**Universidad Mariano Gálvez**

**Chiquimula**

Séptimo Semestre

Ingeniería en Sistemas de Información y Ciencias de la Computación

Compiladores

Ing. Carlos Armando Escobar Garza

**Git y Github**

Concepción de María Monroy Aquino

1390-12-4393

Chiquimula 11/03/2017

**Qué es git**

Es un software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. Al principio, Git se pensó como un motor de bajo nivel sobre el cual otros pudieran escribir la interfaz de usuario o front end como Cogito o StGIT. 3 Sin embargo, Git se ha convertido desde entonces en un sistema de control de versiones con funcionalidad plena.

Git fue creado pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente, es decir Git nos proporciona las herramientas para desarrollar un trabajo en equipo de manera inteligente y rápida y por trabajo nos referimos a algún software o página que implique código el cual necesitemos hacerlo con un grupo de personas.

**Que es github**

Es una forja (plataforma de desarrollo colaborativo) para alojar proyectos utilizando el sistema de control de versiones Git. Utiliza el framework Ruby on Rails por GitHub, Inc. (anteriormente conocida como Logical Awesome). Desde enero de 2010, GitHub opera bajo el nombre de GitHub, Inc. El código se almacena de forma pública, aunque también se puede hacer de forma privada, creando una cuenta de pago.

GitHub es una plataforma de desarrollo colaborativo de software para alojar proyectos utilizando el sistema de control de versiones Git. GitHub aloja tu repositorio de código y te brinda herramientas muy útiles para el trabajo en equipo, dentro de un proyecto.

Además de eso, puedes **contribuir a mejorar el software de los demás.** Para poder alcanzar esta meta, GitHub provee de funcionalidades para hacer un **fork** y solicitar **pulls.**

**Diferencias entre GIT y GitHub**

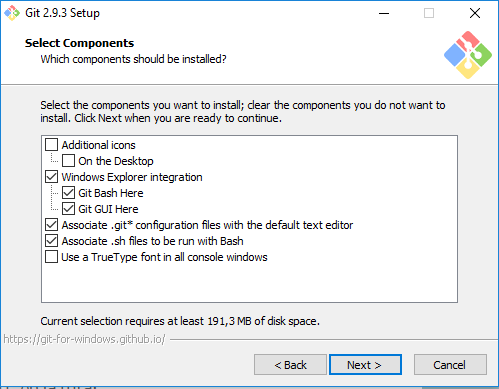
[GIT](http://git-scm.com/) es el software que rastrea. El sistema de control de versiones. La herramienta que utilizaremos en la terminal.

[GitHub](http://github.com/) es la plataforma de "hosting" de los proyectos. Una comunidad llena de personas que desarrollan y comparten, usando GIT.

Se complementan, pero son personajes independientes.

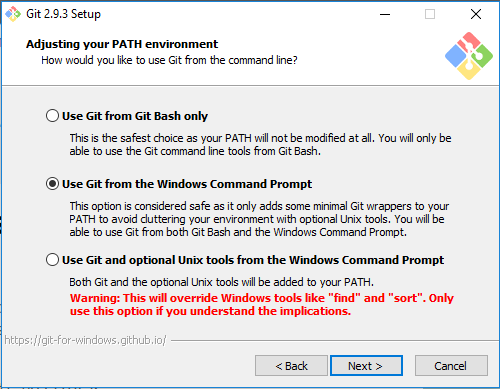
**Cómo instalar Git en Windows**

Lo primero será acceder a la página oficial de Git para Windows (https://git-for-windows.github.io/) y descargarnos el ejecutable pulsando en Download.

Cuando se lance la instalación y después de haber aceptado la licencia de términos de uso y haber elegido una ruta donde instalarlo, veremos la pantalla de los componentes a instalar.

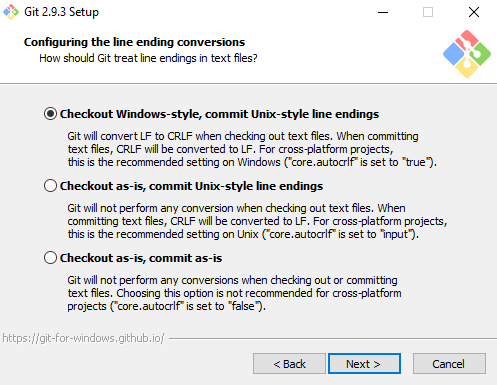
Deja todo por defecto. De esta manera se instalará Git en dos formatos, consola e interfaz gráfica (aunque personalmente esta última no me gusta mucho) y se asociarán los archivos de configuración. git a el editor de texto por defecto, además de poder ejecutar los .sh con la consola de Bash. Sí, aunque no tengamos Linux Git lo trabajaremos con Bash.

Pulsamos Next y pasaremos a la sección donde configuraremos como usar la consola de comandos de Git.

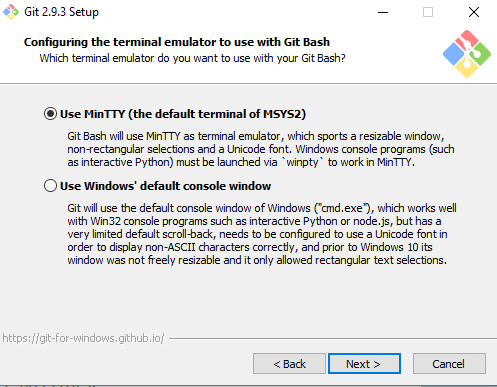


En este caso yo quiero usar **Git** tanto desde **Bash**como desde **CMD**, así que elijo la segunda opción. Si quieres usarlo únicamente desde **Bash**, deberás marcar la primera.

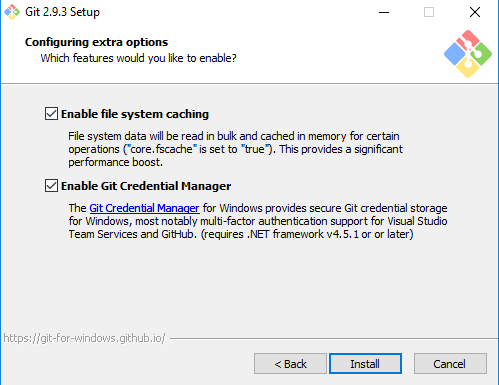
De nuevo le damos a *Next*y procederemos a configurar como tratará **Git**los finales de línea en los archivos de texto.



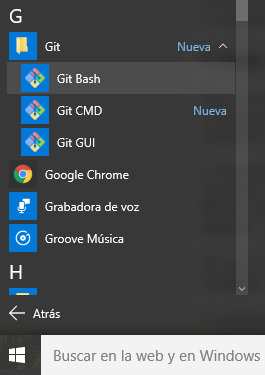
Dejamos la opción por defecto simplemente, para no complicarnos. Pulsamos *Next* para elegir que emulador queremos para poder usar Bash.



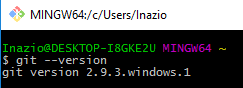
Elijo MinTTY, la terminal por defecto. Es más sencillo de configurar y no tiene tantas limitaciones.  
  
Y por último habilito las dos opciones extra que me ofrece **Git**, habilitar el cache en los archivos de sistema y el administrador de credenciales.

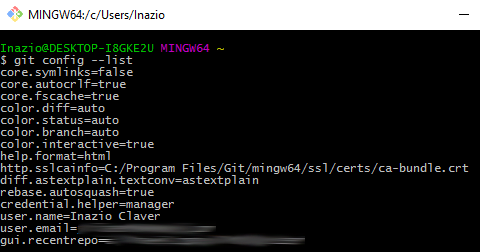


Con estos pasos, y después de pulsar *Install*, ya tendremos listo en nuestra máquina **Git**. Vayamos a la segunda parte, su configuración básica.  
  
**Cómo configurar Git windows**  
  
Una vez que tenemos instalado **Git**, vamos a querer hacer algunas configuraciones para personalizar su entorno. Sólo necesitamos configurarlo una vez, aunque podremos cambiarlo en cualquier momento simplemente volviendo a ejecutar los mismos comandos.  
La herramienta que usaremos se llama git config, que nos permitirá obtener y establecer variables de configuración para el aspecto y funcionamiento de **Git**.  
Dichas variables se almacenarán en sistemas Windows en C:\Users\$USUARIO en un archivo llamado .gitconfig.  
Yo me voy a centrar en configurar mi identidad, pero hay otras muchas cosas que se pueden modificar.  
  
Lo primero será abrir nuestro **Git Bash** recién instalado.



Se nos abrirá una nueva terminal, y ya que estamos vamos a comprobar que versión de **Git** tenemos instalada escribiendo git –versión



Bien, hecho esto vamos a indicarle nuestro nombre y correo electrónico. Esta parte es importante porque todos los **commits** de **Git** usarán esta información.  
  
Escribiremos lo siguiente  
  
git config --global user.name "Inazio Claver"  
git config --global user.email inazio@programando.apasitos  
  
Con la variable --global **Git** usará esta información para todo el sistema. Si en un momento determinado necesitas cambiar esta información para un proyecto en concreto, basta con lanzar el mismo comando sin esa variable cuando estés dentro del proyecto.  
  
Aparte de eso también puedes configurar tu editor de texto por defecto con el siguiente comando:  
  
git config --global core.editor MiEditor  
  
Y la herramienta para diferencias por defecto, que usaremos para resolver los conflictos en los **merge** (cuando unamos las ramas). Se hace así  
  
git config --global merge.tool MiHerramienta  
  
Ahora sólo nos queda comprobar la configuración. Escribe el siguiente comando  
  
git config --list  
  
Veremos algo así

Y esto sería una configuración básica para **Git**. Ya estamos listos para usar nuestra herramienta de control de versiones.

**instalar y configurar Git en GNU/Linux**

Primero abrimos un terminal y actualizamos la lista de paquetes:

sudo apt-get update

sudo apt-get upgrade

Ahora instalamos Git:

sudo apt-get install git

Configuramos nuestro nombre de usuario:

git config --global user.name "usuario"

Configuramos nuestra dirección de e-mail:

git config --global user.email "usuario@email.com"

Creamos una llave SSH:

ssh-keygen -t rsa -C "usuario@email.com"

Copiamos la clave SSH en nuestra cuenta en Git <https://github.com/settings/ssh>:

cat ~/.ssh/id\_rsa.pub

Controlamos si esta correcta:

ssh -T git@github.com

Recibiremos un mensaje diciendo que autenticamos nuestra cuenta pero que Git no provee acceso shell.