

Introducción a la tecnología web

"Cualquier persona sin importar sus estudios previos puede aprender a programar, quizás lo único que se requiere es la paciencia y tener un problema real que resolver."

Mario Alberto Chávez, Ingeniero de software y Ruby. [Aprender a Programar o desarrollar](#)".

Los primeros pasos como Full Stack Developer

¡Bienvenido/a a tu primera bitácora! Aquí encontrarás disponibles los conceptos clave y las herramientas que serán necesarias para prepararte para cada encuentro con tu equipo docente y compañeros/as. Además, motivaremos tu acceso a los canales de comunidad: te propondremos interactuar con otros/as Developers para disipar dudas, **hacer** consultas y, por qué no, conocer y recomendar cosas interesantes sobre los temas que trabajaremos.

En este primer bloque de aprendizaje (que consta de 16 bitácoras para 16 encuentros), aprenderás a estructurar un sitio web mediante HTML y a aplicar los estilos que desees con CSS. En otras palabras, en este bloque trabajaremos en lenguajes de programación para el Front-End, i.e. aquello que observan los/as usuarios/as cuando ingresan en un sitio web. Además, aprenderás buenas prácticas de programación y practicarás con un proyecto real: Podcast Channel.



En particular, abordaremos los siguientes temas:

- **Desarrollo web (conceptos fundamentales para entender qué hacemos cuando programamos):** Entorno / Internet / Web / Cliente - Servidor / Front-end



/ Sitios web estáticos y dinámicos / Aplicaciones web / Desarrollo / Programación / Lenguajes de la web / Programación lógica / Maquetado / Estilado / Herramientas de desarrollo.

- **HTML (la arquitectura de nuestro sitio):** Introducción al lenguaje / Sintaxis / Estructura del documento / Etiquetas semánticas / Etiquetas estructurales / Atributos / Etiquetas funcionales / Rutas relativas y absolutas / Vinculación de recursos / Buenas prácticas.
- **CSS (el estilo estético de nuestro sitio):** Introducción al lenguaje / Sintaxis / Integración con HTML / Selectores por etiqueta / Selectores a través de atributos / Propiedades y valores / Sistema de cascada / Estilos por defecto / Posicionamiento de elementos / Modelo de cajas / Relatividad / Debugging en el explorador / Responsive web design / Unidades de medida relativas / Flexbox / Media-queries / Pseudo clases / Pseudo elementos / Transformaciones / Animación / Recursos externos (Google Fonts, FontAwesome) / Framework y librerías (Bootstrap) / Pre-procesadores (SASS).

¿Qué es programar?

Si nos apuramos, podríamos decir que **programar** es pasar **de ser usuarios a creadores**. Cuando buscamos definiciones acerca de qué es la programación, nos encontramos con respuestas tales como “dar instrucciones al ordenador” o bien “crear software usando un lenguaje de programación”. Según [Wikipedia](#), es el proceso por el cual una persona desarrolla un programa valiéndose de una herramienta que le permita escribir el código (nosotros usaremos HTML, CSS y Javascript) y de otra que sea capaz de “traducirlo” a lo que se conoce como lenguaje de máquina, que puede comprender el ordenador.

Agustín Quetto lo explica de una manera más simple:

“Caminamos hasta que las piernas nos duelan, dormimos hasta que no conciliamos más el sueño y buscamos dinero en la billetera hasta encontrar la cantidad suficiente. Todo posee un cambio de estado, ¿no crees? Si las piernas no duelen, caminamos. Si estamos cansados, dormimos hasta no estarlo y buscamos dinero en la billetera hasta dar con la cantidad necesaria (ojalá siempre fuese así, ¿no?).

Programar no es más que lo que hacemos a diario: un conjunto de instrucciones (...) de manera “digital”. No hay diferencia.

Programar es como hablar con una computadora: decirle qué hacer, dónde y cómo. Para esto podemos poseer valores que cambian, para saber si aún tenemos sueño o no, ciclos para repetir las cosas como ir al trabajo todos los días, preguntar el valor de algo para saber si estamos o no en situación de hacer cierta acción o tomar una decisión.”

[\(“¿Qué es programar? ¿Vivimos programando?”\)](#)



Desarrollar un software supone un conocimiento en programación, pero además, comprender las implicancias de ciertas decisiones a la hora de implementar una solución. Por eso, los/las Full Stack Developers formamos parte -directa o indirectamente- de todo **ciclo de desarrollo de software: la planificación, el análisis, el diseño de la solución, el testing, la integración y el mantenimiento de un producto.**



Las 5 claves que hacen que los/las Full Stack Developers seamos todo terreno

El desarrollo web full stack, de acuerdo con la Encuesta de [Stack Overflow 2019](#), actualmente la ocupación de Full Stack Developers es la más demandada. Iniciarnos en este camino de un [Full Stack Developer](#) significa que vamos a saber cómo agregar valor en las interfaces visibles para los/as usuarios/as (Front-end) como en la lógica que opera detrás (Back-end).

Te contamos cuáles serán las habilidades necesarias para el rol que desempeñes como Full Stack Developer. Como cualquier habilidad, no se adquieren de un día para el otro, sino que se ejercitan en el tiempo y nuestra propuesta es que las trabajes a lo largo de la carrera.

- 1. Resolvemos problemas:** convertimos los requerimientos -es decir, las necesidades de la vida real con las que estamos trabajando- en soluciones. Sabemos qué problema resuelve el software y en cada parte del código orientamos nuestro desarrollo específico a la solución del problema general.

- 2. Ejercitamos la creatividad:** si bien es cierto que existen buenas prácticas y patrones para resolver problemas clásicos, no hay una sola forma de llegar al mismo resultado.

Para satisfacer las necesidades del software que desarrollamos es imprescindible que pongamos nuestra creatividad a disposición de la solución. Una buena forma de hacerlo es buscar diferentes caminos y analizar cuál es el que resolverá nuestro requerimiento con mayor eficiencia.

A veces, también es necesario ir un poco más allá de las soluciones que están al alcance y ampliar nuestra estructura de pensamiento, siendo profundamente críticos/as con el camino conocido y poniendo en práctica nuestra capacidad para desafiar nuestros propios horizontes.

- 3. Ejercitamos el pensamiento abstracto:** los lenguajes de programación nos proveen símbolos y estructuras para poder representar información y la forma en que se va a procesar dentro de nuestro programa.

Estos símbolos y estructuras son parte de un marco de trabajo que aceptamos para poder implementar nuestras soluciones. Este marco es una abstracción.

- 4. Aplicamos el pensamiento lógico:** las bases del pensamiento crítico con las que fuimos educados directa o indirectamente en las distintas etapas de la educación formal nos dotaron de la lógica proposicional para poder argumentar y sacar conclusiones a partir de los datos que tenemos acerca de un problema dado.

Cuando realizamos operaciones de comparación para procesar información que recibe nuestra aplicación, estamos ni más ni menos que aplicando las tablas de verdad que nos provee esta disciplina.

Si no sabes de qué hablamos, ¡no te preocupes! Lo vas a aprender programando :)

- 5. Nos actualizamos:** no debe existir una frase más trillada que “la tecnología avanza a pasos agigantados”. Pero ¡no por eso deja de ser cierta! Los/as Full Stack Developers contamos con una ventaja en ese sentido. Vivimos trabajando con las últimas tecnologías, y si bien es nuestra responsabilidad intentar estar al tanto de lo que pasa allá afuera, estar activos profesionalmente nos da las herramientas para conocer el contexto.



Ahora bien, antes de comenzar con los temas de esta carrera, queremos darte tres reglas que cumpliremos a lo largo de la carrera, y que son fundamentales para maximizar tu aprendizaje:

- **Vale decir “no sé”.** Los mejores Full Stack Developers saben reconocer cuando no saben qué significa algo, qué implica, o cómo resolver situaciones que se les presentan. Esta habilidad es fundamental, ya que es el primer disparador para ir a buscar ayuda (ya sea pedirle a alguien que sepa más, que lo haya vivido antes, que esté dispuesto a debatir o buscar en internet).
- **Si sabes lo que tienes que ‘codear’ pero no sabes cómo hacerlo, entonces no tienes un problema: tienes un desafío.** No saber cómo programar algo puede ser una situación técnica que se puede resolver preguntando a otros o buscando en internet. En múltiples situaciones, verás que la clave no es saber cómo escribir las líneas de código, sino saber o entender el código que hay que escribir. Eso significará que, más allá de tu capacidad técnica, sabes a dónde debes llegar y, por ende, estarás mejor posicionado para buscar las respuestas o caminos posibles.
- **A programar se aprende, valga la redundancia, programando.** Deberás sentarte a programar para aprender a hacerlo. Te acompañaremos en ese trayecto y trataremos de que sea lo mejor posible, pero no te garantizamos que todo será disfrute. Habrá momentos difíciles, otros en los que no tendrás ganas de volver a programar, e incluso algunos en los que te plantees “¿quién me mandó a hacer esto?”. Pero esperamos acompañarte a sobreponerte y continuar intentándolo, valga la repetición, programando.

Sitios web: del dicho al hecho hay servidores, clientes, dispositivos, y navegadores

Ya te contamos con qué desafíos te encontrarás en el camino hacia convertirte en Full Stack Developer de aplicaciones que generan nuevas oportunidades de vida para otros/as usuarios/as. ¡Adentrémonos en el mundo que hay detrás así lo conocemos mejor!

[Ethan Marcotte](#), diseñador web, y creador del libro *Responsive Web Design*, hace referencia a la arquitectura:

“Los cimientos de un edificio definen su plano, que define su estructura, que da forma a la fachada. Cada fase del proceso arquitectónico es más inmutable, más invariable que la anterior. Las decisiones creativas modelan, literalmente, un espacio físico, definiendo la forma en que la gente se moverá a través del mismo por décadas o incluso siglos”.

Como Full Stack Developer es importante que conozcas la estructura de internet y cómo funciona para poder crear sitios web. Sin darnos cuenta, todos los días



estamos en contacto con esta estructura cuando usamos nuestros dispositivos como el celular o la computadora para visitar páginas web de interés o subimos fotos a redes sociales como Instagram, Facebook o Twitter.

Las primeras preguntas que es fundamental hacernos como Full Stack Developers son:

- ¿Cómo funciona internet?
- ¿Cómo es que podemos visualizar un sitio web?
- ¿De dónde sale la información? ¿Qué sucede detrás de las aplicaciones?

Internet es un conjunto descentralizado de redes de comunicación interconectadas que utilizan ciertos protocolos comunes (TCP/IP) para poder “entenderse”. Esto garantiza que las redes físicas heterogéneas que la componen sean parte de una red lógica única de alcance mundial.

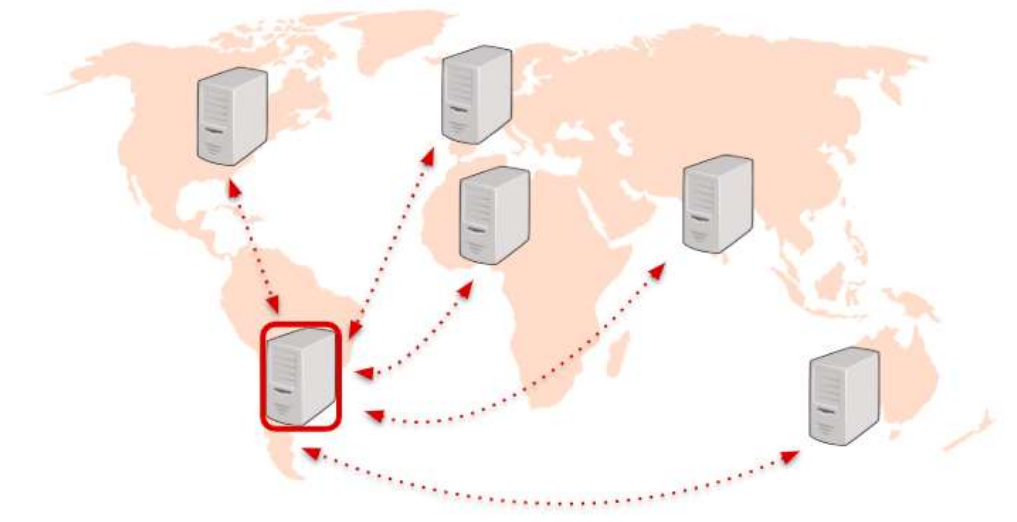
Si bien abordaremos el concepto de internet y su historia en la bitácora 2, es importante que sepas en este punto que el World Wide Web (WWW) no es lo mismo que internet, sino que se trata de uno de los servicios que más éxito ha tenido en internet. La WWW es un conjunto de protocolos que permite, de forma sencilla, la consulta remota de archivos de hipertexto.

Nosotros/as los/as usuarios/as, cuando accedemos a los [sitios web](#) realizamos solicitudes como “clientes” a través de dispositivos (computadoras, teléfonos, tablets).

Los servidores son quienes reciben estas solicitudes y nos dan una respuesta en diferentes formatos según lo que pidamos (imágenes publicadas, nuestro correo electrónico o la página web que queremos ver).

Por lo tanto, **un [servidor](#) se encuentra disponible para otros dispositivos y será quien suministre la información requerida por los clientes.**





Todos nuestros dispositivos se conectan a través de internet con otras computadoras conformando una red que aloja sitios web y funcionan como servidores para que todo el mundo pueda ver lo que solicita.

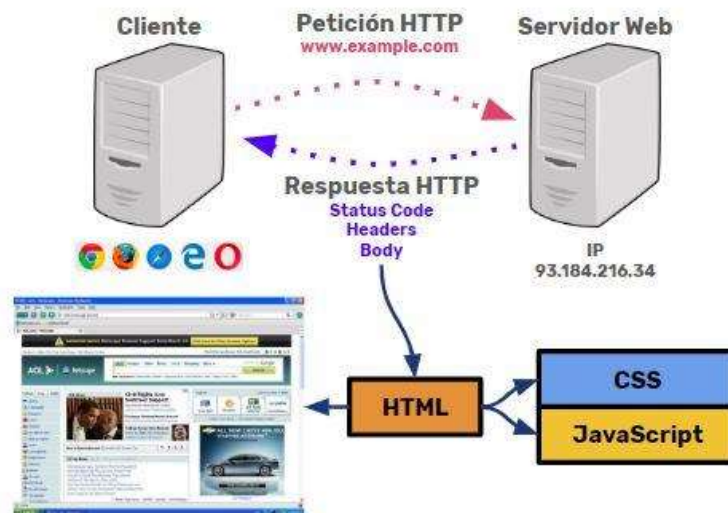
Existen varios [tipos de servidores](#). ¡No te preocupes por aprenderte todos los términos y definiciones, de a poco nos vamos a ir familiarizando! Pero **si tu intención es desarrollar un sitio web debes buscar este alojamiento y el primer al lugar que tienes que acudir es un [servidor web](#).**

¿Qué es un servidor web?

Cada vez que un cliente acceda desde el navegador a nuestro sitio web, el servidor web transferirá todos los archivos necesarios para atender la solicitud y resolverla. Nos conectamos a otras computadoras y no hacemos más que descargarnos una serie de archivos que desde otra computadora conectada a la red actúa de servidor web.

En otras palabras, **un [servidor web](#) almacena todos los archivos que componen una aplicación web y visualiza nuestro producto a los clientes a través del navegador.**

[Emmanuel Orozco](#) nos explica: "sin embargo, tiene que existir algún tipo de estándar para que mi navegador, tu navegador, todos los navegadores y servidores del mundo se comprendan. Ese estándar, se llama protocolo." Mediante el **[protocolo HTTP](#) en el puerto 80 el cliente recibe los archivos que allí están alojados.**



Podemos simplificar los pasos de cliente/servidor de la siguiente manera:

1. Inicio de petición por parte del cliente (ingresa www.google.com).
2. El servidor web recibe la petición y la resuelve.
3. El cliente recibe la respuesta.
4. Cierre de transacción (visualización de la página web).

Para tener tu propio [servidor web](#) podrás basarte en esta estructura clásica de cliente-servidor, y en el momento en que comenciences a desarrollar [aplicaciones web](#) verás que será necesario replicar este escenario.

Qué debemos saber de las aplicaciones web

Para comenzar a interactuar en este mundo de la web debemos conocer cómo funciona a través de internet y también qué lenguajes se utilizan.

Lenguajes web




Hay tres lenguajes básicos que van a tener un rol específico en una [aplicación web](#):



- **HTML** nos permite estructurar el contenido que vamos a brindarle al usuario.
- Con **CSS** vamos a trabajar con la presentación visual del contenido.
- **JavaScript** nos da la posibilidad de interactuar con las acciones del usuario y actualizar el contenido sin necesidad de recargar el sitio.

Los documentos que componen una aplicación web son [archivos de texto](#) plano con una extensión específica que los hace reconocibles para los navegadores en cuanto al rol que cumplen dentro del sitio.

¡Ya te contaremos más en el encuentro que viene y en las siguientes bitácoras!


 **Recuerda** cargar **todas las preguntas** que te hayan surgido sobre los temas cubiertos en la presente bitácora en el **Trello** de tu grupo de estudio. Tu equipo docente armará el repaso en clase a partir de esto.

¡Prepárate para el próximo encuentro!


Profundiza

Internet, redes, web y servidores: probablemente la usemos todos los días, pero, ¿sabemos realmente cómo funciona internet? o ¿qué es una aplicación web? Échale un vistazo a este video en los que te proponemos una primera aproximación a los temas que trataremos durante el encuentro.

 [Cómo funciona internet](#)


 También, puedes profundizar sobre lo que te contamos sobre clientes y servidores: [Modelo Cliente/Servidor](#)

Herramientas


 Ten en cuenta que tu computadora va a ser una pieza fundamental en el camino que estás emprendiendo. Por eso es importante prepararla para que tenga la capacidad de editar y alojar los archivos de tu [aplicación web](#), y a la vez correr un [servidor](#)




[web](#) para que el navegador pueda abrir la aplicación mediante el [protocolo HTTP](#).


 [Descarga Visual Studio Code](#), el editor de código que usaremos. ¡Lo necesitarás para poner manos a la obra durante el encuentro!

 [Sigue estas instrucciones](#) y crea tu primer archivo HTML.

 [Instala Live Server](#), una extensión de Visual Studio Code que nos permite ejecutar un servidor web en nuestra computadora para que automáticamente pueda mostrar en un navegador el sitio web que estamos editando.

Challenge

 Para el encuentro deberás traer (o mostrar en pantalla) un objeto que te represente y deberás presentarte contando por qué lo elegiste y de qué manera te representa.

 Mira el siguiente [video](#) y realiza tu reflexión sobre las siguientes preguntas. Retomaremos el debate en el próximo encuentro:

- ¿Por qué crees que es necesario aprender a programar?
- ¿Cuál crees que es el alcance de la programación?
- ¿Qué opinión te genera la frase de Steve Jobs “Todo el mundo en este país debería aprender a programar por que te enseña cómo pensar”?