

# **Introduction to High Performance Computing**

**Autumn, 2017**

**Lecture 2**

# Announcements

---

- Lecture recordings at <https://imperial.cloud.panopto.eu>  
Will be available by end of day (search for 'M3C')
- Webpage: [imperialhpsc.bitbucket.io](https://imperialhpsc.bitbucket.io)
- Python:
  - Work through material covered in 1<sup>st</sup> two video lectures (on webpage now) before Thursday's lecture
  - 3<sup>rd</sup> and 4<sup>th</sup> lectures will be online by end of today

# Main topics today

---

- **Version control with git and bitbucket**
  - **Background on version control**
  - **Working locally on your computer**
  - **Moving material back and forth from the cloud**

# Software version control

---

- Originally used for large projects with many developers
- Now a standard tool in software engineering
- Slowly becoming a standard tool in scientific computing

# Software version control

---

- Originally used for large projects with many developers
- Now a standard tool in software engineering
- Slowly becoming a standard tool in scientific computing
- Useful for:
  - Collaboratively developing software
  - Keeping track of changes to code
  - Managing different versions of code in an organized manner
- The more complicated the problem, the more important it is to use version control!

# Local version control

---

- **SVN**
  - Was standard tool 5-10 years ago, now losing popularity
  - Uses client-server model
  - Master version and history stored centrally on server
  - What happens if server is down?
- **Git**
  - Rapidly gaining popularity
  - Uses distributed model
  - Software history stored locally in *.git* subdirectory

# Local projects

---

**Good programming practice:**

- 1. Make a plan/outline**
- 2. Take notes and add comments**
- 3. Careful testing and validation**

# Local projects

---

**Good programming practice:**

- 1. Make a plan/outline**
- 2. Take notes and add comments**
- 3. Careful testing and validation**

**Git helps with 2. and 3.**



# Local projects with git

---

Where you make changes →

Working directory

Index

Repository

# Local projects with git

---

Where you make changes →

Working directory

Index

‘Archived’ version(s) →

Repository

# Local projects with git

---

Where you make changes →

Working directory

*git add*

Index

‘Archived’ version(s) →

Repository

# Local projects with git

---

Where you make changes →

Working directory

*git add*

Index

*git commit*

‘Archived’ version(s) →

Repository

# Local projects with git

---

Where you make changes →

Working directory

*git add*

Index

*git commit -m "notes"*

‘Archived’ version(s) →

Repository

# Local projects with git

---

Where you make changes →

Working directory

*git add*

Index

*git status* tells you at which stage files are placed

*git commit -m "notes"*

Repository

‘Archived’ version(s) →


## **Example: Creating a local git repository**

# Initialize

---

**1<sup>st</sup> step: make directory and initialize as a repo:**

```
$ mkdir git_example  
  
$ cd git_example  
  
$ git init  
Initialized empty Git repository in /Users/prasun/Documents/  
repos/git_example/.git/  
  
$ ls -a  
.  .. .git
```



**History of changes will be stored in hidden directory .git**



# Create file, add and commit

---

1. Open text editor and make file, *scientists.txt*
2. Check status

```
$ cat scientists.txt  
Issac Newton  
Albert Einstein
```

```
$ git status  
On branch master
```

Initial commit

Untracked files:  
(use "git add <file>..." to include in what will be committed)

**scientists.txt**

nothing added to commit but untracked files present (use "git add" to track)

# Create file, add and commit

---

**Note:** *git* tells you what to do next

```
$ cat scientists.txt
Issac Newton
Albert Einstein


$ git status
On branch master

Initial commit

Untracked files:
  (use "git add <file>..." to include in what will be committed)

    scientists.txt

nothing added to commit but untracked files present (use "git
add" to track)
```



# Create file, add and commit

---

## 3. Add file to index

```
$ git add scientists.txt
```

```
$ git status
```

```
On branch master
```

```
Initial commit
```

```
Changes to be committed:
```

```
(use "git rm --cached <file>..." to unstage)
```

```
new file:   scientists.txt
```

# Create file, add and commit

---

## 4. Commit file to repository

```
$ git commit -m "initial commit, scientists.txt added to repo"  
scientists.txt  
[master (root-commit) 6e74cfc] initial commit, scientists.txt  
added to repo  
  Committer: Prasun Ray  
  
1 file changed, 2 insertions(+)  
create mode 100644 scientists.txt
```

**Now try *git status*:**

```
$ git status  
On branch master  
nothing to commit, working directory clean
```

# Local projects with git

---

*git diff*

*git diff* compares

Working directory

Index

Repository

# Local projects with git

---

*git diff*

*git diff HEAD* compares

Working directory

Index

Repository

# Local projects with git

---

*git diff*

Working directory

Index

*git diff --cached* compares

Repository

# diff example

---

**Make change to *scientists.txt* and compare to “official” version**

```
$ cat scientists.txt
Issac Newton
Albert Einstein
Galileo Galilei
$
$ git diff HEAD
diff --git a/scientists.txt b/scientists.txt
index 9079fe8..efc1584 100644
--- a/scientists.txt
+++ b/scientists.txt
@@ -1,2 +1,3 @@
    Issac Newton
    Albert Einstein
+Galileo Galilei
```



# Visiting previous versions

---

- *git log* provides project history

```
$ git log
commit aa47da5b74ab89a7929100d62679c2c5b81b2674
Author: Prasun Ray <p.ray@imperial.ac.uk>
Date:    Sun Oct 4 14:49:13 2015 +0100
```

added Galileo

```
commit 6e74cfc8cf12191558751e610d80b76cb321a58c
Author: Prasun Ray <prasun@Prasuns-Air.home>
Date:    Sun Oct 4 14:34:22 2015 +0100
```

initial commit, scientists.txt added to repo

*git diff 6e74cfc8* will list changes made since commit with specified id

# Visiting previous versions

---

- *git log* provides project history
- *git checkout* lets you examine previous versions
- *git checkout master* returns to current version

```
$ git checkout 6e74cfc8c
```

```
Note: checking out '6e74cfc8c'.
```

You are in 'detached HEAD' state. You can look around, make experimental changes and commit them, and you can discard any commits you make in this state without impacting any branches by performing another checkout.

If you want to create a new branch to retain commits you create, you may do so (now or later) by using `-b` with the checkout command again.

Example:

```
git checkout -b new_branch_name
```

```
HEAD is now at 6e74cfc... initial commit, scientists.txt added to repo
```

```
$ cat scientists.txt  
Issac Newton  
Albert Einstein
```

# Local projects with git

---

## List of commands (so far):

*git add*  
*commit*  
*status*  
*diff*  
*log*  
*checkout*

# Local projects with git

---

## List of commands (so far):

*git add*  
*commit*  
*status*  
*diff*  
*log*  
*checkout*

### Getting help:

*git add --help*  
**or**  
*git log --help*

# Local projects with git

---

## List of commands (so far):

*git add*  
*commit*  
*status*  
*diff*  
*log*  
*checkout*

### Getting help:

*git add --help*  
**or**  
*git log --help*

*git rm --cached* **will remove files from index**

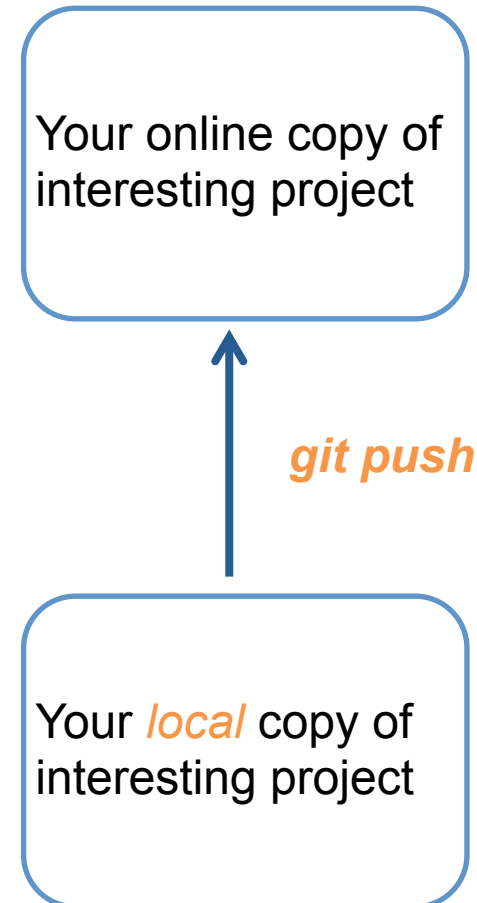
**But be careful, *git rm* will remove files from index *and* working directory**

# Laptop → cloud

---

1. Create online repo in bitbucket:  
*Create* → *Create repository*
2. After creation, bitbucket takes you to  
“Repository setup”
3. Choose either “I’m starting from  
scratch” or “I have an existing project”  
and follow instructions.

Copying & pasting instructions for pre-existing project...



# Laptop → cloud

```
$ git remote add origin https://ImperialHPSC@bitbucket.org/ImperialHPSC/git-example.git

$ git push -u origin --all # pushes up the repo and its refs for the first time
Counting objects: 6, done.
Delta compression using up to 4 threads.
Compressing objects: 100% (2/2), done.
Writing objects: 100% (6/6), 531 bytes | 0 bytes/s, done.
Total 6 (delta 0), reused 0 (delta 0)
To https://ImperialHPSC@bitbucket.org/ImperialHPSC/git-example.git
 * [new branch]      master -> master
Branch master set up to track remote branch master from origin.

$ git push -u origin --tags # pushes up any tags
Everything up-to-date
```

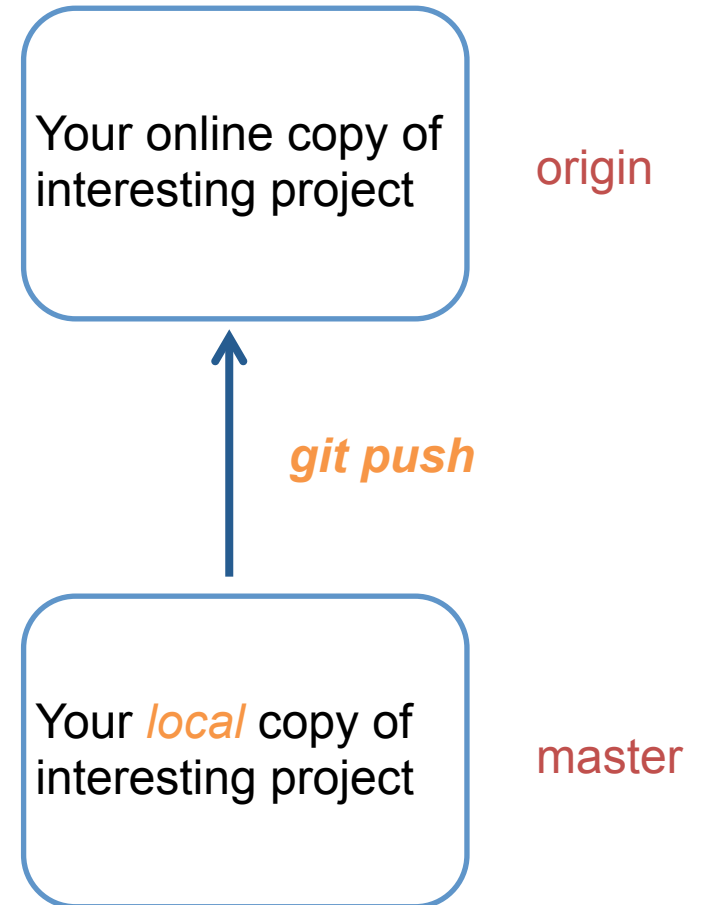
Online repo is named *origin* and the local repo is *pushed to origin*

# Laptop → cloud

---

Push your local repo online for:

- backup
- access to files from other machines
- sharing files



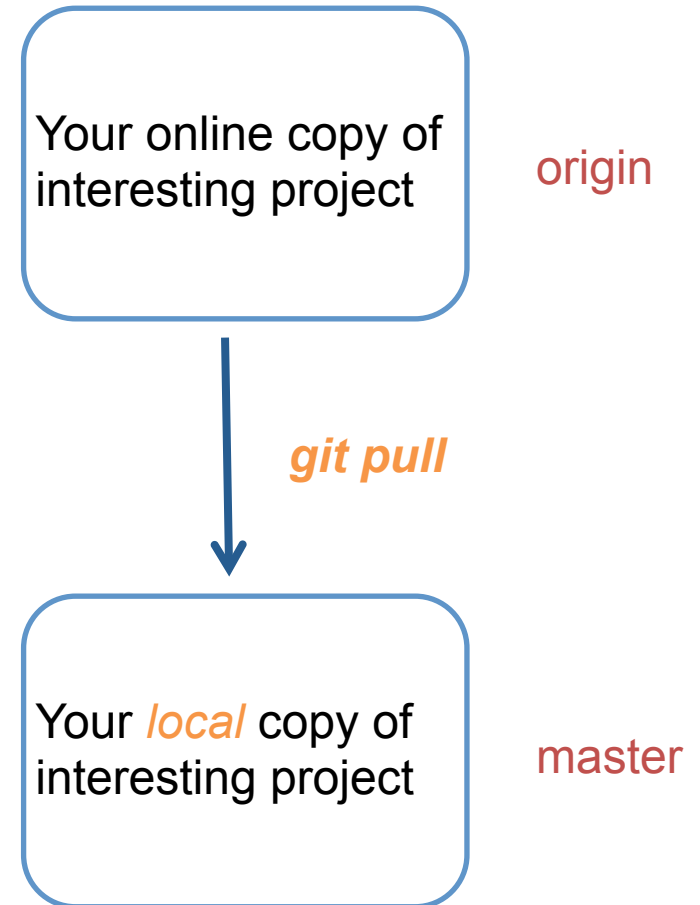


# Laptop ← cloud

---

Push your local repo online for:

- backup
- access to files from other machines
- sharing files
- If online repo gets updated, *pull* update into your local repo



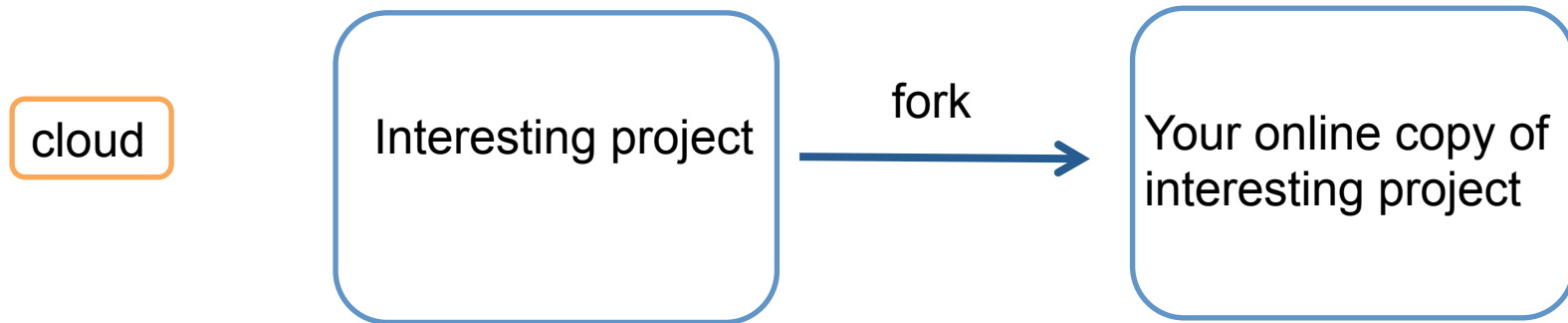
# Online projects

---

- We've seen how to make local projects and *push* them online
- But how do we work on projects that are already online?
  - Will need to *fork* and then *clone* the project

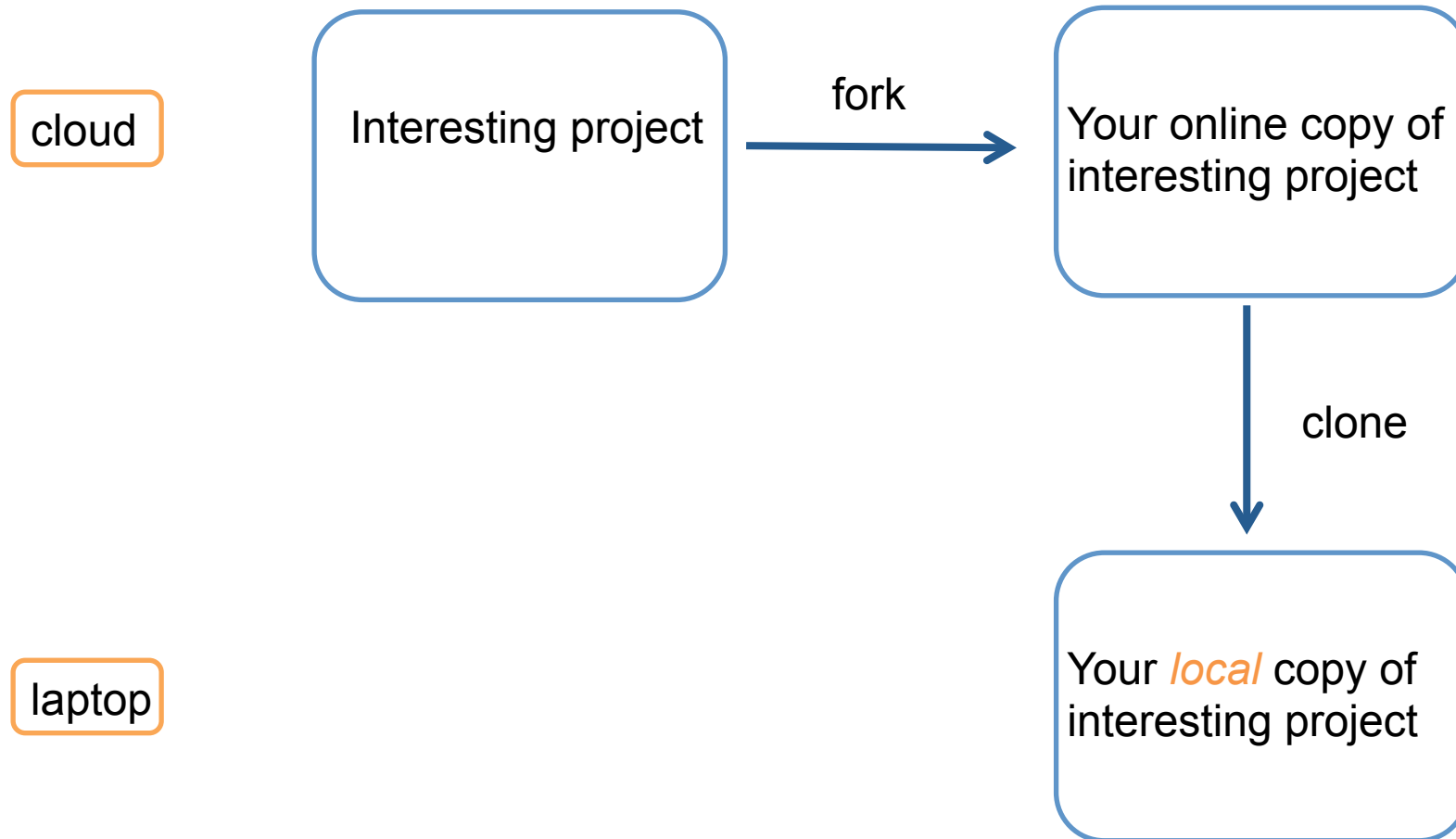
# Online project

---



# Cloud → laptop

---



# Online projects: Example

1. Go to *bitbucket.org/ImperialHPSC/git-example-2* and fork the repo:

The screenshot shows the Bitbucket web interface for the repository `ImperialHPSC / m3c2017`. The browser address bar shows the URL `https://bitbucket.org/ImperialHPSC/m3c2017`. The left sidebar contains navigation links: `Overview` (selected), `Source`, `Commits`, `Branches`, `Pull requests`, `Downloads`, and `Boards`. The main content area displays the `Overview` page, which includes a download button, a dropdown menu set to `HTTPS`, and the repository URL `https://prasunray@bitbucket.org/Impe`. Below this, a table shows repository statistics:

Last updated	2017-10-06	0	2
Access level	Read	Open PRs	Watchers
		1	0
		Branch	Forks

Below the table, the repository description is visible: **Introduction to High Performance Computing**, **Autumn, 2017**, and **Course material**. On the right side, the **Recent activity** section shows a list of commits and repository events, including pushes and the repository's creation.

# Online projects: Example

1. Go to *bitbucket.org/ImperialHPSC/git-example-2* and fork the repo:

The screenshot shows the Bitbucket web interface for the repository 'ImperialHPSC / m3c2017'. The browser address bar shows the URL 'https://bitbucket.org/ImperialHPSC/m3c2017'. The left sidebar contains navigation options: 'Repository', 'Team', 'Project', and 'Snippet'. Below this, under 'GET TO WORK', are options to 'Clone this repository', 'Compare branches or tags', and 'Fork this repository'. The main content area displays the repository overview, including the repository name, a description, and a table of statistics. The 'Recent activity' section on the right shows a list of commits and their authors.

Statistics	Value
Open PRs	0
Watchers	2
Branch	1
Forks	0

**Recent activity**

- 1 commit**  
Pushed to ImperialHPSC/m3c2017  
9bdb773 adding further python material  
Imperial HPSC · 3 days ago
- 1 commit**  
Pushed to ImperialHPSC/m3c2017  
8542f84 adding material for 1st Python vid...  
Imperial HPSC · 3 days ago
- 2 commits**  
Pushed to ImperialHPSC/m3c2017  
80413dd adding lecture 1 material  
ef6a23a first commit  
Imperial HPSC · 4 days ago
- ImperialHPSC/m3c2017**  
Repository created  
Imperial HPSC · 4 days ago

# Online projects: Example

---

## 2. Use *git clone* to make local copy of your fork:

```
$ git clone https://username@bitbucket.org/username/git-example-2.git
Cloning into 'git-example-2'...
remote: Counting objects: 3, done.
remote: Total 3 (delta 0), reused 0 (delta 0)
Unpacking objects: 100% (3/3), done.
Checking connectivity... done.

$ cd git-example-2/
$ ls
mathematicians.txt
```

**Now you have your own copy to play with!**

**Note: If the original repo is updated, bitbucket will tell you and give you the option of syncing your fork**

**or, you can directly pull the updated original repo to your local repo...**

# Online projects: Example

---

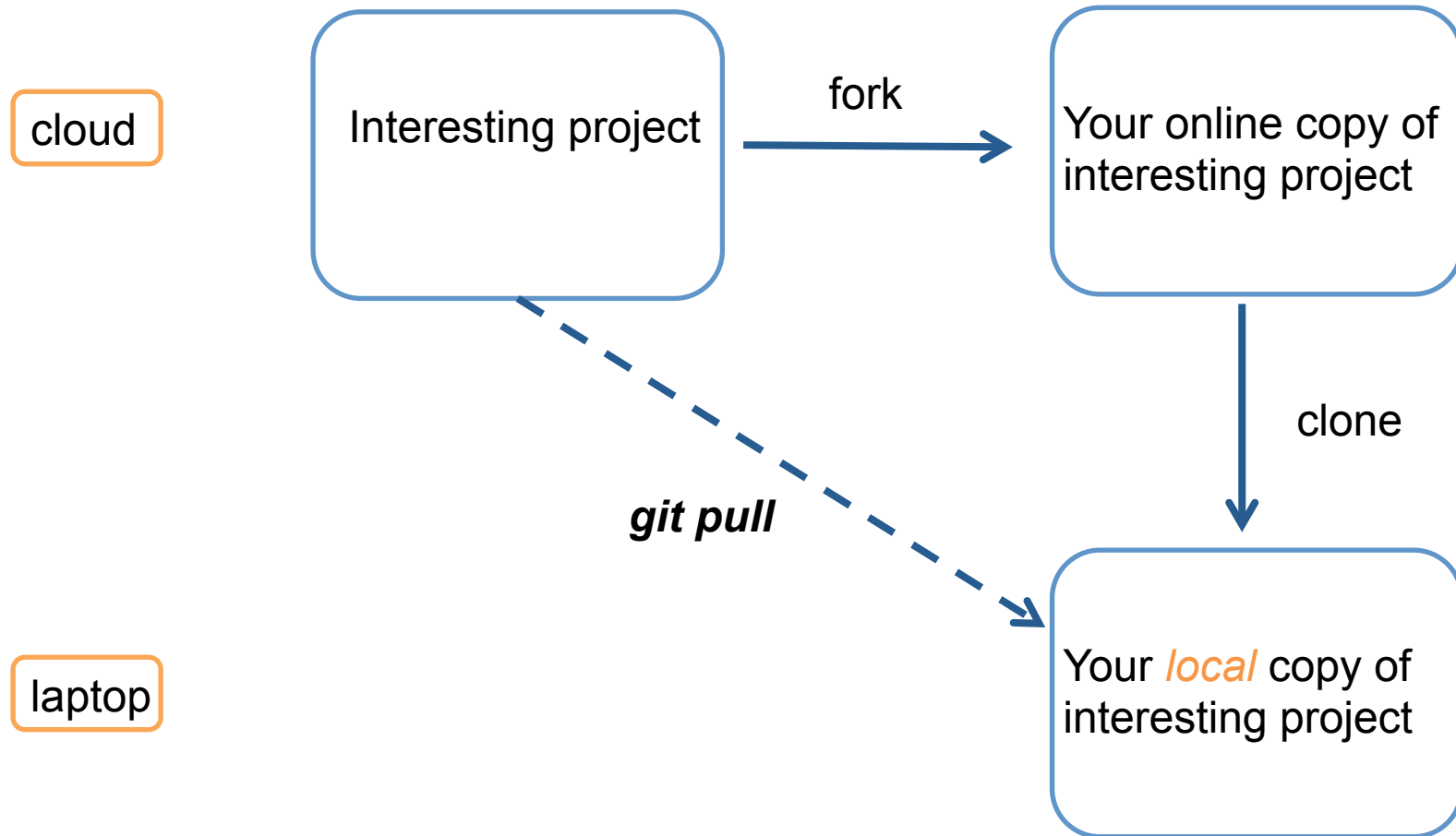
## Notes:

1. You should clone *your fork*, not the original repo!
2. You should push/pull to/from your fork, not the original repo
3. Use *git remote -v* to check that you are pushing/pulling with the right repo



# Cloud → laptop (experienced users)

---



# Cloud → laptop (experienced users)

---

**Procedure: pull from online repo to local branch**

**1<sup>st</sup>, specify online repo:**

*git remote add IHPSC https://username@bitbucket.org/ImperialHPSC/git-example-2.git*

**Here IHPSC is the online repo name**

# Cloud → laptop (experienced users)

---

**Procedure: pull from online repo to local branch**

**1<sup>st</sup>, specify online repo:**

*git remote add IHPSC https://username@bitbucket.org/ImperialHPSC/git-example-2.git*

**Here IHPSC is the online repo name**

**Check which local branch you want to use:**

*git branch* tells you which branch you are on

*git branch -a* lists all branches

*git checkout branch\_name* switches to branch\_name

# Cloud → laptop (experienced users)

---

**Procedure: pull from online repo to local branch**

**1<sup>st</sup>, specify online repo:**

*git remote add IHPSC https://username@bitbucket.org/ImperialHPSC/git-example-2.git*

**Here IHPSC is the online repo name**

**Check which local branch you want to use:**

*git branch* tells you which branch you are on

*git branch -a* lists all branches

*git checkout branch\_name* switches to branch\_name

**For example,**

*git pull IHPSC master*

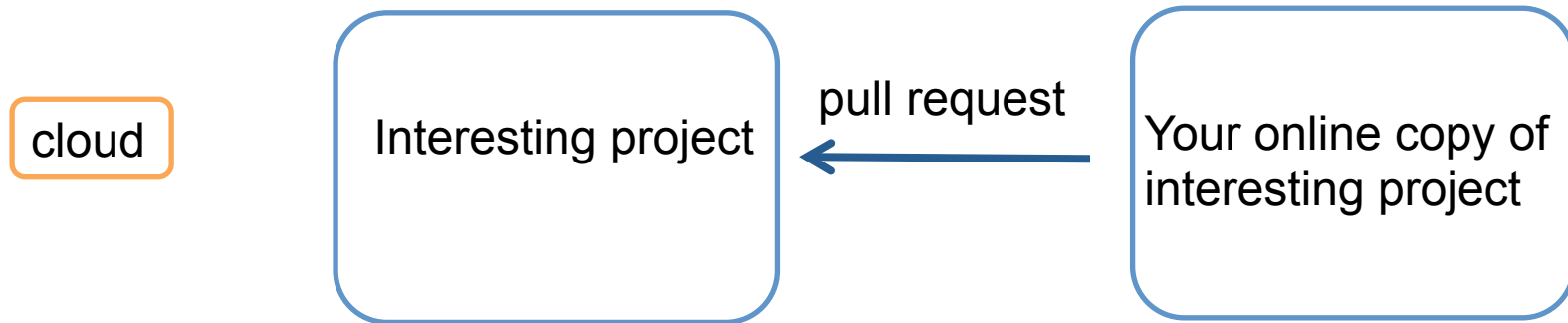
**will pull from the IHPSC online repo to your local master branch**

**Finally, push your local repo to your fork:**

*git push origin master*

# Pull requests

---



**Can ‘suggest’ improvements to original project with pull requests on bitbucket**

**Essential for collaborative work**

**If you accidentally push an update to my repo instead of yours, I (and everyone else) will see a pull request**

- 
- **Lab 1 this week**
    - **Further practice with Unix**
    - **Guide to syncing with the class bitbucket repo**
    - **If you have time, create a personal account on bitbucket in advance**