

# DEPARTAMENTO DE SISTEMAS Y COMPUTACIÓN

# INGENIERIA EN SISTEMAS COMPUTACIONALES

# NOMBRE DE LA MATERIA:

***Ingeniería de Software 20213-6506B / 13:00-14:00***

***Docente: Euri Salgado Escobar***

***Unidad 3 - Desarrollo***

**Práctica: P2 Instala Proyecto**

**Alumno: Álvarez Ortiz Juan Manuel 18070519**

**Fecha de entrega: 2 – Nov – 2021**

**PRÁCTICA No. 2**

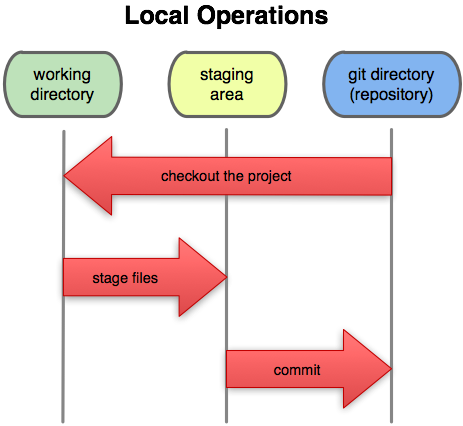
|  |  |  |  |
| --- | --- | --- | --- |
| Departamento Académico | Sistemas y Computación | | |
| Programa Académico | Plan de Estudios | Formato | |
| Ing. en Sistemas Computacionales | ISIC-2010-224 | SyC-03-2016-P | |
| Asignatura | Clave de la Asignatura | Créditos SATCA | Semestre |
| Ingeniería de software | SCD-1011 | 2-3-5 | 7mo |

|  |  |
| --- | --- |
| No. de Práctica | Nombre de la Práctica |
| P\_01\_02 | Inicialización de un proyecto en Git |
| Competencia de la Práctica | |
| Conoce cómo iniciar un nuevo proyecto en Git | |

1. **INTRODUCCIÓN**

**Ciclo de trabajo en Git**

Para iniciar un repositorio, o sea, activar el sistema de control de versiones de Git en tu proyecto, solo debes ejecutar el comando git init.

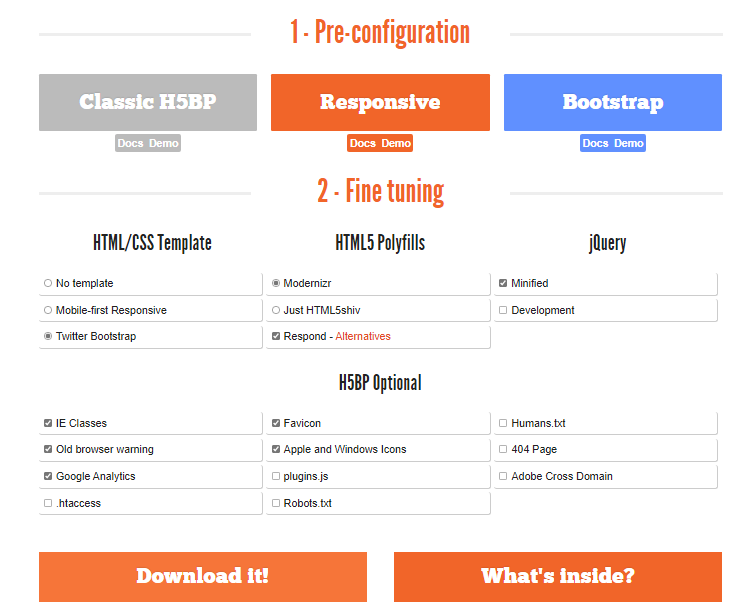
Este comando se encargará de dos cosas: primero, crear una carpeta**.git,** donde se guardará toda la base de datos con cambios atómicos de nuestro proyecto; y segundo, crear un área que conocemos como **Staging**, que guardará temporalmente nuestros archivos (cuando ejecutemos un git add) y nos permitirá, más adelante, guardar estos cambios en el repositorio (al utilizar el comando git commit). Visto de una manera más fácil, el staging es el mostrador de recepciones en la biblioteca, usted coloca los archivos allí para que el bibliotecario los prepare para archivarlos en la biblioteca (carpeta .git).

El **directorio de trabajo (working directory)** es una copia de una versión del proyecto. Estos archivos se sacan de la base de datos comprimida en el directorio de Git, y se colocan en disco para que los puedas usar o modificar.

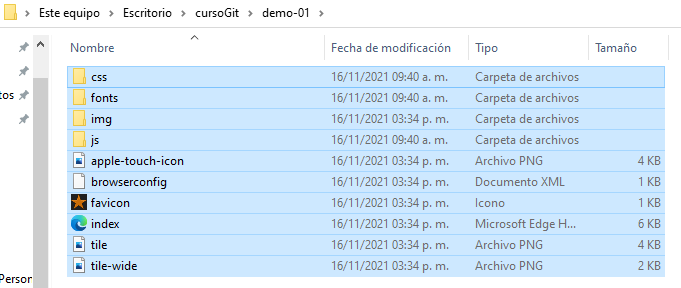
1. **REQUERIMIENTOS**

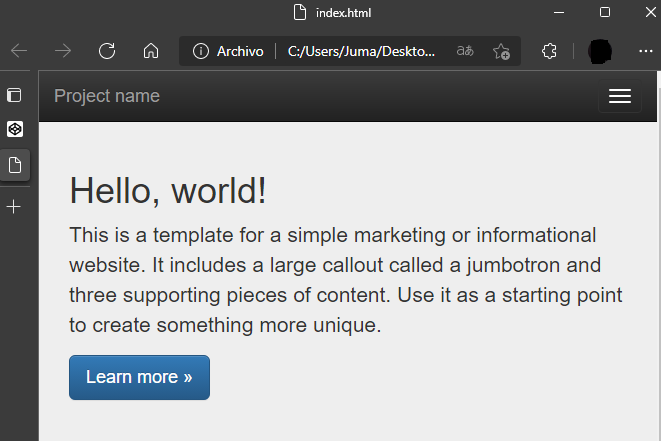
|  |  |
| --- | --- |
| Equipo | Software |
| Computadora con sistema operativo compatible con Git. | -Software Git.  -Editor de texto (Bloc de notas, Notepad++, Atom, Sublime Text). |

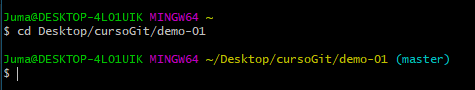
1. **DESCRIPCIÓN DE LA PRÁCTICA.**
2. Para realizar esa práctica, se recomienda tener instalado un editor de texto como Sublime Text 3 o Atom, pues cuando se requiere abrir y/o editar un archivo, si no se tiene instalado un editor de texto, los archivos de editan en Vim. Para iniciar la práctica, entrar a la carpeta ‘cursoGit’ creada en la práctica 1 y dentro de esta carpeta crear otra carpeta llamada ‘demo-01’. Escritorio > cursoGit > demo-01
3. Entrar a la página <http://www.initializr.com/>. En la página indicada,elegir la opción de Bootstrap y descargarla. Dejar las opciones seleccionadas por default



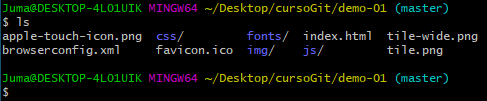
1. Extraer la carpeta descargada y copiar su contenido en ‘demo-01’.



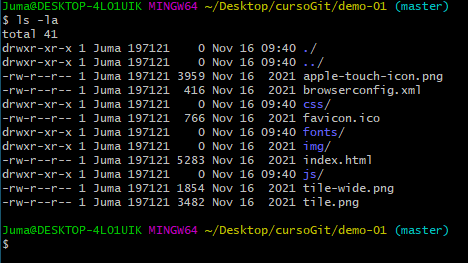
1. En esta carpeta se realiza la práctica número 2. Visualizar en el navegador Google Chrome (o en cualquier otro navegador deseado) el archivo index.html (desde el explorador de Windows, entrar a la carpeta de trabajo actual y dar clic en index.html para visualizar la página web)
2. Abrir Git bash y utilizar el comando cd Desktop/cursoGit/demo-01 para entrar a la carpeta demo-01.



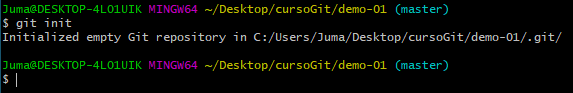
1. **$ ls** Visualizar el contenido de la carpeta con el comando ls



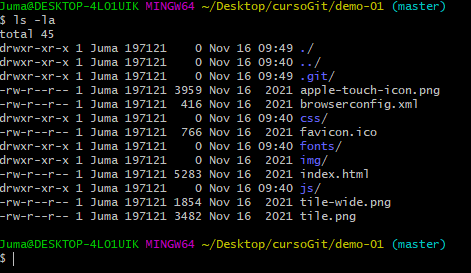
1. **$ ls -la** Visualiza el contenido de la carpeta de trabajo actual, mostrando archivos ocultos.



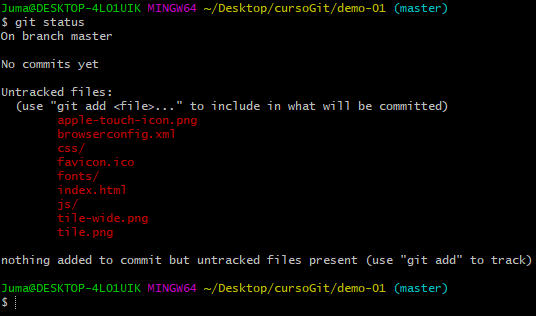
1. **$ git init** Utilizar el comando git init para inicializar un nuevo proyecto al que Git le dará seguimiento.



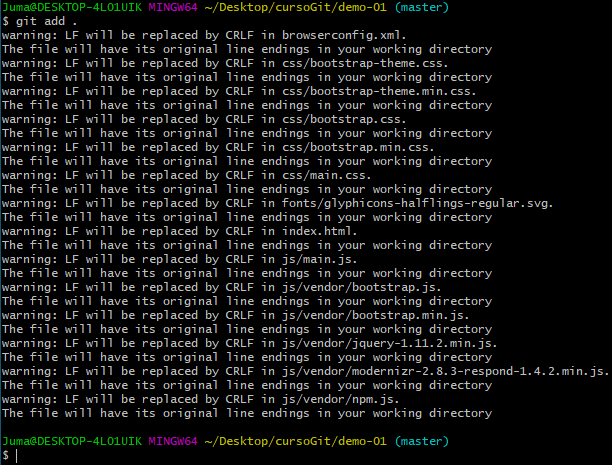
1. **$ ls -la** Volver a aplicar el comando ls -la para verificar que se haya creado la carpeta .git.

****

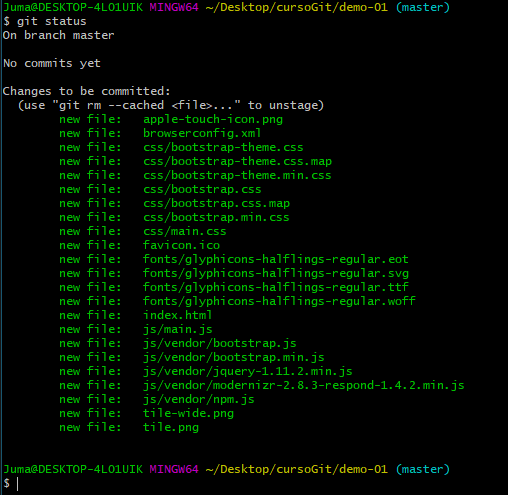
1. **$ git status** Utilizar git status para ver los archivos que aún no están siendo gestionados por Git. Muestra en color rojo las carpetas y archivos que no se les está dando seguimiento.



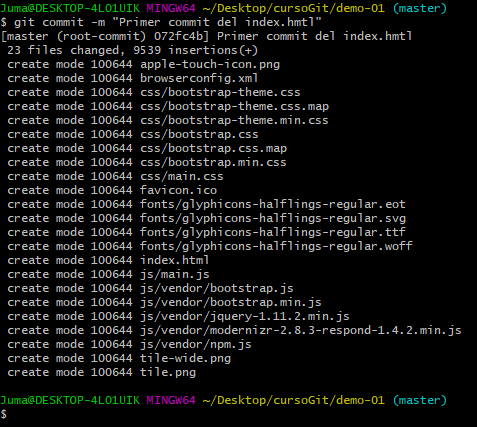
1. **$ git add .** Con el comando git add . añadir todos los archivos para que Git les de seguimiento. Es decir, agrega al escenario los archivos especificados.



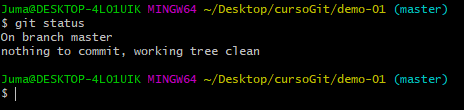
1. **$ git status** Ahora los archivos y carpetas se muestran en color verde. Esto significa que cuando se realice el “snapshot”, estos archivos serán confirmados como parte del seguimiento de Git

****

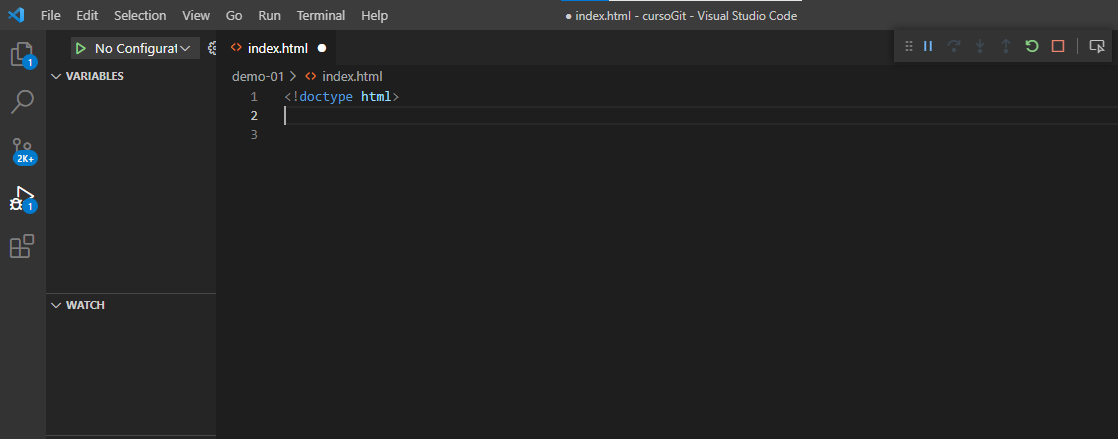
1. **$ git commit -m “Primer commit con información del index.html**” Realizar un commit para guardar el avance, utilizando el comando git commit -m “descripción del commit”, donde -m se refiere al mensaje especificado para el commit. Se realiza un “snapshot” o registro de los archivos confirmados

****

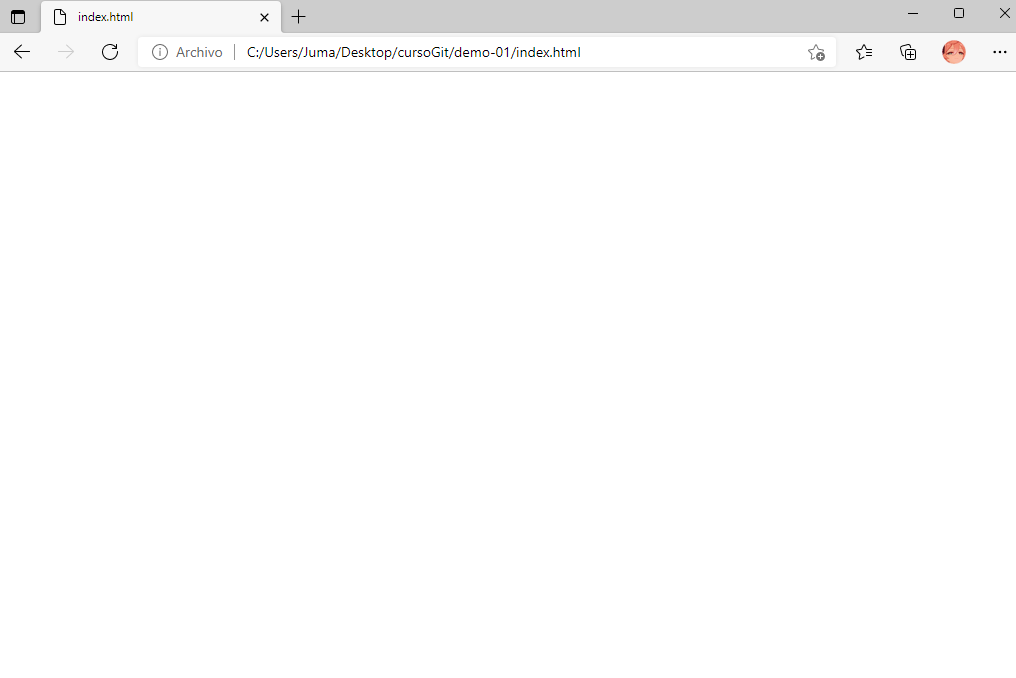
1. **$ git status** Aplicar nuevamente git status para verificar que todos los archivos estén siendo gestionados por Git. Muestra el estado actual del proyecto Git, en este caso muestra el texto “On branch master”, lo cual significa que los archivos modificados forman parte de la rama maestra y todos los cambios que se han realizado hasta el momento están bajo el resguardo de Git. Si se pierde un archivo de trabajo, existe un comando con el cual se puede recuperar el contenido del archivo.



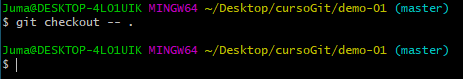
1. Abrir el archivo index.html con un editor de texto y borrar todo el contenido, excepto la primera línea.



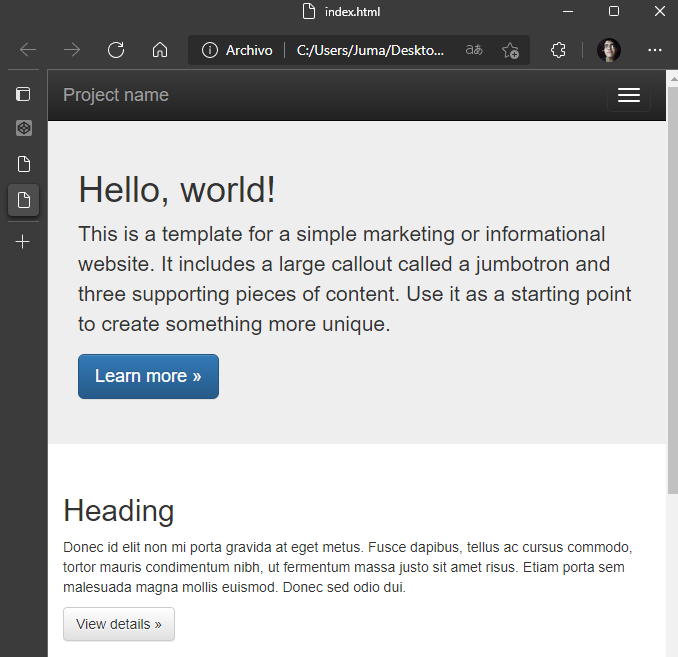
1. Visualizar el archivo index.html en el navegador y verificar que no tiene contenido.



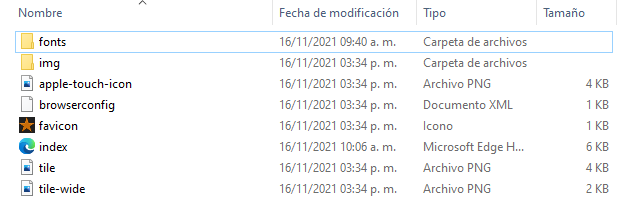
1. **$ git checkout -- .** Utilizar el comando git checkout -- . para recuperar el contenido del index.html.



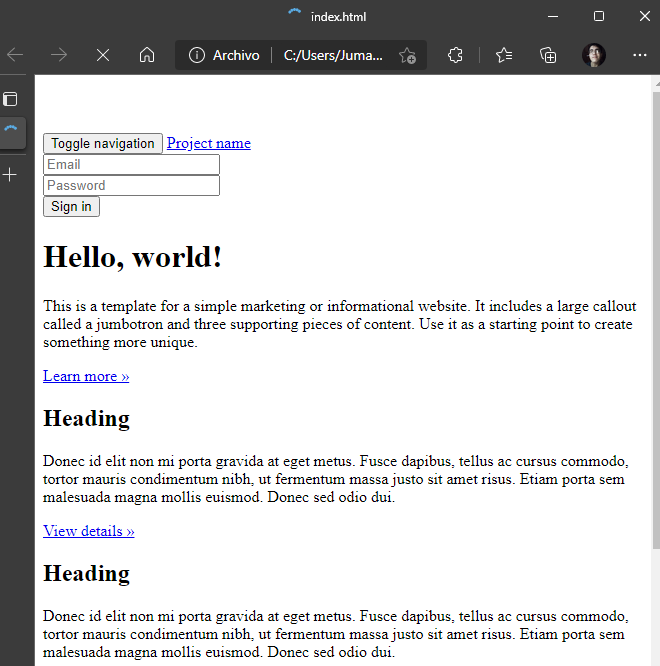
1. Visualizar nuevamente el index.html y verificar que este como antes de borrar su contenido.



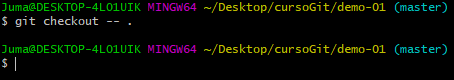
1. Borrar las carpetas css y js.



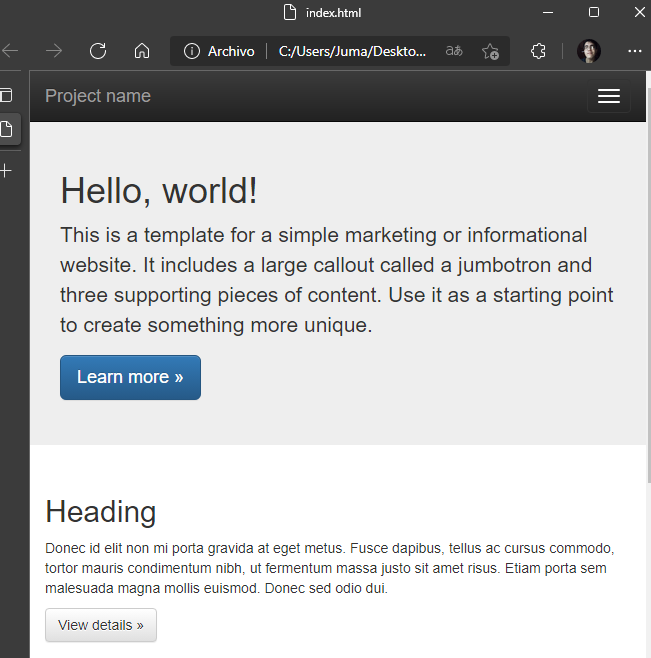
1. Visualizar el index.html y verificar que ya no tiene el estilo que tenía antes.



1. **$ git checkout -- .** Aplicar git checkout -- . para restaurar los archivos eliminados.



1. Visualizar nuevamente el index.html y verificar que tiene el mismo estilo de antes.



Responder las siguientes preguntas:

**Iniciar un proyecto Git**

1. ¿Para qué sirve el comando $ **git init**?

Crea un nuevo repositorio local GIT en el directorio actual

1. ¿Para qué sirve el comando $ **git add .** ?

Se usa para agregar archivos en el área de preparación, por ejemplo un archivo txt:

git add <archivo.txt>

1. ¿Para qué sirve el comando $ **git status**?

Muestra la lista de los archivos que se han cambiado junto con los archivos que se están por preparar o confirmar

1. ¿Para qué sirve el comando $ **git checkout -- .** ?

Crea ramas y ayuda a navegar a través de ellas, por ejemplo:

git checkout -b <branch-name>

1. **FUENTES DE INFORMACIÓN.**

B., G. (2021, 3 agosto). *Comandos de GIT Básicos - Guía Completa*. Tutoriales Hostinger. https://www.hostinger.mx/tutoriales/comandos-de-git