

Software Development Tools

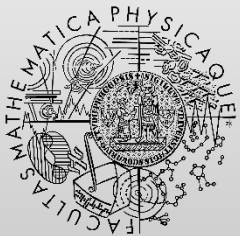
<http://d3s.mff.cuni.cz>

Department of
Distributed and
Dependable
Systems



Pavel Parízek

parizek@d3s.mff.cuni.cz



FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

Goal of this course

- Basic overview of available tools for common tasks in software development
- Practical experience with selected tools
- This can be useful in
 - Studies: assignments, individual projects, team SW projects, bachelor thesis, master thesis
 - Commercial software development: productivity
 - Work on open-source projects (recommended!!)

Expectations

- User knowledge of UNIX/Linux and Windows
 - Command line (shell), writing small scripts, system utilities, common user applications
 - Practical experience is an advantage (for UNIX/Linux)
- Basic knowledge of a mainstream programming language (C/C++, Java, C#, Python, ...)
 - Extent of introductory course at MFF
 - Sufficient to attend in the same term
 - Practical experience is an advantage

Content

- Version control systems (Git) S1
- Software building (Make, Maven, Gradle) S2
- Deployment infrastructure (Docker) S2
- Functional testing (JUnit, NUnit) S3
- Debugging (GDB, Valgrind) S3
- Searching for bugs (SpotBugs, Clang) S3
- Recording events (strace, log4j) S4
- Issue trackers (Bugzilla, JIRA)
- Generating documentation (Doxygen)
- Generating code from templates
- Performance analysis (GProf, JVisualVM) S4

Structure of each lab

- Feedback on the previous assignment
- Introduction to a given domain
 - Basic concepts, what problems the tools address
 - Description of selected tools (technical details)
- Technical details and specific features of tools
 - Syntax of commands, basic configuration
- Practical tasks
 - Individual work during labs, online documentation
- Space for answering your questions (discussing features of tools)
- Providing advice regarding usage of tools in SW development
- Homework assignment

Report

- Purpose: show that you
 - Understand the assignment (specific task)
 - Found out how to solve the given task,
 - Choose the right (proper) solution, and
 - Managed to do it successfully in practice
- Form
 - Text file, ASCII, Czech/Slovak or English
 - Attachments (source code, textual outputs)
- Submission: ReCodEx
 - email as fallback

Report – content

- Your full name and email address
- Specification of individual tasks
 - Copy from the original assignment
 - Put it just before description of your solution
 - The recommended way: edit the file with assignment directly
- Your solution
 - What commands you run (including arguments)
 - Where you run the commands (in which directory, etc)
 - Output of the tool (just important parts): console, files
 - Brief explanation (comment) of your solution (decisions)
 - If there are multiple possible solutions or you decided to use a non-trivial approach

Report – example

John Doe, john@doe.com

1. Create a directory named „test“ in /tmp and discuss situations in which your solution would fail

```
> cd /tmp
/tmp> mkdir test
```

Creating a directory in this way could fail for these reasons: the directory /tmp does not exist or the current user does not have the effective access privilege “x” on the directory, the current user does not have the privilege “w” on the directory “/tmp”, a physical device mapped to the directory /tmp is full (i.e., there is not enough space for a new directory).

2. ...

3. ...

Grading

- Reports
 - Important aspects: provide solution for all tasks, clarity, correctness
 - Unsatisfactory reports → one more week for resubmission
- Credit (“zápočet”)
 - Homework assignments (6 out of 9)
 - At least one assignment from each group of topics (S1-S4)
 - Considering at most two assignments from each group
- Attendance is **optional** but:
 - Highly recommended for topics that you need to learn more
 - Space for questions and discussion (help when you get stuck)
 - Individual consultations not possible (very exceptional cases)
- Other means of fulfilling the course
 - Usage of some tool on a student project (describe your experience)

My vision for the whole course – part I

- Labs: focus on activity of students (practical tasks)
- Interactive mode of teaching: questions, higher student activity
 - Much less frontal lecturing (teacher standing before the class and talking for 90 minutes)
- Students should work (play with the tools, get some experience)
 - “Controlled self-study where I can help to a large extent”
- **Do not be afraid to ask (!!)**, when something does not work for you

My vision for the whole course – part II

- Three groups of tools
 - Very common: you should certainly know all of them and be able to use them
 - Interesting: also important tools that you should be aware of (in my opinion)
 - Other tools: general overview (so you know what to look for in time of need)
- Important **knowledge to take away** from the course
 - How to use the specific tools (commands, configuration)
 - Software engineering processes around them (context)
- Sometimes we do not manage to go through the whole presentation and tasks in the lab => please try the rest at home
- Sample solutions presented very rarely (almost never)
 - Discussing common mistakes at the beginning of the next labs
- Tweaking the content and form every year based on your feedback

My vision for the whole course – part III

- Another view on goals for the course
 - You should learn something really useful here
 - Mostly by **actively playing with** some **tools** yourself
 - Not me just presenting tools (overview, demo)
- Labs will be dedicated to my presentation and interactive discussions (questions, answers)
 - Mostly independent work on practical tasks
 - You need to learn (how to use) those tools (not me)
- Why: usage of tools on daily basis in your life
 - Studies: bachelor thesis, SW project, master thesis
 - Future career in software industry
 - Roles: developer, SW engineer, team leader, ...

Response to student evaluations

- Basic features of some tools (Git, CMake, unit testing) covered at other classes
 - Different tools (learn something new)
 - Advanced features (complex use cases)
- Sharing more general information
 - Not specific just to a single tool (platform)
 - Recommended practice and experience
- Broaden your horizons and knowledge
 - Show tools that you can learn (try, ...)

Related courses

- Highly recommended (!!)
 - Introduction to Software Engineering (NSWI041)
 - Programming of Web Applications (NSWI142)

Contact

- Web: <https://d3s.mff.cuni.cz/teaching/nswi154/>
- Email: parizek@d3s.mff.cuni.cz
- Office 309