Conociendo las herramientas

Git – Github

¿Qué es Git?

Git es un protocolo de control de versiones. Los sistemas de control de versiones son programas que tienen como objetivo controlar los cambios en el desarrollo de cualquier tipo de software, permitiendo conocer el estado actual de un proyecto, los cambios que se le han realizado a cualquiera de sus piezas, las personas que intervinieron en ellos, etc.

Es **distribuido**, es decir que múltiples personas pueden trabajar en equipo, es **open source** y también se adapta a todo tipo de proyectos desde pequeños hasta grandes, además, se pueden fusionar archivos, guarda una línea de tiempo a lo largo de todo el proyecto; es **multiplataforma**, ya que permite ser utilizado en los principales sistemas operativos.

Nos ayudan en muchos ámbitos fundamentales, como por ejemplo:

- Comparar el código de un archivo, de modo que podamos ver las diferencias entre versiones.
- Restaurar versiones antiguas.
- Fusionar cambios entre distintas versiones.
- Trabajar con distintas ramas de un proyecto, por ejemplo la de producción y desarrollo.

Sistema de control de versiones distribuídos.

Git es un protocolo concebido bajo este paradigma. Anteriormente existían otros tipos de sistemas de versionamiento como los locales (los archivos modificados se guardaban como “parches” en nuestro sistema para integrarlos posteriormente) o centralizados (los archivos se almacenan en un VPS –servidor virtual- y cada desarrollador descarga los archivos para luego integrarlos).

La propuesta de Git consiste en un sistema de control distribuído. Los clientes no sólo descargan la última instantánea de los archivos: replican completamente el repositorio. Así, si un servidor muere, y estos sistemas estaban colaborando a través de él, cualquiera de los repositorios de los clientes puede copiarse en el servidor para restaurarlo. Cada vez que se descarga una instantánea, en realidad se hace una copia de seguridad completa de todos los datos.

¿Cómo es el flujo de trabajo en Git?

Todo el código fuente del desarrollo se encuentra comprendido en un contenedor online denominado “Repositorio”. En el mismo, se irán agregando los diferentes avances, denominados commits, en los cuales se incluirán las modificaciones realizadas en el código.

Cada modificación que se hace en el código, se almacena en un Blob, que básicamente es una secuencia de bytes conteniendo la modificación realizada en el archivo de manera incremental.

El repositorio local está compuesto por tres "árboles" administrados por git. El primero es tu Directorio de trabajo que contiene los archivos, el segundo es el Index que actua como una zona intermedia, y el último es el HEAD que apunta al último commit realizado.

Cada cambio se confirma añadiéndolo al INDEX y realizandóle un commit para su confirmación. Cuando ya commiteamos un archivo, podemos pushearlo al repositorio.

A la par de los commits, la estructura del repositorio está organizada en ramas o branchs, las mismas son apuntadores que indican ciertos estadíos del desarrollo de manera organizada. Por ejemplo, cuando vamos a introducir una nueva funcionalidad en el sistema, el proceso correcto es crear una rama particular para comenzar a trabajar en la misma, y posteriormente hacer un merge (integración del código) a la rama principal.

Cuando el desarrollo alcanza un punto óptimo, se definen para el mismo Tags, que tienen como finalidad destacar commits específicos como un reléase, la finalización de la implementación de una nueva finalidad, etc.

¿Qué es GitHub?

GitHub es una de las plataformas de control de versiones más utilizadas del mundo. Utiliza Git como protocolo para la gestión de archivos y permite alojar los mismos en la nube, permitiendo un fácil acceso a todos para el avance en los mismos.

Instalando las herramientas necesarias

Lo primero para comenzar, es tener instalado Git y disponer de cuenta en Github.

Para instalar Git en su escritorio, hay muchas maneras.

Instalar en Windows

Hay un instalador .exe disponible en el siguiente sitio:

Juan Pablo Paillet
Fullstack Developer
http://pailletjp.com

https://gitforwindows.org/

Instalar en Linux

Si quieres instalar Git en Linux a través de un instalador binario, en general puedes hacerlo a través de la herramienta básica de gestión de paquetes que trae tu distribución. Si estás en Fedora, puedes usar yum, ejecutando:

$ yum install git-core

Instalando en Ubuntu / Debian

Si estás en una distribución basada en Debian como Ubuntu, puedes utilizar apt-get:

$ apt-get install git

Instalando en Mac

La forma más sencilla es usar el instalador gráfico de Git, que puedes descargar desde la página de SourceForge:

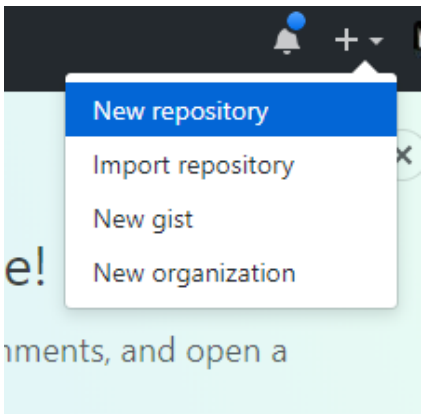
http://sourceforge.net/projects/git-osx-installer/

Crear una cuenta en GitHub

En el caso de aún no disponer de una cuenta, ingrese a https://github.com/ y complete el formulario de registro.

Cuando ya disponga de un usuario, podrá comenzar a configurar repositorios, en los cuales irá subiendo los diferentes avances de de su desarrollo, clonar otros repositorios existentes, y participar de la comunidad de Github.

Comenzando a utilizar GitHub



Para comenzar a desarrollar utilizando Github, es necesario crear un repositorio. Para ello, en el sitio de Github, hacemos clic en la opción "New Repository" ubicada en la esquina superior derecha.

Se nos abrirá un formulario para comenzar a completar los datos identificatorios del mismo.

En "Repository Name" pueden elegir el nombre que deseen para el repositorio. Posteriormente pueden agregar una descripción de manera opcional.

Una vez completados los campos seleccionan "Crear Repositorio".

Juan Pablo Paillet
Fullstack Developer
http://pailletjp.com

Cuando ya disponemos del repositorio creado, podemos comenzar a trabajar de manera local y subir los cambios.

Vamos a simular un flujo de trabajo tradicional a manera demostrativa.

En primer lugar, creamos una carpeta en nuestra PC, le ponemos un nombre de prueba.

Nos movemos a la carpeta, abrimos nuestra terminal y ejecutamos el comando

git init

para comenzar un repositorio local.

Creamos un archivo cualquiera de texto, en el mismo podemos ingresar cualquier contenido, posteriormente ejecutamos el comando

git add .

Para añadir todos los archivos de la carpeta.

Una vez añadidos, hacemos un commit para confirmar los cambios, con el siguiente comando:

git commit -m "Primer commit"

Una vez agregados y commiteados, presionamos git push para subir los cambios a nuestro repositorio ejecutando

git push origin master

Nos saltará un error. Debido a que no hay un repositorio remoto asignado al local.

Para ello, es necesario ejecutar el comando

git remote add origin <server>

Donde <server> es la dirección de nuestro repositorio creado.

Para encontrar la dirección, es necesario ingresar a GitHub, nos dirigimos a nuestro repositorio, y nos aparecerá la url hacia la cual hay que apuntar.

Las estructuras de URLS de repositorios se arman de la siguiente manera:

*https://github.com/**usuario**/**nombreRepositorio**.git*

Una vez identificado el nombre, añadimos el acceso remoto a nuestro repositorio.

*Git remote add origin https://github.com/**usuario**/**nombreRepositorio**.git*

Cuando lo agregamos, podemos pushear los archivos commiteados ejecutando nuevamente

git push origin master

Puede que, para terminar de confirmar, le solicite sus credenciales de Git (usuario y contraseña).

Una vez ingresados, vuelven a ejecutar el comando para pushear los cambios al repositorio.

Como verán, la lógica de Git funciona de la siguiente manera.

A medida que realizamos cambios, se crean archivos ocultos en formato de Bytes que comparan los archivos originales con los modificados con nosotros.

Cuando realizamos modificaciones, las confirmamos realizándoles *commits*, y posteriormente las subimos al repositorio remoto realizando un *push*.

Por cualquier consulta, pueden consultar esta guía rápida y práctica que contiene todo el flujo de trabajo con Git.

http://rogerdudler.github.io/git-guide/index.es.html

Heroku – Docker - Firebase

A la par del sistema de versionamiento, que nos permite acceder desde cualquier dispositivo a las diferentes etapas del desarrollo, es necesario poder alojar nuestro sistema o aplicación en la nube. Para ello existen una multitud de herramientas que utilizando diversos protocolos.

¿Qué es Docker?

Docker es un sistema de código abierto que permite encapsular nuestras aplicaciones en containers (bloques de memoria virtual) fácilmente portables y trasladables de una pc a otra. Funcionan como máquinas virtuales que lo que hacen es cargar solamente los recursos necesarios para poder correr nuestra aplicación sin necesidad de instalar el sistema operativo completo. Toma solamente los recursos necesarios para poder ejecutar el código que estamos desarrollando, siendo muy ligero y fácil de trasladar.

¿Qué es Heroku?

Juan Pablo Paillet
Fullstack Developer
http://pailletjp.com

Para las tecnologías que trabajaremos en este curso, debido a la novedad de las mismas, aún no hay muchos planes shared o compartidos de hosting que estén predefinidos para alojarlas.

Uno de los más populares que existen en la actualidad es Heroku.

Heroku es una Paas (plataforma como servicio) que tiene como finalidad la configuración de un entorno de desarrollo optimizado para las diferentes tecnologías. Su filosofía radica en que los desarrolladores dejen de preocuparse por la infraestuctura, configuración del sistema operativo, compatibilidad entre componentes, y demás cuestiones de configuración, para enfocarse directamente en el desarrollo.

Para nuestro caso, que trabajaremos principalmente con Node.js y MongoDB, es una de las herramientas óptimas, ya que consiste en un entorno ya configurado y listo para ejecutar aplicaciones con este Stack tecnológico.

El mismo utiliza también el protocolo Git, y permite que al pushear nuestros cambios, se realice automáticamente un Deploy (puesta en producción) para que los usuarios puedan visualizar nuestro sitio, aplicación, etc.

¿Qué es Firebase?

A la par de Heroku, otras de las herramientas optimizadas para poder subir nuestras aplicaciones es Firebase de Google. La misma permite, a la par de una plataforma preconfigurada para ejecutar aplicaciones o sitios construídos con el Mean Stack, la posibilidad de utilizar diversas funcionalidades, como autenticación de usuarios, envío de notificaciones, e incluso una base de datos no SQL orientada a documentos (como lo es MongoDB) ya configurada.

A lo largo del curso, iremos utilizando estas tecnologías (principalemente Heroku) de una forma práctica para comprender su funcionamiento.

IDES – Editores de Código. Sublime, Atom, Visual Studio Code.

A la hora de trabajar en nuestro desarrollo, hay una multitud de herramientas preparadas para trabajar, conocidas como IDES (entornos de desarrollo integrado).

Un entorno de desarrollo integrado o entorno de desarrollo interactivo, en inglés Integrated Development Environment (IDE), es una aplicación informática que proporciona servicios integrales para facilitarle al desarrollador o programador el desarrollo de software.

Normalmente, un IDE consiste de un editor de código fuente, herramientas de construcción automáticas y un depurador. La mayoría de los IDE tienen auto-completado inteligente de código (IntelliSense). Algunos IDE contienen un compilador, un intérprete, o ambos, tales como NetBeans y Eclipse; otros no, tales como SharpDevelop y Lazarus.

El límite entre un IDE y otras partes del entorno de desarrollo de software más amplio no está bien definido. Muchas veces, a los efectos de simplificar la construcción de la interfaz gráfica de usuario (GUI, por sus siglas en inglés) se integran un sistema controlador de versión y varias herramientas. Muchos IDE modernos también cuentan con un navegador de clases, un buscador de objetos y un diagrama de jerarquía de clases, para su uso con el desarrollo de software orientado a objetos.

Existen una multitud de IDES, y la elección de cada uno en particular queda en el gusto de cada uno.

Las utilizadas principalmente son:

Sublime Text

Este es uno de los editores más completos en cuanto a complementos, funciones y personalización que podemos encontrar en toda la red. Este editor es de código cerrado y, para poder utilizarlo, debemos pagar una licencia bastante cara, 70 dólares, un precio caro, pero a la altura de todo lo que nos ofrece.

Listar todas las funciones y características que nos ofrece Sublime Text es bastante complicado, por lo que vamos a intentar resumirlas:

- Cuenta con una función que nos permite seleccionar varias líneas a la vez y escribir a la vez en ellas. Muy útil la hora de crear funciones o renombrar variables.
- Nos permite realizar modificaciones en lote de varios puntos de un documento.
- Cuenta con una paleta de comandos que nos da acceso rápido a una completa lista de funciones, como, por ejemplo, acceso a los marcadores o a la lista de sintaxis para colorear un fichero según el lenguaje de programación en el que esté escrito.
- Cuenta con un comando “GoTo” que nos permite desplazarnos muy fácilmente entre varios documentos, muy útil en proyectos grandes. Esta función, además, nos sirve para desplazarnos entre funciones, líneas e incluso palabras.
- La función “Find and Replace” nos permite seleccionar y remplazar rápidamente cualquier contenido del documento.

- Modo “sin distracciones” que nos permite escribir sin ningún otro elemento que pueda hacer que nos distraigamos.
- Cuenta con una completa API escrita en Python y, además, una consola que nos permite experimentar con nuestros proyectos muy fácilmente.
- Además de todo eso, otra característica elemental de este editor es que es totalmente personalizable. Ya sea desde el propio editor como desde sus ficheros de configuración podemos cambiar todo lo que imaginemos, desde los atajos de teclado hasta las macros. Absolutamente todo. Por si fuera poco, este editor puede personalizarse aún más gracias a la gran variedad de extensiones, complementos y add-ons que podemos encontrar por la red.

Podemos descargar Sublime Text desde su propia página web principal.

https://www.sublimetext.com/

Visual Studio Code

Microsoft Visual Studio Code es un editor de texto/IDE de programación desarrollado y lanzado por Microsoft en 2015 como una herramienta de código abierto, un hito impensable para Microsoft y para su marca Visual Studio. Este editor, como hemos dicho, ha sido desarrollado como software de código abierto y se distribuye sin ninguna limitación de forma totalmente gratuita.

Este editor se caracteriza por ser bastante más modular que el anterior. Por defecto, el editor cuenta con las funciones más básicas y elementales y está preparado para adaptarse a los principales lenguajes de programación más utilizados, como HTML/CSS o C. Sin embargo, su mayor potencial se encuentra en las extensiones, y es que desde el mismo editor vamos a poder acceder a una gran variedad de extensiones y complementos que nos va a permitir dotarle de las funciones y características que necesitemos, sin sobrecargar el editor.

Entre otras, las principales características que nos brinda Microsoft Visual Studio Code son:

- IntelliSense, una función que nos permite desde resaltar la sintaxis de nuestros proyectos hasta hacer uso de auto-completar funciones, controlar nuestras variables y, además, ver definiciones de las funciones de los distintos lenguajes de programación.

- Cuenta con un avanzado depurador de código que nos permite conocer todos los problemas y errores en tiempo real, ayudándonos a identificar estos problemas y solucionarlos.
- Se integra en Git. Nos permite versionar nuestros trabajos fácilmente desde esta plataforma.
- Gracias a sus módulos, es compatible con prácticamente cualquier lenguaje de programación, desde los más conocidos hasta los más extraños de los que, probablemente, no hayamos oído hablar nunca.

Atom

El IDE consta de una aplicación de escritorio construida utilizando tecnologías web. La mayor parte de los paquetes tienen licencias de software libre y es construido y mantenido por su comunidad. Atom está basado en Electrón (Anteriormente conocido como Atom Shell), un framework que permite aplicaciones de escritorio multiplataforma usando Chromium y Node.js. Está escrito en CoffeeScript y Less. También puede ser utilizado como un entorno de desarrollo integrado (IDE).

Atom liberó su beta en la versión 1.0, en 2015. Sus desarrolladores lo llaman un "Editor de textos hackable para el siglo XXI".

Sus principales ventajas son:

- Funcionalidades extra (Packages): Ésta es una de las áreas dónde Atom destaca especialmente. Con el "package manager", instalado por defecto, podemos instalar y desinstalar fácilmente casi cualquier función imaginable, pues a día de hoy más de 6500 paquetes de modificaciones se encuentran disponibles.
- Integración con Git: Atom ha sido desarrollado por miembros de la plataforma GitHub, y esto se nota a la hora de trabajar con él. Nuestro proyecto de Atom se sincronizará automáticamente con el repositorio de Git y veremos en todo momento si se encuentra en la misma versión que nuestro repositorio o en qué documentos hay divergencias.
- Personalización: Atom dispone de un documento totalmente editable donde podemos ajustar el estilo de trabajo a nuestras más detalladas preferencias. Desde "convertir las tabulaciones en espacios" y viceversa hasta "guardar automáticamente al perder el enfoque en el archivo".

Siendo pragmáticos, **todos los editores de texto son similares**, y su elección va a depender de los gustos personales de cada uno.

En lo personal, por costumbre utilizo Visual Studio Code, pero es simplemente una cuestión de costumbre.

Juan Pablo Paillet
Fullstack Developer
http://pailletjp.com

Links de Interés

- Git: La Guía Sencilla: http://rogerdudler.github.io/git-guide/index.es.html
- Documentación Oficial de Git: https://git-scm.com/book/es/v1/Empezando
- Guía Oficial GitHub: https://guides.github.com/activities/hello-world/
- Documentación oficial de Docker: https://docs.docker.com/
- Ayuda de Firebase: https://support.google.com/firebase/?hl=es-419
- Documentación Heroku: https://devcenter.heroku.com/categories/reference

Material Adicional 16/08

Chic@s espero no haberlos abrumado con la cantidad de conceptos y demás, como les dije, hay mucho para aprender y, si bien al principio seguramente es complicado asimilar todo, verán que con la constancia conseguirán manejar (al menos desde las bases) todo esto.

Les comparto algunos links de interés:

¿Qué es y como funciona el diseño responsive?

https://blogginzenith.zenithmedia.es/que-es-y-como-funciona-el-diseno-responsive-diczionario/

Curso básico HTML:

https://www.youtube.com/watch?v=cqMfPS8jPys&list=PLhSj3UTs2_yVHt2DgHky_MzzRC58UHE4z

Este es un videocurso que explica las principales etiquetas HTML y como irlas maquetando el archivo. El mismo utiliza Sublime Text como IDE, pero pueden hacerlo con cualquier otro.

Juan Pablo Paillet
Fullstack Developer
http://pailletjp.com

Con que puedan consultar esos 2 links por el momento, es más que suficiente.

Power Point Primer Clase

Asimismo, les comparto el Power Point de lo charlado en la primer clase:

https://docs.google.com/presentation/d/1g0XW5cZ47GT1EFsI_wYVSGHWbLOUlsNGqMbtgYgxI2E/edit?usp=sharing

Link más importante

Por último, el mejor y más importante link de todos

www.google.com

Nunca duden en buscar cualquier inquietud allí, ya que hay comunidades enormes donde podrán obtener ayuda, e incluso muchos desarrolladores que están activamente respondiendo a las preguntas que realicen en las diferentes comunidades (como stackoverflow.com)

Juan Pablo Paillet