

# DESPLIEGUE DE APLICACIONES WEB

## GUÍA DE RESOLUCIÓN DE LA TAREA 05

Jesús Bono – IES CRISTÓBAL DE MONROY

### Contenido

Actividad 1.- Herramientas de documentación de aplicaciones. ....	3
1.1.- Funcionamiento de PhpDocumentor, etiquetas y procedimiento de generación de código. ....	3
1.2.- Instalación de PhpDocumentor. ....	4
Actividad 2.- Documentación de un Script PHP. ....	5
Actividad 3.- Generando documentación en HTML con PHPdocumentor. ....	6
Actividad 4.- Compartiendo nuestros proyectos en Github. ....	7
4.1.- Crea un repositorio público que se llame 'distancia23'. ....	8
4.2.- Crea un fichero README.md (formato Markdown) e incluye tu nombre, primer apellido y el texto 'Actividad Despliegue - Unidad 05 - 2022/23'. ....	8
4.3.- Dentro de este repositorio sube la carpeta 'scripts' que generamos en Apache (código fuente). ....	9
4.4.- Sube ahora también la carpeta 'documentacion', con toda la documentación generada en la actividad anterior. ....	11
Actividad 5.- Instala la herramienta de control de versiones Git en Linux. ....	11
5.1.- Inicializar un repositorio local. ....	12
5.2.- Añadir archivos modificados para el siguiente commit (stage). ....	12
5.3.- Mostrar los archivos modificados en nuestra área de trabajo para el próximo commit. ....	12
5.4.- Hacer un commit con el mensaje "esto es una descripción del commit". ....	12
5.5.- Mostrar todos los commits realizados en la rama actual. ....	12
Actividad 6.- Clonando el repositorio remoto con Git. ....	13
6.1.- Crea una nueva carpeta en tu máquina Linux: /var/www/html/xxxyyy/ donde xxx son las tres primeras letras de tu nombre e yyy son las tres primeras letras de tu apellido. ....	13
6.2.- Clona el repositorio remoto de GitHub. ....	13

6.3.- Comprueba desde el navegador que puedes ver el contenido en <a href="http://localhost/xxxyyy/">http://localhost/xxxyyy/</a> .....	14
6.4.- Genera un token de seguridad para poder realizar operaciones sobre el repositorio remoto de GitHub.....	14
Actividad 7.- Poniendo a prueba el control de versiones.....	15
7.1.- Modifica el código de los scripts PHP que has clonado, cambiando el contenido de algunas etiquetas, como por ejemplo la etiqueta '@version'. .....	15
7.2.- Realiza los pasos necesarios para modificar el repositorio, incluyendo en el commit el mensaje "Modificación de versión distancia23". .....	16
7.3.- Actualiza el repositorio remoto con los cambios realizados en el repositorio local. ....	16
7.4.- Comprueba en Github que se han realizado los cambios, revisando el historial de la rama actual.....	17

## Actividad 1.- Herramientas de documentación de aplicaciones.

Explica brevemente cómo funciona la herramienta Phpdocumentor, indicando las etiquetas más habituales y el procedimiento para generar la documentación una vez que se ha documentado el código. Realiza y documenta la instalación de esta herramienta en tu máquina Linux.

### 1.1.- Funcionamiento de PhpDocumentor, etiquetas y procedimiento de generación de código.

PhpDocumentor es una herramienta utilizada en el lenguaje de programación PHP para generar documentación automáticamente a partir del código fuente. Su objetivo es facilitar la comprensión y el mantenimiento del código, especialmente en proyectos grandes y complejos.

Para generar una documentación útil, es necesario documentar adecuadamente el código fuente. PhpDocumentor utiliza etiquetas especiales, llamadas "etiquetas de documento", para capturar información sobre las clases, métodos, variables y otros elementos del código. Algunas de las etiquetas más habituales son:

- **@access:** Si @access es 'private' no se genera documentación para el elemento (a menos que se indique explícitamente). Muy interesante si sólo se desea generar documentación sobre la interfaz (métodos públicos) pero no sobre la implementación (métodos privados).
- **@author:** Autor del código.
- **@copyright:** Información sobre derechos.
- **@deprecated:** Para indicar que el elemento no debería utilizarse, ya que en futuras versiones podría no estar disponible.
- **@example:** Permite especificar la ruta hasta un fichero con código PHP. phpDocumentor se encarga de mostrar el código resaltado (syntax-highlighted).
- **@ignore:** Evita que phpDocumentor documente un determinado elemento.
- **@internal:** Para incluir información que no debería aparecer en la documentación pública, pero sí puede estar disponible como documentación interna para desarrolladores.
- **@link:** Para incluir un enlace (http://...) a un determinado recurso.
- **@see:** Se utiliza para crear enlaces internos (enlaces a la documentación de un elemento).
- **@since:** Permite indicar que el elemento está disponible desde una determinada versión del paquete o distribución.
- **@version:** Versión actual del elemento.

Una vez creados los DocBlocks configuraremos la ruta del código fuente que la aplicación escaneará en busca de los ficheros, leyendo DockBlocks y generando la documentación relacionada en la ruta que le indiquemos.

El comando de manera simplificada: “phpdoc -d <ruta del código fuente> -t <ruta de destino>”

## 1.2.- Instalación de PhpDocumentor.

Previamente a usar PhpDocumentor se ha de tener instalado un servidor Apache, además de tener instalado php. Instalemos php y sus dependencias:

```
$ sudo apt-get install php-pear, con esto nos instalará la versión 7.4.3.
```

```
$ sudo pear install --alldeps PhpDocumentor
```

Configuramos la ruta con la que queremos trabajar con PhpDocumentor

```
$ sudo pear config-set data_dir /var/www/html
```

Instalamos ahora el módulo de php en apache:

```
$ sudo systemctl stop apache2
```

```
$ sudo apt install libapache2-mod-php
```

```
$ sudo systemctl restart apache2
```

Descargamos y descomprimos PhpDocumentor 1.4.3:

```
$ sudo wget  
sourceforge.net/projects/phpdocu/files/PhpDoc/phpDocumentor-  
1.4.3/PhpDocumentor-1.4.3.tgz
```

```
$ sudo tar xvfz PhpDocumentor-1.4.3.tgz
```

Muevo la carpeta:

```
$ sudo mv PhpDocumentor-1.4.3 /var/www/ PhpDocumentor
```

Borro el fichero descargado:

```
$ sudo rm PhpDocumentor-1.4.3.tgz
```

## Actividad 2.- Documentación de un Script PHP.

En esta actividad debes añadir etiquetas a al menos dos scripts PHP, de creación propia o facilitados por el profesorado. El primer scripts debe contener al menos dos funciones y el segundo script debe contener una clase con al menos dos métodos. Los requisitos que debes tener en cuenta son:

- El nombre de los scripts será **scrip1-XXXX.php** y **script2-XXXX.php**, donde XXXXX será tu primer apellido.
- Debes incluir un **docblock** al inicio, describiendo la **funcionalidad** del script, **autor/a** y **versión**.
- Las **funciones** y **métodos** deben tener un docblock donde se aparezcan los **parámetros de entrada/salida** y la **versión**, además de explicar lo que hace.
- Ambos scripts deben incluir además **2 etiquetas de libre elección** por parte del alumnado, que no se hayan usado previamente en el ejercicio.
- Los 2 scripts se ubicarán en la **carpeta /var/www/html/distancia23/scripts/**

Creamos la carpeta scripts:

```
$ sudo mkdir /var/www/html/distancia23/scripts/
```

Creamos los sripts/ficheros de php e incluimos todo lo pedido en la actividad (APARTADO 2.1.- Funcionamiento de phpDocumentor), guardándolos en la ruta anterior.

Ejemplo de script:

```
<?php
/**
 * Funciones que realizan operaciones de
 * suma, resta, multiplicación y división.
 * Cada función recibe dos parámetros para operar.
 *
 * @author Nombre Apellido
 *
 * @version 1.3
 */

/**
 * @param Integer $num1 primer sumando.
 * @param Integer $num2 segudno sumando.
 * @return Integer Devuelve la suma entre el primer y segundo número.
 */
function sumar($num1, $num2) {
    return $num1 + $num2;
}
```

## Actividad 3.- Generando documentación en HTML con PHPdocumentor.

Ahora que ya has documentado los scripts PHP es momento de generar la documentación de nuestro proyecto. Como la salida de PHPdoc va a ser una estructura web en HTML, vamos a indicarle como directorio de salida una carpeta de Apache, para poder verla desde nuestro navegador. Los requisitos que debes tener en cuenta son:

- La ruta origen de los scripts será la carpeta **`/var/www/html/distancia23/scripts/`**
- La ruta destino de la documentación será la carpeta **`/var/www/html/distancia23/documentacion/`**

Una vez generada la documentación, debes comprobar el resultado obtenido, accediendo desde el navegador a la ruta <http://localhost/distancia23/documentacion/> y navegando por los documentos generados. Incluye capturas en las que se pueda constatar que se ha generado la documentación acorde a las etiquetas incluidas.

Necesitamos crear un directorio de salida para phpDocumentor, y cambiar el propietario de dicho directorio a www-data, en nuestro caso será el directorio `"/var/www/html/distancia23/documentación"`:

```
$ sudo mkdir /var/www/html/distancia23/documentacion  
$ sudo chown www-data /var/www/html/distancia23/documentacion/
```

Usamos el comando `phpdoc -o <formato-de-salida> -d <directorio-de-origen> -f <directorio-desalida>` para generar la documentación:

```
$ sudo /var/www/html/PhpDocumentor/phpdoc -o HTML:frames:phpedit -d  
    /var/www/html/distancia23/scripts/ -t  
    /var/www/html/distancia23/documentacion/
```

Comprobamos la documentación generada accediendo desde el navegador a la ruta <http://localhost/distancia23/documentacion/>

(\*) En el caso de haber tenido problemas, el bajar la versión de php a la 5.6 lo arreglará. Más información de cómo hacerlo:  
<https://diarioprogramador.com/como-instalar-distintas-versiones-de-php-en-ubuntu/>

```
sudo systemctl restart apache2
```

```
sudo add-apt-repository ppa:ondrej/php
```

```
sudo apt update
```

```
sudo apt install php5.6
```

```
sudo update-alternatives --set php /usr/bin/php5.6
```

```
sudo a2dismod php7.4
```

```
sudo a2enmod php5.6
```

```
sudo systemctl restart apache2
```

## Actividad 4.- Compartiendo nuestros proyectos en Github.

En esta actividad vamos a compartir con la comunidad nuestro proyecto y la documentación generada. Para ello usaremos GITHUB. Si no tienes cuenta en esta plataforma, es el momento de que abras una. Una vez que tengas acceso a Github sigue los siguientes pasos:

- Crea un **repositorio público** que se llame '**distancia23**'
- Crea un fichero **README.md** (formato Markdown) e incluye tu nombre, primer apellido y el texto '**Actividad Despliegue - Unidad 05 - 2022/23**'
- Dentro de este repositorio sube la **carpeta 'scripts'** que generamos en Apache (código fuente)
- Sube ahora también la **carpeta 'documentacion'**, con toda la documentación generada en la actividad anterior.

#### 4.1.- Crea un repositorio público que se llame 'distancia23'


### Create a new repository

A repository contains all project files, including the revision history. Already have a project repository elsewhere? [Import a repository.](#)

Owner \*

Repository name \*

1

 J BonoB

/


distancia23

distancia23 is available.

Great repository names are short and memorable. Need inspiration? How about [miniature-waffle?](#)


Description (optional)

Creado para trabajar con la tarea 5 - 22/23

☒  Public

2

Anyone on the internet can see this repository. You choose who can commit.

☐  Private

You choose who can see and commit to this repository.

Initialize this repository with:

☒ Add a README file

3

This is where you can write a long description for your project. [Learn more about READMEs.](#)

Add .gitignore


.gitignore template: None

Choose which files not to track from a list of templates. [Learn more about ignoring files.](#)


Choose a license

License: None

A license tells others what they can and can't do with your code. [Learn more about licenses.](#)

This will set  main as the default branch. Change the default name in your settings.

4

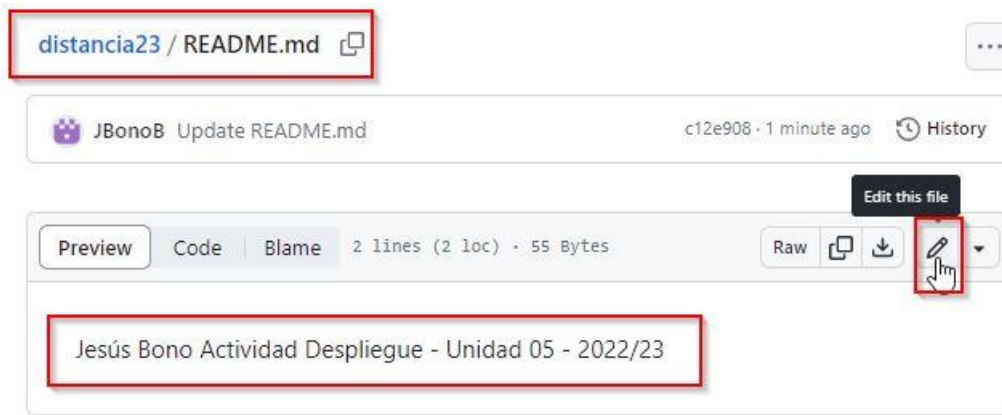
 You are creating a public repository in your personal account.

Create repository

#### 4.2.- Crea un fichero README.md (formato Markdown) e incluye tu nombre, primer apellido y el texto 'Actividad Despliegue - Unidad 05 - 2022/23'

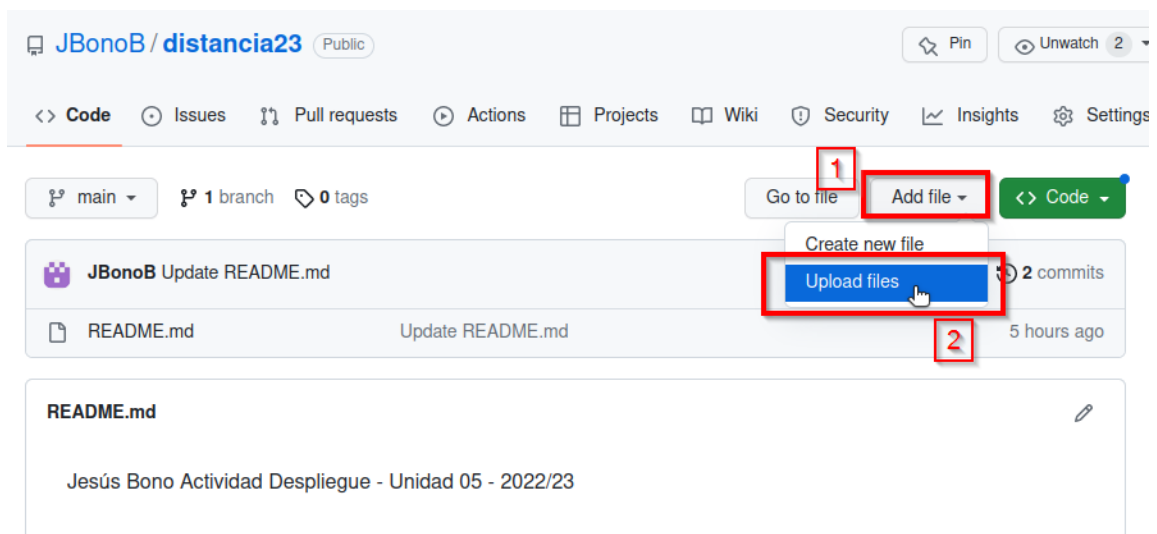
Al crear el repositorio marqué la opción de crear el fichero readme.md con lo que ahora pulso sobre el botón de editar (lápiz) y lo dejo modificado tal y como se pide:






4.3.- Dentro de este repositorio sube la carpeta 'scripts' que generamos en Apache (código fuente)

Selecciono subir ficheros (upload files) y subo la carpeta:




En la siguiente ventana arrastro la carpeta dejando preparado con los archivos que voy a subir:

distancia23 /

  
**Drag additional files here to add them to your repository**  
[Or choose your files](#)

 /scripts/script1-xxx.php

 /scripts/script2-xxx.php

Añado el comentario de lo que voy a subir y verifico que quiero que se suban:



### Commit changes

Subo la carpeta scripts al repositorio

Add an optional extended description...

☒ Commit directly to the `main` branch.

☐ Create a **new branch** for this commit and start a pull request. [Learn more about pull requests.](#)

Commit changes

Cancel

Quedando el repositorio con el fichero readme y la carpeta scripts:

JBonoB **distancia23** Public

Pin

Unwatch 2

<> Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main

1 branch 0 tags

Go to file

Add file

<> Code

JBonoB Subo la carpeta scripts al repositorio

c849dff 1 minute ago 3 commits

scripts Subo la carpeta scripts al repositorio

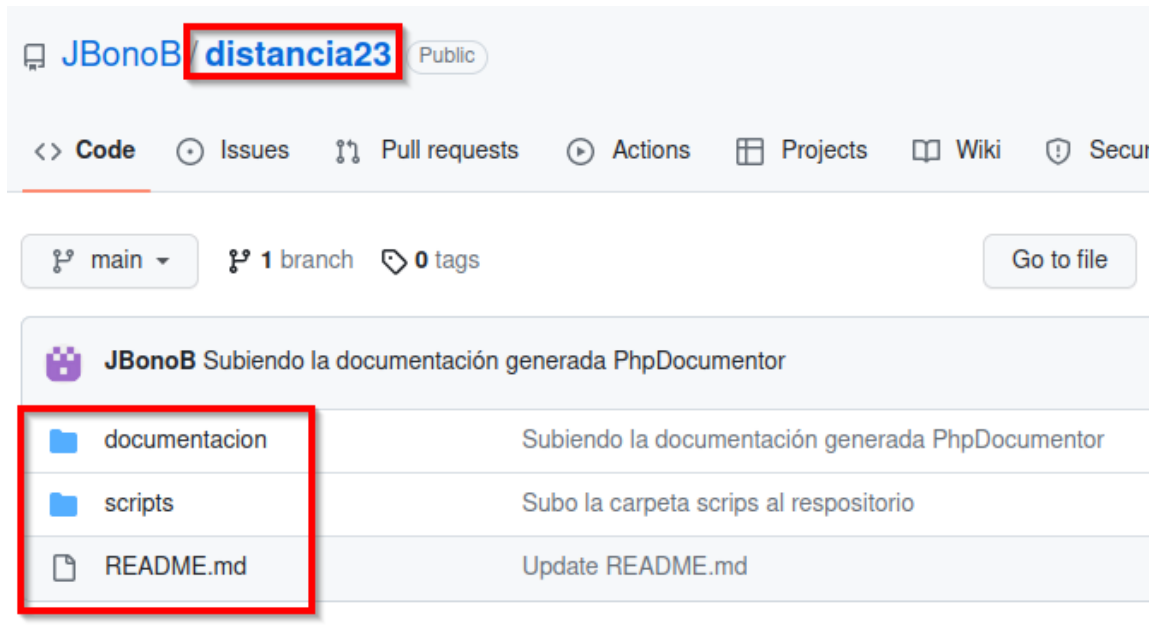
1 minute ago

README.md Update README.md

6 hours ago

4.4.- Sube ahora también la carpeta 'documentacion', con toda la documentación generada en la actividad anterior.

Procedemos de la misma manera que el apartado anterior quedando:



## Actividad 5.- Instala la herramienta de control de versiones Git en Linux.

Vamos a instalar en nuestra máquina Linux el software **Git** para control de versiones. Una vez instalado, debemos configurarlo con nuestro nombre y nuestro correo electrónico. Documenta todo el proceso de instalación y configuración y explica los comandos necesarios para los diferentes apartados.

Para instalar Git:

```
$ sudo apt install git
```

Posteriormente configuramos nuestro nombre y correo:

```
$ git config --global user.name "Jesus"
```

```
$ git config --global user.email "jesus.bono@iescristobaldemonroy.es"
```

## 5.1.- Inicializar un repositorio local

**\$ git init**

```
despliegue@ubuntu:~$ git init
Inicializado repositorio Git vacío en /home/despliegue/.git/
despliegue@ubuntu:~$
```

## 5.2.- Añadir archivos modificados para el siguiente commit (stage)

Lanzo un "ls" para ver todo lo que tengo en la carpeta que estoy situado y posteriormente añado todo al repositorio:

**\$ ls**

**\$ git add .**

```
despliegue@ubuntu:~$ ls
Descargas  Escritorio  Música      Público  Vídeos
Documentos Imágenes    Plantillas  snap
despliegue@ubuntu:~$ git add .
despliegue@ubuntu:~$
```

## 5.3.- Mostrar los archivos modificados en nuestra área de trabajo para el próximo commit

**\$ git status**

Con lo anterior nos mostrará una lista enorme de cambios, una fila por cada archivo nuevo que vamos a subir.

## 5.4.- Hacer un commit con el mensaje "esto es una descripción del commit"

**\$ git commit -m "esto es una descripción del commit"**

## 5.5.- Mostrar todos los commits realizados en la rama actual.

Pedimos que nos muestre el histórico ejecutando git log, y ahí está el commit con nuestro comentario:

**\$ git log**

```
despliegue@ubuntu:~$ git log
commit cf16cd03722e6d68b7bea96962ead63169850d0d (HEAD -> master)
Author: "Jesus" <"jesus.bono@iescristobaldemonroy.es">
Date:   Wed May 31 23:34:18 2023 +0000

    este es una descripción del commit
despliegue@ubuntu:~$
```

## Actividad 6.- Clonando el repositorio remoto con Git.

Una vez que tenemos instalado Git en nuestra máquina, procederemos a sincronizar el repositorio público que creamos en Github. Para ello seguiremos los siguientes pasos:

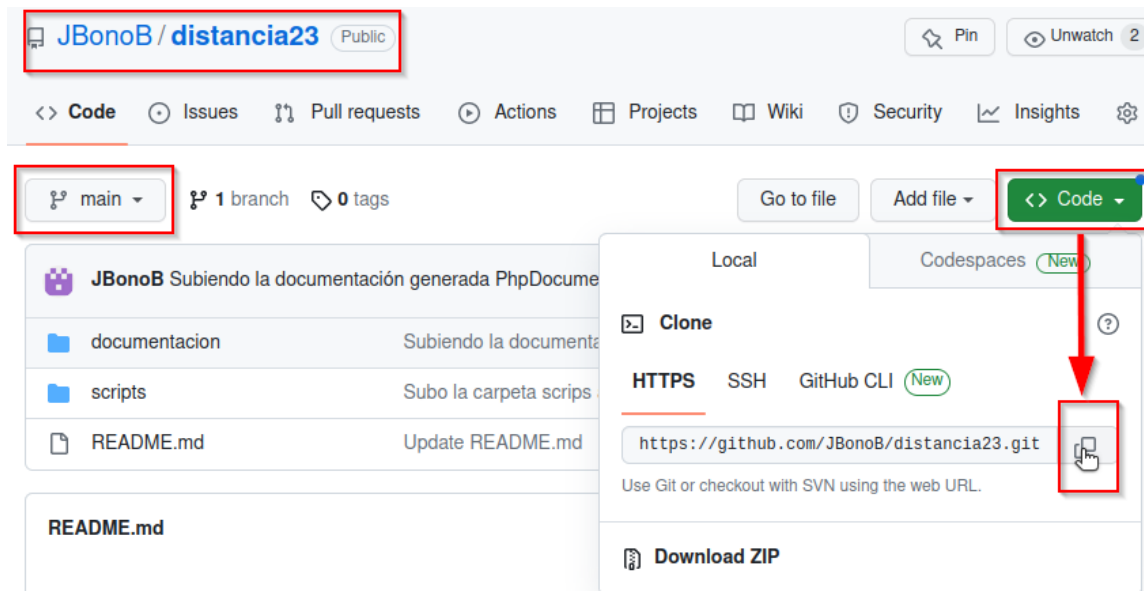
- Crea una nueva carpeta en tu máquina Linux: `/var/www/html/xxxxyy/` donde xxx son las tres primeras letras de tu nombre e yyy son las tres primeras letras de tu apellido.
- Clona el repositorio remoto de GitHub.
- Comprueba desde el navegador que puedes ver el contenido en <http://localhost/xxxxyy/>
- Genera un token de seguridad para poder realizar operaciones sobre el repositorio remoto de GitHub.

6.1.- Crea una nueva carpeta en tu máquina Linux: `/var/www/html/xxxxyy/` donde xxx son las tres primeras letras de tu nombre e yyy son las tres primeras letras de tu apellido.

```
$ sudo mkdir /var/www/html/jesbon
```

6.2.- Clona el repositorio remoto de GitHub.

Buscamos en GitHub la dirección del repositorio para traérnoslo:



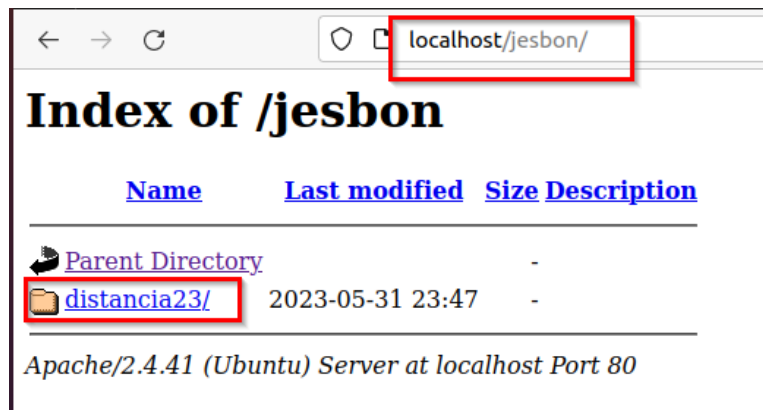
Nos vamos al directorio creado, inicializo repositorio y lanzamos la clonación:

```
$ cd /var/www/html/jesbon
```

```
$ sudo git init
```

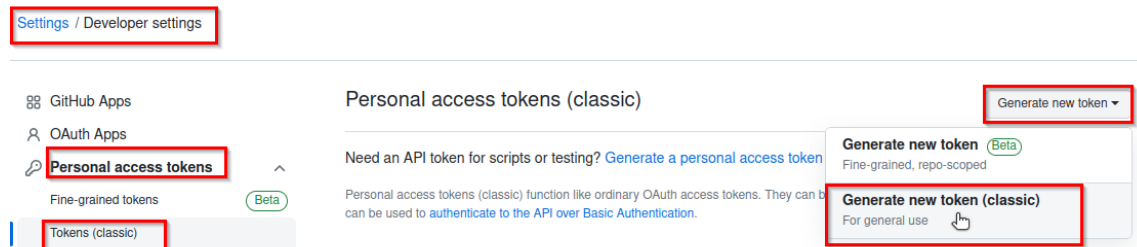
```
$ sudo git clone https://github.com/JBonoB/distancia23.git
```

6.3.- Comprueba desde el navegador que puedes ver el contenido en <http://localhost/xxxxyyy/>



6.4.- Genera un token de seguridad para poder realizar operaciones sobre el repositorio remoto de GitHub.

Desde la interface web de GitHub, botón derecho sobre el usuario y seleccionamos Settings. Buscamos y pulsamos Developer Settings, seleccionamos Personal Access Tokens, seleccionamos Tokens (classic) y pulsamos el botón “Generate new token” y finalmente pulsamos sobre “Generate new token (classic)”:



## New personal access token (classic)

Personal access tokens (classic) function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

### Note

distancia23

What's this token for?

Generate token Cancel

Me lo crea, y de ahí debemos copiar la contraseña para poder trabajar con el:

Tokens you have generated that can be used to access the [GitHub API](#).

Make sure to copy your personal access token now. You won't be able to see it again!

✓ ghp\_sIc0fyXgCRaiAvkFaKTrYtHOL0x3Ku3ZpdJz

Delete

**\$ sudo git config user.password ghp\_slc0fyXgCRaiAvkFaKTrYtHOL0x3Ku3ZpdJz**

```
despliegue@ubuntu: ~  
despliegue@ubuntu:~$ sudo git config user.password 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  
[sudo] contraseña para despliegue:  
despliegue@ubuntu:~$
```

## Actividad 7.- Poniendo a prueba el control de versiones.

Para comprobar cómo funciona el control de versiones, vamos a realizar unas modificaciones en el repositorio local y luego las vamos a sincronizar con el remoto. Realiza los siguientes pasos:

- **Modifica el código** de los scripts PHP que has clonado, cambiando el contenido de algunas etiquetas, como por ejemplo la **etiqueta '@version'**.
- Realiza los **pasos necesarios para modificar el repositorio**, incluyendo en el **commit** el mensaje "**Modificación de versión distancia23**".
- **Actualiza el repositorio remoto** con los cambios realizados en el repositorio local.
- **Comprueba** en Github que se han realizado los cambios, revisando el **historial de la rama actual**.

7.1.- Modifica el código de los scripts PHP que has clonado, cambiando el contenido de algunas etiquetas, como por ejemplo la etiqueta '@version'.

**\$ sudo nano /var/www/html/jesbon/distancia23/scripts/scrip1-xxx.php**



```
despliegue@ubuntu: /var/www/html/jesbon/distancia23/scri...
GNU nano 4.8 scrip1-xxx.php Modificado
<?php
/**
 * Funciones que realizan operaciones de
 * suma, resta, multiplicación y división.
 * Cada función recibe dos parámetros para operar.
 *
 * @author Jesús Bono
 * @version 2.0
 */
/**
```

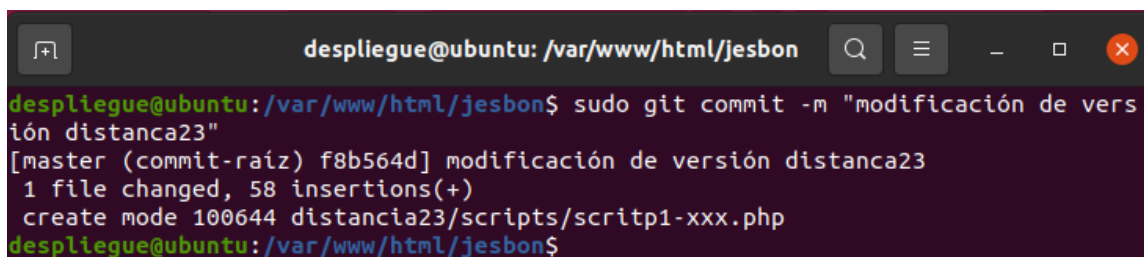
7.2.- Realiza los pasos necesarios para modificar el repositorio, incluyendo en el commit el mensaje "Modificación de versión distancia23".

Nos vamos a la ruta donde creamos el repositorio y con ruta relativa añadimos el archivo y hacemos commit:

```
$ cd /var/www/html/jesbon
```

```
$ sudo git add distancia23/scripts/scrip1-xxx.php
```

```
$ sudo git commit -m "modificación de versión distancia23"
```



```
despliegue@ubuntu: /var/www/html/jesbon
despliegue@ubuntu:/var/www/html/jesbon$ sudo git commit -m "modificación de vers
ión distancia23"
[master (commit-raíz) f8b564d] modificación de versión distancia23
1 file changed, 58 insertions(+)
create mode 100644 distancia23/scripts/scrip1-xxx.php
despliegue@ubuntu:/var/www/html/jesbon$
```

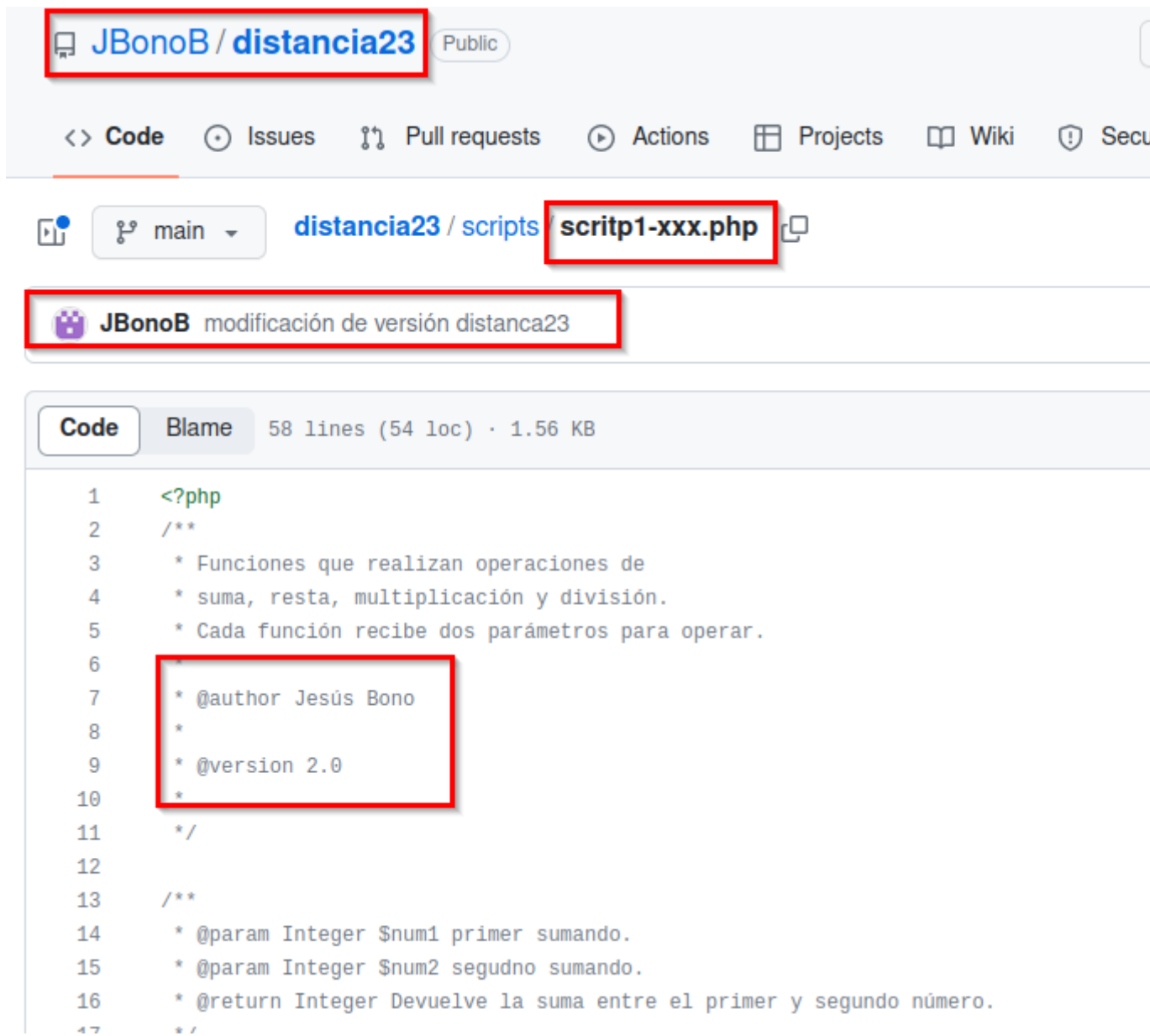
7.3.- Actualiza el repositorio remoto con los cambios realizados en el repositorio local.

Nos posicionamos sobre la carpeta distancia23 y realizamos un push dando los datos de usuario y contraseña (usuario GitHub y contraseña el token):

```
$ sudo git push
```



7.4.- Comprueba en Github que se han realizado los cambios, revisando el historial de la rama actual.



**JBonoB / distancia23** Public

<> **Code** Issues Pull requests Actions Projects Wiki Security

main distancia23 / scripts **scrip1-xxx.php**

**JBonoB** modificación de versión distanca23

**Code** Blame 58 lines (54 loc) · 1.56 KB

```
1 <?php
2 /**
3  * Funciones que realizan operaciones de
4  * suma, resta, multiplicación y división.
5  * Cada función recibe dos parámetros para operar.
6  *
7  * @author Jesús Bono
8  *
9  * @version 2.0
10 *
11 */
12
13 /**
14  * @param Integer $num1 primer sumando.
15  * @param Integer $num2 segudno sumando.
16  * @return Integer Devuelve la suma entre el primer y segundo número.
17  */
```