Git and GitHub

Version Control Systems and Teamwork

SoftUni Team

Technical Trainers

Software University

SoftUni

Software University

https://softuni.bg

Table of Contents

1. Software Configuration Management and **Source Control Systems**
   - Vocabulary: **Clone** a Repo, **Commit** a Changeset, **Push** the Changes, **Pull** Changes, **Merge** Changes
2. Introduction to **Git**
   - Working with Git, Git Bash, and GitHub Desktop
3. Introduction to **GitHub**
   - Create a Repo, Clone, Commit, Push, Conflicts

Have a Question?

Software University

sli.do

#qa-fund

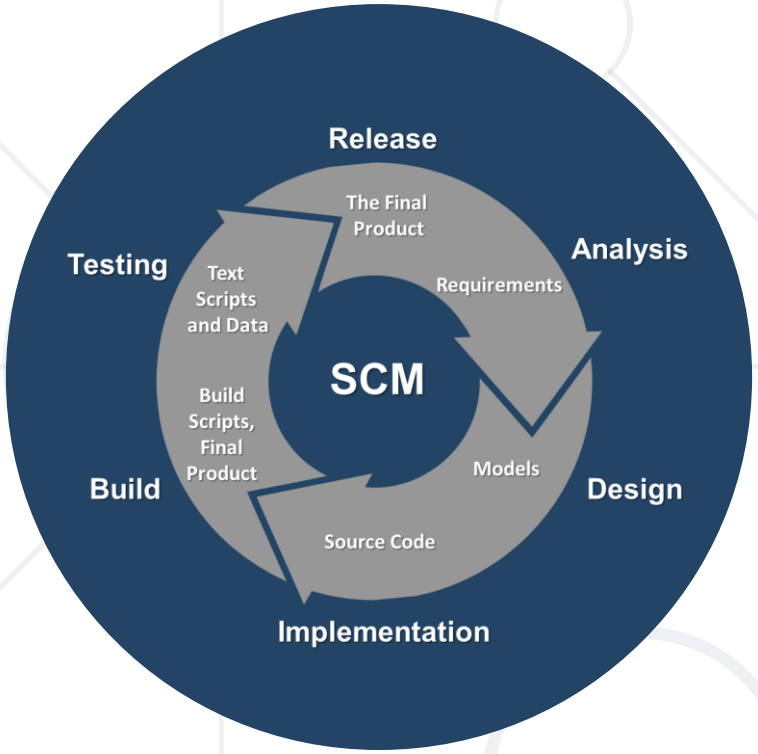
Source Control Systems: Lesson Summary

Software University

- **Source control systems** keep the source code (+ other project assets) in a shared **repository**
  - Developers can **clone** a repository, **pull** the latest version, **commit** & **push** local changes, view the change logs, etc.

- **Git** is the most popular source control system
  - Other version control systems: SVN, TFS, Perforce

- **GitHub** is the #1 site for Git project hosting
  - Git hosting + issue tracker + review system + project tracker + build system



Software Configuration Management

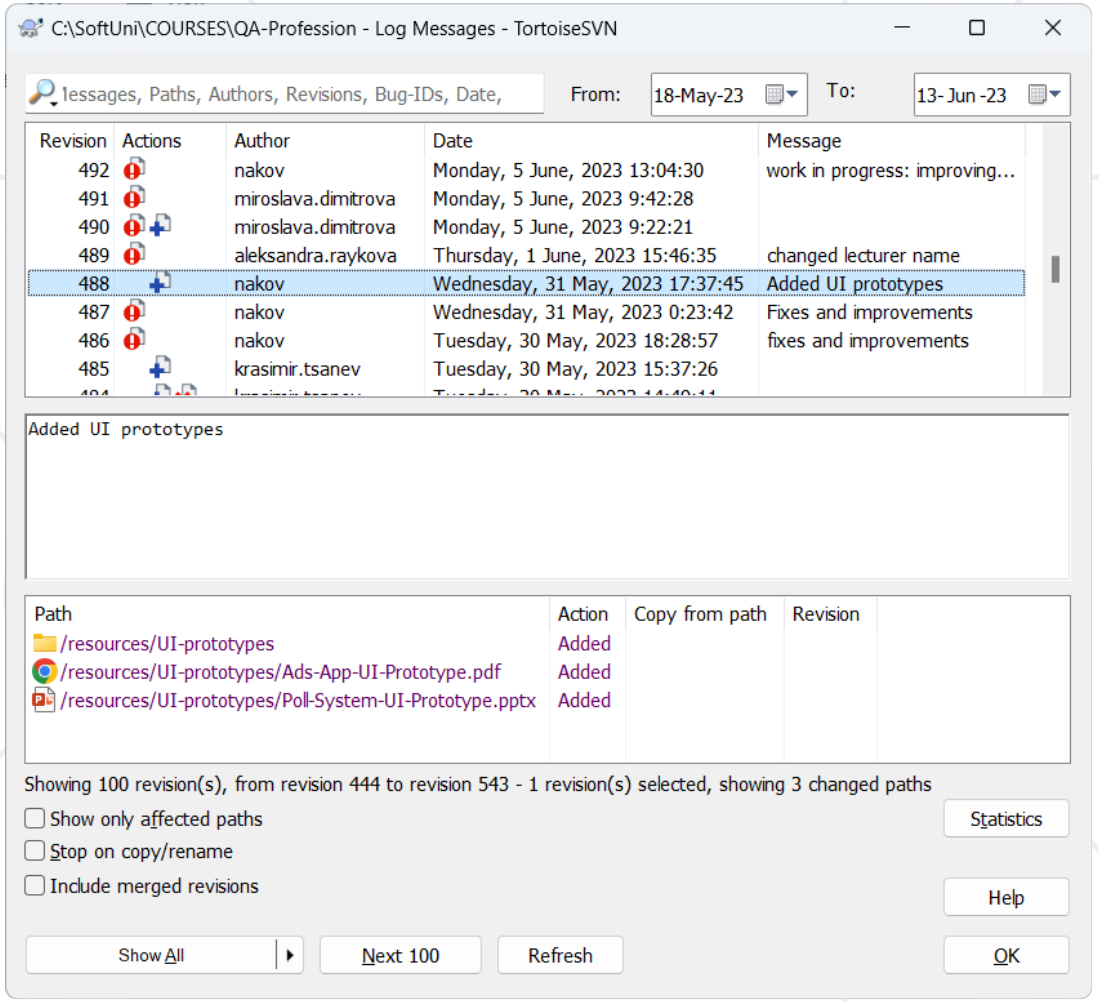
Working on Shared Code: Source Control Systems

Software Configuration Management

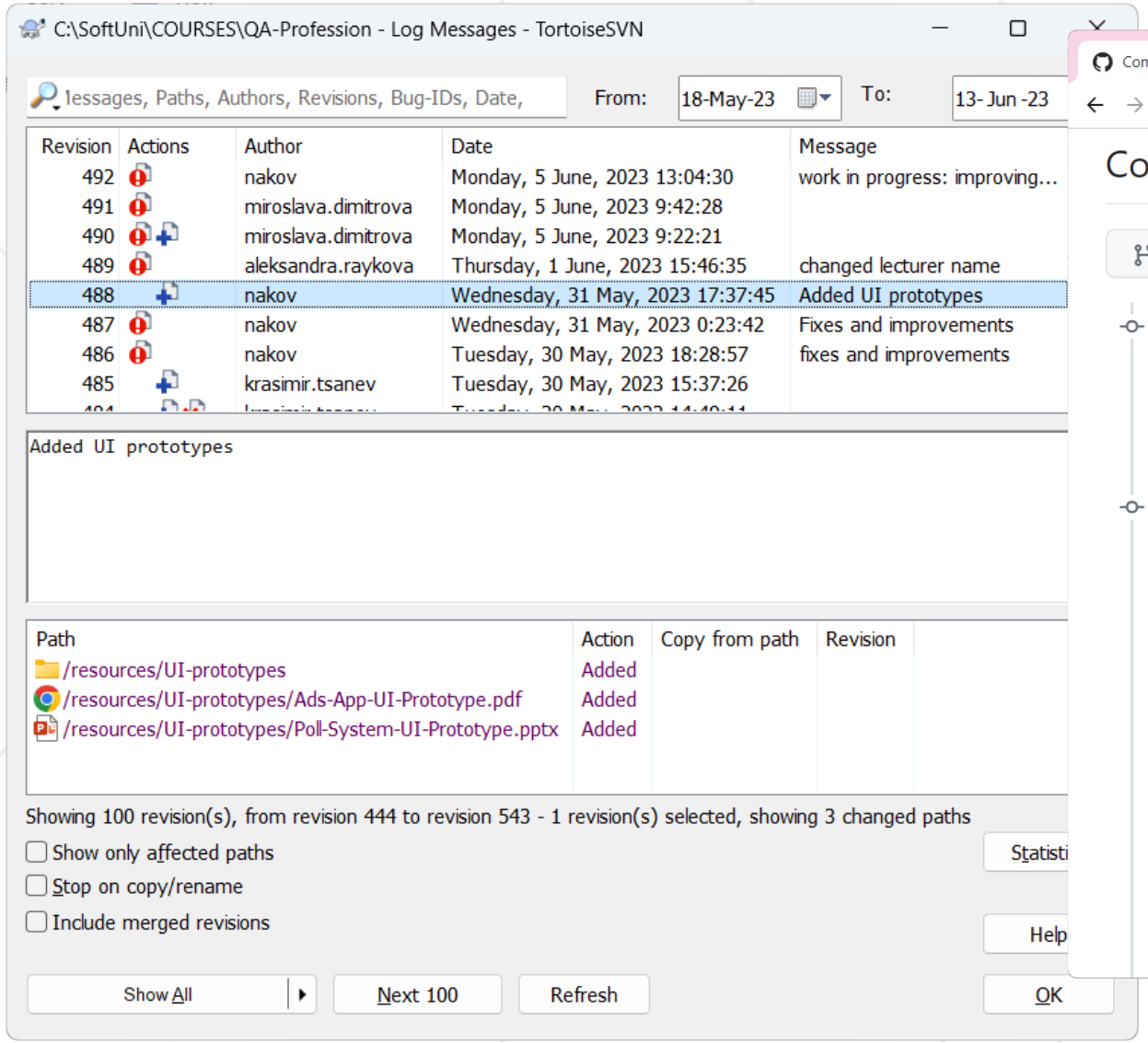
- **Version control** ≈ Software Configuration Management (SCM) ≈ **source control system**
  - A software engineering discipline
  - Consists of techniques, practices and tools for working on **shared source code** and files
  - Mechanisms for **changes tracking** and **conflict resolution**
  - Defines the process of **change management**
  - Keeps track of **what is changing** in the project over time
  - Solves **conflicts** in the changes

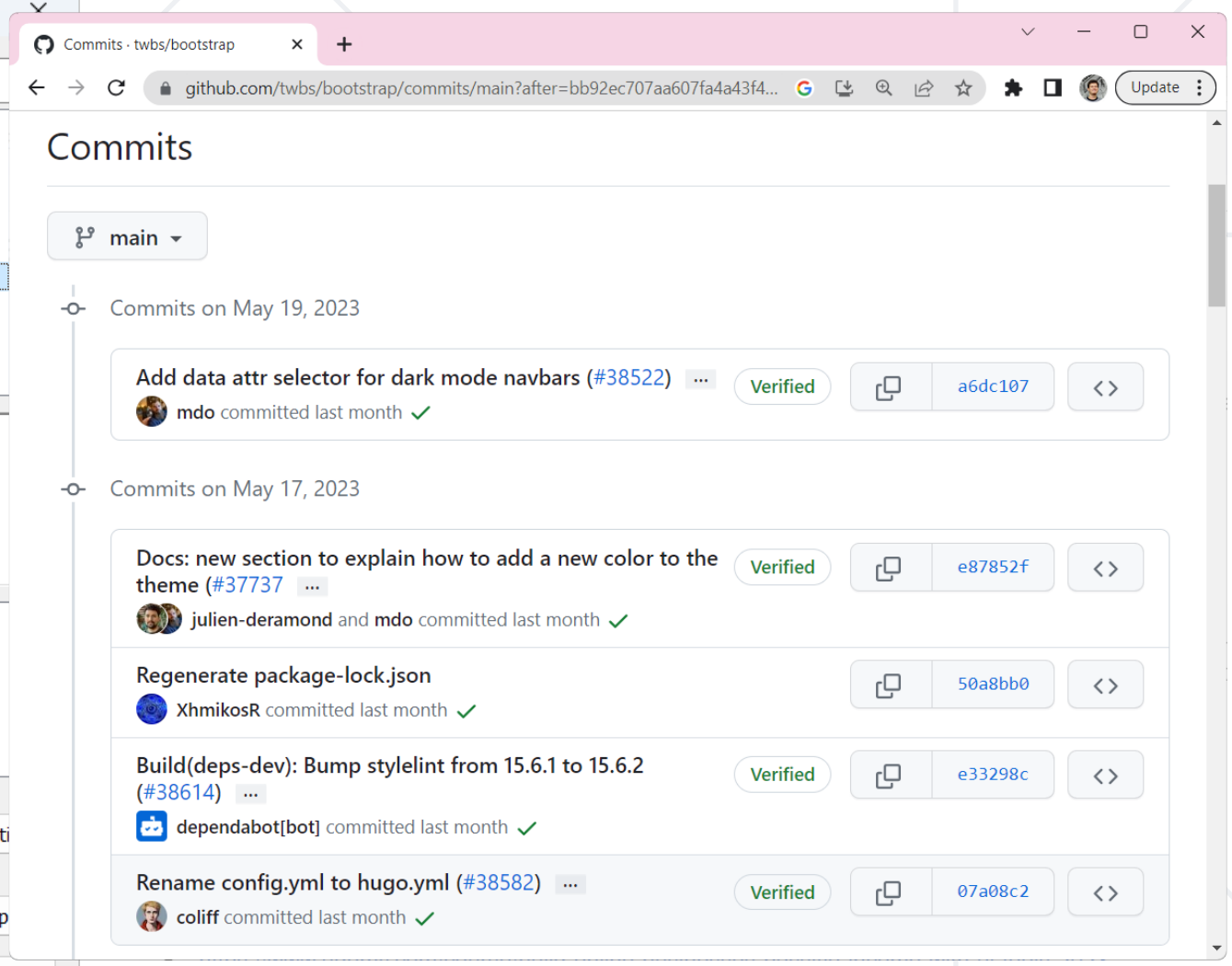
Change Log

- **Version control systems** keep their own **change log** (version history):
  - Who?
  - When?
  - Why?
  - What was changed?
- Old versions can be **reverted back**



Change Log – Examples





https://github.com/twbs/bootstrap/commits

Vocabulary: Repository (Repo)

- **Repo** holds the project in a remote server

Remote

Local

Vocabulary: Clone

- **Clone** == download a **local copy** of the remote project

Vocabulary: Commit

- **Commit** == saves a set of changes locally

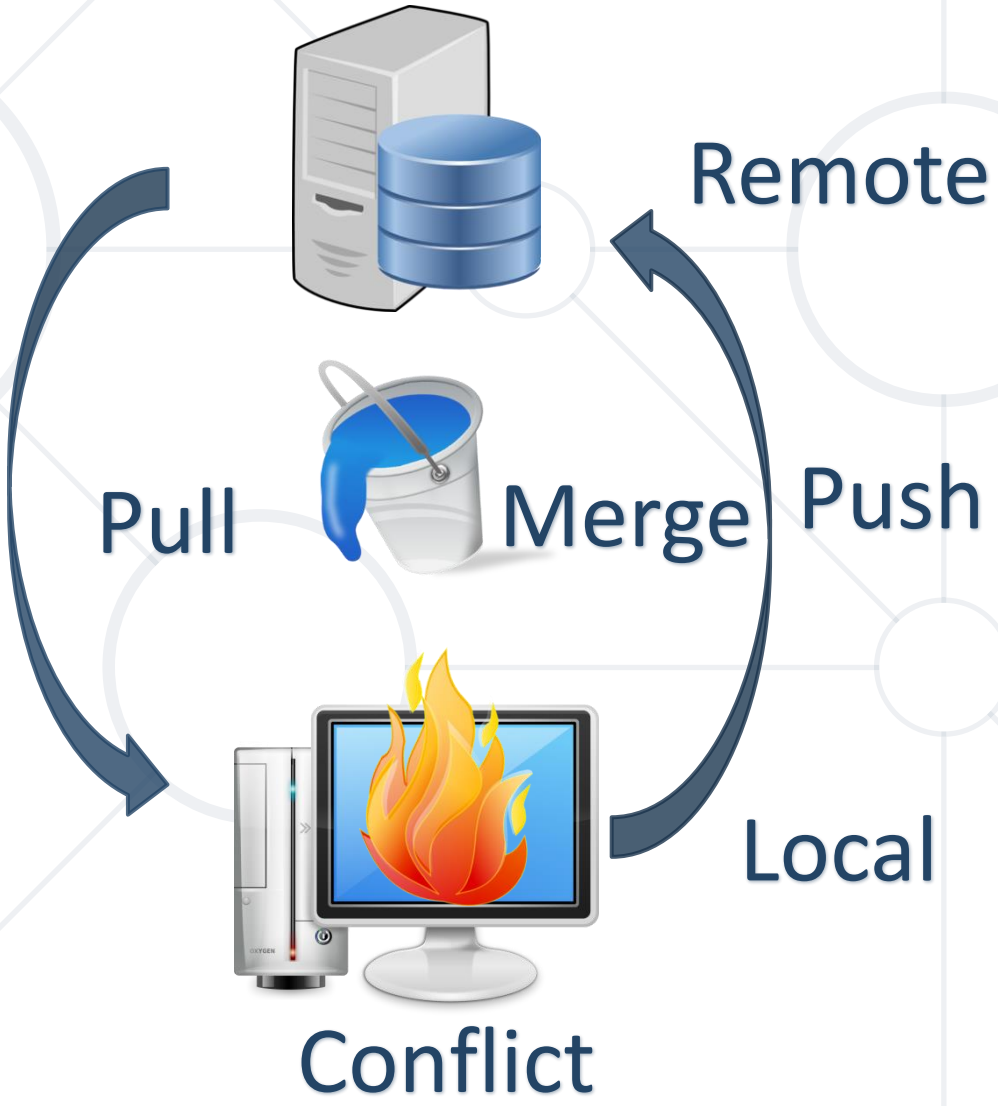
Remote

Commit

Local

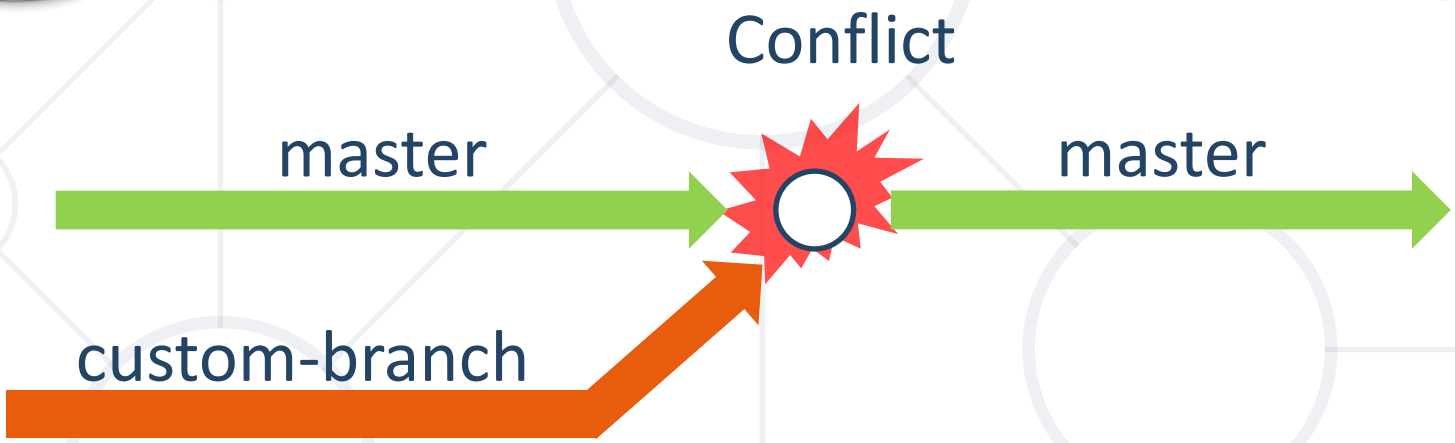
Vocabulary: Sync (Pull / Push)

- **Pull** – **take** and **merge** the changes from the Remote
- **Push** – **send** local changes to the Remote

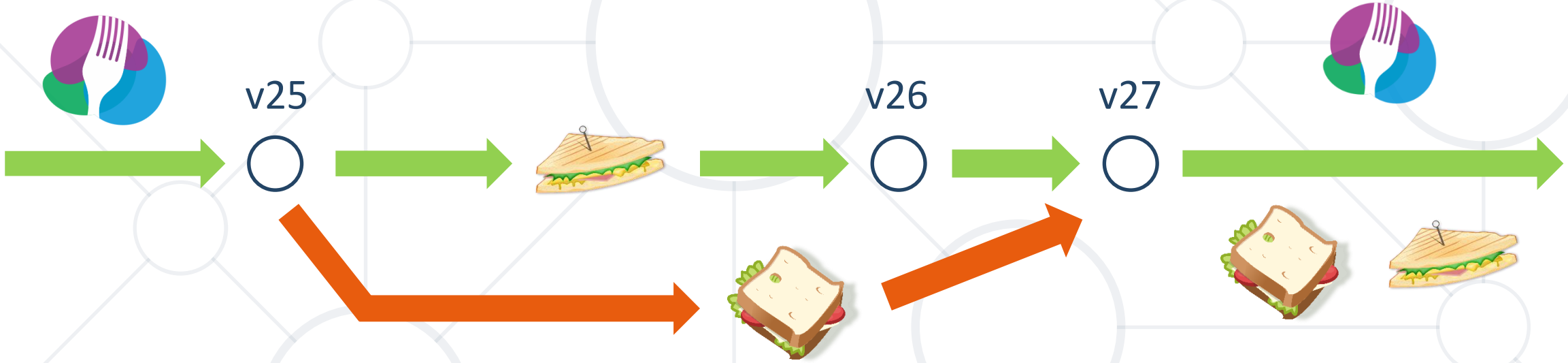


Vocabulary: Branch

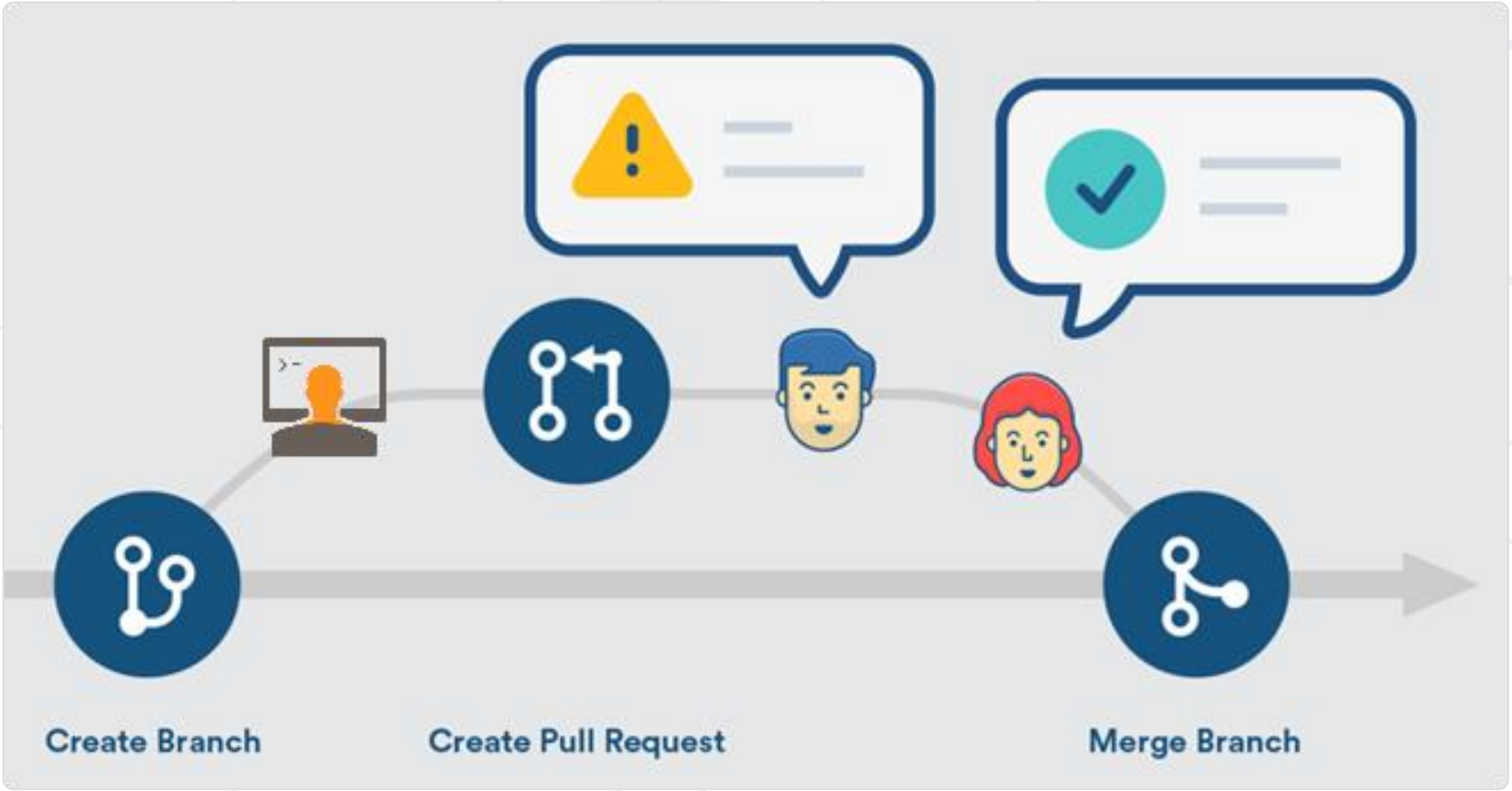
Vocabulary: Merge Branches



Example: Branches



Pull Requests: The Code Review Process



Git

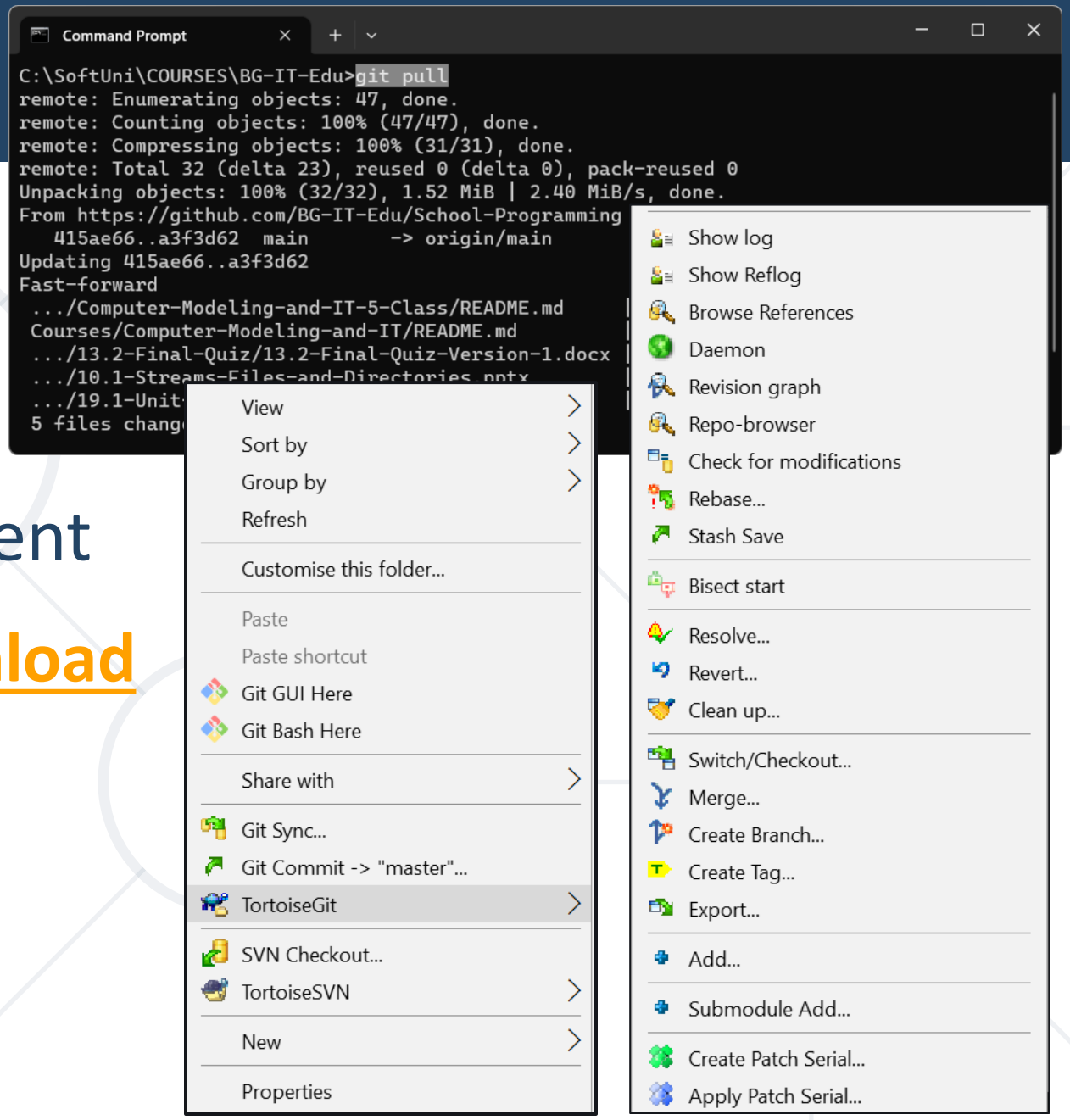
World's #1 Source Control System

What is Git?

- **Git** == distributed **source-control system**
  - The most popular in the world
  - Free, open-source software
- Works with **local** and **remote** repositories
- **Git bash** – command line interface for Git
- Runs on Linux, macOS and Windows (**msysGit**)
  - https://git-scm.com

Using Git

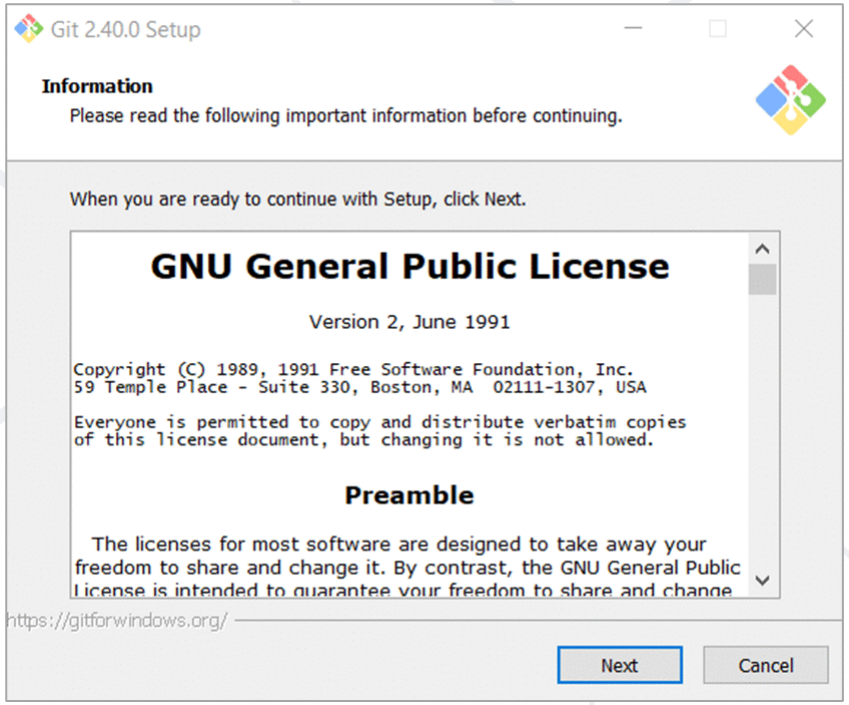
- Console-based Git client
  - git, Git Bash
- TortoiseGit – Windows GUI client
  - https://tortoisegit.org/download
- GitHub Desktop client
  - https://desktop.github.com



Installing Git

- Git installation on Windows: **Git for Windows** (msysGit)
  - https://git-scm.com/downloads
  - Options to select (selected by default)
    - "Use Git Bash Only"
    - "Checkout Windows-style, Commit Unix-style Endings"
- Git installation on **Linux**:

```
sudo apt-get install git
```



Basic Git Commands (1)

- **Cloning** an existing Git repository

```
git clone [remote url]
```

- **Fetch** and **merge** the latest changes from the remote repository

```
git pull
```

- Preparing (**adding** / selecting) files for a commit

```
git add [filename] ("git add ." adds everything)
```

- **Committing** to the local repository

```
git commit -m "[your message here]"
```

Basic Git Commands (2)

- Check the **status** of your local repository (see the local changes)

```
git status
```

- Creating a **new** local repository (in the current directory)

```
git init
```

- Creating a **remote** (assign a short name for remote Git URL)

```
git remote add [remote name] [remote url]
```

- **Pushing** to a remote (send changes to the remote repository)

```
git push [remote name] [local name]
```

GitHub – Example

- **Clone** a repository from GitHub

```
git clone https://github.com/SoftUni/playground
```

- Modify local files

```
notepad README.md
```

- Commit changes (local)

```
git add . & git commit -m "Added something"
```

- Push the changes to GitHub

```
git push
```

Git: Live Demo

Checkout → Modify → Commit → Push

GitHub – Teamwork Example

- **Alice** (terminal)

```
git clone https://github.com/SoftUni/playground
```

```
notepad README.md
```

```
git add . & git commit -m "Added something"
```

```
git push
```

- **Bob** (TortoiseSVN)

```
git clone https://...
```

```
notepad README.md
```

```
git add & git commit
```

```
git push → conflict
```

```
git pull → merge
```

```
git push → success
```

Git: Live Demo

Teamwork and Resolve a Conflict

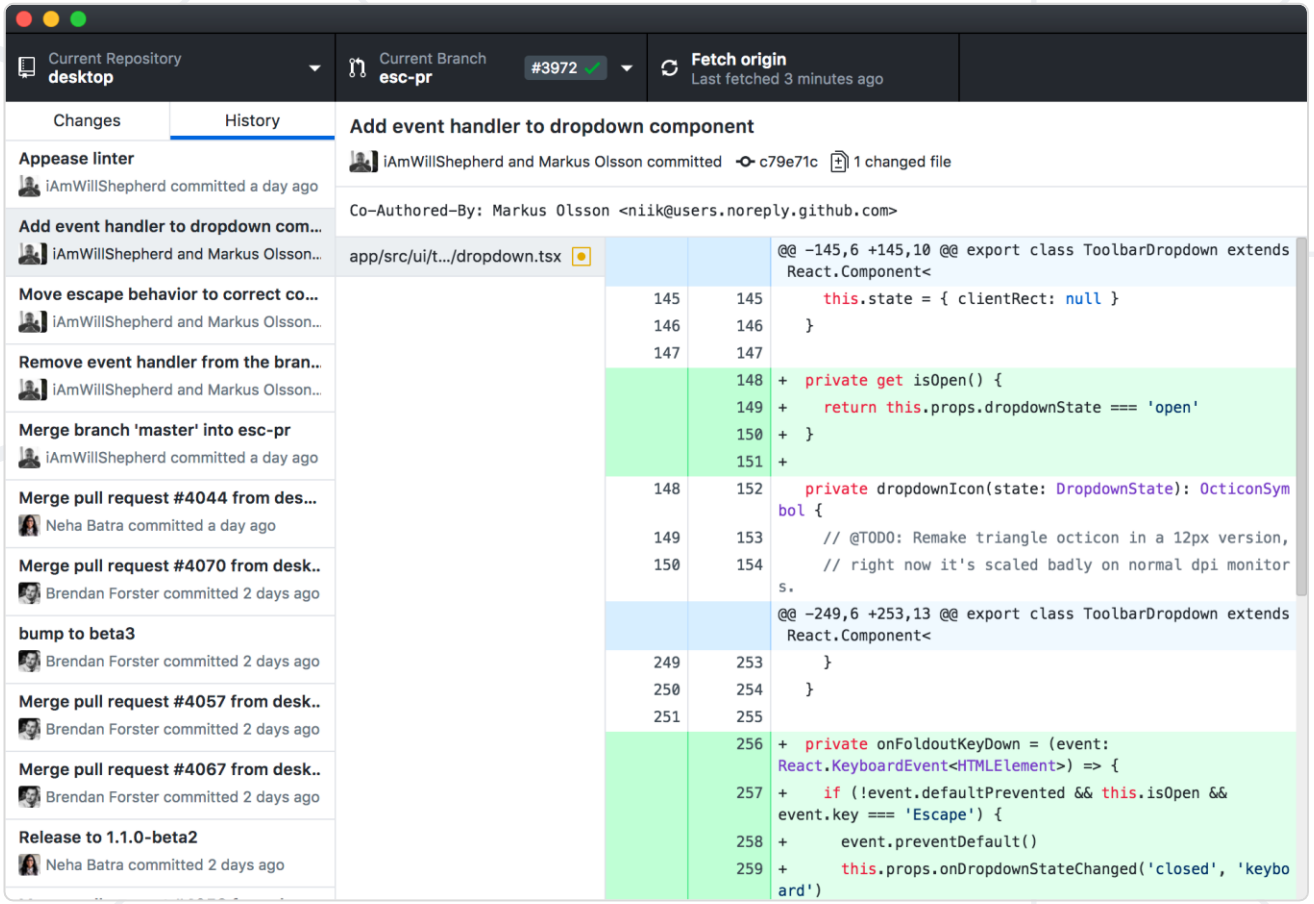
GitHub Desktop

Development Workflow Simplified

What is GitHub Desktop?

Software University

- Simple **Desktop GUI client** app by GitHub
- Translates Git operations from command-line into a **user-friendly UI**
- Functions: **manage repositories**, **branches**, **commits**, and **pull requests** without opening a terminal



GitHub Desktop's Key Features

- **Repository** management
  - Create, clone, pull
- **Change** management
  - Add, commit, push
  - Conflict resolution
- **Branch** management
  - Create, switch, merge
  - Pull requests and merging

https://desktop.github.com

GitHub

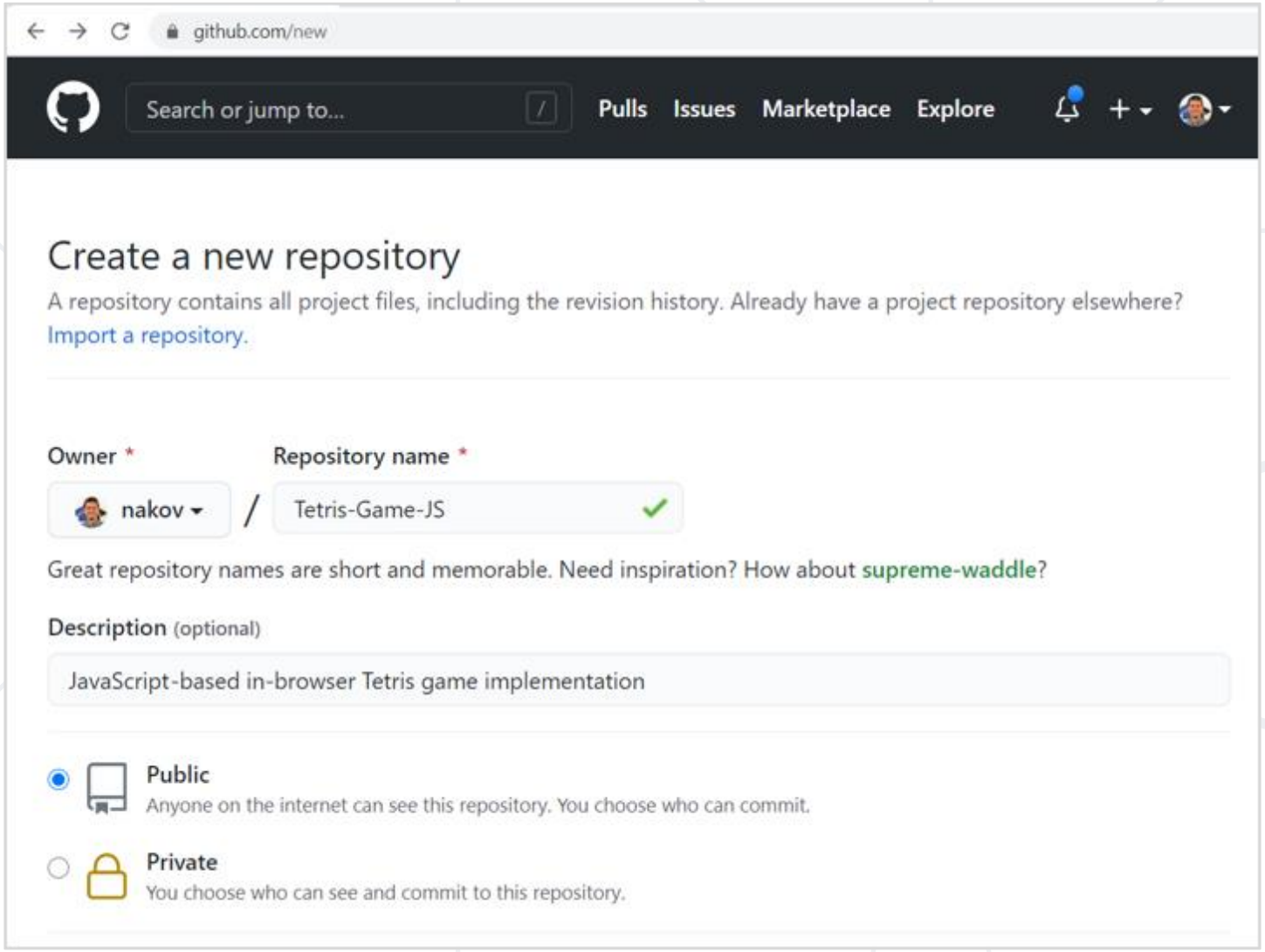
The World's #1 Developer Collaboration Cloud

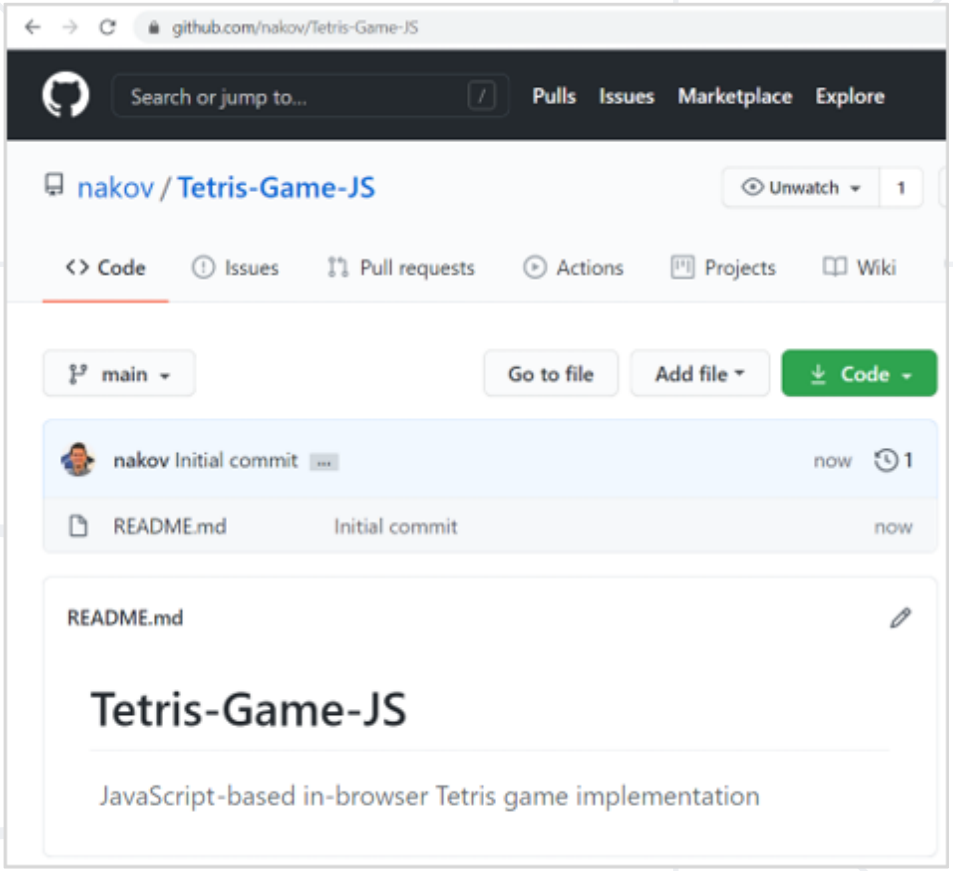
What is GitHub?

- **GitHub** is the world's #1 source code hosting site
  - **Free** for open-source projects
  - **Paid plans** for private repositories
- GitHub provides:
  - Git **source code** repository
  - **Issue tracker** (bug tracker)
  - **Project board** (Kanban style)
  - **Wiki** pages (documentation)
  - **Code reviews** (pull requests)
  - **Build system** (actions)
  - **Site hosting** (pages)
  - **Online IDE** (spaces)

Creating a GitHub Repository

Software University





GitHub – Example

- **Clone** a repository from GitHub

```
git clone https://github.com/SoftUni/playground
```

- **Modify** local files

```
notepad README.md
```

- **Commit** changes (local)

```
git add . & git commit -m "Added something"
```

- **Push** the changes to GitHub

```
git push
```

GitHub: Live Demo

Create Repo → Edit Files → Checkout → Push

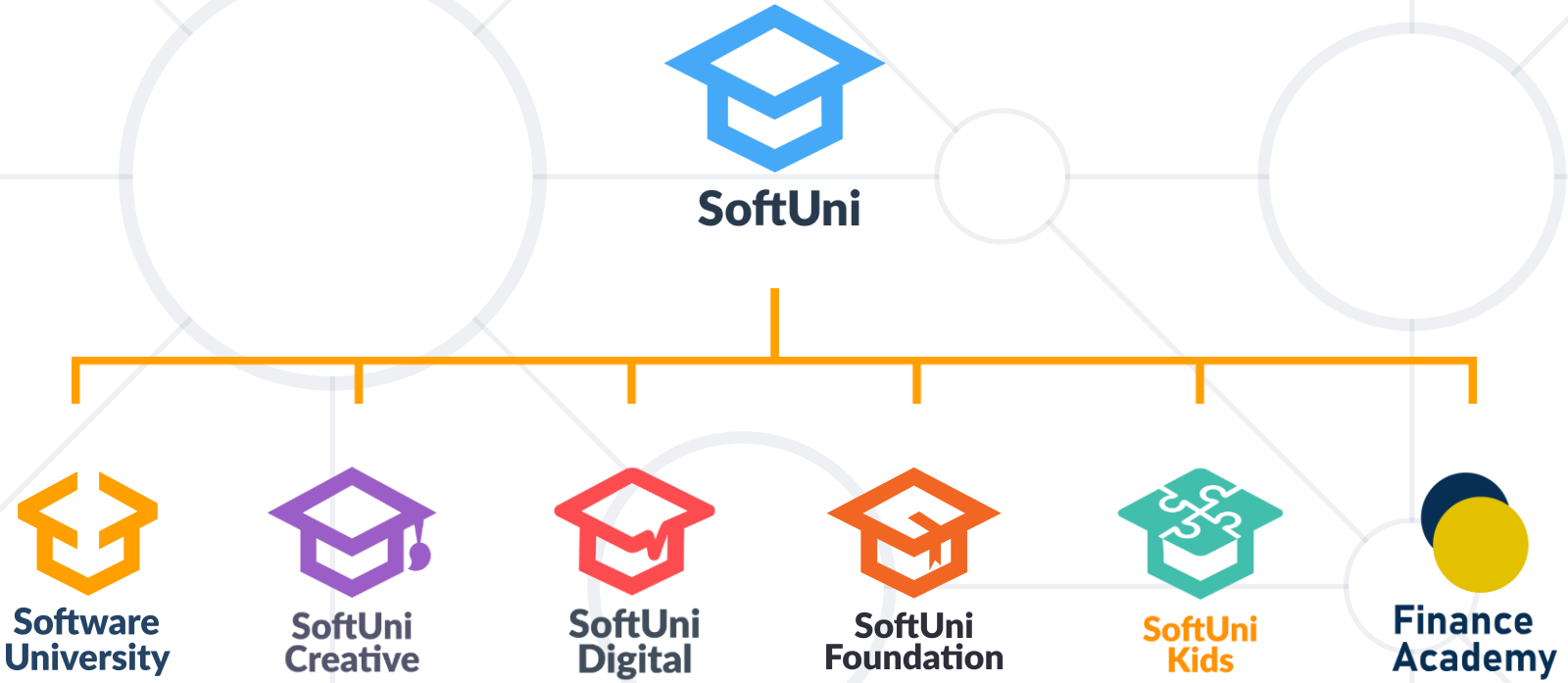
Live Exercises

Creating a Repo, Cloning a Repo, Commit and Push Changes, Resolve Conflicts, Team Interactions

Summary

- Use **version control systems** to work in a team
  - Keep the **shared code** in a central repository
  - Handle merge **conflicts** with ease
- Important **Git** commands:
  - **`clone`**, **`pull`**, **`add`**, **`commit`**, **`push`**
- **Git** clients: **TortoiseGit**, **GitHub Desktop**
- **GitHub** == the world's most used software project hosting platform
  - Git repository, issue tracker, Kanban board, Wiki

Questions?



© SoftUni – https://about.softuni.bg. Copyrighted document. Unauthorized copy, reproduction or use is not permitted.

SoftUni Diamond Partners

Software University

SUPER HOSTING .BG

Coca-Cola HBC Bulgaria

POKERSTARS
POKER | CASINO | SPORTS
a Flutter International brand

INDEAVR
Serving the high achievers

AMBITIONED

DRAFT KINGS

SOFTWARE GROUP

createX

Postbank
Решения за твоето утре

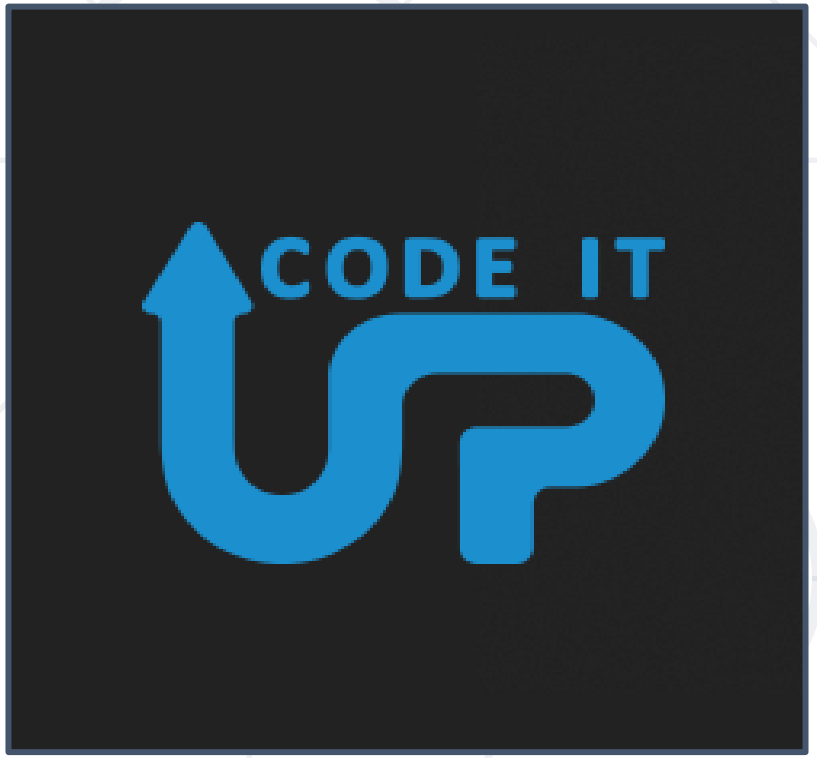
BOSCH

DXC TECHNOLOGY

SmartIT

Educational Partners

Software University



Trainings @ Software University (SoftUni)

- Software University – High-Quality Education, Profession and Job for Software Developers
  - softuni.bg
  - Software University Foundation
  - softuni.foundation
- Software University @ Facebook
  - facebook.com/SoftwareUniversity
- Software University Forums
  - forum.softuni.bg

License

- This course (slides, examples, demos, exercises, homework, documents, videos and other assets) is **copyrighted content**
- Unauthorized copy, reproduction or use is illegal
- © SoftUni – https://softuni.org
- © Software University – https://softuni.bg