**Python es uno de esos lenguajes de programación.** Normalmente se recomienda a principiantes, pero realmente ahora se utiliza a gran escala.

**¿Qué puedes hacer con Python?**

Python te permite:

* Automatizar tareas repetitivas
* Extraer datos de sitios web como Google y Amazon
* Interactuar con servidores y APIs como Facebook, Twilio, Weather Forecasts y Twitter
* Realizar análisis de datos

**¿Qué empresas famosas utilizan Python?**

Python es utilizado por empresas como Dropbox, Google, Pinterest, The Washington Post, Firefox e Instagram. Es ENORME y, sin embargo, sólo es necesario conocer una fracción del lenguaje Python para crear aplicaciones muy atractivas. Muchos desarrolladores experimentados apenas han arañado la superficie. **En este curso, nos centraremos en las partes más importantes del lenguaje Python para que puedas empezar a hacer algún análisis de datos y, potencialmente, conseguir un trabajo.**

Este curso te dará el mapa general de forma rápida para que puedas seguir aprendiendo.

## Front-end vs. back-end (3 min)

**Aún no completado**

## Antes de aprender Python, descubre los fundamentos del desarrollo web

Antes de instalar Python, pensemos por dónde tenemos que empezar con él, y qué queremos construir.

Las aplicaciones web son herramientas a las que se accede a través de Internet. Abres un navegador, y vas a un sitio web, y cargas una aplicación - que vive en el ordenador de otra persona, pero que se te muestra a través de tu navegador.

## Front-end vs. back-end

Toda aplicación tiene un front-end y un back-end.

El front-end es lo que se ve en el navegador, y el back-end es lo que se encarga del trabajo entre bastidores: una base de datos que almacena todo lo que necesitas, y reglas, que conectan la información de tu base de datos con la forma en que quieres que se vea en el navegador de alguien.

### Lenguajes del front-end:

* HTML
* CSS
* Javascript

### Lenguajes de back-end:

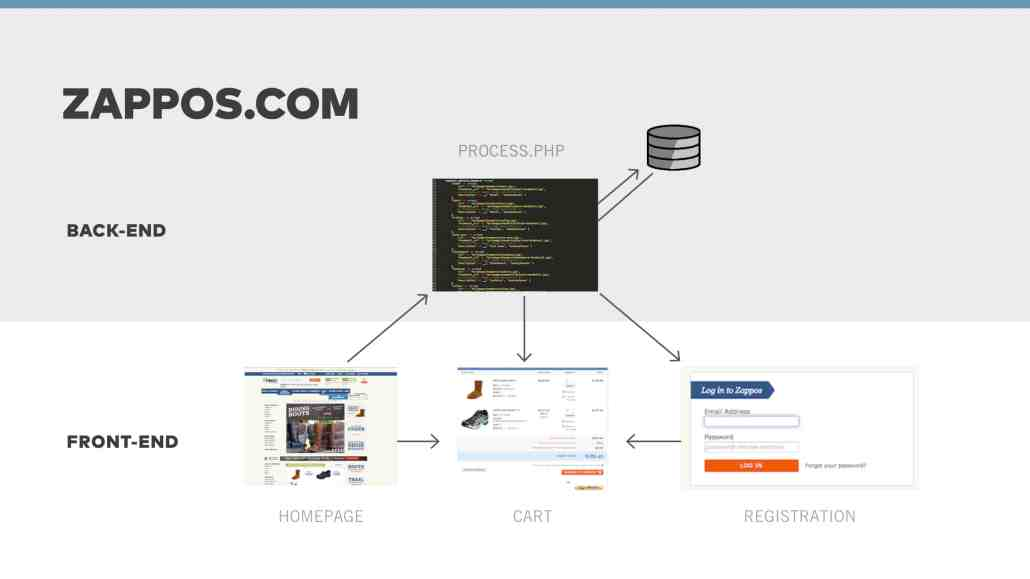
* Python
* Ruby
* PHP

### Bases de datos:

* SQL
* MySQL
* Postgres
* SQLite

Mientras los desarrolladores de frontend construyen el aspecto de un sitio web, los desarrolladores de backend construyen el funcionamiento de un sitio web.

Supongamos que quieres construir un sitio web de WordPress para tu negocio. El desarrollador de frontend crearía el tema: las imágenes, el estilo y la presentación. Mientras que el desarrollador de backend puede trabajar en la gestión de la base de datos, así como los usuarios del sitio, la seguridad y las cuestiones de rendimiento del sitio.



Los desarrolladores de frontend utilizan principalmente tres lenguajes: HTML, CSS y Javascript. En los últimos años, el papel del desarrollador de frontend ha evolucionado, por lo que los desarrolladores de frontend profesionales también podrían considerar el desarrollo de algunas habilidades de JavaScript más intermedias y avanzadas, así como la creación de experiencia en el uso de herramientas de línea de comandos y un marco como React.

Los desarrolladores de backend deben ser competentes en los lenguajes de programación que se renderizan en el lado del servidor de un sitio web o aplicación. Los lenguajes de programación backend más populares son PHP, Ruby, Python, Node.js y Java. Normalmente, un buen desarrollador de backend será un maestro en uno de estos lenguajes, pero lo suficientemente familiarizado para codificar en un segundo o tercer lenguaje de programación. Los desarrolladores de backend también tendrán que dominar el trabajo con bases de datos como MySQL, Oracle y SQL Server.

## Habilidades

Entonces, ¿qué tipo de habilidades necesita un desarrollador de frontend frente a un desarrollador de backend?

Los desarrolladores de frontend trabajan en la apariencia y la interfaz de usuario del sitio web, por lo que, además de los lenguajes de programación mencionados anteriormente, el desarrollador de frontend debe tener algunos conocimientos sobre el uso de herramientas de diseño como PhotoShop, Sketch o Figma. Los desarrolladores de frontend también deben conocer los fundamentos del alojamiento web y la compra de un dominio.

Por otro lado, los desarrolladores de backend deben tener habilidades de pensamiento crítico. Un desarrollador de backend suele depurar el código, así como diseñar sistemas para que el usuario interactúe con el sitio web. Las preguntas que el desarrollador de backend debe ser capaz de responder: ¿Dónde se almacenan los datos? ¿Están almacenados de forma segura? Si el tráfico del sitio se multiplica por 100 de la noche a la mañana, ¿podrá el sitio escalar sin colapsar? ¿Cómo puedo añadir una nueva función al sitio web, sin romper la funcionalidad actual? ¿Cómo puedo probar un sitio web (a menudo en una plataforma de ensayo y/o ejecutando pruebas mediante el desarrollo dirigido por pruebas) para que el usuario final experimente el menor número posible de errores y fallos?

## ¿Qué es un desarrollador Full-Stack?

Las personas que tienen habilidades tanto en el desarrollo del frontend como del backend suelen denominarse desarrolladores full stack. En otras palabras, tienen una gama completa de habilidades que pueden aplicarse a la interfaz de usuario y a todo lo que la hace funcionar en segundo plano.

Algunas personas consideran que un desarrollador full stack no es tan bueno como un desarrollador frontend o backend. A menudo se hace referencia al dicho: "Joven de todos los oficios, maestro de ninguno". Pero también vale la pena señalar que la frase completa es "Jack of all trades, master of none, aunque a menudo es mejor que un maestro de uno".

Como desarrollador, tener conocimientos tanto de frontend como de backend significa más oportunidades. Podrás optar a más puestos de trabajo por contrato, a tiempo parcial o a tiempo completo. Como freelance, podrás aceptar más proyectos sin estar limitado sólo al frontend o al backend.

Desde la perspectiva del cliente o del empleador, podrás entender los proyectos en su conjunto. Tanto cómo tiene que funcionar para el usuario como cómo tiene que funcionar en segundo plano. Les darás un punto de contacto para todas sus necesidades. Y serás capaz de apoyarles cuando las cosas vayan mal en cualquiera de los dos lados. Esto le hace aún más valioso a largo plazo.

La demanda de desarrolladores de frontend y de backend crece continuamente. Por lo tanto, elegir cualquiera de los dos puede ayudarte a crear la carrera o el negocio que siempre has deseado. Asegúrate de explorar ambos mundos del desarrollo para determinar cuál es el más adecuado para ti.

## Conceptos generales (15 min)

**Aún no completado**

Vamos a empezar con los conceptos más básicos ✊🏻

## ****¿Qué es programar?****

Muchas veces hemos utilizado la palabra “programar” sin detenernos a pensar realmente lo que esto significa, ¿se come? ¿es ruso? ¿es humano?, pues la definición más precisa y clara que podemos tener es que **corresponde al acto de dar una secuencia de instrucciones a un computador para que éste al ejecutarlas pueda completar una tarea especificada.**Programar es parecido a una receta, describe los ingredientes (datos) y la secuencia de pasos (proceso) necesarios para convertir los ingredientes en un rico plato.

Los ordenadores tienen su propio lenguaje por lo que no entienden el que los humanos utilizamos, como por ejemplo español, inglés, italiano etc. Es por esto que tomamos un algoritmo y lo escribimos en un **lenguaje de programación** para que pueda comprender lo que le indicamos, este va en una jerarquía desde lenguaje máquina (binario) hasta lenguaje de alto nivel como Java, Ruby, JavaScript, C++, etc.

#### Una buena analogía para la programación

Primero, hay patrones en nuestra vida diaria. El universo opera de una manera un tanto predecible, por ejemplo, el día y la noche, las estaciones, el amanecer y el atardecer. La gente realiza rutinas como levantarse por la mañana, ir a la escuela o al trabajo. Recibimos instrucciones de otras personas como nuestros superiores en el trabajo. La forma en que cocinamos ciertas recetas puede ser explicada en pasos finitos.

En segundo lugar, cada vez que usamos dispositivos inteligentes, algún código se ejecuta en segundo plano. Mover el puntero del ratón de una parte de la pantalla del ordenador a otra puede parecer una tarea sencilla, pero en realidad, muchas líneas de código acaban de ejecutarse. Un acto tan simple como escribir letras en Google Docs lleva a que se ejecuten líneas de código en segundo plano. Todo es código en todas partes.

Todo lo que escribimos en el ordenador, los programas, los llamamos 'código'. El `código' del proyecto, nos referimos a todos los archivos python, archivos auxiliares, y a las sentencias que están escritas en ellos.

## El lenguaje natural de la computadora

Las máquinas tienen su lenguaje natural como los humanos. Los ordenadores no entienden el lenguaje humano. El lenguaje natural de las computadoras es el código binario 1 y 0. Estos representan dos estados: on (1) y off (0).

Ese es el lenguaje natural de los equipos electrónicos. Sería agitado para nosotros como humanos comunicarnos con la computadora en binario.

#### Introduciendo los lenguajes de programación

Para comunicarnos con las máquinas que hablan binario, lo hacemos en un lenguaje más cercano a nuestro propio lenguaje natural. Como el inglés, el francés, el swahili o el árabe. Los lenguajes de programación están más cerca de nuestros lenguajes naturales. Pero son más estructurados y deben ser aprendidos a fondo.

Pueden ser lenguajes de alto o bajo nivel. Los lenguajes de programación de alto nivel están más lejos del lenguaje de las máquinas que los de bajo nivel. Este "más lejos" es usualmente llamado una abstracción, pero no entraremos en eso en esta serie. No nos distraigamos :)

El ordenador necesita una forma de entender nuestro lenguaje humano. Para ello, necesitaremos un traductor.

### **¿Qué es un algoritmo?**

Es la receta para resolver un conjunto de problemas a través de pasos precisos, definidos y finitos que al ser realizados llevan a la solución. Conceptualmente un algoritmo tiene tres componentes:

* Entrada: datos sobre los que el algoritmo opera
* Proceso: pasos que hay que seguir
* Salida: resultado que entrega el algoritmo

### **¿Qué es el lenguaje Binario?**

¿Se bebe? XD Los computadores trabajan con un sistema llamado binario que utiliza dos valores, los cuales son los dígitos “0” y “1” para manipular cualquier tipo de información.

Pizarrón blanco con letras negras

Descripción generada automáticamente con confianza media

¿Podemos hacer la conversión hacia número binario? Sí, obvio! Tenemos que ir dividiendo entre 2 y escribir en una columna a la derecha el resto, se coloca un “0” si el resultado de la división es par y un “1” si es impar, la lista de 0 y 1 se leen de arriba hacia abajo dándonos el nuevo número binario.

### **¿Qué es Hexadecimal?**

El sistema hexadecimal es de base 16, sus números están representados por los 10 primeros dígitos de la numeración decimal y el intermedio que va del número 10 al 15 representados por las letras del alfabeto de la A a la F. Actualmente es uno de los sistemas más utilizados en el procesamiento de datos principalmente por:

1. La simplificación en la escritura de los números decimales, cada 4 cifras binarias se representa por una hexadecimal.
2. Cada cifra hexadecimal se puede expresar a través de cuatro cifras binarias. Para convertir un número binario en hexadecimal se realiza el mismo proceso, pero a la inversa.
3. En el desarrollo web se usa hexadecimal especialmente para definir colores, cada color estará representado por un grupo de seis dígitos en hexadecimal, precedidos por el signo #

### **¿Código fuente?**

El código fuente es el archivo con las instrucciones necesarias ejecutadas en un lenguaje de programación que sirve para ser compilado posteriormente y así poder ser utilizadas por la computadora de manera directa. En palabras simples, no es más que un texto plano escrito sin formatos y en un lenguaje de programación específico, capaz de ser leído por cualquier editor de textos y comprensible por cualquier programador que maneje el lenguaje utilizado.

### **¿Compilador e intérprete?**

Un compilador es un pequeño programa informático, que se encarga de traducir (compilar) el código fuente de cualquier aplicación que se esté desarrollando, es decir, es un programa que se encarga de traducir lo programado a un lenguaje de máquina que pueda ser comprendido por el equipo y pueda ser procesado por este. **“Un compilador es un programa que convierte o traduce el código fuente de un programa hecho en lenguaje de alto nivel, a un lenguaje de bajo nivel (lenguaje de máquina)”.**

El intérprete analiza el código fuente y lo ejecuta directamente. Intérpretes y compiladores tienen diversas ventajas e inconvenientes que los hacen complementario. Un intérprete facilita la búsqueda de errores, pues la ejecución de un programa puede interrumpirse en cualquier momento para estudiar el entorno (valores de las variables, etc.). Además, el programa puede modificarse sobre la marcha sin tener que comenzar nuevamente la ejecución.

**Compilador: sólo traduce**

**Intérprete: decodifica y ejecuta**

Diagrama

Descripción generada automáticamente

### Pseudocódigo

¿Semi código? ¿Casi código? Pseudocódigo es una descripción estructurada de un algoritmo, a pesar de ser muy parecido al código que finalmente se escribirá en el computador, el pseudocódigo está pensado para ser leído por humanos.

La idea es una forma intermedia entre usar lenguaje natural y un lenguaje de programación. No posee una forma estrictamente definida pero normalmente se respeta lo siguiente:

1. Se usa el concepto de variable y asignación
2. Se indican los datos de entrada y de salida
3. Se utilizan instrucciones condicionales del tipo: “si <condición> entonces <instrucciones> sino <instrucciones>
4. Se utilizan instrucciones de ciclo como “repetir hasta que …”

**Podemos ver un ejemplo de pseudocódigo**

Texto

Descripción generada automáticamente

### Diagrama de flujo

Es una representación gráfica de un proceso donde cada paso es representado por un símbolo diferente que contiene una breve descripción de la etapa de proceso. Los símbolos gráficos del diagrama están unidos entre sí con flechas que indican la dirección de flujo del proceso.

El diagrama ofrece una descripción visual de las actividades implicadas en un proceso mostrando la relación secuencial entre ellas, facilitando la rápida comprensión de cada actividad y su relación con las demás.

Para su creación se utilizan símbolos y normas de construcción determinados por estándares internacionales, con el objeto de poder interpretar y ser interpretados por otros programadores.

Algunas consideraciones claves en un diagrama de flujos son:

1. Todos los símbolos deben estar conectados
2. Un símbolo de proceso puede recibir varias líneas
3. Un símbolo de decisión puede recibir varias líneas pero solo devolverá dos: **SI** o **NO**, **VERDADERO** o **FALSO**
4. Un símbolo de inicio jamás recibe líneas
5. Un símbolo de fin no devuelve ninguna línea

### Conceptos generales

Tienes que saber que todos los lenguajes de programación comparten algunos elementos básicos que funcionan y se usan de forma diferente en cada lenguaje, pero que cumplen el mismo objetivo.

Esos elementos son:

* Tipos de datos
* Variables
* Control de flujo
* Ciclos
* Estructuras de datos
* Funciones