Software Development Tools

http://d3s.mff.cuni.cz

Pavel Parízek
parizek@d3s.mff.cuni.cz
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#)
  - Extent of introductory course at MFF
  - Sufficient to attend in the same term
  - Practical experience is an advantage
Software Development Tools

Introduction

- Version control systems (Git, SVN, Hg)  
- Software building (Make, Ant, MS Build)  
- Functional testing (JUnit, NUnit)  
- Debugging (GDB, Valgrind)  
- Searching for bugs (FindBugs, Clang, FxCop)  
- Recording events (strace, log4j)  
- Issue trackers (Bugzilla, Trac, JIRA)  
- Generating documentation (Doxygen)  
- Generating code from templates  
- Performance analysis (GProf, JVisualVM)
Structure of each lab

- Introduction to a given domain
  - Basic concepts, what problems the tools address
  - Description of selected tools (technical details)

- Specific features of tools
  - Syntax of commands, basic configuration

- Practical tasks
  - Individual work during labs, online documentation

- Homework assignment
Purpose: show that you
- Understand the assignment (specific task)
- Found out how to solve the given task, and
- Managed to do it successfully in practice

Form
- Text file, ASCII, Czech/Slovak or English
- Attachments (source code, textual outputs)
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)
    - If there are multiple possible solutions or you decided to use a non-trivial approach
1. Create a directory named „test“ in /tmp and discuss situations in which your solution would fail

```bash
> 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: clarity, correctness, participation at the lab
  ▪ Unsatisfactory reports ➔ one more week for resubmission

• Credit
  ▪ Regular attendance (75%)
    • You can do additional homework assignments (over the minimum of 6) to compensate for skipped labs
  ▪ Homework assignments (6 out of 9)
    • At least one assignment from each group of topics (S1-S4)
    • Attended the lab: basic variant of a homework assignment
    • Missed the lab: few additional tasks (bit more advanced)

• Alternative means of fulfilling the course
  ▪ Skipped labs ➔ more complex tasks over real systems
  ▪ Usage of a 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)

- Students should work (play with the tools, get 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

- I speak quite fast so be patient with me
  - but I am trying to slow down

- Attendance is **optional:** when you skip some lab, you may get an additional tasks for homework to compensate
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
Contact

- Web: [https://d3s.mff.cuni.cz/teaching/nswi154/](https://d3s.mff.cuni.cz/teaching/nswi154/)

- Email: parizek@d3s.mff.cuni.cz

- Office 309