CHATGPT: optimizando la calidad de los resultados

3 EXPLORANDO APLICACIONES

# 05 Para saber más: Prompts - Diversas aplicaciones

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86721/next)

Podemos escribir prompts para diversos fines, tanto para nuestra vida diaria como en el entorno laboral. En los videos anteriores, utilizamos prompts para resúmenes de textos, traducciones e incluso para analizar los sentimientos de personas que compraron un producto. ¿Vamos a explorar algunas posibilidades más?

## Transformando el tono de voz

Vanessa es una influencer digital que comparte consejos de maquillaje y productos de belleza en sus redes sociales. Recientemente, publicó un análisis crítico sobre un nuevo lanzamiento de una marca famosa de cosméticos, resaltando algunos aspectos negativos del producto. Sin embargo, se dio cuenta de que su tono de voz inicial podría haber sido un poco duro. Se le ocurrió la idea de volver a escribir su texto con ChatGPT, utilizando un tono más amigable y objetivo, para transmitir su opinión de manera equilibrada y respetuosa a sus seguidores.

¿Intentemos escribir un prompt con ese propósito?

Vuelva a escribir el texto a continuación, delimitado por " " ", dejando su tono de voz más simpático y amigable:

Texto: " " " Probé el nuevo lanzamiento de la marca XYZ y, siendo sincera, me decepcionó mucho. La calidad del producto es simplemente terrible. El empaque es frágil y se rompió después del primer uso, además, la pigmentación de las sombras es débil y la durabilidad es prácticamente inexistente. No entiendo cómo una marca de renombre puede lanzar algo tan malo al mercado. ¡No lo recomendaría en absoluto! " " "

Como resultado, obtenemos el nuevo texto:

"Tuve la oportunidad de probar el último lanzamiento de la marca XYZ y, para ser sincera, me decepcionó un poco. Me sorprendió la calidad del producto, que no cumplió con mis expectativas. El empaque resultó ser un poco frágil, lamentablemente se rompió después del primer uso. Además, noté que la pigmentación de las sombras no es tan intensa como me gustaría, y la durabilidad también dejó un poco que desear. Confieso que me quedé un tanto perpleja, ya que esperaba más de esta marca de renombre. En fin, mi opinión es que tal vez este producto no sea la mejor inversión, así que no lo recomendaría con mucha énfasis."

¡Observa cómo de esta manera el texto es menos duro y agresivo! Usar ChatGPT para cambiar el tono de voz de un texto puede ser interesante por varias razones. Aquí hay algunas de ellas:

Versatilidad: ChatGPT es capaz de generar texto en diferentes estilos y tonos de voz. Puede transformar un texto formal en un texto más casual, o viceversa. Esto permite adaptar el contenido para diferentes audiencias o situaciones, haciéndolo más atractivo y adecuado para el propósito deseado.

Personalización: Con ChatGPT, es posible personalizar el tono de voz para adaptarse al estilo de una marca, empresa o individuo específico. Esto es especialmente útil en áreas como marketing y publicidad, donde la consistencia de la voz de la marca es crucial para construir una identidad sólida.

Adaptación a diferentes plataformas: Cada plataforma o canal de comunicación tiene sus propias peculiaridades y expectativas con respecto al tono de voz. ChatGPT puede ayudar a adaptar el contenido para diferentes plataformas, como redes sociales, blogs, correos electrónicos, entre otros, para que se alinee con el lenguaje y el estilo apropiados.

Creatividad y entretenimiento: Cambiar el tono de voz de un texto puede ser una forma de estimular la creatividad y crear contenido divertido y original. Se pueden probar diferentes estilos narrativos, haciendo que las historias sean más cautivadoras y atractivas.

Accesibilidad: El uso de ChatGPT para cambiar el tono de voz de un texto también puede beneficiar a la accesibilidad. Por ejemplo, ChatGPT puede ayudar a adaptar el tono para mejorar la descripción textual para que las personas ciegas puedan comprender con más detalle el contenido.

## Convirtiendo formatos

Allan, un analista de datos, recibió un conjunto de información en formato de diccionario Python. Sin embargo, necesita convertir estos datos al formato CSV (valores separados por comas) para poder importarlos a Google Sheets y realizar análisis más avanzados. Ha decidido utilizar ChatGPT para ayudar en la tarea de conversión.

¿Cómo podríamos redactar un indicador para esto? Es importante contextualizar a ChatGPT, indicando que tenemos algunos datos almacenados en un diccionario Python, y también es bueno mencionar el nombre de ese diccionario. Después de eso, podemos pedirle que escriba los datos en el formato deseado y que devuelva solo los datos en ese formato.

El indicador sigue una secuencia lógica de información, comenzando con la contextualización, pasando por la descripción del problema y finalizando con la solicitud específica al modelo.

'001': {

'nombre': 'João',

'edad': 30,

'cargo': 'Gerente',

'salario': 5000.00

},

'002': {

'nombre': 'Maria',

'edad': 25,

'cargo': 'Analista',

'salario': 3500.00

},

'003': {

'nombre': 'Pedro',

'edad': 28,

'cargo': 'Desarrollador',

'salario': 4000.00

}

}

# 06 Desafío: analizando sentimientos en varias reseñas

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86725/next)

Una empresa que fabrica air-fryers está interesada en comprender las opiniones de los consumidores sobre el rendimiento del producto y sus funcionalidades. La empresa desea lo siguiente para cada reseña:

* Un resumen en español con un máximo de 30 palabras;
* Una lista con los puntos fuertes del aparato;
* Una lista con los puntos débiles del aparato;
* El sentimiento final de la persona respecto al producto, es decir, si el sentimiento es positivo, neutral o negativo.

Tu tarea será analizar las siguientes reseñas de personas que compraron el producto en una tienda en línea estadounidense.

* Reseña 1 : I use the air fryer a lot. It works very well. I made hamburgers in it last night, the were wonderful! The rack you put the food on is a little hard to clean. It has lots of crevices that food gets in. The basket itself is very easy to clean. I also do not trust the rack not to fall out when dumping out food, like French fries.
* Reseña 2 : Air fryer works fine until the handle breaks off. There's only 3 screws that attach the handle to a plastic piece on the basket. The plastic piece breaks and the screws fall out so there's no way to attach the handle to the basket rendering the air fryer useless. If there were 4 screws or if the screws attached to metal the handle attachment would have been much more secure and less prone to breaking.
* Reseña 3 : Even though I returned it for the bigger one of the same company I liked the fryer. Ferry ez to use But hard to judge. Seafood button isn't fish and it was a guessing game how to cook the product. I way over cooked it and the chicken legs came out crispy, but the other side was raw. But, this is my first time using an air fryer so maybe it's me. Very sleek on the counter buttons light up real nice. I made tater tots and let me tell you they came out perfect and stayed hot. I like it so much that I returned this one and got the same company Bigger one. But I had to buy it from the place itself not from Amazon and I don't lie that at all. Hope this review helps.

Además, será necesario generar un resultado en formato de salida JSON, para que los datos sean utilizados en análisis posteriores. En este archivo JSON, necesitarás especificar que deseas 5 claves:

* ID de la reseña
* Resumen
* Lista de puntos fuertes
* Lista de puntos débiles
* Sentimiento

Basándose en las respuestas a estas preguntas, la empresa podrá identificar oportunidades de mejora en su producto y mejorar su estrategia de marketing.

¿Listo para empezar?"

# 07 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86715/next)

En esta clase, aprendiste a:

* Hacer un resumen de un texto en tu idioma original;
* Resumir reseñas de productos;
* Enfocar el resumen de la reseña en aspectos específicos;
* Crear un array con las palabras clave de un resumen;
* Utilizar un prompt para analizar una reseña y mostrar el sentimiento de la persona (positivo, neutro o negativo).

4 ESTRATEGIAS PARA TEXTOS LARGOS

4096 tokens en chatgpt3.5

# 02 Tentando resumir un texto con muchos caracteres

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86616/next)

Carla necesita realizar un trabajo de su MBA sobre un extenso y complejo artículo científico. Decidió utilizar ChatGPT para resumir el texto y facilitar su comprensión. Sin embargo, no sabe cómo utilizar ChatGPT para resumir textos largos y requiere ayuda para comprender cómo hacerlo.

Para ayudar a Carla a utilizar ChatGPT para resumir textos largos, elija la opción correcta:

Seleccione una alternativa



Una estrategia para resumir textos largos en ChatGPT es copiar y pegar todo el texto en la ventana de conversación y esperar que ChatGPT haga un resumen completo en una sola respuesta.



Es necesario dividir el texto en partes más pequeñas al resumir un texto extenso en ChatGPT para facilitar la comprensión y asegurar un resumen más preciso y eficiente.



No es necesario utilizar un chat nuevo para cada parte del texto, ya que ChatGPT es capaz de resumir todo el texto en una sola conversación.



Una estrategia eficiente para resumir textos largos en ChatGPT es insertar imágenes y gráficos en el texto para ayudar al algoritmo a comprender mejor la información.

# 04 Para saber más: ¿Qué es OpenAI Playground?

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86722/next)

## Conociendo el Playground

OpenAI Platform es una plataforma en línea desarrollada por OpenAI que permite a los usuarios experimentar y explorar los modelos de lenguaje de OpenAI de manera interactiva. Es un entorno donde puedes interactuar con el modelo de lenguaje, hacer preguntas y recibir respuestas generadas por el modelo en tiempo real.

Esa plataforma está dirigida a desarrolladores, científicos de datos, empresas y profesionales de IA que deseen aprovechar los recursos avanzados. Para acceder, es necesario iniciar sesión en la página de [OpenAI platform](https://platform.openai.com/docs/overview).

La plataforma abarca varios productos y herramientas creados por OpenAI para ayudar en la creación e implementación de aplicaciones basadas en inteligencia artificial. Aquí hablaremos sobre uno de estos recursos disponibles: el Playground. Para acceder, simplemente haz clic en el botón 'Playground' en la parte superior de la página.

Después de hacer clic en Playground, accederemos a la página que se muestra en la figura siguiente. Con la opción “Mode” (Modo) siendo 'Chat', en el lado izquierdo hay un cuadro llamado “System” con el mensaje “You are a helpful assistent” (Eres un asistente útil) . En el centro de la pantalla se encuentra “User” con el mensaje “Enter a user message here” ('Ingresa un mensaje de usuario aquí) y un botón con signo positivo (+) para agregar un mensaje. En la parte inferior hay un botón “Submit” ( Enviar) en verde. Por último, en el lado derecho, hay varias opciones de configuración."

En la interfaz de OpenAI Playground, en el modo 'Chat' (charla), existe la opción de escribir un prompt en el campo "System" (Sistema). Esta es una forma de separar y especificar las tareas que deseas que el modelo de lenguaje realice. El campo "User"(Usuario) es donde puedes ingresar el texto a trabajar o incluso código, permitiéndote separar el contenido en caso de que tu instrucción y tu texto sean largos. Esta separación facilita la comprensión del modelo y ayuda a obtener resultados más precisos.

Es posible elegir qué modelo queremos utilizar en la configuración "Model" (modelo). Para quienes no son suscriptores de ChatGPT versión Plus, solo estará disponible el modelo 3.5. Además, hay algunas configuraciones disponibles en Playground que discutiremos en breve: Temperature (Temperatura), Maximum lenght (Longitud máxima), Top P, Frequency penalty (Penalización de frecuencia) y Presence penalty (Penalización de presencia).

Con estos recursos, Playground permite a los usuarios introducir indicaciones de texto, modificar parámetros y observar las respuestas del modelo, convirtiéndolo en un excelente entorno para trabajar, obtener respuestas optimizadas y generar automatizaciones.

Es importante destacar que, en principio, Playground es gratuito, pero tiene un límite de tiempo de aproximadamente 3 a 4 meses. Al registrarte en una cuenta en OpenAI, recibes automáticamente un crédito inicial de $18, lo que permite generar alrededor de 650,000 palabras. Sin embargo, una vez que se alcance el límite de tiempo o se agoten los créditos, no podrás enviar mensajes a menos que pagues por utilizar el servicio.

## Ajustando las configuraciones

En Playground, puedes controlar los resultados de los prompts ajustando algunas configuraciones que no podemos controlar solo con ChatGPT. ¿Has experimentado el problema de obtener resultados diferentes cada vez que ejecutas un prompt? Existe una configuración que puede resolver este problema, y se llama temperatura.

Cuando configuramos la temperatura en 0 en Playground, el modelo siempre devolverá conclusiones idénticas o muy similares. Por otro lado, si aumentas la temperatura, los resultados no serán tan similares. Por ello, cuando la temperatura es superior a 0, enviar el mismo prompt produce conclusiones diferentes cada vez.

¿Y por qué sucede esto? El modelo predice que el texto es más probable que siga el texto anterior y la temperatura es un valor entre 0 y 2 que básicamente le permite controlar la confianza del modelo al hacer estas Predicciones. La reducción de temperatura significa que habrá menos riesgos y las conclusiones serán más precisas y deterministas. El aumento de la temperatura dará lugar a conclusiones más diversas.

Además de la temperatura, podemos modificar otros parámetros. El 'Maximum Length' define la longitud máxima que una respuesta puede tener. Si estableces un límite numérico, el modelo detendrá la generación de texto una vez que alcance ese límite de caracteres. Esto es útil para controlar la extensión de las respuestas generadas y evitar resultados excesivamente largos.

En cuanto al 'Top P', también conocido como Muestreo de Núcleo, es una técnica que controla la diversidad de las respuestas considerando solo las probabilidades de las palabras más probables. Al establecer un valor para el 'Top P', como 1, por ejemplo, el modelo selecciona las palabras más probables hasta alcanzar una probabilidad acumulada del 100%. Esto evita que el modelo genere respuestas muy raras o improbables.

También contamos con la 'Penalización de Frecuencia', una configuración que controla la repetición de tokens en una respuesta. Establecer un valor más alto para la penalización de frecuencia anima al modelo a evitar repeticiones excesivas y producir respuestas más variadas.

Por último, el Presence Penalty (Penalización de Presencia) es la configuración que controla la preferencia del modelo por incluir o evitar información específica en las respuestas. Cuando aumentas el valor de la Penalización de Presencia, el modelo tiende más a evitar mencionar palabras o frases que hayas proporcionado como instrucción (prompt). Esto significa que el modelo intentará generar respuestas que no incluyan explícitamente la información proporcionada. Por otro lado, al disminuir el valor del Presence Penalty o establecerlo en cero, el modelo tendrá más libertad para incluir la información proporcionada en las respuestas. Esto puede resultar en respuestas más directas y que mencionen explícitamente lo que solicitaste.

La configuración de la Presence Penalty puede ser útil cuando deseas que el modelo genere respuestas más creativas, evitando depender excesivamente de la información proporcionada. Sin embargo, es importante observar que, en algunos casos, especialmente cuando las instrucciones son más largas o complejas, un valor alto de Penalización de Presencia puede hacer que el modelo ignore por completo la información proporcionada.

Estas configuraciones permiten personalizar el comportamiento del modelo de lenguaje según tus necesidades y ajustar el equilibrio entre creatividad, coherencia, longitud de las respuestas y otras características deseadas.

Es importante destacar que OpenAI Platform es un conjunto de servicios y herramientas en constante evolución, por lo que nuevos recursos y productos pueden añadirse a la plataforma con el tiempo. Por ello, es recomendable consultar el sitio web de OpenAI o la documentación oficial para obtener información actualizada sobre los recursos disponibles en OpenAI Platform. ¡Siéntete libre de explorar todo lo que la plataforma tiene para ofrecer!

Para obtener más información sobre la plataforma, puedes visitar la página:

* [OpenAI: Documentation](https://platform.openai.com/docs/introduction)

# 05 Desafío: resumiendo un texto largo

[Próxima Actividad](https://app.aluracursos.com/course/chatgpt-optimizando-calidad-resultados/task/86726/next)

Has recibido un texto extenso y tienes el desafío de resumirlo de manera eficiente, es decir, capturando la información más relevante y esencial del texto original.

Algunos puntos a considerar:

* Limitación de tamaño: ChatGPT tiene un límite de tamaño para las respuestas, por lo tanto, el primer desafío es resumir el texto extenso en un tamaño adecuado sin perder información esencial.
* Comprensión del contexto: El modelo necesita entender el contexto del texto e identificar las partes más relevantes para crear un resumen conciso y preciso.
* Coherencia y cohesión: El resumen necesita mantener la coherencia y la cohesión de las ideas del texto original, evitando información contradictoria o fragmentada. El desafío es asegurar que el resumen sea una representación fiel y lógica del contenido original.

Ponte en marcha y piensa en las estrategias que utilizarás para lograrlo y hacer el resumen con el siguiente texto obtenido del sitio [web de Wikipedia](https://es.wikipedia.org/wiki/Python).

¡Adelante!

Texto: “ Introducción

Python es un lenguaje de alto nivel de programación interpretado cuya filosofía hace hincapié en la legibilidad de su código, se utiliza para desarrollar aplicaciones de todo tipo, por ejemplo: Instagram, Netflix, Spotify, Panda3D, entre otros.Se trata de un lenguaje de programación multiparadigma, ya que soporta parcialmente la orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, dinámico y multiplataforma.

Administrado por Python Software Foundation, posee una licencia de código abierto, denominada Python Software Foundation License. Python se clasifica constantemente como uno de los lenguajes de programación más populares​

Historia

Python fue creado a finales de los años ochenta por Guido van Rossum en Stichting Mathematisch Centrum (CWI), en los Países Bajos, como un sucesor del lenguaje de programación ABC, capaz de manejar excepciones e interactuar con el sistema operativo Amoeba.

El nombre del lenguaje proviene de la afición de su creador por los humoristas británicos Monty Python.

Guido van Rossum es el principal autor de Python, y su continuo rol central en decidir la dirección de Python es reconocido, refiriéndose a él como Benevolente Dictador Vitalicio (en inglés: Benevolent Dictator for Life, BDFL); sin embargo el 12 de julio de 2018 declinó de dicha situación de honor sin dejar un sucesor o sucesora y con una declaración altisonante:

El 20 de febrero de 1991, van Rossum publicó el código por primera vez en alt.sources, con el número de versión 0.9.0. En esta etapa del desarrollo ya estaban presentes clases con herencia, manejo de excepciones, funciones y los tipos modulares, como: str, list, dict, entre otros. Además en este lanzamiento inicial aparecía un sistema de módulos adoptado de Modula-3; van Rossum describe el módulo como «una de las mayores unidades de programación de Python». El modelo de excepciones en Python es parecido al de Modula-3, con la adición de una cláusula else. En el año 1994 se formó comp.lang.python, el foro de discusión principal de Python, marcando un hito en el crecimiento del grupo de usuarios de este lenguaje.

Python alcanzó la versión 1.0 en enero de 1994. Una característica de este lanzamiento fueron las herramientas de la programación funcional: lambda, reduce, filter y map. Van Rossum explicó que «hace 12 años, Python adquirió lambda, reduce(), filter() y map(), cortesía de Amrit Perm, un hacker informático de Lisp que las implementó porque las extrañaba».

La última versión liberada proveniente de CWI fue Python 1.2. En 1995, van Rossum continuó su trabajo en Python en la Corporation for National Research Initiatives (CNRI) en Reston, Virginia, donde lanzó varias versiones del software.

Durante su estancia en CNRI, van Rossum lanzó la iniciativa Computer Programming for Everybody (CP4E), con el fin de hacer la programación más accesible a más gente, con un nivel de 'alfabetización' básico en lenguajes de programación, similar a la alfabetización básica en inglés y habilidades matemáticas necesarias por muchos trabajadores. Python tuvo un papel crucial en este proceso: debido a su orientación hacia una sintaxis limpia, ya era idóneo, y las metas de CP4E presentaban similitudes con su predecesor, ABC. El proyecto fue patrocinado por DARPA. Para el año 2007, el proyecto CP4E se encontraba inactivo; a pesar de ello, Python continúa intentando ser fácil de aprender y no muy arcano en su sintaxis y semántica, con el objetivo de ser entendible incluso para no-programadores.

En el año 2000, el equipo principal de desarrolladores de Python se cambió a BeOpen.com para formar el equipo BeOpen PythonLabs. CNRI pidió que la versión 1.6 fuera pública, continuando su desarrollo hasta que el equipo de desarrollo abandonó CNRI; su programa de lanzamiento y el de la versión 2.0 tenían una significativa cantidad de traslapo.​ Python 2.0 fue el primer y único lanzamiento de BeOpen.com. Después que Python 2.0 fuera publicado por BeOpen.com, Guido van Rossum y los otros desarrolladores de PythonLabs se unieron en Digital Creations.

Python 2.0 tomó una característica mayor del lenguaje de programación funcional Haskell: listas por comprensión. La sintaxis de Python para esta construcción es muy similar a la de Haskell, salvo por la preferencia de los caracteres de puntuación en Haskell, y la preferencia de Python por palabras claves alfabéticas. Python 2.0 introdujo además un sistema de recolección de basura capaz de recolectar referencias cíclicas.

Posterior a este doble lanzamiento, y después que van Rossum dejara CNRI para trabajar con desarrolladores de software comercial, quedó claro que la opción de usar Python con software disponible bajo la GNU GPL era muy deseable. La licencia usada entonces, la Python License, incluía una cláusula estipulando que la licencia estaba gobernada por el estado de Virginia, por lo que, bajo la óptica de los abogados de Free Software Foundation (FSF), se hacía incompatible con GPL. Para las versiones 1.61 y 2.1, CNRI y FSF hicieron compatibles la licencia de Python con GPL, renombrándola como Python Software Foundation License. En el año 2001, van Rossum fue premiado con el FSF Award for the Advancement of Free Software.

Filosofía

Los usuarios de Python se refieren a menudo a la filosofía de Python, que es bastante similar a la filosofía de Unix. El código que siga los principios de Python es reconocido como "pythónico". Estos principios fueron descritos por el desarrollador de Python Tim Peters en El Zen de Python:

* Bello es mejor que feo.
* Explícito es mejor que implícito.
* Simple es mejor que complejo.
* Complejo es mejor que complicado.
* Plano es mejor que anidado.
* Disperso es mejor que denso.
* La legibilidad cuenta.
* Los casos especiales no son tan especiales como para quebrantar las reglas.
* Lo práctico gana a lo puro.
* Los errores nunca deberían dejarse pasar silenciosamente.
* A menos que hayan sido silenciados explícitamente.
* Frente a la ambigüedad, rechaza la tentación de adivinar.
* Debería haber una —y preferiblemente solo una— manera obvia de hacerlo.
* Aunque esa manera puede no ser obvia al principio a menos que usted sea holandés.
* Ahora es mejor que nunca.
* Aunque nunca es a menudo mejor que ya mismo.
* Si la implementación es difícil de explicar, es una mala idea.
* Si la implementación es fácil de explicar, puede que sea una buena idea.
* Los espacios de nombres (namespaces) son una gran idea. ¡Hagamos más de esas cosas!

Desde la versión 2.1.2, Python incluye estos puntos (en su versión original en inglés) como un huevo de pascua que se muestra al ejecutar import this.

Tipos de datos

Los tipos de datos básicos se pueden resumir en esta tabla:

| Tipo | Clase | Notas | Ejemplo |
| --- | --- | --- | --- |
| str | Cadena en determinado formato... | Inmutable | 'Cadena' |
| bytes | Vector o array de bytes | Inmutable | b'Cadena' |
| list | Secuencia | Mutable, puede contener objetos... | [4.0, 'Cadena', True] |
| tuple | Secuencia | Inmutable, puede contener objetos... | (4.0, 'Cadena', True) |
| set | Conjunto | Mutable, sin orden, no contiene... | {4.0, 'Cadena', True} |
| frozenset | Conjunto | Inmutable, sin orden, no contiene... | frozenset([4.0, 'Cadena', True]) |
| dict | Diccionario | Grupo de pares clave:valor | {'key1': 1.0, 'key2': False} |
| int | Número entero | Precisión arbitraria | 42 |
| float | Número decimal | Coma flotante de doble precisión | 3.1415927 |
| complex | Número complejo | Parte real y parte imaginaria j | (4.5 + 3j) |
| bool | Booleano | Valor booleano (verdadero o falso) | True o False |

* Mutable: si su contenido (o dicho valor) puede cambiarse en tiempo de ejecución.
* Inmutable: si su contenido (o dicho valor) no puede cambiarse en tiempo de ejecución.

Condicionales

Una sentencia condicional ejecuta su bloque de código interno solo si se cumple cierta condición. Se define usando la palabra clave if seguida de la condición y el bloque de código. Si existen condiciones adicionales, se introducen usando la palabra clave elif seguida de la condición y su bloque de código. Las condiciones se evalúan de manera secuencial hasta encontrar la primera que sea verdadera, y su bloque de código asociado es el único que se ejecuta. Opcionalmente, puede haber un bloque final (la palabra clave else, seguida de un bloque de código) que se ejecuta solo cuando todas las condiciones anteriores fueron falsas.

>>> verdadero = True

>>> if verdadero: # No es necesario poner "verdadero == True"

... print("Verdadero")

... else:

... print("Falso")

...

Verdadero

>>> lenguaje = "Python"

>>> if lenguaje == "C": # lenguaje no es "C", por lo que este bloque se obviará y evaluará la siguiente condición

... print("Lenguaje de programación: C")

... elif lenguaje == "Python": # Se pueden añadir tantos bloques "elif" como se quiera

... print("Lenguaje de programación: Python")

... else: # En caso de que ninguna de las anteriores condiciones fuera cierta, se ejecutaría este bloque

... print("Lenguaje de programación: indefinido")

...

Lenguaje de programación: Python

>>> if verdadero and lenguaje == "Python": # Uso de "and" para comprobar que ambas condiciones son verdaderas

... print("Verdadero y Lenguaje de programación: Python")

...

Verdadero y Lenguaje de programación: Python

Bucle for

El bucle for es similar a foreach en otros lenguajes. Recorre un objeto iterable, como una lista, una tupla o un generador, y por cada elemento del iterable ejecuta el bloque de código interno. Se define con la palabra clave for seguida de un nombre de variable, seguido de in, seguido del iterable, y finalmente el bloque de código interno. En cada iteración, el elemento siguiente del iterable se asigna al nombre de variable especificado:

>>> lista = ["a", "b", "c"]

>>> for i in lista: # Iteramos sobre una lista, que es iterable

... print(i)

...

a

b

c

>>> cadena = "abcdef"

>>> for i in cadena: # Iteramos sobre una cadena, que también es iterable

... print(i, end=', ') # Añadiendo end=', ' al final hacemos que no introduzca un salto de línea, sino una coma y un espacio

...

a, b, c, d, e, f,

Bucle while

El bucle while evalúa una condición y, si es verdadera, ejecuta el bloque de código interno. Continúa evaluando y ejecutando mientras la condición sea verdadera. Se define con la palabra clave while seguida de la condición, y a continuación el bloque de código interno:

>>> numero = 0

>>> while numero < 10:

... print(numero, end=" ")

... numero += 1 # Un buen programador modificará las variables de control al finalizar el ciclo while

...

0 1 2 3 4 5 6 7 8 9

Listas y Tuplas

* Para declarar una lista se usan los corchetes [], en cambio, para declarar una tupla se usan los paréntesis (). En ambas los elementos se separan por comas, y en el caso de las tuplas es necesario que tengan como mínimo una coma.
* Tanto las listas como las tuplas pueden contener elementos de diferentes tipos. No obstante, las listas suelen usarse para elementos del mismo tipo en cantidad variable mientras que las tuplas se reservan para elementos distintos en cantidad fija.
* Para acceder a los elementos de una lista o tupla se utiliza un índice entero (empezando por "0", no por "1"). Se pueden utilizar índices negativos para acceder elementos a partir del final.
* Las listas se caracterizan por ser mutables, es decir, se puede cambiar su contenido en tiempo de ejecución, mientras que las tuplas son inmutables ya que no es posible modificar el contenido una vez creadas.

Listas

>>> lista = ["abc", 42, 3.1415]

>>> lista[0] # Acceder a un elemento por su índice

'abc'

>>> lista[-1] # Acceder a un elemento usando un índice negativo

3.1415

>>> lista.append(True) # Añadir un elemento al final de la lista

>>> lista

['abc', 42, 3.1415, True]

>>> del lista[3] # Borra un elemento de la lista usando un índice (en este caso: True)

>>> lista[0] = "xyz" # Re-asignar el valor del primer elemento de la lista

>>> lista[0:2] # Mostrar los elementos de la lista del índice "0" al "2" (sin incluir este último)

['xyz', 42]

>>> lista\_anidada = [lista, [True, 42]] # Es posible anidar listas

>>> lista\_anidada

[['xyz', 42, 3.1415], [True, 42]]

>>> lista\_anidada[1][0] # Acceder a un elemento de una lista dentro de otra lista (del segundo elemento, mostrar el primer elemento)

True

Tuplas

>>> tupla = ("abc", 42, 3.1415)

>>> tupla[0] # Acceder a un elemento por su índice

'abc'

>>> del tupla[0] # No es posible borrar (ni añadir) un elemento en una tupla, lo que provocará una excepción

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: 'tuple' object doesn't support item deletion

>>> tupla[0] = "xyz" # Tampoco es posible re-asignar el valor de un elemento en una tupla, lo que también provocará una excepción

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: 'tuple' object does not support item assignment

>>> tupla[0:2] # Mostrar los elementos de la tupla del índice "0" al "2" (sin incluir este último)

('abc', 42)

>>> tupla\_anidada = (tupla, (True, 3.1415)) # También es posible anidar tuplas

>>> 1, 2, 3, "abc" # Esto también es una tupla, aunque es recomendable ponerla entre paréntesis (recuerde que requiere, al menos, una coma)

(1, 2, 3, 'abc')

>>> (1) # Aunque se encuentra entre paréntesis, esto no es una tupla, ya que no posee al menos una coma, por lo que únicamente aparecerá el valor

1

>>> (1,) # En cambio, en este otro caso, sí es una tupla

(1,)

>>> (1, 2) # Con más de un elemento no es necesaria la coma final

(1, 2)

>>> (1, 2,) # Aunque agregarla no modifica el resultado

(1, 2)

Diccionarios

* Para declarar un diccionario se usan las llaves {}. Contienen elementos separados por comas, donde cada elemento está formado por un par clave:valor (el símbolo : separa la clave de su valor correspondiente).
* Los diccionarios son mutables, es decir, se puede cambiar el contenido de un valor en tiempo de ejecución.
* En cambio, las claves de un diccionario deben ser inmutables. Esto quiere decir, por ejemplo, que no podremos usar ni listas ni diccionarios como claves.
* El valor asociado a una clave puede ser de cualquier tipo de dato, incluso un diccionario.

>>> diccionario = {"cadena": "abc", "numero": 42, "lista": [True, 42]} # Diccionario que tiene diferentes valores por cada clave, incluso una lista

>>> diccionario["cadena"] # Usando una clave, se accede a su valor

'abc'

>>> diccionario["lista"][0] # Acceder a un elemento de una lista dentro de un valor (del valor de la clave "lista", mostrar el primer elemento)

True

>>> diccionario["cadena"] = "xyz" # Re-asignar el valor de una clave

>>> diccionario["cadena"]

'xyz'

>>> diccionario["decimal"] = 3.1415927 # Insertar un nuevo elemento clave:valor

>>> diccionario["decimal"]

3.1415927

>>> diccionario\_mixto = {"tupla": (True, 3.1415), "diccionario": diccionario} # También es posible que un valor sea un diccionario

>>> diccionario\_mixto["diccionario"]["lista"][1] # Acceder a un elemento dentro de una lista, que se encuentra dentro de un diccionario

42

>>> diccionario = {("abc",): 42} # Sí es posible que una clave sea una tupla, pues es inmutable

>>> diccionario = {["abc"]: 42} # No es posible que una clave sea una lista, pues es mutable, lo que provocará una excepción

Traceback (most recent call last):

File "<stdin>", line 1, in <module>

TypeError: unhashable type: 'list'

Sentencia match-case

Python cuenta con la estructura match-case desde la versión 3.10. Esta tiene el nombre de Structural Pattern Matching.

match variable:

case condicion:

# codigo

case condicion:

# codigo

case condicion:

# codigo

case \_:

# codigo

Cabe destacar que esta funcionalidad es considerablemente más compleja que el conocido switch-case de la mayoría de lenguajes de programación, ya que no solo permite realizar una comparación del valor, si no que también puede comprobar el tipo del objeto, y sus atributos. Además, puede realizar un desempaquetado directo de secuencias de datos, y comprobarlos de forma específica.

En el siguiente ejemplo, se comprueban los atributos de nuestra instancia de Punto. Si en estos no se cumple que x = 10 y y = 40, se pasará a la siguiente condición.

Es importante anotar que Punto(x=10, y=40) no está construyendo un nuevo objeto, aunque pueda parecerlo.

from dataclasses import dataclass

@dataclass

class Punto:

x: int

y: int

coordenada = Punto(10, 34)

match coordenada:

case Punto(x=10, y=40): # los atributos "x" e "y" tienen el valor specificado

print("Coordenada 10, 40")

case Punto(): # si es una instancia de Punto

print("es un punto")

case \_: # ninguna condición cumplida (default)

print("No es un punto")

En versiones anteriores, existen diferentes formas de realizar esta operación lógica de forma similar:

Usando if, elif, else

Podemos usar la estructura de la siguiente manera:

>>> if condicion1:

... hacer1

>>> elif condicion2:

... hacer2

>>> elif condicion3:

... hacer3

>>> else:

... hacer

En esa estructura se ejecutara controlando la condicion1, si no se cumple pasara a la siguiente y así sucesivamente hasta entrar en el else. Un ejemplo práctico sería:

>>> def calculo(op, a, b):

... if op == 'sum':

... return a + b

... elif op == 'rest':

... return a - b

... elif op == 'mult':

... return a \* b

... elif op == 'div':

... return a / b

... else:

... return None

>>>

>>> print(calculo('sum',3,4))

7

Podríamos decir que el lado negativo de la sentencia armada con if, elif y else es que si la lista de posibles operaciones es muy larga, las tiene que recorrer una por una hasta llegar a la correcta.

Usando diccionarios

Podemos usar un diccionario para el mismo ejemplo:

>>> def calculo(op, a, b):

... return {

... 'sum': lambda: a + b,

... 'rest': lambda: a - b,

... 'mult': lambda: a \* b,

... 'div': lambda: a/b

... }.get(op, lambda: None)()

>>>

>>> print(calculo('sum',3,4))

7

De esta manera, si las opciones fueran muchas, no recorrería todas; solo iría directamente a la operación buscada en la última línea (.get(op, lambda: None)()) y estaríamos dando una opción por defecto. El motivo por el que se usan expresiones lambda dentro del diccionario es para prevenir la ejecución de las instrucciones que contienen a la hora de definir el diccionario. Este únicamente define funciones como valores del diccionario, y posteriormente, al obtener estas mediante get(), se llama a la función, ejecutando la expresión que esta contiene.

Conjuntos

* Los conjuntos se construyen mediante la expresión set(items), donde items es cualquier objeto iterable, como listas o tuplas. Los conjuntos no mantienen el orden ni contienen elementos duplicados.
* Se suelen utilizar para eliminar duplicados de una secuencia, o para operaciones matemáticas como intersección, unión, diferencia y diferencia simétrica.

>>> conjunto\_inmutable = frozenset(["a", "b", "a"]) # Se utiliza una lista como objeto iterable

>>> conjunto\_inmutable

frozenset(['a', 'b'])

>>> conjunto1 = set(["a", "b", "a"]) # Primer conjunto mutable

>>> conjunto1

set(['a', 'b'])

>>> conjunto2 = set(["a", "b", "c", "d"]) # Segundo conjunto mutable

>>> conjunto2

set(['a', 'c', 'b', 'd']) # Los conjuntos no mantienen el orden, como los diccionarios

>>> conjunto1 & conjunto2 # Intersección

set(['a', 'b'])

>>> conjunto1 | conjunto2 # Unión

set(['a', 'c', 'b', 'd'])

>>> conjunto1 - conjunto2 # Diferencia (1)

set([])

>>> conjunto2 - conjunto1 # Diferencia (2)

set(['c', 'd'])

>>> conjunto1 ^ conjunto2 # Diferencia simétrica

set(['c', 'd'])

Listas por comprensión

Una lista por comprensión (en inglés list comprehension) es una expresión compacta para definir listas. Al igual que lambda, aparece en lenguajes funcionales. Ejemplos:

>>> range(5) # La función range devuelve una lista, empezando en 0 y terminando con el número indicado menos uno

[0, 1, 2, 3, 4]

>>> [i \* i for i in range(5)] # Por cada elemento del rango, lo multiplica por sí mismo y lo agrega al resultado

[0, 1, 4, 9, 16]

>>> lista = [(i, i + 2) for i in range(5)]

>>> lista

[(0, 2), (1, 3), (2, 4), (3, 5), (4, 6)]

Funciones

* Las funciones se definen con la palabra clave def, seguida del nombre de la función y sus parámetros. Otra forma de escribir funciones, aunque menos utilizada, es con la palabra clave lambda (que aparece en lenguajes funcionales como Lisp).
* El valor devuelto en las funciones con def será el dado con la instrucción return.
* Las funciones pueden recibir parámetros especiales para manejar el exceso de argumentos.
  + El parámetro \*args recibe como una tupla un número variable de argumentos posicionales.
  + El parámetro \*\*kwargs recibe como un diccionario un número variable de argumentos por palabras clave.

def:

>>> def suma(x, y=2):

... return x + y # Retornar la suma del valor de la variable "x" y el valor de "y"

...

>>> suma(4) # La variable "y" no se modifica, siendo su valor: 2

6

>>> suma(4, 10) # La variable "y" sí se modifica, siendo su nuevo valor: 10

14

\*args:

>>> def suma(\*args):

... resultado = 0

... # Se itera la tupla de argumentos

... for num in args:

... resultado += num # Suma todos los argumentos

... return resultado # Retorna el resultado de la suma

...

>>> suma(2, 4)

6

>>> suma(1, 3, 5, 7, 9) # No importa el número de variables posicionales que se pasen a la función

25

\*\*kwargs:

def suma(\*\*kwargs):

... resultado = 0

... # Se itera el diccionario de argumentos

... for key, value in kwargs.items():

... resultado += value # Suma todos los valores de los argumentos

... return resultado

...

>>>suma(x=1, y=3)

4

>>>suma(x=2, y=4, z=6) # No importa el número de variables por clave que se pasen a la función

12

lambda:

>>> suma = lambda x, y=2: x + y

>>> suma(4) # La variable "y" no se modifica, siendo su valor: 2

6

>>> suma(4, 10) # La variable "y" sí se modifica, siendo su nuevo valor: 10

14

Clases

* Las clases se definen con la palabra clave class, seguida del nombre de la clase y, si hereda de otras clases, los nombres de estas.
* En Python 2.x era recomendable que una clase heredase de object, en Python 3.x ya no hace falta.
* En una clase, un método equivale a una función, y un atributo equivale a una variable.
* init es un método especial que se ejecuta al instanciar la clase, se usa generalmente para inicializar atributos y ejecutar métodos necesarios. Al igual que todos los métodos en Python, debe tener al menos un parámetro (generalmente se utiliza self). El resto de parámetros serán los que se indiquen al instanciar la clase.
* Los atributos que se desee que sean accesibles desde fuera de la clase se deben declarar usando self. delante del nombre.
* En Python no existe el concepto de encapsulamiento, por lo que el programador debe ser responsable de asignar los valores a los atributos.

>>> class Persona():

... def \_\_init\_\_(self, nombre, edad):

... self.nombre = nombre # Un atributo cualquiera

... self.edad = edad # Otro atributo cualquiera

... def mostrar\_edad(self): # Es necesario que, al menos, tenga un parámetro, generalmente self

... print(self.edad) # mostrando un atributo

... def modificar\_edad(self, edad): # Modificando edad

... if 0 > edad < 150: # Se comprueba que la edad no sea menor que 0 (algo imposible) ni mayor que 150 (algo realmente difícil)

... return False

... else: # Si está en el rango 0-150, entonces se modifica la variable

... self.edad = edad # Se modifica la edad

...

>>> p = Persona('Alicia', 20) # Instanciando la clase. Como se puede ver, no se especifica el valor de self

>>> p.nombre # La variable "nombre" del objeto sí es accesible desde fuera

'Alicia'

>>> p.nombre = 'Andrea' # Y por tanto, se puede cambiar su contenido

>>> p.nombre

'Andrea'

>>> p.mostrar\_edad() # Se llama a un método de la clase

20

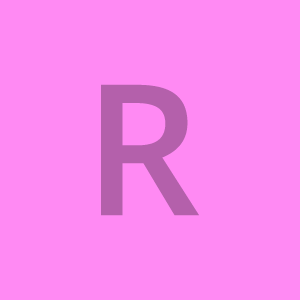
>>> p.modificar\_edad(21) # Es posible cambiar la edad usando el método específico que hemos hecho para hacerlo de forma controlada

>>> p.mostrar\_edad()

21

“

# HTML, CSS y Javascript, ¿cuáles son las diferencias?



rafaella-ballerini

Estos son los tres lenguajes principales que se utilizan en el front-end de nuestra aplicación, es decir, se utilizan en el lado del cliente en el propio navegador. Pero ¿cuál es la diferencia entre ellos? ¿Son realmente todos lenguajes de programación? ¡Descubre más aquí en este artículo!

### **Un poco de la historia de cada uno**

Intente imaginar cómo era hace unos años, cuando las empresas que querían compartir documentos entre ellas, o entre sus propios empleados, no podían hacerlo de forma tan rápida y práctica como hoy, donde con un solo clic podemos hacer todo este transporte. Para ellos, era necesario utilizar otras herramientas que requieren mucho más tiempo. Así, en 1991, Tim Berners-Lee tuvo una gran idea que no solo facilitaría el día a día de empresas, sino que cambiaría el mundo: diseñó el HTML.

Este lenguaje de marcado, por lo tanto, se desarrolló para hacer más práctico el compartido de documentos. Sin embargo, a partir del año siguiente, se creó la World Wide Web (WWW), una red de alcance mundial, que hace de HTML una herramienta utilizada para todo y por todos.

CSS, por otro lado, surgió como resultado de este enorme crecimiento en HTML. Cuanto más se usaba HTML, más usaban los desarrolladores su creatividad para hacer que las páginas fueran hermosas y con estilo. Sin embargo, toda esta parte estética fue escrita en el mismo archivo que la parte estructural, dejándolos gigantes y cada vez más difíciles de entender. Así, en 1995, se creó el CSS para cuidar la parte estética de la página, aparte de la estructura, papel del HTML.

Finalmente, se creó Javascript, fue a partir de una gran competencia entre los gigantes del mercado en ese momento: Microsoft y Netscape.

Todo empezó con la creación de un lenguaje de servidor por Netscape para implementarlo en los servidores web de la empresa. Cuando Microsoft descubrió que esta idea podía tener éxito, no perdió el tiempo y comenzó a desarrollar el lenguaje JScript para implementarlo en sus propios servidores web ISS. Al darse cuenta de que su idea estaba siendo copiada, Netscape decidió desarrollar una tecnología del lado del cliente que se ejecutara en el propio navegador. Terminó haciendo todo más rápido. Por lo tanto, Microsoft volvió a ver una gran oportunidad de mercado y decidió desarrollar un sistema similar al de su competidor.

Entonces había dos tecnologías "client-side": JScript de Microsoft y Javascript de Netcape, cada uno de los cuales funciona solo en los navegadores de sus respectivas empresas. Para estandarizar Javascript, Netscape no perdió tiempo en enviarlo a la empresa ECMA, lo que le permitió incorporarlo a la mayoría de los navegadores existentes. De ahí surgió ECMAscript, que no era más que el nuevo nombre de Javascript, que aún hoy no se usa mucho.

#### **HTML**



Su nombre proviene del inglés Hyper Text Markup Language, que significa "lenguaje de marcado de hipertexto".

¿Qué quieres decir con "lenguaje de marcado"? Bueno, HTML no se considera un lenguaje de programación como los demás y ya sabrás por qué.

Imagínate una página web, como la que estás leyendo ahora mismo. Puedes ver que hay varios elementos separados, como encabezado, título, párrafos, imágenes y tantos otros. Toda la organización de estos elementos se realiza mediante HTML. Se utiliza para crear toda la estructura de la página y, para ello, utiliza las famosas tags (etiquetas) para indicar dónde se implementará cada tipo de elemento.

Por ejemplo, si desea insertar un párrafo en la pantalla, puedes usar la tag <p> para eso, colocando el texto deseado dentro del elemento, como se muestra a continuación:

<p>Este es un párrafo</p>

Por lo tanto, existen varias tags de diferentes estructuras para ensamblar el esqueleto de la página. Pero, ¿cómo es posible que este párrafo sea azul? ¿O cambiar la fuente de la letra? Aquí viene nuestra próxima tecnología: CSS.

### **CSS**



Cascading Style Sheet, más conocido como CSS, es un lenguaje de estilos que, como HTML, tampoco se considera un lenguaje de programación. Es la encargada de separar la parte estructural de la aplicación (que estará en manos de HTML) de la parte estética. Para usar CSS, usamos la siguiente sintaxis:

selector {

propiedad: valor;

}

El selector será el elemento que queramos estilizar (podría ser una tag, una clase, un identificador ...), la propiedad será lo que vamos a cambiar (como color, tamaño de fuente, ancho ...) y el valor será realmente el cambio (como: rojo para color, 18px para tamaño de fuente, y así por delante).

Entonces, cuando usamos una tag HTML como la tag <p> que usamos como ejemplo anterior, podemos diseñarla de la siguiente manera:

p {

color: blue;

}

Con eso, todos los párrafos se volverán azules.

Un dato más interesante: si traducimos el nombre de esta tecnología, tendremos “hoja de estilo en cascada”. ¿Qué significa eso?

CSS se puede escribir dentro del archivo HTML, usando el estilo como elemento <style> o como atributo de algún otro elemento <p style="">. También se puede escribir en un archivo separado, solo CSS, importándolo al documento HTML de la siguiente manera:

<link rel="stylesheet" href="nombre\_del\_archivo\_css.css">

Sin embargo, algo muy importante a destacar es la posibilidad de utilizar más de un archivo CSS al mismo tiempo, para estilizar la aplicación. De ahí la palabra "cascada". Esto permite varias interacciones diferentes, pero debes tener cuidado de no perderte y confundir el código, siempre siguiendo las reglas para esto.

### **Javascript**



Para completar la gran tríada del desarrollo front-end, contamos con Javascript, el único lenguaje de programación de las 3 tecnologías mencionadas.

Puedes ver que ya hemos creado la estructura de la página con HTML y hemos aplicado estilo a los elementos con CSS. Pero, ¿qué pasa con las funcionalidades dinámicas que vemos cuando abrimos un sitio web?, ¿cómo suceden?

Esto lo hace Javascript. Agrega movimiento a las páginas web, además de permitir el procesamiento y transformación de los datos enviados y recibidos. Te permite crear contenidos que se actualizan de forma dinámica y animada, dando vida a aplicaciones que antes solo estaban estructuradas estáticamente con HTML.

Siempre que vea un cuadro de alerta sonando en un sitio web, o un mecanismo de autocompletar en algún campo, sepa que Javascript está funcionando. Un ejemplo de código usando el cuadro de alerta:

alert(‘Hello World’);

Al igual que CSS, Javascript se puede escribir dentro del código HTML:

<script> alert(‘Hello World’); </script>

Cómo en un archivo separado, importándose dentro del atributo de ese mismo elemento:

<script src="nombre\_del\_archivo\_js.js"></script>

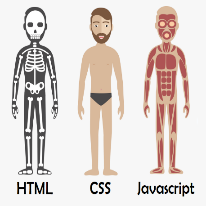
Algo importante a tener en cuenta es que este lenguaje se puede utilizar tanto en el lado del cliente como en el lado del servidor, utilizando tecnologías necesarias para ello, como Node.js.

### **El papel de cada uno en la aplicación.**

Por tanto, después de conocer un poco sobre cada una de estas tecnologías, es posible ver las diferentes funciones que desempeñan dentro de una página web:

* [**HTML**](https://www.aluracursos.com/cursos-online-front-end/html-css): lenguaje de marcado utilizado para estructurar elementos de la página, como párrafos, enlaces, títulos, tablas, imágenes e incluso videos.
* [**CSS**](https://www.aluracursos.com/cursos-online-front-end/html-css): lenguaje de estilo utilizado para definir colores, fuentes, tamaños, posicionamiento y cualquier otro valor estético para los elementos de la página.
* [**Javascript**](https://www.aluracursos.com/cursos-online-front-end/javascript): lenguaje de programación utilizado para hacer la página con más movimiento, pudiendo actualizar elementos de forma dinámica y manejar mejor los datos enviados y recibidos en la página.

Comparando cada uno de ellos ahora, podemos usar el cuerpo humano como una página web como ejemplo, de la siguiente manera:



HTML es el esqueleto, formado por los huesos que lo sostienen. CSS es la piel, el cabello y las ropas, creando la apariencia que realmente vemos cuando miramos a alguien. Y por fin, Javascript es el músculo que le da movimiento al cuerpo.

Si te interesan los temas de tecnología, desarrollo web y programación, les invitamos a conocer los cursos de [**Alura Latam**](https://www.aluracursos.com/), contamos con diversos cursos en el área de [**Front End**](https://www.aluracursos.com/cursos-online-front-end), [**Programación**](https://www.aluracursos.com/cursos-online-programacao), [**Data Science**](https://www.aluracursos.com/cursos-online-data-science) e [**Innovación y Gestión**](https://www.aluracursos.com/cursos-online-inovacao-gestao).

Puedes leer también:

* [¿Qué es front-end y back-end?](https://www.aluracursos.com/blog/que-es-front-end-y-back-end)
* [Comenzando con Front-end](https://www.aluracursos.com/blog/comenzando-con-front-end)
* [Empezar a programar es con JavaScript](https://www.aluracursos.com/blog/empezar-a-programar-es-con-javascript)

# Curso de **HTML y CSS: ambientes de desarrollo, estructura de archivos y tags**

# 03 Para saber más: la importancia de la documentación

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87285/next)

¿Qué es?

La [documentación](https://www.w3schools.com/html/html_intro.asp) es una guía que todo desarrollador debe tomar en serio en su día a día. A través de ella, aprendemos cómo funcionan los lenguajes de programación, así como las herramientas y bibliotecas en el mundo de la tecnología.

Su importancia:

La documentación es crucial para el aprendizaje y el desarrollo de aplicaciones. Después de todo, ¿quién mejor que la persona que creó la herramienta para orientarnos sobre sus funcionalidades, verdad?

Cuándo debemos utilizarla:

Debemos leer la documentación siempre que necesitemos conocer la estructura de un método o cuando queremos saber algún comando o recurso de una biblioteca, e incluso cuando olvidamos cierta funcionalidad y necesitamos recordarla.

Otra forma de ayuda:

También existen las comunidades de tecnología y programación, que son muy útiles para resolver nuestras dudas y aprender más, como por ejemplo [Stack Overflow](https://stackoverflow.com/), que es una de las mayores comunidades de ayuda sobre programación y tecnología en la actualidad. Es una comunidad donde encontrarás preguntas sobre casi todos los lenguajes de programación, entre otras herramientas; es muy recomendable acceder a ella.

# 09 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87276/next)

1 - Construyendo la estructura básica de HTML

Acabas de comenzar una pasantía en una empresa de desarrollo web y tu primera tarea es crear un documento HTML básico para un nuevo proyecto. El objetivo es construir la estructura fundamental de un archivo HTML, incluyendo las etiquetas esenciales: <!DOCTYPE html>, <html>, <head> y <body>. Además, dentro de la etiqueta <head>, agrega una etiqueta <title> con un título de tu elección para la página. Recuerda seguir las prácticas de indentación correctas para mantener el código organizado.

2 - Añadiendo contenido a HTML

Ahora que la estructura básica de tu documento HTML está lista, tu supervisor te pidió que agregues un encabezado y un párrafo al cuerpo del documento. Utiliza las etiquetas <h1> para el encabezado y <p> para el párrafo. Elige un tema para el encabezado y escribe un breve párrafo relacionado con ese tema.

3 - Trabajando con metaetiquetas y título

Necesitas optimizar la sección <head> de la página HTML existente. Agrega una metaetiqueta para establecer la codificación de caracteres como UTF-8 y cambia el título de la página a algo más descriptivo y apropiado para el contenido del sitio.

4 - Organizando contenido con etiquetas HTML

Estás desarrollando la página de inicio de un sitio para un proyecto personal. Necesitas crear un título principal y un subtítulo, seguidos por un breve párrafo explicativo. Utiliza HTML para estructurar estos elementos. Crea un archivo HTML y agrega un título principal con la etiqueta <h1>, un subtítulo con la etiqueta <h2> y un párrafo con la etiqueta <p>. Usa textos de tu elección para cada uno de estos. Recuerda seguir la jerarquía correcta de las etiquetas y verifica el resultado en el navegador.

5 - Añadiendo imágenes con accesibilidad

En este desafío, agregarás una imagen a tu proyecto de sitio personal. Elige una imagen de tu preferencia (puede ser un logotipo o una foto relacionada con el tema del sitio). Utiliza la etiqueta <img> para insertar la imagen en el cuerpo del documento HTML y no olvides incluir el atributo alt para describir la imagen, mejorando la accesibilidad del sitio. Verifica el resultado en el navegador.

6 - Listando elementos en HTML

Ahora, agrega una lista de elementos a tu sitio. Esta lista puede ser de características del proyecto, etapas de desarrollo o cualquier otra información relevante. Utiliza la etiqueta <ul> para una lista no ordenada o <ol> para una lista ordenada, y <li> para cada elemento de la lista. Verifica el diseño en el navegador después de la inserción.

7 - Explorando el impacto del DOCTYPE en HTML

Tu tarea es explorar el impacto de la declaración <!DOCTYPE html> en un documento HTML. Crea un archivo HTML básico con la estructura presentada en el informe, incluyendo elementos como <head>, <title>, <body>, <h1>, <p> e <img>. Primero, visualiza la página con la declaración <!DOCTYPE html>. Luego, elimina la declaración y visualiza nuevamente. Utiliza la herramienta de inspección del navegador para explorar las diferencias en el modo de renderización de la página, centrándote en la presencia o ausencia del "Modo Quirks".

8 - Automatizando la actualización de la página con Live Server

Instala y utiliza la extensión "Live Server" en Visual Studio Code para automatizar la actualización de tu página HTML. Realiza cambios en el código HTML, como agregar texto al párrafo, y observa cómo los cambios se reflejan automáticamente en el navegador.

# 10 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87281/next)

Lo que aprendimos en esta aula:

* La importancia de la documentación de W3S;
* Qué es HTML y por qué se considera un lenguaje de marcado;
* Cómo estructurar un documento HTML con etiquetas y elementos;
* La utilidad de la introducción <!DOCTYPE html>;
* La diferencia entre la metainformación presente en <head> y el contenido presente en <body> de una página HTML;
* Cómo crear textos alternativos (alts) para una imagen;
* Cómo acceder a las Developer Tools (Herramientas para Desarrolladores) de un navegador;
* Qué es el Quirks mode (modo peculiaridad);
* Cómo utilizar extensiones en Visual Studio Code (como Live Server, por ejemplo).

3 LAYOUT Y TAGS SEMANTICAS

# 02 Preparando el ambiente

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87879/next)

Las próximas clases se basarán en Figma, que es una herramienta de prototipado de interfaces, donde construiremos el diseño de la página del Portafolio. Para acceder, [haz clic aquí](https://www.figma.com/design/D464FfN417KSWJHIUVpMnX/Portafolio---Curso?node-id=0-1&t=OzehjJGqjpVT1KLL-0).

Para tener la misma visualización que en la clase, primero necesitas crear una cuenta o iniciar sesión en una existente. Para hacerlo, ve al sitio de [Figma](https://www.figma.com/) y haz clic en Iniciar sesión (si ya tienes una cuenta) o en Registrarse (para crear una nueva cuenta). Después de tener una cuenta conectada, [accede al proyecto en Figma.](https://www.figma.com/design/D464FfN417KSWJHIUVpMnX/Portafolio---Curso?node-id=0%3A1&t=suOeVTCs3gYoKtaM-1)

Atención: Ten en cuenta que las pestañas Diseño e Inspeccionar mostradas en el curso han experimentado actualizaciones recientes en Figma y ahora son opciones premium, es decir, de pago. Pero no te preocupes, ¡en el Figma proporcionado puedes obtener la información de los elementos a través de la pestaña Propiedades! Solo tienes que hacer clic en los componentes y verificar sus propiedades en la pestaña Design, como se muestra a continuación:

Si deseas crear una copia del diseño en tu cuenta de Figma para asegurar el acceso de edición, ve al enlace del diseño original mencionado arriba y en el menú superior de la plataforma, haz clic en el nombre del archivo: Portafolio - Curso. Se abrirán algunas opciones, haz clic en "Duplicar en tus borradores" y el archivo del proyecto se duplicará para tu uso personal.

## Activando WebGL

Si aparece el mensaje: "no podemos abrir ese archivo. No podemos abrir este archivo porque WebGL no es compatible o está desactivado en tu navegador. Si tu navegador admite WebGL, consulta este artículo de ayuda para saber cómo activarlo", significa que WebGL está desactivado en tu navegador y necesitas activarlo.

En Google Chrome, escribe lo siguiente en la barra de búsqueda: chrome://flags/. Luego, busca "WebGL" en la lista y actívalo, cambiando de Desactivado a Activado. Después de eso, aparecerá un botón para reiniciar Google Chrome. Reinicia el navegador y listo.

Si tu navegador no es Google Chrome, el procedimiento es similar. WebGL está desactivado y necesita ser activado.

# 05 Para saber más: aumenta tu productividad en el VSCode con Emmet

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87287/next)

Imagina que eres un desarrollador de software trabajando en un proyecto web en Visual Studio Code (VSCode). Estás inmerso en un entorno ágil donde la productividad y la eficiencia son fundamentales. El proyecto requiere la creación de código HTML y CSS para varias páginas y elementos. Sin embargo, escribir todo ese código repetitivo puede resultar tedioso y consumir mucho tiempo.

El problema

La problemática radica en la necesidad de acelerar el proceso de escritura de código HTML y CSS en VSCode. Los desarrolladores a menudo se encuentran con tareas repetitivas, como crear varias etiquetas, clases e IDs, lo que puede llevar a errores y disminuir la velocidad de desarrollo. Además, los nuevos miembros del equipo pueden tardar en familiarizarse con el código existente si la estructura no está optimizada.

La solución

Para resolver este problema, podemos contar con la ayuda de Emmet, una extensión potente y ampliamente utilizada en VSCode. Emmet: <https://docs.emmet.io/> es una herramienta que permite escribir códigos HTML y CSS de manera extremadamente rápida y productiva. Utiliza abreviaciones para generar estructuras complejas de código con solo algunos comandos, mejorando significativamente la eficiencia del desarrollador.

¿Qué es Emmet?

Emmet, anteriormente conocido como Zen Coding, es una herramienta avanzada de codificación desarrollada por Sergey Chikuyonok. Está diseñado para facilitar la escritura rápida y simplificada de código HTML y CSS. Emmet ofrece una forma inteligente de crear estructuras complejas mediante abreviaturas fáciles de recordar.

Emmet utiliza la sintaxis similar a CSS para crear código HTML y CSS. Con solo unas pocas teclas, puedes generar bloques completos de código rápidamente. Por ejemplo, al escribir ul>li\*3, Emmet creará automáticamente una lista no ordenada (ul) con tres elementos de lista (li).

Instalación de Emmet en VSCode

Si aún no tienes Emmet instalado en tu VSCode, sigue estos pasos:

1. Abre Visual Studio Code.
2. Haz clic en la pestaña de extensiones en el menú lateral izquierdo o usa el atajo Ctrl+Shift+X (Windows/Linux) o Cmd+Shift+X (macOS).
3. En la barra de búsqueda, escribe "Emmet" y presiona Enter.
4. Localiza la extensión "Emmet" en la lista de resultados y haz clic en "Instalar".
5. Después de la instalación, verás la opción "Emmet" activada y lista para usar en VSCode.

Cómo usar Emmet

Con Emmet instalado, puedes comenzar a utilizar sus abreviaturas para acelerar el desarrollo. Algunas abreviaturas comunes incluyen:

* html: Genera la estructura básica de un documento HTML.
* ul>li\*5: Crea una lista no ordenada con cinco elementos de lista.
* div>h1+p: Crea una div que contiene un encabezado (h1) seguido de un párrafo (p).
* input[type=text]+input[type=password]: Crea dos campos de entrada, uno para texto y otro para contraseña.

Estas son solo algunas de las muchas abreviaturas que Emmet ofrece. A medida que te familiarices con ellas, tu productividad aumentará significativamente.

Conclusión

Emmet es una extensión imprescindible para cualquier desarrollador que desee optimizar su productividad en Visual Studio Code. Con sus abreviaturas intuitivas, permite la creación rápida y eficiente de código HTML y CSS, ahorrando tiempo y evitando errores comunes.

# 07 Para saber más: la estructura básica del HTML

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87288/next)

Para crear un archivo HTML funcional, debemos seguir un patrón de construcción utilizando un conjunto de elementos, es decir, hipertextos que se conectan entre sí formando la página. Como se vio anteriormente, los elementos HTML, también llamados etiquetas HTML, se utilizan para informar al navegador sobre el tipo de estructura que se está construyendo, ya sea títulos, párrafos, imágenes, enlaces, entre otros. Un ejemplo práctico de esta estructura básica sería:

<!DOCTYPE html>

<html lang="es-MX">

<head>

<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">

<title>Document</title>

</head>

<body>

</body>

</html>

Etiquetas semánticas

Cuando comenzamos un archivo HTML, es importante saber qué etiquetas deben implementarse y comprender sus funciones dentro del código. Para facilitar este proceso, utilizamos etiquetas semánticas, que son descriptivas sobre el contenido que almacenan, como es el caso de las etiquetas <header>, <main> y <footer>, que conocimos en esta lección. Sirven tanto para optimizar la lectura por parte de los navegadores como para las personas desarrolladoras que realizarán el mantenimiento del código.

Para aprender más sobre las etiquetas que forman parte de la base de un archivo HTML, puedes leer la documentación de MDN "[Semántica](https://developer.mozilla.org/en-US/docs/Glossary/Semantics)". También recomendamos el artículo "[¿Qué es HTML y sus tags? Estructura básica](https://www.aluracursos.com/blog/html-y-sus-etiquetas)", que es el primero de una serie de cinco artículos sobre qué es HTML y sus etiquetas. Puedes encontrarlos aquí en la plataforma de Alura Latam y son perfectos para desarrolladores HTML principiantes.

# 08 Comprendiendo las meta etiquetas iniciales

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87268/next)

Imagina que estás comenzando tu trabajo como desarrollador(a) web en la creación de un nuevo sitio web sobre tecnología. En el primer día, tu jefe te pide que configures las meta etiquetas de una página HTML, asegurándote de que sea interpretada correctamente por los navegadores y dispositivos móviles. Menciona que debes recordar las meta etiquetas básicas que se enseñaron en el curso "Meta Tags" que completaste recientemente.

¿Cuál de las siguientes opciones de código deberías insertar en la sección

del HTML para cumplir con el pedido de tu jefe?

Seleccione una alternativa



<meta charset="ISO-8859-1">

<meta name="viewport" content="user-scalable=no">



<meta name="viewport" content="width=1024">

<meta name="description" content="Una página sobre tecnología">



<meta charset="UTF-8">

<meta name="author" content="Tu Nombre">



<meta charset="UTF-8">

<meta name="viewport" content="width=device-width, initial-scale=1.0">



<meta charset="ASCII">

<meta name="viewport" content="width=device-width, initial-scale=2.0">

# 09 Usando etiquetas semánticas

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87269/next)

Vimos en las clases que para escribir un título, un párrafo, necesitamos respetar una estructura básica de HTML. Ante esto, ¿cuál de las siguientes alternativas contempla la estructura correcta para obtener como resultado un título y dos párrafos según el ejemplo a continuación?

Alura Latam - ¡Sumérgete en Tecnología!

Vas a estudiar, practicar, discutir y profundizar en una plataforma que respira tecnología. Sumérgete con profundidad y navega en otras áreas de Tecnología. Profesional en T.

Seleccione una alternativa



<h1>Alura Latam - ¡Sumérgete en Tecnología!</h1>

<p>Vas a estudiar, practicar, discutir y profundizar en una plataforma que respira tecnología.</p>

<p>Sumérgete con profundidad y navega en otras áreas de Tecnología. Profesional en T </p>



<title>Alura Latam - ¡Sumérgete en Tecnología!</title>

<p>Vas a estudiar, practicar, discutir y profundizar en una plataforma que respira tecnología.</p>

<p>Sumérgete con profundidad y navega en otras áreas de Tecnología. Profesional en T</p>



<h1>Alura Latam - ¡Sumérgete en Tecnología!</h1>

<h2>Vas a estudiar, practicar, discutir y profundizar en una plataforma que respira tecnología.</h2>

<p>Sumérgete con profundidad y navega en otras áreas de Tecnología. Profesional en T </p>



<h1>Alura Latam - ¡Sumérgete en Tecnología!<h1>

<p>Vas a estudiar, practicar, discutir y profundizar en una plataforma que respira tecnología.<p>

<p>Sumérgete con profundidad y navega en otras áreas de Tecnología. Profesional en T<p>

# 10 Diferencia entre ancla y botón

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87270/next)

Hemos visto que para desarrollar diferentes funcionalidades en HTML, es necesario comprender el comportamiento de las etiquetas y que, aunque a menudo dos o más etiquetas sean similares en nombre o propósito, cada una es específica para cierta situación. De esta manera, pensando en las etiquetas <button> para botones y <a> para anclas, marca la alternativa correcta que indique la diferencia entre ellas:

Seleccione una alternativa



Ambas etiquetas son diferentes en nombre, pero idénticas en propósito, ya que ambas se utilizan para indicar que hay un botón clicleable listo para realizar una acción.



Ambas etiquetas son completamente diferentes, ya que mientras <button> indica un enlace clicleable que lleva al usuario a otra página, <a> es responsable de la creación visual de un botón.



La etiqueta <button> es diferente de la etiqueta <a>, ya que, además de la semántica, también tiene un propósito diferente. Usamos <button> para crear un botón de acción y <a> para indicar un enlace.

# 11 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87277/next)

1) Construyendo la estructura HTML del portafolio

Acabas de recibir un diseño de página de portafolio de tu diseñadora. El desafío es transformar ese diseño en una página web funcional. Utilizando HTML, crea la estructura básica de la página. Recuerda incluir los elementos principales: un título principal, un subtítulo, dos botones (Instagram y Github), y un espacio para la imagen de la desarrolladora. Siéntete libre de personalizar el texto y la imagen para adaptarlos a tu portafolio personal.

2) Estructurando la página con etiquetas semánticas

Ahora que entiendes la importancia de las etiquetas semánticas en HTML, es hora de aplicar ese conocimiento. Utiliza las etiquetas header, main y footer para estructurar tu página de portafolio. En el header, inserta un menú o un logotipo. En el main, agrega los elementos del portafolio como título, párrafo, botones e imagen. Por último, en el footer, coloca información como contacto o derechos de autor. Recuerda que cada elemento debe colocarse de manera lógica y estructurada para crear un sitio accesible y bien organizado.

3) Entendiendo y aplicando las etiquetas meta

Explora el uso de las etiquetas meta en el encabezado de tu HTML. Cambia la etiqueta lang a "es", ajusta el charset a "UTF-8", agrega la etiqueta meta para la compatibilidad con Edge y otra para la configuración del viewport. Finalmente, cambia el título de tu página a algo relacionado con tu proyecto, por ejemplo, "Portafolio de [Tu Nombre]". Entiende cómo cada uno de estos cambios afecta la funcionalidad y accesibilidad de tu sitio.

4) Insertando y formateando elementos en HTML

Ahora que ya tienes una base sólida sobre la estructura HTML, vamos a agregar y formatear elementos en tu página de portafolio. Sigue el diseño de Figma y comienza agregando un título (<h1>) con un resalte (<strong>) en una parte de él. Luego, inserta un párrafo (<p>) con una breve introducción sobre ti o tu empresa. Finalmente, agrega enlaces (<a>) a tus redes sociales o portafolios en línea, como Instagram y GitHub. Asegúrate de que cada elemento esté correctamente formateado y colocado según el diseño de Figma.

5) Incorporando imágenes con etiquetas HTML

¡Ha llegado el momento de dar un toque visual a tu página! Elige una imagen que te represente a ti o tu trabajo y agrégala a tu página de portafolio utilizando la etiqueta <img>. Recuerda incluir la propiedad src con la ruta de la imagen y alt con una descripción adecuada de la imagen. Esto no solo mejora la accesibilidad de tu sitio, sino que también ayuda a los motores de búsqueda a entender el contenido de la imagen.

# 12 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87282/next)

Lo que aprendimos en esta aula:

* Cómo consultar el diseño del proyecto en Figma;
* Escribir el código base del archivo HTML, utilizando las etiquetas semánticas que forman parte de la estructura básica del archivo;
* La función de cada etiqueta meta.

4 ESTILIZANDO EL PROYECTO CON CSS

# 01 Proyecto del aula anterior

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87295/next)

¿Comenzando en esta etapa? Aquí puedes descargar los archivos del proyecto que hemos avanzado hasta el aula anterior.

[Descargue los archivos en Github](https://github.com/alura-es-cursos/2067-html-css-desarrollo-estructura-archivos/tree/aula3) o haga clic [aquí](https://github.com/alura-es-cursos/2067-html-css-desarrollo-estructura-archivos/archive/refs/heads/aula3.zip) para descargarlos directamente.

# 04 Incluyendo CSS en la página

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87271/next)

En el video anterior, aprendiste que las "Hojas de Estilo en Cascada" (CSS) describen un conjunto de reglas de formato que controlan la apariencia de una página web. Pensando en eso, marca la alternativa correcta que presente la buena práctica recomendada al utilizar CSS:

Seleccione una alternativa



Se recomienda crear un archivo CSS externo con extensión .css e incluirlo en la estructura head del HTML. De esta manera, podemos tener una estructura de estilos universal para varias páginas.



Se recomienda escribir el CSS internamente en el archivo HTML a través de la etiqueta <style> dentro de la etiqueta <head> de la página. Esto permite organizar mejor nuestro código.



Lo ideal es utilizar el atributo style en elementos HTML para estilizar nuestro contenido de una manera simple y directa. Esta forma es conocida como Inline.

# 05 Estilizando el HTML con CSS

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87272/next)

La forma correcta de incluir estilos CSS en los elementos HTML es conectar la hoja de estilos correctamente en el documento HTML mediante una etiqueta <link>; de lo contrario, los estilos no se aplicarán.

De las alternativas a continuación, selecciona aquella que presenta la forma correcta de escribir la etiqueta <link> y el lugar donde debe contener esa etiqueta:

Seleccione una alternativa



<head>

<link rel="stylesheet">

</head>



<head>

<link rel="stylesheet" href="nomedoarquivo.css">

</head>



<header>

<link rel="stylesheet" href="nomedoarquivo.css">

</header>



<head>

<link rel="stylesheet" href="nomedoarquivo.css"></link>

</head>

# 06 Color de fondo

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87273/next)

¿Cuál de las alternativas cambia el color de fondo para rojo(red)?

Seleccione una alternativa



body {

background-color: blue;

color: red;

font-family: 'Comic Sans MS', sans-serif;

font-size: 16px;

line-height: 1.5;

}



body {

background-color: red;

color: blue;

font-family: 'Comic Sans MS', sans-serif;

font-size: 16px;

line-height: 1.5;

}



body {

back-color: red;

color: blue;

font-family: 'Comic Sans MS', sans-serif;

font-size: 16px;

line-height: 1.5;

}

# 07 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87278/next)

1 - Estiliza tú HTML

En esta lección, aprendimos que con CSS podemos seleccionar elementos de HTML y asignar propiedades de CSS dentro de llaves para estilizar estos elementos.

Lo hicimos asignando un color de fondo negro a la etiqueta <body> a través de la propiedad background-color y cambiamos el color del texto a blanco utilizando la propiedad color, de la siguiente manera:

body{

background-color: black;

color: white;

}

Ahora que ya sabes cómo crear una hoja de estilo para tu página HTML, es el momento de aplicar lo que aprendiste en esta lección. Modifica el estilo de la etiqueta <body>, prueba nuevos colores para el fondo y el texto, ¡y personaliza el proyecto a tu gusto! =)

No olvides aplicar la etiqueta <link rel="stylesheet" href="style.css"> dentro de la etiqueta <head> en tu archivo HTML para que tu estilo funcione correctamente.

2 - Practicando cómo utilizar el CSS

1) Creando y vinculando el archivo CSS

Vas a crear un archivo CSS llamado style.css para estilizar una página web. Después de crear el archivo, tu objetivo es vincular este archivo CSS al HTML. Para lograrlo, usarás la etiqueta <link> dentro de la etiqueta <head> del documento HTML, especificando la ruta del archivo CSS. Este proceso es crucial para que los estilos definidos en CSS se apliquen a la página HTML.

2) Definiendo estilos básicos

En el archivo CSS, define el color de fondo del cuerpo de la página (<body>) como índigo (indigo) y el color del texto como blanco (white). Guarda los cambios y verifica en el navegador si el fondo de la página es índigo y el texto es blanco.

3) Estilizando con colores y formatos

Te han desafiado a dar vida a una página web que actualmente solo tiene HTML. Tu tarea es utilizar CSS para agregar colores y formatos a los elementos de la página, inspirándote en un diseño específico proporcionado en Figma. Concéntrate en seleccionar colores adecuados para fondos, textos y bordes, además de aplicar estilos a botones y encabezados para hacerlos visualmente atractivos. Puedes usar como referencia esta tabla de nombres de colores.

4) Personalizando la página con CSS

Imagina que estás trabajando en un proyecto web y ha llegado el momento de darle vida a tu página con colores, formas y estilos. Tu tarea es crear un archivo CSS para estilizar una página HTML simple, que contiene un encabezado, un párrafo y un pie de página. Utiliza tu conocimiento para definir los colores del texto y del fondo.

5) Aplicando estilos con CSS

Ahora, vayamos un poco más allá. Necesitas crear tres archivos CSS diferentes, cada uno con un estilo único, para el mismo archivo HTML. Esto demostrará cómo el CSS puede cambiar drásticamente la apariencia de una página sin alterar el HTML. Crea estilos variados para los elementos de la página: un archivo con un estilo formal y elegante, otro con un estilo colorido y divertido, y un tercero con un aspecto minimalista y limpio.

6) Personalizando enlaces

La última tarea es estilizar los enlaces en la página. Deberás modificar la apariencia de los enlaces para que se destaquen del fondo y del texto. Utiliza propiedades CSS, como color, para personalizarlos, siguiendo las directrices del diseño proporcionado en Figma.

# 08 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87283/next)

Lo que aprendimos en esta aula:

* Qué es CSS (Hojas de Estilo en Cascada);
* Estilización en la práctica;
* Propiedades de CSS;
* Crear un archivo externo para estilizar la página;
* Integrar el archivo CSS al archivo HTML.

5 MEJORANDO LA ESTILIZACION DE TU CSS

# 03 Para saber más: eligiendo los colores del proyecto

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87289/next)

Llegó el momento de poner manos a la obra. Te desafiamos a elegir colores para tu proyecto, ya que una paleta es esencial para proporcionar una buena experiencia de usuario y enriquecer la identidad de tu página. Y para ayudarte con este desafío, hemos seleccionado algunos sitios.

Coolors

[Coolors](https://coolors.co/) tiene una interfaz muy clara. Con la barra espaciadora de tu teclado, puedes crear varias combinaciones, y una de las funciones más geniales es la opción de bloqueo, que puedes usar si te gusta solo un color. Una vez que haces clic en él, puedes seguir elaborando otras combinaciones teniendo en cuenta el color que elegiste.

Adobe Color

[Adobe Color](https://color.adobe.com/pt/create/color-wheel) presenta una Rueda de Colores que se puede ajustar de diversas maneras para obtener una armonía de colores. Puedes aplicar varias reglas de armonía de colores, como el modo análogo, monocromático, tríada, complementario, cuadrado, compuesto, entre otros.

Color Hunt

[Color Hunt](https://colorhunt.co/) ofrece diversas paletas elaboradas. Puedes encontrar la combinación que más te guste y buscar por palabras clave como pastel, vintage, neón, y así sucesivamente. Y si no encuentras ninguna que te guste, puedes crear tu propia paleta haciendo clic en los tres puntos en la esquina superior derecha de la página.

Color Tool - Material Design

Color Tool es excelente para crear, compartir y aplicar paletas de colores a la interfaz de usuario. También puedes medir el nivel de accesibilidad de cualquier combinación de colores en la pestaña de accesibilidad.

¡Ahora depende de ti! Encuentra la paleta de colores que más te guste y aplícala en el proyecto para darle tu toque personal :D

# 05 Para saber más: resaltando el texto

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87290/next)

Aprendiste a destacar el texto utilizando la etiqueta <strong>, sin embargo, hay otra etiqueta que también se utiliza mucho para esto, que es la etiqueta <span>. A diferencia de <strong>, <span> no aplica negrita por defecto, pero es una excelente forma de resaltar partes del texto en HTML. Mira:

HTML

<h1>Formación de <span>Front-end</span></h1>

CSS

h1{

font-weight: bold;

}

span{

color: #22D4FD;

border: 1px solid #22D4FD;

padding: 10px;

}

Explicando el código anterior:

font-weight: bold; (Texto en negrita)

color: #22D4FD; (Color del texto azul)

border: 1px solid #22D4FD; (Borde con 1 px de ancho, de tipo sólido y color del borde azul)

padding: 10px; (Espaciado interno de 10px)

RESULTADO:

# 06 Desafío: comparta tú proyecto con el mundo

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87274/next)

Ahora que has conocido HTML y CSS y has creado la base de tu portafolio, ¿qué tal personalizar este proyecto y darle tu toque personal?

Puedes personalizar el título y agregar información sobre ti en el párrafo de texto, poner tu foto y los enlaces a tus redes sociales, además de modificar el CSS trabajando con colores y resaltados. ¡Desata tu creatividad!

Además, ¡nos encantaría ver tu progreso en este curso! Así que siéntete libre de compartir tu proyecto con nosotros publicándolo en tus redes sociales como Instagram, Twitter y LinkedIn. Puedes hacerlo de la manera que prefieras, ya sea a través de fotos, historias, videos, etc. ¡Y no olvides etiquetar el perfil de Alura Latam, estaremos atentos a los proyectos!

# 08 Para saber más: profundizando en HTML y CSS

[Próxima Actividad](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87291/next)

Aquí tienes una lista de referencias para que profundices en tus estudios, mejores tus conocimientos y adquieras nuevas habilidades.

[1- HTML, CSS y JavaScript - Las diferencias - (Gratuito, Español, Texto)](https://www.aluracursos.com/blog/html-css-javascript-cuales-son-las-diferencias)

Este artículo ofrece una visión general de las tres principales lenguajes utilizadas en el desarrollo front-end de la programación web. El HTML, como lenguaje de marcado, se utiliza para estructurar elementos en una página web, mientras que el CSS es un lenguaje de estilo que define la apariencia estética de los elementos HTML. Por otro lado, JavaScript es un lenguaje de programación que añade dinamismo e interactividad a las páginas web.

[2 - Introducción a las etiquetas HTML - (Gratuito, Inglés, Tutorial)](https://www.w3schools.com/tags/)

Una guía detallada sobre las diversas etiquetas HTML disponibles. Cada etiqueta se explica con ejemplos prácticos, facilitando la comprensión de cómo usar las etiquetas para estructurar una página web.

[3 - Tutorial de HTML Básico - MDN Web Docs (Gratuito, Español, Online)](https://developer.mozilla.org/es/docs/Learn/Getting_started_with_the_web/HTML_basics)

MDN Web Docs presenta una guía para principiantes sobre HTML, cubriendo los conceptos fundamentales y la estructura de un documento HTML. Este tutorial es detallado y está disponible en portugués, siendo una excelente referencia para quienes están empezando.

[4 - Estructura básica de una página HTML - HTML Dog (Gratuito, Inglés, Online)](https://htmldog.com/guides/html/beginner/)

HTML Dog ofrece una guía simple y clara sobre la estructura básica de una página HTML, explicando la importancia de cada sección, incluyendo <body>, <head>, <h1>, <p> y <img>.

[5 - Tutorial de accesibilidad en la web - WebAIM (Gratuito, Inglés, Online)](https://webaim.org/intro/)

WebAIM proporciona un tutorial introductorio sobre accesibilidad en la web, esencial para entender cómo hacer que el contenido sea accesible para todos los usuarios, incluyendo aquellos que utilizan lectores de pantalla.

[6 - Modo Quirks y estándares en navegadores - MDN Web Docs (Gratuito, Español, Online)](https://developer.mozilla.org/es/docs/Web/HTML/Quirks_Mode_and_Standards_Mode)

MDN Web Docs ofrece una visión detallada sobre el Modo Quirks y el Modo Estándar, explicando cómo los navegadores interpretan el código HTML basándose en la presencia o ausencia del DOCTYPE.

[7 - Uso de extensiones en Visual Studio Code - Visual Studio Code Docs (Gratuito, Inglés, Online)](https://code.visualstudio.com/docs/editor/extension-marketplace)

Documentación oficial de Visual Studio Code sobre cómo utilizar extensiones, incluyendo la instalación y configuración de Live Server, que permite la actualización automática de la página HTML durante el desarrollo.

[8 - Extensión Live Server para Visual Studio Code - GitHub (Gratuito, Inglés, Online)](https://github.com/ritwickdey/vscode-live-server)

Página oficial de la extensión Live Server en GitHub, proporcionando detalles sobre sus funciones, instalación y uso, esencial para un desarrollo más eficiente de páginas web.

[9 - UX/UI Design: Fundamentos para la calidad en la interfaz de usuario - Interaction Design Foundation (Gratuito/Pago, Inglés, Online)](https://www.interaction-design.org/literature/topics/ux-design)

Interaction Design Foundation ofrece recursos educativos completos sobre UX/UI Design, fundamentales para entender la importancia del diseño en la experiencia del usuario.

[10 - Consejos de CSS (Gratuito, Inglés, Online)](https://css-tricks.com/guides/)

CSS Tricks presenta tutoriales completos y consejos sobre CSS, esenciales para dar estilo a páginas web según las especificaciones de diseño.

[11 - Guía de estructuración de páginas HTML con semántica - MDN Web Docs (Gratuito, Español, Online)](https://developer.mozilla.org/es/docs/Web/HTML/Element)

Una guía completa sobre la estructuración de páginas HTML, haciendo hincapié en el uso de etiquetas semánticas como <header>, <main> y <footer>, esencial para crear páginas bien organizadas y accesibles.

[12 - Utilizando Emmet para acelerar el desarrollo HTML - CSS-Tricks (Gratuito, Inglés, Online)](https://css-tricks.com/emmet/)

Un artículo detallado sobre cómo utilizar Emmet, una herramienta de productividad para el desarrollo web, para acelerar la escritura de códigos HTML, incluyendo la

[13 - Introducción a HTML5 y etiquetas semánticas - HTML.com (Gratuito, Inglés, Online)](https://html.com/html5/)

Una guía completa sobre HTML5, destacando la importancia y el uso de etiquetas semánticas como <header>, <main>, <footer> y <strong>, esenciales para crear una estructura de página clara y accesible.

[14 - Uso efectivo de las etiquetas de anclaje en HTML - W3Schools (Gratuito, Inglés, Online)](https://www.w3schools.com/html/html_links.asp)

Este recurso explica cómo utilizar las etiquetas de anclaje (<a>) en HTML, un componente fundamental para crear enlaces que redirigen a otras páginas o recursos.

[15.Cómo insertar imágenes en HTML - MDN Web Docs (Gratuito, Español, Online)](https://developer.mozilla.org/es/docs/Web/HTML/Element/img)

Una guía detallada sobre la etiqueta en HTML, incluyendo cómo usar el atributo src para insertar imágenes y la importancia del atributo alt para la accesibilidad.

[16.Prácticas recomendadas para el diseño responsivo - Smashing Magazine (Gratuito, Inglés, Online)](https://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/)

Este artículo aborda las mejores prácticas para el diseño responsivo en el desarrollo web, algo crucial para garantizar que un sitio funcione bien en todos los dispositivos y tamaños de pantalla.

[17.Introducción a CSS - W3Schools (Gratuito, Inglés, Online)](https://www.w3schools.com/css/)

Un recurso introductorio a CSS, ofreciendo lecciones paso a paso sobre cómo usar CSS para mejorar la apariencia de las páginas HTML, abordando desde la formatación básica hasta conceptos más avanzados.

[18.Cómo utilizar hojas de estilo en cascada (CSS) - MDN Web Docs (Gratuito, Inglés, Online)](https://color.adobe.com/es/create/color-wheel)

Este recurso de MDN Web Docs explica en detalle cómo usar CSS para estilizar elementos HTML, incluyendo cómo crear y vincular archivos CSS externos.

[19. Guía de colores y fuentes en CSS - Adobe Color (Gratuito, Inglés, Online)](https://color.adobe.com/es/create/color-wheel)

Esta guía de Adobe ayuda a entender cómo combinar colores y elegir fuentes en CSS, alineando el diseño de la página web con las especificaciones del proyecto en Figma.

# Curso de **HTML y CSS: Clases Posicionamiento y Flexbox**

# **02 Preparando el ambiente**

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87407/next)

¡Hola Dev!

¡Bienvenido al curso de HTML y CSS: Clases, posicionamiento, y Flexbox!

En este entrenamiento, vamos a utilizar el editor de código [Visual Studio Code](https://code.visualstudio.com/download). Puedes descargarlo tanto para Windows como para Linux o Mac.

Además, estamos utilizando [Chrome](https://www.google.com/intl/en_us/chrome/) como navegador principal, y es importante tenerlo instalado en tu computadora.

Si quieres echar un vistazo al diseño, ¡aquí está el [Figma](https://www.figma.com/file/D464FfN417KSWJHIUVpMnX/Portafolio---Curso?type=design&node-id=0%3A1&mode=design&t=ChSTFZhyABgsuLT0-1) del proyecto!

Y por último, pero no menos importante, para profundizar en CSS y entender cómo funciona la estilización de páginas con tecnologías del mundo real, necesitas descargar el [archivo zip](https://github.com/alura-es-cursos/2067-html-css-desarrollo-estructura-archivos/archive/refs/heads/aula5.zip) del proyecto base o acceder a los archivos en [Github](https://github.com/alura-es-cursos/2067-html-css-desarrollo-estructura-archivos/tree/aula5).

¿Listo para empezar?

# **04 Para saber más: class**

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87408/next)

## Class

Ahora ya sabes que el atributo class permite al CSS seleccionar y acceder a elementos específicos a través de selectores de clase, pero para entender de manera más clara y objetiva, puedes acceder a la documentación oficial de MDN para despejar dudas.

## Nombres de clases en CSS

¿Quieres entender las buenas prácticas para nombrar clases en CSS? Te recomendamos leer el artículo [Nombres de clases en CSS](https://www.aluracursos.com/blog/nombre-de-clases-en-css), que aborda de manera simple y práctica este concepto.

# **05 Los selectores de clase**

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87443/next)

Los selectores definen qué elementos aplica un conjunto de reglas CSS. Imagina que necesitas editar el color de un título <h1>, de negro a rojo, que tiene la propiedad class="titulo". En este caso, ¿cuál sería la sintaxis correcta en CSS para estilizar solo el elemento que tiene esa clase específica?

Analiza las opciones a continuación y marca la alternativa correcta:

* Alternativa correta

h1 {

color: red;

}

* Alternativa correta

.titulo {

color: red;

}

* Para estilizar una clase en CSS, es necesario utilizar el punto antes de llamar al nombre asignado a la clase. Además, es la propiedad color de CSS la que define el valor del color de un elemento, y en este caso, el valor red corresponde al color rojo.
* Alternativa correta

titulo {

color: red;

}

# **06 Para saber más: selectores**

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87409/next)

## Selectores

Ahora que has visto y practicado el concepto de selectores, te invito a leer la documentación de [MDN](https://developer.mozilla.org/es/docs/Web/CSS/CSS_Selectors) para profundizar aún más en el tema.

## Reset CSS

Para adentrarte aún más en este mundo del Front-end, te sugiero que leas el artículo [Reset CSS: ¿Qué es?, Ejemplos, Cómo Crear y Utilizar](https://www.aluracursos.com/blog/reset-css-que-es-ejemplos-como-crear-y-utilizar). Es muy relevante para nuestro entrenamiento, ya que además de mostrar de manera detallada cómo funciona esta herramienta, también aborda conceptos y prácticas que seguramente te ayudarán en tu camino como desarrollador.

# 08 Reset CSS

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87445/next)

Has aprendido sobre la importancia de resetear el CSS antes de estilizar cualquier elemento de una página, ya que esta parte del código elimina todos los estilos por defecto de los navegadores. Teniendo esto en cuenta, analiza el error del código a continuación y marca la alternativa que indique la sintaxis correcta del reset.

\* {

margin: 100;

border: 100;

}

Seleccione una alternativa



\* {

padding: 100;

border: 100;

}



\* {

margin: 0;

border: 0;

}



\* {

margin: 0;

padding: 0;

}

[Discutir en el Foro](https://app.aluracursos.com/forum/curso-html-css-clases-posicionamiento-flexbox/exercicio-reset-css/87445/novo)

# 10 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87416/next)

Lo que aprendimos en esta aula:

* Usar clases en HTML y CSS;
* Buenas prácticas en la creación de clases;
* Posicionamiento de elementos con CSS;
* Importancia de resetear el estándar de CSS;
* Probar los elementos de la página inspeccionando a través de la herramienta de desarrollo;
* Modelo de caja (margen, borde y relleno);
* Diferentes selectores CSS.

# 03 Para saber más: Viewport

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87410/next)

## ¿Qué es el Viewport?

En informática gráfica, el viewport es la porción del área visible de un plano y se utiliza como unidad de medida en CSS para crear páginas web 100% responsivas. En otras palabras, el viewport varía de dispositivo a dispositivo, por ejemplo, en computadoras, tabletas y teléfonos móviles, cada pantalla tiene dimensiones diferentes y mientras una página no responsiva presentaría elementos desproporcionados, una página responsiva utilizando viewport tendría sus elementos adecuados a cada proporción.

Si has mostrado interés en el tema y quieres profundizar más, te recomendamos que leas el artículo [Guía de Unidades en CSS](https://www.aluracursos.com/blog/guia-de-unidades-en-css). A través de él, no solo conocerás el Viewport, sino también otras unidades y conceptos muy utilizados en el día a día del desarrollo Front-end.

# 04 Posicionando una imagen con box-sizing

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87446/next)

Aprendiste en el vídeo anterior que la propiedad box-sizing es responsable de cómo se calculan el ancho y el alto totales de un elemento. Ante esto, imagina que estás creando un proyecto y te encuentras con un error: la imagen utilizada está superando el límite del elemento padre. En otras palabras, deseas que la imagen, como elemento hijo, "se encoja" y se ajuste al contenido, como elemento padre. Marca la alternativa que resuelva correctamente el problema:

Seleccione una alternativa



box-sizing: content-box;

width: 100%;



box-sizing: content-box;

width: 100%;

padding: 5px;



box-sizing: border-box;

width: 100%;

padding: 5px;

# 06 Utilizando Flexbox

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87454/next)

Ahora que has aprendido que Flexbox es una herramienta de CSS que tiene como objetivo organizar los elementos de una página HTML de forma dinámica y mantener un diseño flexible, analiza el fragmento de código HTML a continuación y marca la alternativa que utiliza correctamente el Flexbox CSS:

* Alternativa correta

.sobre-mim {

position: flex;

}

* Alternativa correta

.sobre-mim {

display: flexbox;

}

* Alternativa correta

.sobre-mim {

display: flex;

}

* ¡Correcto! El selector de clase se utilizó correctamente mediante el punto (.) antes del nombre y la propiedad display con el valor flex es la estructura estándar de Flexbox.
* Alternativa correta

sobre-mim {

display: flex;

}

¡Enhorabuena, has acertado!

# Para saber más: guía Flexbox

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87411/next)

## Guía completa de Flexbox

Durante la clase se presentó el [Complete Guide to Flexbox](https://css-tricks.com/snippets/css/a-guide-to-flexbox/). Es recomendada, ya que simplifica y cubre todo sobre el diseño CSS Flexbox, explicando las diferentes propiedades para el elemento padre y los elementos hijos. También incluye historia, demostraciones, patrones y un gráfico de compatibilidad con navegadores.

# 08 Haga lo que hicimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87425/next)

## HTML

Antes de alinear los elementos en CSS, tenemos que crear la clase en el archivo HTML. Para hacerlo, vamos a la etiqueta <main> y dentro de ella agregamos la propiedad class. Recuerda que, al igual que cualquier otra propiedad, debemos utilizar el signo de igual (=) para asignar un valor, o en este caso un nombre, el cual debe estar entre comillas dobles (“”). Sigue el código:

<main class="presentacion">

Observa que hemos definido el nombre de la clase como "presentacion" y no hemos utilizado la tilde, siguiendo así las buenas prácticas mencionadas anteriormente.

## CSS

Ahora que el código HTML está listo, vamos a alinear los elementos en CSS. Para empezar, utilizamos el selector .presentacion para seleccionar solo los elementos que están dentro de nuestra clase creada en el contenido de main. Después de abrir y cerrar las llaves, la estructura básica de CSS, vamos a utilizar solo dos líneas de código para obtener el resultado final. Sigue el código a continuación:

.presentacion {

display: flex;

align-items: center;

}

Observa ahora que hemos utilizado la propiedad display con el valor flex para poder utilizar Flexbox en nuestra estilización. Entonces, hemos escrito align-items con el valor center para alinear los elementos, centrándolos respecto a sí mismos. ¡Y listo! Tenemos nuestro código estructurado y funcional, con los elementos alineados.

# 03 Justificación de elementos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87455/next)

En el video anterior aprendiste cómo alinear elementos a través de justify-content, un comando responsable de justificar, formateando la posición según el requisito del desarrollador. Ahora imagina que recibiste un código listo, el cual, al presentar errores, no está funcionando. Además, también te enviaron imágenes de cómo está y cómo debería ser el proyecto.

Analiza el código y las imágenes a continuación y marca la alternativa que resuelva correctamente el problema.

main { display: flexbox; justify-content: center; }

Como está:

Como debería ser:

* Alternativa correta

main {

display: flex;

justify-content: space-between;

}

* ¡Correcto! Con este código, el proyecto pasa a ser como debería ser, es decir, con los cuadrados en línea, dentro del límite de main y separados por un espacio en el medio. Ambas propiedades se utilizaron dentro del estándar de Flexbox y en el contexto adecuado.
* Alternativa correta

main {

display: flex;

justify-content: center;

}

* Alternativa correta

main {

display: flexbox;

justify-content: space-between;

}

# 04 Para saber más: indentación

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87412/next)

## Una buena práctica al escribir tu código

La indentación es el espaciado aplicado al principio de las líneas de código que ayuda a mantener una jerarquía visual. En algunos lenguajes, este no es un elemento que impida que el código se compile, pero mantener el código indentado es fundamental para facilitar su lectura, especialmente para terceros.

Para ilustrar mejor, aquí hay dos ejemplos:

Código no indentado

<main>

<div class="titulos">

<h1>Título de la página</h1>

</div>

<div class="paragrafos">

<p>Primer párrafo</p>

<p>Segundo párrafo</p>

</div>

</main>

Código indentado

<main>

<div class="titulos">

<h1>Título de la página</h1>

</div>

<div class="paragrafos">

<p>Primer párrafo</p>

<p>Segundo párrafo</p>

</div>

</main>

Si te ha gustado aprender más sobre la estructura y las buenas prácticas de HTML y deseas profundizar, te recomendamos la documentación [Comenzando con HTML](https://developer.mozilla.org/es/docs/Learn/HTML/Introduction_to_HTML/Getting_started) de Mozilla Developer Network.

# 07 Personalizando el título del proyecto

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87456/next)

La personalización de textos, especialmente títulos, mediante estilos y fuentes, es una etapa importante en el desarrollo de una página web, ya que la experiencia del usuario depende en gran medida de la parte visual. Sabiendo esto, imagina que te han asignado la tarea de estilizar el título <h1> de una página, cambiando la fuente predeterminada a la importada Roboto, aumentando su tamaño de 16 píxeles a 20 píxeles y cambiando su color de negro a azul.

Analizando el código base que has recibido a continuación, marca la alternativa que indique correctamente lo que debe ser cambiado:

@import url('https://fonts.googleapis.com/css2?family=Roboto&display=swap');

h1 {

font-size: 16px;

}

* Alternativa correta

h1 {

font-size: 20vh;

font-family: 'Roboto', sans-serif;

font-color: blue;

}

* Alternativa correta

h1 {

font-size: 20px;

font-family: 'Roboto', sans-serif;

color: blue;

}

* ¡Correcto! La sintaxis tanto de las propiedades como de los valores correspondientes se escribió correctamente. La unidad de medida del tamaño de la fuente fue el píxel, como se pidió en el enunciado, se utilizó la fuente Roboto importada y se cambió el color a azul.
* Alternativa correta

h1 {

font-size: 20%;

font-family: 'Roboto', sans-serif;

font-color: blue;

}

* Alternativa correta

h1 {

font-sizing: 20px;

font-family: Roboto, sans-serif;

color: blue;

}

# 08 Para saber más: Google Fonts

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87413/next)

## Google Fonts

Google Fonts es una biblioteca de Google con más de 1000 fuentes de licencia libre. Además de un directorio web interactivo para navegar por la biblioteca, hay atajos para usar convenientemente las fuentes a través de CSS.

## Cómo importar las fuentes

Para ello, recomendamos la lectura de la [guía oficial Primeros pasos con la API Google Fonts](https://developers.google.com/fonts/docs/getting_started?hl=es-419), la cual explica paso a paso cómo funciona y cómo utilizar la biblioteca.

# 09 Haga lo que hicimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87426/next)

1 - Espaciado y márgenes con Flexbox

Aquí, el objetivo es ajustar el espacio entre la sección de texto y la imagen para crear una apariencia similar al diseño de Figma. Utiliza la propiedad justify-content: space-between; en Flexbox para espaciar los elementos y agrega márgenes para evitar que los elementos queden pegados a los bordes de la pantalla.

.presentacion {

display: flex;

align-items: center;

justify-content: space-between;

margin: 10%;

}

2 - Definiendo el tamaño de la sección de texto

En esta actividad, ajustarás el tamaño de una sección de texto para que corresponda a las especificaciones del diseño en Figma. Agregarás una nueva clase llamada presentacion\_\_contenido a la etiqueta <section> en HTML y luego definirás el ancho de esta sección en CSS usando la propiedad width. El valor específico del ancho debe ser extraído del proyecto en Figma.

3 - Ajustando tamaños de títulos y textos

El objetivo aquí es modificar el tamaño de los textos de títulos y párrafos para que correspondan al diseño en Figma. Crearás clases específicas para el título ( presentacion\_\_contenido\_\_titulo) y para el texto ( presentacion\_\_contenido\_\_texto) en HTML. Luego, en CSS, usarás la propiedad font-size para definir los tamaños de fuente para estas clases, con los valores extraídos de Figma.

4 - Importando y aplicando fuentes de Google Fonts

En esta actividad, importarás las fuentes "Krona One" y "Montserrat" de Google Fonts para tu proyecto web. Primero, debes buscar estas fuentes en Google Fonts y utilizar la opción de @import para incluirlas en el archivo CSS. Luego, aplicarás estas fuentes a los elementos de texto específicos en tu página, como títulos y párrafos, usando la propiedad font-family.

5 - Personalizando fuentes de títulos y textos

Después de importar las fuentes, personalizarás el estilo de los títulos y textos de la página para que coincidan con el diseño en Figma. Modificarás la propiedad font-family para los elementos de título y texto en CSS, aplicando las fuentes "Krona One" y "Montserrat" respectivamente. El objetivo es garantizar que el estilo visual de la página web esté alineado con el diseño propuesto.

# 03 Para saber más: la etiqueta div

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87414/next)

## Cuando usar una div

La etiqueta <div> define una división en un documento HTML y suele utilizarse como un contenedor para otros elementos, lo que ayuda en la estilización del bloque. Por este motivo, la <div> se utiliza frecuentemente cuando necesitamos agrupar elementos sin usar las etiquetas semánticas de HTML. Esto se debe a que la <div> no tiene valor semántico. Por lo tanto, no significa nada para los navegadores y motores de búsqueda.

Además, al ser muy utilizada para agrupar elementos, facilita la organización de la información en los diseños. De esta forma, puede ser formateada y manipulada orgánicamente a través de CSS. Por lo general, se acompaña de atributos de ID y clase para facilitar esta organización y formateo.

Si deseas profundizar en el tema, te recomendamos que leas la [documentación oficial](https://developer.mozilla.org/es/docs/Web/HTML/Element/div).

# 05 Editando las tags a

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87457/next)

El elemento de anclaje <a> con el atributo href crea un hipervínculo en las páginas web, a través de una URL que indica el destino del enlace. A partir de esto, imagina que has creado una lista de enlaces en tu página web y necesitas personalizarlos en CSS en dos aspectos: eliminar el subrayado y crear un fondo gris con bordes redondeados.

Marca la alternativa correcta que indique el código CSS responsable de esta personalización:

* Alternativa correta

a {

text-decoration: none;

background-color: grey;

border: 10px;

}

* Alternativa correta

a {

text-decoration: none;

background-color: grey;

border-radius: 10px;

}

* ¡Correcto! Todas las propiedades y valores utilizados están correctos en contexto y sintaxis. Mientras que el valor none es responsable de eliminar las decoraciones del elemento de anclaje, la propiedad border-radius hace que los bordes de los enlaces sean redondeados.
* Alternativa correta

a {

text-decoration: underline;

background-color: grey;

border-radius: 10px;

}

# 06 Haga lo que hicimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87427/next)

1 - Espaciado y tamaño de los botones

Aquí, el desafío es ajustar el espaciado y el tamaño de los botones. Después de haber alineado los botones horizontalmente con Flexbox, necesitas asegurarte de que tengan un tamaño adecuado y un espaciado consistente entre ellos, según el diseño del proyecto. Modifica el CSS para ajustar el tamaño de los botones y agrega margen para crear el espaciado correcto.

2 - Mejorando la visibilidad de los botones

En esta actividad, tu enfoque estará en la accesibilidad y visibilidad de los botones. Guilherme mencionó la dificultad para ver los botones. Por lo tanto, aumenta el tamaño de la fuente, cambia los colores y agrega efectos como sombra o borde para hacerlos más visibles y atractivos. Utiliza tus habilidades en CSS para mejorar la estética y accesibilidad de los botones en la página.

3 - Creando una clase común para botones

En esta actividad, crearás una clase CSS común para los botones "Instagram" y "GitHub" en tu proyecto HTML. Esta clase tendrá propiedades de estilo que harán que los botones se parezcan a los del diseño en Figma, incluyendo color de fondo, tamaño de texto y redondeo de las esquinas.

4 - Ajustando el espaciado interno de los botones

El objetivo de esta actividad es ajustar el espaciado interno de los botones para que coincidan con el diseño en Figma. Utilizarás la propiedad padding para definir el espacio entre el texto del botón y sus bordes, garantizando que el diseño sea consistente y visualmente agradable.

5 - Importando y aplicando la fuente correcta

En esta actividad, te enfocarás en ajustar la fuente de los botones. Importarás la fuente Montserrat con el peso 600 de Google Fonts y la aplicarás a los botones, asegurándote de que el texto coincida exactamente con el diseño en Figma. Además, eliminarás el subrayado predeterminado de los enlaces y ajustarás el color del texto.

# 03 Para saber más: las propriedades de Flexbox

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87415/next)

## Gap

La propiedad gap no es exclusiva de Flexbox, sin embargo, se utiliza casi siempre en conjunto con él. Esta propiedad especifica en CSS el tamaño de los espacios entre filas y columnas en diseños de cuadrícula, flex y de varias columnas. Su sintaxis es bastante simple y acepta uno o dos valores.

Para comprender más sobre esta propiedad en particular, recomendamos leer el artículo [gap](https://css-tricks.com/almanac/properties/g/gap/) en [CSS-Tricks](https://css-tricks.com/).

## Otras propiedades importantes

Si deseas aprender sobre otras propiedades importantes para Flexbox, lee nuestro artículo [Flexbox CSS: Guía Completa, Elementos y Ejemplos](https://www.aluracursos.com/blog/flexbox-css-guia-completo-elementos-y-ejemplos).

# 04 Espaciando elementos hijos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-clases-posicionamiento-flexbox/task/87458/next)

Al definir el valor de la propiedad display de un elemento como flex, creamos un contenedor flex, es decir, comenzamos a utilizar Flexbox. A partir de este punto, todos los elementos dentro del contenedor se llaman elementos hijos y mostrarán comportamientos estandarizados.

Sabiendo esto, marca la alternativa que presente correctamente dos propiedades de Flexbox que, cuando se utilizan en el elemento padre, son responsables del espaciado de los elementos hijos:

* Alternativa correta

Propiedades align-items y justify-content.

* ¡Correcto! Ambas propiedades forman parte de la estructura de Flexbox y actúan en el elemento padre con la responsabilidad de espaciar los elementos hijos.
* Alternativa correta

Propiedades justify-content y column.

* Aunque la propiedad justify-content pertenece a Flexbox y espacia los elementos hijos, column está en el contexto incorrecto, ya que es un valor y no una propiedad.
* Alternativa correta

Propiedades align-items y display.

* Alternativa correta

Propiedades space-between y flex-direction.

Aunque la propiedad flex-direction pertenece a Flexbox y espacia los elementos hijos, space-between está en el contexto incorrecto, ya que es un valor y no una propiedad.

# Centrar un elemento con CSS



Gabs Ferreira

Si desarrollas para la web es más que común querer centrar elementos en la pantalla, ¿verdad? Imaginemos que tenemos la siguiente div:

<div class="container">

<p>Aquí tenemos un texto</p>

<div>

Si quieres centrar el texto dentro de la div, puedes hacer:

.container { text-align: center; }

Este código funciona para centrar cualquier elemento que sea inline (puedes ver cuáles son [inline](https://developer.mozilla.org/en/docs/Web/HTML/Inline_elemente)) o inline-block.

Ah, pero ¿y si no quiero centrar el contenido, sino la div en la pantalla? La forma en que lo hicimos no funcionará. En este caso, básicamente tenemos que hacer dos cosas:

Dar un tamaño a la div y establecer los márgenes izquierdo y derecho en auto:

.container {

width: 700px;

margin-left: auto;

margin-right: auto;

}

Este código funciona para cualquier elemento [**block**](https://developer.mozilla.org/en/docs/Web/HTML/Block-level_elements).

Si te interesa este tema, aquí en [Alura](https://www.aluracursos.com/) tenemos capacitaciones de front-end, para principiantes y para aquellos que ya son desarrolladores web. En ellas, verás cómo programar en Javascript, HTML y CSS para construir sitios web.

# Curso de **HTML y CSS: header, footer y variables CSS**

# 05 Alineamiento con display flex

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87658/next)

Aprendemos que podemos utilizar el display: flex para posicionar elementos en la pantalla, observando su comportamiento predeterminado. Dicho esto, observa el siguiente código HTML, en el cual el elemento padre container contiene cuatro divs y cada uno será un cuadrado en la pantalla:

<body>

<div class="container">

<div class="cuadrado"></div>

<div class="cuadrado"></div>

<div class="cuadrado"></div>

<div class="cuadrado"></div>

</div>

</body>

Ya en CSS, el elemento padre fue estilizado de la siguiente manera:

.container {

margin: 20%; /\* agrega un margen para centrar los cuadrados en la pantalla \*/

display: flex; /\* coloca los cuadrados uno al lado del otro \*/

gap: 10px; /\* agrega espacios iguales entre cada cuadrado \*/

}

Estos códigos muestran el siguiente resultado en la pantalla:

Imagina que deseas alinear estos cuadrados verticalmente, uno debajo del otro, sin embargo, incluso al agregar el display: flex aún no has obtenido ese resultado.

¿Cuál de las siguientes alternativas representa la solución para organizar estos elementos como deseas?

* Alternativa correta

Podemos agregar la propiedad flex-direction: column al CSS de la clase .container.

* Al agregar la propiedad flex-direction: column, los cuadrados se posicionan verticalmente, ya que define que la dirección del display: flex debe ser en "columna".
* Alternativa correta

Podemos agregar la propiedad justify-content: center al CSS de la clase .container.

* Alternativa correta

Podemos agregar la propiedad align-items: flex-start al CSS de la clase .container.

* Alternativa correta

Podemos agregar la propiedad flex-direction: row al CSS de la clase .container.

La propiedad flex-direction define la dirección en la que los elementos del display: flex deben ser dispuestos y ya tiene row como valor predeterminado, por lo que agregar flex-direction: row no producirá ningún cambio.

# 07 Para saber más: border y comentarios en el código

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87685/next)

# **Border**

Durante la clase, personalizamos la propiedad CSS llamada border para añadir el borde alrededor de los botones. Esta propiedad se puede utilizar para definir los valores de ancho del borde, estilo del borde y color del borde, y tiene algunos estilos diferentes, como "dotted", que conocimos en la clase a través de la [documentación](https://www.w3schools.com/css/css_border.asp).

# **¿Qué son los comentarios?**

Los comentarios son notas de texto que se pueden insertar a lo largo del desarrollo y que serán ignoradas por el código, por lo tanto, no interfieren en su funcionamiento.

¿Para qué sirven los comentarios?

Los comentarios pueden ser utilizados para anotar información importante sobre el código, para guardar códigos que estemos probando sin necesidad de borrar y reescribir, para describir alguna función específica y facilitar mantenimientos futuros o incluso para ayudar en la organización de las ideas de construcción del código.

Cómo comentar en HTML y CSS

El uso de comentarios es común en diversos lenguajes y cada uno tendrá su sintaxis particular. Esto también ocurre en HTML y CSS, ya que en cada uno tenemos una manera de agregar comentarios, que vamos a conocer a continuación:

HTML

Los comentarios en HTML se asemejan a una apertura de etiqueta. Abrimos un comentario con . Todo lo que esté entre estos dos marcadores será considerado como comentario, incluyendo saltos de línea:

<p> Esto es un párrafo de texto y será mostrado por el navegador </p>

<!-- Esto es un comentario y será ignorado -->

<!-- Esto es un comentario con salto de línea

y también será ignorado -->

<p> Esto es un párrafo de texto y será mostrado por el navegador </p>

CSS

Los comentarios en CSS también tienen apertura y cierre. Abrimos este tipo de comentario con /\* y cerramos con \*/ y al igual que en HTML, todo lo que esté entre estas marcas será considerado comentario, aunque esté entre más de una línea:

body{

background-color: white;

/\* Esto es un comentario en CSS \*/

/\*

Esto es un comentario en CSS con

salto de línea

\*/

}

Buenas prácticas al comentar

Los comentarios pueden ser muy útiles para facilitar el desarrollo, sin embargo, debemos prestar atención a algunas buenas prácticas para hacer un uso adecuado de ellos:

* Evita agregar muchos comentarios para no generar "desorden" en tu código;
* Evita crear comentarios muy largos. Ten en cuenta que aunque no se ejecuten, estos comentarios ocupan espacio y pueden hacer que tu archivo sea más pesado;
* Utiliza los comentarios a tu favor, es decir, anota información esencial en ellos, ya que esto facilitará que encuentres lo que realmente necesitas recordar al realizar mantenimientos en tu código;
* Al usar comentarios para probar tu aplicación, recuerda eliminar esas notas cuando termines, para que tu código no quede desorganizado y con partes innecesarias;
* Por último, recuerda siempre que un código descriptivo es más duradero que muchos comentarios. Cuando estés trabajando con HTML, recuerda usar las etiquetas semánticas y al crear tus clases CSS, añade nombres significativos. Así, tu código será naturalmente descriptivo y no dependerá tanto de los comentarios, que pueden ser eliminados o perderse durante los mantenimientos =) .

# 08 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87666/next)

1 - Agregando un subtítulo a la página

Imagina que estás desarrollando una página web para compartir tus redes sociales. Necesitas agregar un subtítulo "Accede a mis redes:" encima de los enlaces para Instagram y Github. Tu tarea es editar el archivo index.html para incluir este subtítulo. Recuerda usar la etiqueta <h2> para el subtítulo y colocarlo dentro de la <div> que ya contiene los enlaces.

2 - Cambiando la disposición de los elementos con Flexbox

Ahora que se ha añadido el subtítulo, te das cuenta de que está alineado a la izquierda de los botones en lugar de encima de ellos. Esto sucedió porque los elementos están dispuestos horizontalmente. Tu misión es cambiar la disposición de los elementos (subtítulo y botones) a una disposición vertical. Edita el archivo style.css, modificando la propiedad flex-direction a column dentro de la clase .presentacion\_\_enlaces. Esto cambiará la dirección del Flexbox, alineando los elementos verticalmente.

3 - Ajustando la alineación y el espaciado de los elementos

Después de modificar la dirección del Flexbox, observas que los elementos (texto y botones) están muy cerca unos de otros y alineados a la izquierda. Para mejorar la estética de la página, necesitas centrar verticalmente estos elementos y agregar un espacio entre ellos. En el archivo style.css, dentro de la clase .presentacion\_\_enlaces, añade la propiedad align-items: center para centrar los elementos. Luego, agrega la propiedad gap con un valor de 32px para añadir el espaciado entre cada elemento.

4 - Creando una clase para el subtítulo

Estás desarrollando una página web y has decidido que el subtítulo "Accede a mis redes:" necesita una estilización específica. Para eso, vas a crear una clase CSS para este subtítulo. En el archivo index.html, encuentra la etiqueta <h2> que contiene el subtítulo y añade una clase llamada presentacion\_\_enlaces\_\_subtitulo. Tu tarea es insertar esta clase en la etiqueta <h2> de forma correcta.

5 - Estilizando el subtítulo

Ahora que has creado una clase para el subtítulo, es hora de estilizarlo. En el archivo style.css, añade la clase .presentacion\_\_enlaces\_\_subtitulo y configura las propiedades de estilo. Utiliza la fuente 'Krona One', con un peso de fuente (font-weight) de 400 y un tamaño de fuente (font-size) de 24px. Ajusta la fuente para que sea 'sans-serif' sin comillas. Tu tarea es escribir el código CSS para aplicar estas configuraciones al subtítulo.

6 - Modificando el estilo de los botones

Los botones de tu página necesitan un nuevo aspecto. En el archivo style.css, encuentra la clase .presentacion\_\_enlaces\_\_link y realiza los siguientes cambios: elimina el fondo cian, cambia el color del texto a blanco (#F6F6F6), añade un borde sólido de 2px en color cian (#22D4FD), cambia el ancho a 378px y ajusta el border-radius a 8px. Tu tarea es cambiar el código CSS para reflejar estos cambios, creando botones con bordes cian, texto blanco y esquinas menos redondeadas.

# 04 Buscando archivos en las carpetas

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87659/next)

Aprendemos que una práctica común en el desarrollo con HTML y CSS es crear una carpeta de assets para insertar todas las imágenes del proyecto. Sin embargo, al buscar alguna imagen a través de la etiqueta img en HTML, necesitamos prestar atención al camino de ese archivo que está dentro de las carpetas. Con eso en mente, observe la organización de las carpetas del proyecto a continuación:

Necesitas que el archivo logo.png, ubicado dentro de la carpeta de assets del proyecto, se muestre en el HTML y para eso creas el siguiente código:

<div class="container">

<img src="logo.png" alt="logo del proyecto">

</div>

Dado el código, señale la afirmación correcta sobre lo que ocurrirá con la imagen en la visualización de la página:

* Alternativa correta

La imagen no se mostrará, ya que la ruta en src debería ser /..logo.png.

* Alternativa correta

La imagen no se mostrará, ya que la ruta en src debería ser ../logo.png.

* Alternativa correta

La imagen se romperá y solo se mostrará el texto alternativo, ya que el enlace de la imagen en src está incorrecto.

* El HTML no encontrará la imagen, ya que el enlace en src debería considerar que la imagen está dentro de una carpeta. Por lo tanto, el enlace correcto debería ser ./assets/logo.png. Siempre debemos estar atentos a la organización de las carpetas cuando queremos buscar un archivo a través de su ruta.
* Alternativa correta

La imagen se mostrará normalmente, ya que su ruta se agregó correctamente y no hubo errores de sintaxis al escribir el elemento src.

# 06 Posicionando elementos con Flexbox

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87660/next)

Creas un proyecto HTML con una imagen y un párrafo de texto, los cuales están dentro de una sección, como podemos observar en el código siguiente:

<section>

<img src="logo-alura-latam.png" alt="Logo da Alura Latam">

<p>Estoy estudiando HTML y CSS en Alura Latam!</p>

</section>

Luego, creas un archivo CSS para esta página, donde estilizas algunos detalles según tu preferencia y agregas un display: flex para posicionar los elementos uno al lado del otro, de la siguiente manera:

section{

border-style: solid; /\* Crea un borde sólido \*/

border-color:red; /\* Colorea el borde con el color rojo\*/

margin: 10%; /\* Agrega un margen alrededor de la section \*/

display: flex; /\* Agrega el display flex \*/

}

Con el display: flex, los elementos que antes estaban uno debajo del otro como en esta imagen:

Se posicionan lado a lado en la pantalla, como se muestra en la imagen a continuación:

Observa que incluso al aplicar display: flex a su elemento padre (section), todos los elementos hijos (imagen y párrafo) siguen desplazados hacia la esquina izquierda de la sección.

¿Qué podemos hacer para que la imagen y el párrafo se desplacen al centro dentro de esta sección?

* Alternativa correta

Podemos agregar la propiedad justify-content: space-evenly al CSS de la sección.

* La propiedad justify-content controla la posición de los elementos hijos a lo largo del eje principal y también ayuda a distribuir el espacio libre entre ellos. Con el valor space-evenly, los elementos se distribuyen con espacios iguales entre sí y hasta los bordes. Como solo tenemos 2 elementos, quedarán centrados.
* Alternativa correta

Podemos agregar la propiedad justify-content: space-around al CSS de la sección.

* La propiedad justify-content controla la posición de los elementos hijos a lo largo del eje principal y también ayuda a distribuir el espacio libre entre ellos. Con el valor space-around, los elementos se distribuirán con un espacio alrededor de cada uno.
* Alternativa correta

Podemos agregar la propiedad justify-content: center al CSS de la sección.

La propiedad justify-content controla la posición de los elementos hijos a lo largo del eje principal y también ayuda a distribuir el espacio libre entre ellos. Con el valor center, los elementos se centrarán en su elemento padre, pero sin espacios entre sí.

# 08 Para saber más: Hover

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87687/next)

El hover se utiliza para seleccionar elementos cuando pasas el ratón sobre ellos. Esta y otras informaciones útiles sobre el [hover](https://www.w3schools.com/CSSref/sel_hover.php) las puedes encontrar en la documentación =)

# 09 Desafío: personalizando el proyecto con hover

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87664/next)

¡Ahora que ya conoces el selector hover, tenemos un desafío para ti!

Elige algún elemento en tu portafolio, puede ser tu nombre, el texto sobre ti, el enlace de tus redes sociales o algún elemento nuevo que desees agregar.

En ese elemento, debes hacer alguna personalización utilizando el hover. Aquí puedes explorar colores, tamaños, imágenes y ver cómo cada elemento reacciona al pasar el cursor sobre él.

Recuerda que para usar el hover, simplemente debes llamar en tu CSS al selector de la etiqueta que deseas personalizar, ya sea tu nombre o clase, seguido de :hover. Por ejemplo, si quisieras personalizar un div, llamarías:

div:hover {

/\* inserta aquí tu personalización \*/

}

La idea aquí es darle tu toque personal al proyecto, ¡así que sé creativo en la personalización!

# 04 Manejando las tags semánticas

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87661/next)

Trabajamos con las etiquetas semánticas <header> y <footer> en la organización de nuestro proyecto.

Sobre estas etiquetas, selecciona las alternativas correctas:

* Alternativa correta

La etiqueta <footer> representa el pie de página y es común que contenga información sobre el autor de la página, derechos de autor, direcciones o enlaces a otras páginas relacionadas.

* La etiqueta <footer> será un área informativa sobre la página y sigue el mismo principio que <header>, ya que no debe crearse dentro de otra etiqueta <footer> o dentro de una etiqueta <header>.
* Alternativa correta

La etiqueta <header> representa el encabezado de la página, que tiene como función principal optimizar la navegación del usuario y contendrá elementos que favorezcan este proceso.

* La etiqueta <header> ayuda a que el usuario se oriente en la página y acepta elementos como la barra de navegación, el logotipo de la página, información de autoría e incluso formularios de búsqueda, aunque no es común usarlos en el <header>.
* Alternativa correta

La etiqueta header puede contener títulos de <h1> a <h6>, entre otros elementos comunes al cuerpo del HTML, como la etiqueta <p>.

* Alternativa correta

Al crear un enlace en la página, debe ser añadido dentro de una etiqueta <nav> para que esté correctamente dispuesto en la pantalla.

# 05 Para saber más: Padding

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87689/next)

Observe la siguiente imagen:

La pantalla de arriba muestra los botones que creamos en nuestro proyecto para acceder a las redes sociales. Sin embargo, podemos notar que el ícono y el texto dentro de cada botón están "apretados" por los bordes, lo que desfavorece la apariencia del proyecto.

Todos los botones tienen contenido: un ícono y el texto con el nombre de la red social. Además, todos los botones tienen un borde delgado en azul. Por lo tanto, necesitamos agregar alguna propiedad que cree un espacio entre el contenido y los bordes del botón, para obtener el siguiente resultado:

¡Mucho mejor, ¿verdad? Pero ¿sabes qué propiedad nos ayudó a definir la distancia entre el contenido y el borde?

Si pensaste en padding, ¡acertaste!

El padding es responsable de definir la distancia entre un contenido y sus bordes. En el caso de nuestro proyecto, añadimos un padding: 21.5px 0, que fue suficiente para crear un espacio interno más amplio en los botones. Esta propiedad es extremadamente útil porque garantiza que exista este espacio de "respiración" entre los bordes y el contenido, lo que hace que la apariencia sea más limpia y armoniosa.

Para comprender mejor cómo funciona el padding, puedes consultar la documentación.

# 05 Creando la navegación entre páginas

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87662/next)

Imagina que deseas crear una navegación entre las páginas de Inicio y Sobre mí en tu <header>. ¿Cuál de los siguientes enfoques se recomienda para desarrollar esa navegación?

* Alternativa correta

<header>

<nav>

<a src="index.html">Inicio</a>

<a src="about.html">Sobre mí</a>

</nav>

</header>

* Alternativa correta

<header>

<nav>

<a href="index.html">Inicio</a>

<a href="about.html">Sobre mí</a>

</nav>

</header>

* Al agregar enlaces de navegación al <header>, debemos crear una <nav>, con un <a> para cada enlace, y agregar su dirección en la propiedad href.
* Alternativa correta

<header>

<nav>

<link href="index.html">Inicio</a>

<link href="about.html">Sobre mí</a>

</nav>

</header>

* Alternativa correta

<header>

<a href="index.html">Inicio</a>

<a href="about.html">Sobre mí</a>

</header>

¡Enhorabuena, has acertado!

# 06 Desafío: crea una página curriculum

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87665/next)

Ahora que has aprendido a crear otras páginas con HTML, ¡queremos desafiarte!

Crea otra página en tu proyecto y construye en ella tu currículum.

Puedes seguir la estructura de los currículums tradicionales, con tus datos en la parte superior y tu información dividida en secciones, o puedes dejar volar tu creatividad y hacer algo completamente diferente, según tu criterio.

Es importante que en esta página de currículum añadas una imagen tuya y crees al menos una lista HTML que contenga tus experiencias laborales o educativas.

Además, podría ser interesante añadir otros proyectos en caso de que los tengas.

Recuerda: no tengas miedo de personalizar esta página, después de todo, es tu currículum y cuanto más refleje "tu esencia", mejor.

¡Ahora es tu turno de crear, ¿te animas al desafío?

# 07 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87669/next)

1 - Estilizando el encabezado con CSS

Imagina que trabajas en el desarrollo front-end en un proyecto de sitio web. Tu desafío es estilizar el encabezado de la página de acuerdo con las especificaciones proporcionadas. Utiliza el archivo style.css para aplicar las propiedades CSS en el encabezado.

2 - Ajustando el espaciado del contenido

Ahora, como desarrollador, tu próximo desafío es ajustar el espaciado del contenido de la página para que se alinee correctamente con el encabezado.

3 - Creando y navegando a la página "Sobre mí"

Estás trabajando en un proyecto de sitio web personal y necesitas agregar una página "Sobre mí". Para ello, debes crear un nuevo archivo HTML llamado about.html y configurar un enlace en el menú de navegación de tu sitio principal (index.html) para redirigir a esta nueva página. Además, ajusta el encabezado de la página about.html para que el título de la pestaña sea "Sobre mí" y agrega un <h1> con el texto "Sobre mí" en el cuerpo de la nueva página. Asegúrate de que la navegación entre la página principal y la página "Sobre mí" funcione correctamente.

4 - Ajustando el estilo después de reorganizar archivos

Has notado que, después de mover el archivo styles.css a una nueva carpeta llamada "styles", se perdió el estilo de tu sitio. Para corregir esto, debes actualizar la ruta del archivo CSS en tu archivo index.html. Además, en el archivo styles.css, debes eliminar la subrayado de los enlaces en el encabezado del sitio. Después de estas correcciones, verifica si el estilo se aplicó correctamente y si los enlaces del encabezado ya no están subrayados.

5 - Estructurando la página "Sobre mí" con encabezado y pie de página

En esta actividad, estructurarás la página "Sobre mí" de tu proyecto, reutilizando el encabezado y el pie de página de la página inicial (index.html). Primero, copia el <header> y el <footer> de index.html y pégalos en el archivo about.html. Luego, agrega una sección <main> vacía entre el <header> y el <footer>. Por último, verifica si la estructura copiada funciona correctamente en el navegador, con los enlaces de navegación y el texto del pie de página mostrándose.

6 - Importando y corrigiendo la ruta del archivo CSS en la página "Sobre mí"

El objetivo de esta actividad es importar el archivo de estilos styles.css en la página "Sobre mí" y corregir la ruta del archivo para reflejar la estructura de carpetas actual. En el archivo about.html, agrega la etiqueta <link> para importar styles.css. Recuerda que el archivo CSS se movió a una carpeta llamada "styles", por lo que la ruta de href en la etiqueta <link> debe actualizarse a "./styles/styles.css". Después de esta corrección, guarda el archivo y recarga la página en el navegador para verificar si los estilos se aplicaron correctamente.

# 04 Trabajando con las variables CSS

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87663/next)

Las propiedades de personalización CSS, más conocidas como variables CSS, pueden ser muy útiles para hacer que las actualizaciones de diseño del proyecto sean más prácticas. Pensando en esto, imagina que has creado una variable para almacenar el color que deseas aplicar en el encabezado de tu proyecto, como se puede observar en el siguiente código:

:root{

--color-actualizada-del-proyecto: #eb94bd;

}

.header{

background-color: black;

}

Sin embargo, incluso creando la variable de color actualizada, el encabezado no experimenta ningún cambio.

Observando el código anterior, señala las alternativas que presentan lo que necesitas hacer para poder cambiar realmente el color del encabezado con la nueva variable:

* Alternativa correta

La variable necesita ser declarada con un guión doble (--) al principio, para que sea reconocida como variable y no como propiedad CSS.

* La variable debería ser declarada con el nombre color-actualizada-del-proyecto, ya que esa es la sintaxis correcta para la creación de variables en CSS. El guión doble (--) al principio permite que el código entienda que esta no es una propiedad CSS como background-color, por ejemplo, sino un valor que estás definiendo mediante el uso de una variable.
* Alternativa correta

Para que el color se cambie, la variable no solo necesita ser declarada dentro de :root, también necesita ser aplicada como background-color de .header.

* Para usar las variables, no solo necesitamos crearlas, sino también aplicarlas en nuestro código. En este caso, dentro de .cabecalho, la propiedad background-color debería recibir como valor var(color-actualizada-del-proyecto). Recuerda siempre aplicar tus variables.
* Alternativa correta

La variable debería ser inicializada dentro del selector .header y no dentro de :root.

* Alternativa correta

La variable debería ser declarada con el nombre --color-primario en lugar de color-actualizada-del-proyecto, para que sea reconocida como color primario del proyecto.

# 05 Para saber más: las variables en el mundo de la programación

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87690/next)

¡Un armario lleno de cajones!

Imagina que trabajas en una sala de archivos que tiene un armario muy grande y lleno de cajones. Todos los días, las personas traen sus objetos para que los guardes en un cajón para ellos y, para ello, te entregan una etiqueta con un nombre que se pegará en ese cajón que almacenará el objeto de la persona.

Ana te entregó un bolígrafo y una etiqueta con el nombre: bolígrafoDeAna, y guardaste su bolígrafo en un cajón, donde pegaste la etiqueta. Ella eligió el nombre bolígrafoDeAna, pero podría haber sido cualquier otro nombre y su contenido también podría haber sido cualquier cosa, como un libro, por ejemplo, y no un bolígrafo.

Cuando Ana necesite el bolígrafo, te llamará y pedirá el bolígrafoDeAna, y tú le entregarás el contenido del cajón, es decir, el bolígrafo.

¿Y cómo se relaciona esto con las variables?

Tu armario de cajones en el ejemplo anterior representa la memoria de la computadora. Cuando creamos una variable, estamos solicitando a la computadora que reserve un "cajoncito" en su memoria para guardar algo que necesitaremos usar en el futuro, y lo hacemos asignando un nombre de variable que podremos llamar en cualquier momento y que devolverá el contenido que guardamos dentro de ella. Este nombre puede ser cualquier nombre, sin embargo, cada vez que se solicita, devolverá como respuesta lo que hayas almacenado en él.

¿Qué son las variables?

Las variables son elementos que permiten manipular valores durante la ejecución de tu código, mediante la definición de un nombre para almacenar un valor que se utilizará repetidamente. Esta definición del nombre y del contenido que se contendrá en él es lo que llamamos declaración.

Este valor puede cambiar a lo largo del código, por eso el nombre "variable".

Observa el siguiente ejemplo:

:root{

--tamano-de-la-fuente: 24px;

}

Hemos creado en :root, es decir, en el alcance global de un código, una variable que se declaró con el nombre --tamano-de-la-fuente y se le asignó un valor de 24px. Cada vez que llamemos por el nombre --tamano-de-la-fuente, obtendremos como resultado el valor 24px.

Las variables son utilizadas diariamente por las personas desarrolladoras para poder manipular y reutilizar valores en su código y están presentes en los más diversos lenguajes de programación, ya que son elementos fundamentales para crear cualquier código que tenga la mínima funcionalidad. Por lo tanto, a medida que avances en tus conocimientos en desarrollo, es seguro que trabajarás mucho con variables.

Para obtener más información sobre las variables en CSS, puedes consultar la [documentación](https://developer.mozilla.org/es/docs/Web/CSS/Using_CSS_custom_properties).

# 06 Haga lo que hicimos en aula

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87670/next)

En esta clase conocimos las variables CSS y descubrimos su potencial para la personalización del proyecto.

Ahora, nos gustaría invitarte a hacer lo mismo que se hizo en clase y buscar otras paletas de colores que puedan aplicarse en el proyecto a través de las variables CSS que creamos.

Los colores tienen un gran impacto y al cambiarlos puedes obtener un proyecto con un aspecto totalmente nuevo. Si deseas buscar paletas como se hizo en clase, puedes acceder a [Color Hunt](https://colorhunt.co/). Otra opción interesante es la [Rueda de Colores de Adobe](https://color.adobe.com/pt/create/color-wheel).

Un atajo que puede facilitar mucho el proceso de cambiar los colores en todo el código para asignar en su lugar la variable es ctrl + D. Simplemente selecciona lo que deseas cambiar con el mouse y luego haz clic en ctrl + D hasta que todos los valores que deseas cambiar estén seleccionados. Esto te permitirá cambiar simultáneamente todos los valores siguientes que sean iguales a lo que seleccionaste con el mouse, como podemos ver en el gif a continuación:

gif de código CSS que muestra el uso del atajo ctrl + D. El puntero del mouse selecciona un valor de color #000000 y al presionar el atajo ctrl + D, el mismo valor queda seleccionado en la línea siguiente y se editan simultáneamente a var(--nuevo-color).

También puedes consultar otras opciones de sitios para elegir tu paleta de colores a través de [Para saber más: eligiendo los colores de tu proyecto](https://app.aluracursos.com/course/html-css-desarrollo-estructura-archivos-tags/task/87289), que está en el curso 1 de HTML y CSS.

Además, también puedes cambiar otros elementos mediante la creación de variables, como el tamaño o el peso de las fuentes, por ejemplo. Recuerda que explorar las nuevas herramientas que conocemos en clase es un paso que enriquece mucho el aprendizaje =)

# 07 Para saber más: proyectos que puedes desarrollar para seguir evolucionando

[Próxima Actividad](https://app.aluracursos.com/course/html-css-header-footer-variables-css/task/87691/next)

Para seguir practicando HTML y CSS, es importante que continúes desarrollando nuevos proyectos para mejorar tus conocimientos. Te hemos traído algunas sugerencias geniales:

Página de regalo

¡Imagina usar tus conocimientos en HTML y CSS para regalar algo a alguien!

Puedes crear una página homenajeando a alguien que te importe y regalársela a esa persona.

Es posible hacer una galería con tus fotos, escribir un texto muy bonito y agregar enlaces a cosas que le gusten a la persona.

Estas páginas también se hacen frecuentemente como regalo de cumpleaños.

Puedes añadir recursos como vídeos, imágenes e incluso música al proyecto.

¡De esta manera, mejorarás tus conocimientos y harás feliz a alguien!

Tiendita

¿Qué tal soltar la creatividad e inventar una tiendita de algo que te guste?

Puedes crear una página para mostrar productos de una marca ficticia que te parezca interesante y así practicar display: flex componiendo la visualización de tus productos en la página, por ejemplo.

Página de favoritos

Por último, uno de los proyectos favoritos de los estudiantes de HTML y CSS: ¡Crear una página de favoritos!

Este proyecto es muy divertido porque te permite crear una página con las cosas que más te gustan. Pueden ser películas, series, música, anime, juegos, productos, entre otros.

Aquí puedes elegir un solo tema o crear secciones para cada tema favorito.

Agrega imágenes, enlaces para ver tus favoritos, vídeos sobre ellos, etc.

Puedes ser muy creativo en este proyecto: si eliges música, por ejemplo, crea enlaces para navegar a otras páginas contando la historia de la canción que elegiste, o quién la compuso, o incluso por qué es una favorita.

# Curso de **HTML y CSS: trabajando con responsividad y publicación de proyectos**

# 02 Preparando el ambiente

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87682/next)

Proyecto base:

El proyecto inicial de este curso fue implementado en el curso anterior. Entonces, si tomaste el curso anterior, puedes continuar con el mismo proyecto. De lo contrario, puedes descargar el [proyecto anterior aquí](https://github.com/alura-es-cursos/2080-html-css-header-footer-variables/archive/refs/heads/aula5.zip) o acceder a los [archivos directamente en Github](https://github.com/alura-es-cursos/2080-html-css-header-footer-variables/tree/aula5).

Figma del curso:

En este curso, tomaremos como base el prototipo que está en Figma. Para acceder a él, haz [clic aquí](https://www.figma.com/file/EAQgNcBE5qOCn5qFFYZzNZ/Portafolio---Curso-3?type=design&node-id=0%3A1&mode=design&t=GAxeOiYeBNLtTy4S-1).

Si no tienes una cuenta, créala o, si ya estás registrado, inicia sesión en una cuenta existente. Para ello, ve al sitio web de Figma y haz clic en "Iniciar sesión" (si ya tienes cuenta) o en "Registrarse" (para crear una nueva cuenta).

# 05 Unidad de medida REM

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87713/next)

Hemos realizado una mejora en nuestro código que consistió en cambiar todos los tamaños de fuente que estaban en píxeles a la unidad de medida REM.

Recordando que aún no se ha establecido un tamaño de fuente predeterminado en la raíz del proyecto, ¿qué 2 alternativas siguientes contemplan adecuadamente esta conversión de píxel a REM?

* Alternativa correta

En píxeles font-size: 40px, en REM 2.5rem.

* Como no se ha establecido un tamaño de fuente predeterminado en la raíz del proyecto, un REM por defecto del navegador equivale a 16px, por lo tanto, 40px dividido por 16px del REM es igual a 2.5rem.
* Alternativa correta

En píxeles font-size: 24px, en REM 1.5rem.

* Como no se ha definido un tamaño de fuente predeterminado en la raíz del proyecto, un REM por defecto del navegador equivale a 16 píxeles. Por lo tanto, 24px dividido por 16px del REM es igual a 1.5rem.
* Alternativa correta

En píxeles font-size: 10px, en REM 1rem.

* Alternativa correta

En píxeles font-size: 8px, en REM 1rem.

# 06 Para saber más: la importancia de utilizar unidades de medida relativas

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87683/next)

Es muy importante utilizar unidades de medida relativas en los textos y títulos, e incluso en los elementos de nuestro sitio web, como por ejemplo, las imágenes.

Estas medidas ayudan a las personas con baja visión que necesitan cambiar la apariencia del navegador o aplicar más zoom para aumentar el tamaño de los elementos.

Recuerda que también podemos reducir el zoom de la página cuando queramos tener una visión más amplia de los elementos. ¡Muy útil, ¿verdad?! :)

Este artículo de Alura sobre unidades de medida puede ser de gran ayuda para tu aprendizaje: [Artículo sobre unidades de medida](https://www.aluracursos.com/blog/guia-de-unidades-en-css).

También puedes consultar la documentación sobre las [unidades de medida](https://www.w3schools.com/cssref/css_units.php) que se utilizaron en este curso.

# 03 Adaptando imágenes

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87714/next)

Imagina que estás desarrollando un sitio web personal para mostrar tu portafolio de fotografía. Quieres que las imágenes se muestren de manera elegante en cualquier dispositivo, ocupando siempre la mitad del ancho de la pantalla para no sobrecargar visualmente al visitante, pero sin perder calidad. La elección de la unidad de medida es crucial para alcanzar este objetivo de responsividad.

¿Cuál es la unidad de medida y el valor más adecuados para definir el ancho de una imagen de modo que ocupe, de manera adaptable, la mitad del tamaño de la pantalla?

* Alternativa correta

width: 50 %;

* La unidad de medida porcentual (%) se adapta al tamaño del elemento padre, en este caso, la pantalla. Establecer width: 50%; hace que la imagen ocupe la mitad del ancho de la pantalla, independientemente del tamaño del dispositivo, cumpliendo con el objetivo de responsividad.
* Alternativa correta

width: 50 em;

* Alternativa correta

width: 50 px;

# 04 Para saber más: Dev Tools - La mejor amiga de quien desarrolla

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87684/next)

La herramienta de desarrollo (Dev Tool) es muy útil en el proceso de creación y desarrollo de proyectos web. Además de inspeccionar nuestros códigos HTML y CSS, podemos realizar algunos cambios temporales y nos muestra una vista previa en diferentes tamaños de pantalla y dispositivos, como tabletas y teléfonos móviles, lo cual es de gran ayuda al crear la versión móvil y toda la parte responsiva.

¿Te gusta? :) ¡Ahora tú también puedes intentarlo!

Para abrir las Dev Tools, cuando tengas el navegador abierto, presiona las teclas CTRL + SHIFT + C en Chrome para Windows, o COMMAND + OPTION + C en Mac, o puedes hacer clic con el botón derecho del mouse en la pantalla del navegador y seleccionar "inspeccionar" en el menú desplegable. Mira:

Con las Dev Tools abiertas, puedes:

* Seleccionar el tipo de dispositivo (móvil, tableta, etc.);
* Inspeccionar los elementos HTML;
* Cambiar el tamaño de la pantalla;
* Cambiar temporalmente los estilos CSS de la página;
* Presionar las teclas CTRL + SHIFT + M para abrir la vista responsiva.

# 06 Haga lo que hicimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87709/next)

1) Adaptando imágenes para diferentes tamaños de pantalla

Estás trabajando en un sitio web y necesitas asegurarte de que las imágenes se adapten correctamente a diferentes tamaños de pantalla. Tu tarea es añadir una clase "presentacion\_\_imagen" a las imágenes relevantes en el HTML y definir el ancho de esta clase como 50% en el CSS. Esto hará que las imágenes ocupen el 50% del ancho del elemento padre. Asegúrate de probar en diferentes anchos de pantalla, como 720px, para verificar la responsividad.

2) Ajustando el diseño para diferentes dispositivos

El diseño de tu sitio web necesita ser ajustado para diferentes dispositivos. Utiliza la herramienta "Inspeccionar" del navegador para simular diferentes tamaños de pantalla, como un portátil. Ajusta el diseño de la página para que se adapte de manera armoniosa en pantallas más pequeñas. Esto puede implicar cambiar márgenes, rellenos e incluso reorganizar elementos en la página.

3) Responsividad del contenido de texto

El contenido de texto de tu sitio web no se está adaptando bien a pantallas más pequeñas. Necesitas ajustar la clase ".presentacion\_\_contenido" en el CSS para utilizar anchos relativos en lugar de fijos. Intenta definir el ancho como 50% y observa cómo el texto se adapta cuando cambias el tamaño de la pantalla. Recuerda que la responsividad es clave para una buena experiencia del usuario en dispositivos diversos.

4) Mejorando la responsividad de los enlaces de navegación

Los enlaces de navegación de tu sitio web no están funcionando adecuadamente en diferentes tamaños de pantalla. Tu tarea es ajustar la clase ".presentacion\_\_enlaces\_\_link" en el CSS. Cambia el ancho a un valor relativo, como 50%, y prueba en diferentes tamaños de pantalla. Observa cómo responden los enlaces a estos cambios y ajusta según sea necesario para garantizar una navegación fluida y accesible.

# 03 Media Queries

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87715/next)

Imagina que estás trabajando como desarrollador(a) front-end en un proyecto innovador para una startup de tecnología. El proyecto es una aplicación web responsive que promete revolucionar la manera en que las personas interactúan con el contenido en línea. El diseño de la aplicación es crucial, ya que debe ser accesible y agradable en una amplia gama de dispositivos, desde smartphones hasta escritorios. Durante una reunión de equipo, recibes la tarea de ajustar los estilos de la aplicación para garantizar que se adapte perfectamente a pantallas con una anchura de hasta 700 píxeles. Para ello, necesitas configurar correctamente los puntos de quiebre (Break Points) utilizando Media Queries en CSS.

¿Cuál es el parámetro esperado por @media() para definir un tamaño de pantalla de hasta "700px" de ancho?

* Alternativa correta

@media (width <= 700px) { ... }

* Alternativa correta

@media (min-width: 700px) { ... }

* Alternativa correta

@media (max-width: 700px) { ... }

Esta es la forma correcta de definir un punto de quiebre para pantallas con un ancho de hasta 700 píxeles, asegurando que los estilos dentro de esta consulta se apliquen a dispositivos con un ancho igual o inferior a 700 píxeles.

# 05 Para saber más: más de una Media Query

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87686/next)

¿Sabías que puedes tener más de una media query en tu proyecto? Es posible definir un tamaño de pantalla para el celular, otro para la tableta y dejar el "predeterminado" para el escritorio.

Por ejemplo:

Podemos definir un ancho máximo de "480px" para el celular: @media (max-width: 480px), y en otra media query definir un ancho máximo de "800px" para las tabletas: @media (max-width: 800px), y luego asignamos los ajustes necesarios dentro de cada media query, de esta manera tendremos nuestro sitio 100% responsivo.

También podemos definir rangos para los tamaños de pantalla con una única media query, asignamos el valor mínimo y luego el valor máximo separándolos con el atributo "and", así: @media (min-width: 480px ) and (max-width: 800px), en este caso, los estilos se aplicarán en pantallas de al menos "480px" y como máximo "800px".

Recordando que la mayoría de los accesos a internet hoy en día se realizan a través de dispositivos móviles, por lo tanto, no olvides siempre hacer que tus proyectos y sitios sean responsivos.

Para aprender más, visita el sitio de Mozilla y lee la documentación sobre [media queries](https://developer.mozilla.org/es/docs/Web/CSS/CSS_media_queries/Using_media_queries).

# 06 Para saber más: ¿qué es la responsividad?

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87688/next)

¿Has notado que cuando accedemos a algunos sitios web, cambian su diseño dependiendo del dispositivo? Por ejemplo, si accedes al sitio web de Alura en una computadora, el diseño será un poco diferente que si lo accedes desde un celular.

Pero, ¿qué herramienta es responsable de esto?

¡La responsividad! Cuando el sitio web adapta el tamaño de sus páginas (diseño) según el tamaño de la pantalla del dispositivo desde el cual se accede o cuando reducimos el tamaño de la ventana del navegador (esta transformación también se puede hacer cuando aplicamos un zoom en la página), decimos que este sitio web es responsivo.

La responsividad no solo cambia el tamaño de las fuentes y elementos, sino que también soporta cualquier cambio, como por ejemplo: en el color de fondo, color de texto, bordes, etc. Esto depende de los estilos aplicados dentro de la media query en el archivo CSS.

Si asignas un color de fondo diferente al body dentro de la media query, el color solo cambiará cuando la pantalla alcance el tamaño definido por ella.

En este ejemplo, se aplicó el color morado al body dentro de la media query en el proyecto para pantallas de máximo 1200 píxeles. ¡Mira qué genial es el resultado!

Puedes aprender mucho más aquí en Alura sobre responsividad a través de este excelente curso sobre [CSS: Flexbox e Layouts Responsivos](https://app.aluracursos.com/course/css-flexbox-layouts-responsivos?utm_source=gnarus&utm_medium=timeline).

# 03 Ajustando el ancho

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87716/next)

En el video, fue definido un ancho automático para la sección "presentación de contenido" con width: auto;, hay otra forma de hacerlo que es muy utilizada en la versión móvil.

Marque la opción a continuación que presenta la forma correcta para la versión móvil.

* Alternativa correta

@media (max-width: 1200px) {

.presentacion\_\_contenido {

width: 50%;

}

}

* Alternativa correta

@media (max-width: 1200px) {

.presentacion\_\_contenido {

width: 500px;

}

}

* Alternativa correta

@media (max-width: 1200px) {

.presentacion\_\_contenido {

width: 100%;

}

}

De esta manera, aplicando el 100% en el ancho de la página responsiva, obtendrá el mismo resultado que con width: auto; y se ajustará automáticamente al tamaño de la pantalla.

# 5 Para saber más: creando una cuenta en GitHub

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87692/next)

El primer paso para crear una cuenta en GitHub es acceder al sitio web de GitHub a través de este enlace. Si ya tienes una cuenta, haz clic en "Sign in"; si no tienes una cuenta, haz clic en "Sign up" para crear una nueva:

Verás una página como esta:

En la página de registro de GitHub, encontrarás campos para completar el formulario de la nueva cuenta.

Haz lo siguiente:

1. Ingresa tu correo electrónico;
2. Crea una contraseña;
3. Ingresa un nombre de usuario;
4. Si deseas recibir actualizaciones y anuncios de productos por correo electrónico, ingresa "y" para sí o "n" para no;
5. Realiza la verificación humana;
6. Haz clic en el botón "Create Account".

Después de completar tu información personal y hacer clic en "Create Account", recibirás una confirmación de registro por correo electrónico. A partir de entonces, selecciona las opciones que consideres adecuadas para tus preferencias.

Una vez validada tu cuenta, ¡ya puedes utilizar los servicios de GitHub!

# 06 Para saber más: Git y GitHub

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87694/next)

Git es una herramienta de control de versiones para tus archivos, proyectos o código. Por otro lado, GitHub es una aplicación o sitio web donde puedes almacenar tus proyectos con los archivos incluidos.

Git actúa como intermediario entre nuestra máquina local y GitHub, realizando tareas a través de comandos, donde puedes indicarle qué archivos o carpetas deseas enviar (subir) a GitHub.

Lo bueno de GitHub es que puedes almacenar tus proyectos en él y compartirlos con otros desarrolladores, permitiéndoles hacer sugerencias y ofrecer ayuda en otros proyectos.

Es esencial que todo desarrollador aprenda a utilizar Git y GitHub, por lo que te recomiendo este excelente curso sobre el tema a través del enlace.

También tenemos este artículo de nuestro CEO Paulo Silveira sobre los primeros pasos y la configuración de Git y GitHub aquí en nuestra plataforma de Alura, que puedes consultar en el [enlace](https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos).

Además, también puedes ver los dos videos de nuestra querida instructora Génesys, en la serie “Git y Github para Principiantes”:

# 07 Haga lo que hicimos

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87711/next)

1) Ajustando el espaciado para pantallas más pequeñas

Estás trabajando en un proyecto web y notas que el diseño para pantallas más pequeñas necesita ser mejorado. El contenido parece estar "apretado" en el centro, debido al espacio excesivo en los laterales. Tu objetivo es ajustar el "padding" de la clase .presentacion en CSS para mejorar la visualización en dispositivos más pequeños.

2) Refinando el ancho del contenido en dispositivos más pequeños

Después de ajustar el espaciado, te das cuenta de que el contenido dentro de la clase .presentacion\_\_contenido aún ocupa mucho espacio en pantallas más pequeñas. Tu desafío ahora es ajustar el ancho de esta clase para que se ajuste al nuevo "padding" aplicado en la clase .presentacion.

3) Evaluando el efecto de "width: auto" en dispositivos más pequeños

La última tarea implica evaluar el impacto del cambio de "width" a "auto" en la clase .presentacion\_\_contenido. Necesitas verificar si el ancho del contenido se adapta correctamente al "padding" del 5% aplicado, asegurando una visualización adecuada en pantallas más pequeñas.

4) Creando una cuenta en GitHub

En este ejercicio, crearás una cuenta en GitHub, una plataforma esencial para cualquier persona que trabaje en desarrollo web. GitHub permite alojar y compartir proyectos, facilitando la colaboración. Tu tarea es navegar por el proceso de registro en GitHub, configurar tu cuenta y explorar las principales funcionalidades disponibles.

5) Subiendo un proyecto a GitHub

Ahora que tienes una cuenta en GitHub, el siguiente paso es aprender a "subir" un proyecto. Utilizarás GitHub para alojar tu proyecto web, que incluye archivos HTML, CSS, JavaScript, entre otros. El enfoque estará en el proceso de crear un nuevo repositorio y cargar los archivos de tu proyecto en GitHub.

# 03 Para saber más: repositorio y Readme

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87696/next)

El Repositorio es simplemente una carpeta del proyecto que se guarda en la nube. En el caso de GitHub, creas el repositorio con el nombre de tu proyecto y subes los archivos dentro de esta carpeta. Después de crear el repositorio y colocar los archivos dentro de él, ahora puedes crear y personalizar el famoso archivo Readme.md, donde puedes especificar el proyecto y hablar más sobre él. De esta manera, otras personas que accedan a tu repositorio podrán ver y entender tu proyecto, así como las herramientas utilizadas para desarrollarlo.

La página del archivo Readme.md se puede escribir utilizando HTML o el lenguaje de marcado Markdown, siendo más común el uso de Markdown.

Puedes leer más sobre Markdown a través de este artículo.

Para aprender más sobre Readme, ver ejemplos y crear uno desde cero, visita el enlace del template sobre Readme.

Invita a la gente a conocer los Readmes de los proyectos de Alura.

¡Mira qué genial es el Readme del proyecto Alura Midi del curso de Javascript, accede a: Readme Alura Midi!

Y este otro Readme del proyecto de Android con Kotlin del curso de Android, accede a: Readme Android Kotlin!

También echa un vistazo a este ejemplo de Readme escrito en Markdown hecho específicamente para el repositorio de nuestro curso de HTML y CSS:

Escrito en Markdown

# Portfolio ¡Hola, bienvenido a mi proyecto de portafolio!

![imagen](https://user-images.githubusercontent.com/77756047/211304452-220fedf0-f91b-490f-8a65-a60ce860bc5c.png)

## Herramientas utilizadas:

\* HTML

\* CSS

\* Flex-box

## Hecho por:

###Ana García

### Linkedin: https://www.linkedin.com/school/aluracursos/

Ahora, ¿qué tal editar tu archivo Readme y personalizarlo a tu manera? ¡Explica un poco más sobre tu proyecto, las herramientas utilizadas, el objetivo del proyecto, etc.! Y lo más importante, ¡comparte tu experiencia de aprendizaje con otras personas! =)

# 05 Para saber más: creando una cuenta en Vercel

[Próxima Actividad](https://app.aluracursos.com/course/html-css-trabajando-responsividad-publicacion-proyectos/task/87697/next)

El primer paso para crear una cuenta en Vercel es acceder al sitio web de Vercel a través de este enlace y hacer clic en "Sign up".

En la página de inicio de Vercel, encontrarás un botón de inicio de sesión y otro de registro "Sign Up", ambos en la esquina superior derecha.

Después de hacer clic en "Sign up", verás una página como esta:

En la página de registro de Vercel, encontrarás tres botones en el lado derecho de la página, con la primera opción para iniciar sesión con GitHub, la segunda opción con GitLab y la tercera opción con Bitbucket.

Luego, haz clic en "continuar con GitHub" ¡y listo! ¡Tu cuenta ha sido creada! ¡Ahora solo tienes que utilizar los servicios de Vercel! :)

# Guía sobre cómo instalar Git en diferentes sistemas operativos



Bruno Souza



Git es una herramienta de control de versiones que se puede instalar en varios sistemas operativos. En esta guía, aprenderás cómo instalar Git en Windows, Mac y distribuciones de Linux.

### Instalación en Windows

1. Descarga el instalador de Git

* Visita el sitio oficial de Git (<https://gitforwindows.org/>) y haz clic en el botón "Download" para descargar el instalador de Git más reciente para Windows.
* Haz clic en el archivo descargado para iniciar el proceso de instalación.

2. Inicia la instalación

* Haz clic en "Next" en la pantalla de bienvenida.

3. Selecciona la ubicación de la instalación

* Elige el lugar donde te gustaría que Git se instalara en tu computadora. La ubicación predeterminada generalmente es adecuada, pero puedes cambiarla si es necesario. Haz clic en "Next" para continuar.

4. Selecciona los componentes a instalar

* Verás una lista de componentes para instalar. Las opciones predeterminadas suelen ser suficientes, pero siéntete libre de hacer cambios según tus necesidades. Después de hacer tus selecciones, haz clic en "Next".

5. Selecciona el menú de inicio

* Puedes elegir un nombre para el acceso directo de Git en el menú de inicio. La opción predeterminada es "Git". Después de hacer tu selección, haz clic en "Next".

6. Elige el editor predeterminado que utilizará Git

* Selecciona el editor que prefieres usar con Git. Por defecto, sugerirá el editor Vim, pero puedes elegir otro editor que esté instalado en tu sistema, como Notepad++, VS Code, Atom, etc. Después de hacer tu selección, haz clic en "Next".

7. Elige el nombre de la rama inicial

* Se te pedirá que elijas el nombre de la rama inicial. Los nombres "master" y "main" son comúnmente utilizados, siendo "main" la nueva convención predeterminada. Elige la que prefieras para tu ambiente de trabajo y haz clic en "Next".

8. Ajusta tu PATH

* En la próxima pantalla, se te pedirá que ajustes tu PATH. Elige la opción "Git from the command line and also from 3rd-party software" para asegurarte de que podrás usar Git desde la línea de comandos de Windows o PowerShell. Haz clic en "Next".

9. Elecciones de SSH y Configura los finales de línea

* En las siguientes pantallas, se te pedirá que elijas el cliente SSH, no hay necesidad de cambiar la elección predeterminada, haz clic en “Next” hasta llegar a “Configuring the line ending conversions”, que significa cómo Git debe tratar los finales de línea. La opción recomendada es "Checkout Windows-style, commit Unix-style line endings". Haz clic en "Next".

10. Elige el emulador de terminal

* Aquí puedes elegir el emulador de terminal que Git usará. La opción predeterminada es "MinTTY", pero puedes elegir usar la línea de comandos de Windows si lo prefieres. Haz clic en "Next".

11. Elige el administrador de credenciales

* Aquí debes dejar seleccionado “Git Credential Manager”. Haz clic en "Next".

12. Elige las opciones extra

* Aquí puedes elegir algunas opciones extras. Las opciones predeterminadas son generalmente suficientes para la mayoría de los usuarios. Haz clic en "Next".

13. Instala

* Haz clic en "Install" para comenzar la instalación. Verás una barra de progreso mientras se está instalando Git.

14. Completa la instalación

* Una vez que la instalación se haya completado, haz clic en "Finish". Ahora Git debería estar instalado en tu computadora.

Para verificar si la instalación fue exitosa, puedes abrir Git Bash (o PowerShell) y escribir git --version. Esto debería devolver la versión de Git que acabas de instalar.

#### **Configurando el nombre de usuario en Git**

1. Abre la terminal de Windows o Git Bash, que puedes encontrar escribiendo en la barra de búsqueda.
2. Escribe el siguiente comando:

git config --global user.name "Tu Nombre"

Reemplaza "Tu Nombre" por el nombre que deseas asociar a todos tus commits en Git. Este no necesita ser el mismo que tu nombre de usuario de GitHub o de cualquier otro servicio Git que puedas estar usando, aunque muchas personas prefieren mantener la consistencia.

#### **Configurando el correo electrónico en Git**

En la terminal, escribe el siguiente comando:

git config --global user.email "tu-email@ejemplo.com"

Reemplaza "[**tu-email@ejemplo.com**](mailto:tu-email@ejemplo.com)" por la dirección de correo electrónico que deseas asociar a todos tus commits en Git. Usa el mismo correo electrónico que utilizas para tu cuenta de GitHub u otros servicios Git.

Con estos comandos, has configurado un nombre de usuario y una dirección de correo electrónico para Git, que se asociarán a todos los commits que realices.

Si quieres verificar las configuraciones, puedes usar los siguientes comandos:

* Para el nombre de usuario:

git config --global user.name

* Para el correo electrónico:

git config --global user.email

Estos comandos devolverán el nombre de usuario y el correo electrónico configurados actualmente en Git, respectivamente.

### Instalación en distribuciones de Linux

#### Debian/Ubuntu

Paso 1: Actualiza el sistema

Antes de instalar Git, es una buena práctica actualizar el sistema para garantizar que tengas las versiones más recientes de todos los paquetes. En la terminal, escribe:

sudo apt update

sudo apt upgrade

Paso 2: Instala Git

Ahora puedes instalar Git. En la terminal, escribe:

sudo apt install git

Esto iniciará el proceso de instalación de Git.

Paso 3: Verifica la instalación

Una vez que la instalación esté completa, puedes verificar si fue exitosa comprobando la versión de Git. En la terminal, escribe:

git --version

Esto debería devolver la versión de Git que acabas de instalar.

Paso 4: Configura Git

Finalmente, debes configurar tu nombre de usuario y dirección de correo electrónico en Git. En la terminal, escribe:

git config --global user.name "Tu Nombre"

git config --global user.email "tu-email@ejemplo.com"

Reemplaza "Tu Nombre" por el nombre que deseas usar en Git y "[**tu-email@ejemplo.com**](mailto:tu-email@ejemplo.com)" por tu dirección de correo electrónico utilizada en tu cuenta de GitHub.

¡Y eso es todo! Ahora deberías tener Git instalado y configurado en tu sistema Linux.

#### **Fedora**

Paso 1: Actualiza el sistema

Antes de instalar Git, es una buena práctica actualizar el sistema para garantizar que tengas las versiones más recientes de todos los paquetes. En la terminal, escribe:

sudo dnf update

Paso 2: Instala Git

Ahora puedes instalar Git. En la terminal, escribe:

sudo dnf install git

Esto iniciará el proceso de instalación de Git.

Paso 3: Verifica la instalación

Una vez que la instalación esté completa, puedes verificar si fue exitosa comprobando la versión de Git. En la terminal, escribe:

git --version

Esto debería devolver la versión de Git que acabas de instalar.

Paso 4: Configura Git

Finalmente, debes configurar tu nombre de usuario y dirección de correo electrónico en Git. En la terminal, escribe:

git config --global user.name "Tu Nombre"

git config --global user.email "tu-email@ejemplo.com"

Reemplaza "Tu Nombre" por el nombre que deseas usar en Git y "[**tu-email@ejemplo.com**](mailto:tu-email@ejemplo.com)" por tu dirección de correo electrónico utilizada en tu cuenta de GitHub.

¡Listo! Ahora deberías tener Git instalado y configurado en tu sistema Fedora.

### Instalación en Mac OS

Instalar Git en un Mac es un poco diferente a instalarlo en un sistema Linux, pero sigue siendo un proceso bastante directo.

Paso 1: Verifica si Git ya está instalado

macOS viene con Git instalado por defecto en muchos casos. Para verificar si Git ya está instalado, abre la Terminal y escribe:

git --version

Si Git está instalado, este comando devolverá la versión actualmente instalada. Si Git no está instalado, es probable que este comando inicie el proceso de instalación de Git.

Paso 2: Instala Git (si es necesario)

Si Git no está instalado, puedes instalarlo usando [Homebrew](https://brew.sh/), un gestor de paquetes para macOS. Si no tienes [Homebrew](https://brew.sh/) instalado, puedes instalarlo primero con el siguiente comando en la Terminal:

/bin/bash -c "$(curl -fsSL <https://raw.githubusercontent.com/Homebrew/install/HEAD/install.sh>)"

Una vez que Homebrew esté instalado, puedes usar el siguiente comando para instalar Git:

brew install git

Paso 3: Verifica la instalación

Una vez que la instalación esté completa, puedes verificar si fue exitosa comprobando la versión de Git. En la Terminal, escribe:

git --version

Esto debería devolver la versión de Git que acabas de instalar.

Paso 4: Configura Git

Finalmente, debes configurar tu nombre de usuario y dirección de correo electrónico en Git. En la Terminal, escribe:

git config --global user.name "Tu Nombre"

git config --global user.email "tu-email@ejemplo.com"

Reemplaza "Tu Nombre" por el nombre que deseas usar en Git y "[tu-email@ejemplo.com](mailto:tu-email@ejemplo.com)" por tu dirección de correo electrónico utilizada en tu cuenta de GitHub.

¡Y eso es todo! Ahora deberías tener Git instalado y configurado en tu macOS.

### Alternativa con MacPorts

Si estás utilizando [MacPorts](https://www.macports.org/install.php) para administrar paquetes en tu Mac, también puedes usarlo para instalar Git.

Si Git no está instalado, puedes instalarlo usando [MacPorts](https://www.macports.org/install.php) con el siguiente comando en la Terminal:

sudo port install git

#### **Verifica la instalación**

Una vez que la instalación esté completa, puedes verificar si fue exitosa comprobando la versión de Git. En la Terminal, escribe:

git --version

#### **Configura Git**

Finalmente, debes configurar tu nombre de usuario y dirección de correo electrónico en Git. En la Terminal, escribe:

git config --global user.name "Tu Nombre"

git config --global user.email "tu-email@ejemplo.com"

Reemplaza "Tu Nombre" por el nombre que deseas usar en Git y "[**tu-email@ejemplo.com**](mailto:tu-email@ejemplo.com)" por tu dirección de correo electrónico utilizada en tu cuenta de GitHub.

¡Perfecto! Ahora deberías tener Git instalado y configurado en tu macOS usando MacPorts.

# Curso de **Git y GitHub: repositorio, commit y versiones**

# 02 Explicación

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84969/next)

# ¡Hola!

Es genial tenerte aquí para aprender sobre Git y GitHub. Estas herramientas desempeñan un papel clave en el desarrollo de software colaborativo, permitiendo que los equipos trabajen juntos de manera eficiente y organizada. Exploraremos los conceptos básicos y vamos a comprender cómo usar Git junto con GitHub para administrar los proyectos de manera efectiva.

## División de las clases

| Clase | ¿Qué vamos a aprender en esta clase? |
| --- | --- |
| Cuenta en GitHub | Vamos a conocer la plataforma de alojamiento de código colaborativa más importante del mundo. |
| Commit, VSCode y equipo | Vamos a ver cómo guardar los cambios en el historial del proyecto e invitar a una persona para formar parte. |
| Trabajando localmente | Trabajaremos localmente con Git para mejorar nuestra colaboración y eficiencia. |
| Ramificaciones y merge | Vamos a aprender cómo crear nuevos espacios de trabajo dentro del proyecto y cómo juntarlos. |
| Profundizando en Git y GitHub | Vamos a ver otra forma de creación de nuestro repositorio |

## ¿Necesito instalar algún software o programa para seguir este curso?

Al principio, no es necesario. Debes tener una cuenta de correo electrónico para registrarte en GitHub.

Si hay dudas durante la instalación o durante el curso, ¡cuenta siempre con el foro de Alura!

:)

# 06 ¿Para qué sirve GitHub?

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84698/next)

Un equipo de personas desarrolladoras está trabajando en un proyecto para realizar cambios en la plataforma de cursos de Alura. Una persona está haciendo una mejoría de código, otra necesita corregir un error en el sistema y otra hará alteraciones en los colores del Dashboard.

Estos cambios están ocurriendo simultáneamente y, para tornar ese proceso más ágil, los programadores quieren tener acceso al código actualizado con los cambios realizados por el equipo en tiempo real.

¿Qué puedes hacer para resolver este problema?

* Alternativa correta

Utilizar Github, pues es una plataforma de gestión de código que nos permite gestionar pequeños equipos y es específica para alojamiento de códigos relacionados al front-end.

* Alternativa correta

Podemos usar Github, ya que nos permite unificar los cambios de todo el equipo en un solo lugar, al que llamamos repositorio.

* Sí, con Github, el equipo de desarrolladores podrá manejar con mayor facilidad los cambios del proyecto. Un repositorio es básicamente una carpeta donde quedarán reunidos todos los archivos del proyecto y podrá ser accedido por todos los miembros del equipo.
* Alternativa correta

Copiar las implementaciones en el código y enviarlo a todos los miembros del equipo; y el mismo proceso se haría para ellos.

# 07 Para saber más: Readme

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84704/next)

Ya debes haber instalado algún software que contenía un archivo en .txt en una de sus carpetas con el nombre Readme. Bueno, este archivo significa literalmente ‘’Léame’’, y contiene instrucciones que generalmente son para que tú sepas como hacer la instalación del software.

En proyectos de GitHub, el Readme es un archivo con una extensión .md (Markdown) que contiene las informaciones necesarias para entender el objetivo y el estado de tu proyecto. Markdwon es un lenguaje de marcación de texto, así como HTML, pero más simple. ¿Quiere saber más sobre Markdown? [Paulo Silveira te explicará en este artículo](https://www.aluracursos.com/blog/como-trabajar-con-markdown).

Pero, ¿por qué necesito tener un Readme en el repositorio de mi proyecto?

En el Readme, puedes dar una descripción acerca de qué trata tu proyecto, cuál el estado actual de él, las tecnologías utilizadas, el equipo que trabaja en el desarrollo, como contribuir (se es un proyecto open source), como usar, etc. Existe una gran posibilidad de cosas que puedes poner en tu readme con el objetivo de informar y ser lo más atractivo y organizado posible para quien esté visualizando tu proyecto.

Y para ayudarte en esta tarea de personalizar tu Readme, Camila Fernanda Alves escribió un [artículo increíble](https://www.aluracursos.com/blog/como-escribir-un-readme-increible-en-tu-github) que te mostrará cómo escribir un excelente Readme para tu proyecto.

# 02 Realizando un commit

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84699/next)

El commit es una acción muy importante del Git, que nos ayuda a tener un control de versionamiento de nuestro proyecto. El commit nos permite guardar un estado de nuestro proyecto en un determinado punto.

Teniendo en cuenta esto, cuando realizamos un commit, estamos:

* Alternativa correta

Creando un repositorio en Github.

* Alternativa correta

Agregando cambios más recientes del proyecto.

* En el momento que hacemos un commit, estamos guardando el estado actual de nuestro proyecto.
* Alternativa correta

Estamos enviando modificaciones de nuestro proyecto al repositorio remoto.

# 03 Para saber más: nombramiento de commits

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84705/next)

No existe una regla específica para el mensaje de los commits, pero en la comunidad internacional, especialmente en los países de habla inglesa, se recomienda usar el imperativo al comienzo de los commits.

Además, es importante mencionar de forma descriptiva lo que se cambió, agregó o eliminó del proyecto. Durante las clases, la instructora Ellen presenta de manera didáctica y sencilla cómo hacerlo, pero en un ambiente de trabajo, cada empresa puede presentar una sintaxis estándar a seguir.

# 01 Preparando el ambiente: instalando Git

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84710/next)

¡En esta aula, vamos a trabajar con Git local!

Vamos a instalar Git para comenzar a controlar las versiones de nuestros códigos. Para esto, haz click en este enlace para abrir la página oficial de Git:

<https://git-scm.com/>

Ahora, en base a tu sistema operativo, descarga la instalación de Git, como está ilustrado en la imagen de abajo:

[A través de este tutorial](https://www.atlassian.com/es/git/tutorials/install-git), puedes verificar el método de instalación de acuerdo a tu sistema operativo.

Te recomendamos realizar la instalación antes de continuar el curso. En caso de dudas o si tienes algún problema con la instalación de Git, puedes consultarnos a través del foro.

# 2 Para saber más: factores de autenticación

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84703/next)

Más allá de crear una contraseña fuerte, GitHub ha estado trabajando para fortalecer la seguridad para el inicio de sesión y el control de versiones del proyecto.

Una de esas medidas es el acceso a través de tokens. En este artículo, Camila Alves explica la importancia y el paso a paso de cómo hacerlo.

* [Nueva exigencia de autenticación por token en Git ¿qué es y qué debo hacer?](https://www.aluracursos.com/blog/exigendia-autenticacion-por-token)

Puedes leer también [la documentación](https://docs.github.com/es/authentication/keeping-your-account-and-data-secure/managing-your-personal-access-tokens).

# 05 Para saber más: Git Clone

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84707/next)

Imagina que estás trabajando en un proyecto que ya está configurado en un repositorio de origen, y deseas colaborar con este proyecto. Con git clone, es posible crear una copia de desarrollo en un repositorio local, y todos los cambios que realices van a ser gestionados a partir de este repositorio. El comando git clone es usado para seleccionar un repositorio existente y crear un clon o una copia de él en un repositorio local.

El comando git clone crea una copia de un repositorio git existente, y este repositorio puede ser local o remoto. Además, esta copia es un repositorio git completo, con tu propio historial, gestionamiento de tus propios archivos y es un ambiente aislado como un todo del repositorio original.

Por comodidad, la clonación crea una conexión remota que apunta al repositorio original. Y es esta conexión la que facilita mucho la interacción con el repositorio central. Puedes consultar un ejemplo demostrando el git clone [aquí](https://www.atlassian.com/es/git/tutorials/setting-up-a-repository/git-clone).

Con el git clone también puedes clonar el repositorio para una carpeta específica:

git clone <repositorio> <mi-proyecto-clone>

El repositorio localizado en repositorio es clonado para una carpeta llamada:

mi-proyecto-clone

También puedes configurar el git clone y clonar el repositorio desde una branch específica, diferente a la original, de esta manera:

git clone -branch new\_feature <repositorio>

En el ejemplo anterior se clonaria solamente la branch new feature de repositorio. Otras configuraciones de opciones de git clone [puedes consultar en este enlace](https://git-scm.com/docs/git-clone).

# 06 Para saber más: Git Log

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84706/next)

Si deseas verificar el historial de cambios, los mensajes de commit, el nombre de la persona autora del commit y otras informaciones sobre el proyecto, hay un comando de git que te puede ayudar. Este comando es git log.

Como ya sabemos, los commits poseen hashs que los identifican de una manera única, es decir, no existen dos commits con el mismo hash. Con git log podemos ver el hash y muchas otras informaciones del commit.

Podemos visualizar todos los commits, uno en cada línea con el comando:

git log –oneline

Si, en lugar de menos informaciones, queremos ver más, como las alteraciones del commit, podemos usar:

git log -p

También podemos buscar las informaciones de la persona autora del commit con el comando:

git log --author="user\_name"

Y buscar informaciones por fecha:

git log --since=1.month.ago --until=1.day.ago

En el comando anterior, estamos buscando las informaciones del commit desde hace un mes hasta hace un día.

Tu también puedes formatear la visualización de las informaciones del commit con el comando:

git log --pretty="format:%h %s"

Esto trae el hash seguido del mensaje de commit. Para saber más formas de mostrar las informaciones del commit, [aquí hay una lista](https://devhints.io/git-log-format) de formas en que puedes hacer.

Genial, ¿no? ¿Qué pasaría si te dijeras que hay muchos otros parámetros que podemos pasar en git log? Si deseas saber más sobre cómo mostrar tu información de commit, puedes consultar a [este enlace](https://devhints.io/git-log).

# 07 Para saber más: la consola

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84708/next)

Como vimos durante el curso, existen varios pasos de trabajo dentro de la consola, por lo que es importante prestar atención a los detalles para verificar que los comandos se están ejecutando dentro de la carpeta correcta:

Etapa de configuración:

Carpeta sin el versionado:

Carpeta con el versionado:

# 09 Para saber más: open source

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84709/next)

¿Sabes qué es el código abierto?

El código abierto es un código diseñado para que cualquier persona pueda acceder a él de forma abierta y pueda verlo, modificarlo y distribuirlo según sus necesidades.

Puedes ver más detalles [aquí](https://www.aluracursos.com/blog/open-source-una-breve-introduccion).

# 11 Navegando en el tiempo

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84700/next)

Hiciste cuatro commits en tu código HTML. Los cuales están descritos abajo:

1. Agregando un Title,
2. Agregando una Imagen de fondo,
3. Agregando una Tabla,
4. Agregando un Footer.

Pero los tres últimos commits no fueron aprobados por tu supervisor y necesitas volver al punto inicial del proyecto. ¿Qué comando necesitas utilizar para navegar en el pasado?

Con base en lo que vimos en clase, analice las alternativas abajo y marque solo la correcta.

* Alternativa correta

Utilizamos el comando checkout para volver al punto que queremos.

* Alternativa correta

No es posible retroceder tanto en un proyecto después de tantos commits.

* Alternativa correta

Usamos el comando restore para volver exactamente al punto inicial del proyecto.

Usamos el comando Restore informando el ID del primero commit después de la flag –source.

Te invito a leer más sobre Restore [aquí](https://www.aluracursos.com/blog/git-y-los-nuevos-comandos).

# 13 Lo que aprendimos

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84718/next)

Lo que aprendimos en esta aula:

* Entendimos la diferencia entre Git y Github;
* Vimos cómo descargar el proyecto completo a nuestra máquina con el git clone;
* Aprendimos cómo acceder al historial de commits de un repositorio a través del git log;
* Aprendimos a navegar en la consola a través de los comandos linux;
* Vimos algunas buenas prácticas al nombrar los archivos y de esa manera evitar problemas futuros;
* Vimos cómo utilizar el git commit en la consola;
* Aprendimos a sincronizar el proyecto con el comando git pull;
* El comando git status es súper útil y nos ayuda a entender el momento actual del proyecto;
* Aprendimos cómo agregar los archivos que queremos para el próximo commit usando git add;
* Aprendimos a enviar nuestros cambios/archivos locales con el git push;
* Vimos cómo verificar los cambios dentro de un archivo a través del git diff en la consola y en GitHub;
* Aprendimos a restaurar archivos y el proyecto, volviendo a un estado anterior a través de git restore;

En la siguiente aula:

¡GUAUUUUU! ¡Ahora vamos a aprender cómo trabajar con una nueva rama!

# 04 Merge de branches

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84701/next)

Estamos trabajando con dos branches: la branch main y la branch title. Hicimos varios cambios en la branch title, pero, ahora, queremos traer todo lo que está en la branch title para la main. ¿Cómo podemos hacer eso?

* Alternativa correta

Utilizamos los comandos: git switch main y git merge title.

* ¡Excelente! De esta forma con el git switch volvemos a la branch principal y luego hacemos el merge poniendo todo lo que estaba en la branch title en la branch main.
* Alternativa correta

Usamos el comando: git merge title.

* Alternativa correta

Usaremos los comandos: git switch title y git merge main.

# 05 Desafio: Hash del Merge

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84702/next)

Durante la clase, la instructora Ellen usó el comando git log --oneline para verificar el hash de la modificación creada dentro de la rama de desarrollo en el archivo contacto.html

¿Será que después de la fusión hubo un cambio en este número?

# 05 Para saber más

[Próxima Actividad](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/task/84970/next)

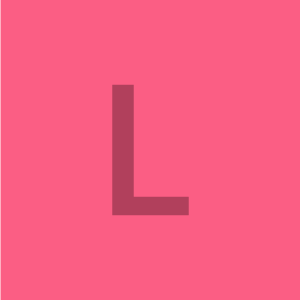
Para reforzar el contenido y seguir aprendiendo, tenemos algunos contenidos de apoyo.

En estos artículos de Luzdalis López, además del paso a paso, hay algunos contenidos extras para profundizar aún más en el tema:

* [Iniciando un repositorio con Git](https://www.aluracursos.com/blog/iniciando-repositorio-con-git)
* [Creando un repositorio en GitHub](https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github?utm_source=gnarus&utm_medium=timeline)
* [Clonando un repositorio con Git y GitHub](https://www.aluracursos.com/blog/clonando-un-repositorio-remoto?utm_source=gnarus&utm_medium=timeline)

¡Buenos estudios!

# Iniciando un repositorio con Git



Luzdalis Lopez



Antes de comenzar a desarrollar este contenido, es conveniente que sepa que este tema tiene una secuencia de tres artículos, donde este es el primero. Por si te interesan los otros dos: el segundo es [Creando un repositorio en GitHub](https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github?utm_source=gnarus&utm_medium=timeline) y el tercero es [Clonando un repositorio con Git y GitHub](https://www.aluracursos.com/blog/clonando-un-repositorio-remoto?utm_source=gnarus&utm_medium=timeline)

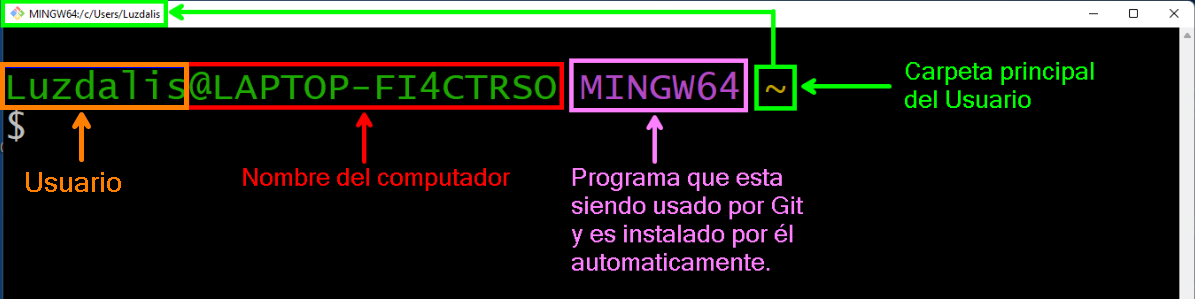
Todo principiante en programación debe tener en cuenta, la necesidad de saber qué es [Git y Github](https://www.aluracursos.com/blog/git-y-github-que-son-y-primeros-pasos), ya que son herramientas que te serán de mucha utilidad a lo largo de tu carrera; por lo que es conveniente que puedas usarlos.

¿Has creado tu primer proyecto y aún no lo has versionado? Si quieres que git comience a rastrear tu código para llevar un registro y control de los cambios realizados al mismo tiempo, pero no te queda muy claro cómo hacerlo; ¡Listo! ¡Este es el artículo ideal que te guiará sobre cómo llevar a cabo este versionado!

### Iniciando un nuevo repositorio en Git (máquina local)

En primer lugar, si no tiene instalado Git, le recomiendo que [lo descargue e instale en su computadora](https://git-scm.com/downloads); en general se instalará por defecto en tu disco duro C y ahí se basará esta práctica.

Con Git instalado, abra la terminal Git Bash, que se iniciará de forma predeterminada en el directorio principal (~) o /c/Users/<User-of-windows-account>/ de su usuario, como se muestra en la siguiente imagen:



Si desea determinar el directorio actual, escriba pwd en la línea de comando $, como se muestra a continuación:



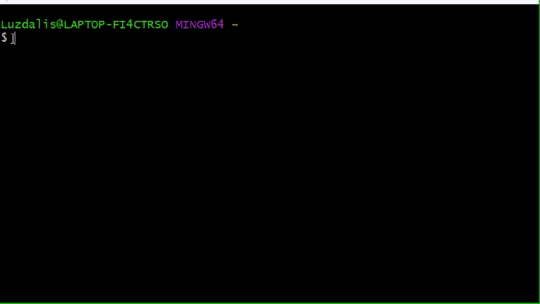
Notas importantes:

* Si vas a usar Git por primera vez, primero deberás configurarlo, estableciendo tu nombre de usuario y dirección de correo electrónico, ya que esta información se usará de manera inalterable en los "commits" que envíes. A continuación les dejo los [*comandos*](https://git-scm.com/book/es/v2/Inicio---Sobre-el-Control-de-Versiones-Configurando-Git-por-primera-vez) para configurar estas variables globales:
* $ git config --global user.name "Tu nombre de Usuario"

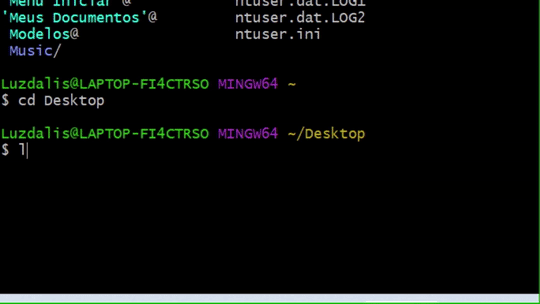
$ git config --global user.email "tu@email.com"

* Previamente, debes conocer la ruta o la ruta donde se encuentra almacenado tu proyecto (carpeta o directorio), para evitar frustraciones al usar la terminal.

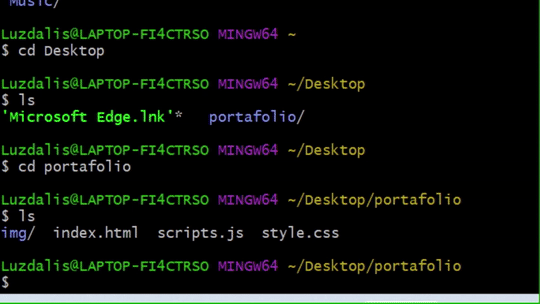
1. Se listarán los archivos del directorio principal y luego se accederá a la carpeta Desktop; allí se encuentra el proyecto, en el cual, se quiere iniciar el repositorio:



1. A continuación se mostrarán los archivos de la alfombra de Escritorio y entrarás al portafolio del proyecto donde se visualizarán tus archivos:



1. Finalmente, se ejecutará el comando: git init, para comenzar a rastrear git en el repositorio indicado:



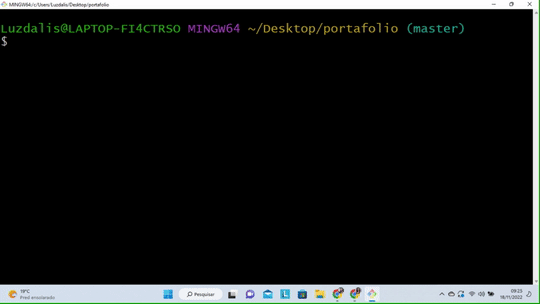
Una vez ejecutado el comando aparecerá el siguiente mensaje indicando que se ha iniciado el repositorio:

`Initialized empty Git repository in C:/Users/Luzdalis/Desktop/portafolio/.git/`

Git mostrará el nombre de la rama principal (rama) que está en uso, que generalmente se llama master.

\*Nota: Para garantizar la compatibilidad futura, se recomienda actualizar el nombre de la rama de master a principal con el siguiente comando:

$ git branch -M main



### Guardar cambios en el nuevo repositorio local

Es importante que tengas en cuenta que cada archivo de tu repositorio puede tener uno de los estados: rastreado o no rastreado.

Los archivos rastreados (tracked files en inglés) son los archivos que git está viendo y pueden estar sin modificar (Unmodified), modificados (Modified) o preparados (Staged ).

Por otro lado, los archivos sin seguimiento (Untracked files) son todos aquellos archivos que no tienen seguimiento y no se encuentran en el área de preparación.

Para ilustrar estos estados se muestra la siguiente imagen, extraída de la documentación oficial de Git:



Si quieres profundizar un poco más en cómo: [guardar cambios en el Repositorio, puedes encontrarlos aquí](<https://git-scm.com/book/es/v2/Fundamentos-de-Git-Saving>- cambios-en-el-Repositorio).

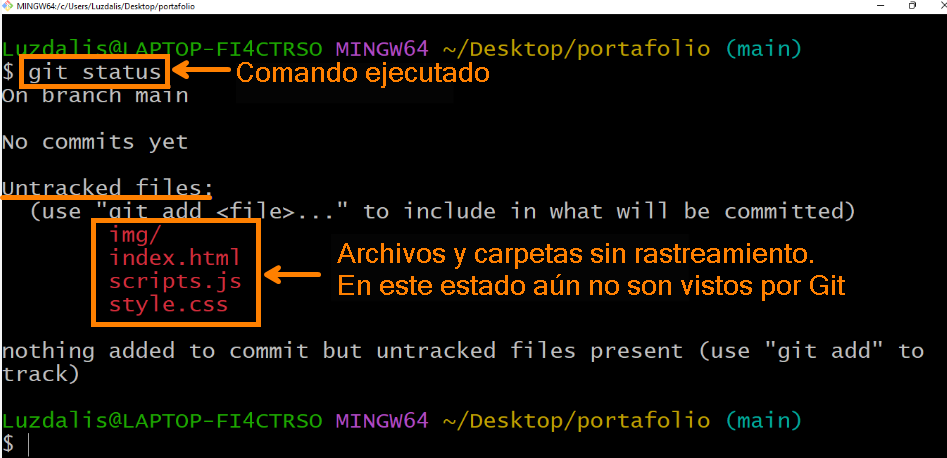
Dentro de este orden de ideas se iniciará el control y registro de versiones del proyecto:

#### Primer paso: ver el estado del repositorio

Para conocer el estado en el que se encuentran los archivos del repositorio utilice el siguiente comando:

* $ git status

Cabe señalar que al iniciar un repositorio; la rama principal main está vacía, es decir, no hay archivo, Git no está rastreando ninguna alfombra.



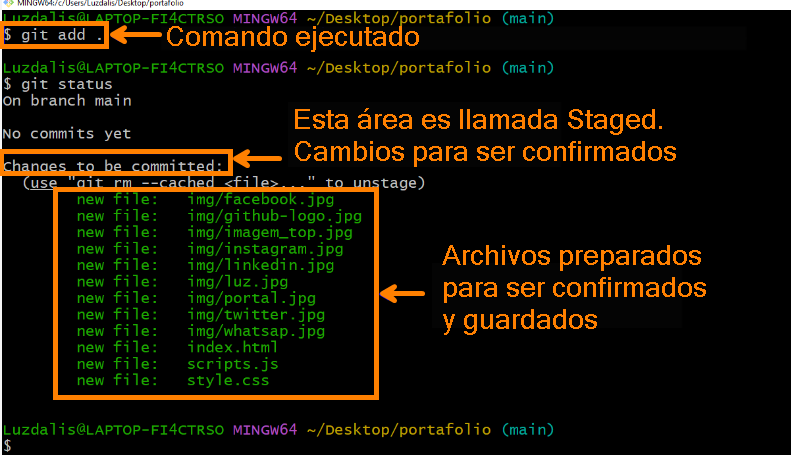
​ Untracked files (Archivos sin rastreamiento).

#### Segundo paso: agregar archivos para confirmar

Para comenzar a rastrear los archivos, se ejecutará el comando git add para agregar contenido a la próxima confirmación. Se puede utilizar de dos formas:

* $ git add nombre\_del\_archivo /\*\*\*Añade un archivo específico\*\*\*/

$ git add . /\*\*\*Adiciona todos los archivos de una sola vez.\*\*\*/

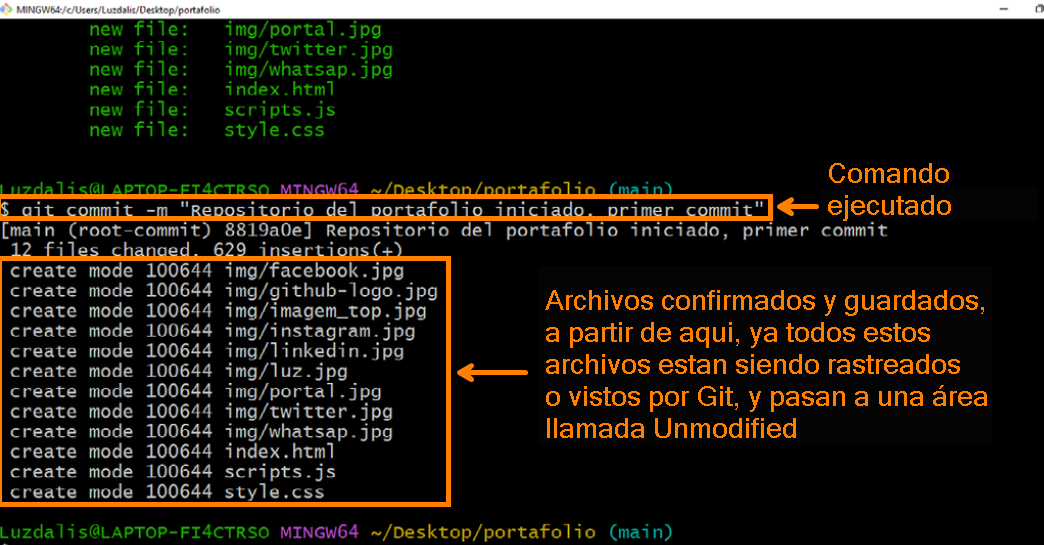


​ Changes to be committed (cambios preparados para confirmación).

#### Tercer paso: confirmar los archivos

Para confirmar los cambios, utilice el comando:

$ git commit -m "Mensaje descriptivo del commit"



### Lista de comandos básicos más usados en el versionado de Git

* git config: Esto es lo primero que debe hacer al instalar Git, defina su nombre de usuario y dirección de correo electrónico.

$ git config --global user.name "Tu nombre de Usuario"

$ git config --global user.email "tu@email.com"

* git init: Inicia o crea un repositorio.

$ git init

* git status: Visualiza el estado del repositorio.

$ git status

* git add: Prepara contenido para la próxima confirmación.

$ git add nombre\_del\_archivo

$ git add .

* git commit: Guarda contenido actual junto con un mensaje de registro de usuario que describe los cambios.

$ git commit -m "Mensaje descriptivo"

* git clone: Clona un repositorio existente.

$ git clone [url]

* git branch: Una branch no es mas que un puntero para desplazamiento de un commit a otro, se puede utilizar de varias formas:

$ git branch //Muestra las branches exitentes en un repositorio

$ git branch nombre\_branch //Crea una nueva branch

$ git branch -M nombre\_branch //para renombrar la branch actual

* git checkout: Para cambiar de una branch para otra.

$ git checkout

$ git checkout -b nombre\_branch //Crea una nueva branch y apunta para ella.

* git log: Muestra el historial de commit, se puede usar de varias formas:

$ git log

$ git log --oneline

* git remote: Muestra el repositorio remoto:

$ git remote

$ git remote -v

* git pull: Descarga el contenido de repositorio remoto y actualiza automáticamente el repositorio local.

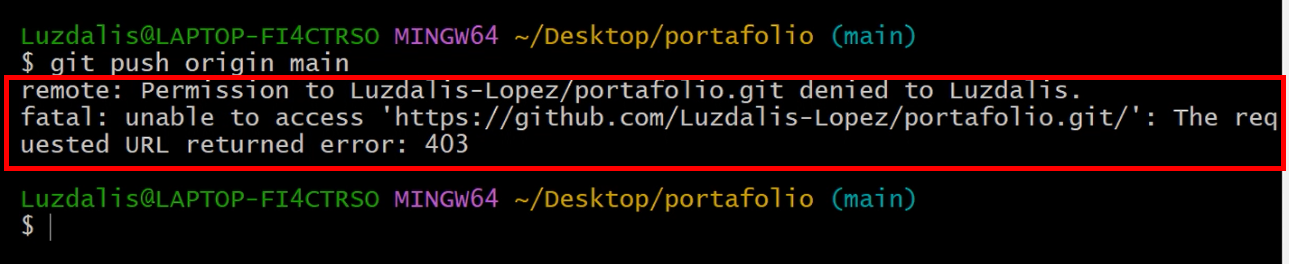
$ git pull

* git push: Envia contenido desde el repositorio local y actualiza automaticamente el repositorio remoto

$ git push

### Algunas recomendaciones:

* Para la creación de alfombras o directorios, evite poner espacios y acentos en el nombre, esto evitará roturas o errores, al ejecutar los comandos en la terminal.
* Si eres nuevo usuario de Git y GitHub, es muy importante que realices las configuraciones de las variables globales y de la clave SSH, para que te evites dolores de cabeza y no logres hacer el versionado correctamente; uno de los errores más comunes vistos como resultado de perder algunos de ellos, es el siguiente en la imagen:



Permission to... denied to... (Permiso denegado)

Este error aparece cuando hemos clonado un repositorio con la dirección URL HTTP vista anteriormente, y estamos intentando enviar actualizaciones al repositorio remoto, solo que GitHub no reconoce quién está enviando los datos; para solucionar este problema debes crear una clave SSH que te autentique para que puedas hacer cambios en ella, luego debes hacer un clon con la dirección URL SSH y listo. Solo necesitarás hacerlo una vez; a partir de ahí tu usuario quedará autenticado para enviar cambios a cualquier repositorio existente en tu perfil de GitHub sin ningún inconveniente.

Para finalizar, te paso, a modo de recordatorio, los demás enlaces de los artículos que te pueden ser de gran ayuda en este inicio de versionado con Git y GitHub, que fortalecerán el aprendizaje en tu carrera profesional:

* El segundo es [Creando un repositorio en GitHub](https://www.aluracursos.com/blog/creando-repositorio-remoto-en-github?utm_source=gnarus&utm_medium=timeline)
* El tercero es [Clonando un repositorio con Git y GitHub](https://www.aluracursos.com/blog/clonando-un-repositorio-remoto?utm_source=gnarus&utm_medium=timeline)

# Buenas practicas en git: evitando errores



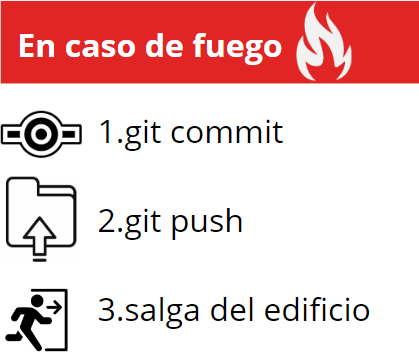
Ingrid Silva

Este artículo forma parte de una serie de cinco artículos que abordan cómo evitar y corregir errores en Git. Si te interesa profundizar en este tema y aprender a evitar errores comunes en Git, te recomiendo leer los siguientes artículos de la serie:

1. Buenas practicas en git: evitando errores
2. Git: errores de comandos y repositorios
3. Git: errores de commit
4. Git: Errores de fusión
5. Git: Errores con el remoto

Cada artículo cubre diferentes aspectos de Git y ofrece consejos prácticos para evitar errores y optimizar tu flujo de trabajo. Espero que encuentres útil esta secuencia de artículos y que te ayude a mejorar tu experiencia con Git.

¡Disfruta de la lectura!



¿Cuántas veces te has encontrado frente a un mensaje de error en Git que te deja desconcertado y frustrado?

Si eres como la mayoría de los desarrolladores, probablemente más de una vez. Git es una herramienta increíblemente poderosa para el control de versiones, pero también puede ser complicada y propensa a errores si no se utiliza correctamente.

En este artículo, te guiaré a través de los pasos fundamentales para evitar errores en Git y te daré consejos y trucos para que puedas aprovechar al máximo esta herramienta esencial. Ya seas un principiante que acaba de empezar a utilizar Git o un desarrollador experimentado que busca mejorar su fluidez, ¡has llegado al lugar correcto!

### 1. **Lea la documentación de Git**

La documentación de Git es una valiosa fuente de información que te ayudará a utilizar Git de manera efectiva, resolver problemas y mantener un flujo de trabajo eficiente. Leerla te permitirá aprovechar al máximo esta poderosa herramienta de control de versiones.

* La documentación oficial de Git: El sitio web oficial de [Git](https://git-scm.com/doc) proporciona documentación completa y detallada sobre todos los aspectos de Git. Aquí encontrarás información sobre comandos, conceptos, configuraciones, flujos de trabajo y mucho más. Se recomienda empezar por la sección "Git Basics" para obtener una base sólida sobre los conceptos y comandos fundamentales.
* Referencia de comandos: La documentación de Git incluye una referencia completa de todos los comandos disponibles en Git. Puedes consultar la descripción de cada comando, sus opciones y cómo utilizarlo correctamente. Esto es especialmente útil cuando encuentres un comando desconocido o necesites entender en detalle cómo funciona.
* Manuales de Git: La documentación de Git también incluye varios manuales específicos sobre temas particulares. Algunos ejemplos son: [**Pro Git**](https://git-scm.com/book/en/v2), Git Glossary,Git Hooks,Git Attributes,etc.
* Ejemplos y tutoriales: Además de la documentación técnica, el sitio web de Git ofrece una sección de "Git Tutorials" con ejemplos prácticos y tutoriales paso a paso. Estos tutoriales cubren una amplia gama de temas, desde los conceptos básicos hasta flujos de trabajo avanzados, colaboración y más.
* Explorando Git en la línea de comandos: La documentación de Git te enseñará a utilizar los comandos, pero también es útil para explorar Git directamente desde la línea de comandos. Puedes utilizar la opción --help con cualquier comando para obtener información breve sobre su uso y opciones. Por ejemplo, puedes ejecutar git commit --help para obtener información sobre el comando commit.

Recuerda que la documentación de Git se mantiene actualizada y brinda información precisa y confiable. A medida que adquieras experiencia con Git, es una buena práctica consultar la documentación regularmente para ampliar tus conocimientos, descubrir nuevas funciones y resolver cualquier problema que pueda encontrar.

### 2. Verificar el estado del repositorio

Antes de realizar cualquier operación en Git, verifica el estado actual del repositorio utilizando el comando git status. Esto te dará información sobre los cambios realizados, los archivos modificados y las ramas en las que te encuentras.

Cuando ejecutas git status, obtendrás información sobre varios aspectos del repositorio:

* Rama actual: git status te mostrará en qué rama te encuentras actualmente.
* Cambios sin confirmar: Git te mostrará una lista de los archivos modificados que aún no se han confirmado mediante un commit.
* Archivos preparados para el commit :Si has utilizado el comando git add para preparar archivos para el siguiente commit, git status te mostrará una lista de los archivos que están en el área de preparación (staging area).
* Archivos no rastreados: Si hay archivos en tu directorio de trabajo que no están siendo rastreados por Git, git status te mostrará una lista de estos archivos.

Además de proporcionar información sobre el estado actual del repositorio, git status también te dará recomendaciones sobre los comandos que puedes ejecutar para manejar esos cambios. Por ejemplo, te puede sugerir usar git add para agregar archivos modificados al área de preparación, o git checkout para descartar cambios en archivos específicos.

### 3. Actualizar tu repositorio antes de hacer cambios

Antes de comenzar a trabajar en tu repositorio local, se recomienda ejecutar el comando git pull para obtener las últimas actualizaciones del repositorio remoto. Esto evitará conflictos y garantizará que trabajes con la versión más reciente del código.

Cuando ejecutas git pull, ocurren dos operaciones principales:

* Descarga (fetch) de cambios: El comando git pull realiza una operación de descarga (git fetch) para obtener las últimas actualizaciones del repositorio remoto. Esta operación descarga los cambios en tu repositorio local sin fusionarlos automáticamente con tu rama actual.
* Fusión (merge) de cambios: Después de descargar los cambios, git pull realiza una operación de fusión (git merge) para combinar los cambios descargados con tu rama local. Durante esta fusión, Git intentará combinar los cambios automáticamente. En caso de conflictos, donde los cambios descargados y los cambios locales se superponen en el mismo archivo o línea, Git marcará los conflictos y requerirá tu intervención para resolverlos manualmente.

Recuerda que git pull realiza una descarga y una fusión automática, lo que implica cambios en tu repositorio local. Siempre es importante revisar los cambios realizados después de ejecutar git pull para asegurarte de que todo esté en orden antes de continuar con tu trabajo.

### 4. Hacer commits frecuentes y significativos

Realiza commits con frecuencia y asegúrate de que cada commit represente una unidad lógica de cambios significativos. Esto facilitará la comprensión de los cambios realizados y ayudará a deshacerlos si es necesario.

* Claridad y comprensión de los cambios: Hacer commits frecuentes y significativos ayuda a mantener un historial claro y coherente de los cambios realizados en tu repositorio.
* Reversión y deshacer cambios: Si algo sale mal o necesitas deshacer los cambios realizados, los commits frecuentes y significativos facilitan la reversión.
* Colaboración y trabajo en equipo: Hacer commits frecuentes y significativos mejora la colaboración entre los miembros del equipo.
* Claridad en los mensajes de commit: Al hacer commits frecuentes, también es importante proporcionar mensajes de commit claros y descriptivos. Un buen mensaje de commit debe ser conciso pero informativo, explicando de manera sucinta qué cambios se realizarán y por qué.
* Mantenimiento y gestión del código: Los commits frecuentes y significativos facilitan el mantenimiento y la gestión del código a largo plazo.

Los cambios en un commit deben estar relacionados entre sí y tener un propósito común. Si tienes cambios que pertenecen a diferentes aspectos o funcionalidades, considera dividirlos en commits separados para mantener la claridad y la coherencia.

### 5. Utilizar ramas separadas y evitar cambios directos en la rama principal (master/main)

Esta estrategia te ayudará a mantener un flujo de trabajo organizado, facilitando la colaboración con otros desarrolladores y minimizando los riesgos de conflictos y errores.

* Evitar cambios directos a la rama principal: La rama principal, generalmente llamada master o main, es la rama principal de tu repositorio y se considera estable. Evitar cambios directos en esta rama ayuda a garantizar que el código en producción sea confiable y funcional. En su lugar, utiliza ramas separadas para realizar cambios y desarrollar nuevas características.
* Utilizar ramas separadas para el desarrollo: Al utilizar ramas separadas, puedes aislar los cambios y trabajar en ellos de manera independiente. Esto te permite desarrollar nuevas funciones, solucionar problemas o realizar experimentos sin afectar directamente a la rama principal. Además, trabajar en ramas separadas facilita el seguimiento de los cambios y simplifica la colaboración con otros miembros del equipo.
* Ramas con características separadas: Si estás trabajando en una función o característica específica, es recomendable crear una rama separada para su desarrollo. Esto te permite mantener el código relacionado con esa función en una rama dedicada y evitar interferencias con otros cambios en desarrollo. Al finalizar la implementación y probar la nueva función, puedes fusionar la rama de características con la rama principal utilizando una operación de fusión (merge) o rebase.
* Fusiones (merges) y rebase: Una vez que hayas desarrollado y probado tus cambios en una rama separada, es importante incorporarlos a la rama principal de manera adecuada. Puedes utilizar la operación de fusión (merge) o rebase para integrar los cambios en la rama principal.

Al utilizar ramas separadas y realizar fusiones o rebase para incorporar los cambios en la rama principal, puedes mantener un flujo de trabajo organizado y minimizar los riesgos de conflictos y errores. Además, esta práctica facilita la revisión y prueba de los cambios antes de que se integren con la rama principal, lo que contribuye a la calidad y estabilidad del código.

### 6. No combine archivos binarios o generados

Evita incluir archivos binarios o generados automáticamente (por ejemplo, archivos compilados, archivos de dependencias) en tu repositorio. Estos archivos pueden causar conflictos y dificultar la colaboración en el proyecto.

* Conflictos y dificultades en la colaboración: Los archivos binarios o generados automáticamente, como archivos compilados, archivos de dependencias o archivos de construcción específicos de plataformas, tienen la característica de cambiar cada vez que se generan.
* Tamaño y rendimiento del repositorio: Los archivos binarios tienden a tener un tamaño mayor en comparación con los archivos de texto plano.
* Reproducibilidad y mantenibilidad: El propósito de utilizar un sistema de control de versiones como Git es rastrear y administrar el código fuente y los archivos relacionados. Los archivos binarios o generados automáticamente no forman parte del código fuente principal y pueden generarse nuevamente a partir del código fuente y las instrucciones de construcción adecuadas.

Para manejar adecuadamente estos archivos, siga estas practicas:

* Utilizar archivos .gitignore: El archivo .gitignore te permite especificar patrones de nombres de archivos o directorios que Git debe ignorar al realizar operaciones como add o commit.
* Separar archivos generados de archivos fuente: Almacena los archivos generados en un directorio separado y agrega ese directorio al archivo .gitignore.
* Proporcionar instrucciones de construcción: En lugar de incluir los archivos binarios en el repositorio, proporciona instrucciones claras sobre cómo generar esos archivos a partir del código fuente y las dependencias necesarias.

Recuerda que cada proyecto puede tener requisitos específicos, por lo que es importante evaluar el contexto y las necesidades de tu proyecto al decidir qué archivos binarios o generados automáticamente deben excluirse del repositorio.

Si deseas profundizar tus conocimientos sobre Git, te invito a realizar nuestro curso "[Git y GitHub: repositorio, commit y versiones](https://app.aluracursos.com/course/git-github-repositorio-commit-versiones/faq)". Este curso, te brindará una sólida comprensión de los fundamentos de Git y GitHub. Aprenderás habilidades clave para administrar repositorios, realizar commits y manejar versiones de manera efectiva.

Además, te recomiendo que explores más sobre los errores comunes en Git. Este artículo es parte de una secuencia de cinco artículos centrados en los errores en Git:

* [Buenas practicas en git: evitando errores](https://www.aluracursos.com/blog/como-evitar-errores-en-git)
* [Git: errores de comandos y repositorios](https://www.aluracursos.com/blog/git-errores-de-comandos-y-directorios)
* [Git: errores de commit](https://www.aluracursos.com/blog/git-errores-de-commits)
* [Git: Errores de fusión](https://www.aluracursos.com/blog/errores-de-fusion)
* [Git: Errores con el remoto](https://www.aluracursos.com/blog/errores-con-el-remoto?utm_source=gnarus&utm_medium=timeline)

Cada artículo aborda diferentes aspectos de Git y proporciona consejos prácticos para evitar errores. Te animo a profundizar en su lectura para mejorar tu dominio de Git. ¡Disfruta aprendiendo!

CURSO

PRACTICANDO LOGICA DE PROGRAMACION – CHALLENGER

# 01 Qué es un Challenge

[Próxima Actividad](https://app.aluracursos.com/course/challenge-encriptador-texto/task/88292/next)

# ¿Cómo estructuramos el aprendizaje?

Nos basamos en el concepto Challenge Based Learning, es decir, un aprendizaje centrado en desafíos que Apple ayudó a crear y que se basa en 3 pilares:

# **Compromiso:**

Trello y Discord: A través de las tarjetas que recibirás en la herramienta Trello, sentirás en la piel lo que es ser un Dev, recibiendo diferentes tareas, algunas muy detalladas, otras no tanto. Podrás aclarar tus dudas sobre las posibles soluciones del desafío en el Discord exclusivo para estudiantes de Alura.

# **Investigar:**

Cursos y formaciones: Nuestros cursos, formaciones y artículos abren muchas posibilidades para que puedas enfrentar el desafío propuesto. Todos estos contenidos forman una guía detallada, pero recuerda que puedes escoger otros caminos y soluciones a la hora de implementar, como en una empresa de verdad.

# **Actuar:**

GitHub y LinkedIn: ¡Ahora es el momento de escribir el código! Pon tu proyecto on line y enséñaselo al mundo. En esta etapa, podrás compartir tu aprendizaje. Hablar y escribir sobre tus puntos de vista te ayudará a crear una conexión con la comunidad y tus colegas de estudio.

# 02 Preguntas frecuentes

[Próxima Actividad](https://app.aluracursos.com/course/challenge-encriptador-texto/task/88293/next)

# Preguntas Frecuentes

¿Qué son los Challenges?

Es una forma de implementar el Challenge Based Learning o sea, aprendizaje basado en desafíos que Apple ayudó a crear. Es un mecanismo donde podrás comprometerte en la resolución de un problema para después investigar soluciones con cursos, contenido y charlas; ¡O incluso con el conocimiento que ya tienes! Finalmente vas a actuar y colocar tu proyecto en el aire. Todo esto mientras comentas y ayudas en proyectos de otros alumnos y alumnas.

¿Si tengo dudas con quien puedo hablar?

En nuestro grupo oficial de Discord tenemos canales específicos para los Challenges. Allí podrás consultar tus dudas con los instructores e instructoras y con el resto de compañeros. También tendremos Lives de mentoría "Alura te responde" durante los desafíos para consultar las principales dudas.

¿Obtendré un certificado?

Una vez realizada la entrega de tu proyecto, podrás descargar tu certificado y la insignia referente al mismo, que deberás compartir en tus redes profesionales como LinkedIn etiquetando a Alura Latam. Recuerda que los Challenges son parte integral de tu formación y su objetivo es aprender a través de la práctica y crear proyectos increíbles e incluirlos en tu portafolio para demostrar tus habilidades.

¿Qué materiales o herramientas necesito instalar para realizar el curso?

Todas las etapas de los Challenges tendrán especificaciones diferentes que iremos avisando conforme los desafíos sean liberados. Te guiaremos durante todo el proceso para que puedas conseguir finalizar tu proyecto sin preocupaciones.

¿Cómo sabré sobre los nuevos retos?

Enviaremos e-mails de recordatorio cada vez que un nuevo desafío o Live esté próximo a iniciar. Es importante revisar con frecuencia las bandejas de Spam, Promociones y Social en tu e-mail. Algunas veces los e-mails pueden ir allá.

# 03 Importancia de este desafío

[Próxima Actividad](https://app.aluracursos.com/course/challenge-encriptador-texto/task/88294/next)

La lógica de programación es esencial para resolver problemas complejos de forma eficiente. El desafío de crear un encriptador de texto permite aplicar conocimientos teóricos adquirido en los cursos de lógica de programación en un contexto práctico, en el cual desarrollarás habilidades en la manipulación de datos y la implementación de algoritmos. Además, fomenta la creatividad y la innovación, crucial en la era digital. Este proyecto te prepara para enfrentar futuros desafíos y destaca la importancia de la lógica en el desarrollo de software.

# **Error de Git: Tienes ramas divergentes y necesitas especificar cómo reconciliarlas.**

[](https://medium.com/@rajlaxmii?source=post_page-----75e97bd8abd2--------------------------------)

[Rathore de Sakshi](https://medium.com/@rajlaxmii?source=post_page-----75e97bd8abd2--------------------------------)

[](https://medium.com/@rajlaxmii?source=post_page-----75e97bd8abd2--------------------------------)

·

[Seguir](https://medium.com/m/signin?actionUrl=https%3A%2F%2Fmedium.com%2F_%2Fsubscribe%2Fuser%2F10dba85c2979&operation=register&redirect=https%3A%2F%2Fmedium.com%2F@rajlaxmii%2Fgit-error-you-have-divergent-branches-and-need-to-specify-how-to-reconcile-them-75e97bd8abd2&user=Sakshi+Rathore&userId=10dba85c2979&source=post_page-10dba85c2979----75e97bd8abd2---------------------post_header-----------)

2 minutos de lectura

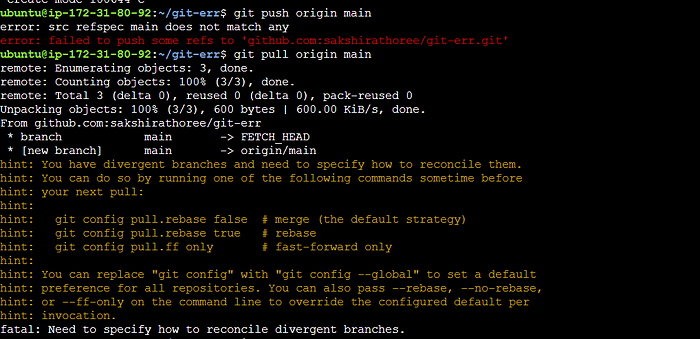
·

14 de agosto de 2023

66

1

Hola estudiantes,  
la mayoría de las veces, mientras trabajan con ramas locales y remotas en git, es posible que se hayan topado con este error al intentar enviar al control remoto (Github) o al extraer del control remoto (Github):



El mensaje que estás viendo indica que estás intentando extraer cambios del repositorio remoto, pero tu rama local y la rama remota han divergido (lo que significa que tienen confirmaciones separadas que no se comparten entre ellas).

Git te pide que especifiques cómo quieres conciliar estas ramas divergentes especificando una estrategia de extracción. Hay algunas opciones entre las que puedes elegir:

**1. Fusionar (estrategia predeterminada):** esto fusionará los cambios remotos en su rama local. Puede usar el siguiente comando para extraer usando la estrategia de fusión:

git pull origen principal

**2. Rebase:** esto volverá a aplicar sus confirmaciones locales sobre las confirmaciones de la rama remota. Esto puede crear un historial más limpio, pero puede requerir la resolución de conflictos. Puede usar el siguiente comando para extraer usando la estrategia de rebase:

git pull --rebase origen principal

**3. Avance rápido (solo ff):** esto solo realizará una extracción si se puede hacer usando una fusión de avance rápido, lo que implica que su rama local está detrás de la rama remota y no tiene confirmaciones adicionales.

git pull --ff-only origen principal

Elija la estrategia que mejor se adapte a su flujo de trabajo y al estado de sus ramas. Si no está seguro, puede hablar con su equipo sobre el enfoque preferido para gestionar las ramas divergentes.

# 02 Figma del desafío

[Próxima Actividad](https://app.aluracursos.com/course/challenge-encriptador-texto/task/88295/next)

# Figma del desafio

Hemos diseñado un modelo en Figma que representa visualmente el Encriptador de texto. Este diseño está pensado para inspirarte y servirte de guía, pero la personalización es fundamental. Te invitamos a modificar y adaptar este modelo a tu estilo único.

* [Challenge ONE Front End - Encriptado de texto | Figma](https://www.figma.com/design/trP3p5nEh7XUyB3n2bomjP/Alura-Challenge---Desafío-1---Lógica?node-id=0-1&t=J3eaw8cZjFaQHPC0-0)

# Copia del Figma

Para realizar copia del figma primeramente debes crear una cuenta gratuita en [www.figma.com](https://www.figma.com/) luego ingresa a este enlace [Challenge ONE Front End - Portafolio | Figma](https://www.figma.com/design/trP3p5nEh7XUyB3n2bomjP/Alura-Challenge---Desafío-1---Lógica?node-id=0-1&t=J3eaw8cZjFaQHPC0-0), ir al menú superior y clicar en Duplicate to your drafts