

**REPORT ON OpenDreamKit DELIVERABLE D4.5****SAGE notebook / JUPYTER notebook convergence**

FLORENT CAYRÉ, JEROEN DEMEYER, NICOLAS M. THIÉRY



Due on	01/09/2016 (M12)
Delivered on	28/02/2017
Lead	Université Paris-Sud (UPSud)
Progress on and finalization of this deliverable has been tracked publicly at: <a href="https://github.com/OpenDreamKit/OpenDreamKit/issues/94">https://github.com/OpenDreamKit/OpenDreamKit/issues/94</a>	

**DELIVERABLE DESCRIPTION, AS TAKEN FROM GITHUB ISSUE #94 ON 2017-02-28**

- **WP4:** User Interfaces
- **Lead Institution:** Université Paris-Sud
- **Due:** 2016-08-31 (month 12)
- **Nature:** Demonstrator
- **Tasks:** T4.1 (#69): Uniform notebook interface for all interactive components, T4.6 (#74) Structured documents
- **Proposal:** p. 48
- **Final report**

The Jupyter Notebook is a web application that enables the creation and sharing of executable documents which contain live code, equations, visualizations and explanatory text. Thanks to a modular design, Jupyter can be used with any computational system that provides a so-called *Jupyter kernel* implementing the *Jupyter messaging protocol* to communicate with the notebook. OpenDreamKit therefore promotes the Jupyter notebook as user interface of choice, in particular since it is particularly suitable for building modular web based Virtual Research Environments.

A notebook interface is such a vital integrative component that, predating Jupyter, the open source mathematical system SageMath had developed as early as 2005 its own solution, which we refer to here as the legacy SageMath notebook. Development was fast tracked to ensure its availability to allow the project to move forward, and it served as major source of inspiration for Jupyter. Meanwhile, moving at a fast pace thanks to its much larger community, Jupyter had, by 2014, basically caught up with the legacy SageMath notebook in terms of functionality.

Building on top of D4.4 (#93): Basic Jupyter interface for GAP, PARI/GP, SageMath, Singular, the goal of this deliverable was therefore to help phase out the legacy SageMath notebook in favour of Jupyter. Outsourcing this key but non disciplinary component is an important step toward the sustainability of ODK's ecosystem (Objective 5).

Since 2014, a lot of work was put into this by the SageMath community, and in particular by Volker Braun. Recently, this work has been continued thanks to ODK. It included two aspects: ensuring that Jupyter included all important features of the legacy SageMath notebook, and enabling a smooth migration path for users:

- ✓ #20690, #22458: Live documentation @fcayre, @nthiery, @videlec

- [ ] Interacts/widgets: full support and backward compatibility with the legacy SageNB #21267 (closed installation issues: #21260 #21261 #20218 #21256) @jdemeyer
- ✓ #19877: Conversion of legacy notebooks to Jupyter notebooks @vbraun, @videlec, @marcinofulus
- ✓ #19740: Migration wizard, as a web application

Future work:

- [ ] WYSIWYG editor for the markdown cells, such as TinyMCE (see this post for motivation)
- [ ] [#20316](https://trac.sagemath.org/ticket/20316): Add button to export SageNB notebooks to Jupyter

## CONTENTS

Deliverable description, as taken from Github issue #94 on 2017-02-28	1
1. Highlight of some of the main features implemented by ODK	2
Appendix A. Screenshots	3

### 1. HIGHLIGHT OF SOME OF THE MAIN FEATURES IMPLEMENTED BY ODK

An important use case of the notebook is *interactive functions*: these allow the user to control the input of a function using *widgets* such as sliders and text boxes. See Figure 1 for an example. This way, the user can easily investigate how some function changes when the input changes. Once the interact is created, it can be used by people having no experience at all with SAGE or PYTHON. Interacts have been implemented independently in the SAGE notebook and in JUPYTER (package `ipywidgets`). For the conversion of SAGE notebooks to JUPYTER to be useful, also interacts should work the same way. This has been implemented in the JUPYTER kernel for SAGE: see Figure 2 for the same example as Figure 1, this time in JUPYTER.

A handy feature of the legacy SAGE notebook is *live documentation*. In the SAGE notebook, the documentation of SAGE is “live”: the examples from the documentation become editable notebook cells. This way, the user can run those examples and experiment with them. This has now been implemented also in the JUPYTER notebook using the Thebe package from O’Reilly Media. Thebe allows embedding JUPYTER notebook cells in arbitrary webpages and this has been used to turn the SAGE documentation live when viewed through the JUPYTER notebook. See Figure 3 for an example.

The default notebook application in SAGE is now `sagenb-export`. This small web application, built on top of Jupyter, acts as migration wizard which can convert legacy SAGE notebooks to JUPYTER. It also has buttons to run either the legacy SAGE notebook server or the new JUPYTER notebook server. See Figure 4.

## APPENDIX A. SCREENSHOTS

**SAGE The Sage Notebook**  
Version 7.6.beta4

admin [Toggle](#) | [Home](#) | [Published](#) | [Log](#) | [Settings](#) | [Help](#) | [Report a Problem](#) | [Sign out](#)

**Interact example** Save Save & quit Discard & quit


last edited Feb 27, 2017, 12:11:09 PM by admin

File... Action... Data... sage ☐ Typeset ☐ Load 3-D Live ☐ Use java for 3-D Print Worksheet Edit Text Revisions Share Publish

This is a simple example of a worksheet with an interactive function plotting a mathematical function in a given range:

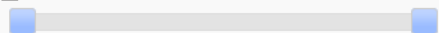
```
@interact
def _(f=input_box(x^2,width=20),
    color=color_selector(widget='colorpicker', label=''),
    axes=True,
    fill=True,
    zoom=range_slider(-3,3,default=(-3,3))):
    show(plot(f,(x,zoom[0], zoom[1]), color=color, axes=axes,fill=fill, figsize=3))
```

f

#0000ff 

axes ☒

fill ☒

zoom   
(-3.0, 3.0)

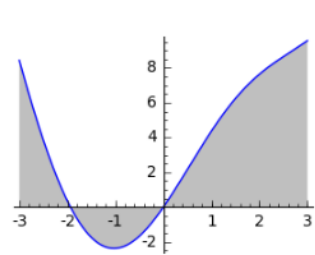


FIGURE 1. An example of an interact in the legacy SAGE notebook.

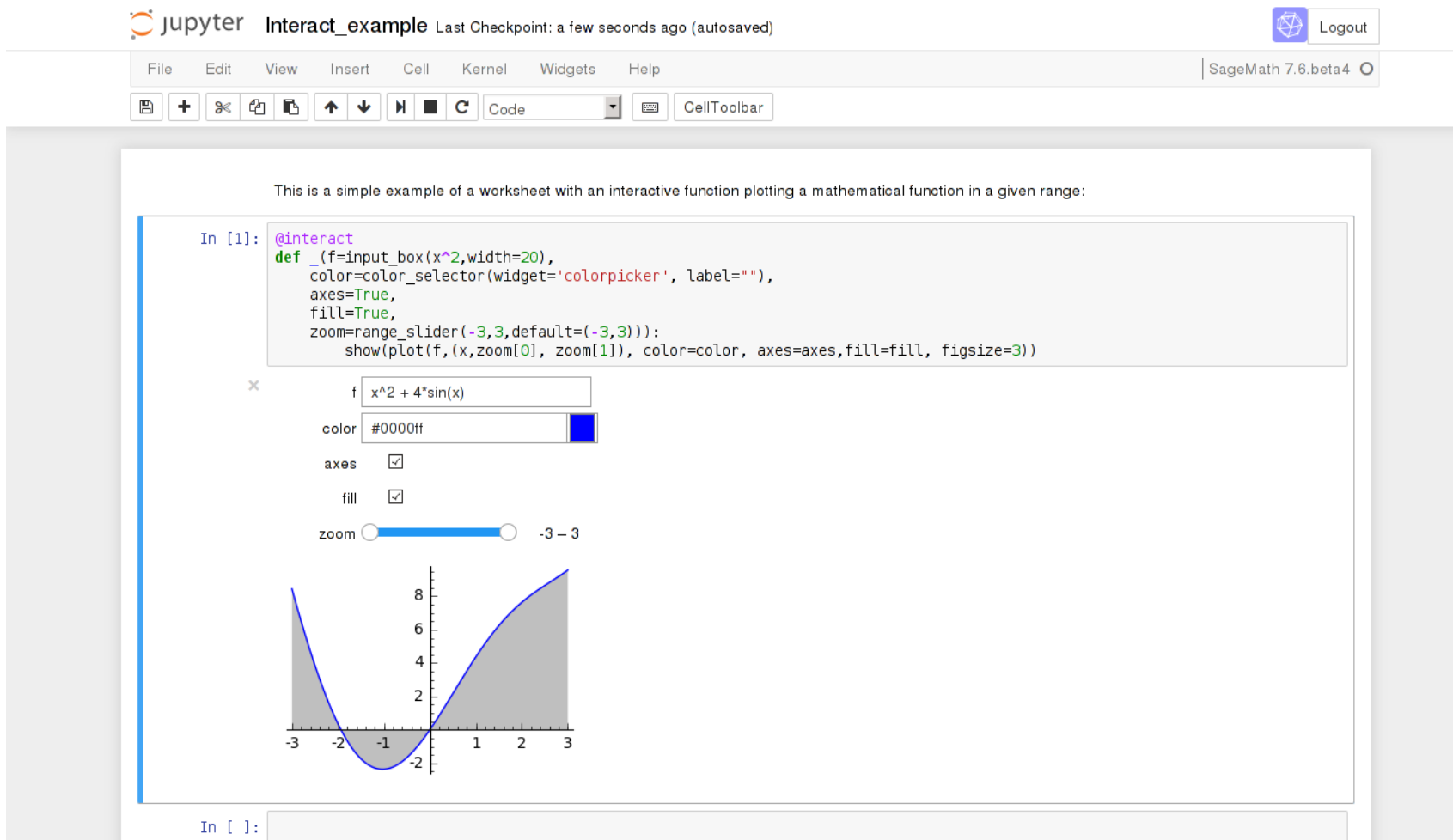


FIGURE 2. The same example interact, converted to JUPYTER using `sagenb_export`. Note the identical functionality, despite a different visual interface.

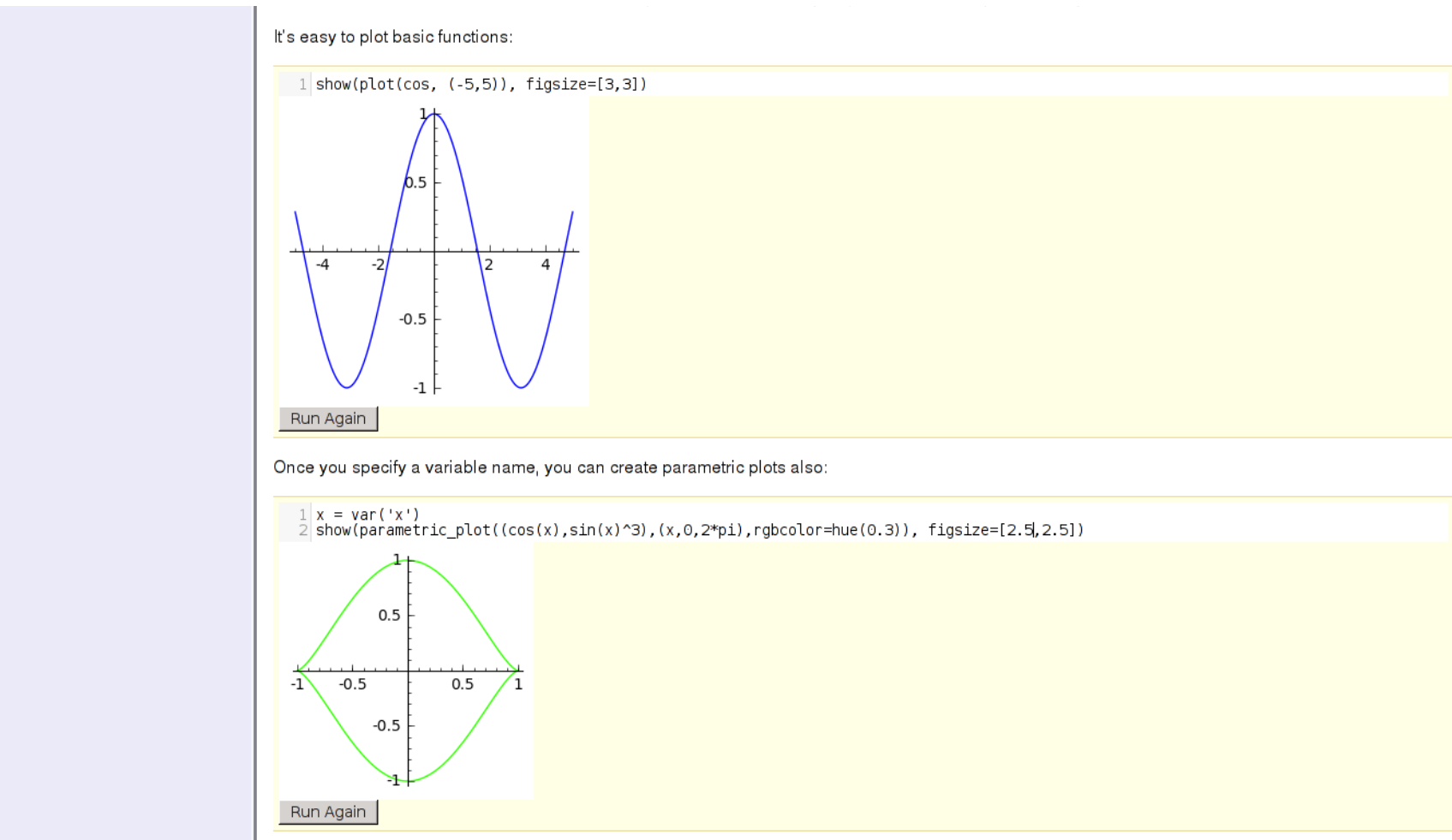


FIGURE 3. Live documentation in the JUPYTER notebook: the user can edit the examples in the documentation and run them using the “Run Again” buttons.

# Sage Mathematics Software



## The Sage notebook has changed

[Take me to the new Sage/Jupyter notebook](#)

[Run the old Sage Notebook](#)

To skip this screen and go directly to the new Jupyter notebook, run `sage --notebook=jupyter`  
To launch the old notebook instead, run `sage --notebook=sagenb` on the command line.

## Convert old notebooks to Jupyter

Click on any of the notebooks below to convert it to a new Jupyter notebook and open it in Jupyter:

ID	Name
admin:28	8. Algebra
admin:33	9. AES
admin:37	2. Analyse en plots
admin:38	3. Combinatoriek
admin:39	4. Lineaire algebra

FIGURE 4. The `sagenb_export` application, showing buttons to run the SAGE or JUPYTER notebook and a list of SAGE notebooks to convert to JUPYTER.

Disclaimer: this report, together with its annexes and the reports for the earlier deliverables, is self contained for auditing and reviewing purposes. Hyperlinks to external resources are meant as a convenience for casual readers wishing to follow our progress; such links have been checked for correctness at the time of submission of the deliverable, but there is no guarantee implied that they will remain valid.