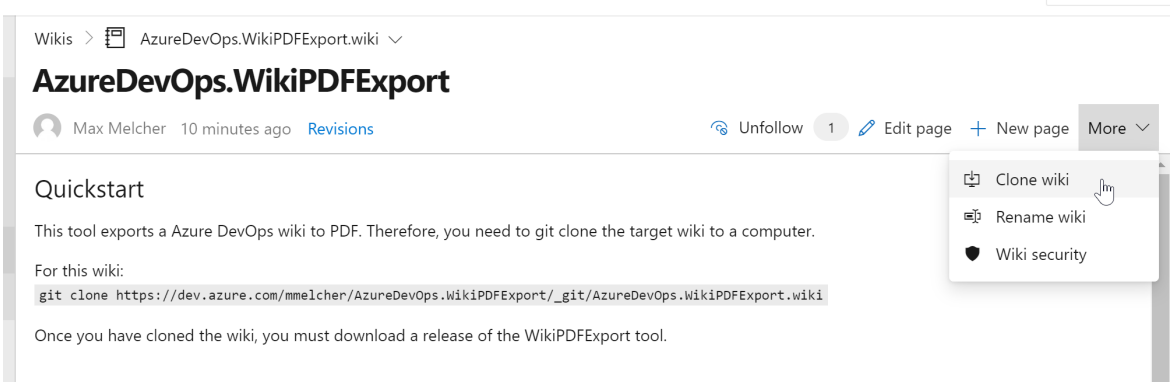
AzureDevOps.WikiPDFExport

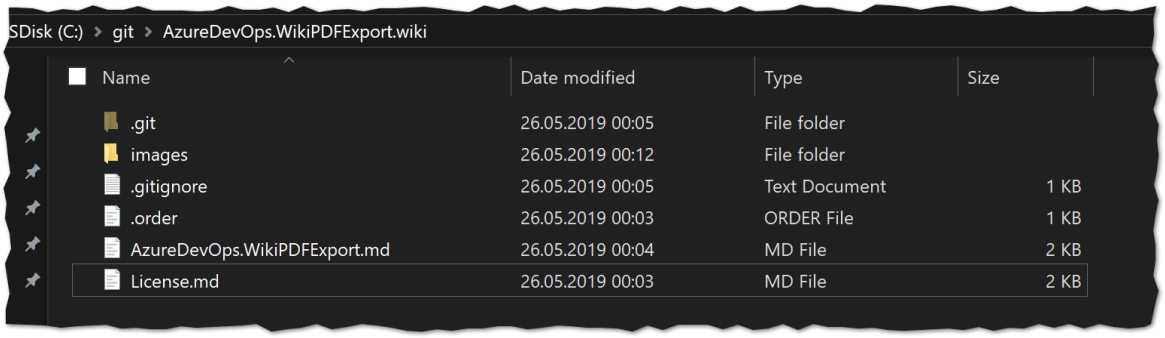
Quickstart

This tool exports a Azure DevOps wiki as PDF. Therefore, you need to git clone the target wiki to a computer. You can get the clone link of the wiki in the top right of the wiki homepage:



To clone this wiki, use the following command: `git clone https://dev.azure.com/mmelcher/AzureDevOps.WikiPDFExport/_git/AzureDevOps.WikiPDFExport.wiki`

The result should look like this:



Once you have cloned the wiki, you must download the Azure DevOps WikiPDFExport tool. azuredevops-export-wiki.exe (~40MB)

You can drop it right into the cloned folder and execute it there. Launched without parameters, the tool will detect all wiki files next to it and convert it to a PDF called export.pdf right next to it. Similar to this pdf.

If you need more control over the output, please see the Configuration Options below or by launching the tool with --help parameter.

ℹ Features

The tool currently supports the following:

- Export all wiki pages (and sub pages) in the correct order including styles and formatting.
- Includes pictures (remote and relative urls)
- Creates PDF bookmarks to all pages for easier navigation within the PDF
- If you link to other wiki pages, the link in the PDF will work, too. Anchor tags supported, see #license or ./TEST%2DPAGE.md#Mermaid)
- Everything self-contained. Download the .exe file, run it, done.
- Tool can be used as part of a build, see Build Task
- It is fast. A PDF with 160 pages is created in less than a second.
- Emoticons and Smileys are supported 😄 ⚠
- Mermaid supported
- Workitems can be referenced and will be included in the pdf as link with the current status, e.g. #7.
- Math/Latex formulas are rendered

⚠ Requirements

The tool is developed as .NET Core 2.2 application, therefore you need to have the runtime installed. Download is available here.

⏬ Download

The download is available here

Configuration Options

-h / --help

Help - outputs the parameters that can be used

-o / --output

The path to the export file including the filename, e.g. c:\export.pdf

-d / --date

The current date will be added to the footer

-b / --breakPage

For every wiki page a new page in the PDF will be created

-h / --heading

For every wiki page create a heading in the PDF. If the file is called Home.md a new #Home-heading is added to PDF.

-s / --single

Path to a single markdown file to convert to PDF. If you want to write your changelog in the wiki, this is your parameter to only convert a single page. -p parameter is required, too.

-p / --path

Path to the wiki folder. If not provided, the current folder of the executable is used.

-v / --verbose

Verbose mode. Logging will added to the console window

-d / --debug

Debug mode. Logs tons of stuff and even exports the intermediate html file

--pathToHeading

Add path of the file to the header

--footer-left, --footer-center, --footer-right, --header-left, --header-center, --header-right,

Headers and footers can be added to the document by the --header-* and --footer* arguments respectfully. In header and footer text string supplied to e.g. --header-left, the following variables will be substituted.

- [page] Replaced by the number of the pages currently being printed
- [frompage] Replaced by the number of the first page to be printed
- [topage] Replaced by the number of the last page to be printed
- [webpage] Replaced by the URL of the page being printed
- [section] Replaced by the name of the current section
- [subsection] Replaced by the name of the current subsection
- [date] Replaced by the current date in system local format
- [isodate] Replaced by the current date in ISO 8601 extended format
- [time] Replaced by the current time in system local format
- [title] Replaced by the title of the of the current page object
- [doctitle] Replaced by the title of the output document
- [sitepage] Replaced by the number of the page in the current site being converted
- [sitepages] Replaced by the number of pages in the current site being converted

⚠ Limitations

So far the following limitations are known:

- TOC (Table of Contents) tag is not supported and will exported as tag
- The tool, sometimes shows an error "Qt: Could not initialize OLE (error 80010106)" - this can be ignored.
- If headers are not formatted properly (#Header instead of # Header), they are rendered incorrectly. I might fix that in the future.
- The tool lacks proper testing because I only have two wikis available

License

See license

Telemetry

The tool uses Application Insights for basic telemetry:

- The duration of the export and the count of wiki pages is tracked and submitted to Azure.
- In the case of an error, the exception is submitted.
- No wiki data/content is submitted.

Thanks

In this tool uses open source libraries that do the actual work - I just combined them to get the export as PDF:

1. CommandLineParser to parse the command line
2. MarkDig to parse markdown files to HTML.
3. DinkToPdf to export HTML to PDF
4. dotnet-warp to release a self-contained exe file

TESTPAGE

The following are tests for the export page

Copy & Paste Picture

Table of Contents

Video

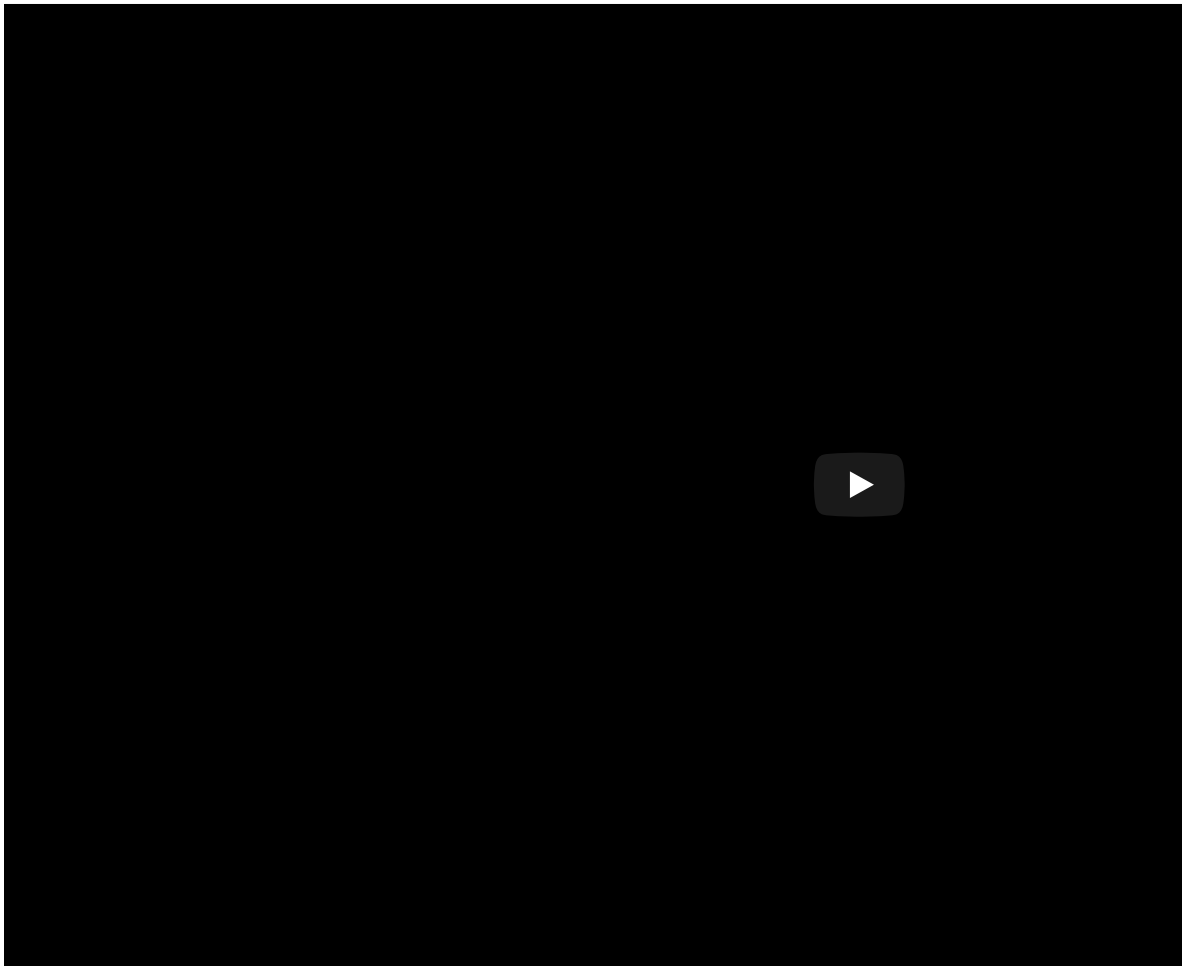


Table Test

a	b
0	1

Formular

$$e = mc^2$$

Mention

@<7C0C20F7-3AB6-6232-8E80-E35152712AF5>

##Malformed header (no space!)

Lists

- 1 Entry
- 2 Entry
- 3 Entry

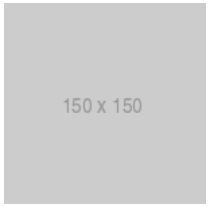
Formats

bold *italic* link `code`

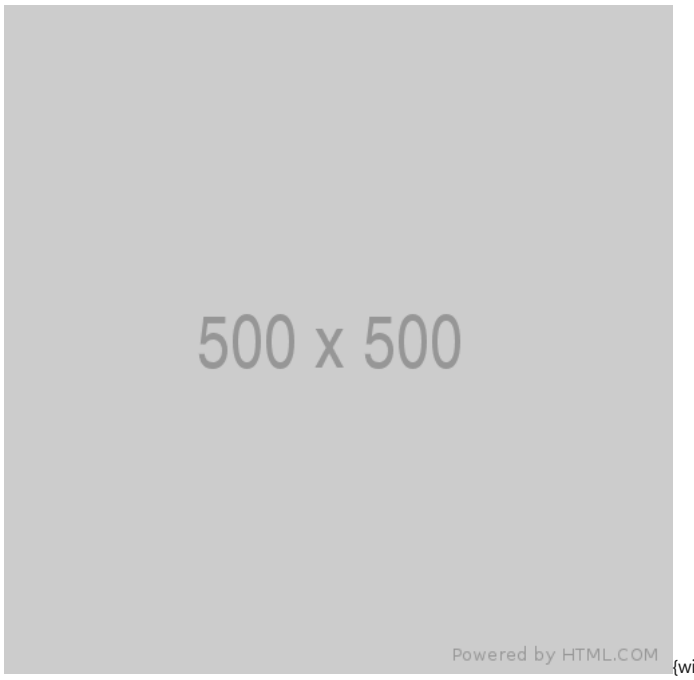
Mermaid

Large Pictures

Pictures with Space in Filename

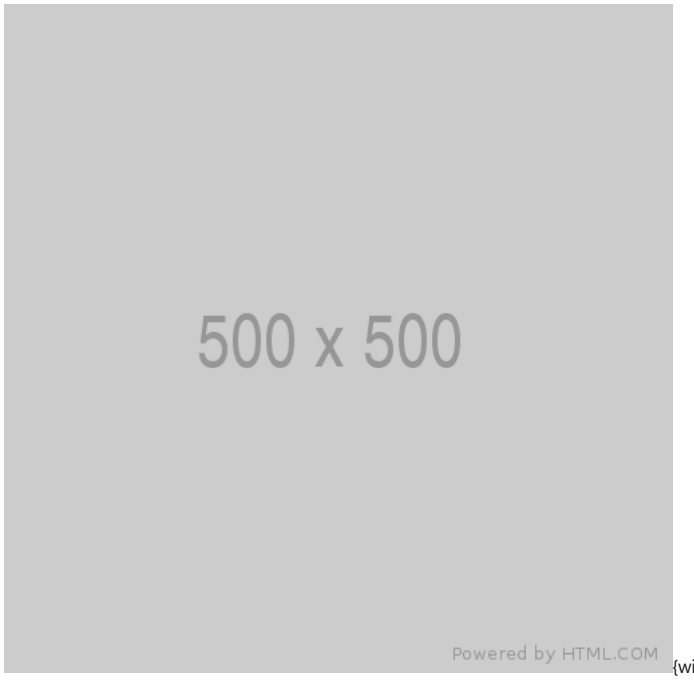


Pictures with Width/Height



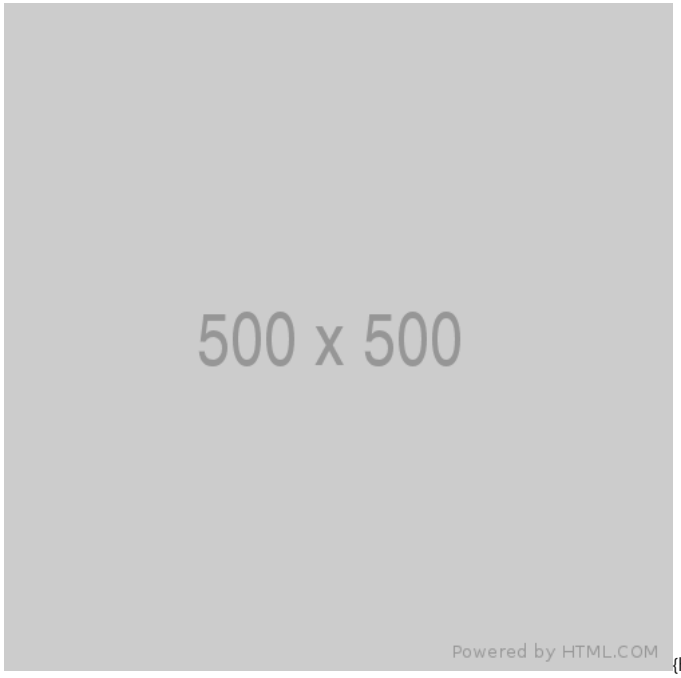
{width=250 height=250}

Pictures with Width only



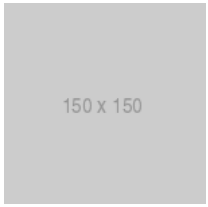
{width=300}

Pictures with Height only



{height=150}

Picture as html



Mathematical notation and characters

Supported in: Pull Requests | Wikis

Both inline and block KaTeX notation is supported in wiki pages and pull requests. The following supported elements are included:

- Symbols
- Greek letters
- Mathematical operators
- Powers and indices
- Fractions and binomials
- Other KaTeX supported elements

To include mathematical notation, surround the mathematical notation with a `$` sign, for inline, and `$$` for block, as shown in the following examples:

Example: Greek characters

$$\alpha, \beta, \gamma, \delta, \epsilon, \zeta, \eta, \theta, \kappa, \lambda, \mu, \nu, o, \pi, \rho, \sigma, \tau, \upsilon, \phi, \ldots$$

$$\Gamma, \Delta, \Theta, \Lambda, \Xi, \Pi, \Sigma, \Upsilon, \Phi, \Psi, \Omega$$

Greek letters

Example: Algebraic notation

Area of a circle is

$$\pi r^2$$

And, the area of a triangle is:

$$A_{triangle} = \frac{1}{2}(b \cdot h)$$

Algebraic notation

Example: Sums and Integrals

$$\sum_{i=1}^{10} t_i$$

$$\int_0^\infty \mathrm{e}^{-x}, \mathrm{d}x$$

Build Task

Using Azure DevOps WikiPDFExport as build task is straightforward.

1. Create a new build definition
2. Git Source is "Other Git"
3. 1. Add the clone url to the wiki to the details and username / password if required
4. Ensure that the agent is a windows agent
5. Add a powershell task with the following code:

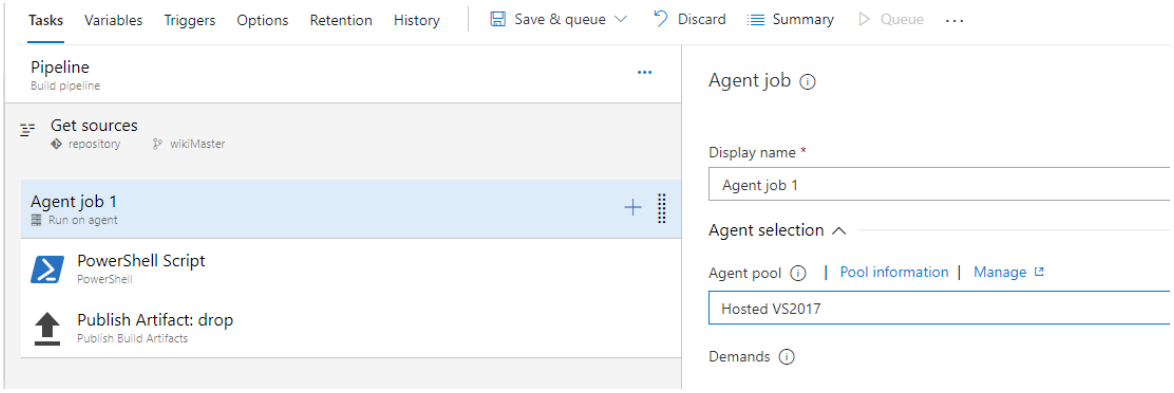
```
#Download url to the export tool
$url = "https://dev.azure.com/mmelcher/8036eca1-fd9e-4c0f-8bef-646b32fbda0b/_apis/git/repositories/e08d1ada-7794-4b89-a3ea-
cb64a26683c3/Items?path=%2Fazuredevops-export-
wiki.exe&versionDescriptor%5BversionOptions%5D=0&versionDescriptor%5BversionType%5D=0&versionDescriptor%5Bversion%5D=master&download=true&r
version=5.0-preview.1"

#filename of the tool
$output = "azuredevops-export-wiki.exe"

#download the file
Invoke-WebRequest -Uri $url -OutFile $output

#launch the tool - adjust the parameters if required
./azuredevops-export-wiki.exe
```

5. Add a second task to publish the PDF as build artifact.

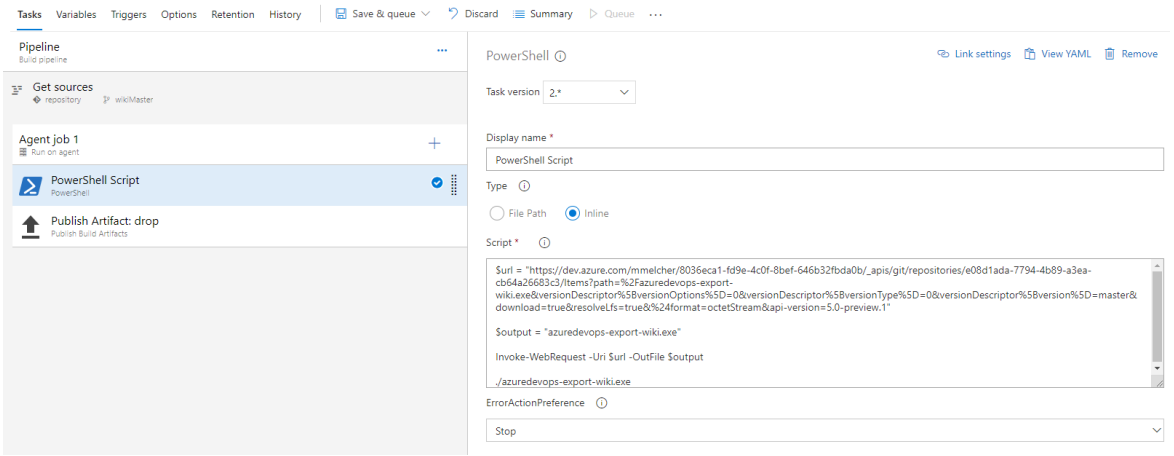
Once the build succeeds, you can download the PDF file from the build page or use it in a release.

Pictures

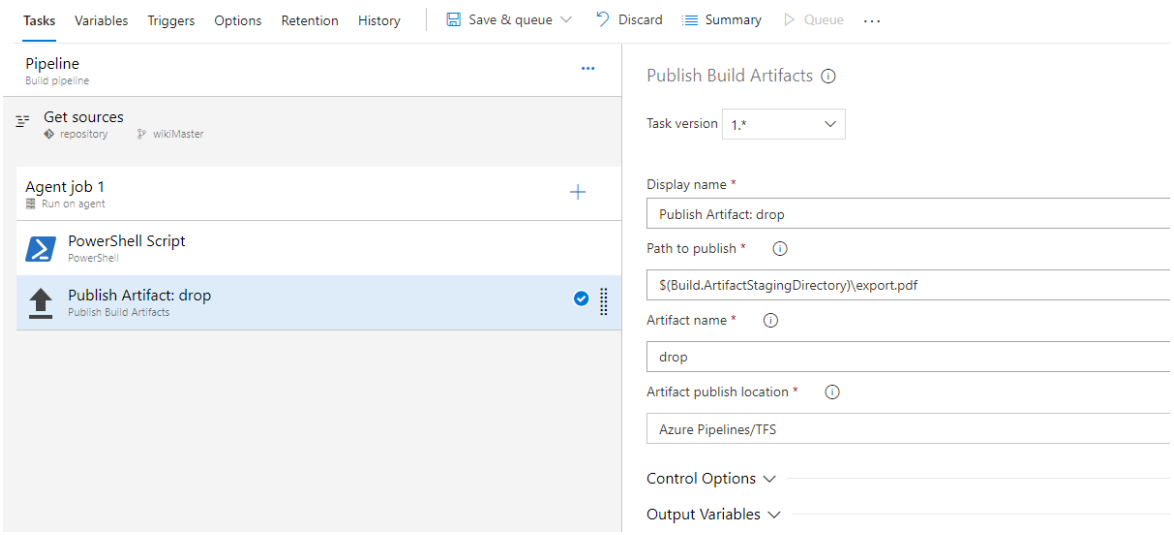
Windows Agent



PowerShell Task



Publish Task


Tasks
Variables
Triggers
Options
Retention
History
Save & queue
Discard
Summary
Queue
Pipeline
Build pipeline
Get sources
repository
wikiMaster
Agent job 1
Run on agent
PowerShell Script
PowerShell
Publish Artifact: drop
Publish Build Artifacts
Publish Build Artifacts
Task version
1.*
Display name *
Publish Artifact: drop
Path to publish *
$(Build.ArtifactStagingDirectory)\export.pdf
Artifact name *
drop
Artifact publish location *
Azure Pipelines/TFS
Control Options
Output Variables

License