Jupyter for Education

Moshe Zadka – https://cobordism.com

Acknowledgement of Country

Hayward (in the San Francisco Bay Area)

Ancestral homeland of the Ohlone people

I live in Hayward, in the San Francisco Bay Area Peninsula. I wish to acknowledge it as the ancestral homeland of the Ohlone people.

1 Introduction

2 Modern teaching

Teaching

A new world...

I am not, and have never been, a full time teacher. As such, this section is not about "teaching" in the abstract: it is there to establish context.

I do not have a time machine. Any information you take from the talk will only be useful to you in the future, when you develop lesson plans and other resources.

What does teaching look like in the mid 2020s?

2.1 A hybrid world

Hybrid world

In-person

Remotely

Self-paced

Traditionally, for millenia of human history, teaching has been an in-person affair. People would come together in close proximity to teach and learn.

Remote teaching is pretty old: as old as radio and telegraph. But for a year in 2020, remote teaching was, pretty much, all we had. Everyone was remote all the time.

Things have changed, but they did not "go back". We live, and will live in the forseeable future, in a "hybrid world". Even if you are teaching fully in person or fully remote, chances are you will collaborate with someone who does the opposite. More likely, some of your teaching will be in-person and some will be remote: maybe at the same time, and maybe not.

In both modes of teaching, we also have students doing some "self-paced" learning. Self-paced learning is a general term: it includes formal education's homework and tests, as well as or teacher-assigned in-class study.

When thinking of developing resources, it is important to consider all three modes. Sometimes, specific parts need to move, or be copied, from one to the other.

2.2 Text

Share text

Contents
Formatting

One important part of teaching is presenting text. Whether it is a teacher writing on the blackboard or an instructor projecting text on a screen, the written word is useful. Even if the advice is to *minimize* text displayed, it is rarely to eliminate it altogether.

2.3 Visuals

```
from PIL import ImageFont, ImageDraw, Image

image = Image.new("RGB", (500, 200), color=(255,255,255))
draw = ImageDraw.Draw(image)
font = ImageFont.truetype("static/DancingScript-Regular.ttf", 50)
small = ImageFont.truetype("static/DancingScript-Regular.ttf", 10)
medium = ImageFont.truetype("static/DancingScript-Regular.ttf", 20)
tiny = ImageFont.truetype("static/DancingScript-Regular.ttf", 5)
draw.text((10, 10), "Worth a...", font=font, fill=(0,0,0))
for i in range(10):
    draw.text((i*45, 55), "word", font=medium, fill=(0,0,0))
for i in range(300):
    x, y = divmod(i, 50)
    draw.text((y*25, 75+x*10), "word", font=small, fill=(0,0,0))
for i in range(700):
    x, y = divmod(i, 50)
    draw.text((y*15, 135+x*7), "word", font=tiny, fill=(0,0,0))
```

Visuals

A thousand words...

```
image
```

"Visual aids" is an important concept of teaching. A picture might not be worth a thousand words, but it is a good anchor for humans, who are visual creatures. Graphs, diagrams, photographs, and other things are all useful to show students and anchor their attention.

2.4 Interactivity

```
from ipywidgets import IntSlider

slider = IntSlider(min=2, max=10)

slider

IntSlider(value=4, max=10, min=2)
```

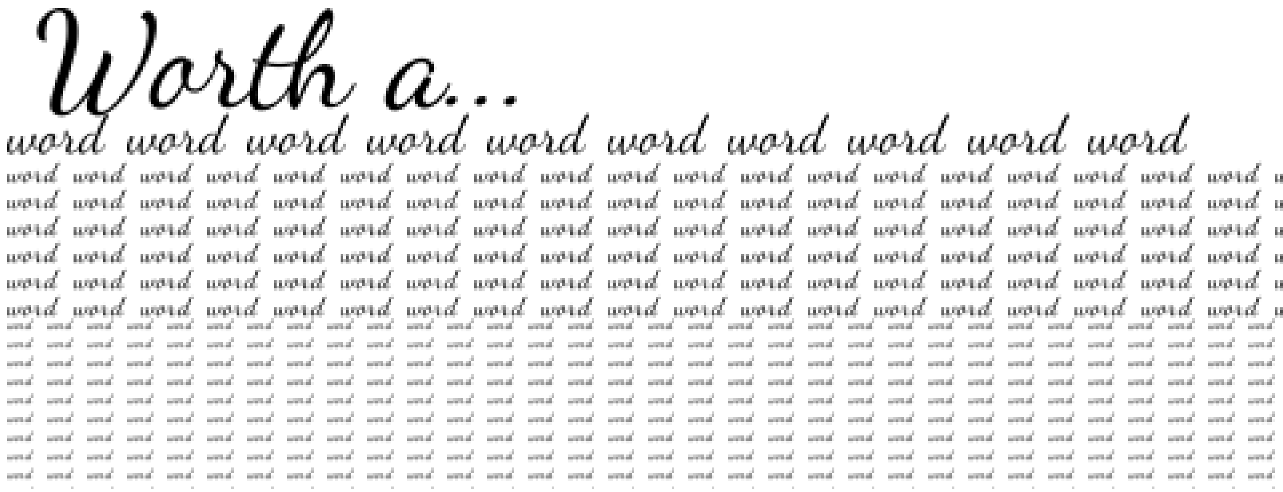

Figure 1: png

Figure 2: png

```
slider_img = Image.open("slider-1.png").convert("RGB")
x, y = slider_img.size
slider_img = slider_img.resize((x//4, y//4))

def draw_sample(value):
    image = Image.new("RGB", (200, 100), color=(255,255,255))
    draw = ImageDraw.Draw(image)
    font = ImageFont.truetype("static/DancingScript-Regular.ttf",
                              10 * slider.value)
    draw.text((10, 10), "Sample", font=font, fill=(0,0,0))
    return image
```

Interactivity...

Change the inputs...

```
slider_img
```

```
draw_sample(slider.value)
```

```
slider = IntSlider(min=2, max=10)
```

Figure 3: png

Figure 4: png

```
slider

IntSlider(value=2, max=10, min=2)

slider_img = Image.open("slider-2.png").convert("RGB")
x, y = slider_img.size
slider_img = slider_img.resize((x//4, y//4))
```

Interactivity...
...see the new outputs

```
slider_img

draw_sample(slider.value)
```

Finally, from the beginning of teaching, some level of interaction was considered crucial. Students need to engage with the material in some way: see

Figure 5: png

how different inputs lead to different results. This is considered a great way to teach, because it is even better at focusing attention on specific actions.

2.5 What is Jupyter?

Jupyter: Quick breakdown

Notebook
Kernel
Cells
Markdown

Before discussing how to use Jupyter, it's good to have a baseline understanding of what it is. Jupyter is a service that (mainly) helps you work with *notebooks*. Notebooks are connected to a *kernel*, which can execute code. Kernels represent language and execution environment.

The notebook is made of *cells*. Executable cells are executed by the kernel, and the value returned from the kernel is rendered to the cell's output in the notebook. Notebooks also support *markdown* cells that serve as documentation or explanation.

2.6 Jupyter: Existing usage

Jupyter: Classical usage

Exploration
Sharing

Jupyter was made, and exists, mainly, as something to support exploratory programming and, then, sharing the conclusion.

2.7 Jupyter Installation

Jupyter: Options

Local
Remote
Or...

Jupyter can be installed in several ways. It can be installed on the local computer. Depending on the local OS and user prefences, this can be DMG, flatpak, using brew, and other methods. The main problem with local installation is that computer OS configurations are diverse.

Around half of the students will follow the instructions and get it up and running. Around a quarter will be able to use the troubleshooting instructions to overcome the problem. Another tenth will be able to follow the "advanced" troubleshooting instructions...

Finally, there'll be a hundredth which still don't have access to Jupyter locally. Getting there requires tens of hours in fine-tuning instructions for installation and troubleshooting.

Remote installation is fine, since Jupyter is accessed through the browser.

2.8 JupyterLite

JupyterLite

In the browser...

Some limitations

The last option is JupyterLite. JupyterLite runs all of Jupyter, including the kernel, in the browser. This means installation is easy!

So, what is the problem? JupyterLite runs inside the browser, which is a weird execution environment. There are no TCP connections, only web sockets. There are no processes, only web workers. Native binary wheels, in general, cannot be installed.

If your use case is a good match for the limitations of JupyterLite, this is great! Make sure to try it out first before making assumptions.

3 Frontal teaching

3.1 Convert

Convert for presentations

HTML

PDF

Slides

When doing frontal teaching, the notebook is often not optimized for displaying directly to the class. Luckily, Jupyter notebooks can be converted to many formats. HTML, PDF, and slide decks are all supported natively by the creation tools, accessible directly from the UI.

3.2 Conversion fine-tuning

nbconvert

Hide cells

Hide cell output

Hide cell input

The UI conversion if backed by the nbconvert utility. Running this utlitity form the command-line gives even more options. As the simplest example, this allows hiding some cells completely (simplifiyng the pedagogic value), hiding the output of the cell (useful when functions called for their side-effects return values) and hiding cell input (so the result of the calculation can be shown without distracting by the actual details of the computation).

3.3 Pipeline

Conversion pipeline

To Markdown...

...then pandoc

For even more sophisticated conversion, use nbconvert to convert to markdown, and then pandoc for conversion to the final output format. This powerful technique allows even more customization. For example, this is basically how this presentation was generated, using the Python package ides.

3.4 Collaboration

Source-control friendly

JSON

Merges can be unpleasant

I don't want to oversell here. Compared to, say, Python source code, notebooks can have challenges with source control. Spurious diffs generated by having the notebook include the numeric index of each output operation can make for an annoying diff to resolve.

In this context, it is important to compare to the baseline of developing teaching resources in something like Keynote. As a JSON file with some attention to how it is rendered as line, the diffs will be at least partially usefl.

If this does become a bottelneck, some pre-commit-based tricks can be used to reduce that.

4 Independent work

4.1 Sharing

Sharing

Supported in Source Collaboration

Automatically rendered

Notebooks are rendered automatically in most popular source collaboration platforms like GitHub or GitLab. This means even before a student downloading a notebook, they can read it as a static presentation. For example, this helps people make sure they download the right notebook.'

4.2 Reading materials

Reading

Inline cells

Code rendering

Widgets acknowledging

For self-paced reading materials, there are a few options. Text can be written in Markdown cells using Markdown formatting for a student to read.

Cells can also generate HTML-displayable content that is rendered in the output. This can include fetching from an API or calculating the text from several resources.

You can also add widgets acknowledging the student has read a section. This can include a short checkbox or more sophisticated things like a multiple choice question ("I agree with the point being made... Yes/No/Maybe") or even a text prompt ("Explain the main themes of this paragraph"). This helps students focus their attention and, maybe, re-read a section if they realize they need to understand it further.

4.3 Setting up exercises

Exercise

Cells to be filled

Widgets

Exercises can use cells which need to be filled by the student or a widget. The option sometimes hinges on which affordance will feel the best for the target audience, and what their expectations are.

4.4 Student verification

Verification

Machine-checkable

Multiple choice

With Jupyter, you can also use widgets for *automatic verification* of the answers. This allows giving students immediate feedback on exercises where the answers is either machine-checkable or multiple choice. This allows students to immediately see their level of mastery of the material, so they can decide if they want to study resources again or ask for further help.

4.5 Teacher feedback

Teacher feedback

Submit notebook

Inline submission through API

Self-paced materials are often submitted for teacher feedback. While avoiding the subject of the usefulness of grading, grading or more detailed feedback are part of teaching for both intrinsic and extrinsic reasons. Students can submit filled notebooks through whatever submission standards already exist (e-mailing, student portal, using a source collaboration platform). Another option is to inline the submission using a widget which connects to existing APIs: either specific to the organization or platform, or more generic such as WebDAV.

5 Summary

5.1 Limitations

Limitations

Server set-up non-trivial

JupyterLite still limited

Jupyter is not a panacea. Then again, no tool is, nor should it be. Specifically, with Jupyter, you have to make a choice between the full-featured version which is hard to deploy or the limited version which is trivial to deploy. Both options are not optimal.

5.2 Advantages

Advantages

Powerful tooling

Evolving ecosystem

Despite of its limitations, Jupyter is a powerful tool to add to your teaching repertoire. It has a lot of existing tools, from UI extensions to backend hosting platforms, which can be used to enhance the learning experience. Even more importatly, the ecosystem is always evolving and getting better. Software-savvy teachers are in a place where they can contribute to the ecosystem tools that make Jupyter even better for teaching!

5.3 Iterating

Early days

Make mistakes

Learn

Share

Do better

These are the early days, still. If you use Jupyter for teaching, you will make mistakes. There are no widely acceptable best practices.

Learn from your mistakes. Share your lessons with the wider community, and work alone and with others to figure out how to do better.

5.4 Call to action

Jupyter: Teaching for 2020s

More visual

More interactive

Powerful flows

Jupyter is a tool that can help you make your lessons more visual, more interactive, and customized learning flows for your students. Why not try using it?