

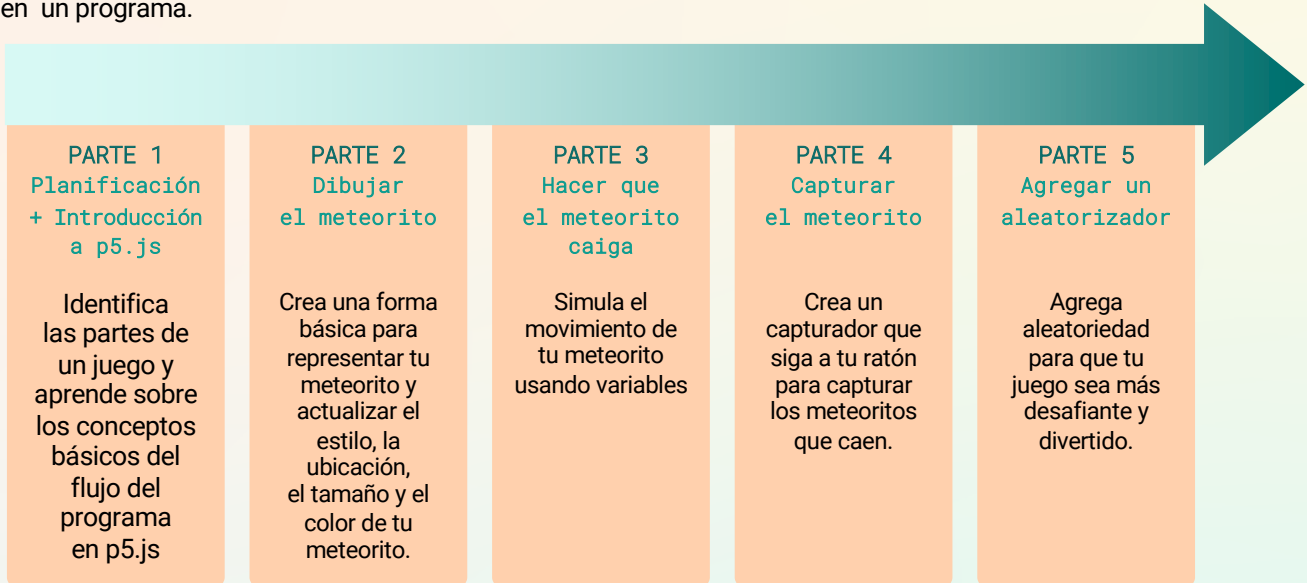


Girls Who Code en casa

Juego Meteor Catcher: Parte 1
Planificación + Introducción a p5.js

Descripción de la actividad

En este proyecto, aprenderás a programar un juego de recolección usando **p5.js**, una biblioteca de JavaScript creada especialmente para artistas y diseñadores. En la **Parte 1** de este proyecto, aprenderás cómo las diferentes partes de un juego funcionan juntas como un sistema y explorarás los conceptos básicos de p5.js. Crearás bordados usando funciones y aprenderás más sobre el flujo del programa o cómo se ejecuta el código en un programa.



Objetivos del aprendizaje

Al finalizar esta actividad, serás capaz de:

- ☐ identificar las partes de un juego y explicar cómo funcionan juntas como un sistema en el programa
- ☐ describir la inspiración detrás de p5.js y navegar por el entorno.
- ☐ describir el flujo del programa utilizando terminología relevante, como sentencias condicionales y flujo de control.

Materiales

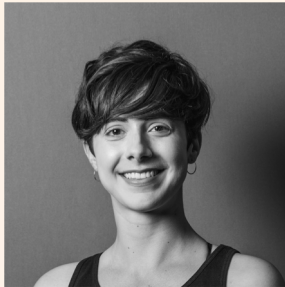
- [Editor en línea p5.js](#)
- [Proyecto de muestra del juego Meteor Catcher](#)
- [Guía de referencia de la parte 1 del juego Meteor Catcher](#)

Conocimientos previos

Antes de comenzar este proyecto, te recomendamos que:

- ☐ puedas explicar con tus propias palabras qué es una [variable](#) y describir cómo pueden usarse en un programa.
- ☐ puedas explicar qué es una [sentencia condicional](#) con tus propias palabras y describir cómo pueden usarse en un programa.

Artículo destacado sobre “Mujeres en tecnología”: Cassie Tarakajian



Fuente de la imagen:
[NYU Tisch](#)

Cassie Tarakajian es desarrolladora de software, ingeniera de hardware, tecnóloga creativa, música y educadora. Cassie también se identifica como [no binaria](#) y usa los pronombres ellos/ellas. Cassie aprendió por primera vez a codificar en la universidad, tomando una clase de [Java](#). Recuerdan aprender solo mediante el uso de un editor de texto muy simple para completar tareas basadas en texto muy simples. Más adelante, se le pidió a Cassie que contribuyera a un proyecto de código abierto llamado [p5.js](#), una biblioteca de JavaScript para hacer arte y sonido interactivos basados en la plataforma de Procesamiento. Como creadora y administradora principal del editor web p5.js, desarrolló el entorno donde las personas podían escribir y ejecutar el código directamente en navegador, lo que facilita el uso para principiantes.

Esta plataforma también es totalmente accesible para los miembros de la comunidad con discapacidad visual, ya que ofrece compatibilidad con lectores de pantalla y vista de alto contraste.

Además del trabajo de Cassie en p5.js, son un ingeniero en [Cycling '74](#), cofundadores de [Girlfriends Lab](#), y Profesores adjuntos en [Tisch School of the Arts at NYU](#). Cassie continúa representando todas estas funciones todos los días para demostrar que la imagen de un programador no se limita a una sola narrativa.

Ve este [video](#) (hasta 6:00) para obtener más información sobre Cassie y su trabajo con p5.js. ¿Deseas obtener más información sobre Cassie y su trabajo? Visita su [sitio web personal](#). Lee este [artículo](#) sobre su trabajo en torno a p5.js y cómo comenzaron.

Reflexión

Ser un experto informático es más que sencillamente ser bueno programando. Tómame unos minutos para reflexionar sobre cómo Cassie y su trabajo reflejan las características que todos los verdaderos expertos informáticos deben desarrollar en sí mismos: valentía, resiliencia, creatividad y propósito.



PROPÓSITO

Al crear p5.js, era importante para Cassie crear un entorno que facilitara el aprendizaje de la programación y que también fuera accesible para una variedad de necesidades. Un gran componente de accesibilidad en p5.js es la compatibilidad con lectores de pantalla y ofrece configuraciones personalizables. ¿Por qué crees que era importante para Cassie crear un editor web “para todos”?

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

Paso 1: Explorar el juego Meteor Catcher (10-15 minutos)

Conoce las partes de un juego (2 minutos)

Todos (o una gran cantidad de) los juegos tienen seis partes: un objetivo, un desafío, una mecánica principal, componentes, reglas y espacio. Estas piezas funcionan juntas como un sistema que genera juego entre las personas involucradas. Como diseñadora de juegos, es esencial que comprenda cómo funcionan juntas todas las partes como un sistema para programar su juego.



- **Objetivo:** ¿Qué tiene que hacer un jugador o equipo para ganar el juego?
- **Desafío:** ¿Qué obstáculos se encuentran en el camino del jugador para alcanzar el objetivo?
- **Mecánica principal:** ¿Qué acciones o movimientos principales hace el jugador para impulsar el juego?
- **Componentes:** ¿Qué partes conforman los materiales de juego?
- **Reglas:** ¿Qué relaciones definen lo que un jugador puede y no puede hacer en un juego?
- **Espacio:** ¿Dónde se lleva a cabo el juego?

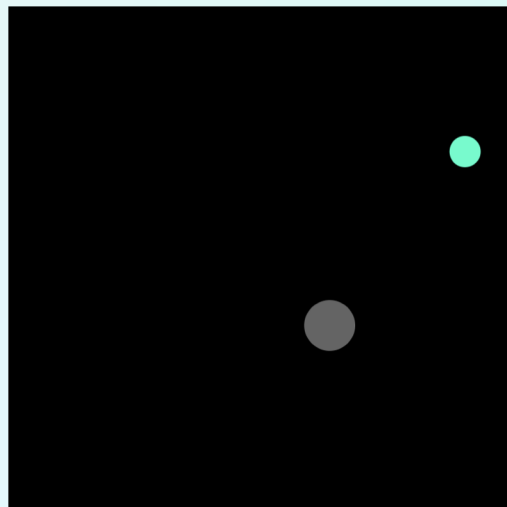
Ejemplo

Ejemplo de Tic Tac Toe

- **Objetivo:** Ser el primero en obtener tres seguidos
- **Desafío:** No sabe dónde colocará su oponente su símbolo
- **Mecánica principal:** Bloqueo y escritura
- **Componentes:** Utensilios de escritura, papel, reproductores, X y O
- **Reglas:** Hay 2 jugadores. Cada jugador alterna los turnos para escribir sus símbolos hasta que la cuadrícula esté llena o hasta que un jugador obtenga tres en una fila, vertical, horizontal o diagonal.
- **Espacio:** Cuadrícula de 3x3 en un papel

Juegue (1 minuto)

La mejor manera de que un diseñador de juegos practique sus habilidades es jugar juegos. Vamos a probar el juego que vamos a construir. Haz clic en este [enlace](#) para jugar el juego durante unos 30 segundos. Mientras juegas, intenta identificar las partes de este juego.



Planificaremos y construiremos un juego sobre la captura de meteoritos para aprender nuevas habilidades de programación, pero en la sección Extensiones tendrás la oportunidad de personalizarla

Paso 1: Explorar el juego Meteor Catcher (cont.)

Identifica las partes de un juego (5 a 10 minutos)

Nuestro objetivo en este paso es comprender cómo funciona el sistema de nuestro juego Meteor Catcher al desglosar los grandes problemas en partes más pequeñas. Este proceso se llama **descomposición**.

Digamos que tu tarea es hacer 100 pizzas para un grupo de niños. Un trabajo enorme, pero si lo dividimos en pasos y subproblemas más pequeños, es mucho más manejable. Por ejemplo, preparar primero toda la masa, cocinar toda la salsa, armar cinco por vez y luego cocinarlas en tandas.



Hay muchas maneras diferentes de dividir un problema complejo en partes más simples. Dado que ya tenemos el juego que jugar, adoptaremos un **enfoque de ingeniería inversa**. Esto significa que en lugar de solo intentar construir el juego desde cero, deconstruiremos el producto terminado para averiguar cómo se hizo. Primero, definiremos todas las partes del juego Meteor Catcher y luego las usaremos para escribir el pseudocódigo. Intenta desglosar las partes del juego Meteor Catcher en el espacio a continuación. No olvides revisar tus ideas con la **Guía de referencia**.

Las partes de un juego Meteor Catcher

- Describir el **objetivo** de Meteor Catcher, el juego que acabas de jugar en el último paso. ¿Qué tiene que hacer un jugador o equipo para ganar el juego?
- Los **componentes** del Meteor Catcher incluyen el meteorito, el capturador, las paredes y el reproductor. Cada componente tiene propiedades únicas (p. ej., tamaño, color, forma, etc.) y acciones (es decir, las cosas que hace, los verbos que asocia con ese componente) que contribuyen al sistema del juego. Por ejemplo, una propiedad del meteorito sería redonda y una acción del meteorito incluiría caer desde la parte superior de la pantalla hacia la parte inferior. *Dedica 2-3 minutos a pensar en las propiedades y acciones de cada componente de la tabla a continuación.*

COMPONENTE	PROPIEDADES	ACCIONES
¿Cuáles son las piezas esenciales para jugar?	¿Cuáles son los atributos o las características del componente? (p. ej., tamaño, color, forma, etc.)	¿Qué hace ? ¿Qué verbos asocias con él?



Paso 1: Explorar el Meteor Catcher (cont.)

- Describir el **espacio** del juego. ¿En dónde ocurre? (Ten en cuenta que a veces el espacio puede ser más de una cosa. Por ejemplo, el ajedrez tiene lugar en el tablero de ajedrez, pero también tiene lugar en una sala de estar, parque o cafetería).
- Definir el **desafío**. ¿Qué obstáculos se encuentran en el camino del jugador para alcanzar el objetivo?
- Describir la mecánica **principal del juego**. ¿Qué acciones o movimientos principales necesita hacer el jugador para jugar el juego? ¿Qué acciones o movimientos principales hace el jugador para impulsar el juego?
- Escribir una lista de las reglas del **juego**. Las reglas determinan lo que podemos y no podemos hacer en nuestro juego. Pueden aplicarse a jugadores, componentes, el espacio, etc.



No olvides revisar sus ideas con la Guía de referencia en la página 3.

Paso 2: Escribir un pseudocódigo para tu juego (5-10 minutos)



En este paso, intente escribir las instrucciones para su programa a un nivel alto en un papel o en la computadora. Esto se denomina **pseudocódigo**. El pseudocódigo es una descripción en lenguaje sencillo de lo que hará su código. Le ayuda a pensar en el flujo y la lógica de su programa para que pueda determinar los pasos que debe seguir para escribir su programa. El pseudocódigo puede tener un aspecto similar a un código sin usar una sintaxis de código específica. Por ejemplo, puedes usarlo u otras palabras clave principales que se aplican a todos los lenguajes de programación. Comienza siempre con el pseudocódigo.

Ya hemos completado algunas cosas para que comiences. También deberás usar las partes del juego de arriba para ayudarte a determinar lo que necesitas incluir. Si te quedas atascado, házte preguntas sobre el juego. Por ejemplo:

- ¿Qué debe suceder al comienzo del juego?
- ¿Qué debe suceder mientras se juega el juego?

Intenta ser específico, pero no te preocupes si no capturas todo o si escribes algo que no sabes cómo hacer. Cada persona escribe el pseudocódigo de manera diferente. Volveremos a este pseudocódigo durante el proyecto, así que asegúrate de guardar tu trabajo.

```
// Pseudocódigo de inicio
Declarar cualquier variable

HÁZLO UNA VEZ
  Establecer el tamaño del lienzo en 400 píxeles por 400 píxeles

HÁZLO EN CADA BUCLE
  Establecer el color de fondo
  // Analizaremos por qué esto vive en draw() frente a setup() más adelante.
  Dibujar el meteorito
  // ¡Intentar agregar el resto por tu cuenta!
```

¿Qué sucede en el juego después de dibujar el meteorito en la pantalla?
Intenta repetir el juego si no puedes recordarlo.

Consejo: Los componentes y las reglas serán especialmente útiles.



No olvides revisar tus ideas con la Guía de referencia en la página 4.

Paso 2: ¡Conoce a p5.js! (10-15 minutos)

P5.js (o solo p5) te permite crear arte interactivo para navegadores web. Es una herramienta para la codificación creativa: proyectos que usan código para la expresión en lugar de solo funcionalidad. P5.js es una biblioteca de JavaScript, un lenguaje de programación que te permite agregar interactividad en la web. Ser una biblioteca significa que p5 es JavaScript, pero los creadores crearon una colección (o biblioteca) de funciones/métodos especializados para que no tengas que hacer todo desde cero. Como está basado en la web, ¡puedes compartir



fácilmente todo tu trabajo! Puedes leer más sobre los orígenes y la comunidad en la [página de inicio de p5](#). Consulta la [página de Presentación](#) para ver algunos ejemplos de proyectos que las personas han realizado con ella.

La p en p5.js significa Procesamiento. El procesamiento es un lenguaje de programación creado para que los artistas y diseñadores integren el código en sus proyectos. El procesamiento se diseñó para que los principiantes creen fácilmente una gama de medios interactivos, desde animaciones hasta visualizaciones de datos, instrumentos musicales, juegos e instalaciones a gran escala. Visita la [página de inicio de Processing Foundation](#) para obtener más información.

Crear tu cuenta (3 a 5 minutos)



Hay dos maneras de usar p5.js: el editor web en línea o un editor de texto y una copia de p5.js que descarga en tu computadora local. La forma más fácil de comenzar con p5 es el editor en línea. Esto te permite escribir código y ejecutar tu programa en un navegador web. En este tutorial, solo utilizaremos el editor web para consultar los pasos e ilustrar ejemplos.

Para comenzar con el editor web, debes crear una cuenta.

- ☐ Visita <https://editor.p5js.org/signup>.
- ☐ **Regístrate.** Completa todos los campos (nombre de usuario, correo electrónico, contraseña y confirmación de contraseña) y luego haz clic en "Regístrate" o puedes elegir iniciar sesión con Google o GitHub.
- ☐ **Confirma tu correo electrónico.** Recibirás un correo electrónico para confirmar y verificar tu cuenta (verifica el correo no deseado si no aparece en 3 a 5 minutos). Haz clic en el enlace, luego inicia sesión con tus nuevas credenciales brillantes (es decir, nombre de usuario y contraseña).
- ☐ **Guarda tus credenciales en un lugar seguro para que puedas iniciar sesión nuevamente.** Si olvidas tu contraseña, ve a la página [Inicio de sesión](#) y haz clic en "Restablecer su contraseña" en la parte inferior.

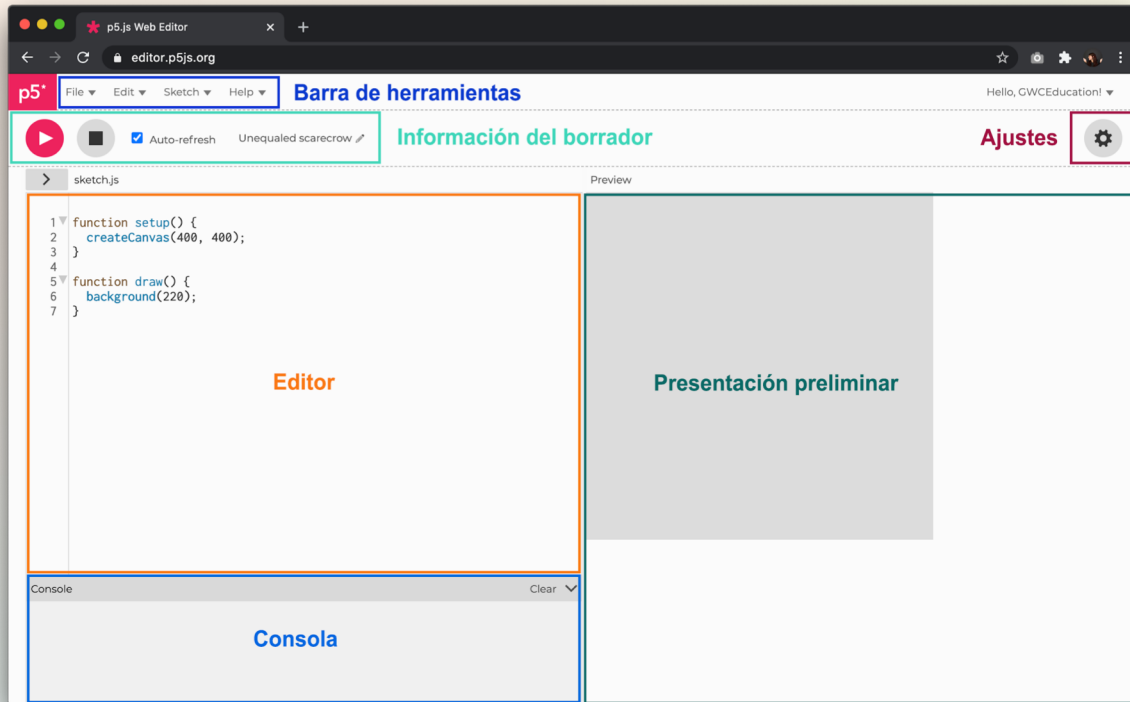
Si tu conexión a Internet es intermitente o prefieres trabajar en un editor localmente, puedes explorar la segunda opción. Consulta esta [página de Introducción](#) para obtener los materiales que necesitarás y las instrucciones sobre cómo descargar la biblioteca. Si necesitas más apoyo, ¡no temas buscar en Google!

Si trabajas sin conexión, los ejemplos pueden parecer diferentes, pero el resultado será el mismo.

Paso 3: ¡Conoce a p5.js! (cont.)

Explora el medio ambiente (5-8 minutos)

Ahora que tienes una cuenta, examinemos la interfaz del editor en línea de la página 5. Este es un entorno de desarrollo integrado o IDE que te permite escribir y ejecutar programas en un solo lugar. Los programas escritos en p5.js se denominan bordados. Puedes pensar en este entorno como un cuaderno de bordados que ya tiene tus herramientas al alcance de tu mano.



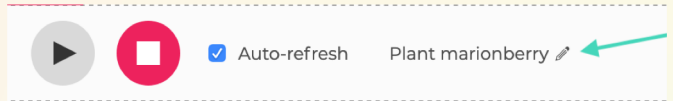
- **Barra de herramientas:** En la parte superior de la página se encuentra la barra de herramientas.
- ◆ En el **menú** Archivo, puedes crear un bordado, guardar un bordado, duplicar un bordado, compartir un bordado en múltiples formatos, descargar archivos de bordado, abrir un bordado y abrir ejemplos. Nota: Algunas de estas opciones no aparecerán hasta que guardes tu bordado.
 - ◆ El menú desplegable **Editar** te permite ordenar tu código, encontrar un personaje o una palabra en tu bordado y navegar por ellos.
 - ◆ En el menú **Bordado**, puedes agregar archivos o carpetas a tu código y ejecutar o parar tu bordado.
 - ◆ Puedes encontrar accesos directos de teclado siempre útiles, un enlace a la página de referencia de p5 y más información sobre p5 en el menú Ayuda.

Ve algunos de los atajos del teclado en p5.js [aquí](#).

Paso 3: ¡Conoce a p5.js! (cont.)

- **Información de bordado:** Debajo de la barra de herramientas hay un botón de reproducir y un botón de parada. El botón Reproducir comienza a ejecutar el programa. El botón de parada detiene el programa. Puedes marcar la casilla “Actualización automática” si deseas que el programa siga funcionando después de hacer cambios en lugar de tener que hacer clic en el botón de reproducción nuevamente.

A la derecha, verás un título completado previamente para tu bordado. Para cambiar el nombre de su bordado, haz clic en el ícono del lápiz y escribe el nuevo título.



- **Configuración:** Puedes acceder a la configuración haciendo clic en el icono de engranaje a la izquierda de la información de Bordado. Aquí puedes cambiar el tema, el tamaño del texto y la configuración de accesibilidad (hablaremos más sobre la accesibilidad en un momento). Te recomendamos que actives el guardado automático en la configuración General.
- **Editor:** El editor es donde escribes tu código. Cada línea tiene un número para que puedas consultarla fácilmente. Las flechas pequeñas junto a un número significan que puedes hacer clic en él para contraer el texto. Por ejemplo, si no necesitas ver comentarios de varias líneas, puedes contraerlos.
- **Vista previa:** Esta ventana muestra los resultados de tu código cuando ejecutas el programa.
- **Consola:** Debajo del editor se encuentra la consola. Esta ventana imprime información sobre tu programa, como mensajes de error o datos a los que deseas acceder en un programa, como el valor de una variable.

Accesibilidad en p5.js (2 min)

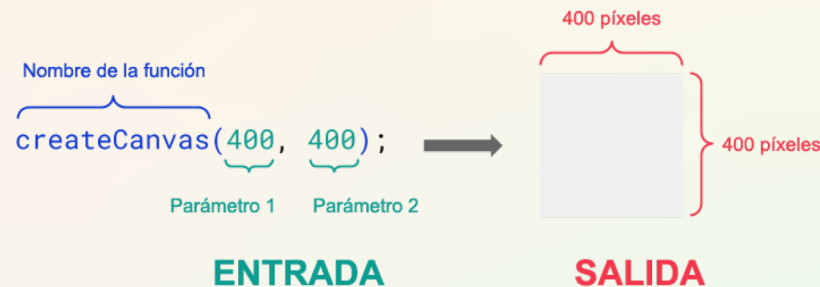


Los desarrolladores de p5 han dado una alta prioridad a hacer que el editor sea accesible para aquellos que tienen problemas visuales. Estas herramientas están en desarrollo activo y forman parte de un proyecto de investigación en curso más grande que se encuentra en NYU. El sitio web del editor en línea y el editor en sí son legibles para los lectores de pantalla. Gran parte del desarrollo de la accesibilidad ha sido para hacer que la salida visual en la ventana de vista previa sea legible por un lector de pantalla. Para obtener más información sobre el uso de esta funcionalidad, consulta [esta página en el sitio web de p5](#).

Mientras continúas aprendiendo a programar en diferentes idiomas y plataformas, siempre debes mantener la accesibilidad y la inclusión a la vanguardia. Históricamente, los diseñadores, ingenieros y programadores no priorizaron a las personas con discapacidades al crear software y hardware. Con el surgimiento del reconocimiento facial y otros software, esto también se aplica a las personas de color, las mujeres y otras comunidades marginadas, ya que los sesgos implícitos de los programadores pueden traducirse en su código. Esto está comenzando a cambiar a medida que aumenta la conciencia, pero todavía queda mucho trabajo por hacer. Tómate el tiempo para asegurarte de que todos puedan usar lo que construyes.

Paso 4: Examinar las funciones de p5.js (5-10 min)

Un **programa** es un conjunto de instrucciones que usted crea para que una computadora las siga. En lugar de escribir las mismas instrucciones una y otra vez, podemos agrupar las instrucciones en partes para poder volver a usarlas más adelante. Estos fragmentos se denominan **funciones**. Las funciones son líneas de código que realizan un conjunto de acciones. Puede pensar en ellas como verbos, *hacen* algo. En p5, damos instrucciones a nuestro programa en la forma de funciones. La mayoría de las funciones que utilizará se definen en la biblioteca p5.js (también puede crear sus propias funciones, pero no cubriremos cómo hacerlo en el tutorial actual).



Cuando llamamos o usamos la función, el programa ejecuta el código dentro de él. Por ejemplo, una de las funciones más importantes es la función `createCanvas()`. Esta función crea el elemento de lienzo que dibuja los gráficos y muestra el bordado. En otras palabras, determina el tamaño de la pantalla. Pero, ¿cómo le decimos a la función qué tamaño de pantalla queremos? Para hacer esto, pasamos **parámetros** a través de la función para obtener la salida que queremos. Los parámetros son valores de entrada que la función utiliza para ejecutar la función. Examinemos la sintaxis de la función `createCanvas()`:

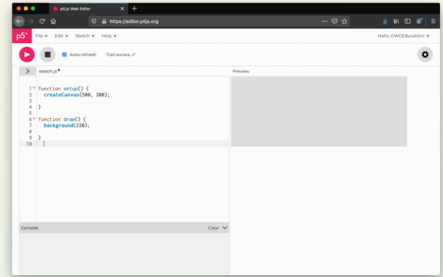
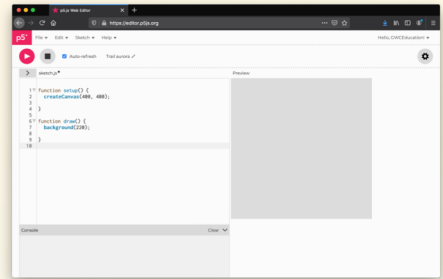
JAVASCRIPT	DESCRIPCIÓN
<code>createCanvas(width, height);</code>	<ul style="list-style-type: none">→ <code>createCanvas</code>: El nombre de la función.→ <code>()</code>: Utilizamos paréntesis para informar a nuestro programa que necesita llamar a la función. A veces incluimos parámetros o entradas en la función dentro de nuestros paréntesis.→ <code>ancho</code>: El primer parámetro que establece el ancho del lienzo en píxeles.→ <code>altura</code>: El segundo parámetro que establece la altura del lienzo en píxeles.→ <code>;</code>: todas las líneas de código en JavaScript deben terminar con un punto y coma.

Paso 4: Examinar las funciones p5.js (cont.)

Los parámetros establecen las dimensiones del lienzo en píxeles. Los píxeles son los componentes gráficos de las pantallas digitales. Cada píxel representa un único punto en la pantalla y tiene un solo color. Deberás incluir esta función en cada bordado de la página 5.

Intenta cambiar los valores de los parámetros para cambiar el tamaño del lienzo:

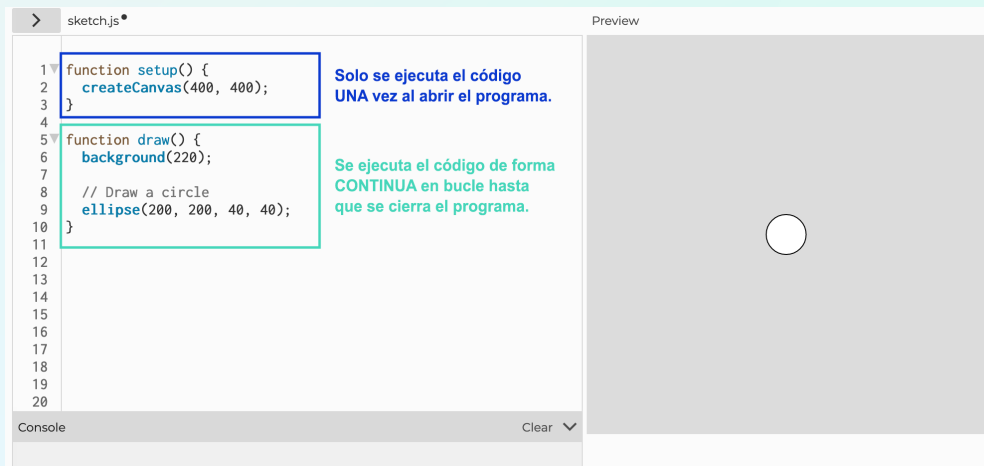
- ☐ **Abre el editor web p5 e inicia sesión.** Quizás hayas notado que el bordado se completó previamente con el código de inicio, incluido nuestro amigo `createCanvas()`. El tamaño predeterminado del lienzo es de 400 píxeles de ancho y 400 píxeles de alto.
- ☐ **Haz clic en el botón Reproducir para ejecutar el código.** Observa el tamaño del lienzo.
- ☐ **Intenta cambiar uno o ambos valores, luego haz clic en el botón Reproducir para ejecutar el código nuevamente.** Marca la casilla de actualización automática para no tener que presionar el botón de reproducción después de cada cambio que realices.



¡Listo! En la ventana de vista previa debe aparecer un cuadro gris con el tamaño de tus parámetros.

Paso 5: Aprender sobre el flujo del programa (5 a 10 minutos)

Sabemos cómo dar las instrucciones de nuestro programa, pero ¿dónde ponemos esas instrucciones? ¿Cuándo corren? ¿Es importante el orden de esas instrucciones? ¿Pueden las funciones entrar en otras funciones? Todas estas preguntas se relacionan con el **flujo del programa**. Esto se refiere al orden en el que el programa ejecuta sus líneas de código. En p5.js, el programa ejecuta cada línea de código en secuencia. Esto significa que ejecuta la primera línea de código, luego la línea 2, luego la línea 3, etc. Más adelante aprenderemos cómo controlar el flujo de su programa con sentencias condicionales, pero primero necesitamos saber sobre las dos funciones principales en p5.js: `setup()` y `draw()`.



Paso 5: Aprender sobre el flujo del programa (cont.)

	DEFINICIÓN <i>¿Qué es?</i>	CONTENIDOS <i>¿Qué debo poner dentro de ella?</i>
setup()	La función <code>setup()</code> se ejecuta solo una vez cuando se inicia su programa. Solo hay una por bordado y no se puede volver a llamar después de la primera vez.	Cualquier función que desees ejecutar inmediatamente cuando se inicie el programa, como el tamaño de la pantalla con <code>createCanvas()</code> , el color de fondo (a veces) y cargar medios como imágenes y fuentes a medida que se inicia el programa. Si creas alguna variable aquí, no podrás acceder a ella en <code>draw()</code> o en otras funciones.
draw()	La función <code>draw()</code> ejecuta las líneas de código contenidas dentro de su bloque continuamente hasta que el programa se detiene. Es el bucle principal y es donde ocurre la acción. Solo hay uno por bordado y se llama después de la función <code>setup()</code> .	Cualquier cosa que desees que suceda repetidamente.

Para comprender mejor las diferencias entre ellos, examinaremos cómo cambia un bordado en función de dónde colocamos la función `background()`. Esto establece el color utilizado para el fondo del lienzo p5.js. Puedes tomar muchos parámetros de valores de color diferentes, incluidos los valores RGB y hexadecimales. Hablaremos más sobre el color en la siguiente sección. Por ahora, solo pasaremos un valor único entre 0 (negro) y 255 (blanco) para un color en escala de grises.

JAVASCRIPT	DESCRIPCIÓN
<code>background(redValue, greenValue, blueValue);</code>	<ul style="list-style-type: none"> → <code>background</code>: El nombre de la función. → <code>()</code>: Utilizamos paréntesis para informar a nuestro programa que necesita llamar a la función. A veces incluimos parámetros o entradas en la función dentro de nuestros paréntesis. → <code>redValue</code>: El valor rojo entre 0 y 255. → <code>blueValue</code>: El valor azul entre 0 y 255. → <code>greenValue</code>: El valor verde entre 0 y 255. → <code>;</code>: Todas las líneas de código en JavaScript deben terminar con un punto y coma.

Colocar la función `background` en el `setup()` o `draw()` de un bordado produce resultados muy diferentes. Considera este código:

```
function setup() {
  createCanvas(400, 400);
}

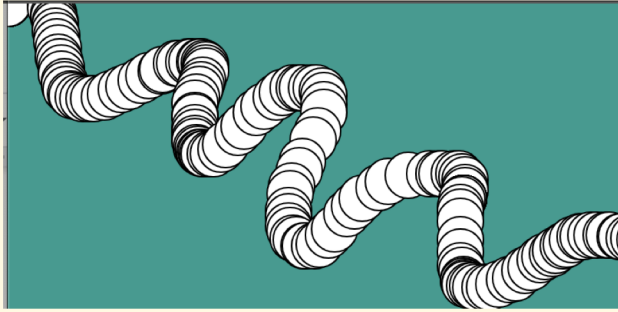
function draw() {
  ellipse(mouseX, mouseY, 50, 50);
}
```

En este momento, este bordado no tiene un fondo. Crea un lienzo y dibuja un círculo o una elipse a la pantalla en la posición del ratón. Ya que la función `ellipse()` está en `draw()`, nuestro programa pinta un nuevo círculo en la pantalla cada vez que el programa pasa por el bucle, casi **60 veces por segundo**, para siempre o hasta que le digamos al programa que se detenga.

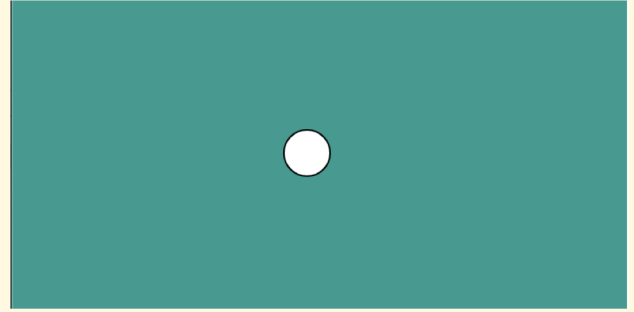
Paso 5: Aprender sobre el flujo del programa (cont.)

Veamos dos bordados de ejemplo para ilustrar el flujo de programa diferente de las funciones `draw()` y `setup()`. Una tiene `background()` en `setup()` y una lo tiene en `draw()`.

BORDADO 1



BORDADO 2



- ☐ Abre el [bordado 1](#) y mueve el ratón sobre el lienzo. El círculo sigue a tu ratón y deja un rastro de otros círculos en tu ruta.
- ☐ Abre el [bordado 2](#) y mueve el ratón sobre el lienzo. Ahora el círculo sigue a tu ratón y no deja un rastro.
- ☐ Piénsalo: ¿Qué bordado tiene la función `background()` en `setup()`? ¿Qué bordado tiene la función `background()` en `draw()`? ¿Por qué?



No olvides revisar tus ideas con la Guía de referencia en la página 5.

Paso 6: Verificar la comprensión (2 minutos)

Tómate un momento para verificar su comprensión del flujo del programa en relación con las funciones `setup()` y `draw()` en p5.js.

Tú decides sorprender a tu amigo preparando su favorito para la cena: una tanda de dumplings. Recuerdas los pasos, pero quieres asegurarte de tenerlos en el orden correcto. Encuentras un programa anterior que escribiste para dumplings, ¡pero está incompleto!

Según lo que aprendiste sobre el flujo del programa, ¿dónde pondrías las siguientes acciones o “funciones” en tu “programa” para que funcione correctamente?

Funciones que necesitas agregar:

- Cerrar envoltorio
- Medir los ingredientes de llenado
- Sacar el dumpling de la bandeja

Programa actual:

```
setup() {  
  Mezclar ingredientes de relleno  
  Recolectar todos los envoltorios de dumplings  
}  
  
draw() {  
  Colocar el relleno en el envoltorio  
  Colocar los dumplings en la bandeja  
  Cocinar los dumplings  
  Comer los dumplings  
}
```



No olvides revisar tus ideas con la Guía de referencia en la página 6.

Paso 7: Compartir su proyecto de Girls Who Code en casa (5 minutos)

Nos encantaría ver tu trabajo y sabemos que a otros también les gustaría. Comparte tu pseudocódigo con nosotros. No olvides etiquetar [@girlswhocode](#) [#codefromhome](#), ¡y quizá la destaquemos en nuestra cuenta!

¡Espera más proyectos de Girls Who Code en casa!

