IP[y]: IPython
Interactive Computing

Warning

This documentation is for an old version of IPython. You can find docs for newer versions here.

Converting notebooks to other formats

Newly added in the 1.0 release of IPython is the `nbconvert` tool, which allows you to convert an `.ipynb` notebook document file into various static formats.

Currently, `nbconvert` is provided as a command line tool, run as a script using IPython. A direct export capability from within the IPython Notebook web app is planned.

The command-line syntax to run the `nbconvert` script is:

```
$ ipython nbconvert --to FORMAT notebook.ipynb
```

This will convert the IPython document file `notebook.ipynb` into the output format given by the `FORMAT` string.

The default output format is html, for which the `--to` argument may be omitted:

```
$ ipython nbconvert notebook.ipynb
```

IPython provides a few templates for some output formats, and these can be specified via an additional `--template` argument.

The currently supported export formats are:

- `--to html`
  - `--template full` (default)

    A full static HTML render of the notebook. This looks very similar to the interactive view.
  - `--template basic`

    Simplified HTML, useful for embedding in webpages, blogs, etc. This excludes HTML headers.
- `--to latex`

  Latex export. This generates `NOTEBOOK_NAME.tex` file, ready for export.
  - `--template article` (default)

    Latex article, derived from Sphinx's howto template.
  - `--template report`

    Latex report, providing a table of contents and chapters.
  - `--template basic`

Table Of Contents

Previous topic

The Jupyter Notebook Format

Next topic

Running a notebook server

This Page

Show Source

Quick search

Go

Enter search terms or a module, class or function name.

Very basic latex output - mainly meant as a starting point for custom templates.

- `--to pdf`

  Generates a PDF via latex. Supports the same templates as `--to latex`.

- `--to slides`

  This generates a Reveal.js HTML slideshow. It must be served by an HTTP server. The easiest way to do this is adding `--post serve` on the command-line. The `serve` post-processor proxies Reveal.js requests to a CDN if no local Reveal.js library is present. To make slides that don't require an internet connection, just place the Reveal.js library in the same directory where your_talk.slides.html is located, or point to another directory using the `--reveal-prefix` alias.

- `--to markdown`

  Simple markdown output. Markdown cells are unaffected, and code cells indented 4 spaces.

- `--to rst`

  Basic reStructuredText output. Useful as a starting point for embedding notebooks in Sphinx docs.

- `--to script`

  Convert a notebook to an executable script. This is the simplest way to get a Python (or other language, depending on the kernel) script out of a notebook. If there were any magics in an IPython notebook, this may only be executable from an IPython session.

- `--to notebook`

  New in version 3.0.

  This doesn't convert a notebook to a different format per se, instead it allows the running of nbconvert preprocessors on a notebook, and/or conversion to other notebook formats. For example:

  ```
  ipython nbconvert --to notebook --execute
  mynotebook.ipynb
  ```

  will open the notebook, execute it, capture new output, and save the result in `mynotebook.nbconvert.ipynb`.

  ```
  ipython nbconvert --to notebook --nbformat 3
  mynotebook
  ```

  will create a copy of `mynotebook.ipynb` in `mynotebook.v3.ipynb` in version 3 of the notebook format.

  If you want to convert a notebook in-place, you can specify the ouptut file to be the same as the input file:

  ```
  ipython nbconvert --to notebook mynb --output mynb
  ```

  Be careful with that, since it will replace the input file.

Note

nbconvert uses pandoc to convert between various markup languages, so pandoc is a dependency when converting to latex or reStructuredText.

The output file created by `nbconvert` will have the same base name as the notebook and will be placed in the current working directory. Any supporting files (graphics, etc) will be placed in a new directory with the same base name as the notebook, suffixed with `_files`:

```
$ ipython nbconvert notebook.ipynb
$ ls
notebook.ipynb   notebook.html    notebook_files/
```

For simple single-file output, such as html, markdown, etc., the output may be sent to standard output with:

```
$ ipython nbconvert --to markdown notebook.ipynb --stdout
```

Multiple notebooks can be specified from the command line:

```
$ ipython nbconvert notebook*.ipynb
$ ipython nbconvert notebook1.ipynb notebook2.ipynb
```

or via a list in a configuration file, say `mycfg.py`, containing the text:

```
c = get_config()
c.NbConvertApp.notebooks = ["notebook1.ipynb",
"notebook2.ipynb"]
```

and using the command:

```
$ ipython nbconvert --config mycfg.py
```

LaTeX citations

`nbconvert` now has support for LaTeX citations. With this capability you can:

- Manage citations using BibTeX.
- Cite those citations in Markdown cells using HTML data attributes.
- Have `nbconvert` generate proper LaTeX citations and run BibTeX.

For an example of how this works, please see the citations example in the nbconvert-examples repository.