Esercitazioni Prova Finale 2019

Part 1 - Intro & Tools

Docente Gianpaolo Cugola

Esercitatore Mario Scrocca

Code: https://github.com/marioscrock/ingsoft-prova-finale-19

Course Info

5 Mar, 12 Mar, 26 Mar, TBA

- Lessons will be on Tuesdays L26.04 14.30 - 17.00 (4 lessons, 3 hours each = 12 hours)

- Labs will be on Tuesdays 14.30 - 18.00 (8 labs, 4 hours each = 32 hours)

- The project will be a board game to be developed in groups of 3 persons

Course Goals

"*La prova finale consiste nella preparazione, **in autonomia** ma sotto la guida del docente e dei responsabili di laboratorio, di un elaborato software da svolgere in Java*"

- No written exam, you will do something practical :)
- You will use your hands and face a real problem
- There is **no correct answer sometimes**, you'll have to justify your choices
- You are alone (as a team :)

Esercitazioni

- Practical, so **bring your own pc** ;)
- **Review theory** before
- Questions strictly related to code seen in class -> Github issues/Pull requests in repo
- All other questions(e.g. tools) -> **Piazza**
- Any doubt? During (**ask** if you do not understand!) or after lesson

Programming

A set of different activities, not only writing code!

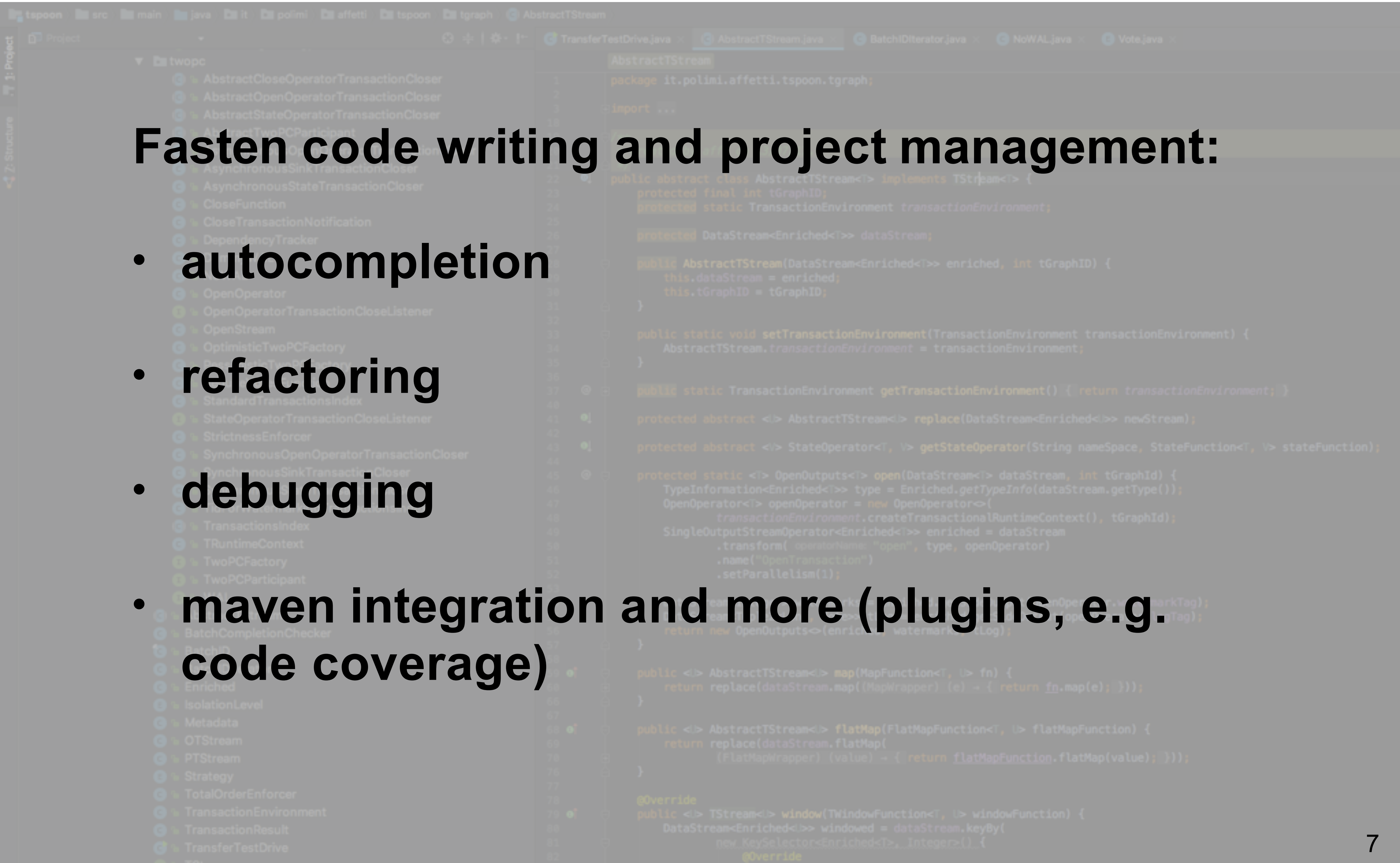
- **System design** (requirements definition, …)
- **Code writing** (and debugging!!)
- **Problem solving**
- **Project management** (managing dependencies, …)
- **Code sharing**

There are well-defined **tools** that help us (developers) in these activities

Programming

- **System design** —> your brain (+ Design Patterns!)
- **Code writing** —> IDE + others
- **Problem solving** —> Google & Stackoverflow
- **Project management** —> Apache Maven + Sonar
- **Code sharing** —> VCS (git) + GitHub

Writing: the IDE



```
package it.polimi.affetti.tspoon.tgraph;

import ...

public abstract class AbstractTStream<T> implements TStream<T> {
    protected final int tGraphID;
    protected static TransactionEnvironment transactionEnvironment;

    protected DataStream<Enriched<T>> dataStream;

    public AbstractTStream(DataStream<Enriched<T>> enriched, int tGraphID) {
        this.dataStream = enriched;
        this.tGraphID = tGraphID;
    }

    public static void setTransactionEnvironment(TransactionEnvironment transactionEnvironment) {
        AbstractTStream.transactionEnvironment = transactionEnvironment;
    }

    public static TransactionEnvironment getTransactionEnvironment() { return transactionEnvironment; }

    protected abstract <U> AbstractTStream<U> replace(DataStream<Enriched<U>> newStream);

    protected abstract <V> StateOperator<T, V> getStateOperator(String nameSpace, StateFunction<T, V> stateFunction);

    protected static <T> OpenOutputs<T> open(DataStream<T> dataStream, int tGraphId) {
        TypeInformation<Enriched<T>> type = Enriched.getTypeInfo(dataStream.getType());
        OpenOperator<T> openOperator = new OpenOperator<>(
                transactionEnvironment.createTransactionalRuntimeContext(), tGraphId);
        SingleOutputStreamOperator<Enriched<T>> enriched = dataStream
                .transform( operatorName: "open", type, openOperator)
                .name("OpenTransaction")
                .setParallelism(1);

        return new OpenOutputs<>(enriched, watermarks, tLog);
    }

    public <U> AbstractTStream<U> map(MapFunction<T, U> fn) {
        return replace(dataStream.map((MapWrapper) (e) → { return fn.map(e); }));
    }

    public <U> AbstractTStream<U> flatMap(FlatMapFunction<T, U> flatMapFunction) {
        return replace(dataStream.flatMap(
                (FlatMapWrapper) (value) → { return flatMapFunction.flatMap(value); }));
    }

    @Override
    public <U> TStream<U> window(TWindowFunction<T, U> windowFunction) {
        DataStream<Enriched<U>> windowed = dataStream.keyBy(
                new KeySelector<Enriched<T>, Integer>() {
                    @Override
```

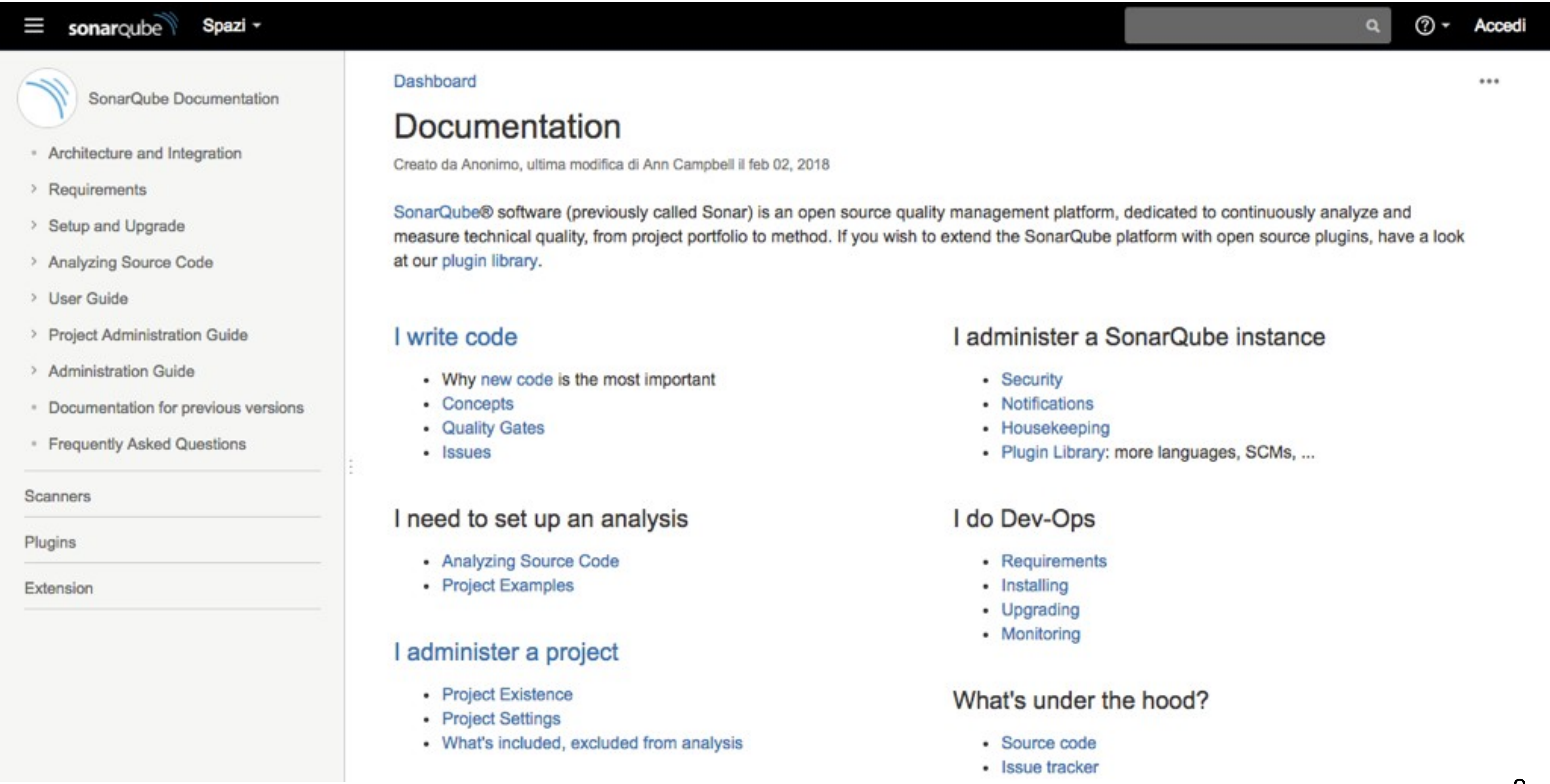
Fasten code writing and project management:

- **autocompletion**
- **refactoring**
- **debugging**
- **maven integration and more (plugins, e.g. code coverage)**

Writing: Sonar

sonarqube.org

Code quality platform



SonarQube Documentation

- Architecture and Integration
- Requirements
- Setup and Upgrade
- Analyzing Source Code
- User Guide
- Project Administration Guide
- Administration Guide
- Documentation for previous versions
- Frequently Asked Questions

Scanners

Plugins

Extension

Dashboard

Documentation

Creato da Anonimo, ultima modifica di Ann Campbell il feb 02, 2018

SonarQube® software (previously called Sonar) is an open source quality management platform, dedicated to continuously analyze and measure technical quality, from project portfolio to method. If you wish to extend the SonarQube platform with open source plugins, have a look at our plugin library.

I write code

- Why new code is the most important
- Concepts
- Quality Gates
- Issues

I need to set up an analysis

- Analyzing Source Code
- Project Examples

I administer a project

- Project Existence
- Project Settings
- What's included, excluded from analysis

I administer a SonarQube instance

- Security
- Notifications
- Housekeeping
- Plugin Library: more languages, SCMs, ...

I do Dev-Ops

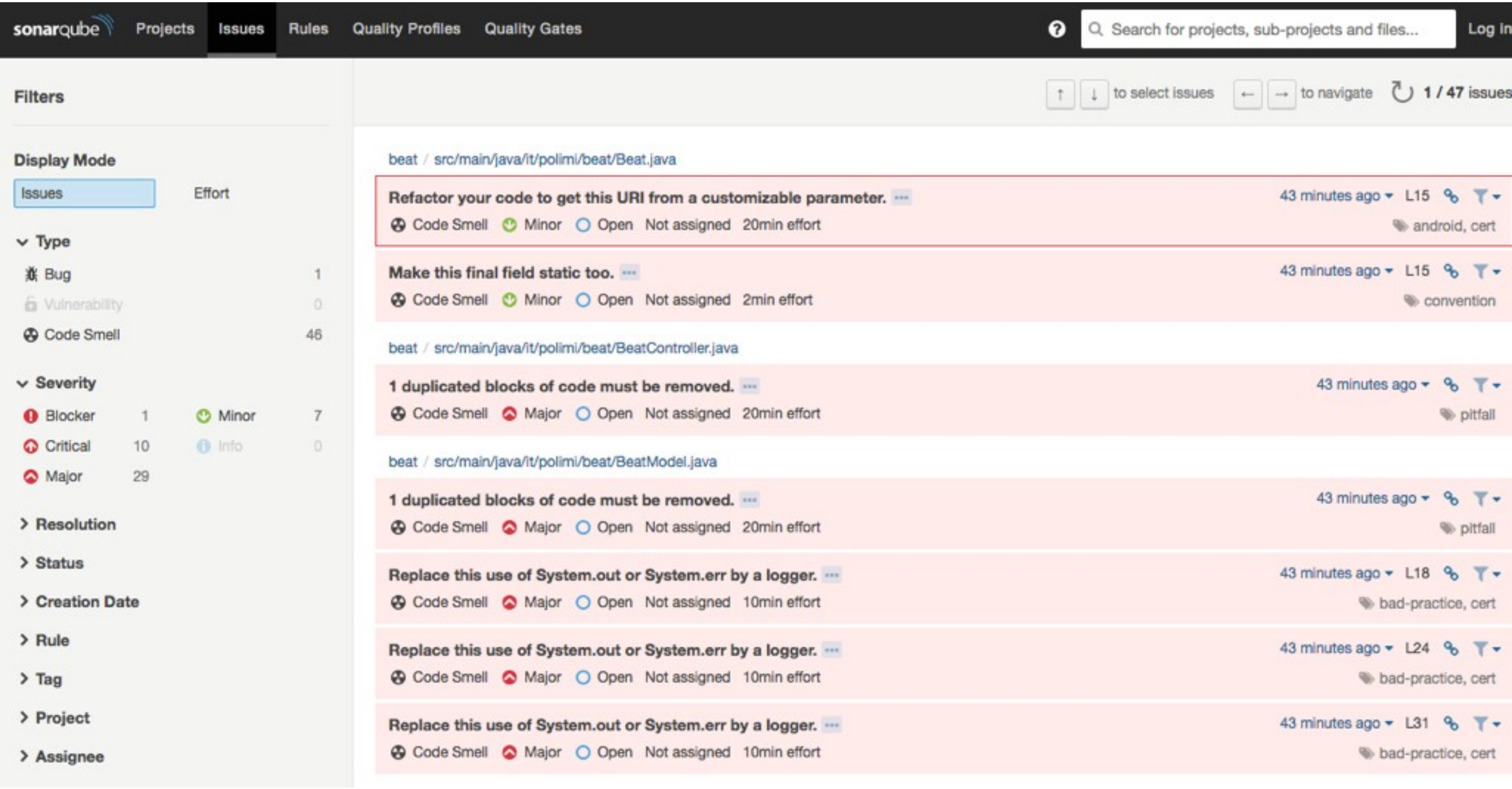
- Requirements
- Installing
- Upgrading
- Monitoring

What's under the hood?

- Source code
- Issue tracker

Writing: Sonar

Signals issues in your code and provides useful insights for best practices by explaining the issues



Type	
Bug	1
Vulnerability	0
Code Smell	46

Severity			
Blocker	1	Minor	7
Critical	10	Info	0
Major	29		

beat / src/main/java/it/polimi/beat/Beat.java

Refactor your code to get this URI from a customizable parameter. — 43 minutes ago — L15
Code Smell · Minor · Open · Not assigned · 20min effort — android, cert

Make this final field static too. — 43 minutes ago — L15
Code Smell · Minor · Open · Not assigned · 2min effort — convention

beat / src/main/java/it/polimi/beat/BeatController.java

1 duplicated blocks of code must be removed. — 43 minutes ago
Code Smell · Major · Open · Not assigned · 20min effort — pitfall

beat / src/main/java/it/polimi/beat/BeatModel.java

1 duplicated blocks of code must be removed. — 43 minutes ago
Code Smell · Major · Open · Not assigned · 20min effort — pitfall

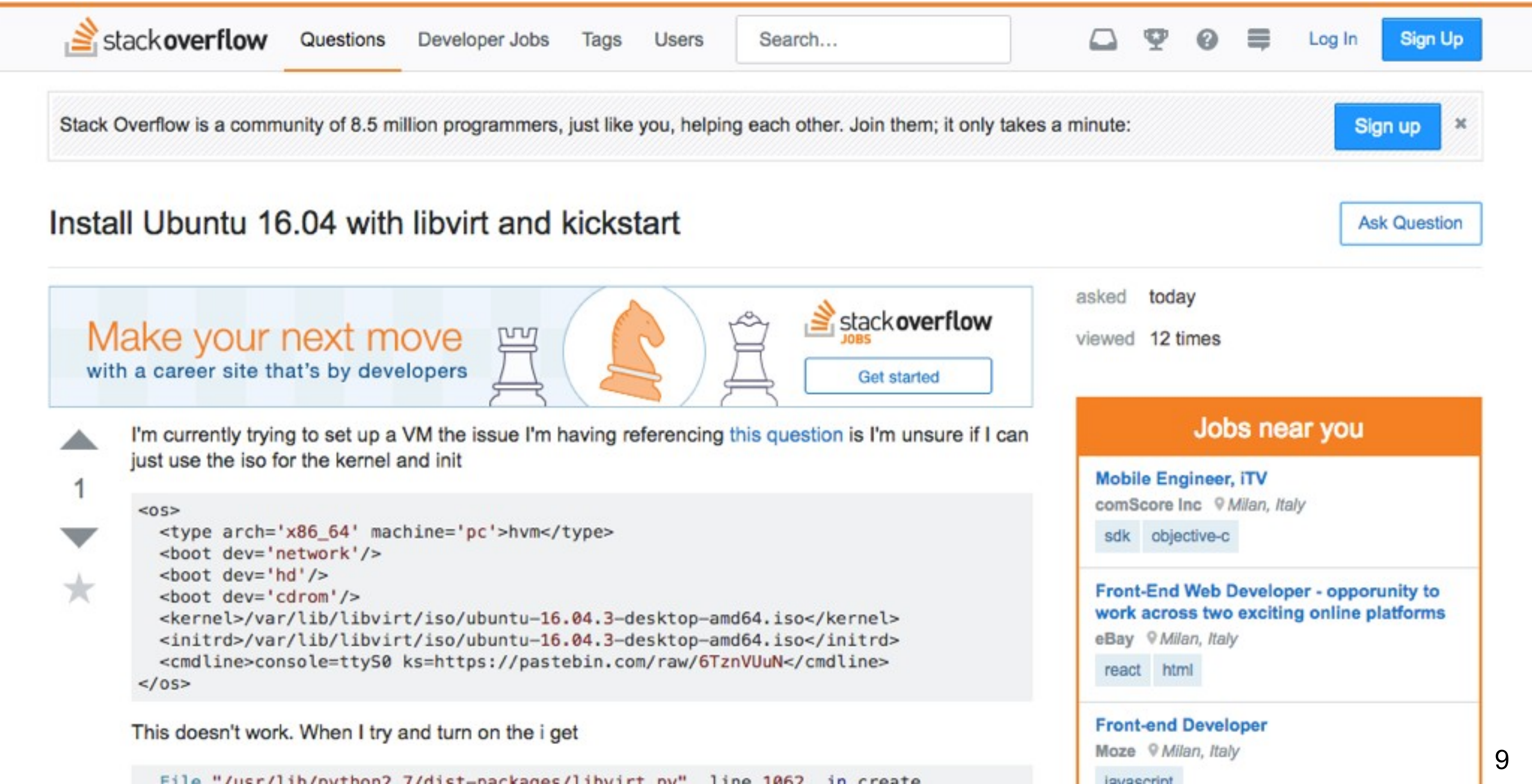
Replace this use of System.out or System.err by a logger. — 43 minutes ago — L18
Code Smell · Major · Open · Not assigned · 10min effort — bad-practice, cert

Replace this use of System.out or System.err by a logger. — 43 minutes ago — L24
Code Smell · Major · Open · Not assigned · 10min effort — bad-practice, cert

Replace this use of System.out or System.err by a logger. — 43 minutes ago — L31
Code Smell · Major · Open · Not assigned · 10min effort — bad-practice, cert

Issues: Google & StackOverflow

stackoverflow.com



Stack Overflow is a community of 8.5 million programmers, just like you, helping each other. Join them; it only takes a minute:

Install Ubuntu 16.04 with libvirt and kickstart

I'm currently trying to set up a VM the issue I'm having referencing this question is I'm unsure if I can just use the iso for the kernel and init

```
<os>
  <type arch='x86_64' machine='pc'>hvm</type>
  <boot dev='network'/>
  <boot dev='hd'/>
  <boot dev='cdrom'/>
  <kernel>/var/lib/libvirt/iso/ubuntu-16.04.3-desktop-amd64.iso</kernel>
  <initrd>/var/lib/libvirt/iso/ubuntu-16.04.3-desktop-amd64.iso</initrd>
  <cmdline>console=ttyS0 ks=https://pastebin.com/raw/6TznVUuN</cmdline>
</os>
```

This doesn't work. When I try and turn on the i get

```
  File "/usr/lib/python2.7/dist-packages/libvirt.py", line 1062, in create
```

Project management: Maven

- A tools for managing your project:

  - **specify dependencies** (local or remote modules)
  - **compile**
  - **test**
  - **package**

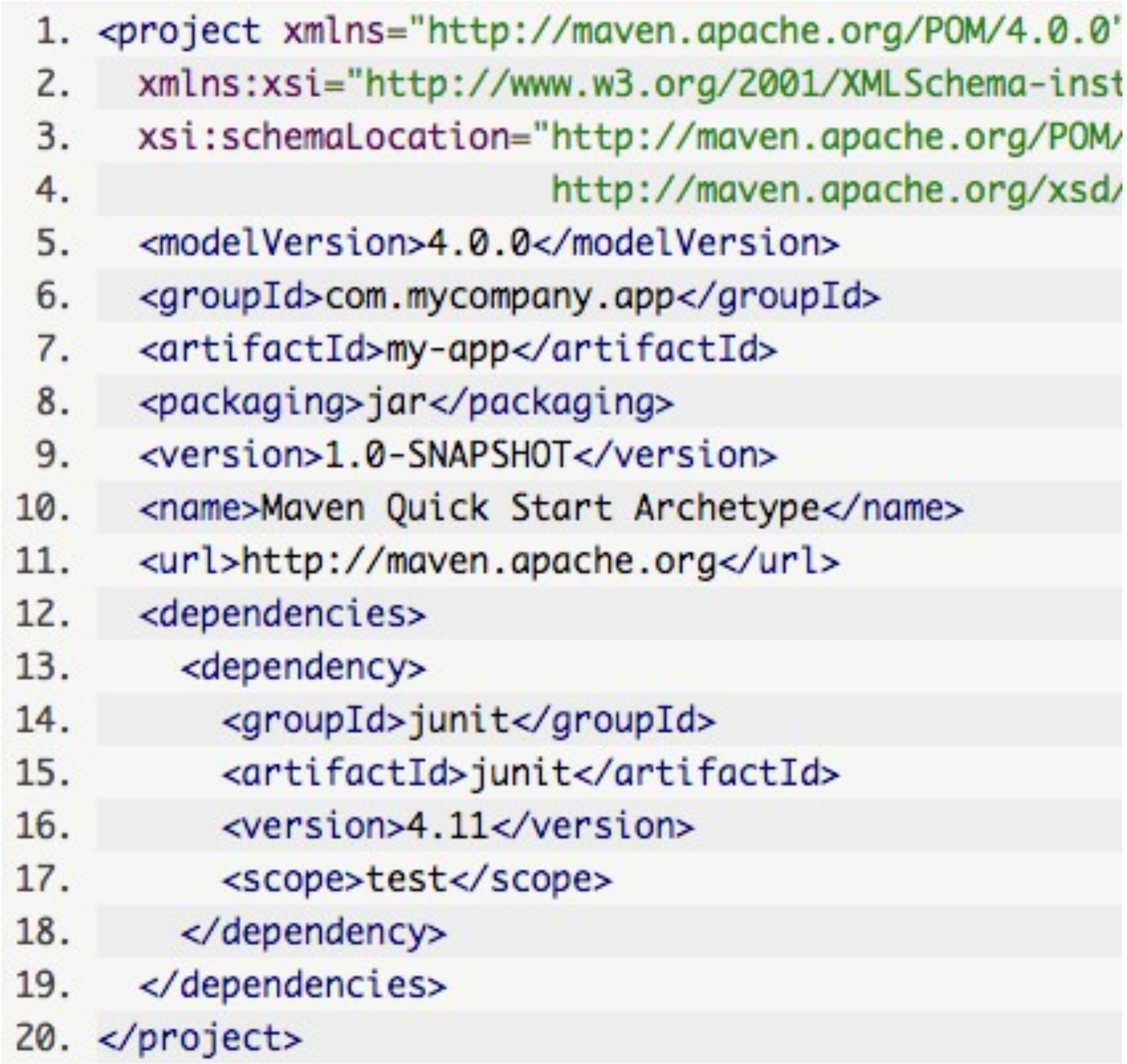
- It helps you in building and installing your project by means of *conventions*.

- Every project has a **POM** (Project Object Model).

https://maven.apache.org/guides/getting-started/

The POM

- An **XML** file that contains the information of this project:
    - It’s ID (groupID, artifactID)
    - Dependencies
    - Build instructions (plugins and configuration)



```
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-inst
  xsi:schemaLocation="http://maven.apache.org/POM/
                      http://maven.apache.org/xsd/
  <modelVersion>4.0.0</modelVersion>
  <groupId>com.mycompany.app</groupId>
  <artifactId>my-app</artifactId>
  <packaging>jar</packaging>
  <version>1.0-SNAPSHOT</version>
  <name>Maven Quick Start Archetype</name>
  <url>http://maven.apache.org</url>
  <dependencies>
    <dependency>
      <groupId>junit</groupId>
      <artifactId>junit</artifactId>
      <version>4.11</version>
      <scope>test</scope>
    </dependency>
  </dependencies>
</project>
```

Goals

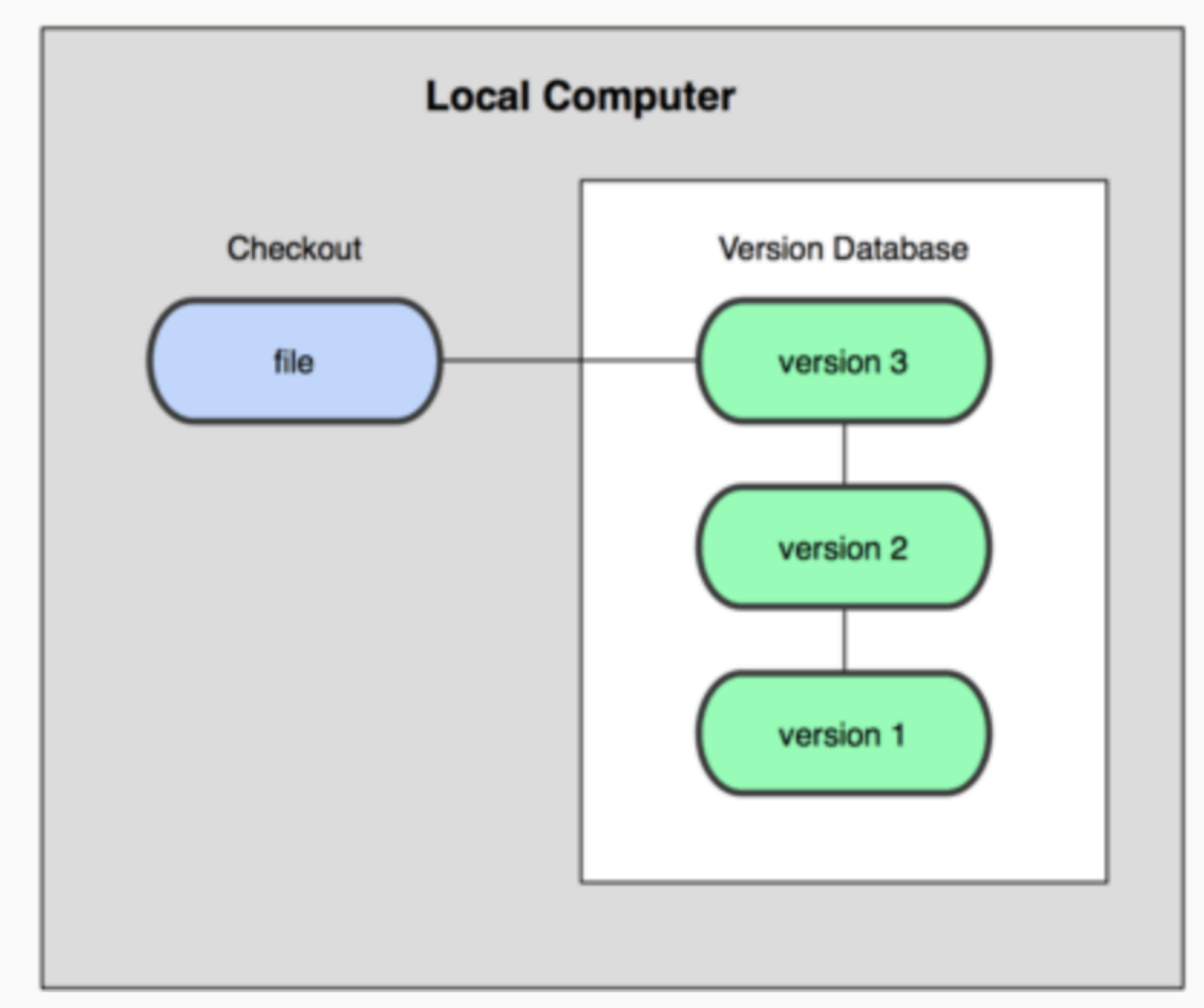
- Maven allows you to run project management **goals**, such as:
  - mvn **generate** … —> generates a project from an *archetype*
  - mvn **compile** —> compiles the sources
  - mvn **test** —> launches tests
  - mvn **package** —> creates a JAR package from the compiled files
  - mvn **install** —> installs the package in the local maven repository

Developing together: **Git**

- **VCS** (Version Control System) records changes to a file or set of files over time so that you can recall specific versions later.
- **Revert** files back to a previous state
- **Review** changes made over time
- **See who last modified** something that might be causing a problem
- In general, if you screw things up or lose files, you can generally recover easily (**NO PANIC!**).

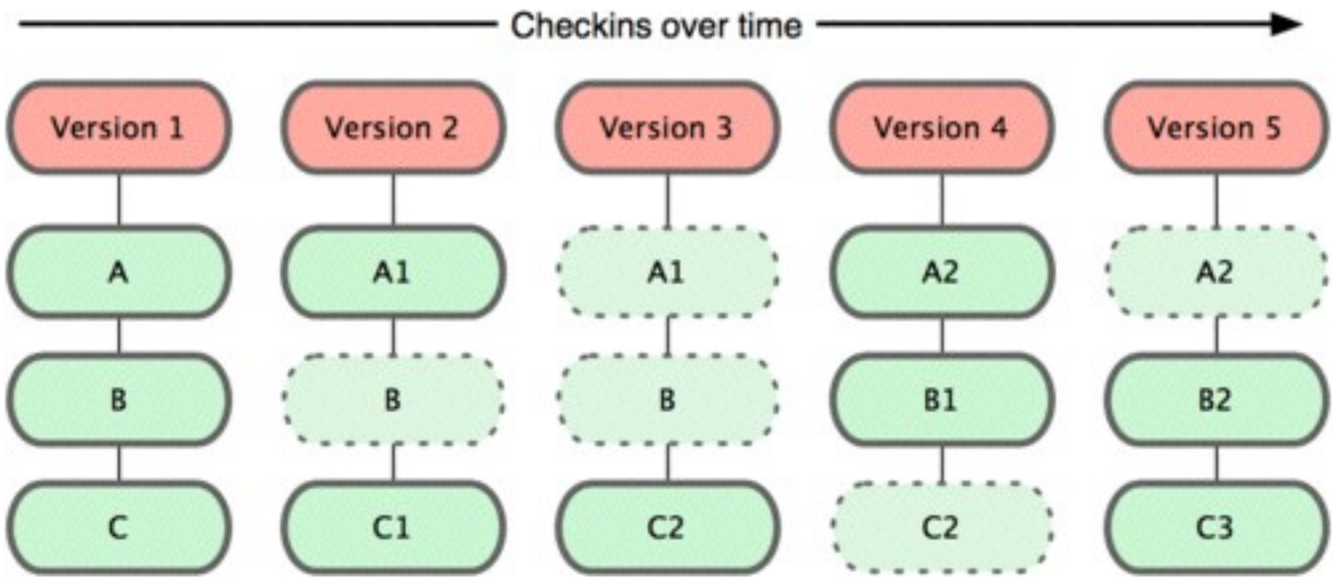
https://git-scm.com/book/en/v1/Getting-Started-About-Version-Control

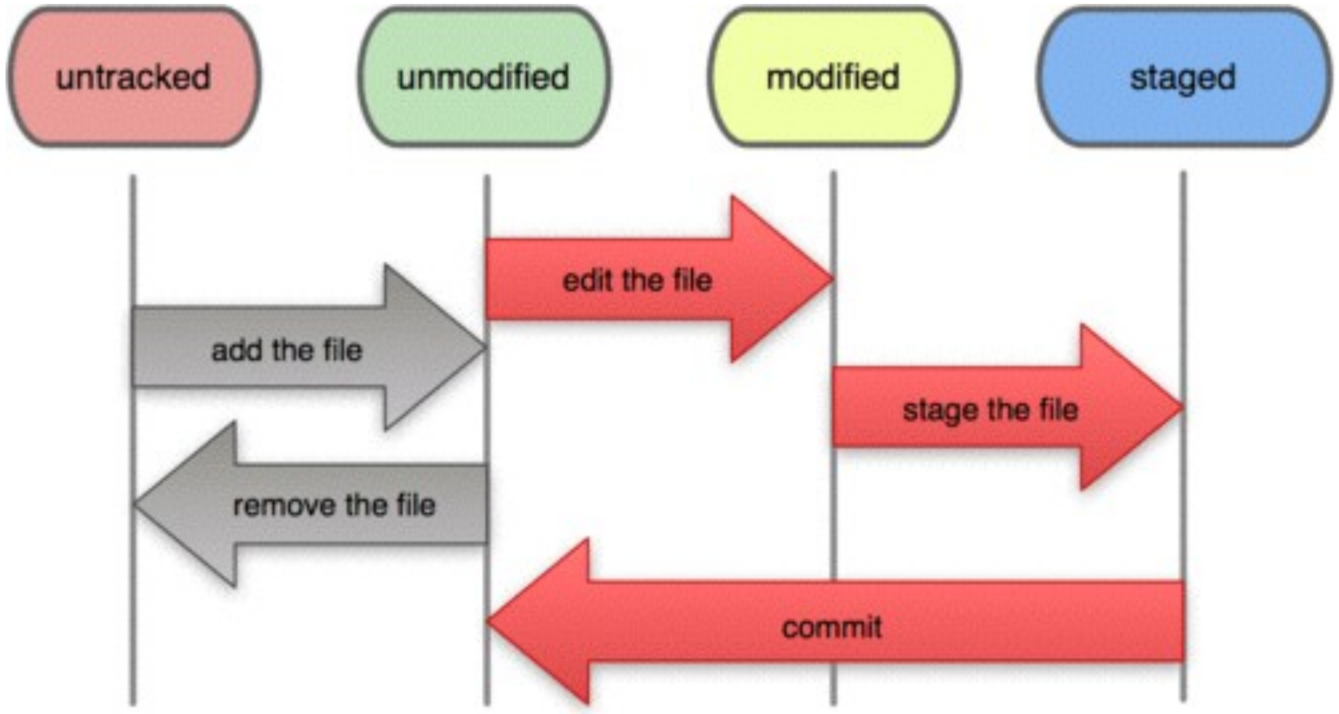
Developing together: **Git**



Courtesy of git-scm.com

Developing together: **Git**

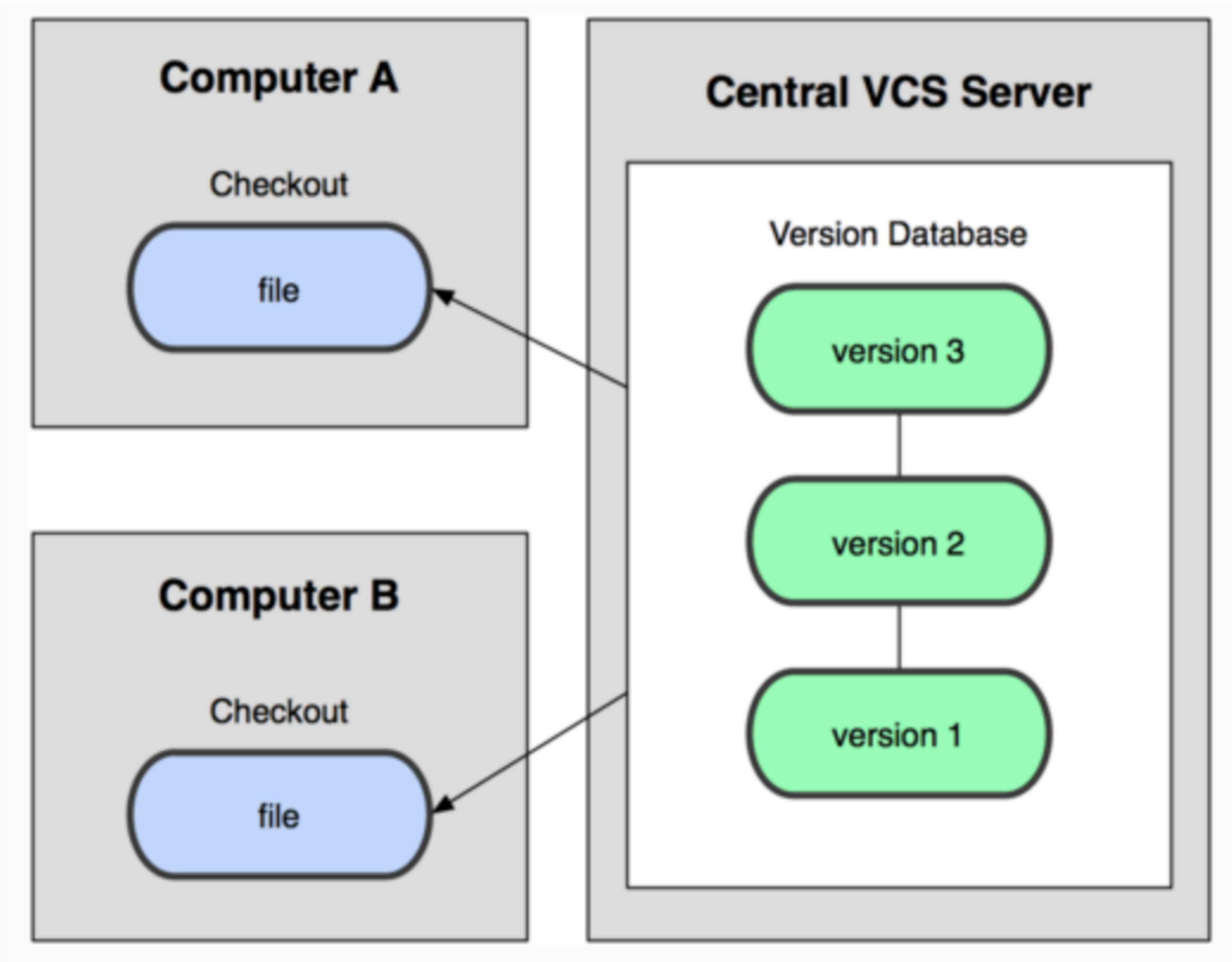




- Git keeps snapshots of files for tracking history
- A file can be untracked, unmodified, modified, staged
- We'll see commands during the course

Courtesy of git-scm.com

Developing together: **Git**

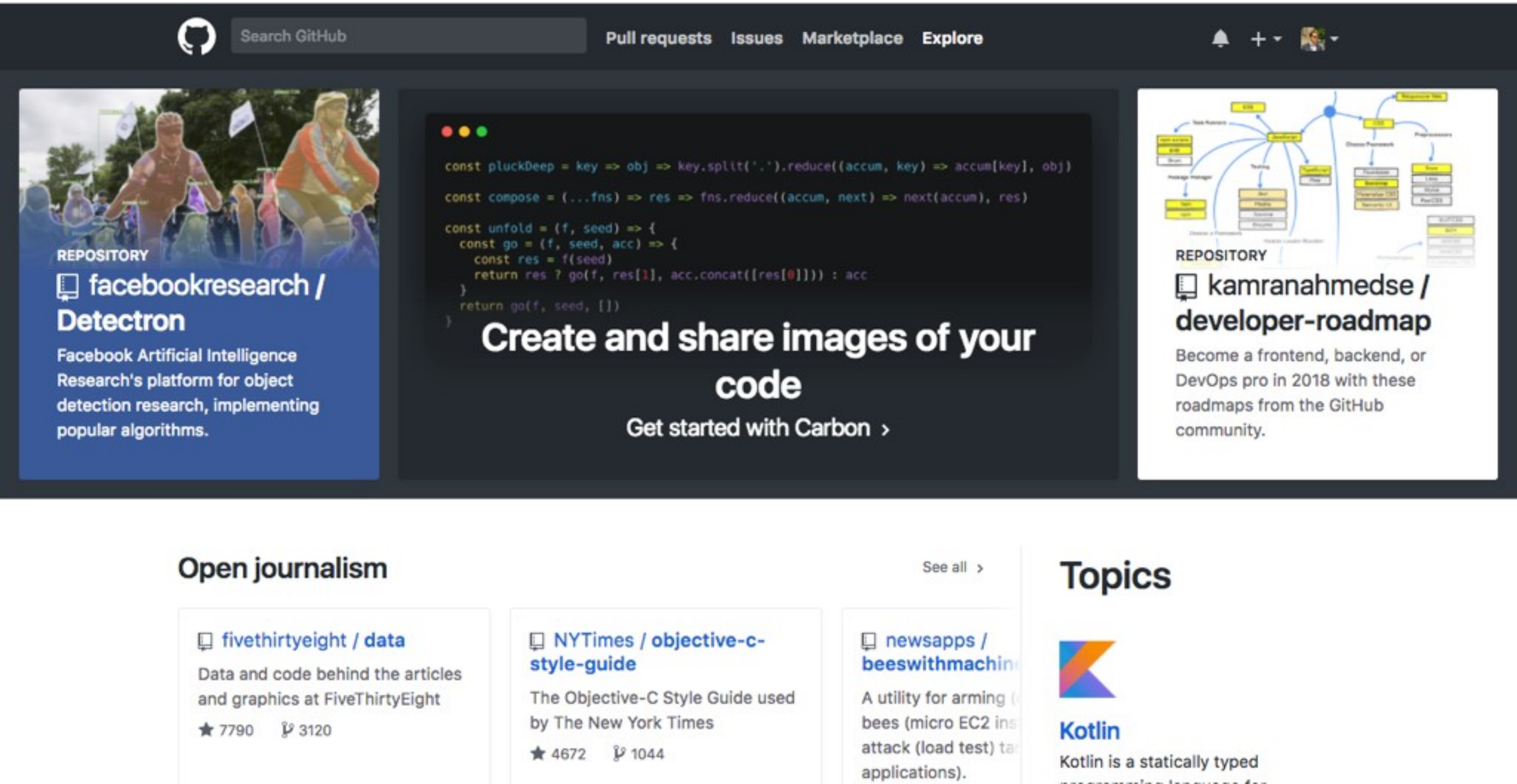


- Central VCS Server
- Every machine fully mirrors the repository
- Local operations on versions and push to remote

Courtesy of git-scm.com

Developing together: **GitHub**

github.com
“Only” a service that hosts a git server.



REPOSITORY

facebookresearch / Detectron

Facebook Artificial Intelligence Research's platform for object detection research, implementing popular algorithms.

```
const pluckDeep = key => obj => key.split('.').reduce((accum, key) => accum[key], obj)

const compose = (...fns) => res => fns.reduce((accum, next) => next(accum), res)

const unfold = (f, seed) => {
  const go = (f, seed, acc) => {
    const res = f(seed)
    return res ? go(f, res[1], acc.concat([res[0]])) : acc
  }
  return go(f, seed, [])
}
```

Create and share images of your code

Get started with Carbon ›

REPOSITORY

kamranahmedse / **developer-roadmap**

Become a frontend, backend, or DevOps pro in 2018 with these roadmaps from the GitHub community.

Open journalism

See all ›

fivethirtyeight / **data**

Data and code behind the articles and graphics at FiveThirtyEight

★ 7790 3120

NYTimes / **objective-c-style-guide**

The Objective-C Style Guide used by The New York Times

★ 4672 1044

newsapps / **beeswithmachin**

A utility for arming (
bees (micro EC2 ins
attack (load test) ta
applications).

Topics

Kotlin

Kotlin is a statically typed

Useful links

REPO: https://github.com/marioscrock/ingsoft-prova-finale-19

- IntelliJ for Students https://www.jetbrains.com/student/
- Get started with Sonar https://docs.sonarqube.org/latest/setup/get-started-2-minutes/
- Get started with Maven https://maven.apache.org/guides/getting-started/
- Git installazione https://git-scm.com/book/it/v1/Per-Iniziare-Installare-Git
- GitHub Education https://education.github.com/pack
- Git cheat sheet https://services.github.com/on-demand/downloads/github-git-cheat-sheet.pdf
- GitHub Training https://lab.github.com/githubtraining