.
`--license` `-l`	string		Required. Only available in purchased licenses.
`--output` `-o`	string	output.docx	Optional. File output.
`--template` `-t`	file or URL (DOCX)	Internal template	Optional. Template to be used.
`-v`		Error level	Optional. Level of verbosity.

`--version` `-V`	Optional. Version information.
`--help` `-h`	Optional. Show help information.

CHANGELOG

What's new in **dynadocx**.

1.1.0

Sep 01, 2021

- *New package: ARM64 Linux.*
- *Actions: get_styles.*
- *HTML tags: a (id, class, data-style, href [anchor and external links], style), br (data-type [empty, column, page, textWrapping]), h1, h2, h3, h4, h5, h6 (id, class, data-style, style), hr (id, class, style), mark (id, class, style), sub (id, class, style), sup (id, class, style).*
- *HTML attributes: data-style (a, h1, h2, h3, h4, h5, h6, p, span tags).*
- *CSS styles: color (name, HEX, RGB) (hr tag), height (cm, in, mm, pc, pt, px) (hr tag), text-align (center, left, right) (hr tag), vertical-align (baseline, sub, sup, text-bottom, text-top) (span tags), width (cm, in, mm, pc, pt, px) (hr tag).*
- *Supported decimal digits, - and _ in attribute names.*
- *Base CSS styles: mark {background-color: yellow;}.*
- *Encode special characters automatically from JSON contents.*
- *Support @import url at-rule.*
- *id attribute in p and heading tags generates a bookmark.*
- *New checks when using LibreOffice without setting a custom path.*

1.0.0

Apr 01, 2021

- *Actions: default, get_variables, transform.*
- *HTML tags and attributes: p (id, class, style), span (id, class, style), i (id, class, style), u (id, class, style), em (id, class, style), cite (id, class, style), b (id, class, style), strong (id, class, style), ins (id, class, style), del (id, class, style), strike (id, class, style), s (id, class, style).*
- *CSS styles: background-color (name, HEX, RGB) (p, span tags), color (name, HEX, RGB) (p, span tags), font-size (cm, in, mm, pc, pt, px) (p,*

span tags), font-family (p, span tags), font-style (italic, oblique, normal) (p, span tags), font-width (bold, bolder, 700, 800, 900, normal) (p, span tags), page-break-before (always, auto) (p tags) text-decoration (underline, line-through, none) (p, span tags).

- *Layout options: columns, height, margins, size, orientation, width.*
- *Template: remove_placeholders, replace_checkboxes, replace_lists, replace_tables, replace_texts.*
- *Blocks: clone_block, remove_block_placeholders, remove_blocks.*
- *Design page: color, borders.*
- *Properties: app_version, auto_created_at, auto_modified_at, category, content_status, created_at, creator, description, keywords, last_modified_by, modified_at, revision, subject, title.*
- *Settings: align_borders_and_edges, decimal_symbol, do_not_shade_form_data, do_not_track_formatting, do_not_track_moves, hide_grammatical_errors, hide_spelling_errors, lang, mirror_margins, proof_state, track_revisions, update_fields, view.*
- *Read JSON, HTML, CSS and DOCX contents from the filesystem and as URL (http and https).*
- *Support the following content types when doing replacements: body, headers, footers, footnotes, endnotes and comments.*
- *Transform options using LibreOffice program: DOCX to PDF, ODT, RTF and DOC; DOC, ODT and RTF to DOCX.*

CONTENTS AND STYLES

INTRODUCTION

dynadocx accepts **HTML tags** and **CSS styles** for adding new contents in documents.

Besides compatible, standard tags and styles, **dynadocx** also supports many other **custom contents and styles** in order to get access to the MS Word functionalities that do not have a direct match with the tags, attributes or styles set in the standard HTML/CSS.

Not all HTML tags and CSS styles are compatible with **dynadocx**. The page Contents and Styles Reference details the supported tags, attributes, properties, styles and values.

DOCX documents use UTF-8 as charset. **It is highly recommended** that all added contents use UTF-8 directly. In case of using a different charset, **dynadocx** will automatically try to transform it to UTF-8.

REFERENCE

Contents and styles **are added** with a file or URL HTML/CSS and the "-i" parameter:

```
dynadocx -i content.html
```

It is possible to include **one or more** content files with the same command.

COMPATIBLE HTML TAGS

tag	description
<a>	link
**<b>, **	bold text
** **	line, page and column break
<cite>, , <i>	italic text
<del>, <s>, <strike>	line through text
<ins>, <u>	underline text
<h1>, <h2>, <h3>, <h4>, <h5>, <h6>	heading
<hr>	horizontal line
<mark>	highlighted text
<p>	paragraph
****	inline container
<sub>	subscript text
<sup>	superscript text

COMPATIBLE CSS SELECTORS

selector	description
element	tag {style: value;}
class	.class {style: value;}
id	#id {style: value;}
element.class	tag.class {style: value;}
element#id	tag#id {style: value;}
element, element	tag, tag {style: value;}

DETAIL OF COMPATIBLE TAGS, ATTRIBUTES AND STYLES

<A>

attribute	description
class	one or more classnames
data-style	custom character style (rStyle). Default: Hyperlink
href	unique id
id	external link (http://domain) or bookmark link (#id)
style	inline CSS styles

style	description
background-color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
font-family	Font family. Needs to be available in the OS or embedded in the document. Inherit.
font-size	cm, in, mm, pc, pt, px. Inherit.
font-style	Font style: italic, normal, oblique. Inherit.
font-weight	Font weight: bold, bolder, normal, 700, 800, 900. Inherit.
text-decoration	Text decoration: underline, line-through, none. Inherit.
vertical-align	Vertical align: baseline, sub, sup, text-bottom, text-top.

attribute	description
data-type	Break type: empty, column, page, textWrapping.

<H1>, <H2>, <H3>, <H4>, <H5>, <H6>

attribute	description
class	one or more classnames
data-style	custom paragraph style (pStyle). Default: Heading1 (<h1>), Heading2 (<h2>), Heading3 (<h3>), Heading4 (<h4>), Heading5 (<h5>), Heading6 (<h6>)
id	unique id. Generate a bookmark
style	inline CSS styles

style	description
background-color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
font-family	Font family. Needs to be available in the OS or embedded in the document. Inherit.
font-size	cm, in, mm, pc, pt, px. Inherit.
font-style	Font style: italic, normal, oblique. Inherit.
font-weight	Font weight: bold, bolder, normal, 700, 800, 900. Inherit.
page-break-before	Do a page break: always, auto.
text-align	Text alignment: center, distribute, justify, left, right. Inherit.
text-decoration	Text decoration: underline, line-through, none. Inherit.

<HR>

Attribute	description
class	one or more classnames
id	unique id. Generate a bookmark
style	inline CSS styles

style	description
color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit. Default: ACA889
height	Height: cm, in, mm, pc, pt, px. Default: 1.5
text-align	Text alignment: center, left, right. Inherit. Default: center
width	Width: cm, in, mm, pc, pt, px. Default: full width

<P>

attribute	description
class	one or more classnames
data-style	custom paragraph style (pStyle)
id	unique id
style	inline CSS styles

style	description
background-color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
font-family	Font family. Needs to be available in the OS or embedded in the document. Inherit.
font-size	cm, in, mm, pc, pt, px. Inherit.
font-style	Font style: italic, normal, oblique. Inherit.
font-weight	Font weight: bold, bolder, normal, 700, 800, 900. Inherit.
page-break-before	Do a page break: always, auto.
text-align	Text alignment: center, distribute, justify, left, right. Inherit.
text-decoration	Text decoration: underline, line-through, none. Inherit.

<SPAN>

(<b>, <cite>, <del>, , <i>, <ins>, <mark>, <s>, <strike>, , <sub>, <sup>, <u>)

attribute	description
class	one or more classnames
data-style	custom character style (rStyle)
id	unique id
style	inline CSS styles

style	description
background-color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
color	Color: name, HEX, rgb, rgba (alpha value is ignored). Inherit.
font-family	Font family. Needs to be available in the OS or embedded in the document. Inherit.
font-size	Font size: cm, in, mm, pc, pt, px. Inherit.
font-style	Font style: italic, normal, oblique. Inherit.
font-weight	Font weight: bold, bolder, normal, 700, 800, 900. Inherit.
text-decoration	Text decoration: underline, line-through, none. Inherit.
vertical-align	Vertical align: baseline, sub, sup, text-bottom, text-top.

How to add CSS

CSS can be added using the following ways:

Inline CSS

```
<p style="color: red;">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua.</p>
```

Internal CSS (style tags)

```
<head>
  <style>
      p {
          font-weight: bold;
      }
  </style>
</head>
```

External CSS

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
```

```
<head>
  <link rel="stylesheet" href="https://www.domain.com/styles.css">
</head>
```

Import CSS

```
<head>
  <style>
      @import "styles.css";
  </style>
</head>
```

```
<head>
  <style>
      @import url("https://domain.com/styles.css");
  </style>
</head>
```

BASE CSS

The following default styles are applied when importing HTML:

```
mark {
    background-color: yellow;
}
```

SAMPLES

ADD HTML/CSS USING INLINE STYLES:

```
<p>Lorem Ipsum dolor sit amet, consectetur adipiscing elit, sed do
eiusmod tempor incididunt ut labore et dolore magna aliqua.</p>
<p style="color: red;">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua.</p>
<p style="text-align: justify;"><span style="font-size: 18px; font-
weight: bold;">Lorem Ipsum</span> dolor sit <em>amet</em>,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua.</p>
```

```
dynadocx -i content.html
```

ADD HTML/CSS USING STYLE TAGS:

```
<head>
  <style>
      .strong {
          font-weight: bold;
      }

      p {
          font-family: Arial;
      }

      p.hl {
          font-family: "Times New Roman";
          font-size: 20px;
          text-align: justify;
      }

      span.hl {
          color: #CCBB11;
          text-decoration: underline;
          font-style: italic;
      }
  </style>
</head>
<p><span class="strong">Lorem Ipsum</span> dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua.</p>
<p class="hl strong">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua.</p>
<p><span class="hl">Lorem Ipsum</span> dolor sit <em>amet</em>,
consectetur <span class="hl">adipiscing elit</span>, sed do eiusmod
tempor incididunt ut labore et dolore <span style="text-decoration:
line-through;">magna aliqua</span>.</p>
```

```
dynadocx -i content.html
```

ADD HTML/CSS USING LINK TAGS:

```
<head>
  <link rel="stylesheet" href="styles.css">
</head>
<p><span class="strong">Lorem Ipsum</span> dolor sit amet,
consectetur adipiscing elit, sed do eiusmod tempor incididunt ut
labore et dolore magna aliqua.</p>
<p class="hl strong">Lorem ipsum dolor sit amet, consectetur
adipiscing elit, sed do eiusmod tempor incididunt ut labore et
dolore magna aliqua.</p>
<p><span class="hl">Lorem Ipsum</span> dolor sit <em>amet</em>,
consectetur <span class="hl">adipiscing elit</span>, sed do eiusmod
tempor incididunt ut labore et dolore <span style="text-decoration:
line-through;">magna aliqua</span>.</p>
```

```
dynadocx -i content.html -t template.docx
```

TEMPLATES

INTRODUCTION

Besides generating documents from scratch, **dynadocx** has many options for **working with templates**. This allows to change existing contents in documents: texts, tables, lists, checkboxes and others.

There are other **additional options** available for working with placeholders (variables). E.g., cleaning variables or limiting the replacements to be performed.

Use the syntax **${ }** to define placeholders (variables). E.g.: ${VAR}, ${NAME}, ${value}, ${new_content}...

Any document **can function** as a template. It is possible to generate documents with MS Word, LibreOffice, GoogleDrive or any other DOCX editor.

DOCX documents use UTF-8 as charset. **It is highly recommended** that all added contents use UTF-8 directly. In case of using a different charset, **dynadocx** will automatically try to transform it to UTF-8.

REFERENCE

Replacements and options applied to templates **are defined** with the parameter "-d" and a JSON file. To indicate the DOCX document that will work as a template use the parameter "-t":

```
dynadocx -d data.json -t template.docx
```

The following is the **complete set** of key/values admitted with their default values:

```
{
  "template": {
    "remove_placeholders": false,
    "replace_checkboxes": [
      {
        "placeholders": [
          ""
        ],
        "values": [
          ""
        ],
        "options": {
          "content_type": "all",
          "max_replacements": ""
        }
      }
    ],
    "replace_lists": [
      {
        "placeholders": [
          ""
        ],
        "values": [
          [
            ""
          ]
        ],
        "options": {
          "content_type": "all",
          "max_replacements": ""
        }
      }
    ],
    "replace_tables": [
      {
        "placeholders": [
          [
            ""
          ]
        ],
        "values": [
          [
            [
              ""
            ]
          ]
        ],
        "options": {
          "content_type": "all",
          "max_replacements": ""
        }
      }
    ],
```

```
    "replace_texts": [
      {
        "placeholders": [
          ""
        ],
        "values": [
          ""
        ],
        "options": {
          "content_type": "all",
          "max_replacements": ""
        }
      }
    ]
  }
}
```

DETAIL OF AVAILABLE OPTIONS

REMOVE_PLACEHOLDERS

If true, it **deletes the document placeholders**. This option is the last one to be executed in order to avoid erasing placeholders needed to be replaced with the rest of options.

REPLACE_CHECKBOXES

Check and uncheck checkboxes (content control and legacy form field).

Key	value
placeholders	Placeholder names, without ${ }. Placeholders **are assigned** in the checkbox properties in the Tag (content control checkboxes) and in the Help Text (legacy form field checkboxes). To check these properties in some MS Word versions it may be necessary to enable Design Mode.
values	New values: true (checked) or false (unchecked). Each value **matches a placeholder** of the previous key (placeholders) depending on its position.
content_type	Content type to replace the placeholder: all, document, headers, footers, footnotes, endnote, comments.
max_replacements	Not zero-based, replaces first N matches. If not set, replace all.

REPLACE_LISTS

Replace placeholders **by text values** cloning the list.

Key	value
placeholders	Placeholder names, without ${ }. Each set of values **matches an existing placeholder** in the list designed to be cloned and replaced.
values	New values. Each set of values **matches the placeholders** of every cloned list of the previous key (placeholders).
content_type	Content type **to replace** the placeholder: all, document, headers, footers, footnotes, endnote, comments.
max_replacements	Not zero-based, **replaces first N** matches. If not set, replace all.

REPLACE_TABLES

Replace placeholders **by text values** cloning the rows.

Key	value
placeholders	Placeholder names, without ${ }. Each set of values **matches the existing placeholders** in the row set to be cloned and replaced.
values	New values. Each set of values **matches the placeholders** of every row cloned of the previous key (placeholders).
content_type	Content type to **replace the placeholder**: all, document, headers, footers, footnotes, endnote, comments.
max_replacements	Not zero-based, replaces **first N matches**. If not set, replace all.

REPLACE_TEXTS

Replaces placeholders **by text values**.

Key	value
placeholders	Placeholder names, without ${ }. Every placeholder to be replaced **must have a value** assigned in the following key (values).
values	New values. Every values **matches a placeholder** of the previous key (placeholders) depending on its position.
content_type	Content type to **replace the placeholder**: all, document, headers, footers, footnotes, endnote, comments.
max_replacements	Not zero-based, replaces **first N matches**. If not set, replace all.

Samples

- Replace a placeholder by a text value in all content types:

```
{
  "template": {
    "replace_texts": [
      {
        "placeholders": [
          "VAR_1"
        ],
        "values": [
          "New value"
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE THREE PLACEHOLDERS BY TEXT VALUES IN ALL CONTENT TYPES:

```
{
  "template": {
    "replace_texts": [
      {
        "placeholders": [
          "VAR_1",
          "VAR_2",
          "VAR_3"
        ],
        "values": [
          "New value 1",
          "New value 2",
          "New value 3"
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

• Replace placeholders by text values in headers and footers content types:

```
{
  "template": {
    "replace_texts": [
      {
        "placeholders": [
          "VAR_1",
          "VAR_2",
        ],
        "values": [
          "New value 1",
          "New value 2"
        ],
        "options": {
          "content_type": "headers"
        }
      },
      {
        "placeholders": [
          "VAR_3"
        ],
        "values": [
          "New value 3"
        ],
        "options": {
          "content_type": "footers"
        }
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REMOVE ALL PLACEHOLDERS:

```
{
  "template": {
    "remove_placeholders": true,
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE TWO PLACEHOLDERS BY TEXT VALUES AND THEN REMOVE ALL PENDING PLACEHOLDERS:

```
{
  "template": {
    "replace_texts": [
      {
        "placeholders": [
          "VAR_1",
          "VAR_2"
        ]
        "values": [
          "New value 1",
          "New value 2"
        ]
      }
    ],
    "remove_placeholders": true,
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE PLACEHOLDERS IN A TABLE:

```
{
  "template": {
    "replace_tables": [
      {
        "placeholders": [
          [
            "NAME",
            "STREET",
            "PHONE"
          ],
          [
            "INFO1",
            "INFO2"
          ]
        ],
        "values": [
          [
            [
              "John",
              "Mary",
              "Tom",
              "Andy"
            ],
            [
              "Street 1,2B",
              "Avenue 89",
              "Street 6T",
              "Avenue 12"
            ],
            [
              "+00 1002",
              "03 2112",
              "+00 122 1002",
              "+00 122 1003"
            ]
          ],
          [
            [
              "Info A",
              "Info B"
            ],
            [
              "Description A",
              "Description B"
            ]
          ]
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE PLACEHOLDERS IN A LIST:

```
{
  "template": {
    "replace_lists": [
      {
        "placeholders": [
          "VAR_1",
          "VAR_2"
        ],
        "values": [
          [
            "New value 1",
            "New value 2",
            "New value 3"
          ],
          [
            "New value A",
            "New value B",
            "New value C"
          ]
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE PLACEHOLDERS IN A LIST WITH MULTIPLE LEVELS:

```
{
  "template": {
    "replace_lists": [
      {
        "placeholders": [
          "VAR"
        ],
        "values": [
          [
            "New value 1",
            [
              "New value 1.1",
              "New value 1.2",
              "New value 1.3",
              [
                "New value 1.1.1",
                "New value 1.1.2",
                "New value 1.1.3"
              ],
              "New value 1.4"
            ],
            "New value 2",
            [
              [
                "New value 2.1.1"
              ]
            ],
            "New value 3"
          ]
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

- REPLACE CHECKBOXES VALUES:

```
{
  "template": {
    "replace_checkboxes": [
      {
        "placeholders": [
          "CHECKBOX_1",
          "CHECKBOX_2"
        ],
        "values": [
          [
            true,
            false
          ]
        ]
      }
    ]
  }
}
```

```
dynadocx -d data.json -t template.docx
```

DESIGN AND LAYOUT

INTRODUCTION

Properties of **sections and pages** can be modified with **dynadocx** too: *design* and *layout* are the options to do the task.

The available *design* options set the **page design** like, for instance, color and borders. On the other hand, *layout* contains the **section options** like size, orientation and columns.

These options can be used with both new documents and templates.

DOCX documents use UTF-8 as charset. **It is highly recommended** that all added contents use UTF-8 directly. In case of using a different charset, **dynadocx** will automatically try to transform it to UTF-8.

REFERENCE

Properties and settings are defined with the parameter "-d" and a JSON file:

```
dynadocx -d data.json
```

The following is the **complete set** of values admitted with their default values:

```
{
  "design": {
    "page": {
      "borders": {
        "color": "auto",
        "display": "",
        "offset_from": "page",
        "space": 24,
        "style": "single",
        "width": "4",
        "border_top": {
          "color": "auto",
          "space": 24,
          "style": "single",
          "width": 4
        },
        "border_right": {
          "color": "auto",
          "space": 24,
          "style": "single",
          "width": 4
        },
        "border_bottom": {
          "color": "auto",
          "space": 24,
          "style": "single",
          "width": 4
        },
        "border_left": {
          "color": "auto",
          "space": 24,
          "style": "single",
          "width": 4
        }
      },
      "color": "",
    }
  },
```

```
"layout": {
    "columns": {
      "number": 2,
      "space": 720,
      "columns": [
        {
          "space": 720,
          "width": 2000
        },
        {
          "space": 720,
          "width": 2000
        }
      ]
    },
    "height": 15840,
    "margins": {
      "bottom": 0,
      "gutter": 0,
      "header": 0,
      "footer": 0,
      "left": 0,
      "right": 0,
      "top": 0
    },
    "size": "A4",
    "orientation": "portrait",
    "width": 12240
  }
}
```

DETAIL OF THE AVAILABLE OPTIONS

DESIGN

Key	value
borders	Page borders. It may be defined globally with key borders or individually with border_top, border_right, border_bottom and border_left.
color	Backgroud color and border color. HEX value without #.
display	Pages to display the border: allPages, firstPage, notFirstPage.
offset_from	How the relative positioning of the borders are calculated: page, text.
space	Spacing offset. Specified in points.
style	Border style: single, dashDotStroked, dashed, dashSmallGap, dotDash, dotDotDash, dotted, double, doubleWave, inset, nil, none, outset, thick, thickThinLargeGap, thickThinMediumGap, thickThinSmallGap, thinThickLargeGap, thinThickMediumGap, thinThickSmallGap, thinThickThinLargeGap, thinThickThinMediumGap, thinThickThinSmallGap, threeDEmboss, threeDEngrave, triple, wave.
width	Border width. From 2 to 96. Specified in eighths of a point.

LAYOUT

Key	value
columns	Section columns. Columns may have or have not the same size. The number key states the number. The space key sets the space after the column in twips (twentieths of a point). When assigning different sizes for columns, it specifies a value per column, stating the space and width values in twips (twentieths of a point).
height	Section height. Specified in twips (twentieths of a point).
margins	Section margins: Specified in twips (twentieths of a point).
orientation	Section orientation: landscape, portrait.
size	Preset sizes: A4, A4-landscape, A3, A3-landscape, letter, letter-landscape, legal, legal-landscape. These preset sizes can be customized using the following keys: height, width, orientation and margin.
width	Section width. Specified in twips (twentieths of a point).

Samples

- Set a background color:

```
{
  "design": {
    "page": {
      "color": "DEEAF6"
    }
  }
}
```

```
dynadocx -d data.json
```

- Set page borders:

```
{
  "design": {
    "page": {
      "borders": {
        "color": "F7CAAC",
        "offset_from": "page",
        "space": 30,
        "style": "dotted",
        "width": 28
      }
    }
  }
}
```

```
dynadocx -d data.json
```

- SET PAGE BORDERS WITH CUSTOM TOP AND BOTTOM BORDERS:

```
{
  "design": {
    "page": {
      "borders": {
        "color": "F7CAAC",
        "offset_from": "page",
        "space": 30,
        "style": "dotted",
        "width": 12,
        "border_top": {
          "color": "000000",
          "style": "dashed",
          "width": 24
        },
        "border_bottom": {
          "space": 2,
          "style": "double",
          "width": 24
        }
      }
    }
  }
}
```

```
dynadocx -d data.json
```

- SET COLUMNS:

```
{
  "layout": {
    "columns": {
      "number": 3,
      "space": 120
    }
  }
}
```

```
dynadocx -d data.json
```

- SET CUSTOM COLUMNS:

```
{
  "layout": {
    "columns": {
      "columns": [
        {
          "space": 720,
          "width": 6000
        },
        {
          "space": 720,
          "width": 2000
        }
      ]
    }
  }
}
```

```
dynadocx -d data.json
```

- SET PAGE MARGINS:

```
{
  "layout": {
    "margins": {
      "bottom": 20,
      "gutter": 0,
      "left": 40,
      "right": 40,
      "top": 20
    }
  }
}
```

```
dynadocx -d data.json
```

- SET A PRESET SIZE:

```
{
  "layout": {
    "size": "A3"
  }
}
```

```
dynadocx -d data.json
```

- SET A CUSTOM LAYOUT:

```
{
  "layout": {
    "orientation": "landscape",
    "height": 11906,
    "width": 16838
  }
}
```

```
dynadocx -d data.json
```

- SET A BACKGROUND COLOR IN A TEMPLATE:

```
{
  "design": {
    "page": {
      "color": "C8F296"
    }
  }
}
```

```
dynadocx -d data.json -t document.docx
```

SETTINGS AND PROPERTIES

INTRODUCTION

It is possible to **custom options** which influence the appearance and behavior of the document and fields such as author, description and others, with the help of the *properties* y *settings*.

For instance, **you can define** author, review number, category and title with the *properties*. **And custom** default language, show or hide grammatical errors or update automatically field values when opening the document, all of it with the *settings* options.

These options can be used with both new documents and templates.

DOCX documents use UTF-8 as charset. **It is highly recommended** that all added contents use UTF-8 directly. In case of using a different charset, **dynadocx** will automatically try to transform it to UTF-8.

REFERENCE

Properties and **settings** are defined with the parameter "-d" and a JSON file:

```
dynadocx -d data.json
```

The following is the **complete set** of key/values admitted with their default values:

```
{
  "properties": {
    "app_version": "",
    "auto_created_at": false,
    "auto_modified_at": false,
    "category": "",
    "content_status": "",
    "created_at": "",
    "creator": "",
    "description": "",
    "keywords": "",
    "last_modified_by": "",
    "modified_at": "",
    "revision": "",
    "subject": "",
    "title": ""
  },
  "settings": {
    "align_borders_and_edges": false,
    "decimal_symbol": "",
    "do_not_shade_form_data": false,
    "do_not_track_formatting": false,
    "do_not_track_moves": false,
    "hide_grammatical_errors": false,
    "hide_spelling_errors": false,
    "lang": "en-US",
    "mirror_margins": false,
    "proof_state": "",
    "track_revisions": false,
    "update_fields": false,
    "view": ""
  }
}
```

DETAIL OF AVAILABLE OPTIONS

PROPERTIES

Key	value
app_version	Force a MS Word version: XX.YYYY format.
auto_created_at	Set the current date and time as created_at.
auto_modified_at	Set the current date and time as modified_at.
category	Category field.
content_status	Content status field.
created_at	Force a created at date. W3CDTF without time zone.
creator	Creator field.
last_modified_by	Last modified by field.
modified_at	Force a modified at date. W3CDTF without time zone.
revision	Revision number.
subject	Subject field.
title	Title field.

SETTINGS

Key	value
align_borders_and_edges	Align paragraph and table borders with page border.
decimal_symbol	Radix point for field code evaluation.
do_not_shade_form_data	Do not show visual indicator for form fields.
do_not_track_formatting	Do not track formatting revisions when tracking revisions.
do_not_track_moves	Do not use move syntax when tracking revisions.
hide_grammatical_errors	Do not display visual indication of grammatical errors.
hide_spelling_errors	Do not display visual indication of spelling errors.
lang	language-region
mirror_margins	Mirror page margins.
proof_state	Spelling and grammatical checking state: clean or dirty
track_revisions	Track revisions to document.
update_fields	Automatically recalculate fields on open.
view	Document view setting: masterPages, none, normal, outline, print, web

Samples

- **Change default language:**

```
{
  "settings": {
    "lang": "en-US"
  }
}
```

```
dynadocx -d data.json
```

- **Set custom properties:**

```
{
  "properties": {
    "category": "A custom category",
    "creator": "A custom creator",
    "description": "A custom description",
    "revision": 2,
    "keywords": "keyword1, keyword2, keyword3"
  }
}
```

```
dynadocx -d data.json
```

- **Set current datetime when generating the document:**

```
{
  "properties": {
    "auto_created_at": true,
    "auto_modified_at": true
  }
}
```

```
dynadocx -d data.json
```

• SET CUSTOM SETTINGS AND CUSTOM PROPERTIES IN THE SAME JSON:

```
{
  "properties": {
    "author": "A custom author",
    "creator": "A custom creator"
  },
  "settings": {
    "hide_grammatical_errors": true,
    "hide_spelling_errors": true
  }
}
```

```
dynadocx -d data.json
```

• SET CUSTOM SETTINGS AND CUSTOM PROPERTIES USING TWO JSON:

```
{
  "properties": {
    "author": "A custom author",
    "creator": "A custom creator"
  }
}
```

```
{
  "settings": {
    "hide_grammatical_errors": true,
    "hide_spelling_errors": "true"
  }
}
```

```
dynadocx -d data1.json data2.json
```

• CHANGE THE DEFAULT LANGUAGE AND PROPERTIES FROM A TEMPLATE:

```
{
  "properties": {
    "author": "A custom author",
    "creator": "A custom creator"
  },
  "settings": {
    "lang": "en-US"
  }
}
```

```
dynadocx -d data.json -t document.docx
```

GET PLACEHOLDERS

INTRODUCTION

dynadocx defines placeholders (variables) in templates with the syntax ***${VAR}***, where VAR is the name of the placeholder (variable), which admits any value, and ***${ }*** are the symbols used to distinguish them from the rest of the elements of the document.

Among the available actions when executing **dynadocx** there is *get_variables*, which returns a JSON **with the existing variables** in a DOCX.

REFERENCE

In order to **obtain the placeholders** existing in a DOCX, **dynadocx** uses the action *get_variables* with the parameter "-a" and a call to the template to be read:

```
dynadocx -a get_variables -t document.docx
```

Placeholders are returned **as a JSON** content, arranged by the content type keys: comments, document, endnotes, footers, footnotes and headers.

The following is an example of a **generated JSON**:

```
{
"comments":
  [
    "VAR_COMMENT_1"
  ],
"document":
  [
    "VAR_BODY_1",
    "VAR_BODY_2"
  ],
"endnotes":
  [
    "VAR_ENDNOTE_1"
  ],
"footers":
  [
    "VAR_FOOTER_1",
    "VAR_FOOTER_2"
  ],
"footnotes":
  [
    "VAR_FOOTNOTE_1"
  ],
"headers":
  [
    "VAR_HEADER_1",
    "VAR_HEADER_2"
  ]
}
```

Samples

- Get placeholders from a DOCX:

```
dynadocx -a get_variables -t document.docx
```

GET STYLES

INTRODUCTION

All **documents include** custom styles that can be applied when adding contents. For example, a paragraph can apply a custom paragraph style with preset styles.

dynadocx includes the action **get_styles**, which returns a JSON **with the existing styles** in a DOCX.

REFERENCE

In order to **obtain the styles** existing in a DOCX, **dynadocx** uses the action *get_styles* with the parameter "-a" and a call to the template to be read:

```
dynadocx -a get_styles -t document.docx
```

Styles are returned **as a JSON** content, arranged by the style type.

The following is an example of a **generated JSON**:

```
{
"character":
  [
    "DefaultParagraphFont",
    "HeaderChar",
    "FooterChar",
    "FootnoteTextChar",
    "FootnoteReference",
    "EndnoteTextChar",
    "EndnoteReference",
    "CommentReference",
    "CommentTextChar",
    "CommentSubjectChar",
    "BalloonTextChar"
  ],
"paragraph":
  [
    "Normal",
    "Header",
    "Footer",
    "FootnoteText",
    "EndnoteText",
    "CommentText",
    "CommentSubject",
    "BalloonText"
  ],
"table":
  [
    "TableNormal"
  ]
}
```

Samples

- Get styles from a DOCX:

```
dynadocx -a get_styles -t document.docx
```

- Get styles from the default base template:

```
dynadocx -a get_styles
```

TRANSFORM

INTRODUCTION

HTML and CSS to DOCX is not the only transformation **dynadocx** allows. **Other formats are available** such as PDF, DOC or ODT. **dynadocx** performs transformation between formats with LibreOffice, and so it is compatible with Windows, Linux and macOS. **dynadocx** is not limited to straight transformations with LibreOffice, but also allows modification and adaptation of documents in order to achieve the best possible result.

To do the transformations **it is necessary to install** LibreOffice. It is highly recommended to download and install the latest available version at libreoffice.org.

The following are the supported transformations:

From	To
DOCX	PDF, ODT, DOC, RTF, TXT
DOC, ODT, RTF, TXT	DOCX

REFERENCE

After installing LibreOffice it is possible to do the transformations, provided that the "soffice" command **is defined in the PATH** of the environment variables. In other words, the transformation can be done if that command can be executed from any path.

This way, the transformation works **by indicating** the action *transform* with the parameter "-a", the input file, parameter "-i" and the output file, parameter "-o":

```
dynadocx -a transform -i document.docx -o document.pdf
```

The extension of the output document **establishes the new format** of the document.

If the "soffice" command is unavailable in the global path, it is required to **specify its path** in the option libreoffice_path in a JSON file:

```
{
  "config": {
    "transform": {
      "libreoffice_path": ""
    }
  }
}
```

The following is the **complete set** of key/values admitted:

key	value
libreoffice_path	Path to LibreOffice main exec.

Here follows examples for Windows:

```
{
  "config": {
    "transform": {
      "libreoffice_path": "C:\\Program
Files\\LibreOffice\\program\\soffice.exe"
    }
  }
}
```

for Linux:

```
{
  "config": {
    "transform": {
      "libreoffice_path": "/opt/libreoffice/program/soffice"
    }
  }
}
```

...and for macOS:

```
{
  "config": {
    "transform": {
      "libreoffice_path":
"/Applications/Libreoffice.app/program/soffice"
    }
  }
}
```

When indicating the path to LibreOffice, transformation is done with a reference **to the matching JSON** with the parameter "-d":

```
dynadocx -d data.json -i document.docx -o document.pdf
```

Document transformation only works with documents that exist **in the filesystem**, not with URLs.

Document transformation admits **a single file** as input and another one as output, but not multiple values.

Samples

- **Transform DOCX to PDF:**

```
dynadocx -a transform -i document.docx -o document.pdf
```

- **Transform DOC to DOCX:**

```
dynadocx -a transform -i document.doc -o document.docx
```

- **Transform DOCX to PDF using a relative path as output:**

```
dynadocx -a transform -i ../source/document.docx -o
../../folder/document.pdf
```

- **Transform DOCX to PDF using an absolute path as output:**

```
dynadocx -a transform -i document.docx -o /folder/document.pdf
```

dynadocx

VISIT US FOR MORE INFORMATION

www.dynadocx.com