# Including Interactive Elements

## Introduction

:::{important}   
Our platform for digital publishing displays books as [static web pages](https://en.wikipedia.org/wiki/Static_web_page). Because of that, **features that require a server running in the background are not supported**. Please make sure that an object you want to add to a digital version of your book can be handled by a static web site.  
:::

All dynamic functionalities must be performed in a browser of anyone who views your book. This is a very powerful solution but has its limitations. Some dynamic features require a server or a database, which are not provided in our setup. If a feature cannot be performed in a viewer’s browser, it will not be rendered in your digital book and can even break part of your book of being shown.

Nearly all dynamic functionalities that can be shown in a static web page can be included in a digital version of your book as well. This includes interactive maps, graphs, charts, diagrams, animations, video or audio players, and even games your readers can play – as long as it doesn’t require a running server.

**Figure 1:** Tic tac toe game working without a server. License: MIT.

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/tic-tac-toe-game.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/tic-tac-toe-game.png |
| ## Source | Paweł Kamiński |
| ## License | MIT |
| ## Authors | Paweł Kamiński |
| ## Height | 400 |

### “Will I Be Able To Use My Pre-Existing Jupyter Notebook?”

Yes, you will be able to use your pre-existing Jupyter notebook. We can show your Jupyter notebook (data, code and outputs), but we will not execute your code for you. You must execute your notebook and export its outputs in a way described in later parts of these guidelines. This solution gives you great flexibility in generating your book’s contents. This means that you should be able to use the work you have done on your `.ipynb` file, regardless which Python version you have been using (or maybe you used something completely different instead), what are the libraries that you have chosen, what hardware you operated on (for example you can perform your computations utilizing GPU instead of CPU), etc. It should work as expected, assuming the generated output can be included in a static web page.

### Thinking About Longevity

When preparing your book for publishing, please consider its life after the release. Whenever you include a link to an external source, bear in mind that a target of such link might change or cease to exist in the future.

:::{important}  
When using hyperlinks to web pages, please archive the exact version of the page you want to link to, archive it using the [Internet Archive project](https://web.archive.org/) and put the archived link as an address of the hyperlink (e.g. to https://web.archive.org/web/1993043000000/link/to/your/actual/target).  
:::

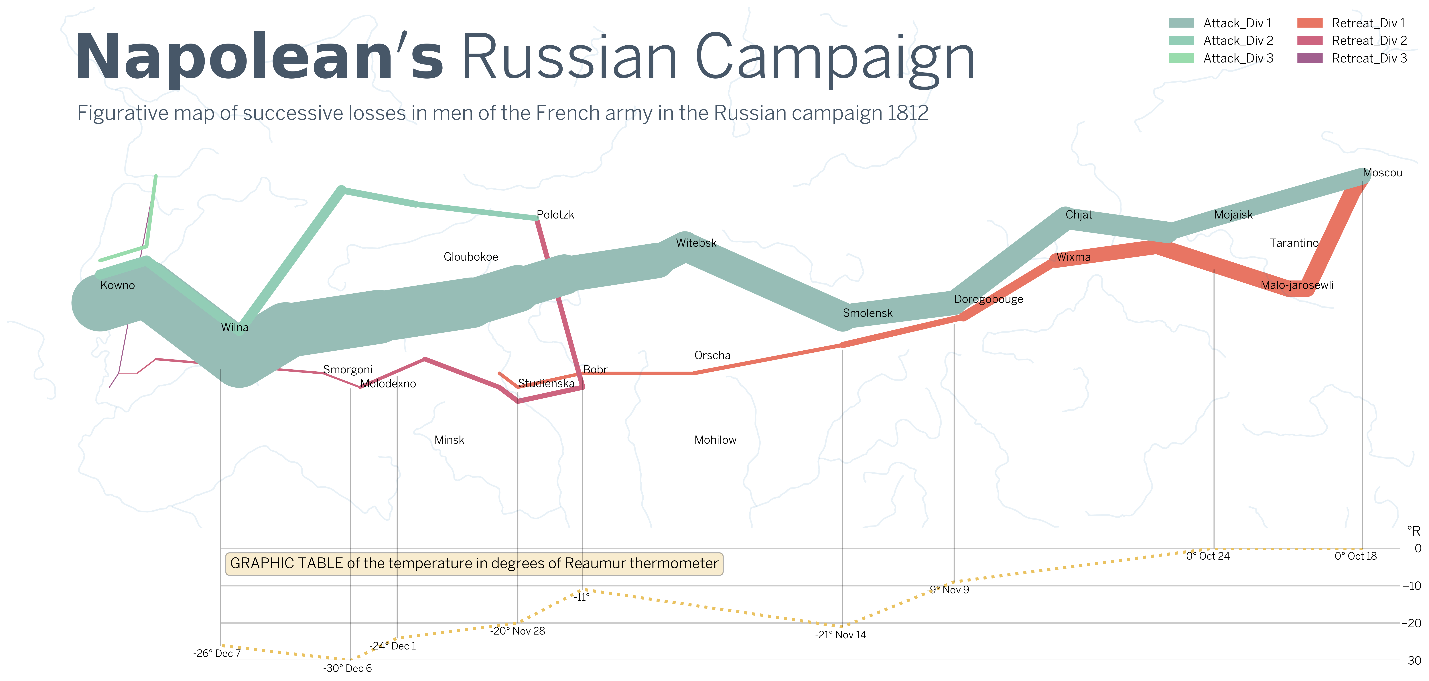
You should also apply this caution when dealing with any other external sources. Whenever you directly or indirectly use a resource that is hosted by a service like YouTube, Vimeo, Sketchfab, Observable, GitHub, Google Drive, Google Sheets, Tableau, etc., you have to take note of its terms of service. In general, there is no guarantee that a source hosted there will stay unchanged. Wherever possible, include files directly in your `assets` folder instead of links to resources.

## Non-Dynamic Objects

:::{tip}   
The simplest and best way to include a non-interactive visualization in your book is to insert it as an image.  
:::

Say you already have a Jupyter notebook, but instead of incorporating it as a whole in your book you want to include a single visualization generated in it. As an example, let take a  [the famous Charles Minard's chart recreated by Amit Amola using Python and Matplotlib](https://github.com/amitamola/minard_chart). The process of exporting it as an image is very simple, but it depends on which notebook application you are using.

**Figure 2:** Recreation of Minard's Chart of Napoleon's Russian Campaign made by Amit Amola (GPL-3.0 license).

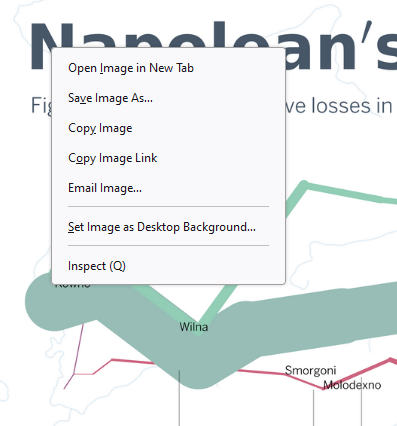
**

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/minards\_chart\_1812.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/minards\_chart\_1812.png |
| ## Author | Amit Amola |
| ## Source | <https://github.com/amitamola/minard_chart> |
| ## License | GPL-3.0 |

### Jupyter Notebook / Google Colab

In Jupyter Notebook or in Google Colab, just right click on a visualization, save the image and import it in your `.docx` file as you normally would with an image.

**Figure 3:** Saving object from Jupyter Notebook as an image. Example map made by Amit Amola (GPL-3.0 license).

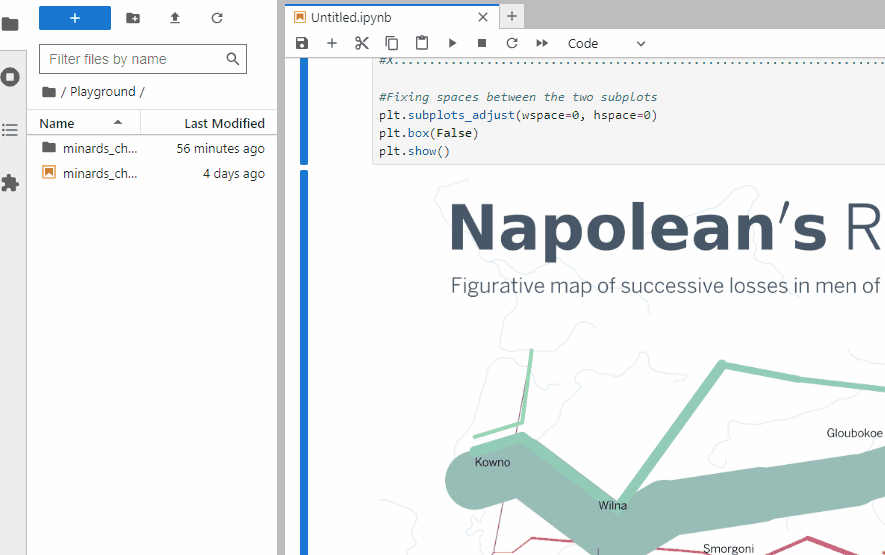


|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/save\_as\_image\_jupyter\_notebook.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/save\_as\_image\_jupyter\_notebook.png |
| ## Author | Amit Amola |
| ## Source | <https://github.com/amitamola/minard_chart> |
| ## License | GPL-3.0 |

### Jupyter Lab

In Jupyter Lab, you can just drag-and-drop the output of your notebook to the File Browser tab on the left of the screen. Download the object and insert it in your `.docx` file as you normally would with an image.

**Figure 4:** Saving object from Jupyter Lab as an image. Example map made by Amit Amola (GPL-3.0 license).

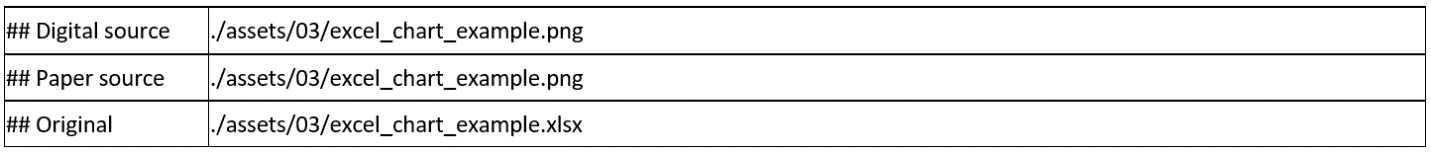


|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/save\_as\_image\_jupyter\_lab.gif |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/save\_as\_image\_jupyter\_lab.png |
| ## Author | Amit Amola |
| ## Source | <https://github.com/amitamola/minard_chart> |
| ## License | GPL-3.0 |

## Microsoft Office objects

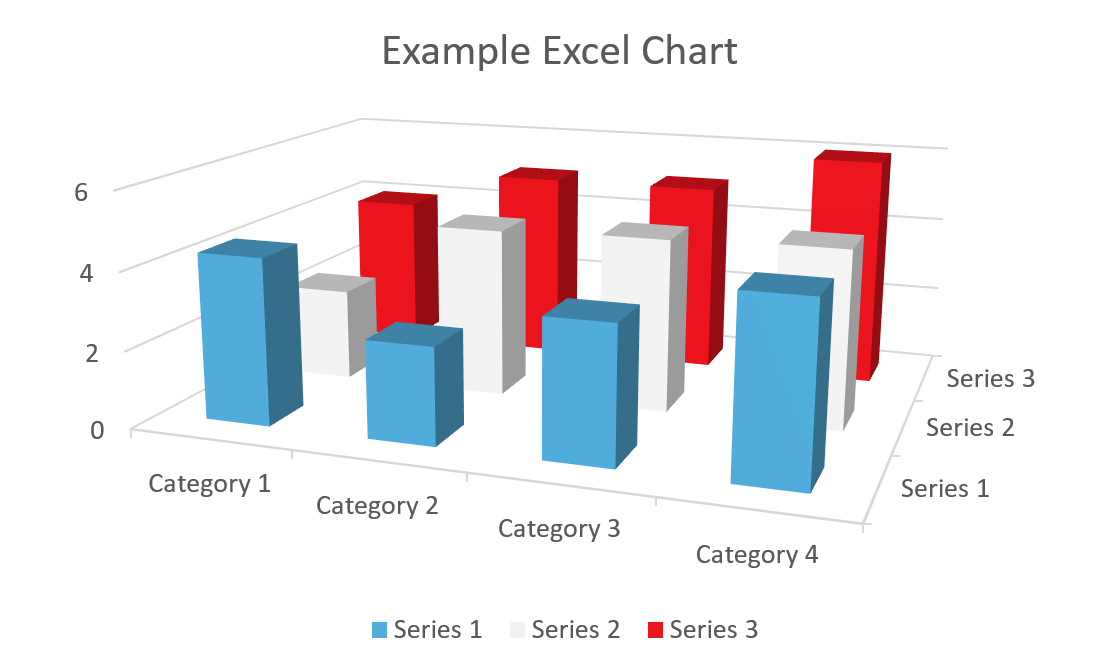
You can also enrich your book with objects from your MS Word, MS Excel, MS PowerPoint, etc. This includes, among others, Charts, 3D models, WordArts and SmartArts objects. However, just inserting them to your `.docx` file will not work. Instead, you must save them to a separate file and reference them in your metadata table, at least in the “*## Original*” field. It is recommended that you take a screenshot of your object and use the image as the “*## Digital input*” and the “*## Paper input*” in the metadata table as well. Bear in mind that such screenshots are required to have high quality to be published.

**Figure 5:** Example of a metadata table referencing an Excel chart.



|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/excel\_chart\_metadata\_tbl.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/excel\_chart\_metadata\_tbl.png |
| ##Source | Snapshot of PC screen of Paweł Kamiński |
| ##Author | Paweł Kamiński |
| ##Alt-text | Example of a Metadata Table required to include MS Excel chart in a publication |

**Figure 6:** Excel chart rendered from the metadata table.



|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/excel\_chart\_example.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/excel\_chart\_example.png |
| ## Source | ./assets/04\_interactivity\_in\_docx/excel\_chart\_example.xlsx |
| ##Alt-text | “3-D Colum” chart in MS Word based on quasi-random data. |
| ## Author | Paweł Kamiński |

:::{tip}  
When dealing with Excel charts and similar objects, you should save them to a separate file and insert a metadata table with the “*##Source*” field pointing to the separate file (e.g. to the `.xlsx` file) and the “*##Digital input*” and the “*##Paper input*” fields pointing to an high-resolution image of the chart.  
:::

## Sketchfab 3D Models

You can easily include a 3D model hosted on Sketchfab in your book. To do that go to the web page on [sketchfab.com](https://sketchfab.com/), select the "Embed" option and copy to clipboard the code (including the embed wrapper div). Save this div in a html file in your book's assets directory. Then use metadata table and the "## Digital input" field to point to the new html file. Please be aware that using this method your book only references a 3D model. If it is removed from Sketchfab or moved to a different place, your book will point to an empty location.

**Figure 7:** Kinora 3D model by Tim van der Heijden (license CC-BY-4.0).



|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/kinora3d.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/kinora.png |
| ## Source | https://sketchfab.com/3d-models/kinora-02361e835a944450b9933c131cec73b3 |
| ## Author | Tim van der Heijden |
| ## Height | 160 |
| ## Delete previous | Yes |
| ## License | CC-BY-4.0 |
| ## Alt text | A model of the Kinora viewer, an early 1900s motion picture technology |

## Dynamic JavaScript Objects

JavaScript is a language that was specifically designed to run natively in web browsers. This means that all modern web browsers can interpret and run JavaScript code without needing any additional plugins or tools. On the other hand, tools like Jupyter Notebooks are fantastic tools for data analysis and exploration, but they’re not designed for creating elements of interactive web pages.

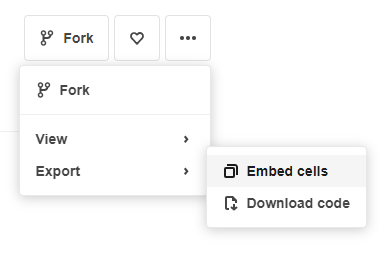
### Using Observable Notebook

The easiest way to include a JavaScript object, like a map or a graph, is to use the export option of an [Observable notebook](https://observablehq.com/).

For example, if you would like to include [this chart](https://observablehq.com/@mbostock/the-wealth-health-of-nations) in your book, here are the steps to follow.

#### Step 1

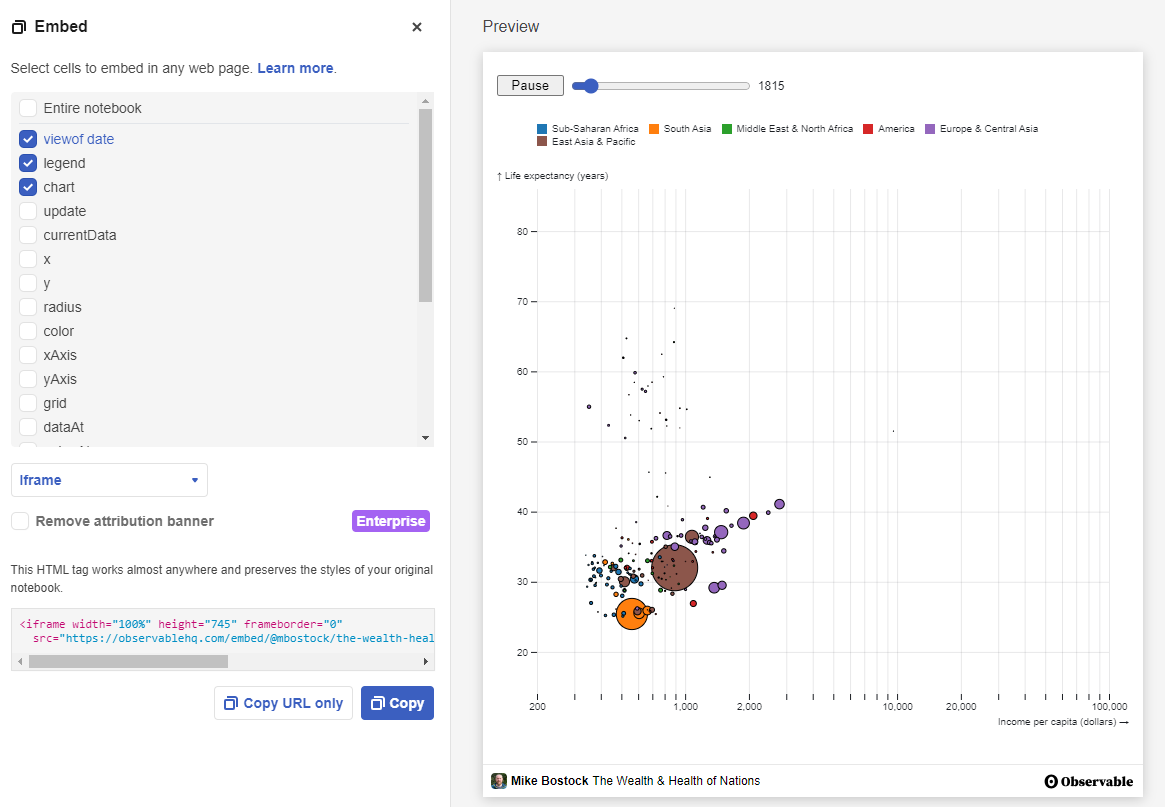
**Figure 8:** Click on the “Embed cells” button.



|  |  |
| --- | --- |
| ##Digital-input | ./assets/04\_interactivity\_in\_docx/observablehq-step1.png |
| ##Paper-input | ./assets/04\_interactivity\_in\_docx/observablehq-step1.png |
| ##Source | observablehq.com |
| ##Author | Screenshot taken by Paweł Kamiński |
| ##Alt-text | Dropdown menu of observablehq.com showing “Embed cells” as an export options |

#### Step 2

**Figure 9:** Select cells to embed, click “Copy URL only”.



|  |  |
| --- | --- |
| ##Digital-input | ./assets/04\_interactivity\_in\_docx/observablehq-step2.png |
| ##Paper-input | ./assets/04\_interactivity\_in\_docx/observablehq-step2.png |
| ##Source | <https://observablehq.com/@mbostock/the-wealth-health-of-nations> |
| ##Author | Mike Bostock is the author of the chart, screenshot was taken by Paweł Kamiński |
| ##Alt-text | Dropdown menu of observablehq.com showing “Embed cells” as an export options |
| ## License | ISC License, Copyright 2012–2020 Mike Bostock |

#### Step 3

**Figure 10:** In your Word file, create a Metadata table. Make sure it has a a) “*##Digital input*” field with the URL you’ve copied from the Observable notebook and b) “*##Height*” field with “height” value from the `<iframe>` snippet shown at the bottom of the page in Observable. Remember to add appropriate metadata about the author and the source.

A screenshot of a computer

Description automatically generated

|  |  |
| --- | --- |
| ##Digital-input | ./assets/04\_interactivity\_in\_docx/observablehq-step3.png |
| ##Paper-input | ./assets/04\_interactivity\_in\_docx/observablehq-step3.png |
| ##Source | Paweł Kamiński |
| ##Author | Paweł Kamiński |
| ##Alt-text | Example of a Metadata Table required to insert Observable notebook |

#### Result

**Figure 11:** Income per capita, population and life expectancy 1800-2005 graph made by Mike Bostock and published under ISC License.

|  |  |
| --- | --- |
| ## Digital input | <https://observablehq.com/embed/@mbostock/the-wealth-health-of-nations?cells=viewof+date%2Clegend%2Cchart> |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/wealth-and-health-of-nations.png |
| ## Source | Mike Bostock notebook on observablehq.com |
| ## Height | 745 |
| ## Author | Mike Bostock |
| ## License | ISC License, Copyright 2012–2020 Mike Bostock |

This method should work on almost every visualization as long as the Observable portal supports exporting objects as `<iframe>`. Some other examples are available [here](https://observablehq.com/explore).

### Other JavaScript Objects

If you feel competent in your skills in programming JavaScript, you can try to write your own html file and embed it in your book. This might be an interesting solution to include a custom JavaScript widget or visualization in your book. To embed a custom html file in your book, write the html location as the “*##Digital input*” in a metadata table.

**Figure 12:** Locations around the World of University of Luxembourg and De Gruyter.



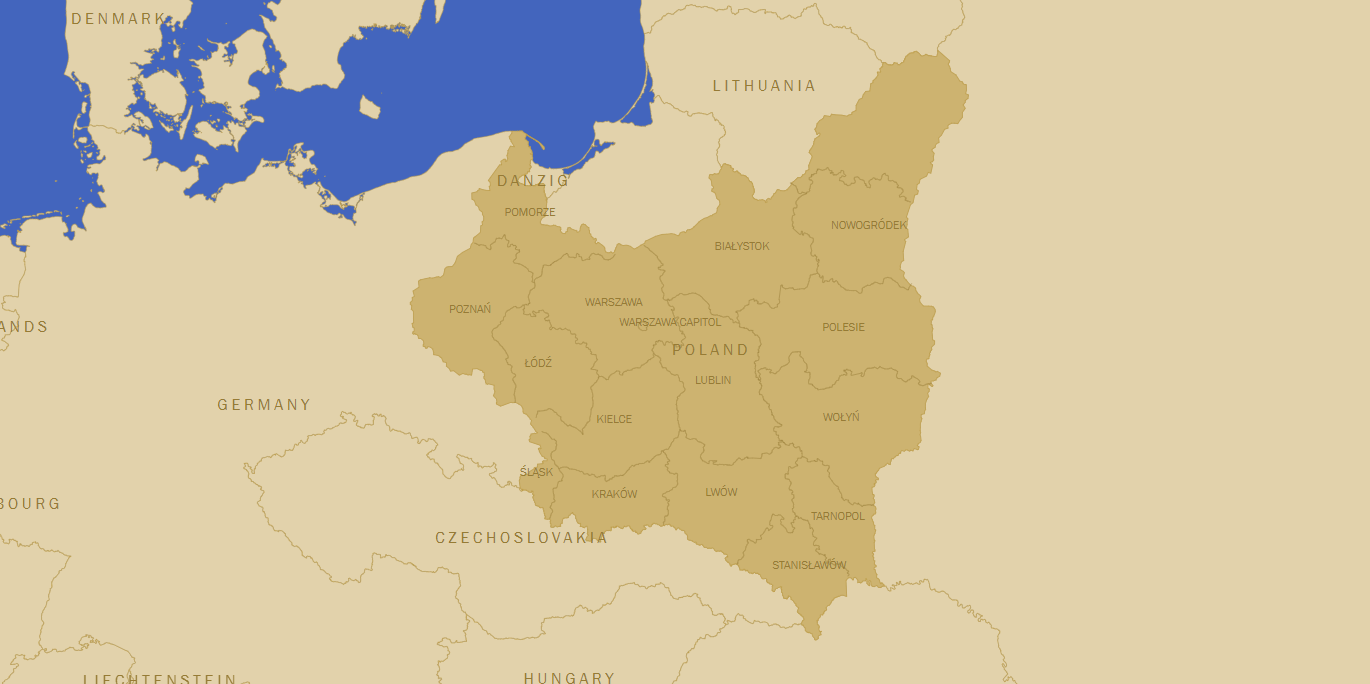
|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/leaflet\_custom\_embedding.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/dgt\_and\_unilux\_locations\_map.png |
| ## Source | Map was made by Paweł Kamiński using leaflet.js, which was written by Volodymyr Agafonkin, Iván Sánchez Ortega and other contributors (<https://github.com/Leaflet/Leaflet/graphs/contributors>) |
| ## Author | Map was made by Paweł Kamiński using leaflet.js, which was written by Volodymyr Agafonkin, Iván Sánchez Ortega and other contributors (<https://github.com/Leaflet/Leaflet/graphs/contributors>) |
| ## Height | 320 |
| ## Delete previous | Yes |

:::{important}  
Please be mindful, that **you, as an author, are responsible for the html you include in your book**. For example, if you include an unverified script from a doubtful source, you may involuntarily expose your readers to security threats. Other common problems include adding a feature that might not be rendered properly on all devices (e.g. on older browsers or on mobile phones), adding features that currently work but might break in the future or adding elements that negatively impact performance of your digital monography.  
:::

Writing code for maps, graphs, diagrams, etc. is a skill on its own. Please refer to separate documentation appropriate for a given topic to have a better understanding of how to construct such elements.

Another challenge is to get and transform data to use in your visualization. Here is an example of fetching a [TopoJSON](https://web.archive.org/web/20240123181231/https:/en.wikipedia.org/wiki/GeoJSON#TopoJSON) data before constructing a map.

**Figure 13:** Map of Poland as of April 1938 made by Iwona Rajca, @EveTheAnalyst. See also [a tutorial on making similar maps](https://web.archive.org/web/20221204213018/https:/datawanderings.com/2019/07/08/merging-historical-maps/).

**

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/1938\_april\_poland.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/1938\_april\_poland.png |
| ## Source | <https://web.archive.org/web/20221204213018/https:/datawanderings.com/2019/07/08/merging-historical-maps/> |
| ##Author | Iwona Rajca (@EveTheAnalyst) |
| ## Height | 450 |
| ## Delete previous | Yes |

Our platform for digital publishing gives you great flexibility regarding the elements you can embed in your book. Nearly every object that can be embedded as a `<iframe>` html object can be also part of your book. Use this ability with highest responsibility.

## Dynamic Objects From Jupyter Notebook

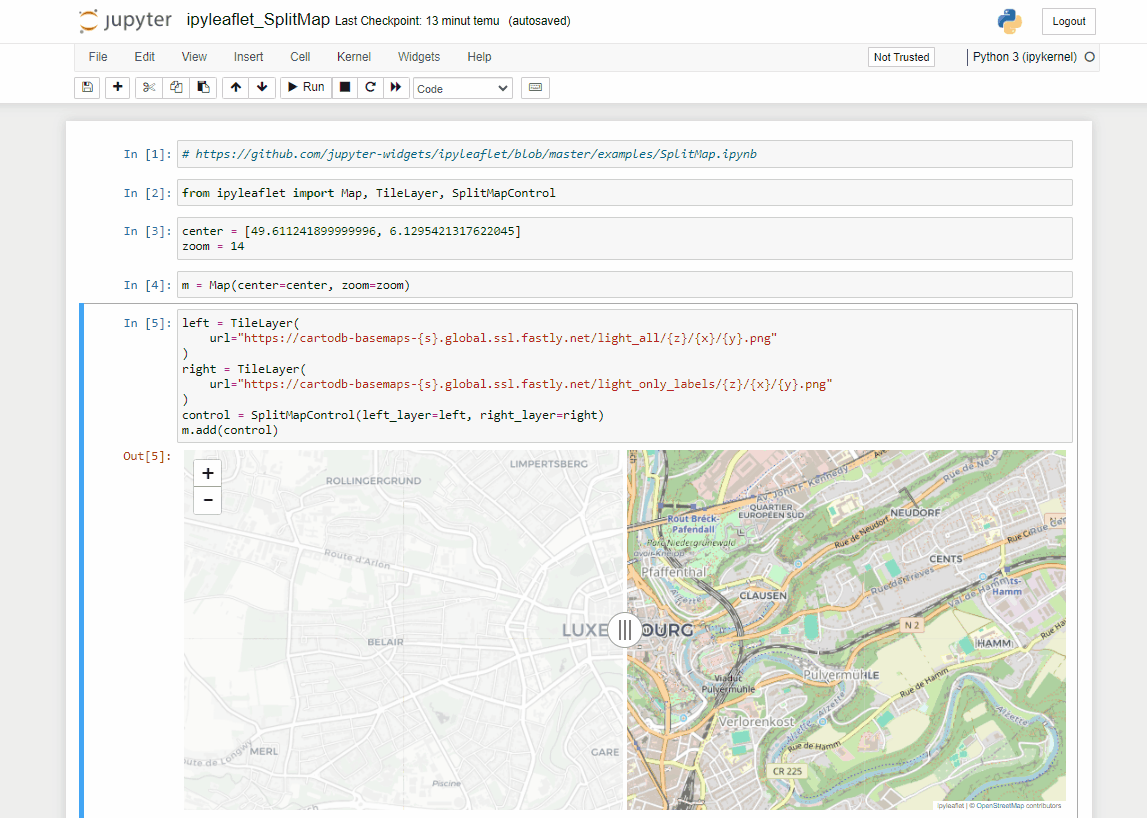
### Jupyter Widgets

There is a subset of Jupyter interactive visualizations called [jupyter-widgets](https://github.com/jupyter-widgets/ipywidgets) that do not need a server to add interactivity to your book called.

#### Step 1

Copy the html object (in Jupyter notebook select “Widgets” > “Embed Widgets” > “Copy to Clipboard”).

**Figure 14:** Example of accessing html describing a map in Jupyter Notebook.



|  |  |
| --- | --- |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map.png |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map.gif |
| ## Source | <https://github.com/jupyter-widgets/ipyleaflet/blob/master/examples/SplitMap.ipynb> |
| ##Author | Author of the example is Martin Renou (<https://github.com/martinRenou>) with a slight modification from Paweł Kamiński. Map layer data from CartoDB (<https://carto.com/>). Main software used in this example is leaflet.js, which was written by Volodymyr Agafonkin, Iván Sánchez Ortega and other contributors (see <https://github.com/Leaflet/Leaflet/graphs/contributors>). |

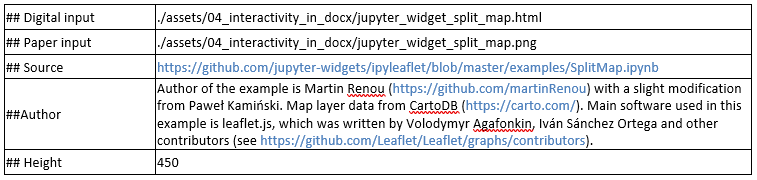
#### Step 2

Paste the copied data into `.html` file.

#### Step 3

Point to .html file in your `.docx` file.

**Figure 15:** Example of Metadata Table to insert html to your manuscript.



|  |  |
| --- | --- |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map\_metadata\_tbl.png |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map\_metadata\_tbl.png |
| ##Source | Snapshot of PC screen of Paweł Kamiński |
| ##Author | Paweł Kamiński |
| ##Alt-text | Example of a Metadata Table required to include html in the digital version |

#### Result

****

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/jupyter\_widget\_split\_map.png |
| ## Source | <https://github.com/jupyter-widgets/ipyleaflet/blob/master/examples/SplitMap.ipynb> |
| ##Author | Author of the example is Martin Renou (<https://github.com/martinRenou>) with a slight modification from Paweł Kamiński. Map layer data from CartoDB (<https://carto.com/>). Main software used in this example is leaflet.js, which was written by Volodymyr Agafonkin, Iván Sánchez Ortega and other contributors (see <https://github.com/Leaflet/Leaflet/graphs/contributors>). |
| ## Height | 450 |
| ## Delete previous | Yes |

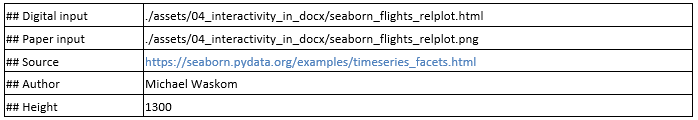
### Jupyter Notebooks As .html

You can also include the whole `.ipynb` notebook inside an `.docx` file. To do that, just export the contents of your `.ipynb` to `.html` and refer to this `.html` file in a metadata table. How to perform the export operation depends on the tool you are using.

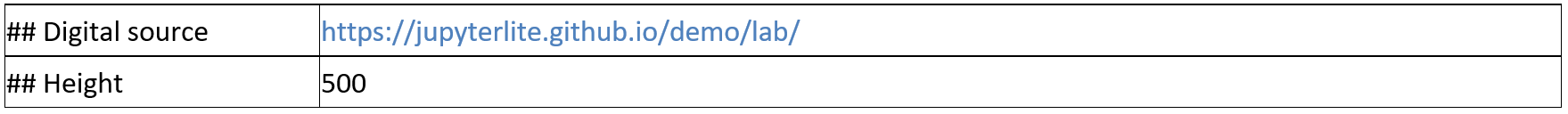
In Jupyter notebook, you must click the “File” button in the top left corner of the window and select “Download as” > “HTML (.html)”.

In Jupyter Lab, select “File” > “Save and Export Notebook As…” > “HTML”.

Then just reference the `.html` file in the metadata table in your `.docx` file.



|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/seaborn\_flights\_metadata\_tbl.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/seaborn\_flights\_metadata\_tbl.png |
| ## Source | Snapshot of PC screen of Paweł Kamiński |
| ## Author | Paweł Kamiński |
| ## Alt-text | Example of a Metadata Table required to include `.ipynb` files as `.html` |



**Figure 16:** Monthly distribution of passengers (1949-1960).

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/seaborn\_flights\_relplot.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/seaborn\_flights\_relplot.png |
| ## Source | <https://seaborn.pydata.org/examples/timeseries_facets.html> |
| ## Author | Michael Waskom |
| ## Height | 1300 |

If interactivity works in the stand-alone `.html` file, it most likely will also work in the digital version of your book.

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/plotly\_interactive\_chart.html |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/plotly\_interactive\_chart.png |
| ## Source | <https://plotly.com/python/line-and-scatter/>  <https://github.com/plotly/plotly.py/blob/doc-prod/doc/python/line-and-scatter.md> |
| ## Author | Plotly Technologies Inc. |
| ## Height | 730 |

### JupyterLite

To give your readers availability to interact with your code, one option is to use the JupyterLite. Because it runs entirely in the browser and does not need a server, you will be able to include it a digital version of your book.

:::{important}  
Although JupyterLite is currently being developed by core Jupyter developers, the JupyterLite project is still unofficial. Not all the usual features available in JupyterLab and the classic Jupyter Notebook will work with JupyterLite, but many already do. There is also no guarantee that JupyterLite will be developed and maintained in the future.  
:::

To include a JupyterLite terminal inside your digital page, insert a metadata table referring to the JupyterLite demo hosted on GitHub. You can also choose among available options, e.g. Jupyter Lab vs Jupyter Notebook, Python vs JavaScript, etc. In this example, we do not point to any “## Paper input”, so that our paper version will leave that object out entirely.

|  |  |
| --- | --- |
| ## Digital input | ./assets/04\_interactivity\_in\_docx/jupyterlite\_demo\_lab\_metadata\_tbl.png |
| ## Paper input | ./assets/04\_interactivity\_in\_docx/jupyterlite\_demo\_lab\_metadata\_tbl.png |

**Python example**

Here is an exemplary code you can try:

:::{code} python  
from sklearn import datasets  
import matplotlib.pyplot as plt  
  
iris = datasets.load\_iris()  
\_, ax = plt.subplots()  
scatter = ax.scatter(iris.data[:, 0], iris.data[:, 1], c=iris.target)  
ax.set(xlabel=iris.feature\_names[0], ylabel=iris.feature\_names[1])  
\_ = ax.legend(  
 scatter.legend\_elements()[0], iris.target\_names, loc="lower right", title="Classes"  
)   
:::

|  |  |
| --- | --- |
| ## Digital input | https://jupyterlite.github.io/demo/repl/index.html?kernel=python |
| ## Source | <https://jupyterlite.github.io/demo/repl/index.html?kernel=python> <https://github.com/jupyterlite/demo> |
| ## Author | Jupyterlite and Repl was made by Jeremy Tuloup and other contributors (<https://github.com/jupyterlite/demo/graphs/contributors>, https://github.com/jupyterlite/jupyterlite/graphs/contributors) |
| ## Height | 500 |

**JavaScript example**

|  |  |
| --- | --- |
| ## Digital input | https://jupyterlite.github.io/demo/repl/index.html?kernel=javascript |
| ## Source | <https://jupyterlite.github.io/demo/repl/index.html?kernel=python> <https://github.com/jupyterlite/demo> |
| ## Author | Jupyterlite and Repl was made by Jeremy Tuloup and other contributors (<https://github.com/jupyterlite/demo/graphs/contributors>, https://github.com/jupyterlite/jupyterlite/graphs/contributors) |
| ## Height | 500 |