Jupyter Notebooks, Automatic Grading and Moodle Submission

Lars Pieschel

November 1, 2020

1 Installation

You need a local installation of python, Anaconda with a new virtual environment is recommended. Use `pip install` or `conda install` to install the packages `notebook` and `nbgrader` (note: for `conda install` you need to change channel to conda-forge). If you only want to grade submissions, then you don't need the `notebook` package.

2 Creating Assignments with Formgrader

Choose a folder and execute the command `jupyter notebook`. This will start a notebook server and open a browser with the link to the server. After this click on the **"Formgrader"** tab to open the formgrader interface. Click on **"Add new assignment"** and enter an assignment name. Then click the name in the list of assignments to open up the assignment folder in jupyter notebook. Here create a new notebook and open it. To make metadata editing easier, click on **"View→Cell Toolbar→Create Assignment"**. This will add a small dropdown to every cell with which you can choose the cell type for nbgrader. There are five cell types. **"Manually graded answer"** and **"Manually graded task"** are not interesting for us. **"Read-only"** can't be edited by students. **"Autograder answer"** are cells in which a task is given and **"Autograder tests"** are cells for unittests for the task cells. Write code snippets in a autograder answer cell and surround the part that the student should write with two comments "### BEGIN SOLUTION" and "### END SOLUTION". For autograder tests use assertions to test the solution. Tests are by default visible for students. If you want to hide tests from the students then you have to surround the tests with two comments "### BEGIN HIDDEN TESTS" and "### END HIDDEN TESTS". Don't forget to set the amount of points a test cell gives when it doesn't fail.

When you are finished editing the notebook, close it and click generate in the formgrader. This will generate the student version of the assignment in the folder **/release/**.

3 Distributing Notebooks via Moodle

Create a new assignment, enter a name and upload the student version of the assignment notebook. Under **"Feedback types"** check the boxes **"Offline grading worksheet"** and **"Feedback files"**. Also change the maximum amount of points to that of the notebook.

4 Grading Notebooks

Note: The given scripts only work when your moodle language is set to english. You can change the language in the top right.

Download the Scripts from this repository. In moodle select the assignment, view all submissions, reveal identities and then download all submissions and copy the zip folder to **/imports/⟨assignment⟩.zip** in your jupyter folder. Download the grading worksheet and do the same (**/imports/⟨assignment⟩.csv**). Then execute `python collect_files.py ⟨assignment⟩ ⟨workbook name⟩`. Then run `nbgrader autograde ⟨assignment⟩` and `nbgrader generate_feedback ⟨assignment⟩`. Finally create a folder **/exports/** if it doesn't exist already and run `python update_gradesheet.py ⟨assignment⟩`. There should now be two files in the **/exports/** folder, a csv and a zip file.

5 Uploading Grades and Feedback

Note: Uploading the gradesheet only works when your moodle language is set to english. You can change the language in the top right.

After the grading step, on the moodle submissions page, choose **"Upload grading worksheet"**, choose the csv file from the exports folder and put a check in the box **"Allow updating records that have been modified more recently in Moodle than in the spreadsheet."** and submit. Then choose **"Upload multiple feedback files in a zip"** and upload the zip file.

6 Summary

Note: Set your moodle's language to english or some steps will not work. You can change the language in the top right.

1. Create Notebook
2. Upload to moodle, check offline grading and feedback, set max points
3. Download submissions, rename, put into imports
4. Download grading sheet, rename, put into imports
5. python collect_files.py ⟨assignment⟩ ⟨workbook name⟩
6. nbgrader autograde ⟨assignment⟩
7. nbgrader generate_feedback ⟨assignment⟩
8. python update_gradesheet.py ⟨assignment⟩
9. Upload gradesheet csv to moodle, check overwrite box
10. Upload feedback zip to moodle

7 Troubleshooting

`[AutogradeApp | ERROR] While processing assignment XYZ, the kernel became unresponsive and we could not interrupt it. This probably means that the students' code has an infinite loop that consumes a lot of memory or something similar. nbgrader doesn't know how to deal with this problem, so you will have to manually edit the students' code (for example, to just throw an error rather than enter an infinite loop).`

When you see this error without there being an infinite loop in the notebook, it may be caused by a problem with nbconvert. Try downgrading to nbconvert version 5.6.1.