**Funval International**

**Introducción al Desarrollo Web**

8 mins

Tabla de contenido

Fundamentos de la Programación

¿Qué es la Programación?

Desarrollo Web

Páginas web estáticas vs Dinámicas

Páginas Estáticas

Ejemplo de página estática

Páginas Dinámicas

Ejemplo de página dinámica

Recursos Para comenzar

Enlaces

Bienvenido a Funval

Este es tu primer día en el camino a convertirte en un Desarrollador Web Full Stack, por lo que te invitamos a prestar mucha atención a todo lo que aprenderás. Toma apuntes de lo que consideres mas relevante y anota los temas que sean más complicados de entender para que puedas resolver tus dudas posteriormente.

Sin mas rodeos, comencemos 😉.

Fundamentos de la Programación#

La programación en pocas palabras, es darle datos e instrucciones a una computadora para que procese esos datos siguiendo las instrucciones que le dimos para que nos devuelva un resultado. ¿Y para qué? para resolver problemas de nuestra vida como hacer operaciones matemáticas, sacar estadísticas, automatizar procesos, etcétera.

Pero esos problemas se pueden resolver de diferentes maneras, algunas más rápidas y eficientes y otras no tanto. Pero todas con el fin de encontrar la solución al problema.

¿Qué es la Programación?#

Es un término común que se puede describir según la Real Academia de la Lengua como “Acción o efecto de programar”. ¿Y programar? se define como Idear y ordenar las acciones necesarias para realizar un proyecto.

Todo esto en términos generales, no se refiere específicamente a la programación de lenguajes informáticos, pero nos da una idea de lo que es.

Para ejecutar lo que el usuario desea hacer en su computadora, o bien para resolver un problema específico, este precisa buscar un software que realice o ejecute con exactitud la tarea que se ha planteado o elaborar y desarrollar (escribir) un programa que la realice. El trabajo de elaboración de un programa se denomina “programación”. Pero la programación no es solo el trabajo de escritura del código, sino todo un conjunto de tareas que se deben cumplir, a fin de que el código que se escribió resulte correcto, robusto, y cumpla con el objetivo o los objetivos para los que fue creado.

Las afirmaciones que se derivan de lo anterior son varias:

Conocer las herramientas, los formalismos y los métodos para transformar un problema en un programa escrito en un lenguaje (que posiblemente no será el lenguaje máquina), y para qué dicho programa pueda ser transformado en un código ejecutable.

Saber transformar el problema inicial en un algoritmo y luego en un programa.

La primera afirmación es genérica y se considera para varias categorías de problemas para resolver. Por su parte, la segunda es específica de un problema determinado que se tiene que resolver, para lo cual existen diversas metodologías específicas de resolución para este tipo de problemas.

Para los casos de problemas muy generales, en ocasiones existen métodos conocidos que solo se adaptan a un problema en particular. El método es, por lo general, un algoritmo o una técnica de programación.

Desarrollo Web#

Web Developing ¿Y el Desarrollo Web?

La aplicación de la programación y la lógica son muchos: Inteligencia Artificial, Administración de base de datos, Desarrollo de videojuegos, Desarrollo de Sistemas operativos, etc.

Entre todas estas ramas encontramos el Desarrollo Web.

Desarrollo web es un término que define la creación de sitios web para Internet o una intranet.

El Programador Web es un profesional que trabaja con desarrolladores de páginas web, softwares, operadores de sistemas (SysOps) que ejecuta y/o desarrollar las versiones e implementaciones de códigos. El puesto requiere de personas que también posean soft skills o habilidades blandas relevantes que impacten positivamente el control de calidad de las pruebas y los equipos de operaciones; además de fomentar un entorno colaborativo.

Dentro del Desarrollo Web encontramos distintos roles:

Desarrollador web front-end es el profesional responsable de proyectar y construir la interfaz del usuario en un proyecto web. Es quien cuida de la visualización y la interfaz gráfica.

Desarrollador web back-end Son los que trabajan con aplicaciones haciendo código que sólo el servidor entenderá.

Desarrollador web full-stack - Por supuesto que nada impide que un desarrollador trabaje en ambas áreas. Por eso el término full-stack se refiere a que el profesional que trabaja tanto con el back-end como el front-end.

Aquí te vamos a enseñar lo necesario para que tengas las habilidades de un desarrollador Full-Stack así que ¡preparate para aprender creando! 😁.

Comenzaremos con las habilidades de un desarrollador Front-end y seguiremos con el Back-end.

Las primeras tecnologías que aprenderás son: HTML5, CSS3 y JavaScript. Los dos primeros tienen como objetivo darle estructura, contenido y diseño a una página web, y el último se encarga de la lógica que hay detrás de una página.

Páginas web estáticas vs Dinámicas#

Al empezar en el mundo del desarrollo web, seguro te estarás preguntando ¿Cómo inicio a hacer páginas web? Veamos primero las dos diferencias entre estos productos web.

estatic\_vs\_dinamic

Páginas Estáticas#

Lo primero, hay páginas estáticas y páginas dinámicas. Las páginas estáticas van a ser todos los productos web o todas las páginas que van a tener pura información para consumir, ya sea texto e imágenes, y que esta información no va a cambiar. Siempre va a ser la misma. Por ejemplo, algún blog post, si te metes algún blog de alguien, siempre vas a poder ver los blogs de esa persona.

Esta información raramente va a cambiar. Posiblemente van a llegar personas nuevas al equipo o se van a ir personas o van a agregar cierta información distinta de la empresa o de los productos que tienen, pero gran parte de la información se va a quedar ahí de forma estática por mucho tiempo, y si tú generas esta página, posiblemente te vayan a contactar meses después para poderte decir como “oye, queremos hacer este cambio, ahora queremos agregar nuevas fotos o una nueva dirección de donde estamos nosotros”. O “hay 2 miembros nuevos del equipo, creemos que agregue su información y su foto”. Pero estas páginas siguen siendo estáticas porque la información perse nunca va a cambiar. Si tú entras, hoy vas a ver lo mismo. Si entras mañana, no va a haber algún tipo de cambio.

Lo más importante es que estas páginas no están conectadas a un servidor o a una base de datos, no necesitas generar interacción con la base de Datos porque no hay algo que generes este cambio del lado del cliente. El cliente desde el navegador no tendrá alguna interacción que le permita cambiar información de la página web, no tendrá algún login, etc.

Ejemplo de página estática#

Estatic Example

Las Landing page son un buen ejemplo de páginas estáticas. Como en este caso. La información que verán los clientes siempre es la misma y no cambiará a menos que se solicite. No hay opción de hacer login y esto es solo informativo.

Este tipo de páginas no es menos importante, simplemente tiene un objetivo distinto a una página dinámica y son más sencillas de construir.

Páginas Dinámicas#

Las páginas dinámicas, también conocidas como web Apps, ya dejan de ser páginas y se convierten más en aplicaciones por todas las cosas que vas a poder hacer con estas mismas.

Estas sí están conectadas a una base de Datos, por lo que el usuario, o sea nosotros a través del navegador, si podemos generar una interacción con la página o con la aplicación. Ahora se tiene que guardar cierta información o estado de lo que estemos haciendo, se esperaría poder tener un login para yo tener mi cuenta y poder guardar información. Se esperaría poder interactuar ya sea con productos si quiero comprar algo, si es que estoy viendo algo de mis seguidores en las redes, o si quiero ver a las personas que sigo.

Ejemplo de página dinámica#

Dinamic Example

En páginas como Twitter, tú si puedes interactuar. Puedes dejarla de esa forma y dentro de unos 15 minutos si actualizas, la información será distinta. Esta página está en comunicación con el servidor para traer los nuevos cambios según el usuario lo solicite. La información que ves allí es específica para ti, y otro usuario podría ver twits distintos.

Recursos Para comenzar#

Antes de continuar, debemos asegurarnos de tener los siguientes programas que formarán parte de este curso:

VSCode: Este es el editor de código que estaremos utilizando a lo largo del curso, si bien es cierto que existen varios editores y puedes usar el que te sea mas cómodo, te recomendamos este para que todos estemos trabajando en las mismas situaciones y sea más fácil poder interactuar. En caso aun no lo tengas, puedes descargarlo desde el siguiente enlace:

Download VS Code

Navegadores: Es probable que ya cuentes con algunos de estos, sin embargo cuando trabajamos en desarrollo web, es importante contar con diferentes opciones ya que nuestra página web puede comportarse de manera diferente en cada uno de estos por los distintos motores de interpretación que utilizan. Algunos de los navegadores que te recomendamos tener para tus pruebas son:

Brave

Chrome

Edge

Firefox

En cada uno de los anteriores links, tendrás una opción de descarga.

Si tienes dificultades para instalar alguno de los recursos mencionados, puedes postear tu error o duda en el chat grupal donde podremos ayudarte 😀

Enlaces#

Al terminar la lectura anterior, revisa los siguientes artículos donde encontrarás más información de los temas:

Páginas Estaticas vs Dinámicas - Open Webinars

¿Qué es desarrollo web? - Articulo Tekla

------------------------------------------------------------------------------------------------------------------------------

Funval International

GitHub

5 mins

Tabla de contenido

¿Qué es GitHub?

Ventajas de GitHub

1. Colaboración Efectiva:

2. Control de Versiones Robusto:

3. Gestión de Problemas:

4. Integración con Herramientas:

Uso de GitHub junto con Git en la Terminal

1. Clonar un Repositorio Remoto:

2. Crear un Nuevo Proyecto y Hacer Cambios:

3. Agregar y Confirmar Cambios:

4. Configurar un Repositorio Remoto en GitHub:

5. Subir Cambios al Repositorio Remoto:

Competidores de GitHub

Ejemplo de Subir un Proyecto a GitHub

Recursos

En el mundo del desarrollo de software, la colaboración eficiente y la gestión de versiones son aspectos cruciales para el éxito de cualquier proyecto. En este contexto, GitHub se ha destacado como una herramienta fundamental que permite a los desarrolladores trabajar en equipo de manera efectiva, rastrear cambios en el código y mantener un control riguroso sobre las versiones de sus proyectos. En este artículo, exploraremos en detalle qué es GitHub, las ventajas de su uso, cómo se integra con Git a través de la terminal, sus competidores y proporcionaremos un ejemplo práctico de cómo subir un proyecto creado desde cero a GitHub mediante la línea de comandos.

¿Qué es GitHub?#

GitHub es una plataforma de desarrollo de software basada en la web que ofrece servicios de alojamiento de repositorios de control de versiones. Fundada en 2008 por Chris Wanstrath, PJ Hyett y Tom Preston-Werner, GitHub se ha convertido en la plataforma líder para la gestión colaborativa de proyectos de desarrollo de software. La plataforma utiliza el sistema de control de versiones Git, creado por Linus Torvalds en 2005, como su base, lo que permite a los desarrolladores trabajar en equipo, realizar seguimiento de cambios en el código y gestionar el flujo de trabajo de desarrollo de manera efectiva.

Ventajas de GitHub#

El uso de GitHub ofrece una serie de ventajas significativas para los desarrolladores y equipos de desarrollo de software:

1. Colaboración Efectiva:#

GitHub facilita la colaboración entre desarrolladores al proporcionar herramientas para realizar contribuciones, revisar y comentar el código de otros, y coordinar esfuerzos en proyectos compartidos.

2. Control de Versiones Robusto:#

GitHub se basa en Git, un sistema de control de versiones altamente eficiente. Esto permite a los equipos mantener un registro completo y detallado de los cambios realizados en el código, lo que facilita la identificación de errores y la recuperación de versiones anteriores.

3. Gestión de Problemas:#

La plataforma GitHub incluye funciones de seguimiento de problemas (issues) que permiten a los equipos crear, asignar y dar seguimiento a problemas y tareas específicas en un proyecto. Esto mejora la organización y la comunicación en el equipo.

4. Integración con Herramientas:#

GitHub se integra fácilmente con una amplia variedad de herramientas de desarrollo, como sistemas de compilación automática, pruebas unitarias y servicios de implementación continua (CI/CD), lo que facilita la automatización de tareas y la mejora de la eficiencia del flujo de trabajo.

Uso de GitHub junto con Git en la Terminal#

Para utilizar GitHub junto con Git a través de la terminal, sigue estos pasos esenciales:

1. Clonar un Repositorio Remoto:#

Para comenzar a trabajar en un proyecto de GitHub, primero debes clonar el repositorio remoto en tu máquina local utilizando el comando git clone seguido de la URL del repositorio.

git clone https://github.com/tu\_usuario/tu\_proyecto.git

2. Crear un Nuevo Proyecto y Hacer Cambios:#

Crea o copia los archivos de tu proyecto en el directorio local clonado. Realiza las modificaciones necesarias en los archivos.

3. Agregar y Confirmar Cambios:#

Utiliza los siguientes comandos de Git para agregar los cambios realizados y confirmarlos con un mensaje descriptivo:

git add .

git commit -m "Añadir cambios al proyecto"

4. Configurar un Repositorio Remoto en GitHub:#

En GitHub, crea un nuevo repositorio vacío en tu cuenta. Luego, vincula el repositorio local con el remoto utilizando los comandos git remote:

git remote add origin https://github.com/tu\_usuario/tu\_proyecto.git

5. Subir Cambios al Repositorio Remoto:#

Finalmente, sube los cambios al repositorio remoto en GitHub utilizando el comando git push:

git push -u origin main

Esto cargará tus cambios en la rama principal (main) del repositorio en GitHub.

Competidores de GitHub#

Aunque GitHub es líder en su categoría, existen competidores notables en la gestión de repositorios de control de versiones. Algunos de los competidores más destacados incluyen:

GitLab: GitLab es una plataforma similar a GitHub que ofrece tanto una versión en la nube como una versión de código abierto que se puede alojar en servidores propios. Ofrece una amplia gama de características, incluyendo integración continua y entrega continua (CI/CD).

Bitbucket: Bitbucket, propiedad de Atlassian, es otra plataforma de control de versiones que se integra estrechamente con otras herramientas de Atlassian, como Jira y Confluence. También ofrece capacidades de alojamiento de repositorios Git y Mercurial.

GitKraken: GitKraken es un cliente de Git con una interfaz gráfica intuitiva que facilita el uso de Git y GitHub. Es especialmente popular entre los desarrolladores que prefieren una experiencia visual.

Ejemplo de Subir un Proyecto a GitHub#

Supongamos que deseas subir un proyecto nuevo a GitHub:

Crea un nuevo repositorio vacío en GitHub.

En tu terminal, navega al directorio del proyecto y ejecuta los siguientes comandos:

git init

git add .

git commit -m "Primer commit"

git remote add origin https://github.com/tu\_usuario/tu\_proyecto.git

git push -u origin main

Estos pasos inician un nuevo repositorio Git local, agregan y confirman los cambios, establecen un repositorio remoto en GitHub y suben los archivos al repositorio en la rama principal (main).

En resumen, GitHub se ha convertido en una herramienta esencial para la colaboración y el control de versiones en el desarrollo de software. Ofrece una amplia gama de ventajas, desde la colaboración efectiva hasta la integración con herramientas de desarrollo y un control de versiones sólido. Integrar GitHub con Git a través de la terminal es una habilidad esencial para los desarrolladores, y esta guía proporciona los pasos básicos para comenzar a utilizar GitHub de manera efectiva. Aunque GitHub es líder en su espacio, es importante estar al tanto de sus competidores y evaluar cuál es la mejor opción para las necesidades específicas de tu proyecto.

Recursos#