

Welcome to RES 2016

RES, Lecture 00

Olivier Liechti
Simon Oulevay
Laurent Prévost

heig-vd
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

Agenda

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

Background



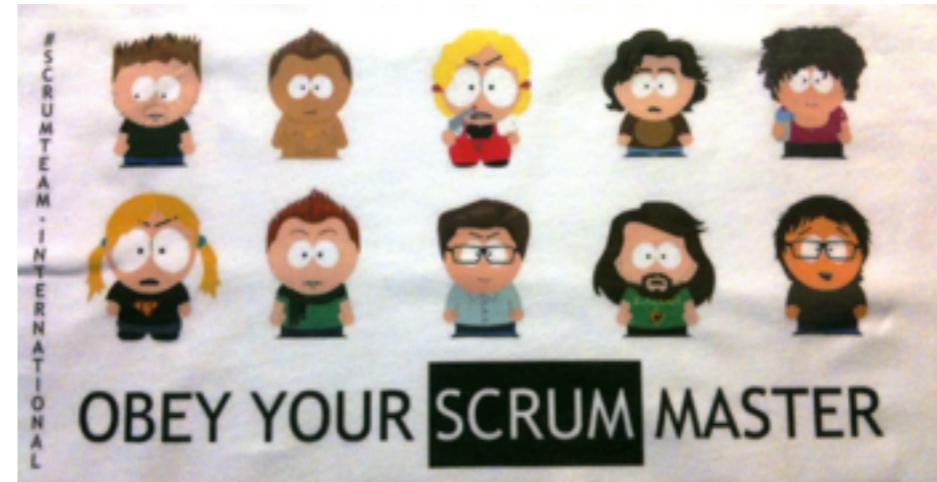
Personal background

heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud



Personal background

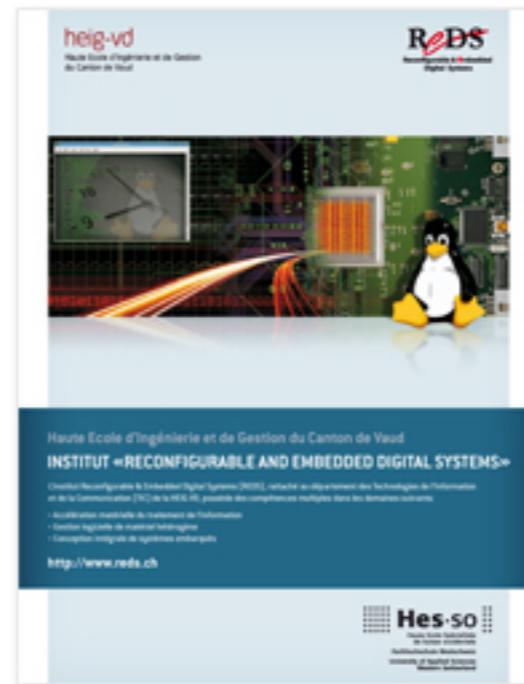


R&D at HEIG-VD



heig-vd

Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

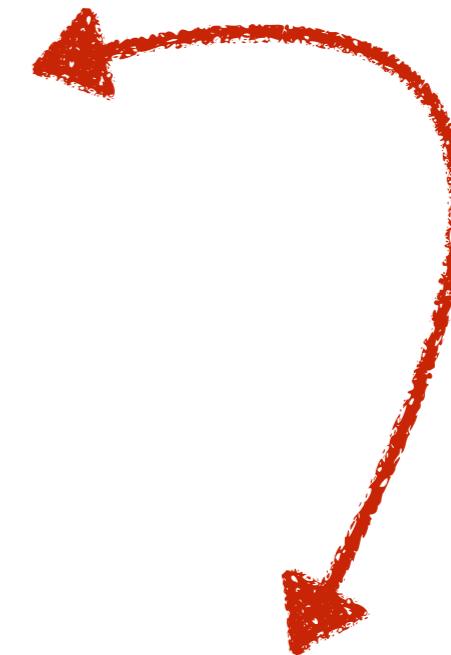
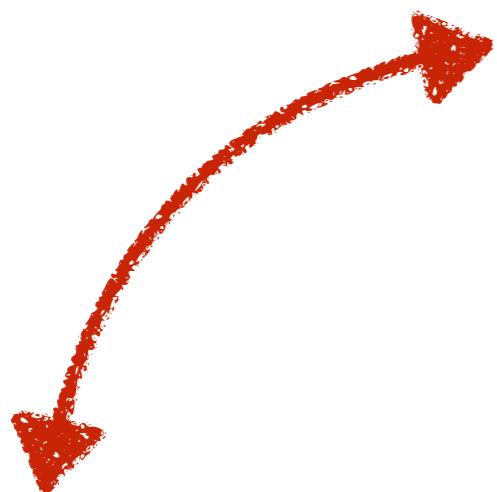


Startups



PROBE DOCK

Why do we do applied research?



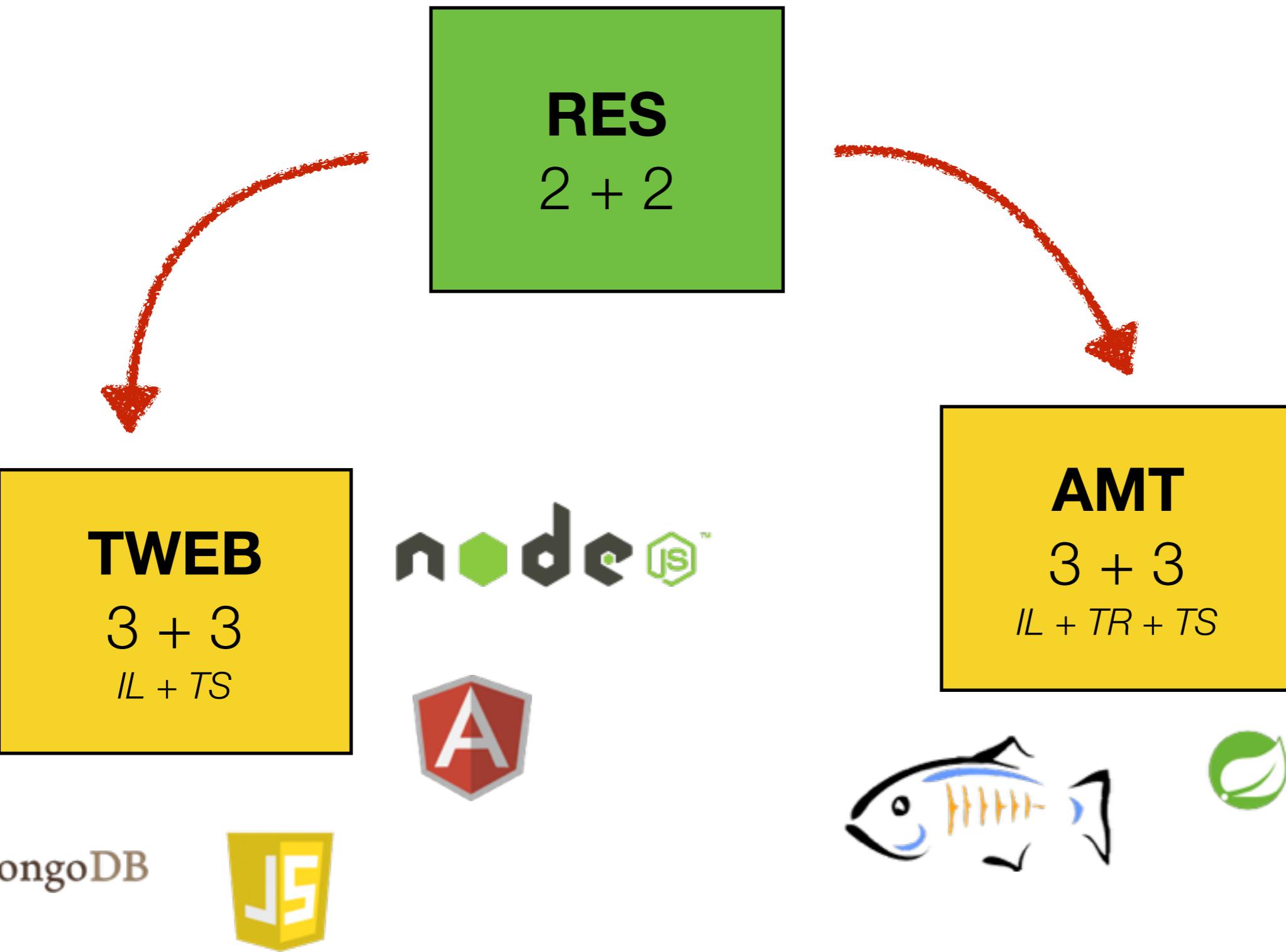
There are plenty of opportunities!



**I WANT YOU
TO LIVE YOUR BEST LIFE!**

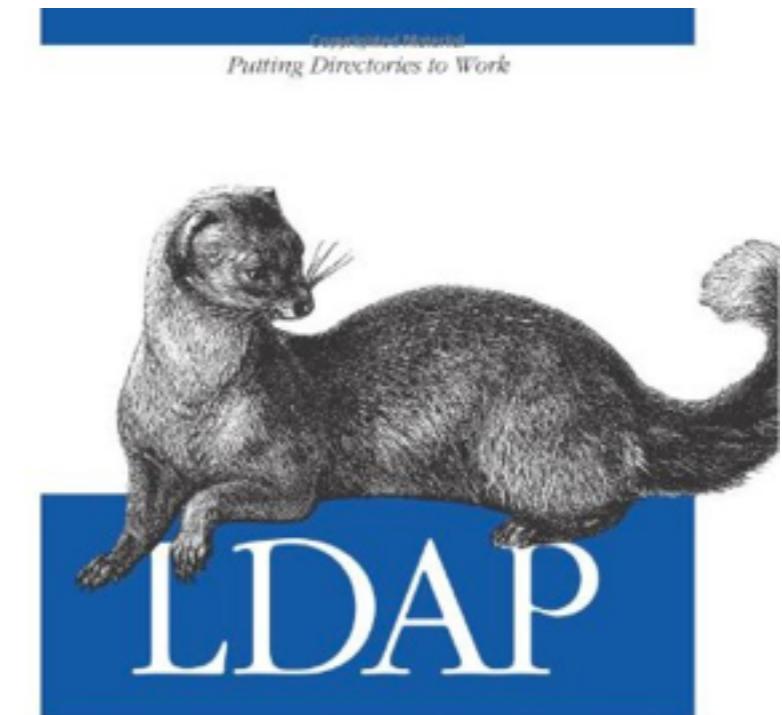
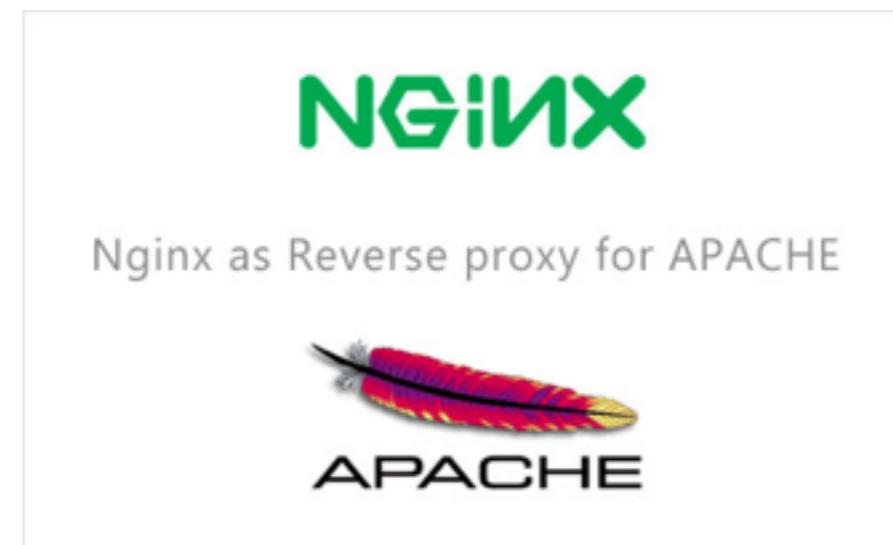
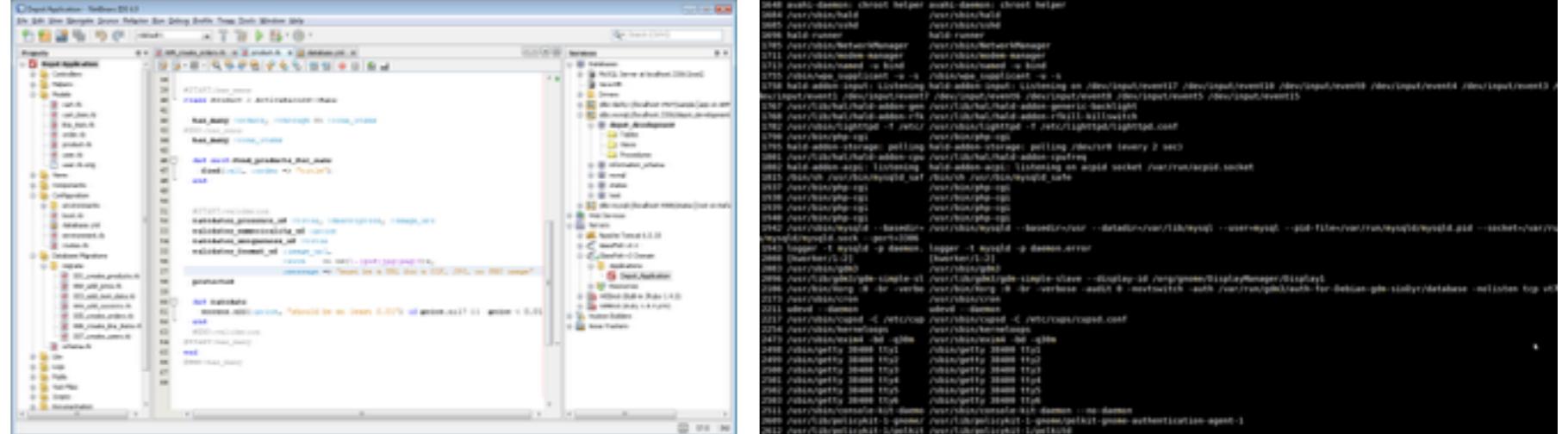
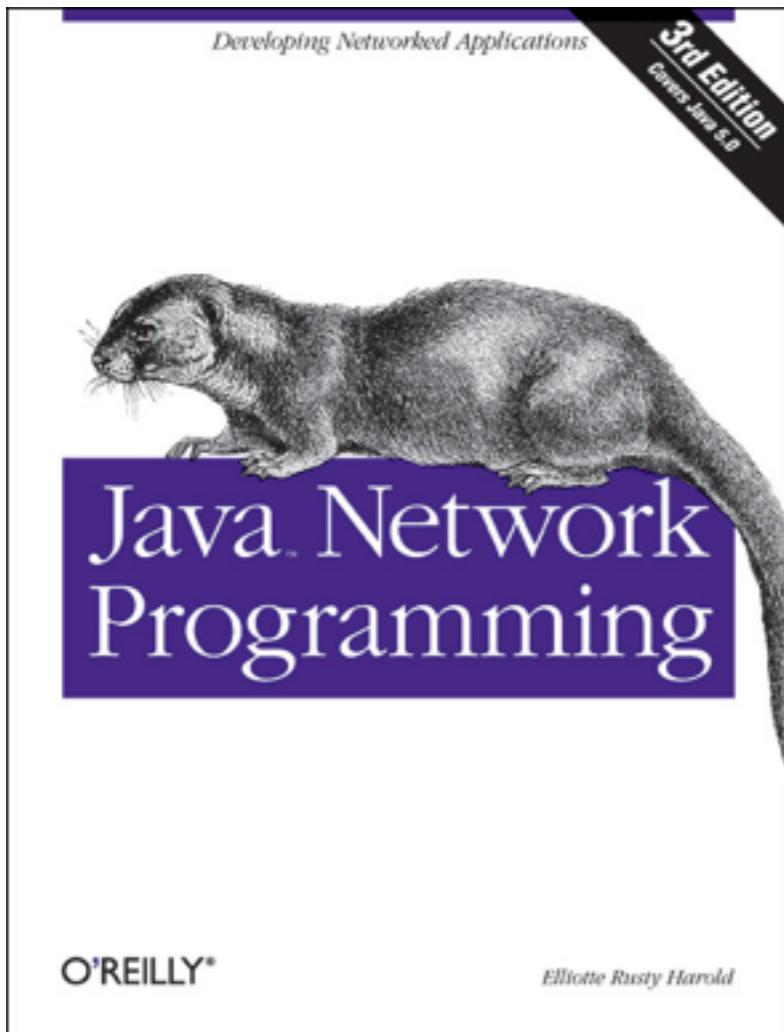
Course Objectives





 mongoDB





High-level planning

- **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 **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

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

Cours	32
Concepts de programmation réseau et présentation 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

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.

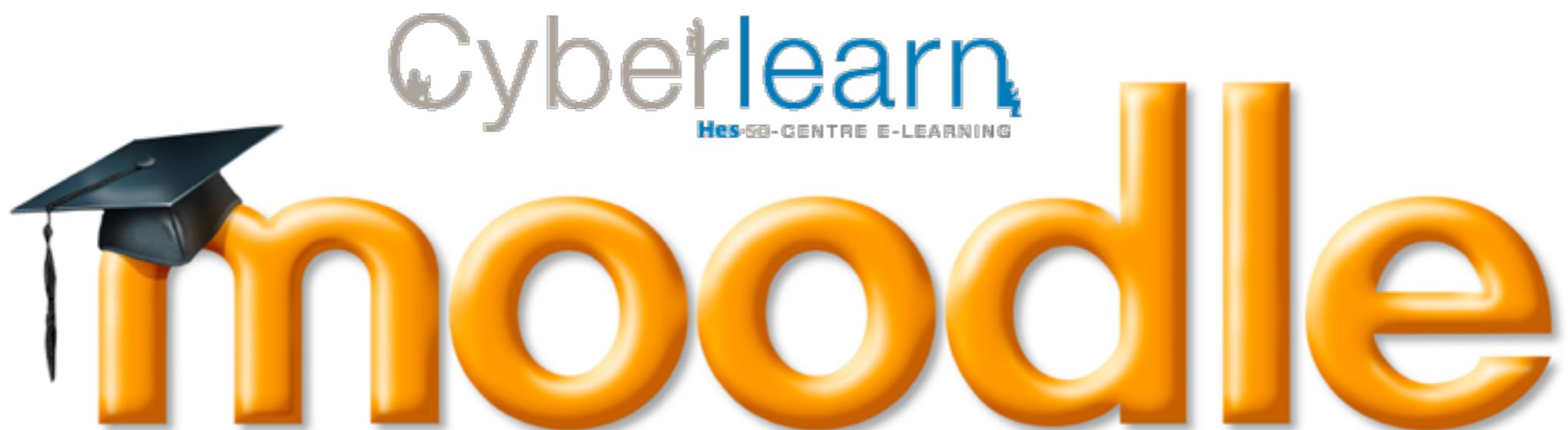
laboratoire: ils seront évalués sur la base des rapports de manipulation, à 3 reprises au minimum

Note finale

Cours	25%
Laboratoire	25%
Examen	50%

- **Continuous evaluation for “cours”, in addition to the written tests:**

- 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.
- Whenever I have given enough assignments to have a total of **8 possible points**, I will compute the grades for the class with this formula: **$\min(6.0, (\text{number of points} / 7 * 5) + 1)$** .
- **Read your mails carefully!!!**



[http://cyberlearn.hes-so.ch/
course/view.php?id=8206](http://cyberlearn.hes-so.ch/course/view.php?id=8206)

SEMAINE 1 (22 FÉVRIER)

Le premier objectif de la première semaine est de présenter les objectifs du cours, d'expliquer comment il prépare à deux cours conséquents de 3ème année (AMT et TWEB) et de faire un survol rapide de la matière.

Le deuxième objectif est de nous familiariser avec les outils et les processus que nous allons utiliser dans beaucoup de labos. Nous allons nous appuyer sur git et GitHub et sur des outils de génie logiciel. Il est donc important de comprendre à quoi servent ces outils et comment nous allons les utiliser.



Connaissances actuelles et attentes

Ce questionnaire va nous aider à mieux comprendre l'état de vos connaissances actuelles et d'identifier les attentes particulières que vous avez par rapport au cours.

15 minutes

Tools



Git & Github

- **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

```
$ 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

```
$ 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.

Working Dir, Staging Area & Repository

This is a local directory, not
a remote server!

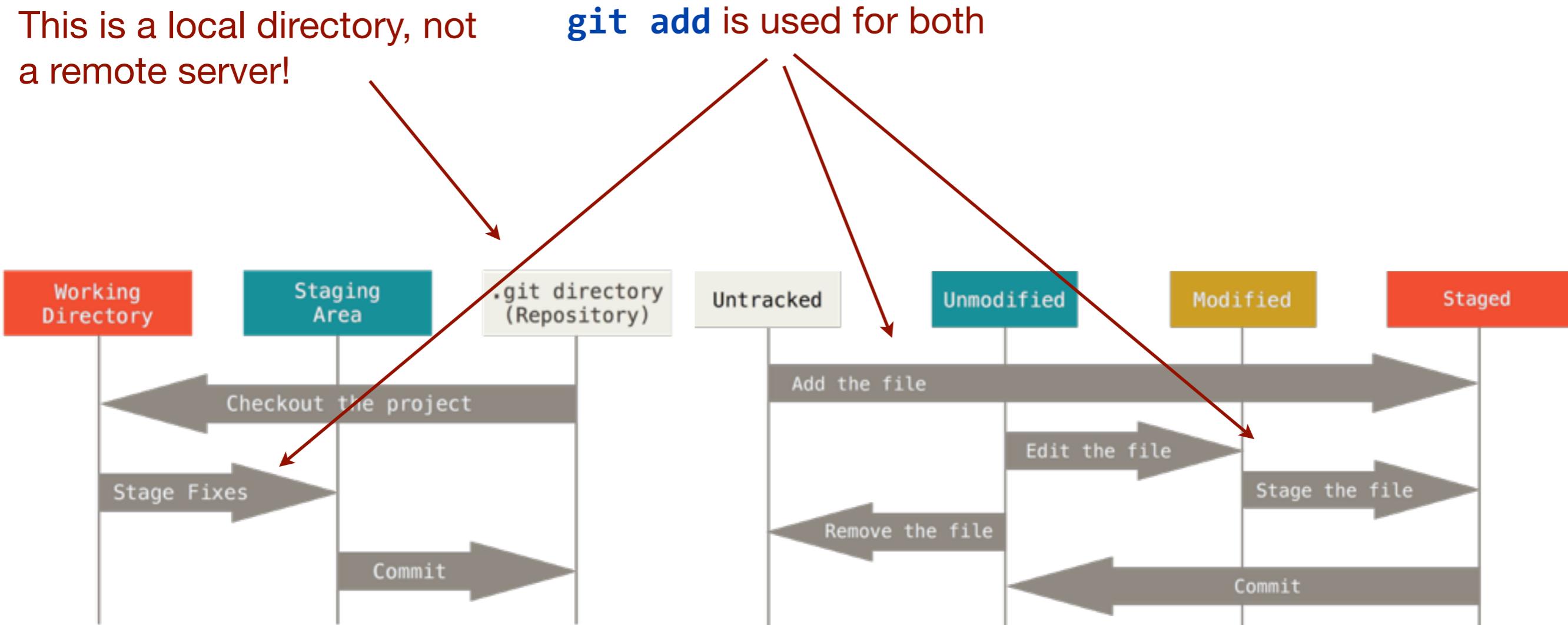
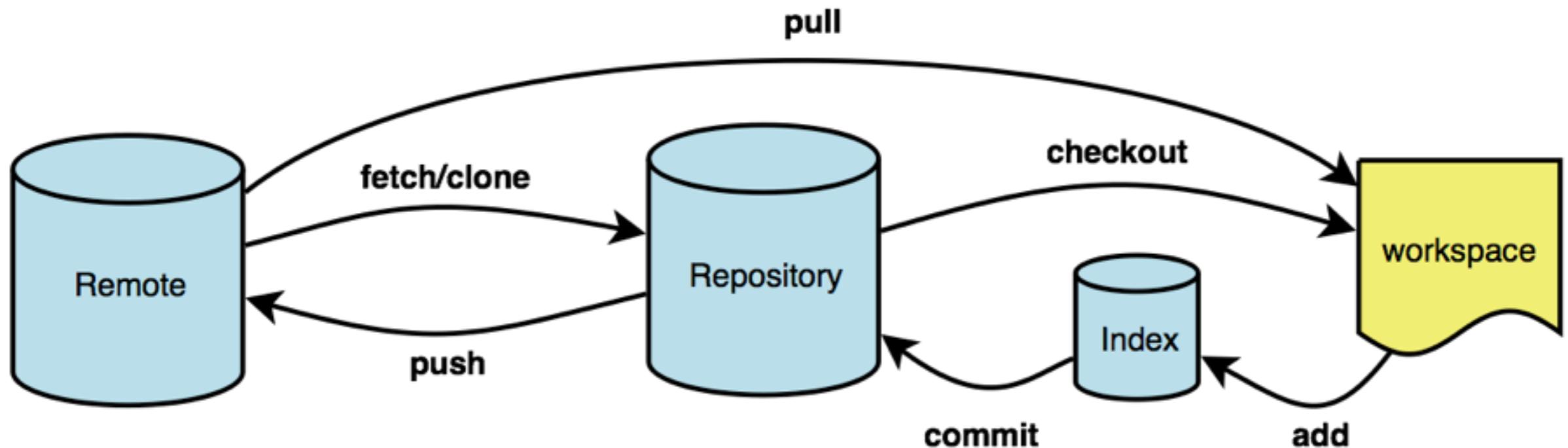


Figure 1-6. Working directory, staging area, and git directory.

Figure 2-1. The lifecycle of the status of your files.

Git & Remote Repositories



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



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

GitHub Setup

- **Sign up for GitHub and get your own account:**

- Go to <http://www.github.com>

- Add your **SSH key**:

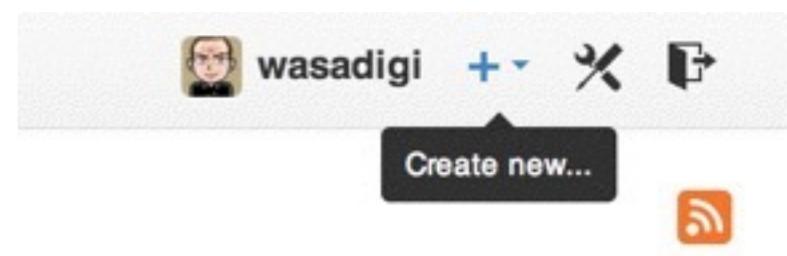
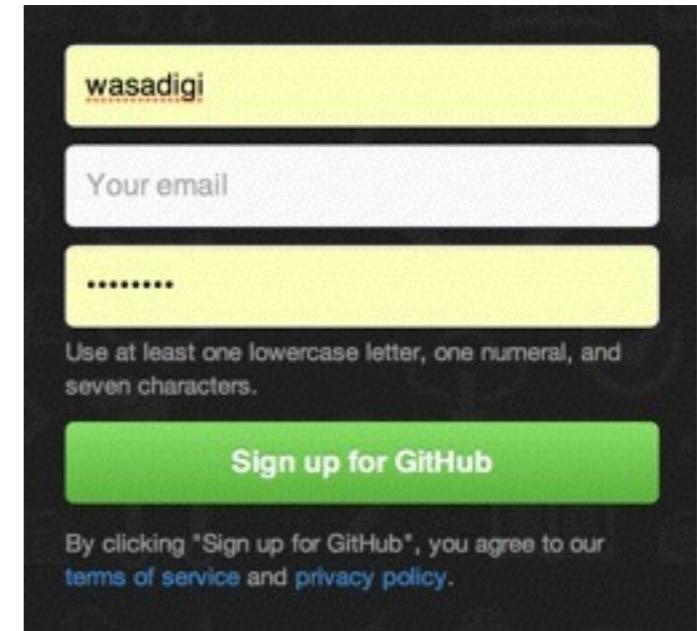
- Go to your accounts settings. You will find an option to manage your SSH keys.

- 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.

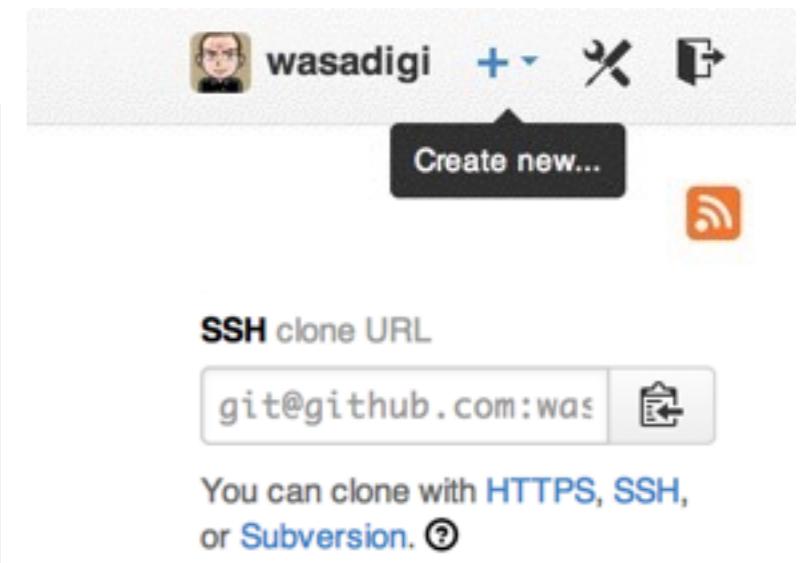


Validate your setup

- **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
```



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

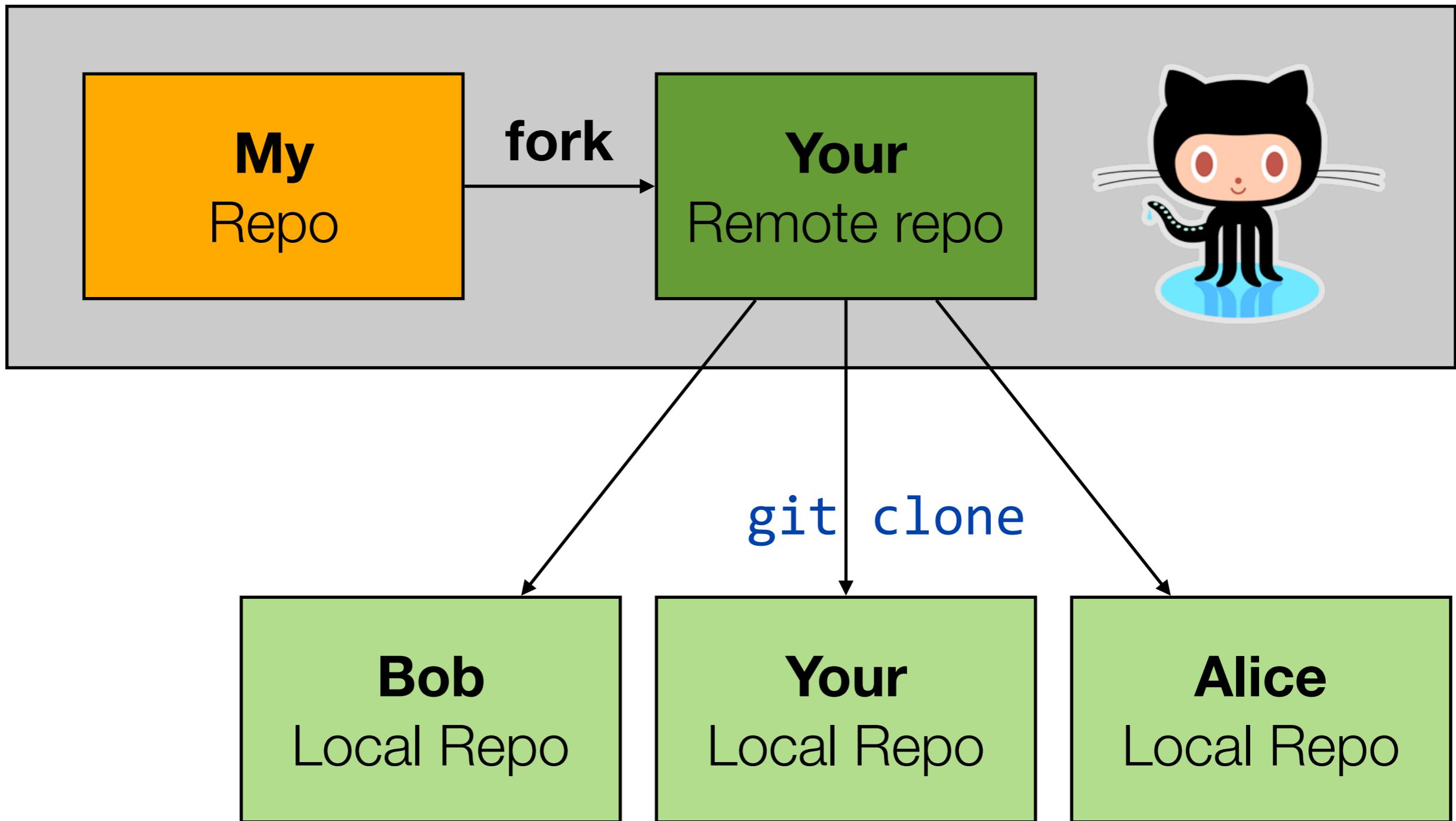
Forks & Clones

heig-vd

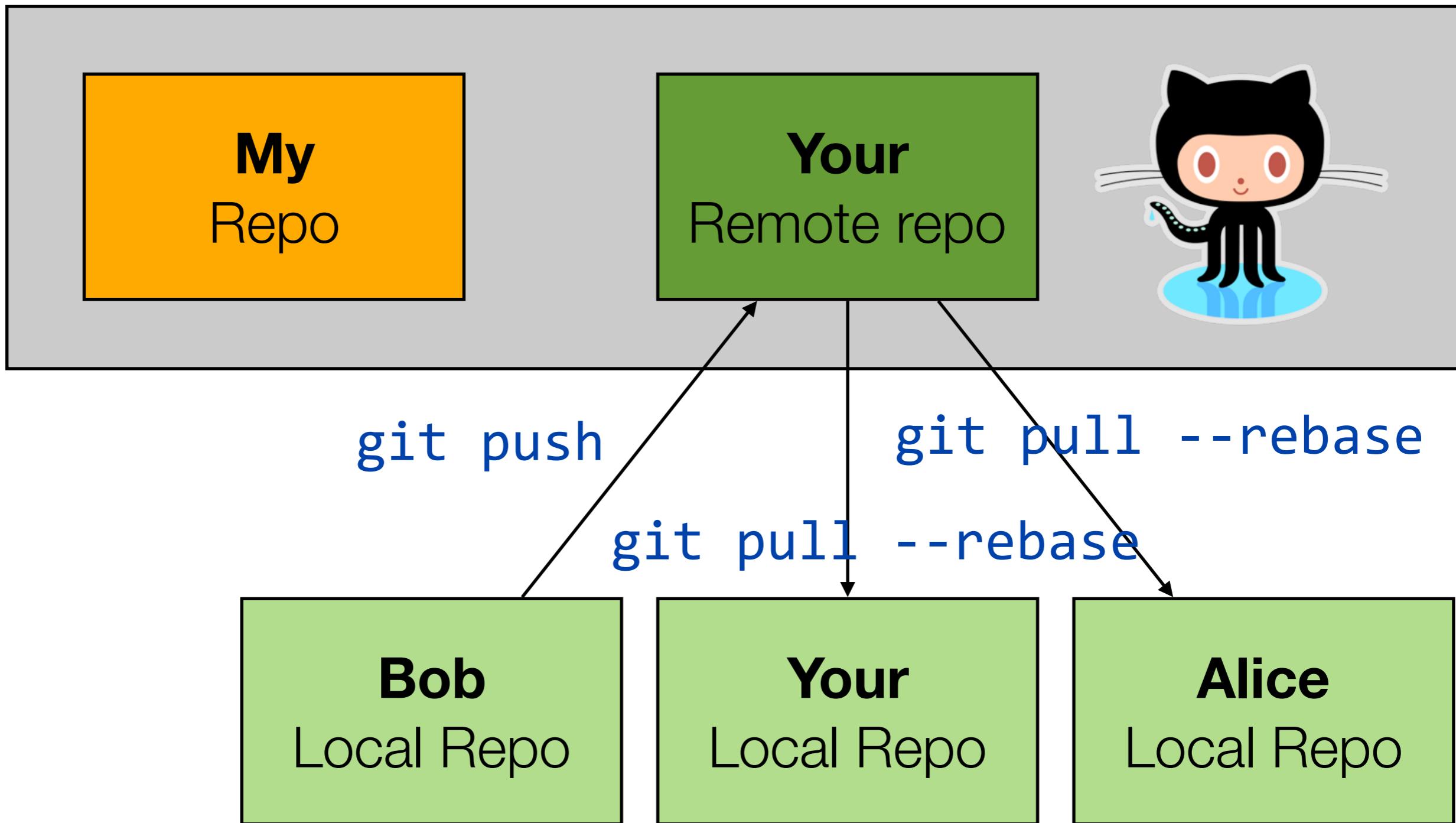
Haute Ecole d'Ingénierie et de Gestion
du Canton de Vaud

A screenshot of a GitHub repository page for 'wasadigi/Teaching-COMEM-MWS'. The page shows basic repository statistics: 6 commits, 2 branches, 0 releases, and 1 contributor. The 'branch: master' dropdown is set to 'Teaching-COMEM-MWS'. A commit history is listed, with the most recent commit by Olivier Liechti 6 hours ago. The main content area features a large heading: 'Welcome to the Mobile Web Services (MWS) Git Repository'. Below this, a section titled 'Introduction' contains the text: 'This repository is used for the Mobile Web Services course, organized at the University of Applied Sciences of Western Switzerland in the COMEM Department.' On the right side of the page, there is a sidebar with links for Code, Issues, Pull Requests, Wiki, Pulse, Graphs, Network, and Settings. At the bottom right, there are buttons for SSH clone URL, Clone in Desktop, and Download ZIP. A red arrow points from the top text 'If you do this, you will have YOUR clone of MY repo hosted on Github' down to the 'Fork' button in the top right corner of the GitHub interface.

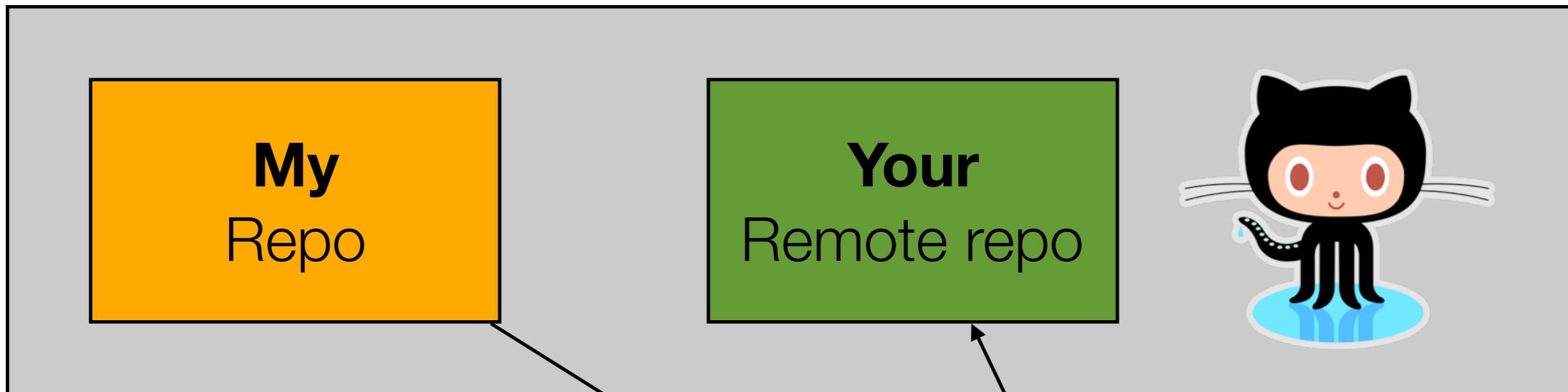
Forking My Repo on GitHub



Forking My Repo on GitHub



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



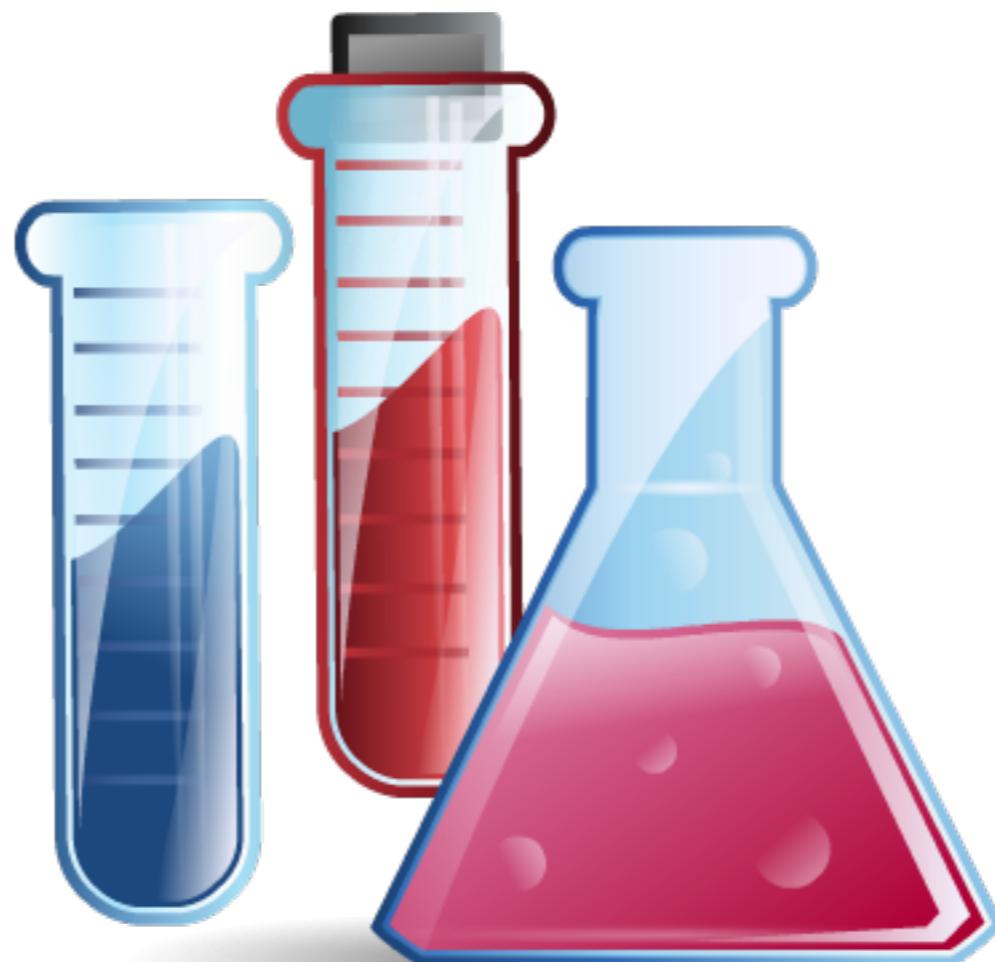
```
# configuration, do it once  
git remote add upstream git://github.com/UU/RR.git  
git remote -v
```

git fetch upstream
git rebase upstream/master

git push



Lab Introduction



Important points

- **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!!!)
- How to **check** your submission?

Short-term Planning

Wed 24.02.2016	Sat 27.02.2016 23:00	Wed 02.03.2016	Fri 04.03.2016 16:00
Course Intro & guidelines		Course Behind the scenes + next topic (IOs)	
Lab Study and work	Intermediate deadline	Lab Catch-up and coach	Final deadline
	<i>If all your tests are green, you get a 6.0</i>		<i>Your max grade is 5.0. (-0.3 for every red test)</i>