Welcome to RES 2018

RES, Lecture 00

Olivier Liechti
Miguel Santamaria

heig-vd

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD

www.heig-vd.ch

https://github.com/wasadigi

https://github.com/edri

Agenda

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **Background**
  - Who are we?
  - What do we do at the HEIG-VD and why?
- **Course objectives**
  - Network programming
  - Application-level protocols
  - Web infrastructures
- **Tools**
  - Crash course on Git & GitHub

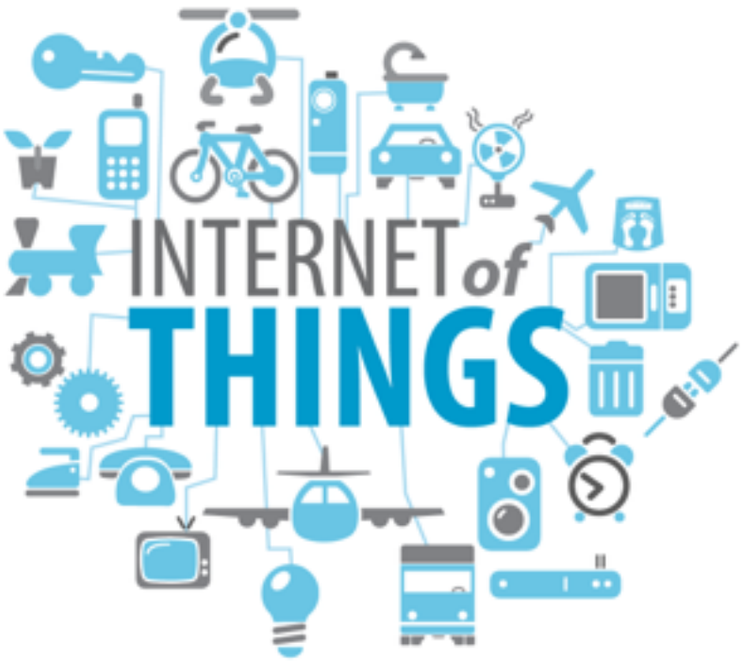
Background

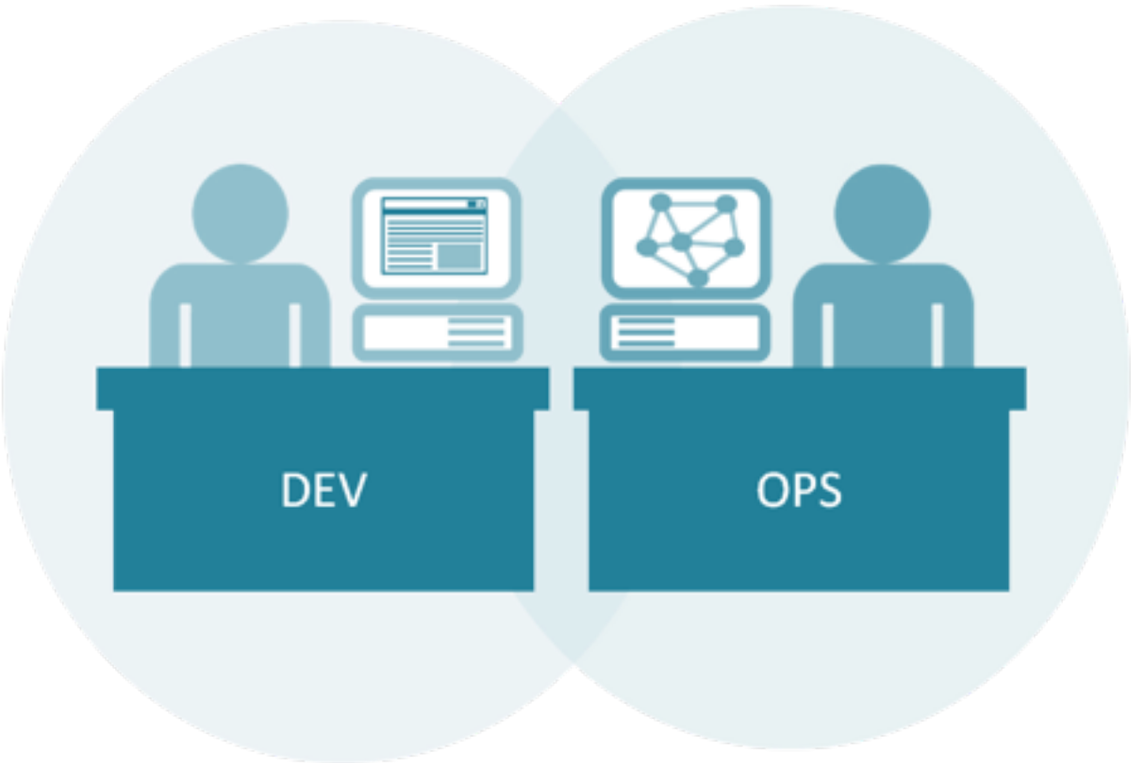
Personal background

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Personal background

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch





R&D at HEIG-VD

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Startups

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

NETGUARDIANS
Risk Under Control

sysmosoft

OBJECTIS

STIGNERGY
Energy Control & Optimization

novaccess

SimplicityBio
IN SILICO
BIOMARKER DISCOVERY

Future instruments

eBiz
Games

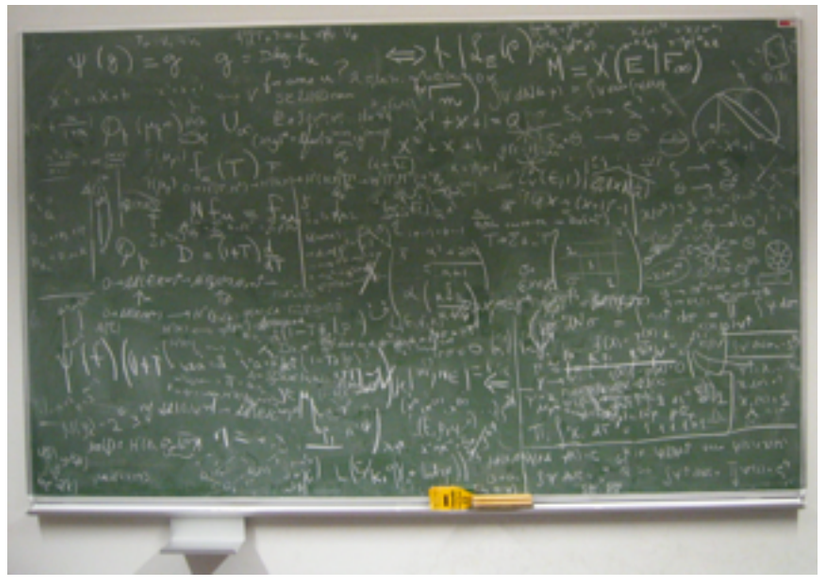
VIACAM
vision for safety

strong.codes

AVALIA

Why do we do applied research?

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



There are plenty of opportunities!

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

I WANT YOU

TO LIVE YOUR BEST LIFE!

Course Objectives

RES

2 + 2

TWEB

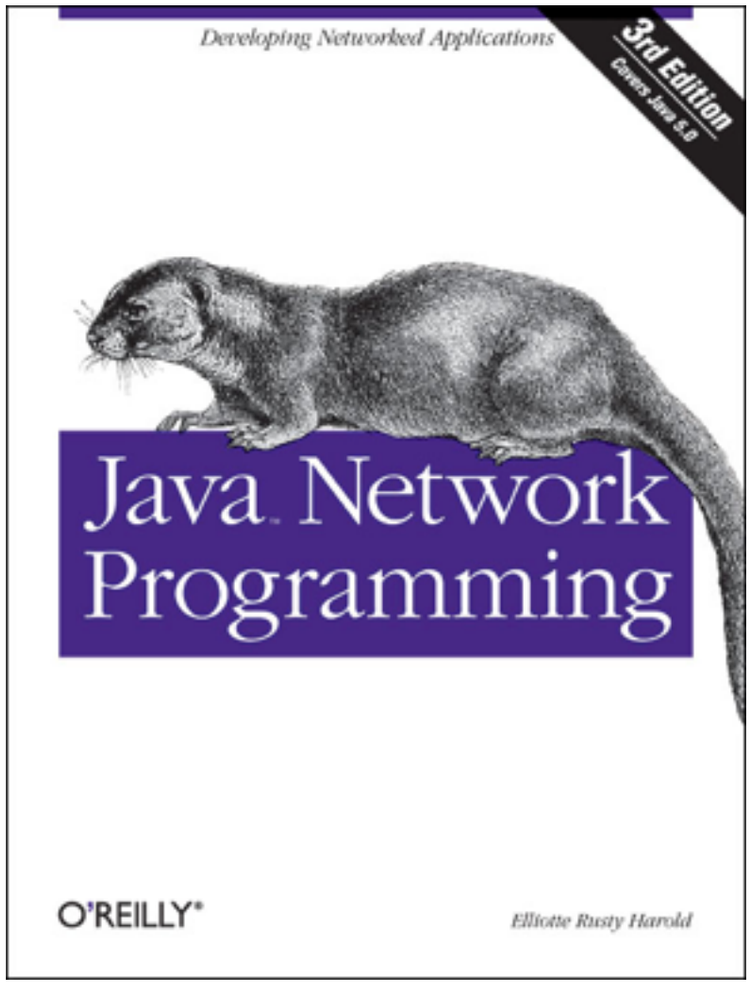
3 + 3

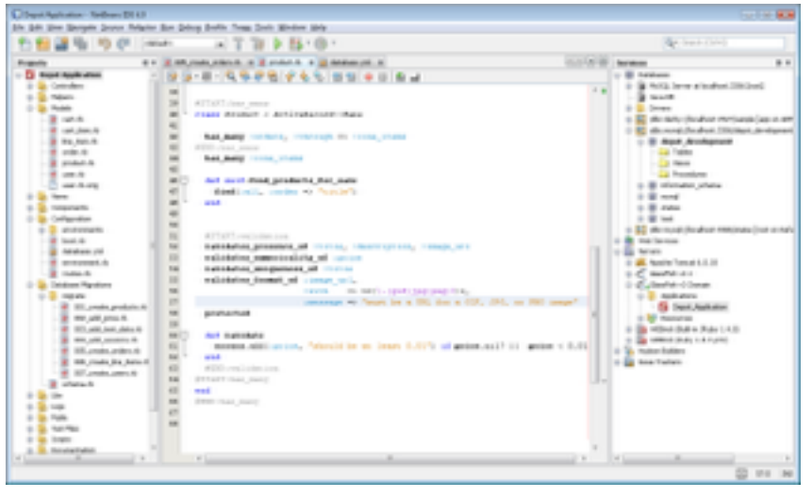
IL + TS

AMT

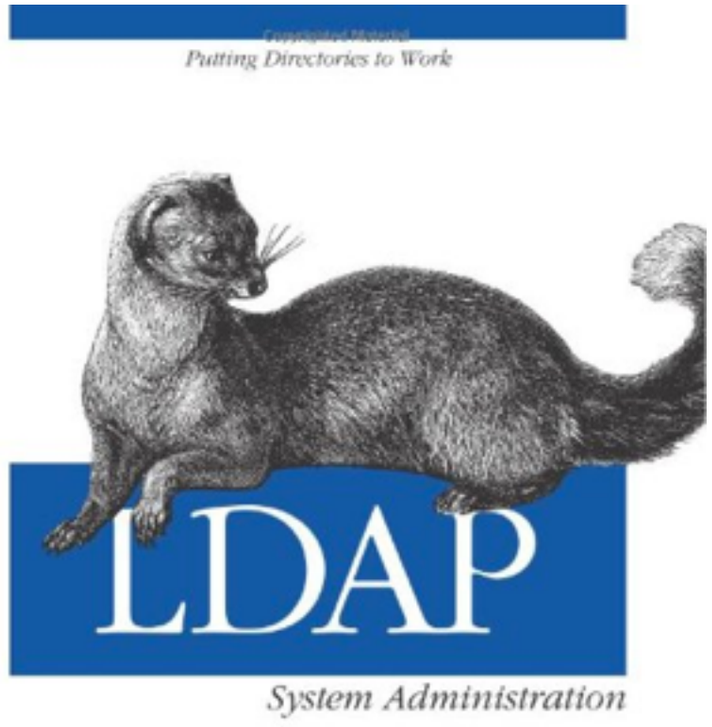
3 + 3

IL + TR + TS


Developing Networked Applications
3rd Edition
Covers Java 6.0
Java Network Programming
O'REILLY
Elliotte Rusty Harold




NGINX
Nginx as Reverse proxy for APACHE
APACHE


Putting Directories to Work
LDAP
System Administration

O'REILLY
Gerald Carter
Copyrighted Material
docker

High-level planning

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **IO programming** in Java (starting with files, encodings, etc.)
- **Network programming** in Java (and Javascript)
  - How do we write TCP clients and servers?
  - How do we use UDP in our own programs?
  - The **SMTP protocol**
  - The **HTTP protocol**
    - Model, syntax, mechanisms
  - Design and implementation of **web infrastructures**
    - Servers, proxies, reverse proxies, load balancers
    - Building a data center on your laptop with Docker
- *(Managing user identities with **LDAP**...)*

GAPS: plan d'études

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Conditions pour la programmation automatique de cette unité selon le plan d'études :

L'étudiant-e doit avoir obtenu une note supérieure ou égale à la limite de compensation dans les unités : TIB

L'étudiant-e doit avoir suivi ou suivre en parallèle les unités : POO1

Objectifs

Ce champ est obligatoire

- Programmation Réseau
  - Etre capable de concevoir une application client-serveur
  - Etre capable d'implémenter un client et un serveur en utilisant la Socket API dans différents langages
- Protocole HTTP
  - Connaître les concepts principaux du protocole
  - Etre capable de concevoir et réaliser une infrastructure HTTP avec un reverse proxy et plusieurs serveurs
  - Etre capable d'implémenter le protocole en utilisant la Socket API
- Protocole LDAP et annuaires Internet
  - Connaître le modèle LDAP et les éléments principaux du protocole
  - Etre capable d'installer et de configurer un serveur LDAP
  - Etre capable d'utiliser un client LDAP pour accéder à un serveur
  - Etre capable de transformer et d'importer des données dans un annuaire
- Protocoles de messagerie
  - Connaître les principaux protocoles relatifs à la messagerie électronique
  - Etre capable d'implémenter un client de messagerie simple
- Protocoles de transfert de fichiers et d'accès à distance
  - Connaître les protocoles de transfert de fichiers et d'accès à distance, ainsi que leurs principales utilisations (y compris tunneling/forwarding)
  - Etre capables d'utiliser des outils de synchronisation de fichiers à distance (e.g. rsync, ...)

GAPS: plan d'études

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Cours	32
Concepts de programmation réseau et présention de la Socket API dans différents langages	10
HTTP: étude du protocole et des éléments liés à l'infrastructure (e.g. reverse proxy)	10
LDAP: étude du modèle, du protocole et des éléments liés à l'infrastructure	6
Messagerie: études des protocoles principaux	4
Protocoles de transferts de fichiers et d'accès à distance: études des protocoles et outils (e.g. rsync)	2

Laboratoire	32
Développement d'une application client-serveur	10
HTTP: Développement d'un client et/ou d'un serveur simple	4
HTTP: Conception et implémentation d'une infrastructure avec un reverse proxy et plusieurs serveurs	6
LDAP: Mise en oeuvre d'un serveur, conception d'un schéma et import de données	6
Messagerie: implémentation d'un client simple	4
Mise en oeuvre des outils (e.g. rsync)	2

Evaluation

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

Contrôle de connaissances

Ce champ est facultatif

cours: l'acquisition des matières de cet enseignement sera contrôlée au fur et à mesure par des tests et des travaux personnels tout au long de son déroulement. il y aura au moins 2 tests d'une durée totale de 2 périodes.

laboratoire: ils seront évalués sur la base des rapports de manipulation, à 3 reprises au minimum

Note finale

Cours	25%
Laboratoire	25%
Examen	50%

- **There will be 1 "standard" written test.**
- **The second "written test" will be the result of multiple challenges:**
  - On a regular basis, I will ask you to **complete short assignments** (quizz, small exercise, single development question, etc.). **Often, the assignment will need to be completed during the class or the lab session!**
  - Often, you will either get **'1'** or **'0' point** for completing the assignment or not. In exceptional cases, I might give a bonus with '2' points. There might be assignments with more weight, hence more points.
  - You will have the opportunity to get a total of **6 possible points**. At the end of the semester, I will compute the grades for the class with this formula: `min(6.0, number of points obtained + 1)`.
  - **Pay attention to the "RES 2018 - Annonces" Telegram channel.**
- **At the end of the semester, we will do a BIG lab (http infrastructure). It will have a heavy weight in the lab grade.**

Cyberlearn

Hes-so-CENTRE E-LEARNING

https://cyberlearn.hes-so.ch/course/view.php?id=**11307**




https://goo.gl/forms/GD6QKLcd48i3n10u2

10 minutes

Tools

Environment

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- For the labs, we will use tools that run across various platforms (Mac OS, linux, Windows). A Unix environment will sometimes make things a bit easier.
- You have the choice to...
  - Use a native Unix environment (MacOS, linux)
  - Use a Windows environment (you will sometimes have to do extra configuration and we you will need to search for the information!)
  - ~~Use a virtual machine~~
- ~~We have prepared a virtual machine, that you can use with Vagrant~~
  - ~~https://github.com/SoftEng-HEIGVD/Teaching-HEIGVD-RES-2016-LabBox~~

Git & GitHub

HAUTE ÉCOLE D'INGÉNIERIE ET DE GESTION DU CANTON DE VAUD
www.heig-vd.ch

- **Step 1: install Git**
  - http://git-scm.com/downloads
  - http://git-scm.com/book/en/Getting-Started-Installing-Git
    - Check point: are you able to invoke the git command from the shell?
- **Step 2: configure Git**
- https://help.github.com/articles/set-up-git
- **Step 3: configure SSH**
  - http://guides.beanstalkapp.com/version-control/git-on-windows.html#installing-ssh-keys

Using Git locally

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

```
$ mkdir my-project
$ cd my-project
$ git init
$ ls -al
```

- **You do not *have to* use a server**: Git is already useful to manage versions of your files on your local machine.
- The **`git init` command** creates a **local repository**. If you look carefully, you will see **a hidden `.git` directory**, where Git keeps all of his data.
- **Important**: your **`my-project`** directory is your **working directory**. If you simply create files in it, they will not immediately be part of your repository!

Using Git locally

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

```
$ echo "text a" > a.txt
$ git status
$ git add a.txt
$ git commit -m “First version of a.txt”
$ echo “my mod on text a” > a.txt
$ git status
```

- A **commit** is a **snapshot** of your repository. Git maintains a **graph of commits** and you can always **recover the state** of a particular commit.
- When **you have modified files in your working directory**, you need to specify which ones should be **part of the next commit**.
- You use the **`git add`** command to add a file to the so-called **staging area**. It will be part of the next commit.
- You use the **`git status`** command to **check the content** of your working directory and of your staging area.

More info: http://git-scm.com/book/en/Git-Basics-Recording-Changes-to-the-Repository

Using Git locally

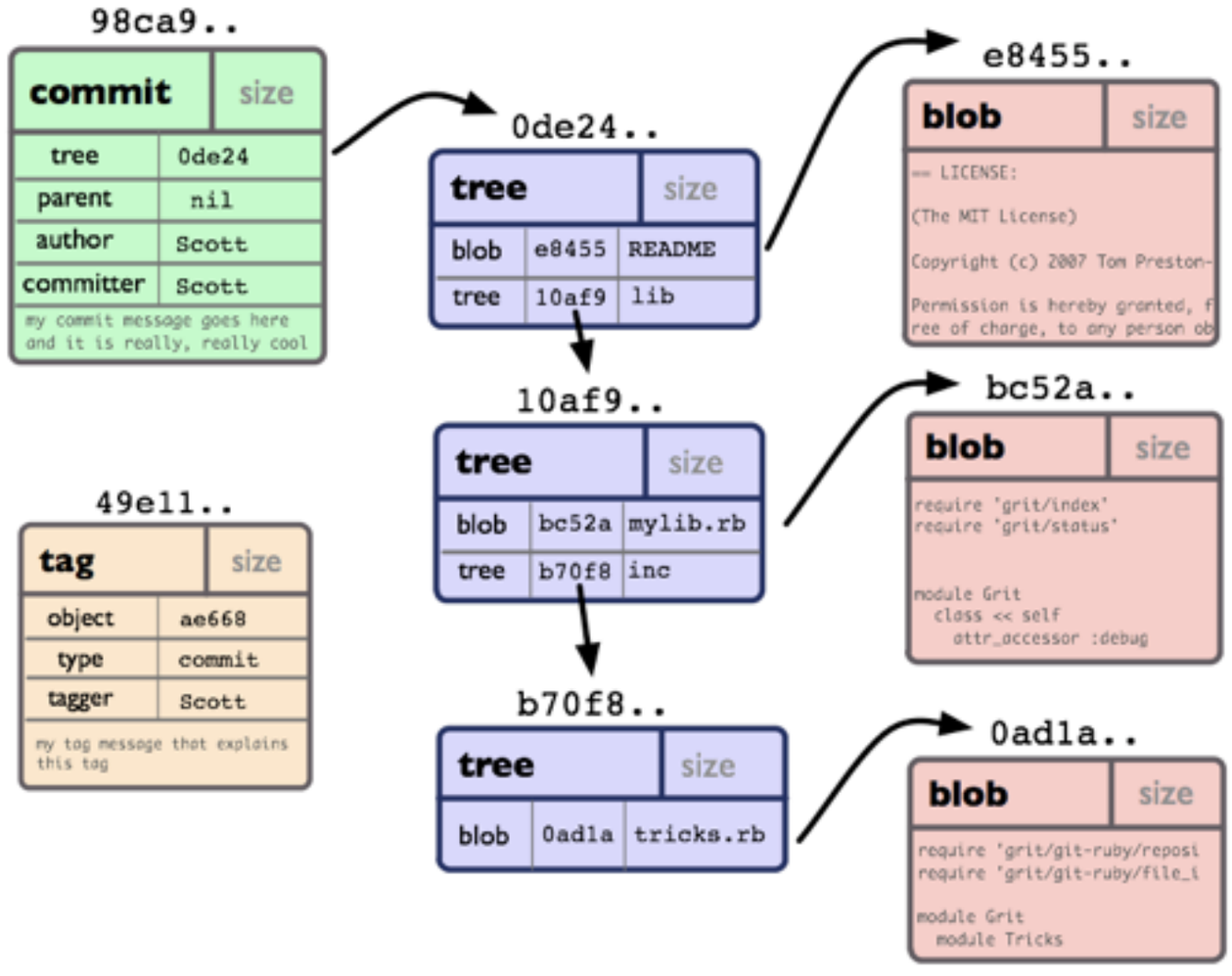
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **Git is a "content-addressable file system"**
- **Git uses a key-value store:**
  - When you store a file in a repo, git computes a SHA-1 hash of its content.
  - The hash is used as a key to index the file in the store.
  - For this reason, two files with exactly the same content are stored only once in the git repository.
  - **Go in the .git hidden directory, have a look at the ./objects directory and you will find this key-value store.**

The Git Object Model

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

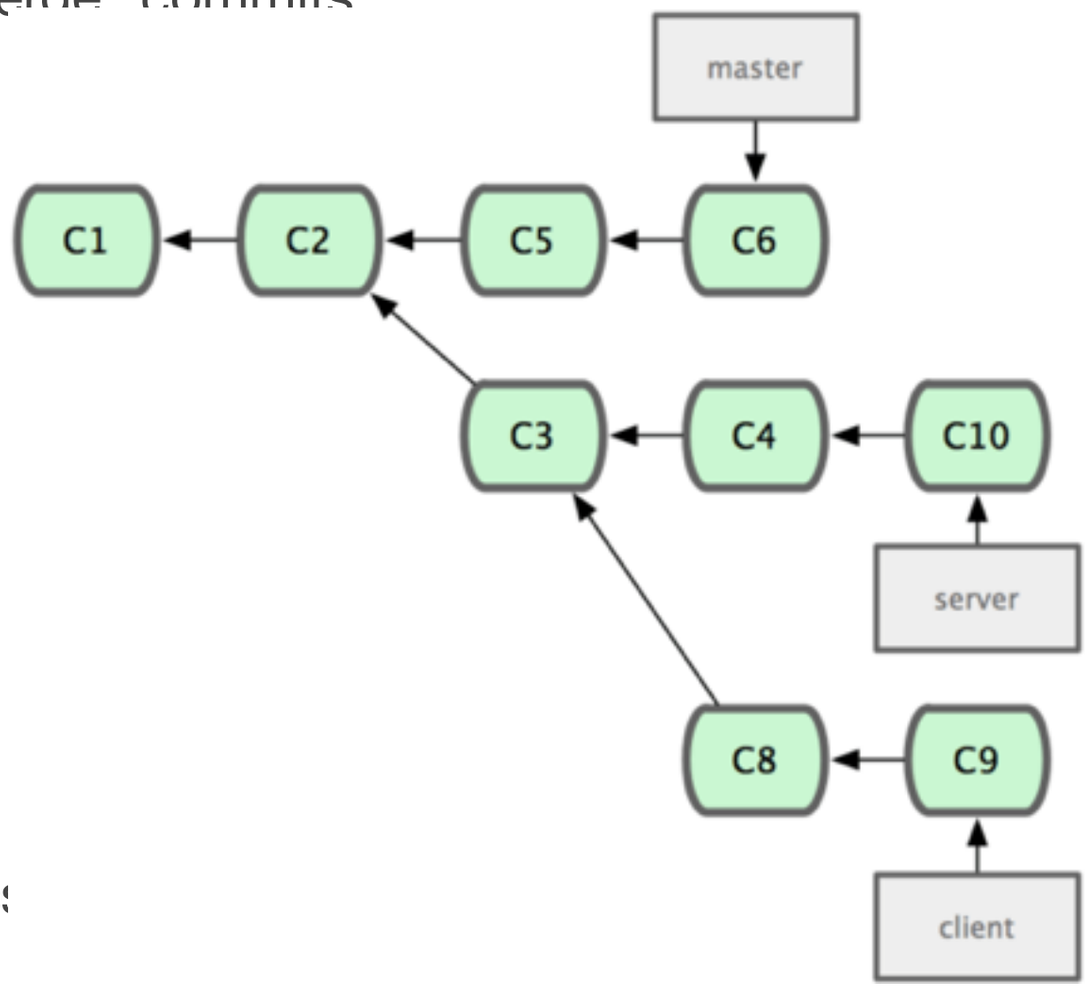
- **BLOBs** store **files**.
- **Trees** store **directory structures** in the file system. A tree has pointers to subtrees and BLOBs.
- **Commits** identify snapshots in the history. **Every commit points to a version of the top-level tree**. A commit also stores metadata: author, parent, message, etc.
- **Annotated Tags** (created with git tag -a) are used to mark releases.
- Recommendation: do not use lightweight tags (created with git tag, without -a)



The Git history is a DAG structure

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

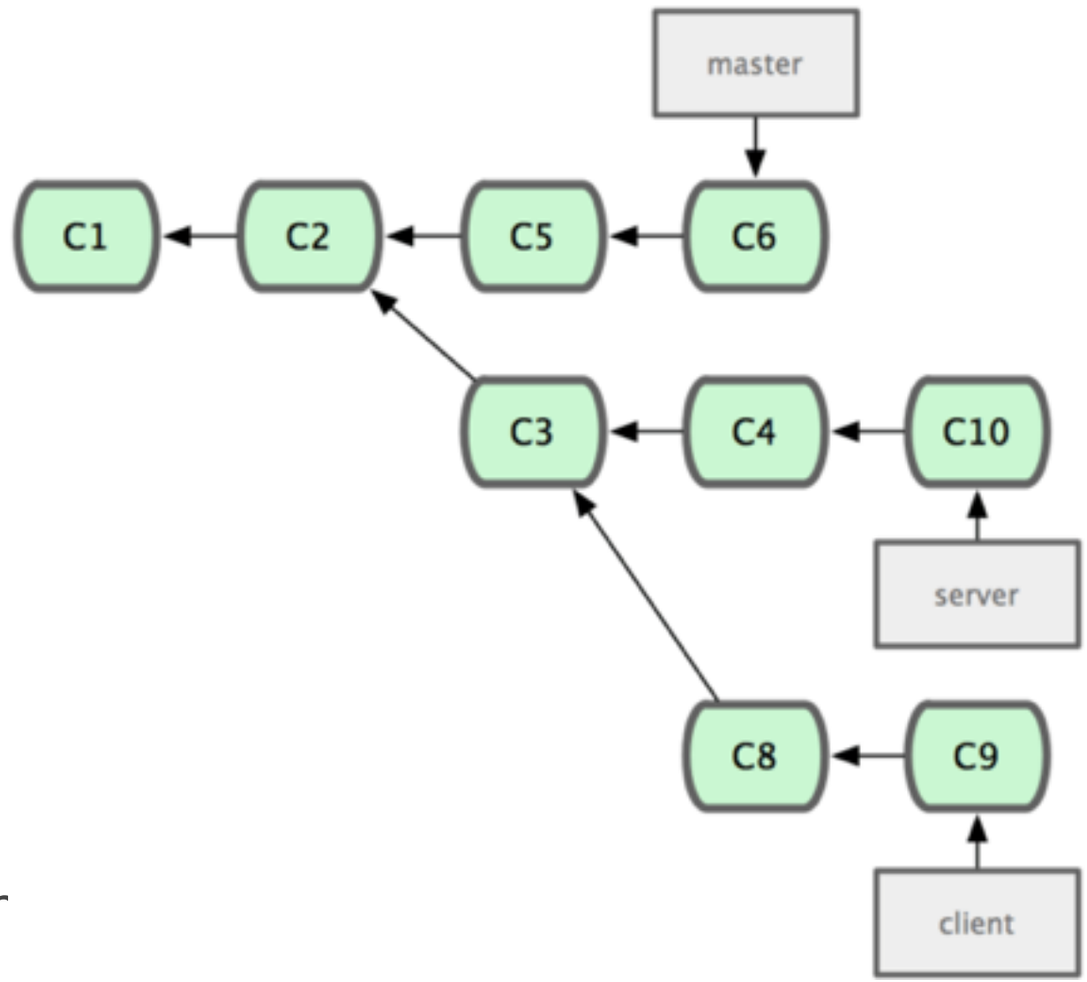
- **A DAG is a Directed Acyclic Graph**
  - Except for the initial commit, every commit has a least one parent.
  - Commits with more than one parent are "merge" commits
- **In this example:**
  - C1 is the initial commit
  - There are no merge commits
  - master, server and client are branches
  - The branches have never been merged
  - The last commit on the client branch is C9
- All branches have C1 and C2 as common ance

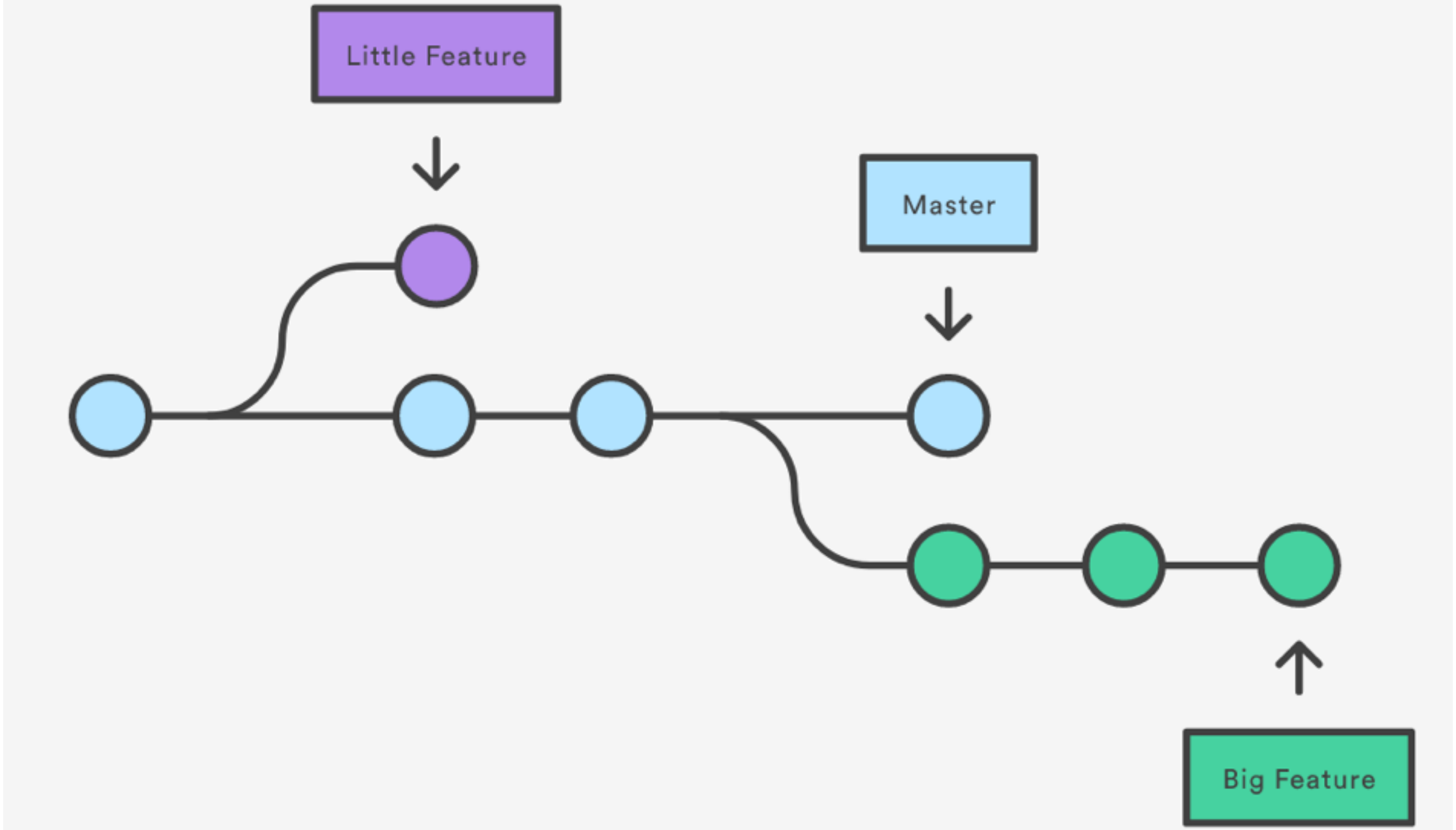


Git branches are only pointers

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- Branches are stored in files located in the **.git/refs/heads** directory.
- **Each branch is stored in a file, which only contains the hash of a commit.**
- **What is HEAD?**
  - HEAD means "the tip of the current branch" (last commit of the current branch)
  - When you checkout a branch, HEAD is the name of this branch.
- **What is HEAD^n?**
  - It is a reference to the nth parent of HEAD
  - HEAD^1 means the parent of the last comm
  - HEAD^2 means its grand-parent

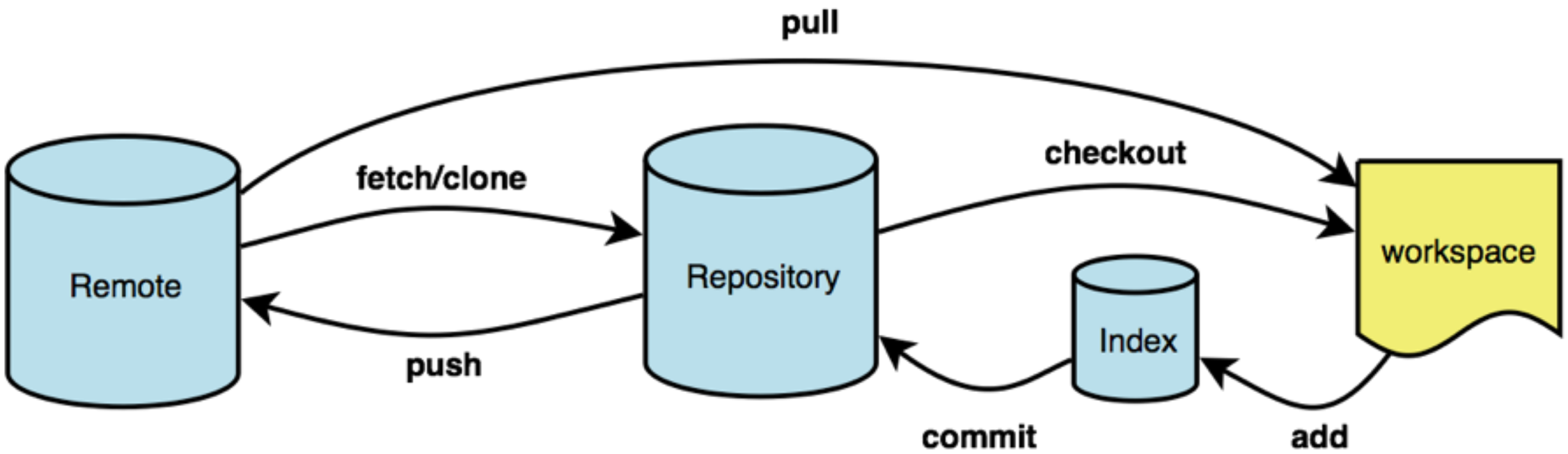




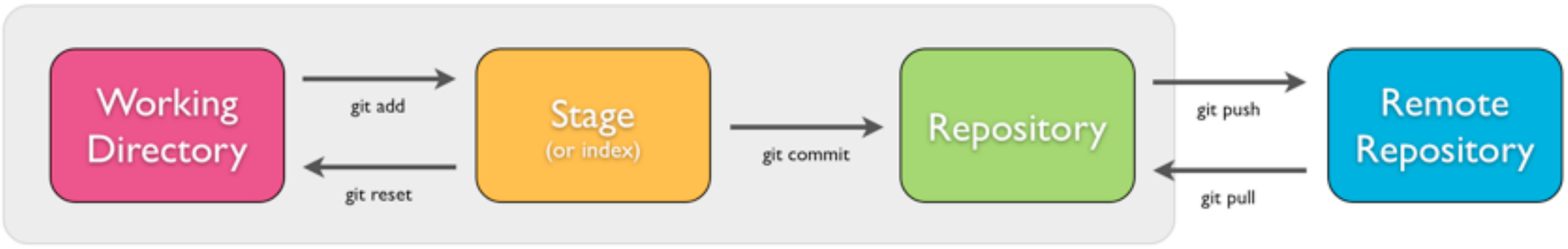
https://www.atlassian.com/git/tutorials/using-branches

Git & Remote Repositories

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



Source: http://illustrated-git.readthedocs.org/en/latest/

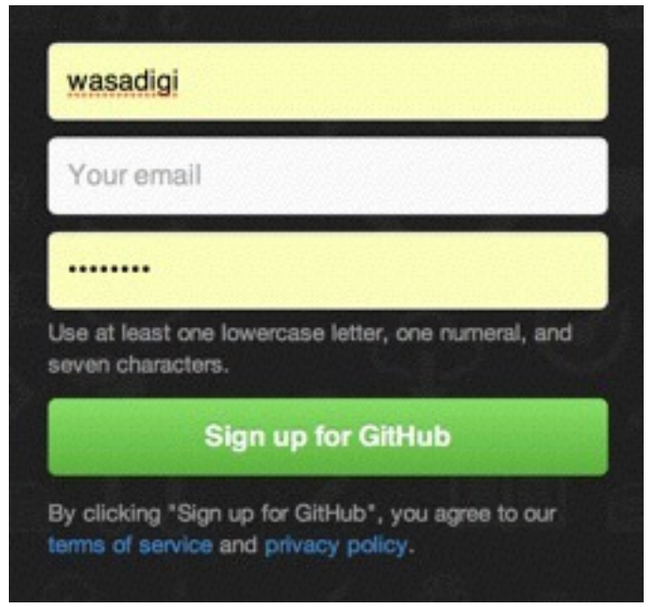


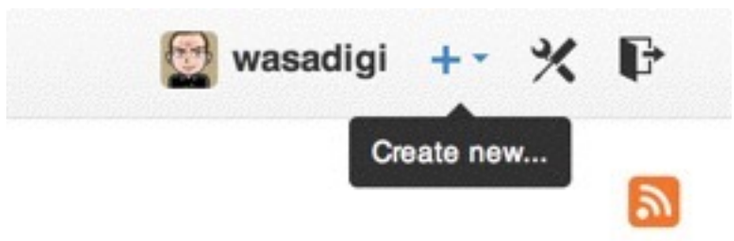
Source: http://bramus.github.io/ws2-sws-course-materials/xx.git.html#/4/1

GitHub Setup

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **Sign up** for GitHub and get your own account:
  - Go to http://www.github.com
- Add your **SSH key**:
  - Go to your accounts settings. You will find an option to manage your SSH keys.
  - If you don't have a SSH key yet, follow the instructions in the online help.
  - If you are using windows, you will need to use Git BASH.
- **Create** your first repo, hosted on Github.
- **Copy the SSH URL** of the repo.

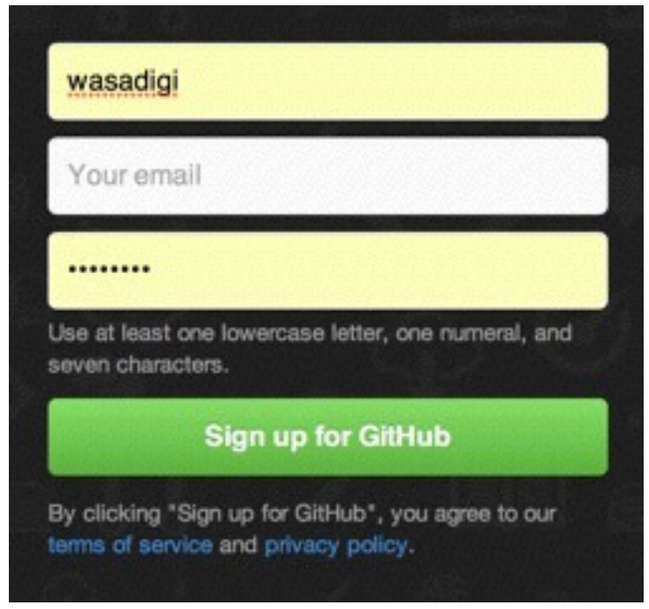


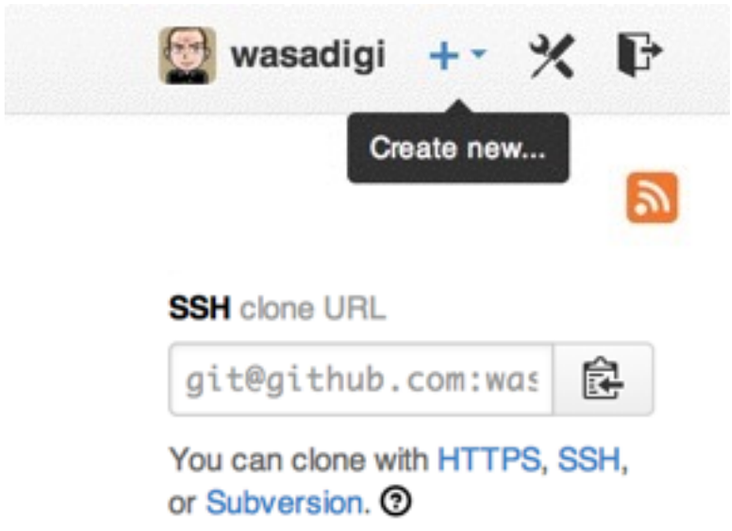


Validate your setup

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **Clone your repo to your laptop**
  - Open a **terminal window** (Terminal on Mac OS, Git BASH on Windows, etc.)
  - Create a **new directory** to host your clone of the repo and get into it.
  - **Clone** the repo, using the **SSH URL**.
  - **Create** a file, **add** it to the staging area, **commit** the changes and finally **push** the commit Github.



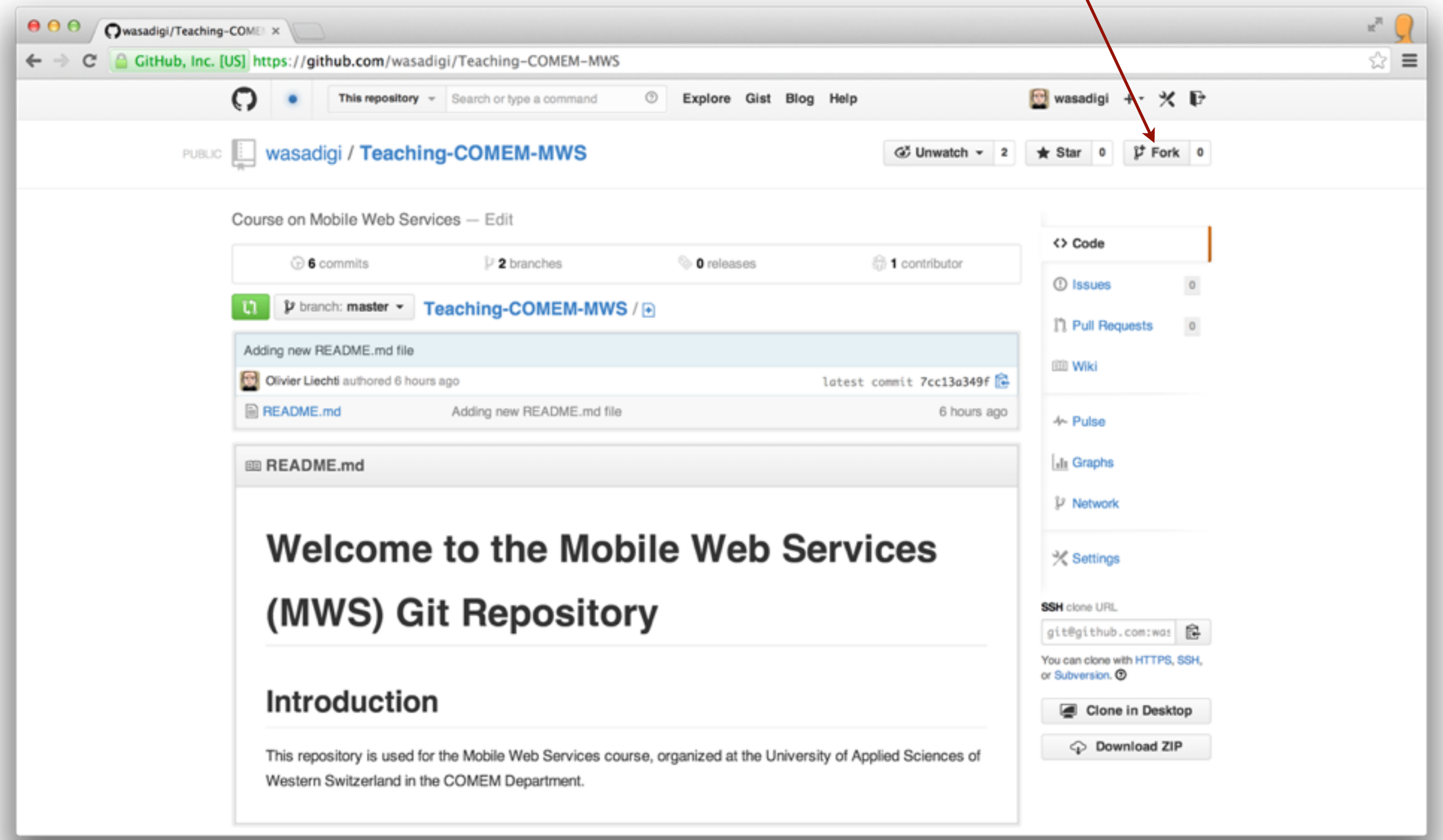



```
$ mkdir myspace
$ cd myspace
$ git clone git@github.com:UUUUU/RRRRR.git
$ git touch firstFile.txt
$ git add firstFile.txt
$ git commit -m “I have added my first file”
$ git push
```

Forks & Clones

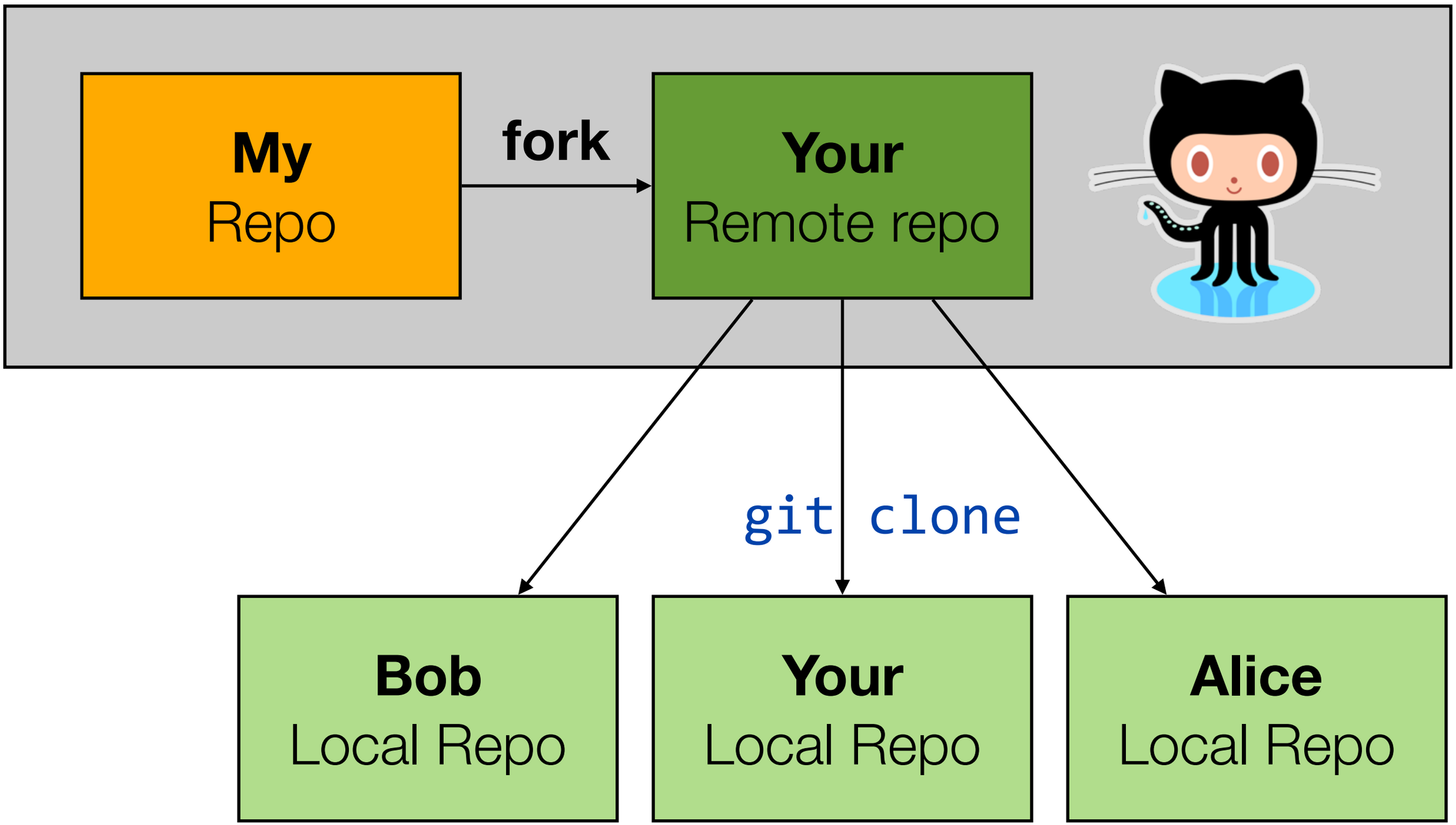
If you do this, you will have YOUR clone of MY repo hosted on Github

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



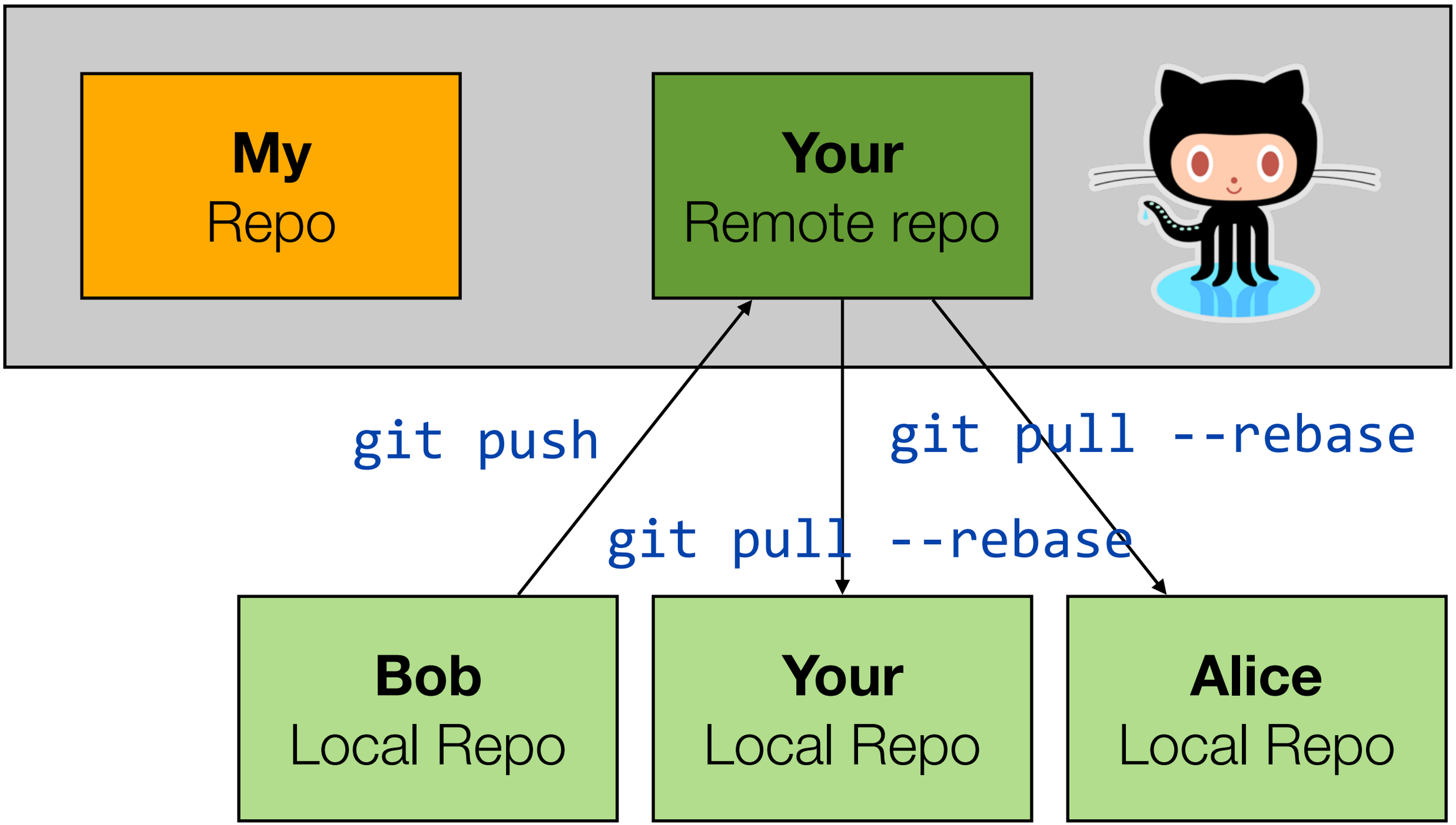
Forking My Repo on GitHub

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



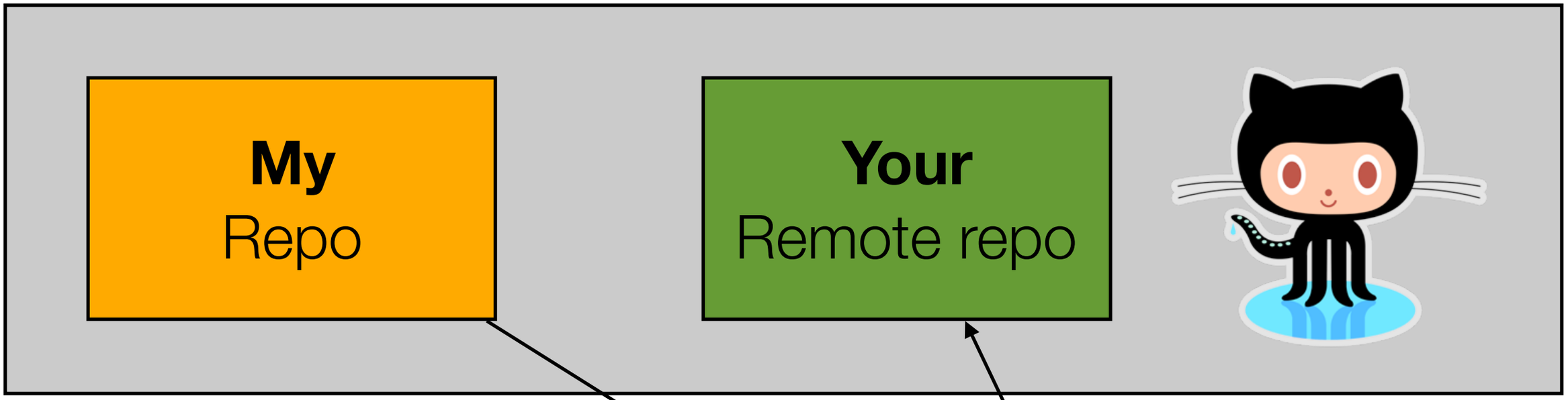
Forking My Repo on GitHub

HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch



How Do Will You Get **My** Updates?

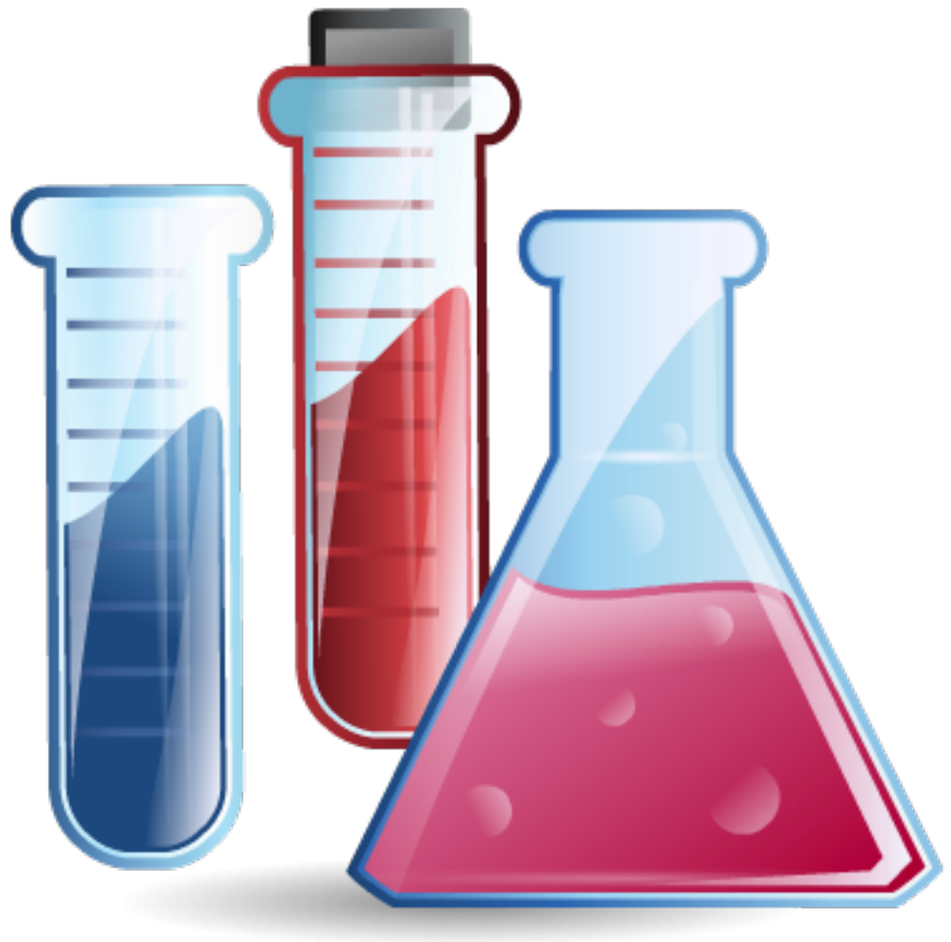
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch




```
# configuration, do it once
git remote add upstream git://github.com/UU/RR.git
git remote -v
```

```
git fetch upstream
git rebase upstream/master
```

Lab Introduction



Important points

heig-vd
HAUTE ÉCOLE
D'INGÉNIERIE ET DE GESTION
DU CANTON DE VAUD
www.heig-vd.ch

- **GitHub workflow** (start by forking, not cloning the source repo)
- ~~Understand the Lab Box setup and how you can access project files both from your host and from the VM~~
- What is **maven**?
- How do we use **unit tests**?
- **Where to write your code** (NOT in the test package!!!)

Short-term Planning

Week 1 19-23.02.2018	Sunday **25.02.2018 08:00**	Week 2 26-02.02.2018	Tuesday **27.02.2018 15:00**
Course Intro & guidelines		**Course** Java IOs	
Lab Study and work	**Intermediate check** I will let you know about current status	**Lab** Preparation next lab	**Final deadline**
	This is your safety check. Meet this deadline and you are sure to get the challenge point.		*I will do the final check here.* *Students who have all tests in green get 1 pt for the first challenge.* *Others get 0 pt.*