



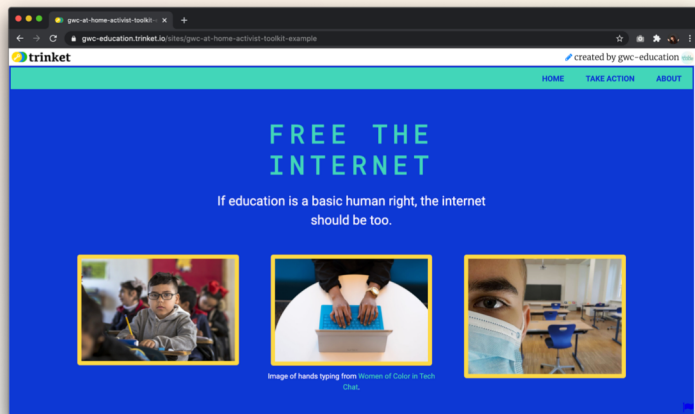
Girls Who Code en casa

Kit de herramientas para activistas, parte 5

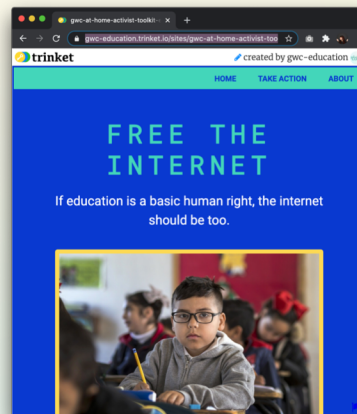
Creación: Hacerlo adaptable

Descripción de la actividad

Tu sitio web se ve genial con el CSS que agregaste en la última actividad. En esta parte de la serie, exploraremos la forma en que puedes comprobar que tu sitio web se ve genial en tamaños diferentes de pantallas, como las pantallas de las computadoras portátiles, tabletas y dispositivos móviles. Esto se denomina diseño adaptable.



Tamaño de pantalla en una computadora portátil

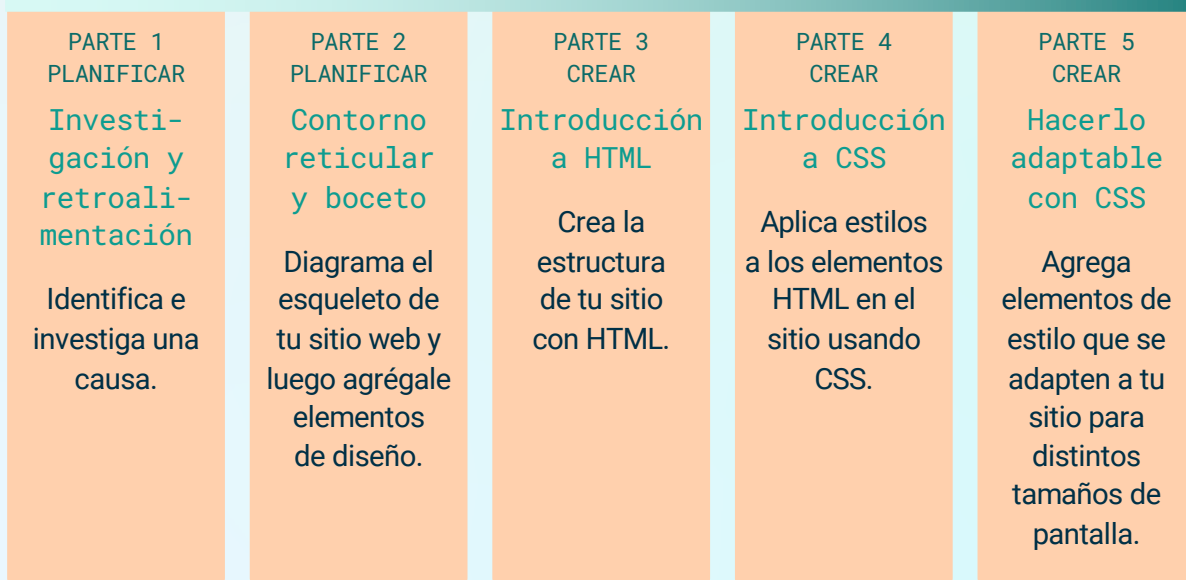


Tamaño de pantalla de un dispositivo móvil

Materiales

- Computadora
- [Trinket](#) o un editor de texto de tu elección
- El código que escribiste en la parte 4
- [Código del proyecto de ejemplo final](#)

*Nota: Si **no** terminaste la parte 4, puedes acceder a las partes 1, 2 y 3 [aquí](#). Si tienes un poco de experiencia con HTML, puedes hacer un remix y utilizar el [código de ejemplo de la parte 4](#) que termina al final de la parte 4.*



Artículo destacado «Mujeres en tecnología»: Teagan Widmer



Fuente de la imagen: [Techies](#)

Cuando Teagan estaba en la escuela de posgrado, se dio cuenta de que safe2pee, un sitio web que ayudaba a los miembros de comunidad LGBTQ a encontrar baños seguros, había dejado de funcionar de manera inesperada. Aunque se había especializado en teatro y no tenía experiencia en programación, decidió asistir a un encuentro de programadores que la llevaría con el tiempo a crear REFUGE Restrooms, un localizador de baños género neutral de código abierto.

Como resultado del éxito de su aplicación, decidió seguir desarrollando software y terminó como facilitadora e ingeniera de pila completa en Future

Advisor, un sitio web que proporciona asesoramiento en inversiones. Hasta hoy, Teagan sigue asesorando a la comunidad trans y manteniendo REFUGE Restrooms, la cual desde entonces ha sido lanzada en dispositivos iOS y Android.

Dale un vistazo a este video de [Social Good Apps Breakfast](#) para escuchar a Teagan explicar qué la motivó a crear su primera pieza tecnológica.

Reflexión

Ser una experta informática significa mucho más que simplemente ser buena programando. Dedica unos minutos a reflexionar sobre cómo Teagan y su trabajo reflejan las características que todos los verdaderos expertos informáticos deben desarrollar en sí mismos: valentía, resistencia, creatividad y propósito.



PROPÓSITO

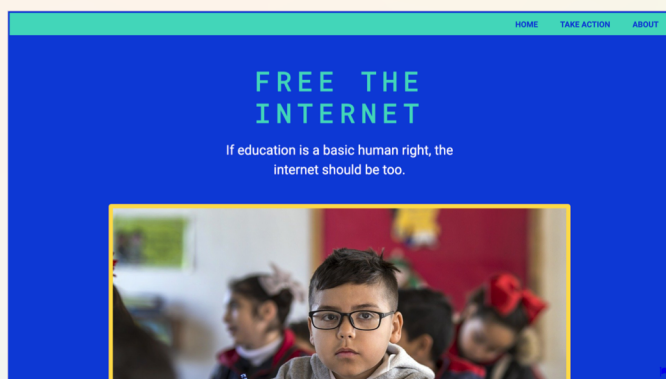
¿Qué ventaja puede obtener el mundo en general de las personas de todas las comunidades que aprenden informática?

Comparte tus respuestas con un familiar o amigo. Anima a otras personas para que lean sobre Teagan y se unan a la charla.

Primer Paso: Aprende acerca de un diseño adaptable (2 a 4 minutos)

La pregunta más importante que queremos responder con esta actividad es: **¿Cómo diseño un sitio web para muchos dispositivos distintos?**

Hasta el momento, tu sitio web solo funciona bien en una pantalla del tamaño de una computadora portátil. Abre tu proyecto de sitio web y compara su aspecto en la ventana de *resultado* más pequeña (a la derecha del editor) con su aspecto en el modo de pantalla completa (a modo de recordatorio, haz clic en **Share** [Compartir] > **Publish** [Publicar] > **Site URL** [Sitio URL]).

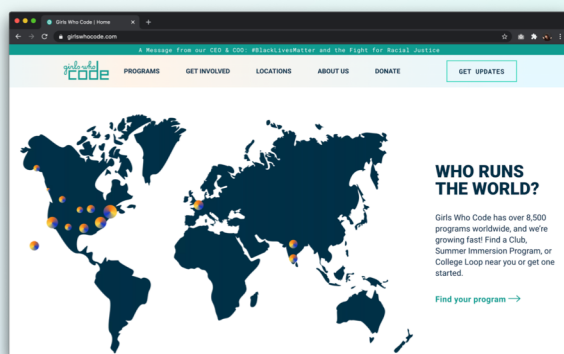


En la pantalla de tamaño más pequeño de la derecha, el texto del encabezado no está alienado correctamente, hay mucho espacio a los lados y las imágenes son muy pequeñas a la vista.

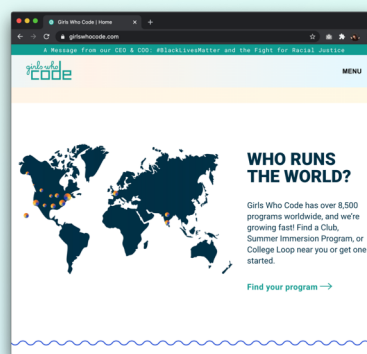
En la pantalla más pequeña, es probable que se estropee el estilo. Esto significa que se rompen las reglas de estilo de tu CSS actual y no se visualiza correctamente. Si miras tu proyecto en un teléfono inteligente, también es probable que se rompan las reglas.

Para diseñar para pantallas de varios tamaños, tenemos que usar un **diseño adaptable**. Un diseño adaptable es una forma de aplicar estilos a tu sitio web para que se adapte al tamaño de la pantalla.

Veamos el sitio web de Girls Who Code como ejemplo:



Computadora portátil
769 px a 1024 px



Tableta
481 px a 768 px



Teléfono móvil
320 px a 480 px

La pantalla de computadora portátil muestra la barra de navegación completa y coloca la imagen del mapa junto al texto del mapa. La pantalla de la tableta condensa la barra de navegación en un menú desplegable y reduce el tamaño de la imagen del mapa. La pantalla del teléfono móvil envuelve con el texto del mapa la parte de arriba y de abajo de la imagen del mapa. Todos estos cambios aumentan la facilidad de uso del sitio web a pantallas de todos los tamaños.

Primer Paso: Aprende acerca de un diseño adaptable (continuación)

Los sitios web adaptables están diseñados de modo tal que los elementos de la página pueden cambiar de tamaño y ajustar su diseño a distintos tamaños de pantalla. Actualmente, más y más personas acceden a sitios web en teléfonos y tabletas, por lo que se ha vuelto cada vez más importante diseñar y crear sitios web que funcionen en varios tamaños de pantalla. Aprenderás dos técnicas que puedes utilizar para hacer adaptable tu sitio: flexbox y consultas de medios. Gracias a ellas, los diseñadores pueden lograr diseños populares de manera más sencilla y con un código más limpio. Lee acerca de ellas a continuación.

FLEXBOX

Flexbox te permite organizar y colocar grupos de elementos visuales en un sitio web de manera más sencilla que el modelo de cajas. Esto es particularmente útil en la disposición de las imágenes.

CONSULTAS DE MEDIOS

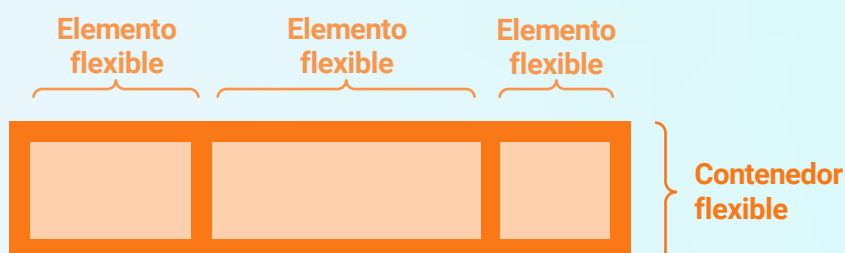
Las consultas de medios te permiten crear conjuntos de reglas CSS que solo se implementan si la pantalla es de un tamaño determinado.

Segundo Paso: Te presentamos a Flexbox (2 a 4 minutos)

En la última actividad, aprendiste que el modelo de cajas es una herramienta que te permite distribuir imágenes en tu sitio. Es excelente para aplicar estilos a algunas partes de tu sitio, pero se vuelve complejo rápidamente si tienes una colección de elementos a los que quieres darles un tamaño y una alineación particular. Flexbox es excelente para esto. Puedes colocar fácilmente un conjunto de tres imágenes en forma horizontal, de izquierda a derecha, o centrar un elemento de manera vertical y horizontal dentro de otro elemento o aplicar estilos a barras de navegación. Dale un vistazo a esta página para conocer [más sobre los usos de flexbox](#).



Flexbox es una forma abreviada para **CSS Flexible Box Layout Module** (Módulo de diseño de cajas flexibles de CSS), y forma parte de la versión actual de CSS. Flexbox permite que los elementos HTML cambien de tamaño y posición en función del tamaño de la pantalla. Estos elementos se encuentran en un contenedor flexible especial que tiene un tamaño definido. El **contenedor flexible** permite que los elementos se encojan, expandan o ajusten en nuevas líneas para llenar el espacio libre que hay dentro del contenedor. Esto significa que ninguno de los elementos se desbordará fuera del contenedor.



Ejemplos de desafíos que ayuda a resolver flexbox:

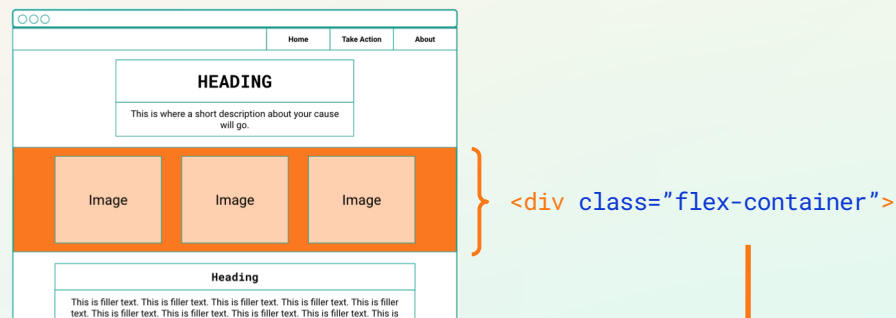


Segundo Paso: Te presentamos a Flexbox (continuación)

Los elementos incluidos en el contenedor flexible se consideran **elementos flexibles**. El estilo de un contenedor flexible afectará a todos los elementos dentro del contenedor, pero puedes añadir propiedades al elemento flexible individual para obtener una mayor personalización de la disposición.

Tercer Paso: Agrega elementos al contenedor flexible (2 minutos)

Para comenzar, tienes que poder definir y seleccionar el contenedor flexible. Puedes hacerlo agregando una etiqueta `<div>` con un atributo de clase a los elementos que quieres distribuir con flexbox. En este caso, los elementos que queremos distribuir con flexbox son las imágenes y cualquier atribución de imagen que corresponda. Si recuerdas la parte 3, agregamos una etiqueta `<div>` a la fila de imágenes en nuestro contorno reticular. Convertiremos esta etiqueta `<div>` en nuestro contenedor flexible.



Ahora, agreguemos esto a nuestro proyecto:

- Ve a tu archivo **index.html**.
- Agrega un atributo de clase a la etiqueta `<div>` que contenga *todas* las imágenes que quieres incluir. Dale un nombre reconocible, como "flex-container".
- Ahora tenemos que crear el conjunto de reglas CSS asociado. Ve a tu archivo **style.css**.
- Busca el comentario `/*Images*/` bajo el comentario `/*HOMEPAGE*/`. Dado que los estilos de nuestro flexbox son para imágenes, lo incluiremos aquí. Si lo deseas, puedes agregar otro comentario para recordarte que estos conjuntos de reglas son para un flexbox.
- En este comentario, utiliza el nombre de la clase que acabas de darle a la etiqueta `<div>` en tu archivo **index.html** para definir el selector de clase de CSS del contenedor flexible. Recuerda que los selectores de clase empiezan con un símbolo `.`

index.html

```
<main>
  <!--Images-->
  <div class="flex-container">
    <div>
      
    <div>
      
      <p class="attribution"> Image of hands typing
    </div>
```

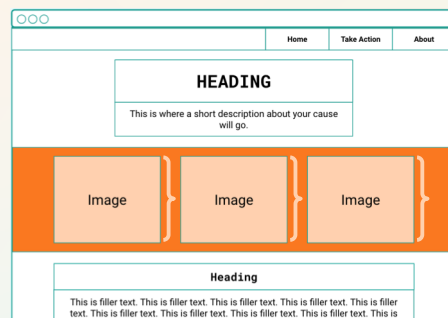
style.css

```
/* Images */
.flex-container {

}
```

Cuarto Paso: Agrega elementos al contenedor flexible (2 minutos)

Ahora que tenemos preparado nuestro contenedor, tenemos que ser capaces de definir y seleccionar los elementos flexibles. Seguiremos un proceso muy parecido al que usamos al crear nuestro contenedor flexible: asigna un atributo de clase a cada etiqueta `<div>` de los elementos que quieres incluir como un elemento flexible individual. En este caso, nuestros elementos flexibles son las imágenes individuales y cualquier atributo de imagen asociado. Dado que ya agregamos etiquetas `<div>` a cada imagen en la parte 3, todo lo que tenemos que hacer es agregar el atributo de clase a la etiqueta `<div>` y crear el correspondiente conjunto de reglas en nuestro archivo CSS.



`<div class="flex-items">`



index.html

```
<main>
<!--Images-->

<div class="flex-container">
  <div class="flex-item">
    
  </div>
  <div class="flex-item">
    
    <!--WOCinTEch asks for image attribution-->
    <p class="attribution"> Image of hands typing
  </div>
  <div class="flex-item">
    
  </div>
</div>
```

style.css

```
/* Images */

.flex-container {
}

.flex-items {
}
```

Ahora, agreguemos esto a nuestro proyecto:

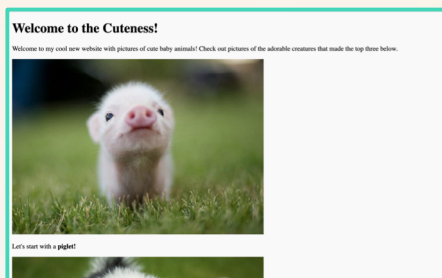
- Ve a **index.html**.
- Agrega un atributo de clase a la primera etiqueta `<div>` que contenga una imagen (y texto, si lo tienes) que quieras incluir como elemento flexible. Dale un nombre reconocible, como `"flex-item"`.
- Agrega un atributo de clase a la segunda etiqueta `<div>` que contenga una imagen (y texto, si lo tienes) que quieras incluir como elemento flexible. Dale el **mismo nombre al atributo de clase** que le diste al primer elemento flexible.
- Repite el último paso para la tercera imagen.
- Ahora tenemos que crear el conjunto de reglas CSS asociado. Ve a tu archivo **style.css**.
- En el conjunto de reglas que creaste para el contenedor flexible, utiliza el nombre de clase que agregaste a las etiquetas `<div>` anteriormente para definir el selector CSS de clase para tus elementos flexibles. Recuerda que los selectores de clase empiezan con un símbolo `.`

En el siguiente paso, exploraremos algunas propiedades clave de flexbox para aplicar estilos que te permitirán activar tu contenedor y los elementos flexibles. Utilizarás estas propiedades en tu propio diseño.

Quinto Paso: Examina las propiedades de estilos de flexbox (10 minutos)

Volvamos a nuestro lindo sitio web de animales de la parte 3 para explorar cómo funcionan las propiedades de estilos de flexbox. Debido a que este sitio es muy sencillo, podremos hacernos una mejor idea de cómo funcionan estas propiedades de flexbox. Después de explorar estas propiedades, las aplicaremos a tu proyecto. En esta sección, no tienes que escribir ningún código, pero puedes hacer un remix del proyecto si lo deseas y usar Tinker para los valores con el fin de ver cómo funcionan.

Para comenzar, veamos [cómo se visualiza actualmente el proyecto](#):



Todos los siguientes elementos están apilados uno sobre el otro: un encabezado, un texto introductorio, una imagen de un cerdito, el subtítulo del cerdito, una imagen de una mofeta bebé, el subtítulo de la mofeta bebé, una imagen de un erizo bebé, el subtítulo del erizo bebé y la imagen del texto de atribución.

Los únicos conjuntos de reglas CSS que ahora tenemos son los que aparecen a continuación: Examinemos esos primero:

```
img {  
  width: 100%;  
}
```

```
.subtitle {  
  text-align: center;  
}
```

Utilizamos la propiedad `width` para definir el ancho de nuestra imagen. Más adelante, haremos que cada imagen se ajuste al ancho total del contenedor de elementos flexibles. En vez de usar píxeles, utilizamos un valor porcentual para que las imágenes se adapten al tamaño de la pantalla.

En nuestro **archivo index.html**, creamos un atributo de clase con el nombre `subtitle` para el texto debajo de nuestras imágenes: `<p class="subtitle"></p>`. En nuestro archivo **style.css**, creamos un conjunto de reglas con el selector de clase `.subtitle` y agregamos la propiedad `text-align` para centrar el texto.

Metas de estilo de flexbox

Antes de examinar las propiedades flexibles, desglosemos nuestras metas de estilo:

- **Meta 1:** Disponer tres lindas imágenes de animales en una fila.
- **Meta 2:** Mantener todas las imágenes en una línea.
- **Meta 3:** Si el tamaño de la pantalla cambia, las imágenes se deben agrandar o encoger de manera proporcional.
- **Meta 4:** Los anchos de las imágenes deben ser iguales.
- **Meta 5:** Debe haber una cantidad pequeña de espacio alrededor de cada imagen de modo que no se amontonen una al lado de la otra.

Quinto Paso: Examina las propiedades de estilos de flexbox (continuación)

Basándonos en lo que hemos aprendido hasta el momento, flexbox es una solución perfecta para este desafío. Para aprender sobre las propiedades de flexbox, agregaremos las propiedades una a una para ver lo que hacen. Comenzaremos con el contenedor flexible.

Propiedades del contenedor flexible

Podemos lograr muchas de nuestras metas con las tres propiedades que están a continuación. Solamente aplicaremos estas propiedades al contenedor flexible, ya que queremos que se apliquen a todos nuestros elementos flexibles; en este caso, los elementos flexibles son nuestras imágenes.

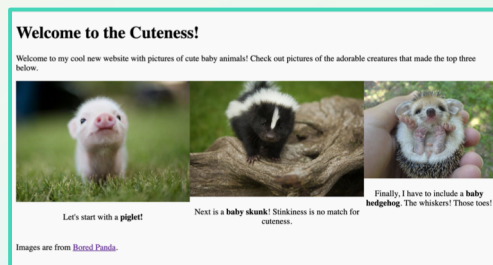
display

La primera propiedad que necesitamos es la propiedad `display`. Para crear un contenedor flexible, tenemos que establecer el valor de `display` a `flex` (flexible). Esto es la magia que convierte nuestra etiqueta `<div>` en un contenedor flexible. Vamos a agregarla al conjunto de reglas del contenedor flexible y veremos lo que pasa.

CSS

```
.flex-container {  
  display: flex;  
}
```

RESULTADO



META REVELADA

Meta 1: Disponer tres lindas imágenes de animales en una fila.

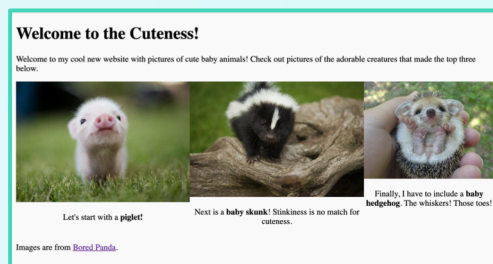
flex-direction

En este momento, nuestras imágenes ya están en una fila. Sin embargo, todavía tenemos que especificarle al navegador que queremos que nuestros elementos flexibles estén en una fila. Para lograrlo, podemos utilizar la propiedad `flex-direction` y establecerla en `row` (fila). La propiedad `flex-direction` le indica al navegador la dirección en que deben ir los elementos flexibles: de izquierda a derecha, de derecha a izquierda, de arriba a abajo, de abajo a arriba. Básicamente, puedes imaginar que le indica al navegador que cree una fila o una columna. A la propiedad `flex-direction` se le pueden asignar los valores `row` (fila), `row-reverse` (fila inversa), `column` (columna) o `column-reverse` (columna inversa).

CSS

```
.flex-container {  
  display: flex;  
  flex-direction: row;  
}
```

RESULTADO



META REVELADA

Meta 1: Disponer tres lindas imágenes de animales en una fila.

Quinto Paso: Examina las propiedades de estilos de flexbox (continuación)

flex-wrap

Por defecto, los elementos flexibles intentarán colocarse en una misma línea. En ocasiones, querrás cambiar esto y permitir que los elementos se ajusten y puedan colocarse en la siguiente línea. Según la meta 2, solo queremos que nuestras imágenes estén en una línea, así que definiremos esa propiedad y estableceremos el valor a nowrap (sin ajuste). Esto significa que las imágenes no estarán en más que una línea. A la propiedad flex-wrap pueden asignarse los valores nowrap (sin ajuste), wrap (ajuste) o wrap-reverse (ajuste inverso).

CSS

```
.flex-container {  
  display: flex;  
  flex-direction: row;  
  flex-wrap: nowrap;  
}
```

RESULTADO



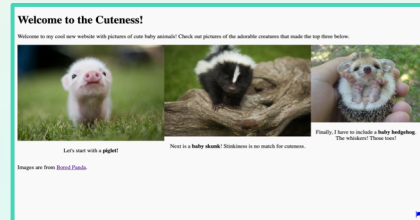
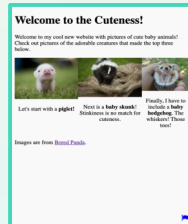
Nota: Al establecer el valor a wrap, los elementos se colocan en otra línea.

META REVELADA

Meta 2: Mantener todas las imágenes en una línea.

Esas son las tres propiedades principales que necesitamos. Antes de cerrar completamente la llave, hagamos una prueba para verificar la **meta 3**: cambiar el tamaño de la imagen de manera proporcional al tamaño de la pantalla.

Arrastra el lado derecho de tu pantalla tan lejos como puedas hacia la izquierda. Deberías ver un cambio en las imágenes a medida que cambia el tamaño de la ventana del navegador.



Propiedades de los elementos de flexbox

También podemos agregar propiedades a los elementos flexibles. Hay propiedades especiales para los elementos flexibles que puedes aplicar, además de las propiedades habituales de CSS que aprendiste en la parte 4. En este proyecto, solo tendremos que utilizar una propiedad especial de los elementos flexibles: flex-basis. También incluiremos una propiedad que ya conoces: margin. Incluiremos ambas propiedades en el conjunto de reglas de .flex-item.

Nota: No tenemos que definir la propiedad display para los elementos flexibles, solamente para los contenedores flexibles.

flex-basis

La propiedad flex-basis establece el tamaño inicial del elemento flexible antes de que el espacio sea distribuido según las reglas del contenedor flexible. Vamos a utilizar esta propiedad para definir el ancho de nuestros elementos. Al utilizar valores porcentuales, las imágenes se agrandarán o encogerán de manera proporcional.

Quinto Paso: Examina las propiedades de estilos de flexbox (continuación)

Dado que establecimos nuestra propiedad `flex-wrap` a `nowrap` (sin ajuste), todas las imágenes aparecen en una línea. Ahora, **no** tienen un ancho igual porque los tamaños de las imágenes son todos distintos, así que tenemos que definir el ancho que queremos que tengan nuestros elementos. Establecemos `flex-basis` a `100%` para indicarle al contenedor flexible que cada elemento que está en su interior debe tener el mismo ancho.

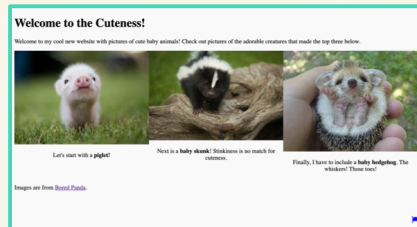


El erizo bebé no tiene el mismo ancho.

CSS

```
.flex-item {  
  flex-basis: 100%;  
}
```

RESULTADO



¡El erizo bebé tiene el mismo ancho!

META REVELADA

Meta 4: Los anchos de las imágenes deben ser iguales.

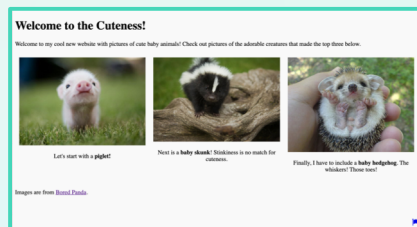
margin

Queremos un poco de espacio alrededor de nuestros elementos flexibles para que las imágenes no se toquen entre ellas. Una solución es utilizar la propiedad `margin` para establecer un margen alrededor de los elementos flexibles. Si establecemos el valor `margin` a `1%`, este es el resultado:

CSS

```
.flex-item {  
  flex-basis: 100%;  
  margin: 1%;  
}
```

RESULTADO



META REVELADA

Meta 5: Debe haber una cantidad pequeña de espacio alrededor de cada imagen de modo que no se amontonen una al lado de la otra.

¡Eso es todo! Logramos todas nuestras metas para el sitio web de nuestros lindos animales. En la sección siguiente, practicaremos la aplicación de estos conceptos en tu proyecto del kit de herramientas para activistas.

VE MAS ALLÁ

Los contenedores flexibles y los elementos flexibles tienen muchas más propiedades, pero estas son las únicas que analizaremos. Flexbox puede parecer confuso al principio, así que prueba las distintas propiedades una a una. Para aprender más sobre las propiedades de flexbox y sus valores, consulta [la guía completa de Flexbox de CSS Tricks](#). También te recomendamos que juegues [Flexbox Froggy](#) para practicar de manera divertida las propiedades de flexbox.

Sexto Paso: Agrega propiedades de flexbox a tu proyecto (5 a 7 minutos)

Ahora que sabemos más sobre las propiedades de los contenedores y elementos flexibles, podemos utilizarlas para aplicar estilos a nuestra fila de imágenes. Agrégalas a tu proyecto del kit de herramientas para activistas.

Agrega propiedades del contenedor flexible

Incluye las siguientes propiedades dentro del conjunto de reglas de tu contenedor flexible. El nombre de nuestro selector de clase es `.flex-container`, pero podrías haber elegido otro nombre. Recuerda: Debe ser el mismo nombre del atributo de clase en tu archivo **index.html**.

CSS

```
.flex-container {  
  display: flex;  
  flex-direction: row;  
  flex-wrap: nowrap;  
}
```

DESCRIPCIÓN

- `.flex-container`: Agrega el selector de clase.
- `display`: Establece esta propiedad a `flex` (flexible) para crear un contenedor flexible.
- `flex-direction`: Muestra los elementos flexibles en una fila o una columna.
- `flex-wrap`: Mantén todos los elementos flexibles en la misma línea.

Agrega propiedades a los elementos flexibles

Incluye las siguientes propiedades dentro del conjunto de reglas de tu elemento flexible. El nombre de nuestro selector de clase es `.flex-item`, pero podrías haber elegido otro nombre. Recuerda: Debe ser el mismo nombre del atributo de clase en tu archivo **index.html**.

CSS

```
.flex-item {  
  flex-basis: 100%;  
  margin: 1%;  
}
```

DESCRIPCIÓN

- `.flex-item`: Agrega el selector de clase.
- `flex-basis`: Establece el tamaño de cada elemento antes de distribuir cualquier espacio adicional. Queremos que todos los elementos se coloquen en la misma línea, así que no queremos espacios adicionales.
- `margin`: Agrega un poco de espacio alrededor de los elementos flexibles. Prueba valores diferentes hasta que quedes conforme con la manera en que se visualiza.

Pruébalo

Veamos cómo se ve:

- Guarda tu proyecto.
- Abre tu proyecto en una nueva pestaña si todavía no lo está. (Para hacerlo, haz clic en **Share** [Compartir] > **Publish** [Publicar] > **Site URL** [Sitio URL]).
- Vuelve a cargar la pestaña con tu sitio web completo para que se actualice con los cambios.
- Arrastra el lado derecho de la ventana de tu navegador hacia la izquierda para que disminuya el tamaño de la pantalla. Deberías ver un cambio en el tamaño de las imágenes a medida que la pantalla se hace más pequeña.

Sexto Paso: Agrega propiedades de flexbox a tu proyecto (continuación)

Si no funciona de la manera en que se supone que debes hacerlo, vuelve atrás y comprueba que:

- No hay errores tipográficos en las propiedades y los valores.
- Hay un punto y coma ; al final de cada propiedad.
- Todas las llaves {} están cerradas.
- El selector CSS de los conjuntos de reglas del contenedor flexible y de los elementos flexibles es el mismo que el atributo de clase <div> en tu archivo **index.html**.

Séptimo Paso: Te presentamos las consultas de medios (2 a 4 minutos)

Si haces memoria, al principio de esta sección definimos un diseño web adaptable como una forma de aplicar estilos a tu sitio web que le permitan cambiar o adaptarse según el tamaño de la pantalla. Flexbox es una herramienta que puedes utilizar en un diseño adaptable, pero en ocasiones es posible que quieras cambiar el estilo de un elemento según el tamaño de la pantalla.

Por ejemplo, si cambias el tamaño de la ventana de tu navegador al mínimo, tal vez observes que se hace bastante difícil ver tu fila de imágenes. Flexbox únicamente nos llevará hasta aquí. También tenemos que utilizar **consultas de medios**

Las consultas de medios son la segunda herramienta que te permite ajustar tu CSS según el tamaño de la pantalla del dispositivo o el tipo de medio. Pueden utilizarse para revisar todo tipo de escenarios posibles:

- Ancho y altura del área visible de una página web (también llamado viewport)
- Ancho y altura de un dispositivo
- Orientación (es decir, horizontal o vertical)
- Resolución de pantalla

Nos centraremos solamente en las consultas de medios de tamaño de pantalla, pero puedes aprender más sobre otros tipos de medios en [W3Schools](https://www.w3schools.com/css/css3_media_queries.asp).

Analicemos a fondo la sintaxis.



@media

Para crear una consulta de medio, utilizaremos las reglas @media para definir el tipo de medio al que queremos aplicar los estilos y las condiciones (como el tamaño de la pantalla) que debe revisar el navegador. Si la condición establecida por esta regla se cumple, el navegador aplica los conjuntos de reglas CSS que hay en el interior de las llaves.

Séptimo Paso: Te presentamos las consultas de medios (continuación)

CSS

```
@media mediatype and (media feature) {  
  /* CSS rule sets */  
}
```

Aquí tienes una explicación de la sintaxis:

- **@media**: La palabra clave para indicar que hay una consulta de medio.
- **mediatype**: Especifica el tipo de medio al que se aplicarán los estilos. Estos valores pueden ser screen (pantalla), print (impresión), speech (habla) o all (todos). Solo trataremos el valor screen (pantalla).
- **and**: Esta palabra clave agrega la primera condición (mediatype) a la segunda condición (media feature).
- **()**: Utilizamos paréntesis para indicar que estableceremos una condición usando una o más media features (características de medios).
- **media feature**: Estas características nos permiten hacer pruebas para ciertas condiciones de la consulta de medio, como si o no la pantalla es más grande que un tamaño dado. Puedes combinar varias características utilizando and (y), or (o) o not (no), etc. Consulta la [lista completa de características de medios](#) de W3School's.
- **{ }**: Todos los conjuntos de reglas que se aplicarán si la condición es verdadera deben estar entre llaves.
- **/* CSS Rule Sets */**: Agrega todos los conjuntos de reglas CSS que quieras aplicar si se cumple la condición. Puedes agregar todas las reglas que desees.

Octavo Paso: Agrega un breakpoint a tu sitio (5 minutos)

Breakpoint, o punto de quiebre en español, es un término que escucharás mucho en un diseño adaptable. Un breakpoint es el punto en que se introduce una consulta de medio para evitar que se interrumpa el estilo. Es probable que hayas visto “romperse” un sitio web cuando no está diseñado para un teléfono móvil. Las imágenes o el tamaño de la fuente se vuelven muy pequeños a la vista, el texto se mueve fuera de un cuadro, un botón se aplasta en un lado, etc.

Intenta cambiar el tamaño de la pantalla de tu navegador para ver cómo cambian los elementos según el tamaño de la pantalla. ¿Cuál crees que es el tamaño de la pantalla cuando se “rompe”?

Aquí tienes algunos breakpoints comunes que puedes utilizar como referencia:

- 320 px a 480 px para dispositivos pequeños (como teléfonos inteligentes)
- 481 px a 768 px para dispositivos medianos (como tabletas)
- 769 px a 1024 px para dispositivos medianos a grandes (como computadoras portátiles)
- 1025 px a 1200 px para dispositivos grandes (como computadoras de escritorio)
- 1201 px y más para dispositivos muy grandes (como televisores)

Octavo Paso: Agrega un breakpoint a tu sitio (continuación)

Agrega un breakpoint

Creemos el primer breakpoint con la regla `@media` para corregirlo. En este momento, si nuestra pantalla es aproximadamente 850 píxeles, los elementos flexibles (por ejemplo, las imágenes) se vuelven muy pequeños. En lugar de estar en una fila en la misma línea, sería mejor que estuvieran en una columna o en varias líneas. Esto se parece a la propiedad `flex-wrap` que aprendimos antes. Podemos hacer que las imágenes se ajusten (es decir, se muevan) a la siguiente línea cuando la pantalla alcanza un determinado tamaño.

Ahora, agreguemos esto a nuestro proyecto:

- Ve al archivo **styles.css**. Busca el bloque de comentario que dice `/* CHANGE CSS BASED ON SCREEN SIZE*/`.
- Debajo del comentario `/*Breakpoint 1*/`, agrega una regla `@media` que verifique si la pantalla tiene un ancho máximo de 850 píxeles.
- Ahora tenemos que agregar los conjuntos de reglas CSS que queremos que cambien cuando se cumpla esta condición. En el interior de las llaves de tu consulta de medio, crea otro conjunto de reglas para el selector de clase de tu contenedor flexible. Nuestro selector es `.flex-container`, pero el tuyo puede ser diferente.
- Agrega la propiedad `flex-wrap` y establece su valor a `wrap` (ajustar).
- Si quieres incluir otros conjuntos de reglas, puedes agregarlos debajo de `.flex-container`.

```
/*Breakpoint 1*/
@media screen and (max-width: 850px){

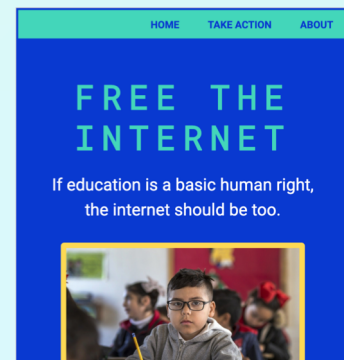
    /*Wrap images onto new lines*/
    .flex-container {
        flex-wrap: wrap;
    }

    /*Add more rule sets if you want*/
    header, main {
        padding-left: 50px;
        padding-right: 50px;
    }
}
```

Esta propiedad le indica al navegador que mueva los elementos flexibles a otra línea si la pantalla tiene menos de 850 píxeles. Hagamos una prueba. Guarda tu proyecto y vuelve a cargarlo o ábrelo en una ventana de tamaño completo del navegador, después arrastra el lado de la ventana del navegador para reducir su tamaño.

¡Funciona! Cuando el ancho de la pantalla llega a 850 píxeles, las imágenes se mueven a otras líneas. Si hay otros cambios que quieras hacer relacionados con el tamaño de la pantalla, sigue el mismo proceso. Puedes hacer todos los cambios que quieras.

Si lo deseas, también puedes agregar otros breakpoints. Si quieres aprender más sobre el diseño web adaptable, dale un vistazo a estos recursos de [Mozilla](#) y [Free Code Camp](#).



Noveno Paso: Extensiones (5 a 15 minutos)

En esta serie de actividades, solo tratamos superficialmente lo que puedes hacer con HTML y CSS. Puedes seguir agregando contenido a tu sitio con HTML o continuar aplicando estilos con selectores y conjuntos de reglas CSS.

Aquí tienes otras ideas que podrías probar:

Extensión 1: Termina de personalizar tu CSS

Si no has terminado de agregar propiedades CSS o quieres probar otras propiedades de estilo, ¡prueba esta extensión!

Extensión 2: Agrega otra página a tu sitio

Puedes agregar una página de recursos para que tu público obtenga más información o una página sobre otro tema relacionado con tu causa.

Extensión 3: Haz que tu sitio web sea más accesible.

Lee sobre los principios de [Web Accessibility Initiative](#) y luego intenta implementar uno o más en tu sitio.

Extensión 4: Agrega más medios

Los medios, como las imágenes, el audio y los videos, pueden ayudarte a comunicar tu mensaje de maneras que no puede el texto. Intenta agregar uno o más de los siguientes:

- Incrustar videos de YouTube con `<iframe>` es una forma divertida de hacer que tu sitio web sea interactivo. Comienza con este [tutorial de W3Schools](#).
- Una [galería de imágenes](#) es una opción excelente si quieres darle a tu público un conjunto de objetos visuales relacionados con tu causa.
- Si reuniste historias de audio y quieres agregarlas a tu sitio web, puedes cargar los archivos a tu carpeta de recursos digitales y luego utilizar la etiqueta `<audio>` para agregarlos a tu sitio. Este [tutorial de W3Schools](#) te ayudará a comenzar.

Una vez que termines tu sitio, ¡compártelo con tu público! Recuerda: La web es una forma poderosa de comunicar un mensaje. La programación es solo una manera con la que puedes actuar a favor de causas que son importantes para ti.

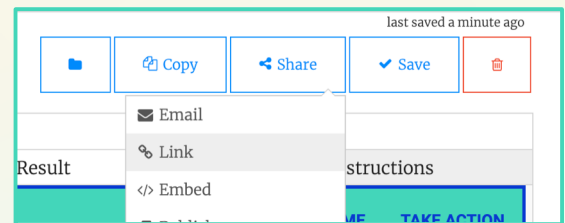
Paso Décimo: ¡Comparte tu proyecto de Girls Who Code en casa! (5 minutos)

Hemos llegado al final de nuestro viaje con el kit de herramientas para activistas. ¡Felicidades!

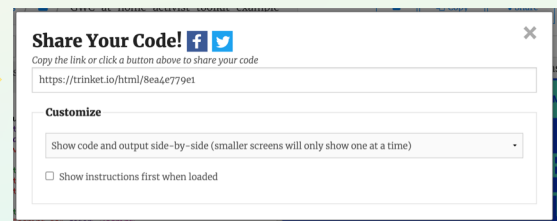
Nos encantaría ver tu trabajo y sabemos que a otros también les gustaría. Comparte tu proyecto final con nosotros. No olvides etiquetar @girlswhocode #codefromhome, ¡y quizá te destaquemos en nuestra cuenta!

Sigue los siguientes pasos para compartir tu proyecto:

- Primero guarda tu proyecto.
- Haz clic en el icono **Share** (Compartir).
- Elige la opción **Link** (Vincular) en el menú desplegable.
- Copia y pega este enlace donde sea que quieras compartirlo.



Enlace al proyecto



¡Espera más proyectos de Girls Who Code en casa!

