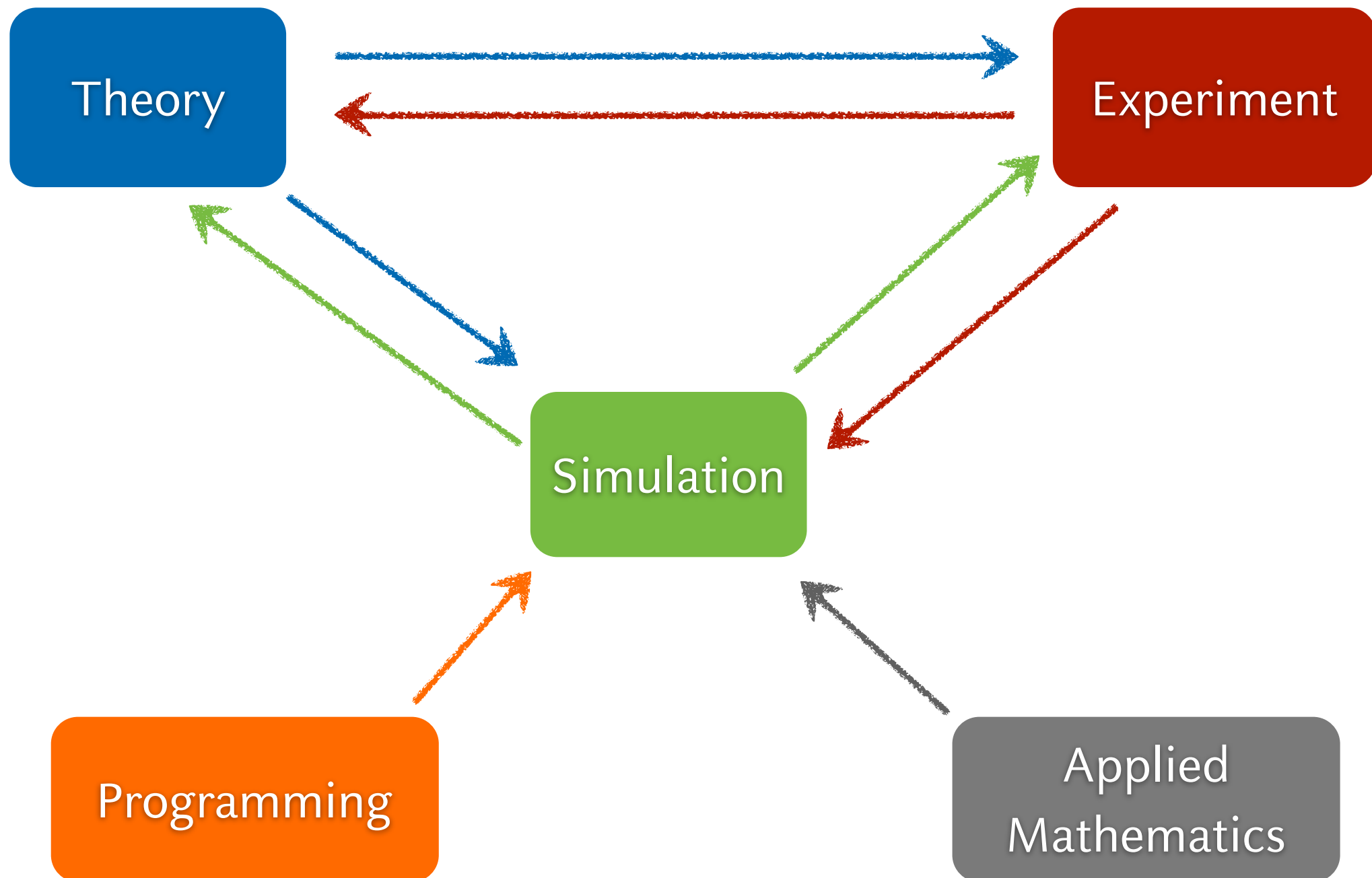


# Numerical Simulations II WS18/19

Dr. Götz Lehmann

## Organization of the class

- 2h lectures + 2h lab/tutorial per week + final exam  $\Rightarrow$  6 CP
  - Lecture: Tuesday, 16:30-18:30, HS 6C
  - Final exam will (probably) be writing a computer code (problem of organization)
- Attendance at lab classes is mandatory
- Homework: To be handed in via GitHub, 2/3 of problems have to be solved
- Lab assistant: Lars Reichwein



# Topics of the lecture

- Programming
- Numerical methods

## Numerical methods

- Focus will be on initial value problems for
  - ordinary differential equations (ODEs)
  - partial differential equations (PDEs)
- This will require dealing with
  - finite difference approximations to derivatives
  - systems of linear equations
  - root finding of nonlinear equations
  - discrete Fourier transformation

## Problems to be discussed

### ODEs

$$\frac{d\vec{y}}{dt} = f(\vec{y}(t), t)$$

Single-step methods: Runge-Kutta

Multi-Step methods: Adams Methods

Verlet Methods

$$\begin{aligned}\dot{p} &= -\frac{\partial H(p, q)}{\partial q} \\ \dot{q} &= \frac{\partial H(p, q)}{\partial p}\end{aligned}$$

Symplectic methods for Hamiltonian systems

## Problems to be discussed

### PDEs

$$\nabla^2 \phi = -\varrho$$

Poisson eq.

$$\frac{\partial}{\partial t} u - u \frac{\partial}{\partial x} u = 0$$

Burgers eq.

$$\frac{\partial^2}{\partial t^2} f - \nabla^2 f = 0$$

Wave eq.

$$i \frac{\partial}{\partial t} \psi + q \frac{\partial^2}{\partial x^2} \psi + |\psi|^2 \psi = 0$$

NLSE

$$\frac{\partial}{\partial t} f + \nabla \cdot j = 0$$

Continuity eq.

$$\frac{\partial}{\partial t} f - D \nabla^2 f = 0$$

Diffusion eq.

## Programming

- C++, the language
  - as the better C
    - datatypes, functions, pointers, references
  - beyond C: Object oriented programming
    - classes, namespaces
  - external libraries: *Avoid Not invented here syndrome*
  - The basics of multi-core applications
  - You will need a C++ compiler for your homework!
- C++, the eco-system
  - Git as an example for a version control system
  - Makefiles / CMake
  - Valgrind



# How to get your hands on C++

## ■ Windows:

- MS Visual Studio via [www.dreamspark.com](http://www.dreamspark.com)

- ...you're on your own

## ■ Linux:

- Lab class will use Linux

- proper installation next to Windows (if your advanced, you know what you do)

- live USB (fairly easy, use Linux Live USB creator to create bootable USB stick)

- <http://www.linuxliveusb.com> (choose a “persistant” capable distribution)

- virtual machine (medium complexity, best result for the price)

- Install VirtualBox for Windows <https://www.virtualbox.org>

- Download a Linux distribution (Fedora, OpenSUSE, Mint, ....)

- Install Linux in the virtual machine

## ■ Mac:

- Install XCode app. Done. Easy as that.

# If you want to work with our Linux....

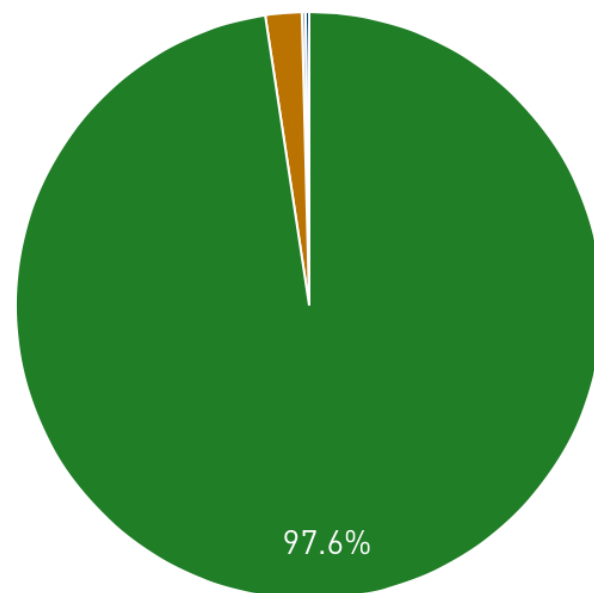
- Download & Install VirtualBox
- Download the Appliance file we provide at:  
<http://goetz.tp1.uni-duesseldorf.de/data/NumSimII.ova> (~ 2.5 GB)
- In VirtualBox, import the Appliance, leave all settings as they are
- Now you should have a working virtual machine that contains all the stuff we need
- Username and all passwords (i.e. for user and for root user) are „simu“

## Linux is the backbone of computational science

Virtually all big machines use Linux

Everything beyond one computer runs on Linux...

Operating system Family System Share



- Linux
- Unix
- Mixed
- Windows

top500.org



SGI UV2000, 512 Cores, ZIM

## Git & GitHub

- Git is software that allows you to organize your source-code
- We will use GitHub to *hand out & collect* sources for lab classes and homework



**Introducing GitHub**  
**Bell & Beer**



**Preißel, Stachmann**



**Riedel**

## Single User Git scenario - Using Git as a log book

repository

A repository contains a set of files  
May reside on a local disk or somewhere on the internet

local work directory

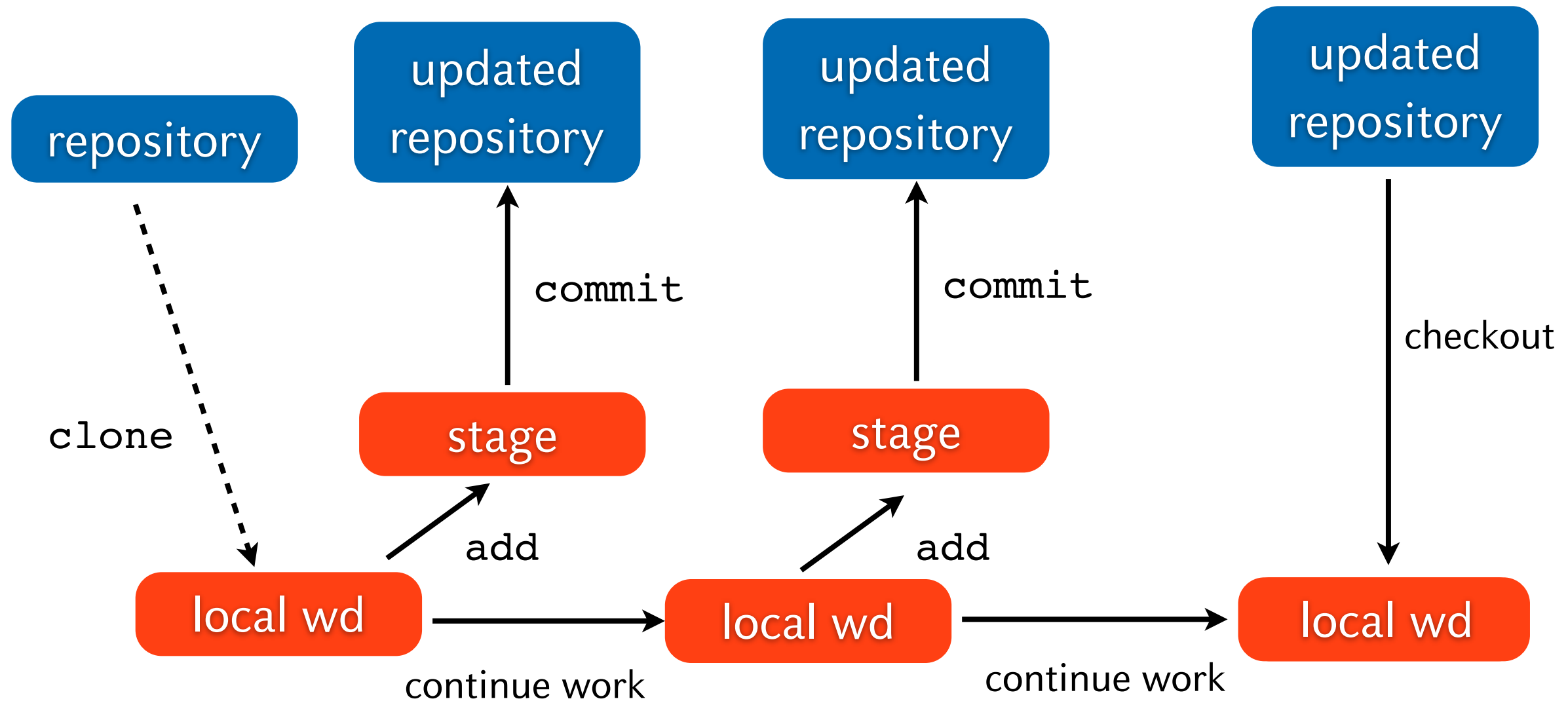
A directory on your hard disk

By `cloning` the repository into your local working directory, you create a local copy of the files contained in the repository.

After editing the files, you `add` the changes to the staging area. This is an intermediate step before sending them actually to the repository by `committing` them.

The repository keeps track of all older versions of all files. You can easily compare changes between different `commits`.

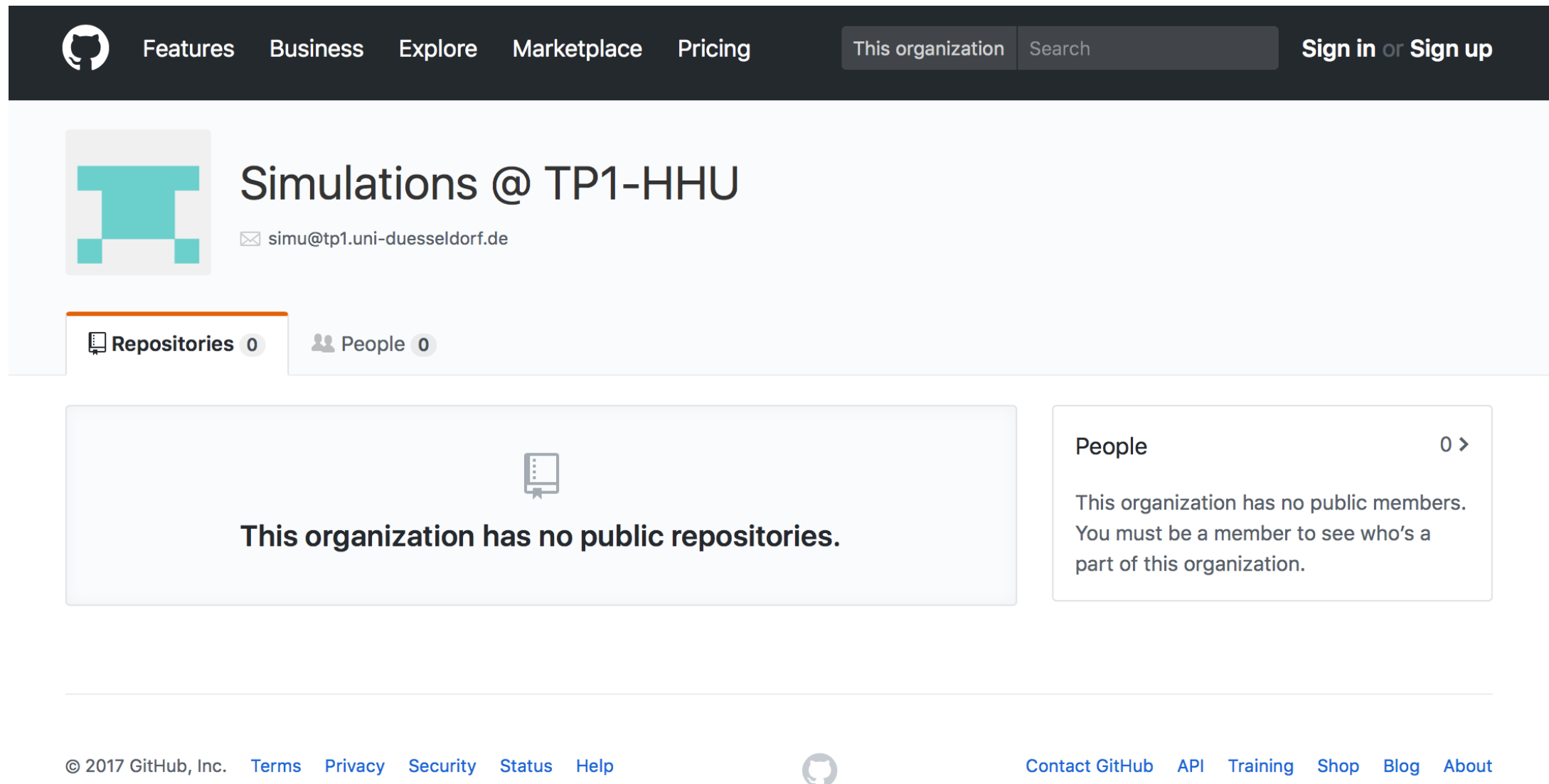
## Single User Git scenario - Using Git as a log book



- GitHub is a platform which allows to host repositories online
- Accounts are free, you will have to create one
  - free accounts can only have „public“ repositories, i.e. everybody can read all the files
  - pay accounts can have a certain number of „private“ repositories
- Our account is named TP1-HHU, thus our address is  
<http://github.com/TP1-HHU>
- We will use „GitHub“ and „GitHub Classrooms“
- GitHub Classrooms is a way in which we will hand out and collect Homework assignments



- At <http://github.com/TP1-HHU> there is not much to see for you right now, because everything we have there is *private*



The screenshot shows the GitHub organization page for "Simulations @ TP1-HHU". The organization's email is simu@tp1.uni-duesseldorf.de. The page indicates that there are 0 repositories and 0 people. A message states: "This organization has no public repositories." Another message states: "This organization has no public members. You must be a member to see who's a part of this organization." The footer includes copyright information for GitHub, Inc. and links to Terms, Privacy, Security, Status, Help, Contact GitHub, API, Training, Shop, Blog, and About.

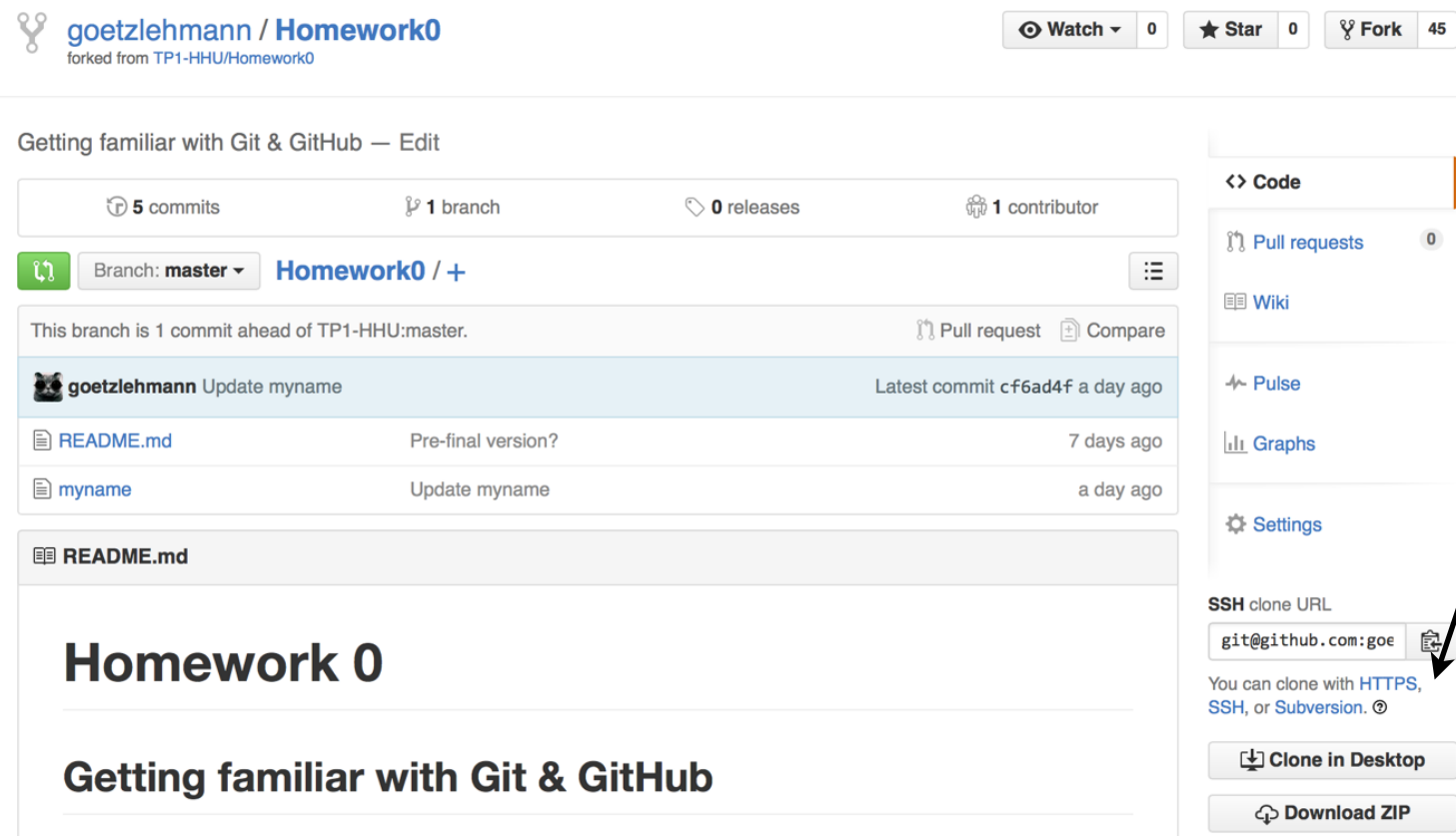


## Cloning your remote repository

Terminal commands:

```
git clone /path/to/repository ← for a local repository
```

```
git clone username@host:/path/to/repository ← for a remote repository
```



The screenshot shows a GitHub repository page for 'goetzlehmann / Homework0'. The repository is forked from 'TP1-HHU/Homework0'. It has 5 commits, 1 branch, 0 releases, and 1 contributor. The main branch is 'master'. The repository is 1 commit ahead of the upstream 'TP1-HHU:master'. The latest commit is 'goetzlehmann Update myname' with hash 'cf6ad4f' from a day ago. The repository contains files 'README.md' and 'myname'. The README.md file is open, showing the title 'Homework 0' and the subtitle 'Getting familiar with Git & GitHub'. On the right side, there are links for 'Code', 'Pull requests', 'Wiki', 'Pulse', 'Graphs', and 'Settings'. The 'SSH clone URL' is 'git@github.com:goe'. Below it, it says 'You can clone with HTTPS, SSH, or Subversion.' and there are buttons for 'Clone in Desktop' and 'Download ZIP'. An arrow points from the 'HTTPS' link to the list of steps on the right.

1. Press HTTPS here

2. Copy HTTPS URL

3. Clone:

```
git clone https://github.com/  
goetzlehmann/Homework0.git
```

## Add & Commit

- Your local repository consists out of three things:
  - The working copy of your files
  - The index and the head (which are invisible to you if you don't look closely)
- `git add` and `git commit` are the most frequent commands that you will use
- `git add <filename>`  
*Stages* the changes made to the file for a subsequent commit to the repository. The idea is to first gather all the changes to different files which should be submitted to the repository upon the next commit.
- `git commit`  
This actually sends all changes which have been staged to the local repository. Upon execution of this command a text editor will open up, where you should enter a commit comment. This editor might be VI, tricky because you need to know the keyboard shortcuts to find your way out! Better: `git commit -m „Changed variable names for`

## Pushing to remote repository

- After committing the files we can push the changes in the local repository to the remote repository residing on GitHub by doing a push
- `git push origin master`  
Submits the changes from local to remote repository. This will be done via the HTTPS protocol since we used this protocol when we cloned the repository. This will ask for username and password on GitHub to verify that you are allowed to submit to the repository.
- `git remote show origin` / `git remote -v`  
Displays from where the repository was cloned and to which remote repository changes will be pushed.

## Complete Example

```
DeepThought2s-MacBook-Pro: ~/work/test $git clone https://github.com/goetzlehmann/Homework0.git
Cloning into 'Homework0'...
remote: Counting objects: 15, done.
remote: Total 15 (delta 0), reused 0 (delta 0), pack-reused 15
Unpacking objects: 100% (15/15), done.
Checking connectivity... done.
```

Clone

```
DeepThought2s-MacBook-Pro: ~/work/test $cd Homework0/
DeepThought2s-MacBook-Pro: ~/work/test/Homework0 $ll
total 16
```

Edit

```
-rw-r--r--  1 goetz  staff   2,3K 20 Okt 19:37 README.md
-rw-r--r--  1 goetz  staff   12B 20 Okt 19:37 myname
```

```
DeepThought2s-MacBook-Pro: ~/work/test/Homework0 $vi myname
```

```
DeepThought2s-MacBook-Pro: ~/work/test/Homework0 $git add myname
DeepThought2s-MacBook-Pro: ~/work/test/Homework0 $git commit -m "inserted my name"
[master 743d7cb] inserted my name
Committer: Götz <goetz@DeepThought2s-MacBook-Pro.local>
Your name and email address were configured automatically based
on your username and hostname. Please check that they are accurate.
You can suppress this message by setting them explicitly:
```

Add & Commit

```
git config --global user.name "Your Name"
git config --global user.email you@example.com
```

After doing this, you may fix the identity used for this commit with:

```
git commit --amend --reset-author
```

```
1 file changed, 1 insertion(+), 1 deletion(-)
```

```
DeepThought2s-MacBook-Pro: ~/work/test/Homework0 $git push origin master
Username for 'https://github.com': goetzlehmann
Password for 'https://goetzlehmann@github.com':
Counting objects: 5, done.
Delta compression using up to 8 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (3/3), 307 bytes | 0 bytes/s, done.
Total 3 (delta 0), reused 0 (delta 0)
To https://github.com/goetzlehmann/Homework0.git
cf6ad4f..743d7cb master -> master
```

Push

Now everything is live and updated in your private GitHub repository. Send a pull request next.

## Branches

- Each repository may have several branches. The default branch is called `master`.
- If bigger changes are made to a project, first a new branch is created, e.g. `new_feature`. The new branch contains all files that were present at the time that branching was initiated.
- All changes which are related to the new feature are then committed to the new branch `new_feature`.
- Once the development in the branch is done, the changes are merged into the master branch and the new branch is deleted.
- For us the `master` branch is usually sufficient, you do not need to create branches. If something behaves weird in Git/GitHub, checkout if you did not accidentally create a new branch.