



Programación de Microcontroladores ARM

CETAM - PUCP



Sesión 2 - 02/07/2024:

- Lenguaje de programación C
 - Variables
 - Funciones
 - Arreglos y punteros
- Sistema de control de versiones GIT.
- Github
- Aplicaciones Git-Github



Lenguaje de programación C

Programación de microcontroladores ARM - Sesión 2



Hola mundo en C

En este primer programa evaluaremos la forma de programar en C, así como el uso de variables y las herramientas de compilación

```
1  int main(void)
2  {
3      printf("Hello World")
4  }
```



Generación de un archivo ejecutable

Para poder ejecutar un archivo en C se debe ejecutar la instrucción "Make" para compilar el archivo en C y obtener el archivo ejecutable.

```
1  int main(void)
2  {
3      printf("Hello World")
4  }
```



Etapas de compilación

Las etapas de generación del archivo ejecutable son las siguientes:

- Pre-processing
- Compilación
- Ensamblaje
- Enlazamiento

```
> nm execute
```

```
000000000040039c r __abi_tag
0000000000401161 T add
0000000000404030 B __bss_start
0000000000404030 b completed.0
0000000000404020 D __data_start
0000000000404020 W data_start
0000000000401070 t deregister_tm_clones
00000000004010e0 t __do_global_dtors_aux
0000000000403e08 d __do_global_dtors_aux_fini_array_entry
0000000000404028 D __dso_handle
0000000000403e10 d _DYNAMIC
0000000000404030 D _edata
0000000000404038 B _end
00000000004011e8 T _fini
0000000000401110 t frame_dummy
0000000000403e00 d __frame_dummy_init_array_entry
0000000000402154 r __FRAME_END__
0000000000404000 d _GLOBAL_OFFSET_TABLE_
                                w __gmon_start__
```

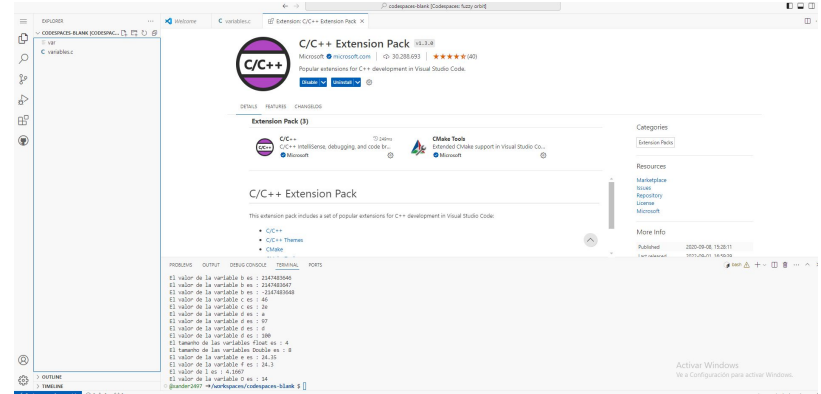


Github codespaces

En esta sesión y las clases posteriores se utilizará Github Codespaces para poder generar los archivos ejecutables y realizar los procesos de compilación. El enlace para acceder es el siguiente :

- <https://github.com/codespaces/>

Este editor requiere del uso de una cuenta de github y permite el acceso a una terminal y al entorno gráfico de programación de visual studio.



Tipos de variables en C

En C se tiene los siguientes tipos de datos:

- char - 1 Byte
- short - 2 bytes
- int - 4 Bytes
- long - 8 Bytes
- float - 4 Bytes
- double - 8 Bytes

```
int Numero = -20;
```

```
int Edad = 15;
```

```
unsigned int num = 230;
```

```
long var = -2356854;
```

```
unsigned long distancia = 48526324;
```

```
float peso = 62.50;
```

```
char character = 'a';
```



Librería stdint

En C se tiene la librería estándar int que define tipos de datos numéricos de longitud fija que no dependen de la arquitectura del procesador, los principales son los siguientes:

- uint8_t - 1 Byte
- uint16_t - 2 Byte
- uint32_t - 4 Byte
- uint8_t - 1 Byte
- int16_t - 2 Byte
- int32_t - 4 Byte



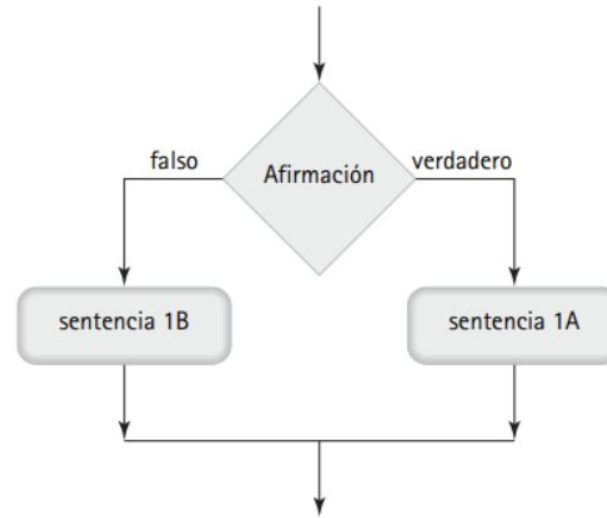
Operadores aritméticos

Operador	Nombre	Ejemplo	Resultado
+	Suma	$10 + 5$	15
-	Sustracción	$10 - 5$	5
*	Multiplicación	$2 * 2$	4
/	Division	$5.0 / 2.0$	2.5
%	Modulo	$10 \% 3$	1



Sentencias selectivas

Las sentencias selectivas comprueban la veracidad o falsedad de una declaración booleana y ejecutan líneas de código dependiendo del resultado



Operadores booleanos

Los operadores booleanos dan como resultado un valor booleano y son utilizadas en sentencias selectivas e iterativas

Operadores lógicos y relacionales

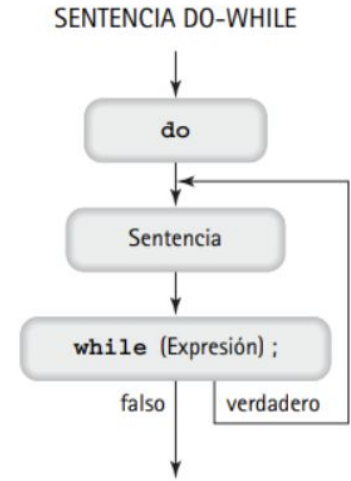
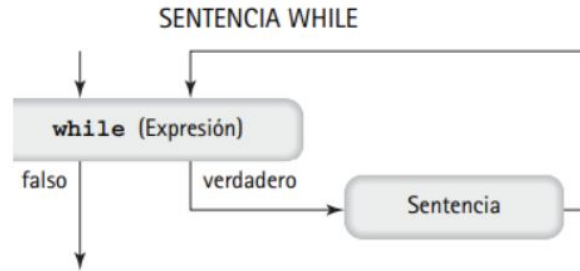
Operador	Nombre	Ejemplo	Resultado
&&	And		
	Or		
!	Negación		
==	Es igual a	3 == 3 3 == 4	True False
!=	No es igual a	3 != 4	True
<	Es menor que	2 < 1	False
<=	Es menor o igual	4 <= 4	True
>	Es mayor que	3 > 2	True
>=	Es mayor o igual que	5 >= 4	False



Sentencias iterativas

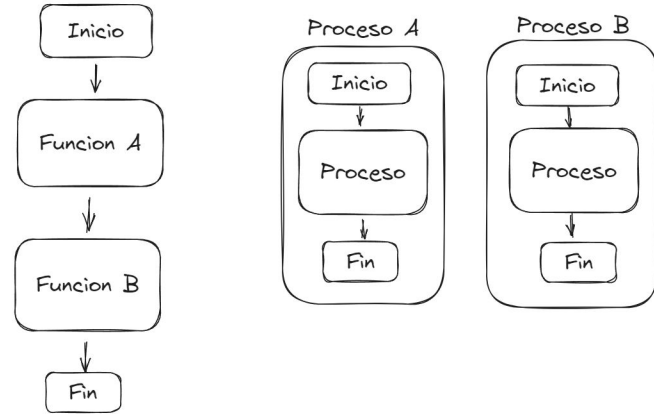
Las sentencias iterativas permiten recorrer ciclos o bucles a través de la comprobación de una sentencia booleana. Las principales sentencias utilizadas para bucles son:

- for
- while
- do-while



Funciones y procedimientos en C

Las funciones y métodos permiten la modularidad y reutilización en el código C por lo que se recomienda su uso. La declaración de funciones debe realizarse de forma adecuada para que el compilador no produzca errores.



Arreglos y Vectores en C

Los vectores en C se definen como una colección de datos de un tipo en específico, el cual es declarado al momento de inicializar un arreglo.

- <tipoDato> <nombre>[<cantidad>]
- int arregloNumeros[50];

```
/* Declaración de un array. */  
  
#include  
  
main() /* Rellenamos del 0 - 9 */  
{  
    int vector[10],i;  
    for (i=0;i<10;i++) vector[i]=i;  
    for (i=0;i<10;i++) printf(" %d",vector[i]);  
}
```

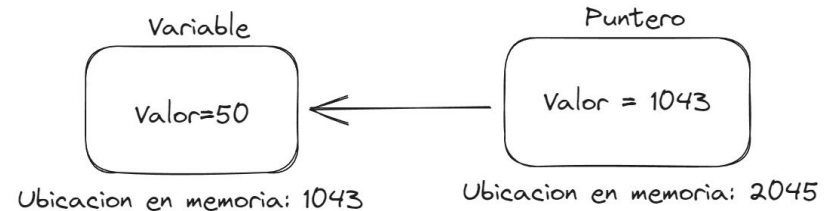


Punteros en C

Los punteros en C permiten almacenar direcciones de memoria permitiendo modificar los parámetros por referencia en el llamado de funciones.

Para la utilización de esta estructura se requiere los siguientes operadores:

- * - Apunta a ...
- & - Dirección de ...



Estructuras en C

A partir de la definición de punteros se pueden crear estructuras en C, en las cuales se encapsulan diferentes variables de diferentes tipos. Estas estructuras nos permiten definir un conjunto de bytes a partir de un nombre común.

```
struct keyword  structure tag  
    ↖          ↗  
struct Student  
{  
    char *name;  
    int std_id;  
    float test_score;  
};
```

} members





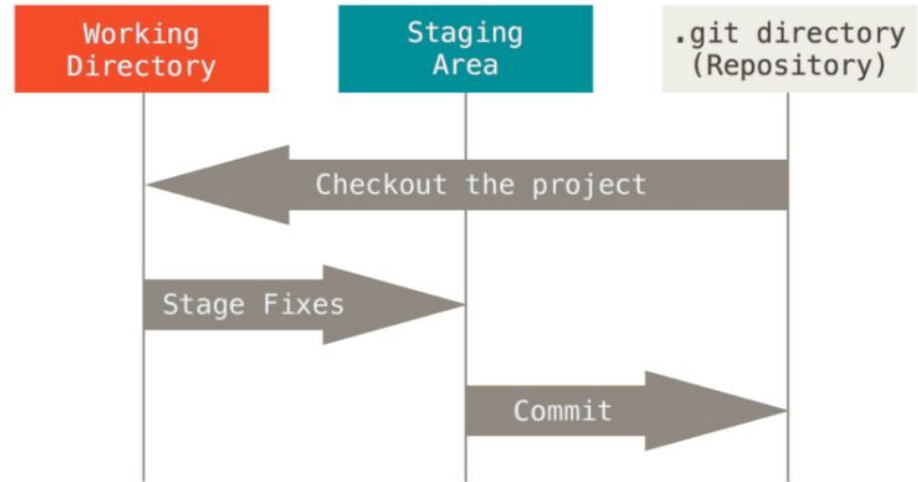
Sistema de control de versiones GIT

Programación de microcontroladores ARM - Sesión 2

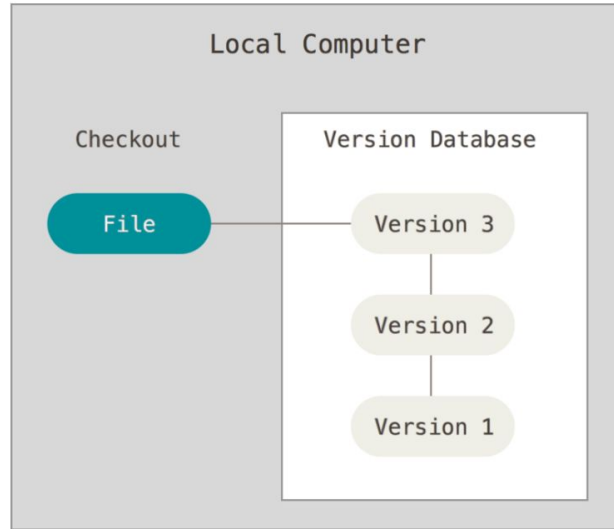


¿Qué es GIT?

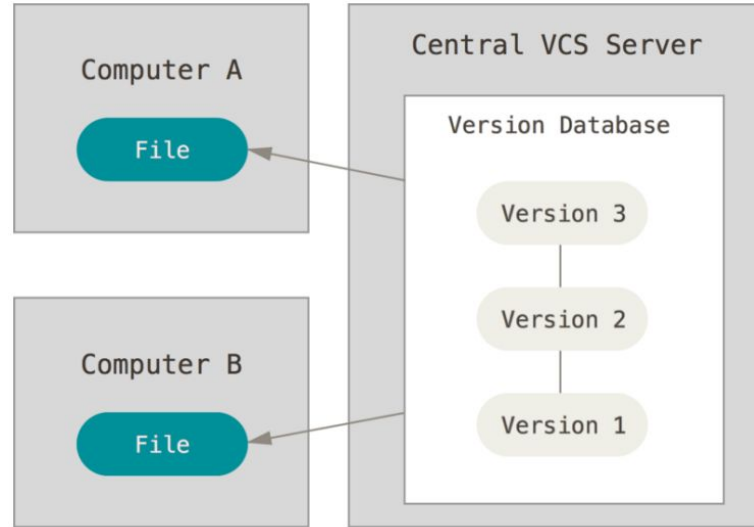
Es un sistema de control de versiones que permite mantener el historial de cambios entre los archivos de un determinado proyecto. Esta característica permite que los cambios realizados puedan ser revertidos en caso se detecten errores.



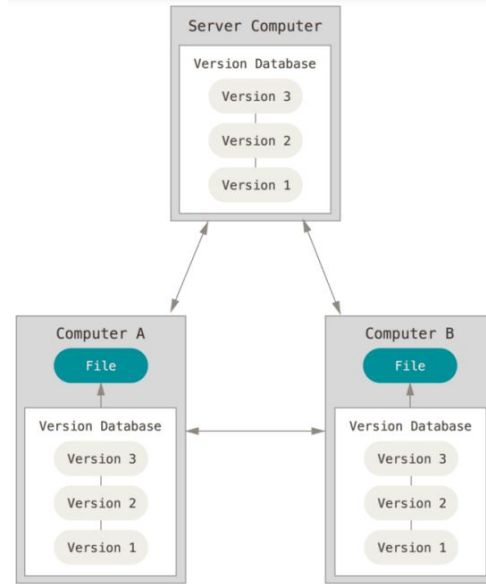
Sistema local de control de versiones



Sistema centralizado de control de versiones

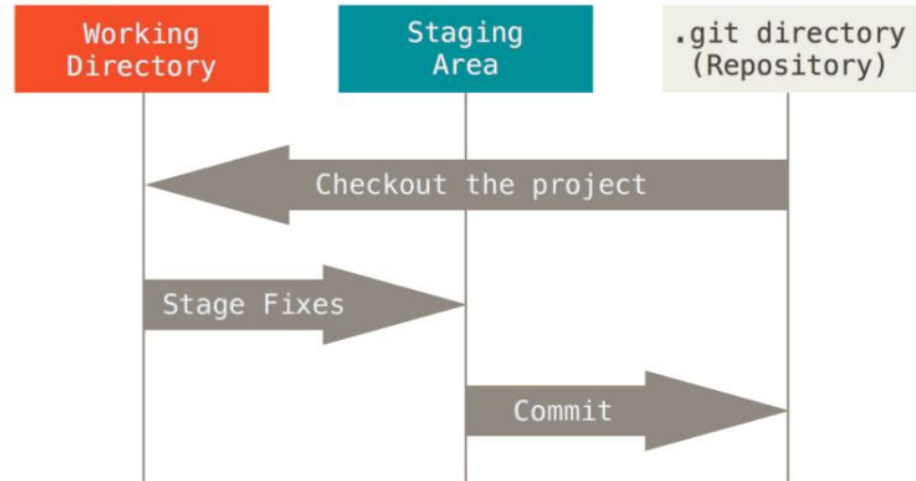


Sistema distribuido de control de versiones



Estados de un repositorio GIT

- Archivos no trackeados.- Son archivos de los cuales no se lleva un registro de sus versiones y son ignorados por el sistema
- Archivos agregados.- Son archivos colocados en el área de carga, poseen modificaciones pero aún no se encuentran trackeados
- Archivos trackeados.- Son archivos del cual se lleva el registro de sus versiones; todos los archivos agregados pasan a este estado luego de un "commit".



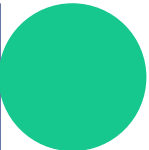
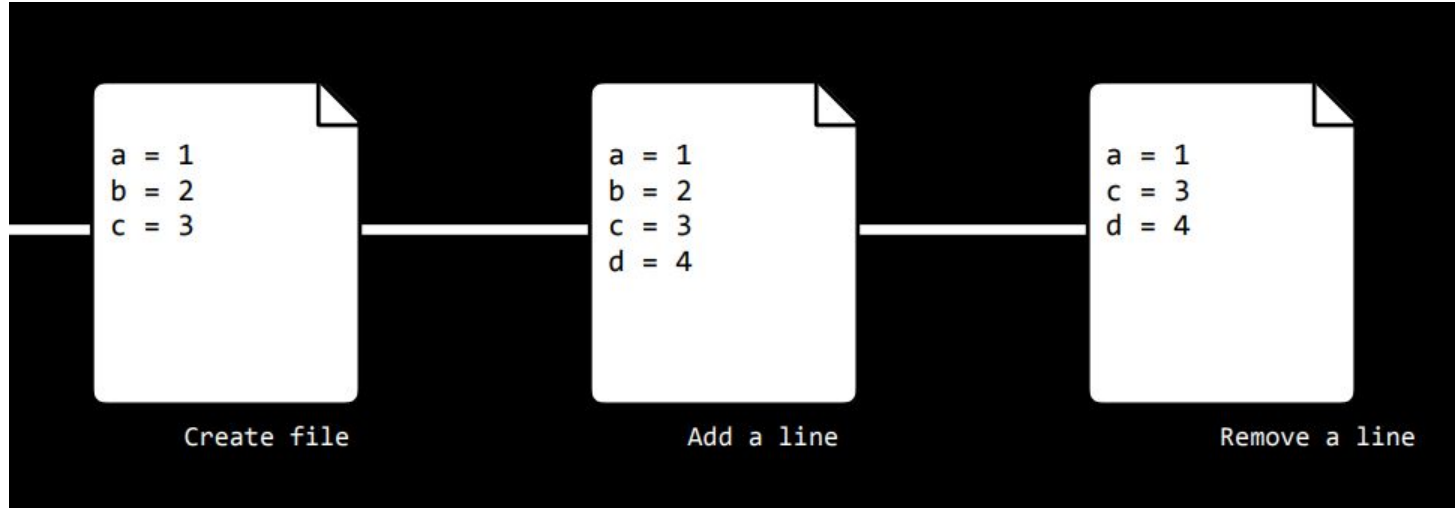
Características de GIT

Dentro de las características de git tenemos lo siguiente:

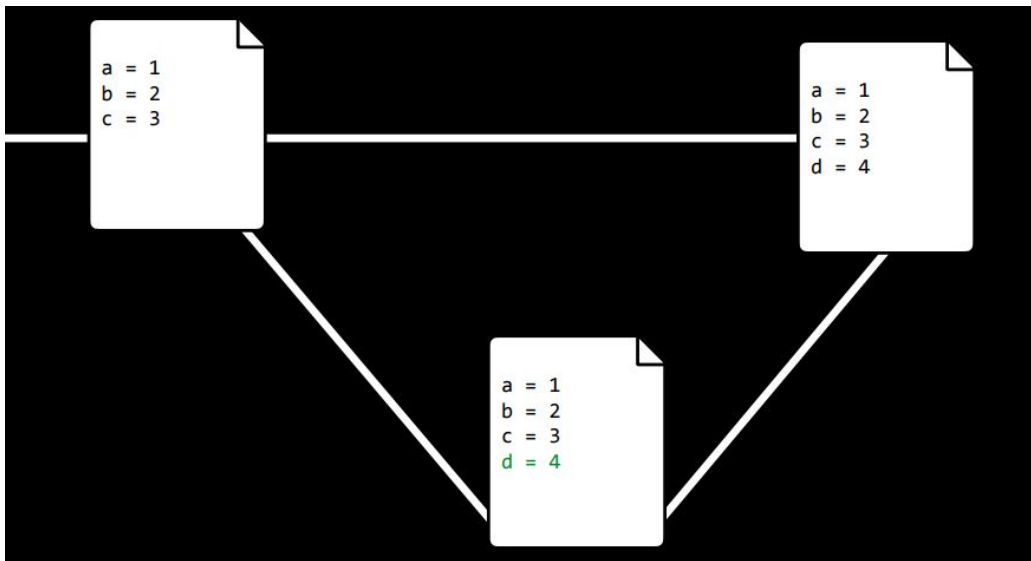
- Verificar los cambios entre diferentes versiones
- Sincronizar los cambios entre diferentes desarrolladores
- Crear ramas de experimentación del proyecto
- Regresar a versiones anteriores del proyecto



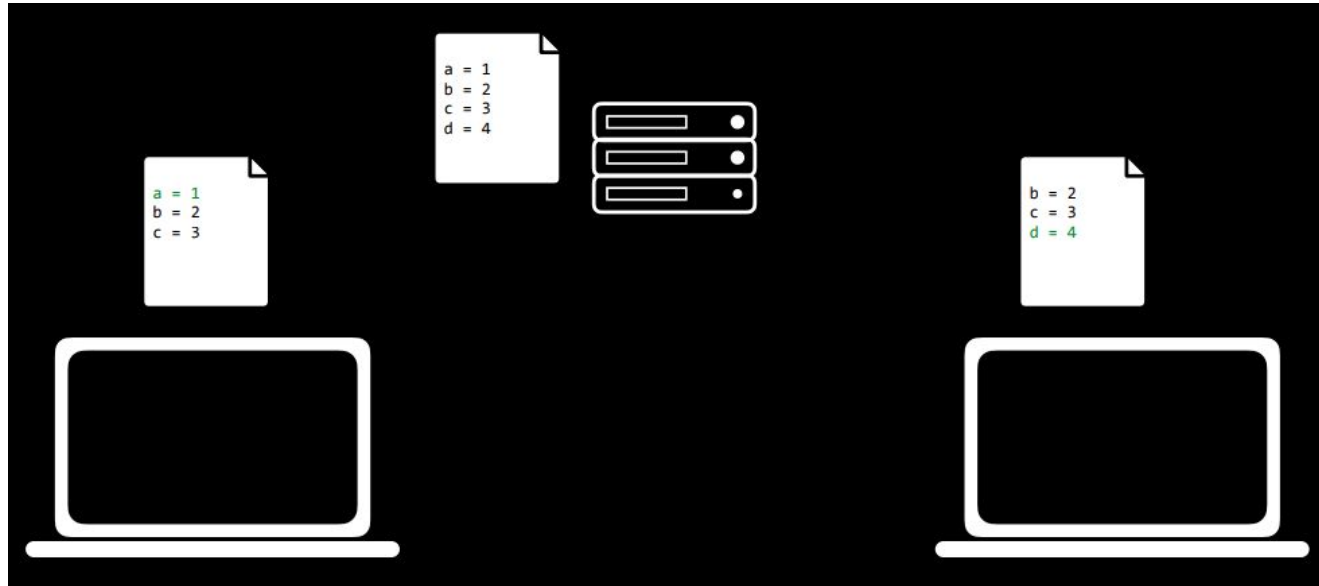
Verificar los cambios de código



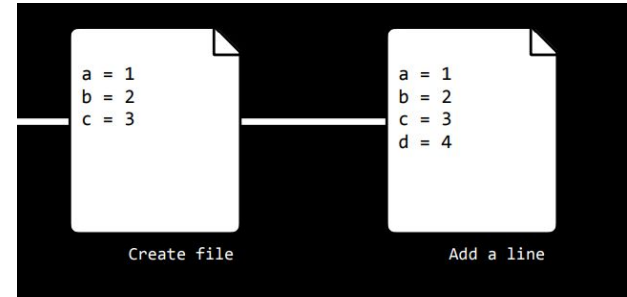
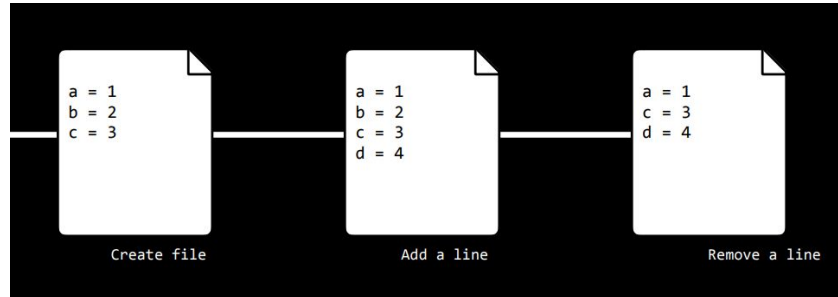
Verificar cambios en el código sin perder la versión original



Sincronizar cambios entre diferentes desarrolladores



Regresar a versiones anteriores del código



Principales aplicaciones de Git

Dentro de las características de git tenemos lo siguiente:

- Verificar los cambios entre diferentes versiones
- Sincronizar los cambios entre diferentes desarrolladores
- Crear ramas de experimentación del proyecto
- Regresar a versiones anteriores del proyecto





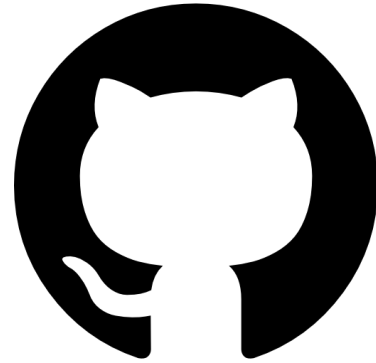
Github

Programación de microcontroladores ARM - Sesión 2



¿Qué es Github?

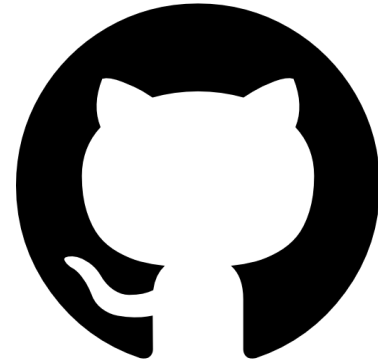
Github es un servidor de almacenamiento para repositorios de git, en donde se almacenan los diferentes proyectos realizados. Además posee un herramientas gráficas para revisar las diferentes versiones de un repositorio



Repositorios en Github

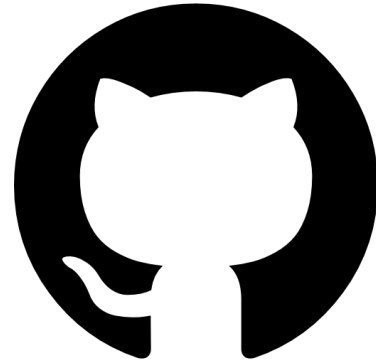
Verificando algunos repositorios notables:

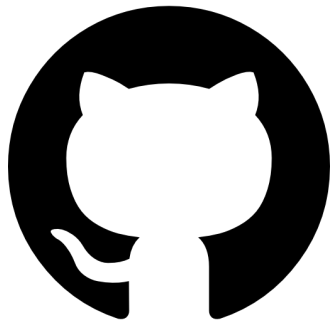
- <http://rafalab.github.io/>
- <https://github.com/dmalan>



Git push y Git clone

Los comandos Git Push y Git Clone permiten extraer y subir los cambios realizados a un repositorio de github. Cada repositorio posee en su configuración un origen remoto al cual se hace la petición al momento de clonarlo o actualizarlo.





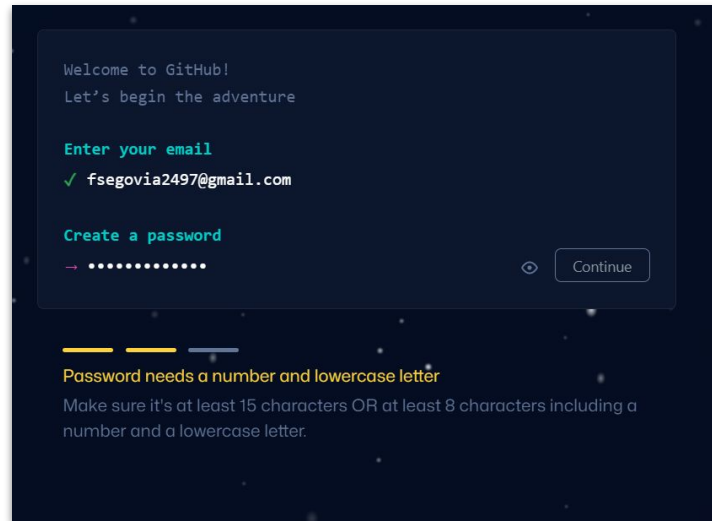
Aplicaciones Git y Github

*Programación de microcontroladores ARM -
Sesión 2*



Crear una cuenta en github y generar el token

- Ingresar nuestro Email y crear un password

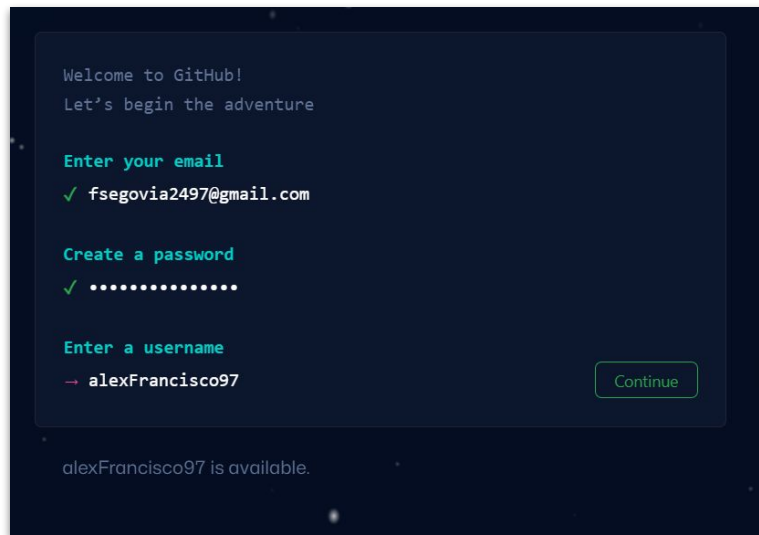


The screenshot shows the GitHub account creation page. At the top, it says "Welcome to GitHub! Let's begin the adventure". Below this, there are two sections: "Enter your email" and "Create a password". The email section shows a green checkmark and the email address "fsegovia2497@gmail.com". The password section shows a red checkmark and a series of dots representing the password. To the right of the password field is a "Continue" button. Below the password field, there is a progress bar with three segments, the first two of which are yellow. Below the progress bar, there is a warning message: "Password needs a number and lowercase letter". Below the warning message, there is a note: "Make sure it's at least 15 characters OR at least 8 characters including a number and a lowercase letter."



Crear una cuenta en github y generar el token

- Ingresar nuestro Email, generar una contraseña y un nombre de usuario



Wellcome to GitHub!
Let's begin the adventure

Enter your email
✓ fsegovia2497@gmail.com

Create a password
✓

Enter a username
→ alexFrancisco97

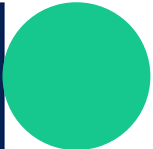
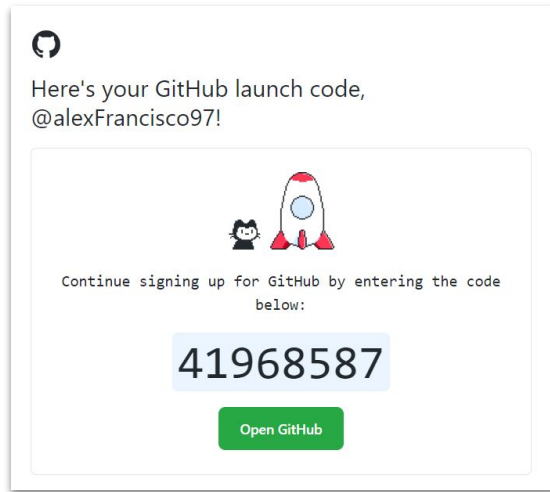
Continue

alexFrancisco97 is available.



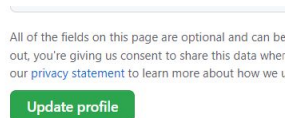
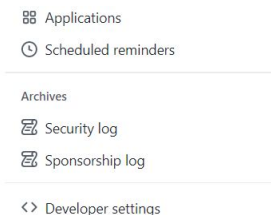
Crear una cuenta en github y generar el token

- Confirmar el correo electrónico



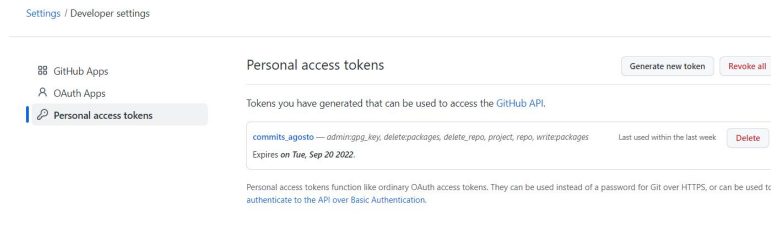
Crear una cuenta en github y generar el token

- En configuración ingresar a 'Opciones de desarrollador' y 'Generar nuevo token'



Contributions & Activity

- ☐ **Make profile private and hide activity** ⓘ
Enabling this will hide your contributions and activity from leaderboards and releases.
- ☐ **Include private contributions on my profile** ⓘ
Your contribution graph, achievements, and activity will be visible on your profile.



Crear y trackear un repositorio

- git init #Inicializa un repositorio
- git log #Muestra el historial de versiones del repo
- git status #Muestra el estado de los archivos en el repo
- git add #Agrega archivos al área de pre-carga
- git commit #Crea una nueva versión del repositorio
- git clone #Crea una copia local de un repositorio remoto

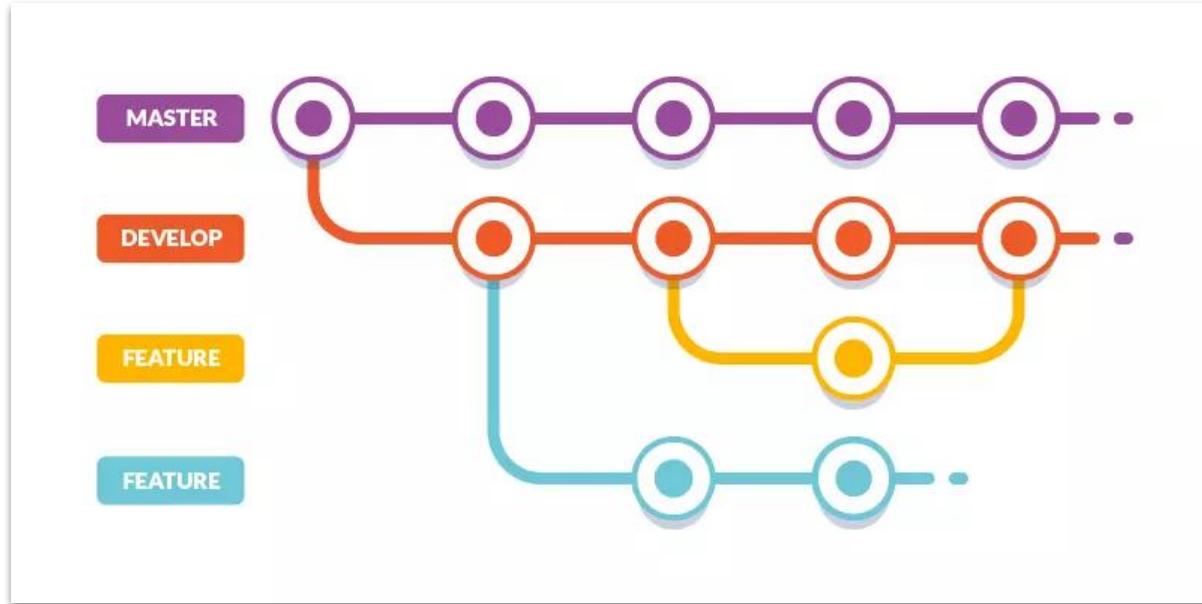


Comandos más comunes en Git

- git log #Muestra el historial de versiones del repo
- git status #Muestra el estado de los archivos en el repo
- git clone #Crea una copia local de un repositorio remoto
- git branch #Permite crear diferentes ramas en el repo
- git checkout #Permite visualizar las diferentes versiones
- git reset #Eliminar los cambios posteriores
- git merge #Permite combinar dos ramas diferentes



Ramas en un repositorio



Subir cambios a un repositorio remoto

- `git remote` `#Visualiza las fuentes remotas`
- `git remote add <url>` `#Permite agregar una fuente remota al repo`
- `git push origin master` `#Empuja los cambios locales al repo remoto`



git

