# Iansasciidoc Previewfrompackagesmarkdownprevie package

Ians adaption of markdown-preview to preview asciidoc files.

~~**Be sure to disable** atom-language-asciidoctor which is an atom package. It does some strange things, for example, if it is enabled may package will no longer open files with extensions: .txt, .adoc and possibly others occasionally like .ron.~~

Atom packages for AsciiDoc should not be enables including:

- `language-asciidoc` ,: Syntax highlighting and snippets for AsciiDoc & `autocomplete-asciidoc` .
- `asciidoc-preview` : Show a preview for the AsciiDoc has been fixed and should be OK but it is hoped that the current package will replace that and be more resillient to changes in pulsar and its dependencies.
- `asciidoc-image-helper` : When pasting an image into an Asciidoc document, this package will paste clipboard image data as a file into a folder specified by the user.
- `asciidoc-assistant` : install Atom AsciiDoc basic packages with one package.

Add this to `config.cson` under core. It ensures that adoc & asciidoc files are treated as text not as YAML type files:

```
core:
  customFileTypes:
    "text.plain": [
      "adoc"
      "asciidoc"
    ]
```

Make sure `source.asciidoc` is included in package.json like this below. The other entries may vary from this.

```
"grammars": {
  "order": 0,
  "type": "array",
  "default": [
    "source.gfm",
    "source.litcoffee",
    "text.html.basic",
    "text.md",
    "text.plain",
    "text.plain.null-grammar",
    "source.asciidoc"
  ],
  ```
```

  Any .adoc files to be previewed should show file type as ```source.asciidoc```. If that is not included under grammars above then adoc syntax highlighting will be absent.

## What does work
* ctrl-alt-shft-c will preview files as if adoc.  ~~At the moment these have to be changed to .XXX cos .adoc & .ad files are blocked from somewhere in the original markdown-preview.~~
* ctrl-alt-shft-g will render the file in external falkon browser.
* ctrl-alt-shft-s will save files as pdf and preview and render this file in pulsar. Using node asciidoctor-web-pdf.js. Note it is quite slow. (Also tried to run asciidoctor-pdf.rb but this fails wi no output.)
* ctrl-shft-s if cursor is in the preview pane will save as html and show this source file in pulsar. However, if there are certain complex links and the like in the adoc file the save to html will fail silently. Which links cause this is not clear at the moment but most adoc files will save properly as html. This file can be opened in pulsar html-preview (ctrl-shift-H) which gives a preview functionally different from adoc-preview.
* ctrl-shft-s if cursor is in the adoc source pane it will save file as .adoc. You can change the output name if you choose.

(The strange key letters are those used in an earlier package incremented. G = Falkon, C = Asciidoc, S for save but wi alt added.)

## What does not work
Infront matter :backend:  is unlikely to work cost backends are written in ruby & the js versions are needed in pulsar.


# Notes on config.cson

Usually in ~/.pulsar/.config.cson

This is how mine looks:

```cson
"*":
  "asciidoc-preview":
    baseDir: "-"
    exportAsPdf:
      enabled: true
    frontMatter: true
    renderOnSaveOnly: true
  core:
    closeDeletedFileTabs: true
    customFileTypes:
      "source.asciidoc": [
        "adoc"
        "asciidoc"
```
```

file-types [] # Should be removed as it is outdated & customFileTypes used instead.

<hr> <hr><hr>

# README.md for original markdown-preview

# IGNORE below for reference only

Show the rendered HTML markdown to the right of the current editor using <kbd>ctrl-shift-m</kbd>.

It is currently enabled for `.markdown`, `.md`, `.mdown`, `.mkd`, `.mkdown`, `.ron`, and `.txt` files.

![iansasciidoc-previewfrompackagesmarkdownpreview20240610nomd](https://cloud.githubusercontent.com/assets/378023/10013086/24cad23e-6149-11e5-90e6-663009210218.png)

## Customize

By default Iansasciidoc Previewfrompackagesmarkdownpreview20240610nomd uses the colors of the active syntax theme. Enable **Use GitHub.com Style** in the *package settings* to make it look closer to how markdown files get rendered on github.com.

![iansasciidoc-previewfrompackagesmarkdownpreview20240610nomd GitHub style](https://cloud.githubusercontent.com/assets/378023/10013087/24ccc7ec-6149-11e5-97ea-53a842a715ea.png)

When **Use GitHub.com Style** is selected, you can further customize the theme of the Markdown preview with the **GitHub.com Style Mode** setting. Since the GitHub website has a light theme and a dark theme, `iansasciidoc-previewfrompackagesmarkdownpreview20240610nomd` allows you to choose which theme to use when previewing your files. By default, it will use whatever mode is preferred by your system, but you can opt into “Light” or “Dark” to force it to use a particular theme.

No matter which theme you use, you can apply further customizations in your `styles.less` file. For example:

```
.iansasciidoc-previewfrompackagesmarkdownpreview20240610nomd prec {
  background-color: #444;
}
```

## Language identifiers in fenced code blocks

A detailed Markdown specification helps to ensure that Markdown is displayed consistently across multiple parsers. Sadly, the same isn’t true of code block language identifiers — the strings you use to tell the renderer what sort of code is inside a code block.

The CommonMark specification [explicitly avoids standardizing these identifiers](https://spec.commonmark.org/0.31.2/#info-string):

> *The first word of the info string is typically used to specify the language of the code sample, and rendered in the class attribute of the code tag. However, this spec does not mandate any particular treatment of the info string.*

There are several valid ways to infer specific languages from language identifiers such as `js`, `less`, `coffee`, and `c`. This package supports the following systems, configured via the **Syntax Highlighting Language Identifiers** setting:

- [Linguist](https://github.com/github-linguist/linguist): Used by GitHub (previously the default and only language identification system).
- [Chroma](https://github.com/alecthomas/chroma): Used by CodeBerg/Gitea/Hugo/Goldmark.
- [Rouge](https://github.com/rouge-ruby/rouge): Used by GitLab/Jekyll.
- [HighlightJS](https://highlightjs.org/): Used in a number of places, but most relevantly on the [Pulsar Package Registry](https://web.pulsar-edit.dev/) website.

If none of these systems meets your needs, you may specify custom language identifiers. This may not be as portable as the systems described above, but it will at least produce the desired outcome on your own system.

The setting **Custom Syntax Highlighting Language Identifiers** lets you define a list of custom language identifiers that match up to languages available within your Pulsar installation.

For example, if you wanted to map `j` to JavaScript and `p` to Python, you'd add the following text to the **Custom Syntax Highlighting Language Identifiers** field:

```
j: source.js, p: source.python
```

Now `iansasciidoc-previewfrompackagesmarkdownpreview20240610nomd` will understand what to do with fenced code blocks that begin with <code>\`\`\`j</code> or <code>\`\`\`\`p</code>. These custom identifiers will work alongside whatever system you've chosen with **Syntax Highlighting Language Identifiers**, but will supersede that system in the event of conflict.