



Școala  
informală  
de IT



Școala  
informală  
de IT

# Agenda

- React Course Structure
- Git & GitHub
- Visual Studio Code
- Shortcuts and useful extensions
- Chrome Developer Tools
- npm





# Course structure

# Course structure

- Tools. Javascript Refresher.
- First React Project. React Components
- Props. State. Events
- Styling React Components
- Debugging React Components
- Fragments & Refs
- Create backend using Node JS, Express and MongoDB
- React Router
- Redux basics
- Final Exam. Final Project.





git

+

GitHub

# What is GIT?

- **Distributed VCS(Version Control System)**
- **Coordinates work in teams(multiple developers)**
- **Tracking history(who made what changes and when)**
- **Possibility to go to any version from history anytime**
- **Local & Remote repositories(GitHub, Bitbucket)**



1. It is a software

2. It is installed locally on the system

3. It is a command line tool

4. It is a tool to manage different versions of edits, made to files in a git repository

5. It provides functionalities like Version Control System Source Code Management

1. It is a service

2. It is hosted on Web

3. It provides a graphical interface

4. It is a space to upload a copy of the **Git** repository

5. It provides functionalities of Git like VCS, Source Code Management as well as adding few of its own features

# GIT/GitHub initial setup

- **git-scm.com**
- **Download Git for Windows/Mac**
- **Install with almost default options**
- **Open gitbash on Windows or terminal on MAC**
- **Type** `git --version`
- **Type** `git help` **to check the basic commands**
- `git help <command>` **to see a specific command details**
- **Install gitextensions** → <http://gitextensions.github.io/>



# Git Initial Configuration



Open gitbash:

1. `git config --global user.name "Andrei Popescu"`
2. `git config --global user.email "andrei.popescu@gmail.com"`
3. `git config --global --list`

# Basic git bash(linux) commands

|              |   |
|--------------|---|
| <b>cd ..</b> | move up one level in the directory tree structure |
|--------------|---|

|           |                            |
|-----------|----------------------------|
| <b>cd</b> | change directory to \$HOME |
|-----------|----------------------------|

|                             |                               |
|-----------------------------|-------------------------------|
| <b>cd /chosen/directory</b> | change to specified directory |
|-----------------------------|-------------------------------|

|                          |                   |
|--------------------------|-------------------|
| <b>touch [file_name]</b> | create a new file |
|--------------------------|-------------------|

|           |                         |
|-----------|-------------------------|
| <b>ls</b> | list files in directory |
|-----------|-------------------------|

|              |                                  |
|--------------|----------------------------------|
| <b>ls -a</b> | list all files, including hidden |
|--------------|----------------------------------|

|            |   |
|------------|---|
| <b>pwd</b> | show the directory currently working in |
|------------|---|

|                          |                        |
|--------------------------|------------------------|
| <b>mkdir [directory]</b> | create a new directory |
|--------------------------|------------------------|

|                       |               |
|-----------------------|---------------|
| <b>rm [file_name]</b> | remove a file |
|-----------------------|---------------|

|                               |                                |
|-------------------------------|--------------------------------|
| <b>rm -r [directory_name]</b> | remove a directory recursively |
|-------------------------------|--------------------------------|

|                                |   |
|--------------------------------|---|
| <b>rm -rf [directory_name]</b> | remove a directory recursively without requiring confirmation |
|--------------------------------|---|

|                                     |  |
|-------------------------------------|--|
| <b>cp [file_name1] [file_name2]</b> | copy the contents of the first file to the second file |
|-------------------------------------|--|

|  |  |
|--|--|
| <b>cp -r [directory_name1] [directory_name2]</b> | recursively copy the contents of the first directory into the second directory |
|--|--|

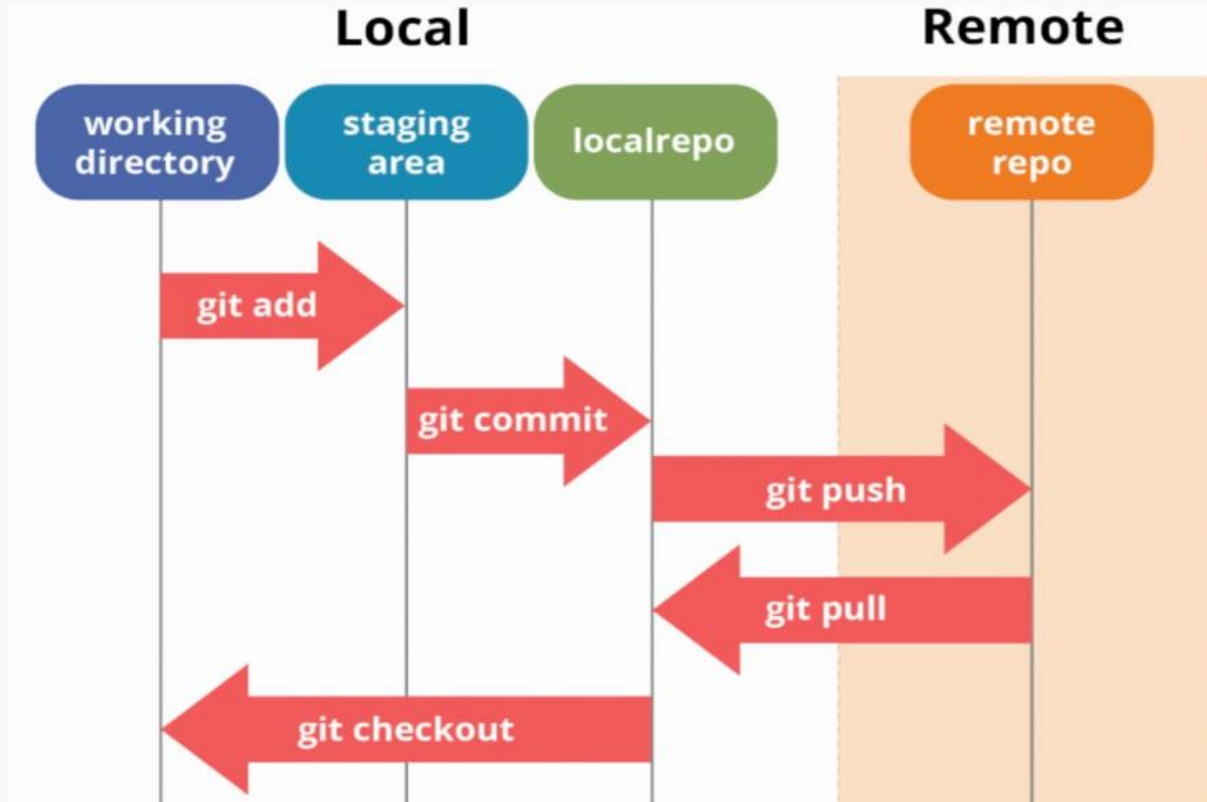
|                                     |                                 |
|-------------------------------------|---------------------------------|
| <b>mv [file_name1] [file_name2]</b> | rename file_name1 to file_name2 |
|-------------------------------------|---------------------------------|



# GitHub Workflow(Setup the account)

- **Create github account**
- **Create new repository**
- **Use** `git clone your_repo.git`
- **Create a new folder Demo and inside a new file**  
**first\_file.txt**
- **Commit and push** 😊

# GitHub Workflow



# Useful git commands

| Git task                          | Notes  | Git commands  |
|-----------------------------------|--|---|
| Status                            | List the files you've changed and those you still need to add or commit:                                       | <code>git status</code>   |
| Add files                         | Add one or more files to staging (index):  | <code>git add &lt;filename&gt;</code><br><code>git add .</code> |
| Commit                            | Commit changes to head (but not yet to the remote repository):   | <code>git commit -m</code><br><code>"Commit message"</code>     |
|                                   | Commit any files you've added with <code>git add</code> , and also commit any files you've changed since then: | <code>git commit -a</code>                                      |
| Push                              | Send changes to the master branch of your remote repository:   | <code>git push origin</code><br><code>master</code>             |
| Update from the remote repository | Fetch and merge changes on the remote server to your working directory:  | <code>git pull</code>   |
| Branches                          | Create a new branch and switch to it:  | <code>git checkout -b</code><br><code>&lt;branchname&gt;</code> |





# Workshop

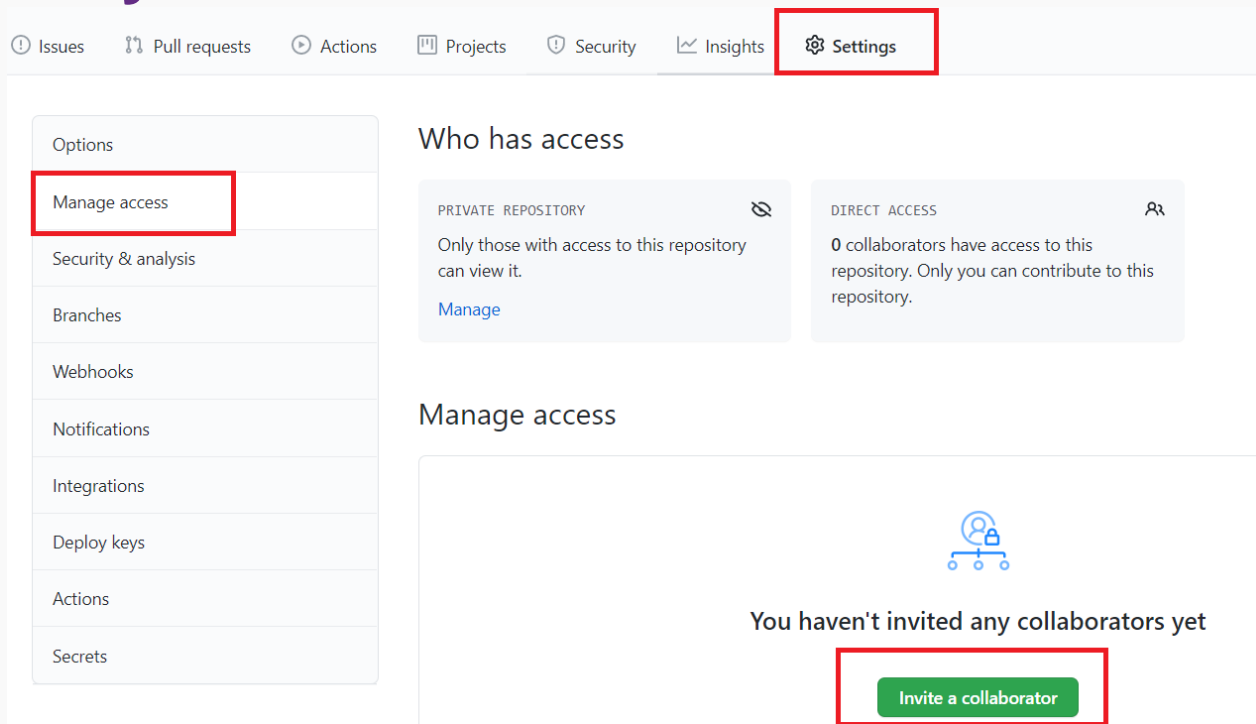


# Workshop 1 – Be organized!

- Create a new repository on GitHub for managing all the resources from this entire course: name it ReactDev\_YourName
- Clone this new **repository** on your personal PC/Laptop
- In the main folder create the following 3 folders: In each folder create a new txt file – random/dummy content
- Commit this file with a specific message and push it on GitHub



# Workshop 1 – Add the mentor as a reviewer to your Repository



The screenshot displays the GitHub repository settings interface. At the top, a navigation bar includes links for Issues, Pull requests, Actions, Projects, Security, Insights, and Settings. The 'Settings' link is highlighted with a red box. On the left, a sidebar lists various settings categories: Options, Manage access (highlighted with a red box), Security & analysis, Branches, Webhooks, Notifications, Integrations, Deploy keys, Actions, and Secrets. The main content area is titled 'Who has access' and shows two access types: 'PRIVATE REPOSITORY' (locked) and 'DIRECT ACCESS' (unlocked). The 'DIRECT ACCESS' section indicates that 0 collaborators have access. Below this, the 'Manage access' section is shown, featuring a lock icon and the text 'You haven't invited any collaborators yet'. A green button labeled 'Invite a collaborator' is highlighted with a red box.

Issues Pull requests Actions Projects Security Insights **Settings**

Options

**Manage access**

Security & analysis

Branches

Webhooks

Notifications


Integrations

Deploy keys

Actions


Secrets

## Who has access

**PRIVATE REPOSITORY** 


Only those with access to this repository can view it.

[Manage](#)

**DIRECT ACCESS** 

0 collaborators have access to this repository. Only you can contribute to this repository.

## Manage access



You haven't invited any collaborators yet

**Invite a collaborator**







# Visual Studio Code



# Visual Studio Code Instalation

- [code.visualstudio.com](https://code.visualstudio.com)
- Open your Git project folder(check for .git folder)
- **Ctrl + ,** - settings → features → terminal → edit in settings.json
- Add the following line:  
`"terminal.integrated.shell.windows": "yourPah\\Git\\bin\\bash.exe",`
- Use **Ctrl + `** to open/close terminal in VS Code

# Visual Studio Code Shortcuts

- Bookmark the following link on your browser:
- <https://code.visualstudio.com/shortcuts/keyboard-shortcuts-windows.pdf>
- To be productive: use the mouse only when the keyboard cannot be used
- Learn 2 shortcuts every week, share with the colleagues when you find a very useful shortcut

# Visual Studio Code Extensions

- Live Server
- Prettier
- JavaScript ES6 Code Snippets
- Turbo Console Log
- Bracket Pair Colorizer
- Git Lens
- ES7 React/Redux/GraphQL/React-Native snippets

# NPM – Node Package Manager





- **Node Package Manager**
- **Node js must be installed first**
- <https://nodejs.org/en/download/>
- NPM - registry where it contains all the different JS libraries(JQuery, Bootstrap)
  - `npm --v`
  - `npm install --g create-react-app`
  - `npx create-react-app test`

<https://medium.com/swlh/npm-in-less-than-10-minutes-6b321d566271>

# Resources

- <https://developers.google.com/web/tools/chrome-devtools>
- <https://guides.github.com/introduction/git-handbook/>
- <https://gitimmersion.com/>
- <https://code.visualstudio.com/shortcuts/keyboard-shortcuts-windows.pdf>
- <https://support.microsoft.com/en-us/help/12445/windows-keyboard-shortcuts>

