

Git & GitHub



Índice Sesión 6



1. Repaso

- Terminal
- GitKraken

2. Resolución de Conflictos

Repaso Terminal (10 minutos)

Crear repositorio remoto

- Desde GitHub, botón “New”
- Poner nombre: “repo_nombreapellido”
- Inicializar con fichero README

Clonar repositorio en local

- **git clone** *url_repositorio*
- Usar URL de GitHub
- Usar Protocolo HTTP

Crear fichero en Atom

- Guardarlo en la carpeta del repositorio local
- Poner nombre: “fichero_nombreapellido”
- Debe incluir una frase filosófica

Guardar y comitear los cambios en el repositorio local

- **git status** (debe aparecer el fichero en rojo)
- **git add .**
- **git status** (debe aparecer el fichero en verde)
- **git commit -m “mi frase favorita”**

Repaso Terminal (10 minutos)

Subir los cambios al repositorio remoto

- **git push**
- **git status** (debe aparecer *nothing to commit*)
- Comprobar fichero en repositorio remoto en GitHub

Crear nueva rama y cambiar a ella

- **git branch *ejercicio_merge***
- **git branch** (lista ramas existentes)
- **git checkout *ejercicio_merge***
- **git branch** (lista ramas existentes, nueva rama seleccionada)

Subir nueva rama a repositorio remoto

- **git push origin *ejercicio_merge***
- Comprobar repositorio remoto tiene dos ramas

Crear fichero en Atom

- Guardado en la carpeta del repositorio local
- Poner nombre: "fichero2_nombreapellido"
- Debe incluir una serie

Repaso Terminal (10 minutos)

Guardar y comitear los cambios en el repositorio local

- **git branch** (debe estar en la rama *ejercicio_merge*)
- **git status** (debe aparecer el fichero en rojo)
- **git add .**
- **git status** (debe aparecer el fichero en verde)
- **git commit -m "mi serie favorita"**

Subir los cambios al repositorio remoto

- **git push** (**git push --set-upstream origin ejercicio_merge**)
- **git status** (debe aparecer *nothing to commit*)
- Comprobar fichero en repositorio remoto en GitHub (debe estar en la rama nueva)

Mergear rama ejercicio_merge en rama master

- **git checkout master**
- **git branch** (*master* seleccionada)
- **git merge ejercicio_merge**
- **git status** (la rama esta por delante del remoto)
- **git push**
- Comprobar repositorio remoto, rama *master* con todos los ficheros
- Enviar captura de pantalla a Slack

Repaso GitKraken (5 minutos)

Crear repositorio remoto

- Desde GitHub, botón "New"
- Poner nombre: *"repokraken_nombreapellido"*
- Inicializar con fichero README

Clonar repositorio en local

- **Open GitKraken**
- Clone option -> Clone from URL
- Usar URL de GitHub y usar Protocolo HTTP

Crear fichero en Atom

- Guardarlo en la carpeta del repositorio local
- Poner nombre: *"fichero_nombreapellido"*
- Debe incluir una frase filosófica

Guardar y comitear los cambios en el repositorio local

- En GitKraken, en la rama master de LOCAL, clicar en "WIP"
- Fichero debe aparecer en "Unstaged Files", clicar en "Stage all changes"
- Fichero debe aparecer en "Staged Files", escribir un mensaje *"mi frase favorite"* en "Commit Message" y clicar "Commit changes"

Repaso GitKraken (5 minutos)

Subir los cambios al repositorio remoto

- Clicar “Push”
- Comprobar fichero en repositorio remoto en GitHub

Crear nueva rama y cambiar a ella

- Clicar “Branch”
- Introducir nombre de la rama *ejercicio_merge*

Crear fichero en Atom

- Guardado en la carpeta del repositorio local
- Poner nombre: “fichero2_nombreadepellido”
- Debe incluir una serie

Repaso GitKraken (5 minutos)

Guardar y comitear los cambios en el repositorio local

- En GitKraken, en la rama master de LOCAL, clicar en "WIP"
- Fichero debe aparecer en "Unstaged Files", clicar en "Stage all changes"
- Fichero debe aparecer en "Staged Files", escribir un mensaje "mi serie favorita" en "Commit Message" y clicar "Commit changes"

Subir los cambios al repositorio remoto

- Clicar "Push"
- Escribir cuando lo solicite el mismo nombre para la rama remota (se crea automaticamente)
- Comprobar repositorio remoto tiene dos ramas
- Comprobar ficheros en repositorio remoto en GitHub

Mergear rama ejercicio_merge en rama master

- Doble click en LOCAL rama *master*
- Click derecho en rama *ejercicio_merge*
- Seleccionar "Merge ejercicio_merge into master"
- Clicar "Push"
- Comprobar repositorio remoto, rama *master* con todos los ficheros
- Enviar captura de pantalla a Slack

No description, website, or topics provided.

[Edit](#)[Manage topics](#)

3 commits

2 branches

0 releases

1 contributor

Branch: master ▾

[New pull request](#)[Create new file](#)[Upload files](#)[Find File](#)[Clone or download ▾](#)

franguerrerosanchez Mi serie favorita

Latest commit ba148b3 18 minutes ago

README.md	Initial commit	1 hour ago
fichero2_franguerrero	Mi serie favorita	18 minutes ago
fichero_franguerrero	Mi frase favorita	37 minutes ago

[README.md](#)

repo_franguerrero

Resolver Conflictos con GitKraken: Merge

Modificar
fichero_nombreapellido
en rama master

Cambiar a rama
ejercicio_merge

Modificar
fichero_nombreapellido
en rama ejercicio_merge

Mergear rama
ejercicio_merge en rama
master

Resolver Conflictos con GitKraken: Push

Compañero 1 crea repo con fichero de película y hace colaborador a compañero 2

Compañero 2 clona el código modifica fichero de película con el autor en local y sube el código a remoto

Compañero 1 modifica fichero de película cambiando la película en local

Compañero 1 intenta subir cambios a remoto

Conflict



The screenshot shows a code editor interface with a merge conflict in the file `test/tests/diff.js`. The editor is split into two panes, A and B, representing the conflicting versions of the code.

Pane A: Commit ec56e4 on master

```
5 var fse = promisify(require("fs-extra"));
6 var local = path.join.bind(path, __dirname);
7
8 function getLinesFromDiff(diff) {
9   return diff.patches()
10    .then(function(patches) {
11      return Promise.all(_.map(patches, function(patch) {
12        return patch.hunks();
13      }));
14    })
15    .then(function(listOfHunks) {
16      var hunks = _.flatten(listOfHunks);
17      return Promise.all(_.map(hunks, function(hunk) {
18        return hunk.lines();
19      }));
20    });
21 }
```

Pane B: Commit e3862c on find-similar-spec

```
5 var local = path.join.bind(path, __dirname);
6
7 var findSimilarOpts = {
8   flags: Diff.FIND.RENAMES |
9         Diff.FIND.RENAMES_FROM_REWRITES |
10        Diff.FIND.FOR_UNTRACKED
11 };
12
```

Output: conflict 1 of 2

The output pane shows the result of the conflict resolution, where the code from Pane B is merged into the code from Pane A. The conflict is resolved by inserting the code from Pane B into the code from Pane A.

```
5 var fse = promisify(require("fs-extra"));
6 var local = path.join.bind(path, __dirname);
7
8 var findSimilarOpts = {
9   flags: Diff.FIND.RENAMES |
10         Diff.FIND.RENAMES_FROM_REWRITES |
11        Diff.FIND.FOR_UNTRACKED
12 };
13
14 function getLinesFromDiff(diff) {
15   return diff.patches()
16    .then(function(patches) {
17      return Promise.all(_.map(patches, function(patch) {
18        return patch.hunks();
19      }));
20    });
21 }
```

Right Panel: Merge Conflicts

The right panel shows the status of the merge. It indicates that the merge is in progress and that there is a conflict in the file `test/tests/diff.js`. The panel includes buttons for **Abort Merge**, **Mark all resolved**, and **Commit and Merge**.

Commit Message

The commit message is set to `Merge branch 'find-similar-spec'`. The conflicts are listed as `test/tests/diff.js`.

Buttons:

- Commit and Merge** (green button)
- Abort Merge** (red button)
- Provide Feedback** (blue button)