Exercise Sheets 11–13 – Final Project

Sebastian Höffner Aline Vilks

Deadline: Wed, 7 July 2017 08:00 +0200

Submission and Grading

For the final project, submit your complete project folder as a zip file. Rename the example projects folder (`crashers`) to a name which fits your project. This is the directory you will submit eventually.

The final project counts for up to three individual exercise sheets. Each part (Proposal, Project code, Documentation) is worth one sheet.

Project structure

We attached a simple example project which you can use to build your own. The structure is as follows:

```
crashers (rename this)
├── docs
│   ├── conf.py
│   ├── index.rst
│   ├── modules (created on build)
│   ├── _templates
│   ├── _static
│   ├── Makefile
│   └── make.bat
└── src
    └── code files and dirs
```

Some hints about dealing with this project tree:

- You will have to replace everything in the `src` folder, and might have to delete the `docs/modules` directory from time to time.
- Write your code inside the `src` directory.
- Inside the `docs` directory, run `make html` to create beautiful documentation.
- Inside the `docs/_builds/html` directory (created with `make html`) you can run `python -m http.server 8080` to show the documentation.

- The documentation is viewable at localhost:8080[1] in your browser.

Packages

Attention

For this exercise sheet you will need an additional Python package. Install it using:

```
pip install sphinx
```

If you have any troubles with installing it, let us know as soon as possible so we can resolve issues fast.

You can find the documentation here:

- sphinx: http://www.sphinx-doc.org/en/stable/index.html

Part 1: Proposal

For the first part of the final project, navigate to the `docs` directory and open the file `index.rst`. Write down your ideas and goals about your project.

Remember to also adjust the `docs/conf.py` – you only need to change the project title and authors.

Think about a project proposal. Here are some guidelines of what can be included:

- ☐ What is the goal of your project? What "problem" do you solve?
- ☐ Can users interact with your program? If so, how?
- ☐ Do you use special libraries?
- ☐ What is your personal challenge?
- ☐ Can you use it for other courses? Is it just for fun?
- ☐ ...

Compile your documentation (`make html` inside the `docs` directory) to see if everything is fine.

In case you don't have any ideas for projects, check out our attached project proposals.

[1] http://localhost:8080

Part 2: Project code

Implement your project! This is a complete freestyle exercise, as you set the rules yourself!

Just make sure you put everything into the `src` directory so that your project works fine.

Part 3: Documentation

In case you wrote documentation for important functions while you were going through the second part, you are almost done with this part as well.

Add your results to the `index.rst` and update it if you changed anything while you were coding. Your python docs should be added automatically.

Bonus

If you feel really fancy, read up about how to write ReST documentation files for sphinx and structure your documentation accordingly.

Important remarks

- If you have no project ideas, feel free to talk with us. We can help you.
- Your projects don't need to be very big, but you should demonstrate that you can structure your code in functions, modules, maybe classes. Make sure to use string formats where useful, remove debug print statements.