skillup

Git

План

- команди git bash
- npm
- встановлення npm
- підключення до проекту бібліотек
- система контролю версій git
- індексація змін
- коміти
- гілки
- злиття змін
- редагування комітів

Git Bash

ls - Показати файли в цій папці, крім прихованих
cd - Перейти в конкретний каталог
mkdir - Створити папку
mkdir -p {app1, app2} - Створити відразу кілька папок
touch index.html - Створити файл index.html
rm -r test - Видалити папку test з файлами всередині неї
mv app1 /*.* app2 - Перемістити всі файли з папки app1 в папку app2
clear - Очистити консоль

http://rightblog.ru/3274

npm

- це менеджер залежностей(packages) для мови програмування JavaScript
- складається з командного рядка, який також називається npm, а також онлайн-бази даних публічних та приватних залежностей, яка називається реєстром npm

https://docs.npmjs.com/

Встановлення npm

- встановлення node.js https://nodejs.org/en/download/ , перевірити чи встановлено в командному рядку ввести `node --version` `node -v`
- в папці з проектом вводимо `npm init` або `npm init -f` для встановлення параметрів по замовчуванні

Після цих пунктів створюється файл `package.json` через який керуємо залежностями

https://docs.npmjs.com/

package.json

```
{
  "name": "demo",
  "version": "1.0.0",
  "description": "Demo package.json",
  "main": "main.js",
  "dependencies": {
    "mkdirp": "^0.5.1",
    "underscore": "^1.8.3"
  },
  "devDependencies": {},
  "scripts": {
    "test": "echo \"Error: no test specified\" && exit 1"
  },
  "author": "Sitepoint",
  "license": "ISC"
}
```

Встановлення залежностей, бібліотек

- `npm install bootstrap`
- `npm install bootstrap@latest` - встановити останню версію
- `npm install bootstrap@3.3.7` - встановити певну версію
- `npm install gulp --save-dev [-D]`- зберегти в devDependendencies
- `npm install bootstrap --save` - зберегти в dependendencies
- `npm uninstall bootstrap --save-dev` - видалити залежність

```
{
  "name": "my_package",
  "version": "1.0.0",
  "dependencies": {
    "my_dep": "^1.0.0"
  },
  "devDependencies" : {
    "my_test_framework": "^3.1.0"
  }
}
```

Розгортання проекту

- `npm install -` встановлює всі залежності з devDependendencies і dependendencies
- всі залежності встановлюються в папку `node_modules`
- шляхи до залежностей можна писати відносно папки `node_modules`

Git

- система контролю версій
- допомагає розробникам масштабувати проекти
- працювати кільком розробникам одночасно на проекті

Перед початком роботи потрібно встановити git https://git-scm.com/downloads

GitHub веб-сервіс для спільної розробки програмного забезпечення

Info

Туторіали:

- https://githowto.com/uk/setup
- https://codeguida.com/post/453/
- https://github.com/nicothin/web-development/tree/master/git
- https://learn.javascript.ru/screencast/git

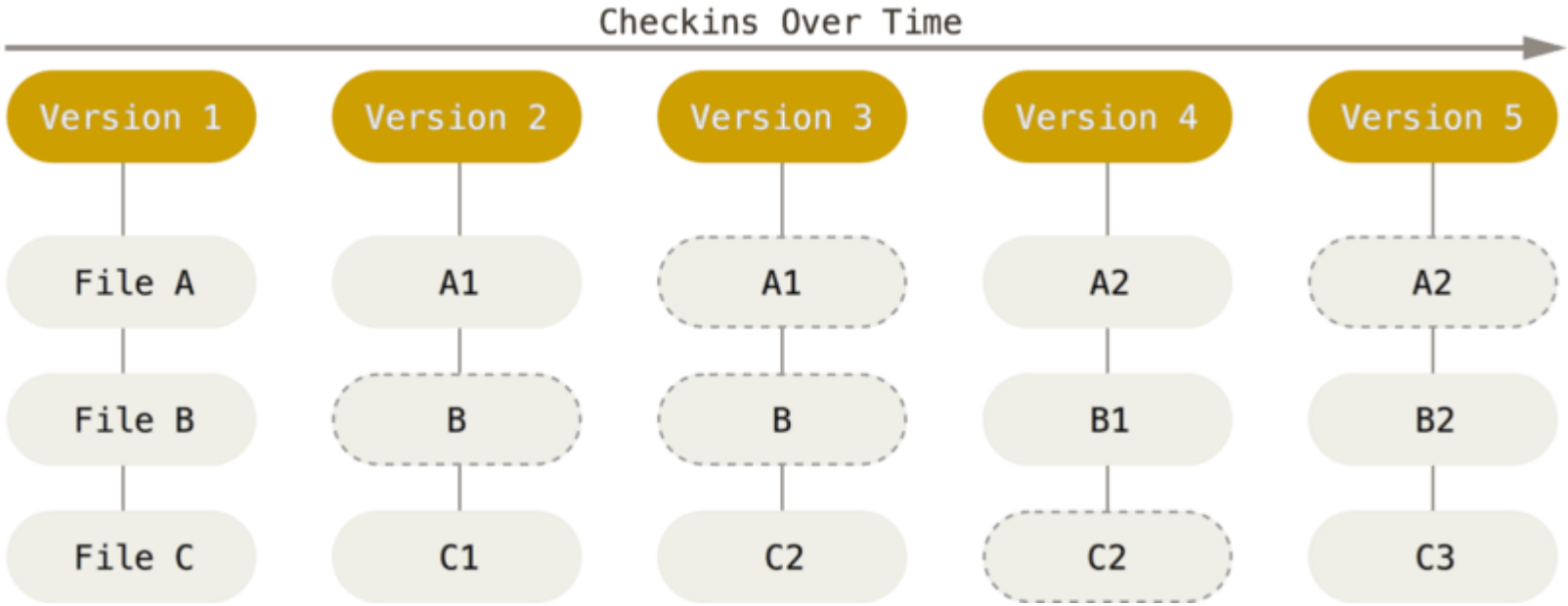
Ігри:

- https://try.github.io/

Книга

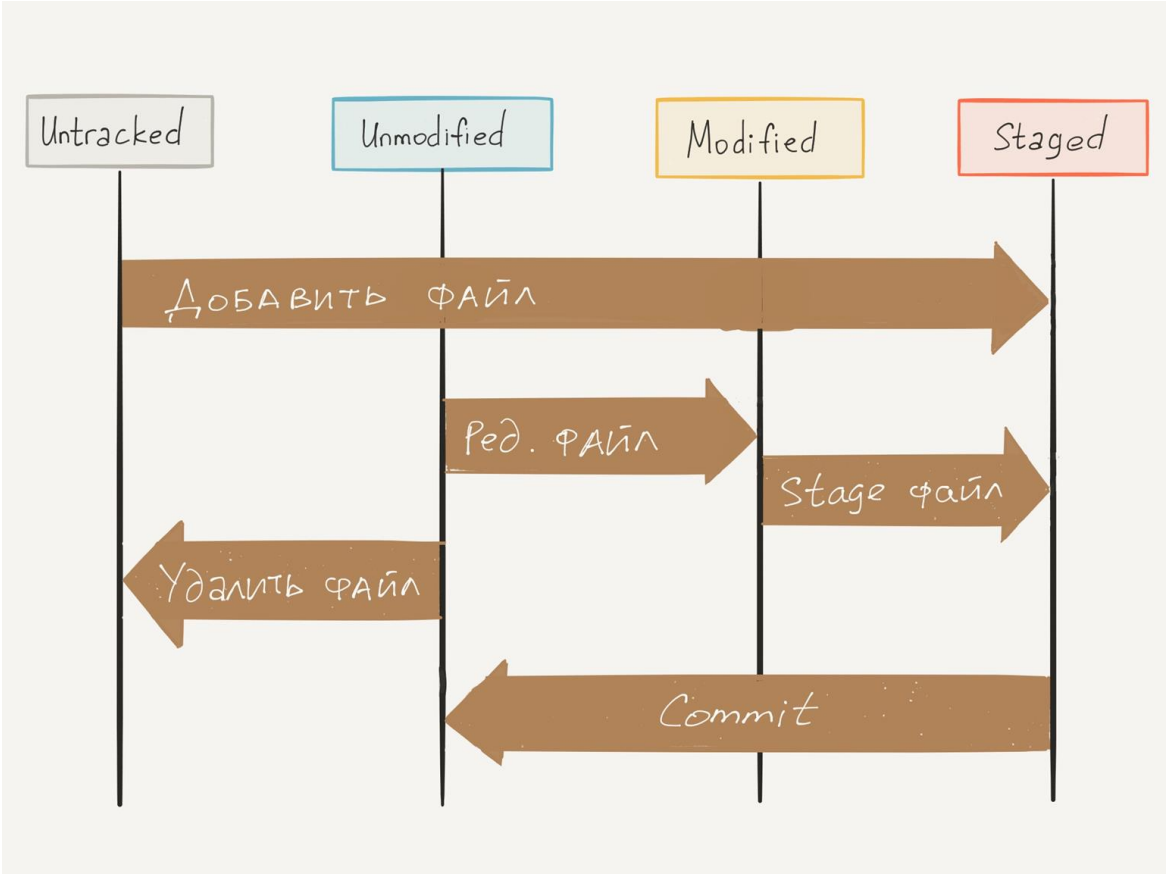
- https://git-scm.com/book/uk/v2

Зберігає змінни, а не файли



Стани файлів

- Збережений у коміті означає, що дані безпечно збережено в локальній базі даних.
- Змінений означає, що у файл внесено редагування, які ще не збережено в базі даних.
- Індексований стан виникає тоді, коли ви позначаєте змінений файл у поточній версії, щоб ці зміни ввійшли до наступного знімку коміту.


Untracked
Unmodified
Modified
Staged
Добавить файл
Ред. файл
Stage файл
Удалить файл
Commit

Створення репозиторію “репи”

```
git init
```

Це дасть git право на доступ до цієї папки і створить прихований каталог *.git*, де буде зберігатися історія репозиторію і його конфігурації.

Команди, конфігурація

Ім'я та адресу електронної пошти

```
git config --global user.name "Your Name"
git config --global user.email "your_email@whatever.com"
```

`--global` задає параметри глобально

Коли задали свої данні то кожен крок, який ми робимо в git буде тепер прив'язуватися до наших ім'я і адреси

Клонування

```
git clone git@github.com:nicothin/web-design.git
```

клонує віддалений репозиторій

Додавання файлів до відстеження, індексація відслідковуваних

`git add text.txt` додає файл до відслідковуваних

`git add .` додає всі редаговані файли до відслідковуваних

Перевірка стану

`git status`

Створення коміту

```
git commit -m "Initial commit."
```

Історія комітів

```
git log
```

```
git log --pretty=oneline
```

Підключення до віддаленого репозиторію

```
git remote add origin https://github.com/codeguida/nolink.git
```

встановлює зв’язок з віддаленим репозиторієм

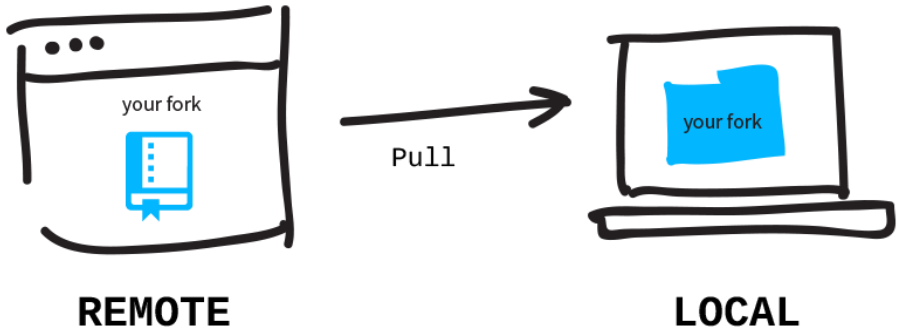
Завантаження на віддалений репозиторій

```
git push origin master
```

приймає два параметри - назва дистанційного репозиторію (у нашому випадку *origin*) і гілка, на яку ми хочемо завантажити коміт (за замовчування для кожного репозиторія встановлена гілка *master*).

Отримання змін з віддаленого репозиторію

```
git pull origin master
```



Гілки (branch)

- копія оригінального проекту
- мають свою історію та ізолюють свої зміни одна від одної, поки ви не вирішите з'єднати їх.Це робиться по деяким причинам:
- Вже працююча, стабільна версія коду не буде порушена
- Багато додатків можуть бути безпечно розроблені одночасно різними людьми
- Розробники можуть працювати на своїй власній гілці, без ризику зміни свого коду кимось іншим
- Коли не впевнені, що краще. Кілька версій одного і того ж додатку можуть бути розроблені на окремих гілках, і потім порівнюватися

Граф гілок

Робота з гілками

`git branch newBranch` - створення нові гілки

`git branch` - список гілку

`git checkout newBranch` - перехід на іншу гілку

`git checkout -b newBranch` - створити нову гілку і зразу на неї перейти

`git branch -d newBranch` - видалити гілку

Об’єднання гілок (merge)

- об’єднує зміни з однієї гілку в поточну
- можуть виникати конфлікти, тоді потрібно їх вирішити і закомітити

```
git merge newBranch
```

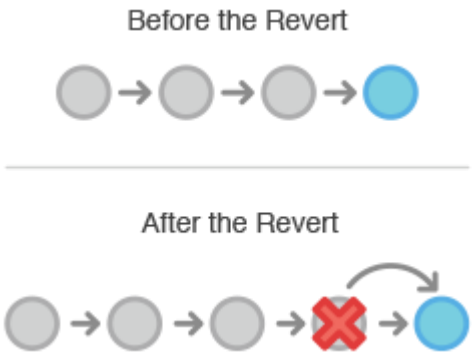
Редагування коміту

`git commit --amend` - якщо ви зробили помилку в коміті чи маленьке доповнення і не хочете робити новий коміт, то зміни додадуться до попереднього коміту

Відмінна коміту

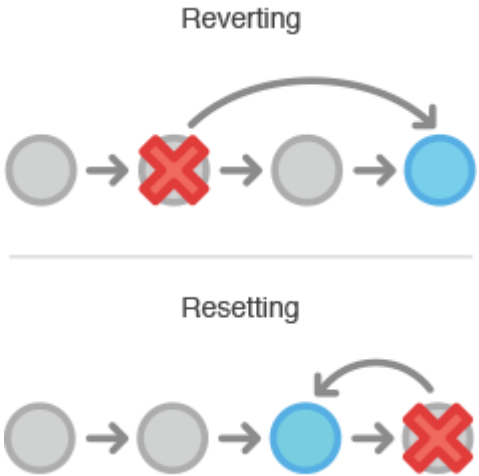
`git revert HEAD --no-edit` - відміннить всі змінни в останьому коміті

`git revert b10cc12` - відміннить всі змінни в певному коміті



Видалення коміту

`git reset --hard b10cc12` - поверне репозиторій до вказаного коміту, всі коміти після нього будуть знищенні!



Файл .gitignore

- вказуються файли і папки, які ми не хочемо комітити

```
*.log
build/
node_modules/
.idea/
my_notes.txt
```

Відео

https://www.youtube.com/watch?v=JfpCicDUMKc

https://www.youtube.com/watch?v=PEKN8NtBDQ0&list=PLUCs6dmWDiDAeBxKDQkSSa34Kz4LpHTpu

https://www.youtube.com/watch?v=en6gms6e54Q&list=PLIU76b8Cjem5B3sufBJ_KFTpKkMEvaTQR