



Using Github



What Github Does

- Online project hosting site.
- "Remote" git repository with access control
- Issue Tracking
- Project Boards
- Documentation and web pages (github.io)
- Integrates with other services
 - Travis CI for automatic testing

Github Profile

Example of SKE student profiles.

1. Real name
2. Photo
3. (Optional) Email
4. Description of you



Jirayu Laungwilawan
Jirayul

Faculty of Engineering , Major -
Software and Knowledge
Engineering.

Follow

Block or report user

📍 Thailand

✉ jirayu.l@ku.th

🌐 <https://github.com/Jirayul>



Kongpon Charanwattanakit
kykungz

Software Developer, Undergraduate
Software and Knowledge
Engineering Student

Follow

Block or report user

👤 Kasetsart University

📍 Bangkok, Thailand

✉ jackykongpon@gmail.com

🌐 <https://kykungz.github.io/>

Creating and using a Repository

Case 1: *Project code is on your local computer.
You want to copy it to Github.*

Case 2: *Project already exists on Github.
You want to copy it to your computer.*

Special Case:

Case 3: *A new project (no files yet).*

Case 1: Starting from Local Project

You already have a project on your computer

1. Create a **local** "git" repository.

```
cmd> git init
```

```
# These two files are typical
```

```
cmd> git add .gitignore README.md
```

```
# Add some source code
```

```
cmd> git add src/*.java (for example)
```

```
# Commit code to github
```

```
cmd> git commit -m "initial code checkin"
```


Case 1: Remote must be empty

2. Create an empty repository on Github.

Create a new repository

A repository contains all the files for your project, including the revision history.

Owner

 fatalaijon ▾

Repository name

/ demo



Great repository names are short and memorable. Need inspiration? How about **symmetrical**

Description (optional)

Demonstration project

Case 1: add Github as remote

3. Copy the URL of new Github repository (https or ssh).

Quick setup — if you've done this kind of thing before

or



We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

4. In your local project, add Github as a remote repository named "origin":

```
cmd> git remote add origin  
      https://github.com/fatalaijon/demo.git
```

5. Push (copy) the local repository to Github

```
cmd> git push -u origin master
```

You only need "-u origin master" the first time you push to Github. Next time, just type "git push".

Case 2: Starting from Github

A project already exists on Github.

You want to "clone" it your local computer.

1. Get the Github project URL

`https://github.com/user/demo.git`

or: go to project on Github and click on



and copy the URL.

2. In your **workspace directory**, type:

`cmd> git clone https://github.com/user/demo`

*NOTE: "git clone" creates a new directory named "demo" inside your current directory. If this directory already exists, clone **won't work**.*

Case 2: ready to use

That's it!

Github is automatically the remote named "origin".

Just "**git push**" your committed work to github.

You can use a different project name

The name of your **local directory** (cloned from Github) can be **different** from the Github repository name.

1) Specify local directory name when you "**clone**":

Clone "**demo**" into local directory "**mydemo**"

```
cmd> git clone https://github.com/fatalai  
jon/demo.git mydemo
```

Syntax: `git clone remote_url local_repo_name`

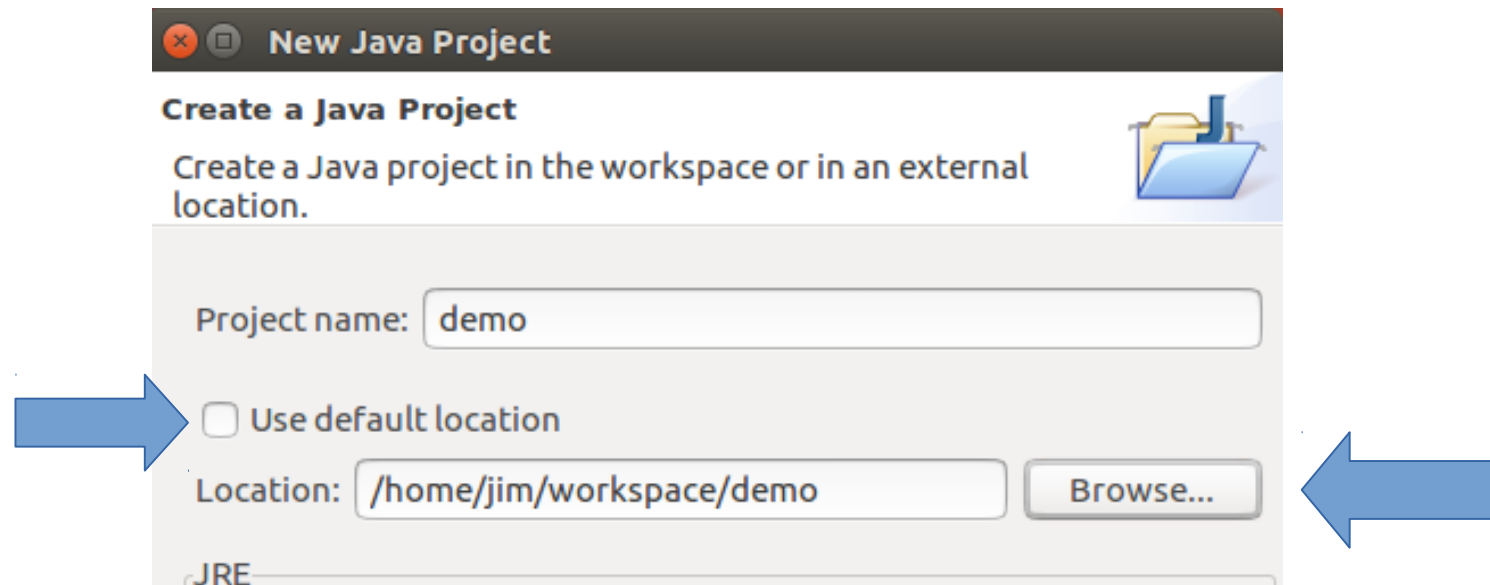
-- or --

2) or rename the directory yourself!

use the command line or any file manager

Eclipse create project with existing source

After cloning code from a remote, create an IDE project from "existing source code".



1. uncheck "use default location"

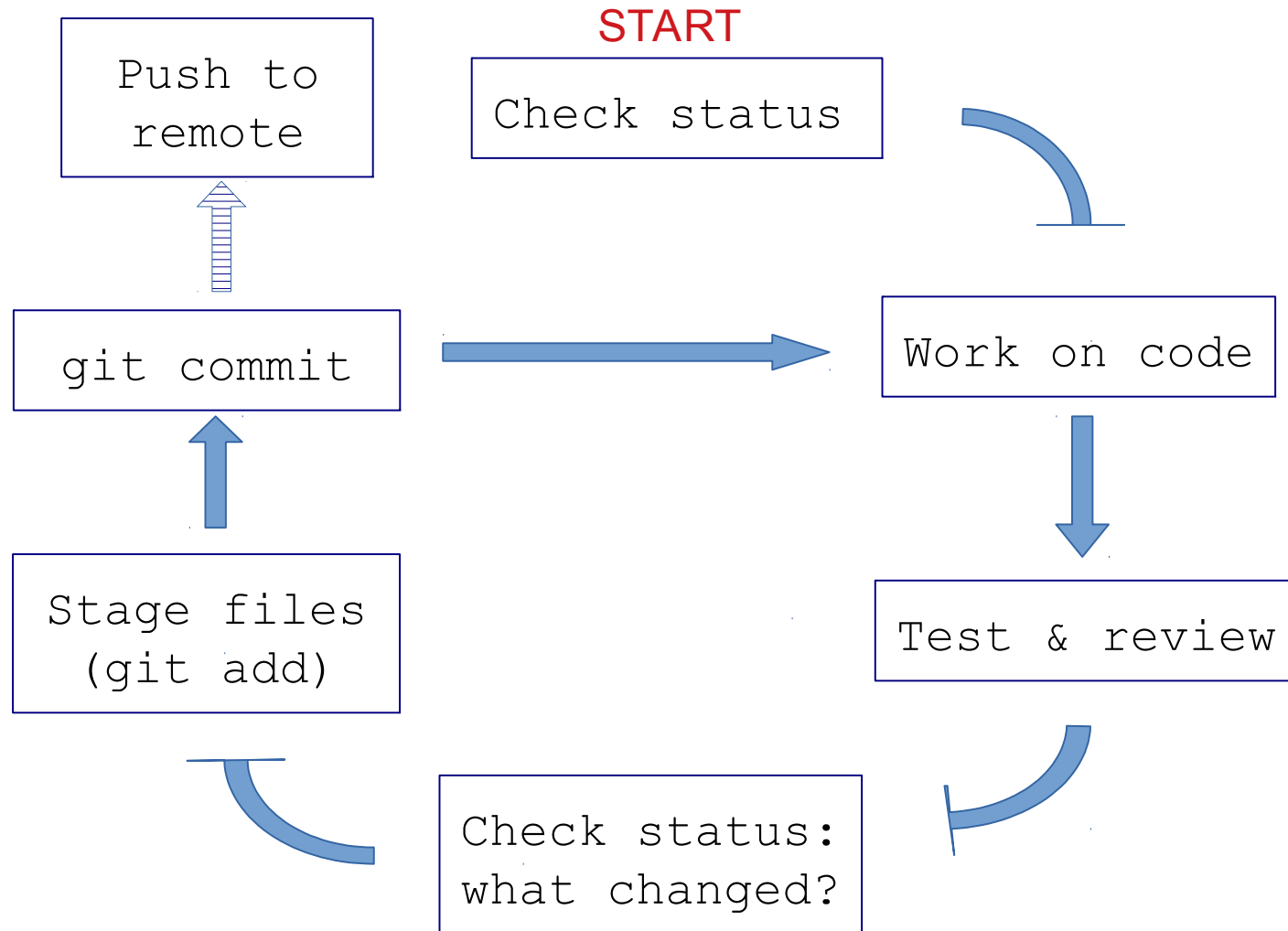
2. locate the project folder & select it

Simpler: Most IDE have a menu function to create a project from a remote git repository. They clone it and create project all in one.

Comparison of 2 Cases

(done in class)

Workflow for an *individual* project



Git Workflow for an *Individual* project

1) Check status of your working copy (*)

```
cmd> git status
```

It should be clean. If not, do "git diff" and then...

2) Commit changes or update your working copy.

```
(git diff, git add -u, git commit)
```

3) Do some work:

Code, test. Code, test. Review.

(*) *if you work on more than one computer, you need to "fetch" or "pull" any work from Github that is not on this computer (i.e. this local repo).*

Git Workflow (cont'd)

4) After code-test-reivew: check status again

```
cmd> git status
```

```
Changes not staged for commit:
```

```
    modified:   src/Problem2.java
```

```
Untracked files:
```

```
    src/Problem3.java
```

5) **Add** and **commit** your work to the local repository

```
cmd> git add src/Problem2.java src/Problem3.java
```

```
cmd> git commit -m "Solved problems 2 and 3"
```

```
[master 29abae0] Solved problem 2 and 3
```

```
2 files changed, 44 insertions(+), 5 deletions
```

Git Workflow (update remote)

6) Push the changes to Github

```
cmd> git push
```

```
Compressing objects: 100% (12/12), done.
```

```
Writing objects: 100% (12/12), 3.60 KiB,  
done.
```

```
Total 12 (delta 9), reused 0 (delta 0)
```

```
remote: Resolving deltas: 100% (9/9), ...
```

```
To https://github.com/fatailaijon/demo.git
```

```
468abdf..29abae0  master -> master
```

7) Take a break.

That's it! Repeat the cycle as you work.

Github Workflow for Team Projects

On a **team project**, other people will commit files to the **same** Github repository!

You should update your local repository from Github before trying to "push" your work to Github.

Use "**Github Flow**" as workflow in team projects.

"Github Flow" is a separate topic. Its good for both team and solo projects.

Github Flow is the convention for team work in this course.