

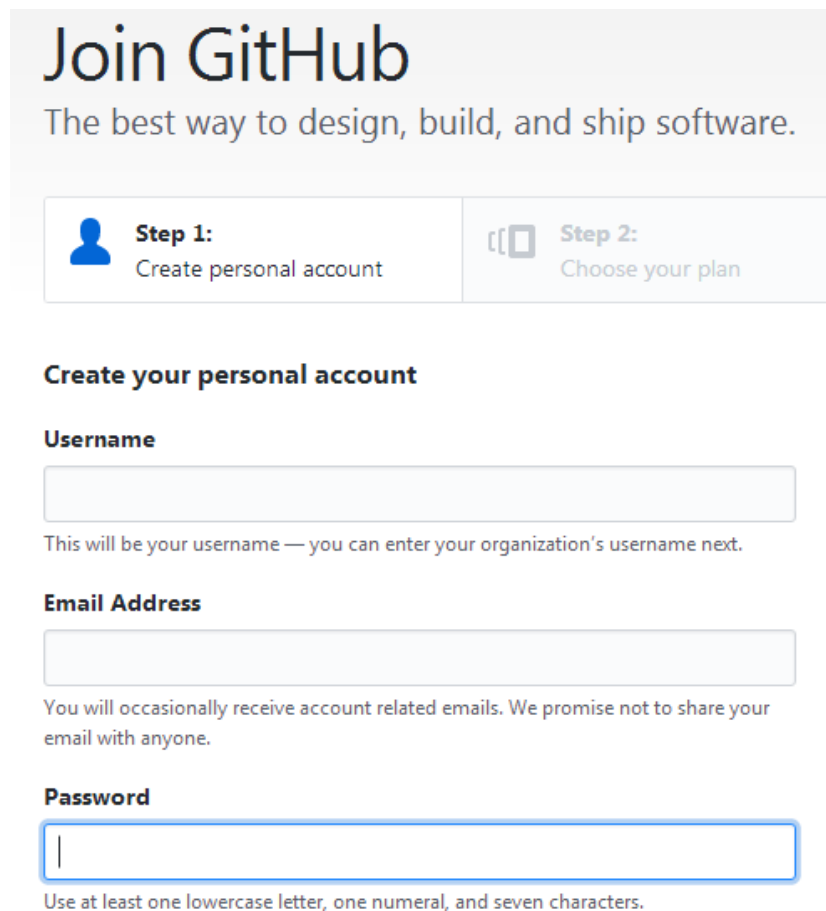
LAB 1 – Controlo de versões de código com GitHub

Neste lab vai fazer o upload do seu projecto para o GitHub e em seguida fazer o seu *download*.

Nota: Este lab assume que está a fazer o desenvolvimento do projecto no servidor 10.10.23.183. As instruções de acesso a este servidor encontram-se em http://intranet.deei.fct.ualg.pt/DAW/DAW_instrucoes_acesso.html

1. Cria uma conta

Começa por criar uma conta em GitHub (<https://github.com/>) com o teu browser





The screenshot shows the GitHub 'Join GitHub' page. At the top, it says 'Join GitHub' and 'The best way to design, build, and ship software.' Below this, there are two steps: 'Step 1: Create personal account' and 'Step 2: Choose your plan'. The 'Step 1' section is active and contains the following fields:

- Create your personal account**
- Username**: A text input field. Below it, a note says: 'This will be your username — you can enter your organization's username next.'
- Email Address**: A text input field. Below it, a note says: 'You will occasionally receive account related emails. We promise not to share your email with anyone.'
- Password**: A text input field. Below it, a note says: 'Use at least one lowercase letter, one numeral, and seven characters.'

e cria um repositório: chama-lhe **LAB1**

Create a new repository


A repository contains all the files for your project, including the revision history.


Owner
  ▾

Repository name

Great repository names are short and memorable. Need inspiration? How about **curly-octo-telegram**.

Description (optional)

☒  **Public**
Anyone can see this repository. You choose who can commit.

☐  **Private**
You choose who can see and commit to this repository.

2. Cria um *snapshot* local do teu projecto

Faz login no servidor 10.10.23.184. Executa o seguinte conjunto de comandos para criar uma pasta para o teu projecto:

```
a12345@daw:~$ cd ~/public_html/  
a12345@daw:~/public_html$ mkdir LAB1  
a12345@daw:~/public_html$ cd LAB1  
a12345@daw:~/public_html/LAB1$ git init
```

Com o teu editor de texto favorito crie o ficheiro `index.html` com o seguinte texto

```
a12345@daw:~/public_html/LAB1$ nano index.html
```

```
<html>  
<head>  
  <title>Helo</title>  
</head>  
<body>  
  <p>Helo Worl!</p>  
</body>  
</html>
```

O passo seguinte é adicionar todos os ficheiros do projecto (agora apenas um...) ao snapshot local com o comando `git add -A`:

```
$ git add -A
```

Nota: Este comando adiciona ao snapshot todos os ficheiros na pasta actual **excepto** aqueles especificados no ficheiro escondido `.gitignore`.

Para tornar o snapshot efectivo use o comando `commit` :

```
$ git commit -m "1a versao do projecto"
```

onde “1a versao...” é um texto sugestivo, que informa sobre uma correcção de um bug, uma nova funcionalidade...

3. Faz o upload do projecto para o GitHub

Nota: substitui “jsilva” pelo teu username

```
$ git remote add origin https://github.com/jsilva/LAB1.git
$ git push -u origin --all # pushes up the repo and its refs for the
first time
```

```
Username for 'https://github.com': jsilva
Password for 'https://jsilva@github.com':
```

Feito! O teu projecto LAB1 já está alojado no GitHub

4. Trabalha num project branch

Um “project branch” permite trabalhar localmente numa versão do projecto, de forma independente da versão “master”

```
$ git checkout -b LAB1v2
```

Altera o ficheiro “index.html” com o teu editor favorito

```
a12345@daw:~/public_html/LAB1$ nano index.html
```

```
<html>
<head>
  <title>Hello</title>
</head>
<body>
  <p>Hello World!</p>
</body>
</html>
```

Torna definitiva a alteração e faz o “upload” do branch para o GitHub

```
$ git add -A
$ git status
$ git commit -m "correcao de erros ortograficos"
$ git push origin LAB1v2
```

faz o “merge” localmente com a versão “master”

```
$ git checkout master
$ git merge LAB1v2
$ git push
```

5. Faz mais melhoramentos no project branch

```
$ git checkout LAB1v2
```

Altera o ficheiro “index.html” com o teu editor favorito

```
a12345@daw:~/public_html/LAB1$ nano index.html
```

```
<html>
<head>
  <title>BIG Hello</title>
</head>
<body>
  <h1>Hello World!</h1>
</body>
</html>
```

Torna definitiva a alteração e faz o “merge” com a versão “master”

```
$ git add -A
$ git commit -m "alteracao do estilo"
$ git checkout master
$ git merge LAB1v2
$ git push
```

(Repeat this step as often as you wish, during the development of your project)

6. Recupera a versão anterior do projecto

```
$ git log
$ git reset "commit"
```

onde “commit” é a referência da segunda versão do projecto (exemplo: be871b9675b8983a3bb7f39d427c71a26dc37a1f)

7. Faz o download do projecto

Vamos criar uma nova pasta e testar o download/clonagem do projecto (substitui jsilva pelo teu username ou de um teu colega de turma...)

```
$ cd ~/public_html  
$ git clone git://github.com/jsilva/LAB1.git LAB1_download  
$ ls
```

O directório `public_html` deve agora conter duas pastas, “LAB1” e “LAB1_download” e estas pastas devem ser visíveis no URL

<http://all.deei.fct.ualg.pt/~a12345>

(substitui “12345” pelo teu número de aluno)

8. Conclusão

O lab considera-se validado quando a tua pagina em GitHub tiver pelo menos dois “commits” e um “branch” no repositório LAB1

Referências:

- http://intranet.deei.fct.ualg.pt/DAW/DAW_instrucoes_acesso.html
- <https://github.com>
- <http://intranet.deei.fct.ualg.pt/DAW/Bash.pdf>