# Documentation to

# Helse Vest Implementation Guideline

# Installation procedure

**Step 1. Clone the documentation source**

<https://bitbucket.org/FuroreNL/helsevestig.git>

**Step 1. Clone the publication tool source code**

<https://github.com/furore-fhir/Fhir.Publication.git>

**Step 2. Compile the publication tool**

Open the solution *Fhir.Publicatin.sln* in visual studio and build the *Fhir.Publisher* project.

**Step 3. Open the command line**

**Step 4.**

In the command line add to path, the path to the executable of the publication tool.

For example:

C:>path %path%;c:\git\fhir.publication\fhir.publisher\bin\debug

**Step 4. Locate the documentation**

In the command line, go to the documentation source directory

Then go to the subdirectory HelseVestSource

**Step 5. Run the generator**

From the command line run the publication tool and provide the make file:

fhirdoc make.mk

The tool will generate a “Generated” directory next to (on the same level) the folder where the make file is.

**How the make file commands work**

A make file consists of statements. Each line contains one statement.

A statement starts with a file selection statement followed by one or more processors separated by two right arrows “>>”.

|  |  |
| --- | --- |
| **Format** | **Example** |
| select <mask> >> processor >> … | select \*.md >> markdown >> save |

# Selection statement

A file selector statement starts with the word “select” followed by a one or more file masks. There are two options:

* Recursive (also include subdirectories)
* Output (select files from the output directory)

Mask may contain subdirectories.

|  |  |
| --- | --- |
| **Format** | **Example** |
| select <mask> | select \*.md |
| select <mask>,<mask>, … | select \*.jpg,\*.png,\*.gif |
| select <masks> <options> | select \*.mk -recursive |
|  | select \*.html -output |

# Make Processors

# Markdown

|  |  |
| --- | --- |
| **Format** | **Example** |
| markdown | select \*.md >> markdown >> save |

# Template

A template file is any file that contains the text “%body%”. A template processing command can refer to any file in the source directory.

|  |  |
| --- | --- |
| **Format** | **Example** |
| template <filename> | template layout.html |
| template $<stashkey> <filename> | template $tmp layout.html |

# Razor

A razor processing command processes razor files, which is a mix between html and c#. A razor file usually has the .cshtml extension. All c# code is processed to html.

|  |  |
| --- | --- |
| **Format** | **Example** |
| razor | razor |

# Copy

The copy processing command processes all input files by directly copying them to the output directory assuming that no other processing has been done on it. To save processed files, use the save command.

|  |  |
| --- | --- |
| **Format** | **Example** |
| copy | copy |

# Stash

The stash processor puts all files in the stage in a named stage and sets it aside for use in a different pipeline. The name of this stage must start with “$”.

|  |  |
| --- | --- |
| **Format** | **Example** |
| stash $<stashkey> | stash $tmp |

# Save

The save processor saves all files, whether processed or not to the output directory. Any processor can change the default extension of a file. This extension is used for saving. With the save command you can also explicitly set the extension.

|  |  |
| --- | --- |
| **Format** | **Example** |
| save | save |
| save <extension> | save .html |

# Attach

The attach processing command attaches files from a ***stash*** to a file in the pipeline. The file mask of the attach command accepts index wildcards. These wildcards point to parts of the input filename where the parts are separated by dots.

|  |  |
| --- | --- |
| **Format** | **Example** |
| **attach** $<stashkey> <file-mask> | attach $tmp $1.temp.html |

*Example use of attach:*

In this example footers are collected in a $footers stash. Then the footers are attached to documents where the first part of the footer matches the second part of the name of the document. The result is concatenated. (document.chapter1.html + chapter1.footer.html)

|  |
| --- |
| select \*.footer.html >> stash $footers  select document.\*.html >> attach $footers $2.footer.html >> concatenate >> save |

# Concatenate

Concatenates all attached (see the attach processing command) documents to the end of the document in the pipeline..

|  |  |
| --- | --- |
| **Format** | **Example** |
| concatenate | (see the examples from the attach command) |

# Make

The make processing command executes all input as a make file. Effectively making it possible to run one make file from another.

|  |  |
| --- | --- |
| **Format** | **Example** |
| make | select chapter6\\*.mk >> make |