ESTUDIANTE:			
MODALIDAD:	TÉCNICO EN PROGRAMACIÓN DE SOFTWARE	FECHA:	ENERO 2024
DOCENTES:	MÓNICA CAMARGO OSORIO	**CURSO:**	
OBJETIVO GENERAL:	• Dar una breve introducción sobre la programación de Arduino con creación de circuitos en Tinkercad. • Desarrollar la habilidad de crear un circuito usando la programación de Arduino en Tinkercad.		

¿Qué es un circuito?

Imagina un circuito como un sistema de carreteras para la electricidad. Está formado por componentes electrónicos (como resistencias, transistores, LED, etc.) conectados entre sí por cables. Estos componentes trabajan juntos para realizar una función específica, como encender una luz, medir una temperatura o controlar un motor.

Elementos básicos de un circuito:

- **Fuente de alimentación:** Proporciona la energía necesaria para que el circuito funcione (por ejemplo, una batería o un adaptador).
- **Conductores:** Generalmente, los cables de cobre permiten el flujo de corriente eléctrica entre los componentes.
- **Componentes electrónicos:** Cada componente tiene una función específica (resistencias, limitan la corriente, transistores, amplifican señales, etc.).

¿Cómo se integra la creación de un circuito con la programación de Arduino en Tinkercad?

Con la herramienta de Tinkercad además de diseñar el circuito que se va a realizar, podemos desde ese mismo sitio crear el código del funcionamiento del circuito con la programación de Arduino. En resumen, esta herramienta combina de manera sencilla y visual la creación de circuitos y la escritura de código.

Proceso

Diseño del Circuito:

- **Interfaz intuitiva:** Tinkercad te ofrece una interfaz gráfica donde puedes arrastrar y soltar componentes electrónicos como resistencias, LED, sensores, motores, etc.
- **Conexiones virtuales:** Los componentes se conectan mediante cables virtuales, lo que te permite visualizar de forma clara cómo está estructurado tu circuito.
- **Biblioteca de componentes:** Tinkercad cuenta con una amplia variedad de componentes electrónicos, incluyendo diferentes modelos de Arduino.

Programación de Arduino:

- **Editor de código integrado:** Una vez diseñado el circuito, puedes acceder al editor de código, donde escribirás las instrucciones que Arduino ejecutará.
- **Lenguaje C/C++:** Tinkercad utiliza el lenguaje de programación C/C++, el mismo que se utiliza en el entorno de desarrollo de Arduino.

- **Integración con el circuito:** El código que escribes interactúa directamente con los componentes del circuito que has diseñado. Por ejemplo, puedes encender un LED, leer el valor de un sensor o controlar un motor.

Simulación:

- **Verificación del funcionamiento:** Antes de construir físicamente tu proyecto, puedes simular el circuito en Tinkercad para verificar que funciona como esperas.
- **Visualización en tiempo real:** La simulación te permite ver cómo reaccionan los componentes a tu código.

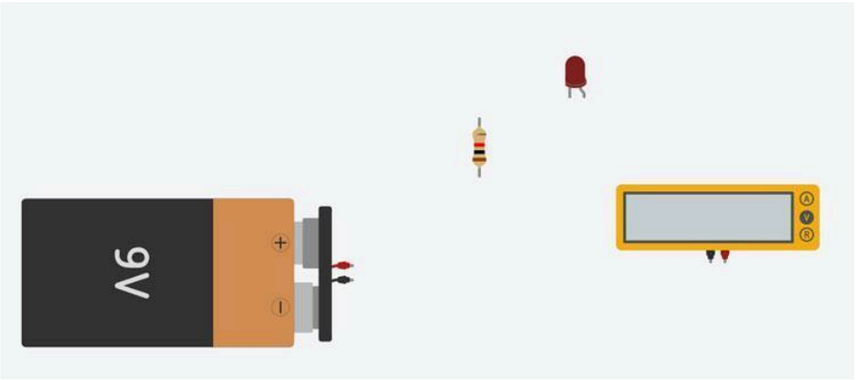
AHORA VAMOS A CREAR UN CIRCUITO

- **Ubicación de los componentes**

Seleccionamos, arrastramos y soltamos una batería de 9 voltios hacia la zona de trabajo.

Podemos girar un componente, para ello seleccionamos la batería (se bordea en color azul) y con la tecla R del teclado la giramos. Al estar seleccionado un componente aparece un cuadro (resaltado dentro de un óvalo en la imagen superior) donde podemos editar algunas de sus características.

Tinkercad ofrece muchos atajos de teclado, si acercas el puntero a un icono, si tiene atajo de teclado, lo mostrará. De igual manera vamos a extraer un LED, un resistor y un multímetro, los acomodamos como aparece en la imagen inferior.



Analicemos algunos detalles.

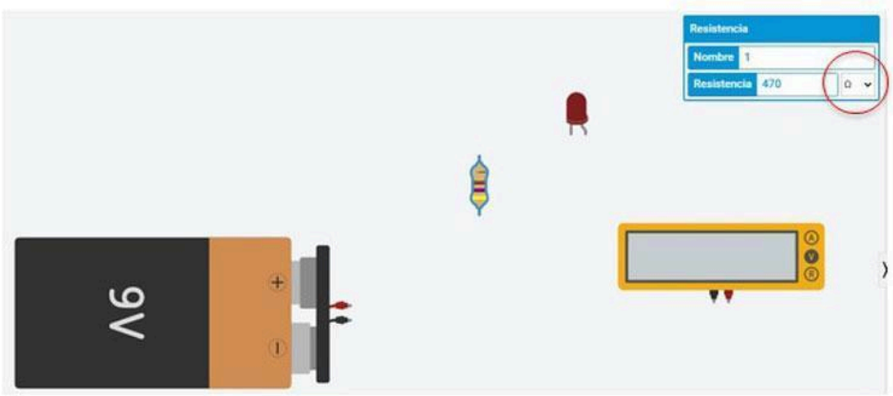
- Debemos cambiar el resistor a 470Ω para lograr una corriente aproximada de 19mA.

- El ánodo del LED (patita más larga) está a la derecha y sería mejor colocarla a la izquierda.
- Necesitamos unir los componentes del circuito eléctrico.

- **Cambio de los parámetros de un componente**

Para cambiar el valor de la resistencia del valor de 1kΩ, que trae por defecto, al que necesitamos, que es 470Ω seleccionamos la resistencia y en el cuadro que aparece ajustamos adecuadamente.

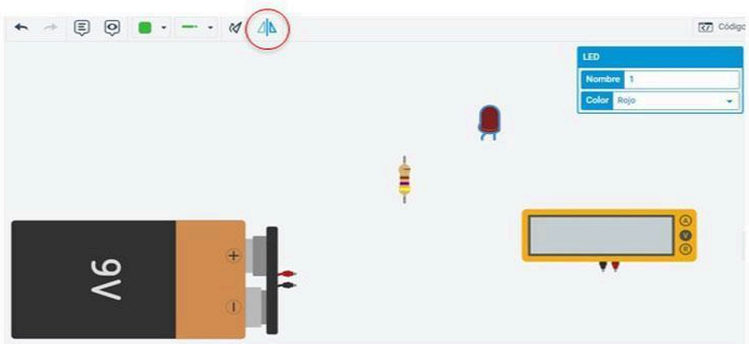
Hay que tener en cuenta la escala de los prefijos decimales que aparece resaltada en círculo rojo, ya que por defecto está en kilo (k).



- **Componente de simetría**

Para invertir la posición del LED y facilitar la colocación de las uniones, usamos el **Componente de simetría** que nos ofrece **Tinkercad**.

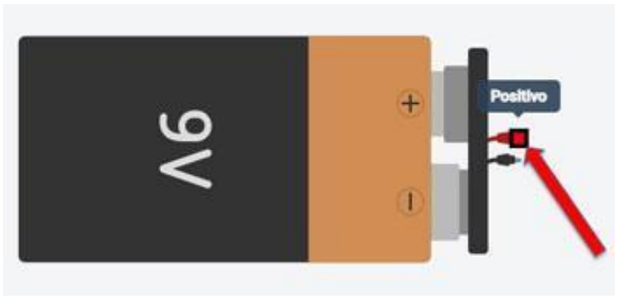
Para activarlo seleccionamos el LED y damos clic en el **Componente de simetría** (encerrado en un círculo rojo) o con el atajo de teclado que es la letra **M**, así se invertirá el LED, ahora tenemos el cátodo a la derecha.



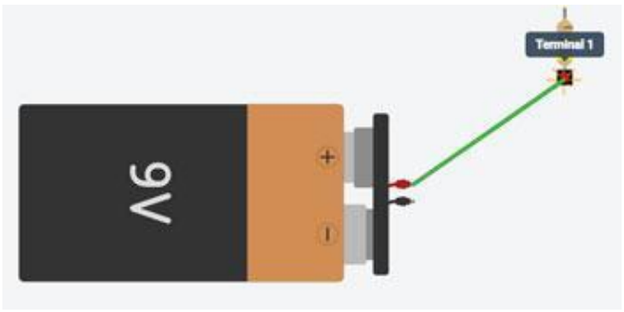
- **Unir los componentes**

Primero el positivo de la batería al extremo inferior de la resistencia.

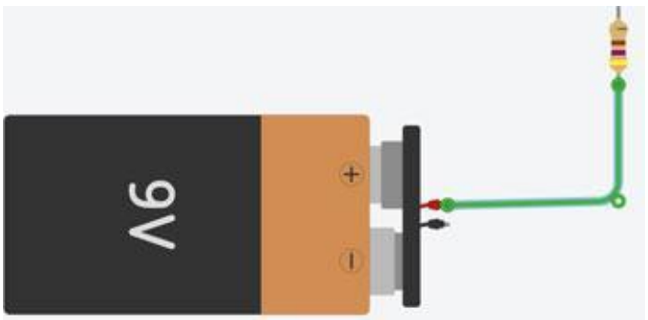
Tan sencillo como arrimar el puntero del ratón al terminal positivo de la batería.



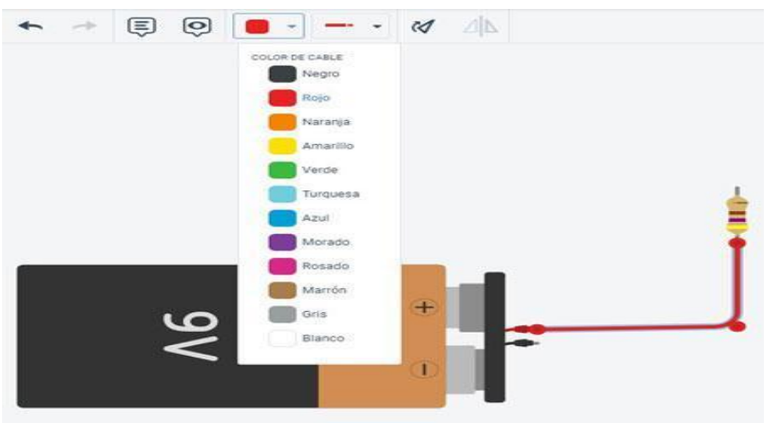
Cuando aparezca un pequeño cuadrado se presiona el botón izquierdo del ratón y se arrastra hasta el punto donde queramos conectar, en este caso el extremo inferior de la resistencia de 470Ω. Aparecerá otro cuadro en la resistencia, hacemos clic izquierdo del ratón y quedarán conectados los componentes.



Cuando se quiera realizar un doblez en un cable, primero lo seleccionamos y con el puntero llevamos el punto central que aparece en el cable hasta donde se desee.

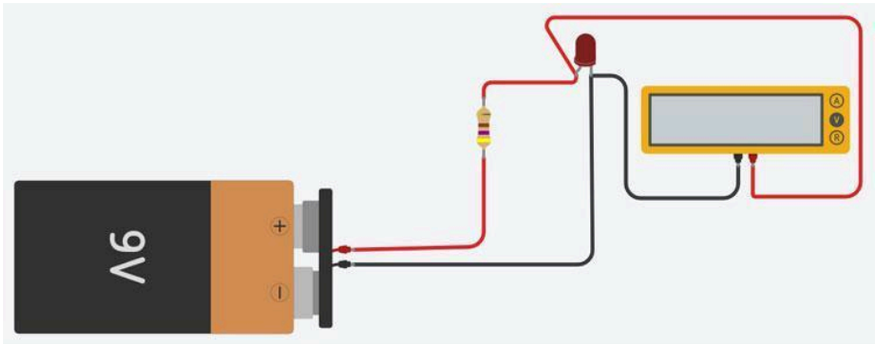


Para cambiar el color del cable, primero lo seleccionamos y con el icono **Color de cable** y su menú desplegable escogemos el color deseado o también con los números del teclado que están encima de las letras; por ejemplo, el número 2 corresponde al color rojo.



Si te equivocas con algún cable o componente, lo seleccionas y con la tecla **Suprimir,** lo eliminas, o con el atajo de teclado **Ctrl + Z** puedes deshacer uno a uno los últimos cambios realizados.

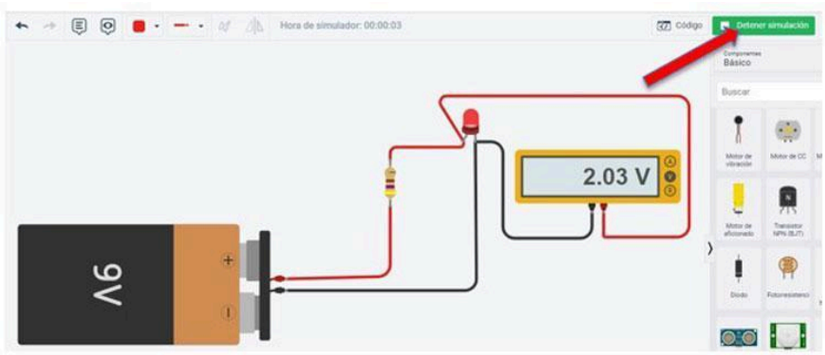
Terminamos nuestro circuito y conectamos el multímetro para medir la tensión en el LED.



Con el multímetro podemos realizar mediciones de corriente, voltaje y resistencia.

- **Iniciar la simulación**

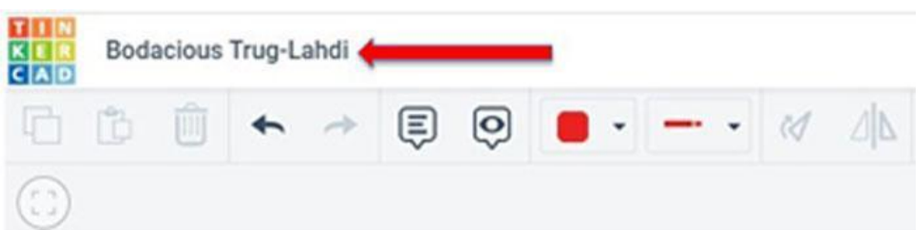
Podemos iniciar y detener la simulación de nuestro circuito con el botón **Iniciar simulación/Detener simulación**. Esta acción empieza a correr una **Hora de simulador** que nos indica horas, minutos y segundos en tiempo real.

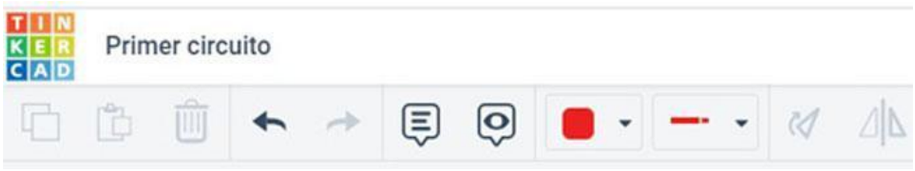


Recuerda que si deseas hacer nuevos cambios en el circuito debes primero detener la simulación

- **Colocando nombre a nuestro circuito**

Tinkercad coloca un nombre aleatorio a nuestros proyectos, pero podemos cambiarlo. Para esto seleccionamos el nombre aleatorio del proyecto, entrará en modo de edición (color azul) y le colocamos el nombre deseado a nuestro circuito.



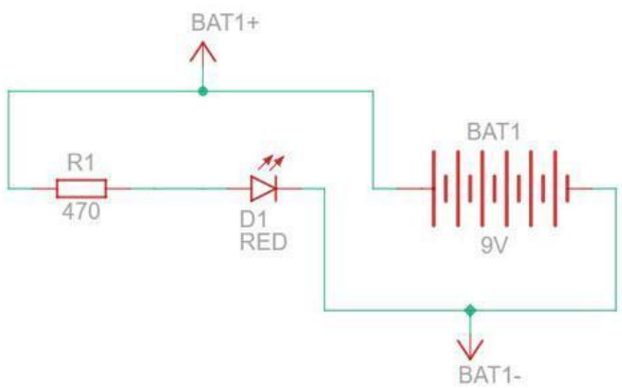


- **Diagrama esquemático**

Hasta ahora hemos estado trabajando en la **Vista de circuito** donde nos ubica **Tinkercad** por defecto

Pero también podemos obtener el diagrama esquemático de nuestros circuitos. Para ello seleccionamos la opción **Vista esquemática**

Y ya tenemos el diagrama esquemático de nuestro circuito. **Tinkercad** nos ofrece la posibilidad de descargar el esquemático en un archivo PDF.



- **Listado de componentes**

Tinkercad nos permite obtener la lista de los componentes que hemos utilizado en el circuito, para esto seleccionamos la opción **Lista de componentes**.

Obtenemos el listado de los componentes y podemos descargar un archivo CSV.

Ya conocemos las bases de funcionamiento de **Tinkercad**, vamos ahora a hacer un ejemplo con Arduino.

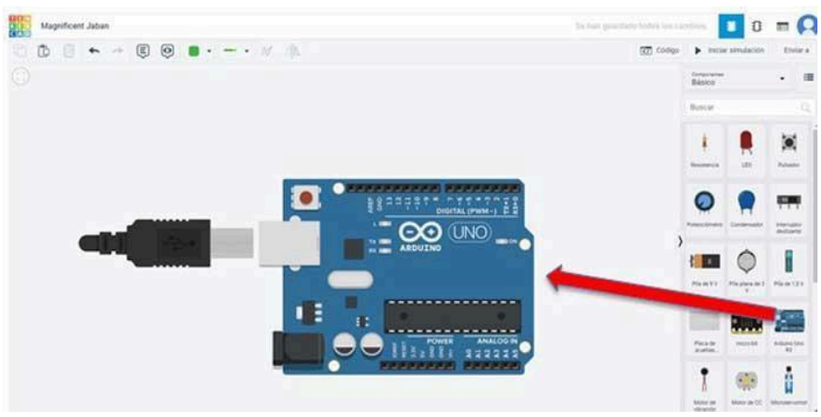
Programar y simular Arduino con Tinkercad

Vamos a realizar el ejemplo que trae **Tinkercad** para usar Arduino con el editor de código de bloques.

- **Editor de código de bloques en Tinkercad**

Creamos un nuevo circuito.

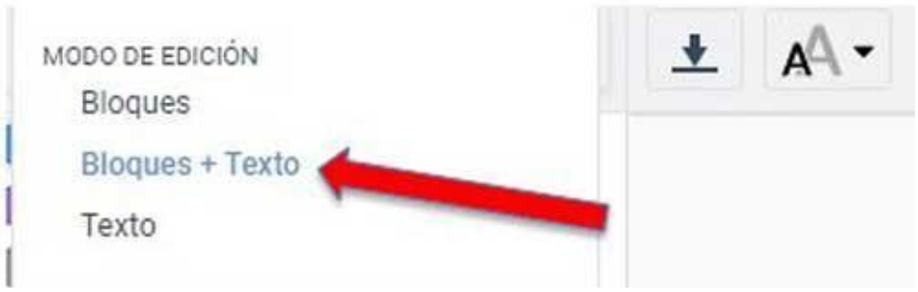
Arrastramos un Arduino Uno de la galería de componentes.



Seleccionamos el botón **Código** y en **Cambiar el modo de edición** damos clic en la flecha del menú desplegable.

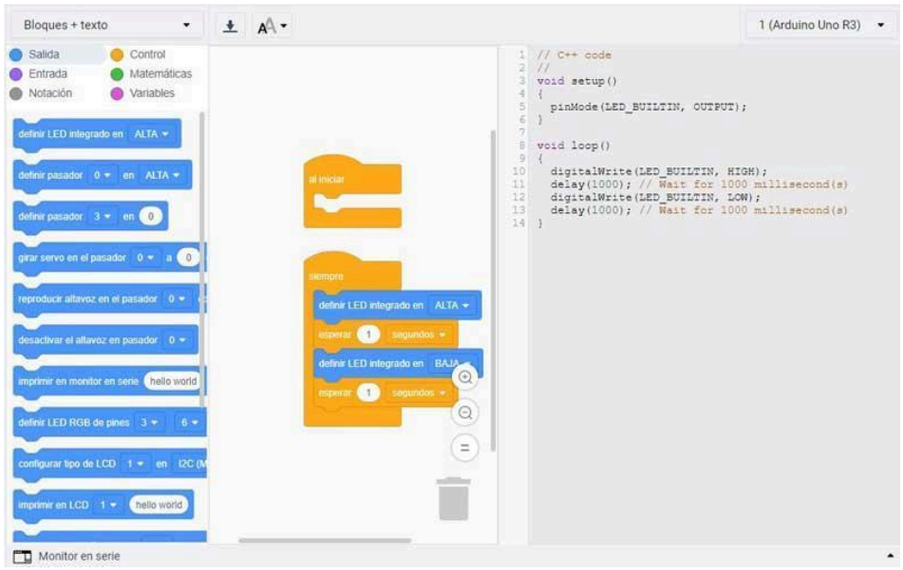
Podemos seleccionar la opción **Bloques**.

Pero esta vez y para comprender mejor cómo funcionan los bloques, es conveniente que podamos ver el código correspondiente también. Así que seleccionamos **Bloques + Texto**.



Es probable que si hemos trabajado un código anteriormente, el asistente de **Tinkercad** nos pregunte si estamos seguros de activar el editor de bloques, seleccionamos **Continuar.**

Podemos observar el editor de **Bloques + Texto** de **Tinkercad**.



Detallamos algunos aspectos del código de bloques y su correspondencia con el editor de texto.

- El bloque llamado **al iniciar** aparece vacío y corresponde a la función *void setup()*. Esto sucede ya que el ejemplo trabaja con el LED integrado de la placa Arduino UNO que es llamado LED-BUILTIG y está conectado al pin 13.
- El bloque denominado **siempre** corresponde a la función *void loop()*.
- El primer bloque azul **definir LED integrado en ALTA** corresponde a la función *digitalWrite (LED_BUILTIN, HIGH)*.
- El primer bloque **esperar (1) segundos** corresponde a la función *delay(1000)*.
- El segundo bloque azul **definir LED integrado en BAJA** corresponde a la función *digitalWrite (LED_BUILTIN, LOW*).
- El segundo bloque **esperar (1) segundos** corresponde a la función *delay(1000)*.

Podemos iniciar la simulación y veremos parpadear el LED de la tarjeta Arduino.

Puedes practicar modificando algunos valores o agregando otros LEDs y resistencias al diagrama pictórico y agregando nuevos bloques al código.

Si has trabajado anteriormente con código de bloques lo encontrarás muy sencillo, de lo contrario puede que te parezca un poco confuso. Afortunadamente, podemos realizar la programación con código de texto.

Vamos ahora a realizar un semáforo.

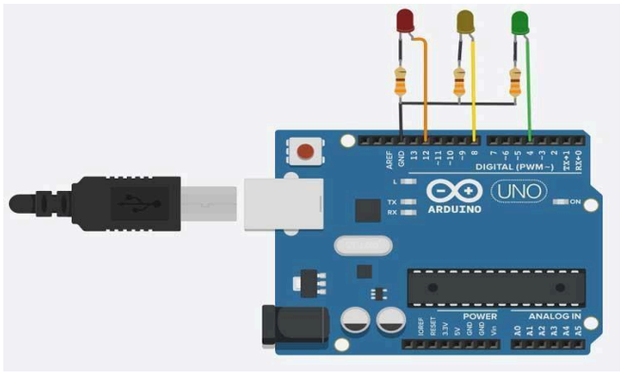
- **Editor de código convencional**

Inicialmente, extraemos los siguientes componentes de la galería.

- Arduino Uno
- Tres LED, uno rojo, uno amarillo y otro verde
- Tres resistencias de 330Ω

En **Tinkercad** puedes usar las opciones tradicionales de Copiar (Control + C) y Pegar (Control + V) cuando necesites duplicar componentes.

Armamos el siguiente diagrama pictórico.



Usaremos este código.

```
// Declaramos las variables que vamos a usar
int rojo = 12;    // Pin del LED rojo
int amarillo = 8; // Pin del LED amarillo
int verde = 4;    // Pin del LED verde

void setup() {
  // Configuramos los pines como salida
  pinMode(rojo, OUTPUT);
  pinMode(amarillo, OUTPUT);
  pinMode(verde, OUTPUT);
}

void loop() {
  // Encendemos el LED verde
  digitalWrite(verde, HIGH);
  // Espera 10 segundos
  delay(10000);

  // Apagamos el LED verde y encendemos el amarillo
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, HIGH);
  // Espera 2 segundos
  delay(2000);

  // Apagamos el LED amarillo y encendemos el rojo
  digitalWrite(amarillo, LOW);
  digitalWrite(rojo, HIGH);
  // Espera 10 segundos
  delay(10000);

  // Apagamos el LED rojo y encendemos el amarillo
  digitalWrite(rojo, LOW);
  digitalWrite(amarillo, HIGH);
  // Espera 2 segundos
  delay(2000);

  // Apagamos el LED amarillo y encendemos el verde
```

```
  digitalWrite(amarillo, LOW);
  digitalWrite(verde, HIGH);
  // Espera 10 segundos
  delay(10000);
}
```

El código es sencillo, primero declaramos las variables para los LEDs rojo, amarillo y verde.

```
// Declaramos las variables que vamos a usar
int rojo = 12;    // Pin del LED rojo
int amarillo = 8; // Pin del LED amarillo
int verde = 4;    // Pin del LED verde
```

En el *void setup()* configuramos los pines de Arduino que comandan los LEDs como salidas.

```
void setup() {
  // Configuramos los pines como salida
  pinMode(rojo, OUTPUT);
  pinMode(amarillo, OUTPUT);
  pinMode(verde, OUTPUT);
}
```

En el *void loop()* hacemos la rutina para el funcionamiento del semáforo.

Damos tiempos de encendido de 10 segundos a los LEDs verde y rojo y de 2 segundos al LED amarillo. Recuerda que la función *delay()* recibe su parámetro en milisegundos.

```
void loop() {
  // Encendemos el LED verde
  digitalWrite(verde, HIGH);
  // Espera 10 segundos
  delay(10000);

  // Apagamos el LED verde y encendemos el amarillo
  digitalWrite(verde, LOW);
  digitalWrite(amarillo, HIGH);
  // Espera 2 segundos
  delay(2000);

  // Apagamos el LED amarillo y encendemos el rojo
  digitalWrite(amarillo, LOW);
  digitalWrite(rojo, HIGH);
  // Espera 10 segundos
  delay(10000);

  // Apagamos el LED rojo y encendemos el amarillo
  digitalWrite(rojo, LOW);
  digitalWrite(amarillo, HIGH);
  // Espera 2 segundos
```

```
  delay(2000);

  // Apagamos el LED amarillo y encendemos el verde
  digitalWrite(amarillo, LOW);
  digitalWrite(verde, HIGH);
  // Espera 10 segundos
  delay(10000);
}
```

En **Tinkercad** seleccionamos el botón **Código** y luego **Texto**.

El asistente nos preguntará si estamos seguros de cerrar el editor de bloques, seleccionamos **Continuar.**

```
// C++ code
//
void setup()
{
  pinMode(LED_BUILTIN, OUTPUT);
}

void loop()
{
  digitalWrite(LED_BUILTIN, HIGH);
  delay(1000); // Wait for 1000 millisecond(s)
  digitalWrite(LED_BUILTIN, LOW);
  delay(1000); // Wait for 1000 millisecond(s)
}
```

Aparecerá por defecto el código del LED parpadeando (*Blink*), el cual seleccionaremos en su totalidad y lo borraremos.

Ahora tenemos dos opciones para crear el código.

- Creamos el código directamente en **Tinkercad**.
- Creamos el código en el IDE de Arduino y luego lo copiamos y pegamos en **Tinkercad**.

Tú decides si te sientes a gusto trabajando directamente en **Tinkercad** o no.

Recuerda que las palabras reservadas del Arduino son en idioma inglés, si tratas de traducir la página de **Tinkercad** al español es probable que el software no funcione bien.

Ya podemos iniciar la simulación de nuestro semáforo.

ACTIVIDAD TEÓRICA

1. ¿Qué es Tinkercad y para qué se utiliza?

_____
_____
_____
_____
_____
_____

2. ¿Cómo funciona la interfaz de Tinkercad para circuitos?

_____
_____
_____
_____
_____
_____

3. ¿Cómo se crea un circuito básico en Tinkercad?

_____
_____
_____
_____
_____
_____

4. ¿Cómo se utiliza Tinkercad para programar Arduino?

_____
_____
_____
_____
_____

5. ¿Qué es una librería en Tinkercad y cómo se utiliza?

_____
_____
_____
_____
_____

ACTIVIDAD PRÁCTICA

1. Crea un circuito con un LED que parpadee a diferentes velocidades.

2. Construye un botón que encienda y apague un LED.

3. Diseño 3D:
   - Modela un objeto simple como un cubo o una esfera.
   - Personaliza un objeto existente de la biblioteca de Tinkercad.
   - Crea un diseño paramétrico que puedas modificar fácilmente.