## General feedback

* I participated in the Code Refinery workshop at Aarhus last week. Thank you very much for a good course! The workshop has inspired me to explore further into software development.
* Thank you for the great workshop! I enjoyed it a lot
* Thank you very much for organizing this workshop. I learnt a lot from it.
* Thanks a lot. I really appreciated all of your efforts in this workshop. It's very helpful for me to do reproducible research. :D
* Thank you [...] for the workshop. It was a great learning experiences on many new things. Git, Jupyter, automated testing and CMake were new to me and the workshop helps me a lot as I have started using these tools for my work now.
* I had a great time. Maybe if you bring the course to Odense at SDU, I'll see you there!

## 

## Git-intro

* Good examples
* Clearly explained
* Could be just a bit faster
* Guacamole recipe was very clear. Git is nice to learn interactively
* The class got a little confusing sometimes, could have been prepared better
* Good hands-on sessions, very informative
* Nothing really, even the jungle of platforms and versions were navigated expertly
* Course material is really nice both for following the flow of workshop and for reviewing
* I would prefer more structure, I had a hard time listening to a lot of seemingly random talk. And it seems like the “cosy” stuff in the beginning was VERY slow and then when things advanced the pace got really fast
* Very useful
* Maybe can make it more interesting by working some real problems (one for each)
* General introduction very clear
* In some points not so clear (last part of this morning lecture)
* Github example was good
* The presentation could have been faster in the start
* The diagrams showing graphically how git works were great for understanding
* Nice introduction to git for people who don’t know it. Examples were nice
* For an interactive session like this the speaker should be able to understand and correctly reply to questions. I did not find this satisfactory
* Nice introduction to git
* Proper duration. Takes time to learn in a way that “sticks”
* Slow start for experienced users
* The flow is good. Interactions are good
* More examples and more time in the exercise part would be more helpful
* Good walkthrough of all the essentials
* Somewhat slow pace
* Nice introduction with step by step example that allow to understand the basics
* You’re very good at tuning in to people’s pace
* Try more to highlight everyday “wins”
* Example has easy text files
* Stick more to the original plan of steps. Could reduce confusion
* Discussions help a lot
* More exercises would help
* Good simple tutorials that convey the fundamental functions and features
* The answers were vague sometimes (as in this criticism...)
* Hands on and fun, made the idea of git clear
* Not clear when we were supposed to start exercises on our own. We would go very slowly, and then suddenly he would execute 6-7 commands in silence
* Practical hands-on exercises
* I thought it was quite a large jump in difficulty level in the morning, switching from basics to branching and merging
* Good solid intro, with time for explanations
* nothing , really

## Jupyter

* Very good example notebook to start with
* I learnt a lot, even though i’ve used ipython before. Good content
* Too little time, the session could have been longer
* Good hands-on sessions, very informative
* Nothing really, even the jungle of platforms and versions were navigated expertly
* Many small exercises was good.
* More examples/exercises in beginning as it was after lunch so i was at leas a bit tired
* Very useful
* The lecture was a bit hasty in general
* Some python examples may be skipped
* Make the installation requirements more explicit, there were a few things that were installed in the session
* The speed of progression in the material was good, not too fast, not too slow
* Great introduction. After this I will probably try it out
* Examples were very nice and informative
* A very nice introduction to a concept i was not familiar with
* Maybe there could be some mentions of actual uses relevant for the people in the audience (i could think of some, but more suggestions would be nice)
* Nice intro and overview of what jupyter is and can do
* Try to come up with more somewhat “real” examples (such as the polynomial fit, which was very good)
* Nice overview of what you can do
* Maybe spend some more time on the “under the hood” part to make connection to “pure coding” clearer
* Nice introduction to magic operators
* Good coverage
* Not mentioning other possibilities such as spyder
* Nice to learn about [tools so famous] (?) already written notebook great help to start
* A little bit too fast presentation for a software that has a lot of options
* Nice and easy to follow for beginners
* Want to learn more about jupyter
* Very clear and interesting presentation with good tutorials
* Installation instructions could have been a bit clearer
* Nice examples with a lot of variety of possible different uses
* Maybe a little more details of what is going on behind the scenes (why does it launch in a browser?). How to save things?
* Good selection of “show cases” in that short amount of time
* Too few exercises
* Very comprehensive introduction. Good pace
* Use git :-)

## 

## Both Git-intro and Jupyter

* Great preparation + materials + really lucid + linear presentations
* Minor point, but what about more about practical use? E.g.. how to share jupyter notebooks? When do you branch or not? How to choose between terminal or a software solution for interacting with git?
* Competent instructors

## Git collaborative

* Good example, and that we really sorted out branch fork and we could do the steps in different ways
* Separate git product and github features
* Very useful exercises. Is it possible to start later say 9:30 am? Ok, I am just kidding. I would get up earlier
* Fantastic intro to remotes, branching, fetching, pull requests etc
* It’s nice that the git part continues today, as it starts to sink in
* Nice groupwork session, fun to try something close to realistic project
* It was a bit difficult to figure out what to do in the groupwork sessions - we did figure it out though
* Perhaps it could be made so everyone tries to be “admin” on the fork
* Figures were nice and informative. My memory got refreshed with some stuff i forgot!
* You should: explain carefully what are features of git and what are “peripheral” features of git-based services (e.g. “fork” on github etc)
* Though i am working with github for some time, I learned a lot today like branches, --rebase, bare repos. Very useful - thanks!
* Very nice walk through with easy to understand material
* Very slow pace resulted in us not getting through all the material
* Selection of topics was relevant
* Draw a clearer line between pure git functions and workflow provided by portals such as github

## Git archeology

* Very useful commands in git which were new to me
* Too short and i would also have heard about branch design
* Bisect exercise was good
* Amazing! Great new tool: git bisect
* More advanced method (bisection) are great/interesting
* The exercise was nice, as the prepared material was good for learning
* Could have been longer as the material was very interesting

## Git general comments

* (branch design) Did we cover this, i am a bit confused about what we talked about and if we ever covered this
* There is a bit of command overflow. Could it be possible to compile a list and put it on permanent display? Either on whiteboard or in a webpage within the coderefinery website (a reference list)

## IDE

* Good to think about the whole workflow and show IDE
* Didn’t apply to me really
* Difficult to follow compared to terminal
* IDE is a good tool and it would be better if the talker can prepare a simple example that’s easy to understand for us
* Do it more like the jupyter module. It was not clear if we were meant to follow. Many people “zoned ut”. Pity
* Not having the code and material available makes it hard to follow
* The examples seemed good
* IDE talk probably useful to some users, but not to me personally
* Nice tool to see
* Session was somewhat unstructured. I think the session would benefit from having a nice predefined project and follow a well defined tour of the program with the speaker as guide
* Maybe useful for people working exclusively with python but not for me. I use python, R, bash, snakemake (I am emacs guy)
* Pycharm session could be shortened or skipped - alternatively be livened up and clear in focus
* Nice to see quick overview of [cool?] features
* Little too slow going through, hard to keep focus, perhaps an overview of what we would go through would help
* Fine with introduction to possible [..illegible..]
* Very specific and uninformative presentation, should not be included in the future

## Automated testing

* Very good exercise
* Maybe if time permits, discuss more how to implement in other languages
* Interesting lessons with git/travis/coveralls. Not sure if I will use it, because it’s not worth the investment for the scale and type of programming that I do
* travis/coveralls exercise was very nice and informative! Nicely showed forking workflow and CI workflow
* Coveralls - very cool
* Testing session: don’t change it
* Really nice integrated exercise and nice that you were alert to the vibe in the room and therefore went fast onto exercise, also good to sum up and go through what happened

## CMake

* Maybe it will be better to explain how it works, and how to design it
* It is a good thing to learn cmake which will help me a lot in my future code compilation
* A more time and more demo would be more helpful
* The basic functions/abilities of cmake were made clear, somewhat
* The tutorial was not helpful. I just ended up copy/paste stuff in files and new commands like a monkey. It all worked, but i didn’t learn much
* Cool tool, good hands on intro to the basics
* Became too advanced, too quickly, without any explanation about what was going on under the hood
* Useful tool
* A little confusing. Maybe i knew too little from start because there were a lot of steps left unexplained
* Very interesting to see how to make libraries, create installer etc
* Too fast and unstructured, hard to follow what is going on
* Very interesting to learn that makefile is not only able to compile code but can also efficiently replace script.
* Maybe give more examples of what can be done
* Not relevant to python/perl/R
* But helpful for C/C++
* Didn’t attend
* Seems easier but it is hard for me to follow, because i do not have the background. More time in this may be better
* Interesting tool. Will consider using it in the future.
* I would have liked to go into a bit more detail from the start, in order to better understand what is going on
* Extremely clear...
* But in some points difficult to follow
* Having never used make, cmake or been working with compiled languages and building programs i got lost really quickly. Second half was roughly speaking gibberish to me. It would have helped me to have firmly established what stuff odes in the beginning (.neq. Why we want to use it)
* I will say Bjørn did a good job at going through commands / code blocks one by one and stating what they did

## Modular code development

* Good discussion
* Some real code structures could be discussed
* Very good
* Include some references for further reading
* Module programming was super useful, nice conceptual illustrations
* Really interesting and important topic. Could have an own course with only modular code strategies
* If time was not an issue, i would like to see real examples of bad and good code and more discussions
* I would suggest exemplifying some of the concepts also in code, as the discussion can get somewhat abstract and theoretical otherwise
* Maybe show some “common” bad code and explain why it is bad…
* It is good to do not introduce the different idea on a specific language but keep it general
* Maybe provide more examples of bad coding from your own experience
* Good tips for beginner
* Very useful to know recommendations for different languages
* I was offered very good suggestions. Very good structured workshop, i benefitted a lot. Thank you very much
* Maybe some more examples of bad and good code.
* Good idea to group people using the same language
* interesting
* Could perhaps include some more examples of “good code” and “bad code”
* Good
* Really interesting. One of main reasons for me to join CodeRefinery
* discussing in a language sorted frame worked really well
* Improvement could be example of good modular code written as schematic/abstract as possible to be relevant to all

## Software licensing

* It’s good to be introduced with licensing
* Due to short time the flow is a bit fast
* Good advice
* A couple of practical use cases in the real world would help
* Good and not too long presentation of the issue
* Goot to get a bit of an idea of the topic
* Maybe include a visual overview of the most common licenses
* Licensing seemed a bit out of place in a practical programming course
* Very nice presentation of a very under appreciated problem/aspect of code development
* I like that you included this topic
* Maybe 10 more minutes with examples when it’s good to choose different licenses
* Very good, with a great overview
* More details of how to specifically put the necessary files in the correct places in your project
* First time that i heard about the different licenses. It is very useful to learn about it
* Good to know before i get troubled
* It is good to know our rights and obligations
* Good (maybe too fast)
* Very interesting and important. Nice quick overview
* Improvement could be emphasis on making the decision. Perhaps some different cases/scenarios

## Documentation

* It is very new. So, it’ll help me to document my code in future
* Time is short. More documentation would be more help
* Good exercises, everything worked
* Good tutorials for very neat services
* Hands-on examples
* Make a bit clearer how git pages and readthedocs are different
* Great examples, very clear intro to read-the-docs
* Not clear about git-hub pages and what kind of format it supports
* Nice presentation showing some nice and interesting tools
* Maybe start the session with a creation of a simple README.md file for github
* Useful to do all the steps yourself even if it’s just test cases
* ?
* The github service could have been more flashed out
* The different services seemed good
* Maybe spend a little bit more time to present the different possibilities of RST
* Cool tools
* (sphinx) One of the best of all in this workshop, thanks a lot I love it!!
* (github-pages) it is cool. Easy way to host a site
* We can document our own codes, that’s cool.
* Because of time limit, i can only have a flavor of this. Hope i will use this in the future
* Also an interesting topic
* Not sure if i will use it or not
* Good, clear with good exercises
* Interesting tools, good with something practical in the afternoon
* The pace was a bit fast with the code-along

## General

* Hall not spacious enough
* chairs /tables not very comfortable for a full day session
* But overall its a great work, especially when free of cost
* Idea: make set of small exercises and examples to be studied before start of the workshop? To bring people on equal footing
* Overall a very nice course and 3 days well spent!
* Some things to change:
* Sometimes could have moved faster or divided into two groups with different speeds
* Small thing: on your gh-pages, open links to a new tab :) I was going back and forth a lot
* I would have liked to spend more time on modularizing code. More concrete examples, more code studies
* Thanks!
* A very nice workshop!
* Presenters seemed very knowledgeable and competent, and VERY dedicated to teach the subjects!
* In general: excellent course!
* Good choice of subjects. Good mix of interactive/noninteractive sessions
* Very little need of improvement
* Perhaps overview pages for commands covered would be helpful