Git 101

[Version control for artists and designers]

@MiglenaMinkova

Version control systems (VCSs)

---Local----

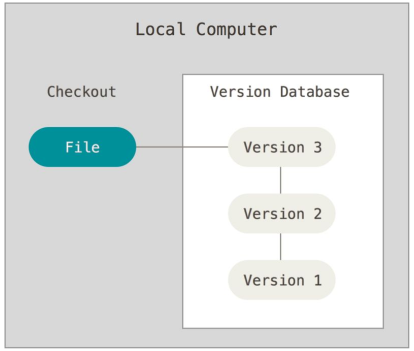
SCCS, RCS

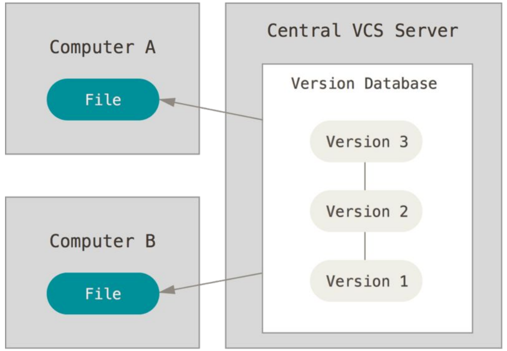
---Centralized---

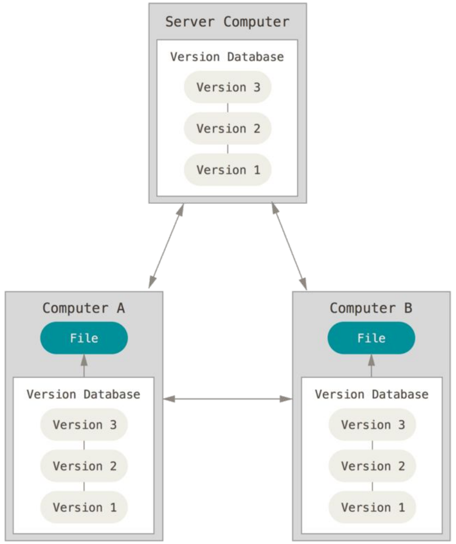
CVS, SVN, Perforce

---Distributed---

Git, Bazaar, ArX, BitKeeper, Mercurial, Monotone.







What is git ?

git - the stupid content tracker*

- Free and open source distributed version control system
- Fast, lightweight and with safeguards against data corruption
- Extremely popular choice for open source projects = industry standard
- Used for code but open to all knowledge workers
- Large community, abundant documentation & training materials
- Tracks changes to directories and any type of file in them
- Various GUI clients

Git basic concepts

---Workflow---

example.txt

-Make new file in working directory

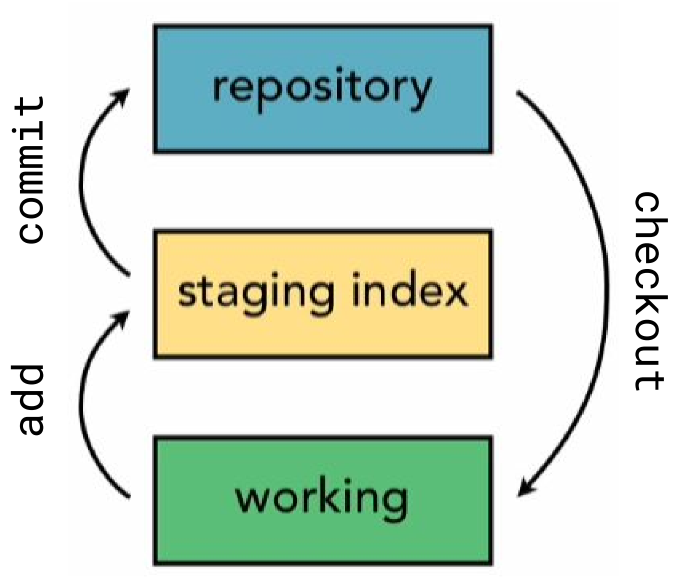
git add example.txt

- Add to staging index

git commit -m “commit message”

- Commit to repository
- Repeat for further edits

---Three tree architecture---



Git basic concepts

---SHA-1---

Commit ID:
4ab32805bc4850ffa83741baa0ed8
3d958eef59d

- Same files with same changes have the same checksum
- Preserves data integrity

Author
Parent commit
Commit message

---HEAD pointer---

- Reference variable pointing to the tip of the working branch of the repository or to the **last committed change to the repository.**

Git 101 -- Practice

---

1. **Install & configure**
2. **Create a local repository**
3. **Create, add and commit**
4. **Roll back to previous version**
5. **Compare and view changes**
6. **Branch**
7. **Merge & delete branches**
8. **Resolve merge conflicts**

[Prerequisite] Do not be afraid of the command line!

Git 101

---

1. Install & configure

- Download from https://git-scm.com/downloads
- Open CLI
- Check all went ok

```
$ git --version
```

- Configure Git

```
$ git config --global user.name "Your Name"
$ git config --global user.email yourName@example.com
```

- Helpful git commands

```
$ git help  $ git status $ git log  $ gitk  $ gittutorial
```

Git 101

---

2. Create a local repository

- Make a new project folder on your computer
- Navigate to it

```
$ cd/folder1/subfolder/newProject
```

- Put it under version control (initialise)

```
$ git init
```

- Helpful CLI commands

```
pwd | cmd     cd [/folder/subfolder]     ls
```

Git 101

----

3. Create, add and commit

- Create an image and plain text file and save them in your working directory.
- Add image & text file to staging index

  ```
  $ git add text.txt image.png | $ git add .
  ```

  - Check status

  ```
  $ git status
  ```

- Commit both files to repository

  ```
  $ git commit -m “added and image and text files”
  ```

- Edit and repeat!

Git 101

4. Roll back to previous version

- View the commits history

```
$ git log
```

- Search all commit messages across branches

```
$ git log --all --grep=”search-term”
```

- Chose a previous version to roll back to and copy commit id

```
$ git checkout 3430af9434bc84
```

- “Detached HEAD” state
- Return to current commit

```
$ git checkout master
```

Git 101

----

5. Compare and view changes

- Compare the last 2 committed versions

```
$ git diff HEAD^ HEAD
```

- Compare working directory to staging area

```
$ git diff
```

- Compare staging index to last commit

```
$ git diff --cached
```

- Compare working tree to last commit

```
$ git diff HEAD
```

Git 101

----

6. Branch

- Create a branch **`$ git branch branchName`**
- List all branches **`$ git branch`**
- Switch to a branch **`$ git checkout branchName`**

- Edit documents, add & commit

```
$ git commit -am “commit message”
```

- Change to master branch

```
$ git checkout master
```

Git 101

----

7. Merge & delete branches

- Merge **$ git merge branchName**
- Delete after merging **$ git branch -d branchName**
- Force delete without merging **$ git branch -D branchName**

8. Resolve merge conflicts

- Manual: view changes with git diff, edit and git commit -a to finish the merge.
- Git merge too
- Git checkout --ours/-theirs

Git’s Pros & Cons

+ Sophisticated functionality

+ Good training and documentation

- Command line interface instead of GUI

- Local

- Not binary friendly

What is GitHub ?

Freemium web-based hosting service for git repositories

- Widely used for code but increasing for non-code projects
- Collaboration platform at the epicenter of open source community
- Same git functionality + **additional features, tools and integrations**
- Web-based & desktop **graphical user interface** (GUI)
- **Cloud-hosted** public and private repos

Copy someone else's project into your GitHub account :

- use it as a starting point of your own
- experiment without affecting the original
- contribute to it by proposing changes

More features…

- **Issues :** bug tracking, discussion & miscellaneous
- **Projects :** kanban-style project boards for visually organising and planning
- **Wiki :** documentation & collaborative editing
- **Pages :** free user and project websites hosted on GitHub
- **Insights : Pulse & Graph** present visual history of the repository
- **Social features :** Profiles, Newsfeeds, Follow, Stars & Watching
- **Gist :** sharing parts and snippets

- **Free/Open API :** GraphQL & REST API
- **Developer tools and apps @GitHUb Marketplace**
- **Integrations**

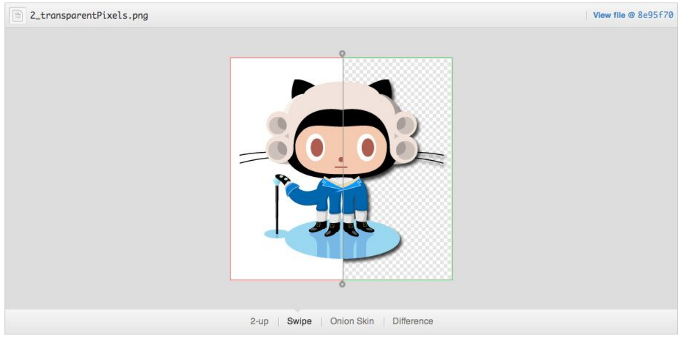
Non-code files

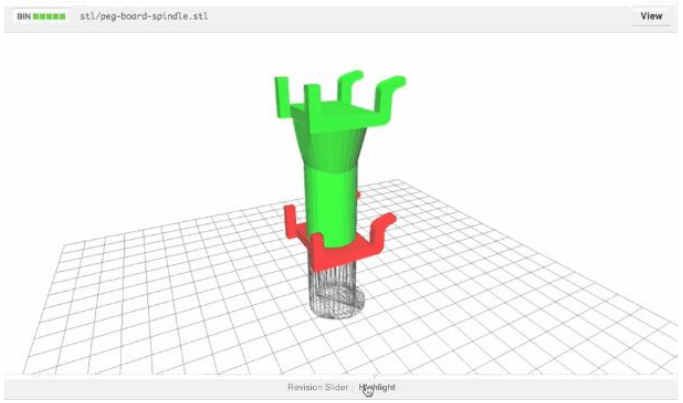
Images

- Formats: PNG, JPG, GIF, PSD, SVG & PDF*
- Display: Dynamic/Static SVG
- Diff views: 2-up, Swipe, Onion skin, Difference

3D files

- Formats: STL < 10MB
- Display: Interactive 3D plane with
  Wireframe, Surface angle, Solid,
- Diff views: Revision slider and Highlights





GitHub 101 -- practice

----

1. **Put your local repo on Github**

[homework] Use the GitHub web GUI to recreate git 101 -- practice

GitHub 101

----

1. Put your local repo on Github -- In practice

- Create a github account https://github.com/
- Create an empty repo on GitHUu that matches the name of your project

- Connect local to remote repo
  - -- Copy the HTTPS address of your GitHub repo
  - -- On your terminal navigate to your project folder
  - -- Set up a remote

```
$ git remote add origin url-of-your-githubrepo.git
```

  - -- Push contents of local repo to remote

```
$ git push origin master
```

GitHub 101

- Tadaaa! See you repo on GitHub