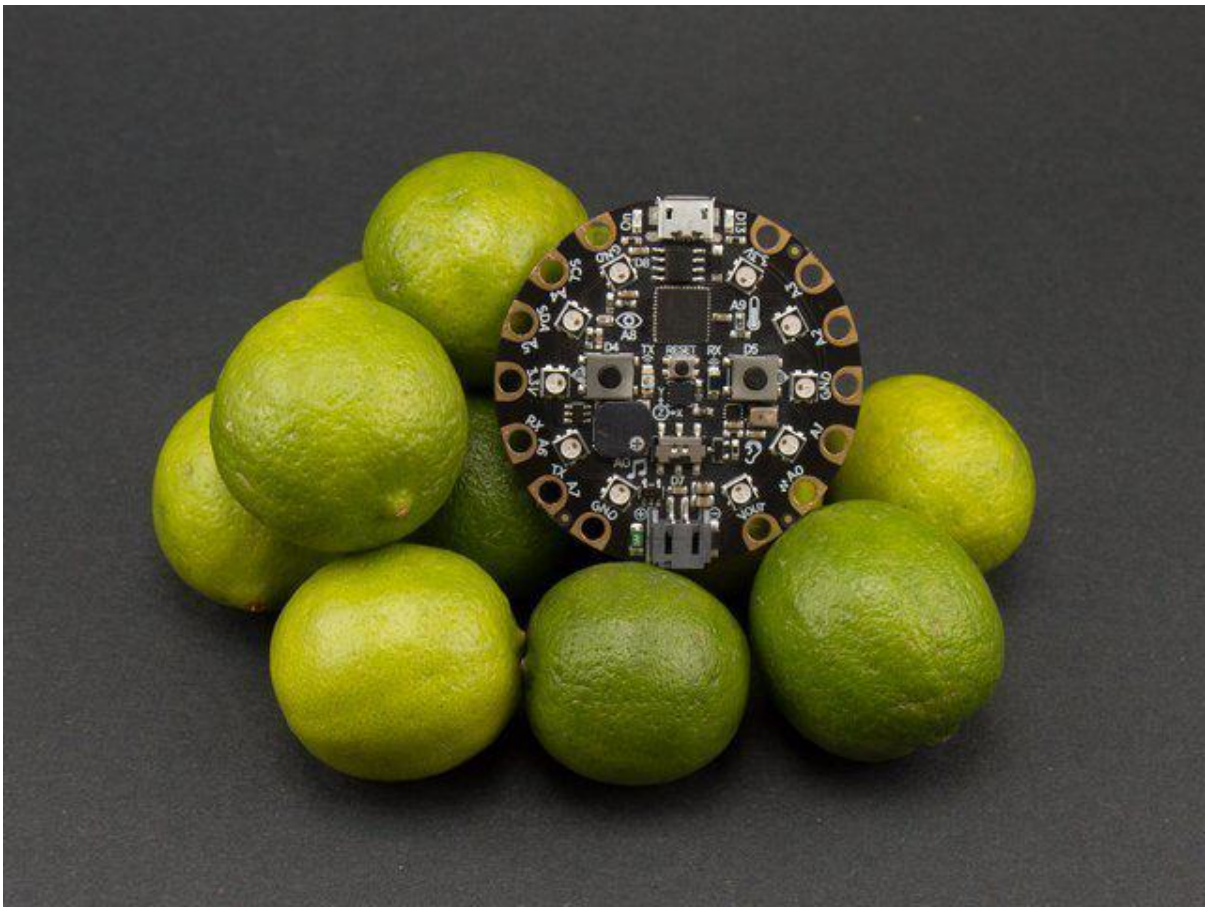
Circuit Playground Express: Piano de Limones

Created by Kattni Rembor

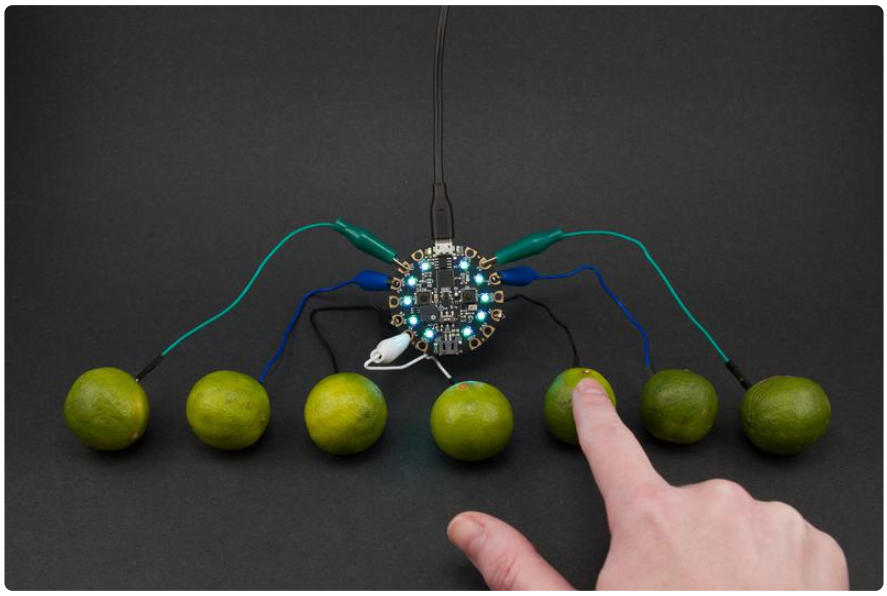


https://learn.adafruit.com/circuit-playground-express-piano-de-limones

Last updated on 2023-08-29 04:29:01 PM EDT

Table of Contents

Introducción



Esta guía de va a introducir a la tarjeta Circuit Playground Express (o CPX para abreviar) y CircuitPython. La Circuit Playground Express es una fantástica tarjetita llena de todo tipo de sensores, interruptores y luces. CircuitPython es un derivado también open-source, de MicroPython, diseñado específicamente para usar con microcontroladoras de Adafruit como la CPX.

Este proyecto te va a enseñar como usar CircuitPython para manipular y usar los pads de toque capacitivo, los Neopixeles, el interruptor deslizante y el parlante integrado. ¡Luego los vas a combinar para crear un piano con tonos que prenden según los colores del arcoiris!

Nota del traductor

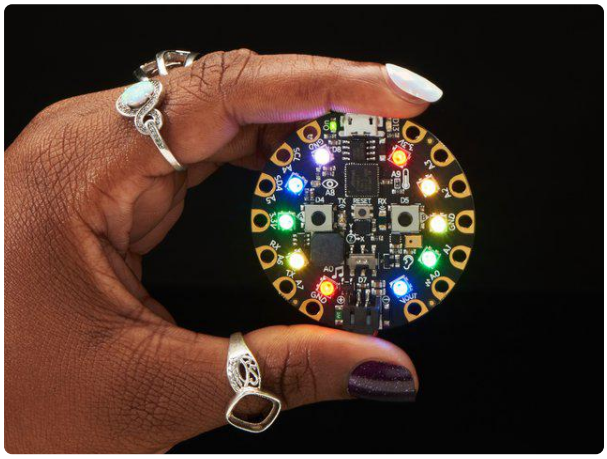
La autora de la guía, Kattni Rembor, no solo hizo un interesante proyecto sino que también le puso un nombre en inglés, que tiene un doble sentido que por más que traté, no he logrado traducir de forma que se mantenga dicho doble sentido. Así que no queda otra, que explicarlo.

El nombre original: "Piano in the Key of Lime", "key" sería la llave , y "of lime" sería porque está construido con frutas, en este caso limas que son similares a limones.

Partes requeridas

Este proyecto va a utilizar la CircuitPlaygroundExpress, conectada a una computadora por medio de un cable micro USB. Los clips de lagarto son utilizados para conectarse

a los pads de toque capacitivo, y cables tipo jumper para extender el tamaño de los cables.



Circuit Playground Express
Circuit Playground Express is the next step towards a perfect introduction to electronics and programming. We've taken the original Circuit Playground Classic and...
https://www.adafruit.com/product/3333

Small Alligator Clip to Male Jumper Wire Bundle - 12 Pieces
For bread-boarding with unusual non-header-friendly surfaces, these cables will be your best friends! No longer will you have long strands of alligator clips that are grabbing little...
https://www.adafruit.com/product/3255

USB cable - USB A to Micro-B
This here is your standard A to micro-B USB cable, for USB 1.1 or 2.0. Perfect for connecting a PC to your Metro, Feather, Raspberry Pi or other dev-board or...
https://www.adafruit.com/product/592

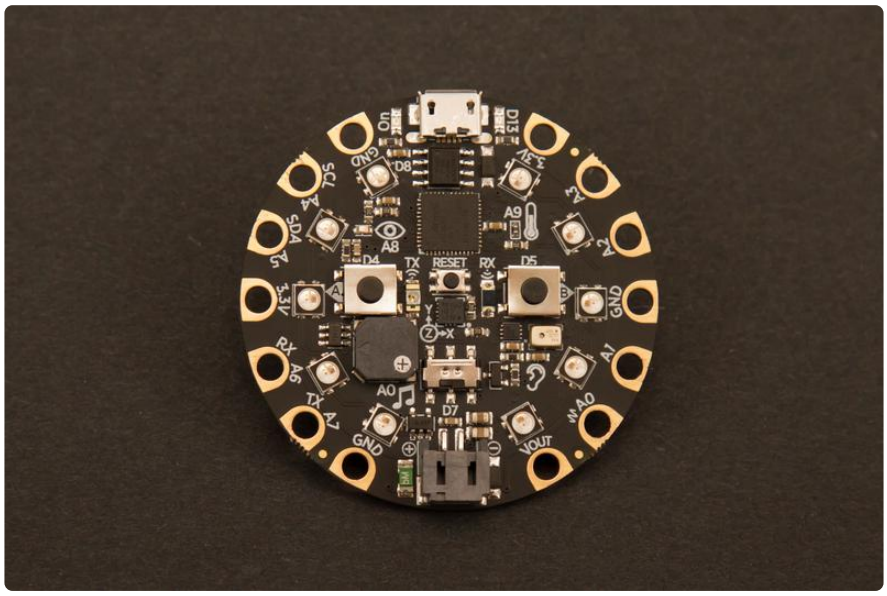
Y el toque final: ¡teclas de limones! Las limas son más pequeños que los limones regulares, y hacen unas excelentes teclas para un piano.

Las secciones a continuación te van a introducir a la CPX y a CircuitPython. ¡Comencemos!

Conociendo a la Circuit Playground Express

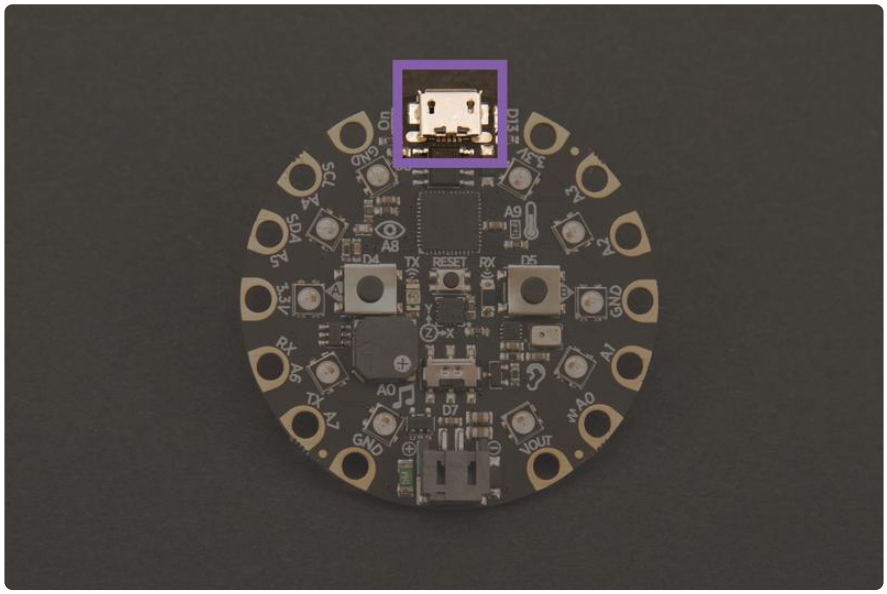
La Circuit Playground Express es una tarjeta microcontroladora que viene llena de cosas bonitas. Puedes ver esta Guided Tour () para más detalles. Se diseñó para que trabaje con 3 lenguajes de programación diferentes. ¡Hay mucho que aprender de esta tarjeta!

El código para este proyecto va a ser escrito en CircuitPython. En la CPX, utilizamos el puerto micro USB, los sensores de toque capacitivo, el interruptor deslizante, los 10 Neopixeles y el parlante integrado.



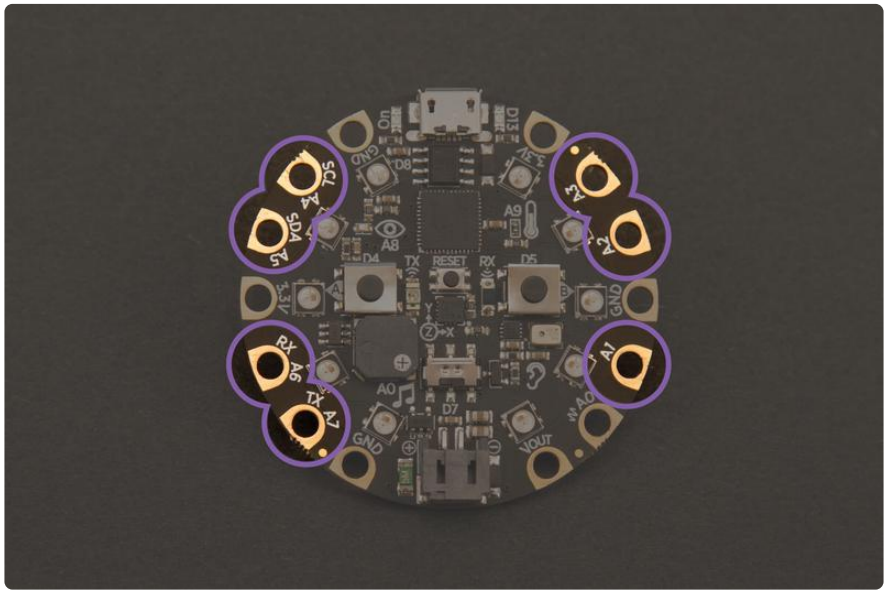
Alimentando con Micro USB

Conecte su cable micro USB al puerto micro USB de su CPX. Este puerto se encuentra arriba en la tarjeta, a la derecha del LED indicador de alimentación.



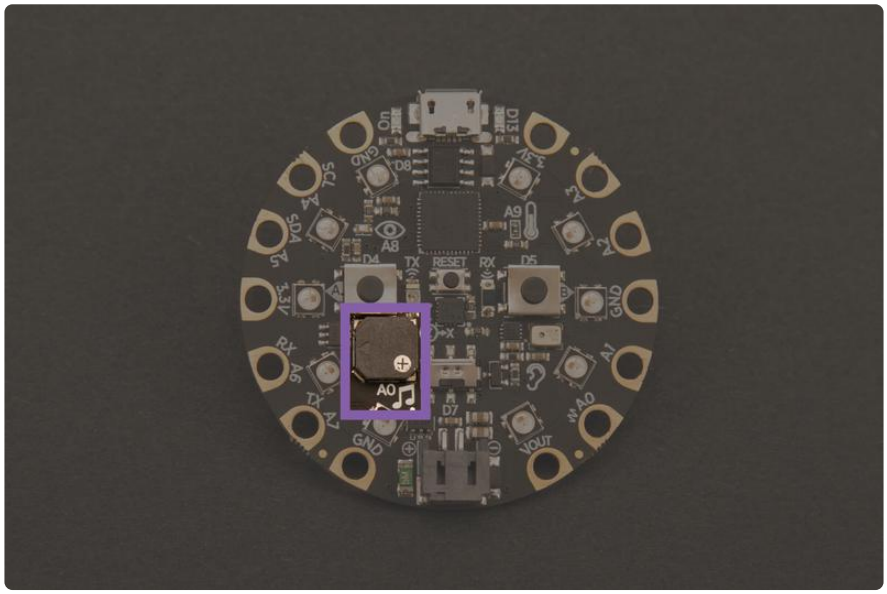
Pads de Toque Capacitivo

Alrededor del borde de la CPX, hay catorce pads amigables a conectores de lagarto. Dentro de estos catorce, hay siete pads que tienen función de toque capacitivo, etiquetados A1-A7. Vamos a usarlos para generar luces y tonos.



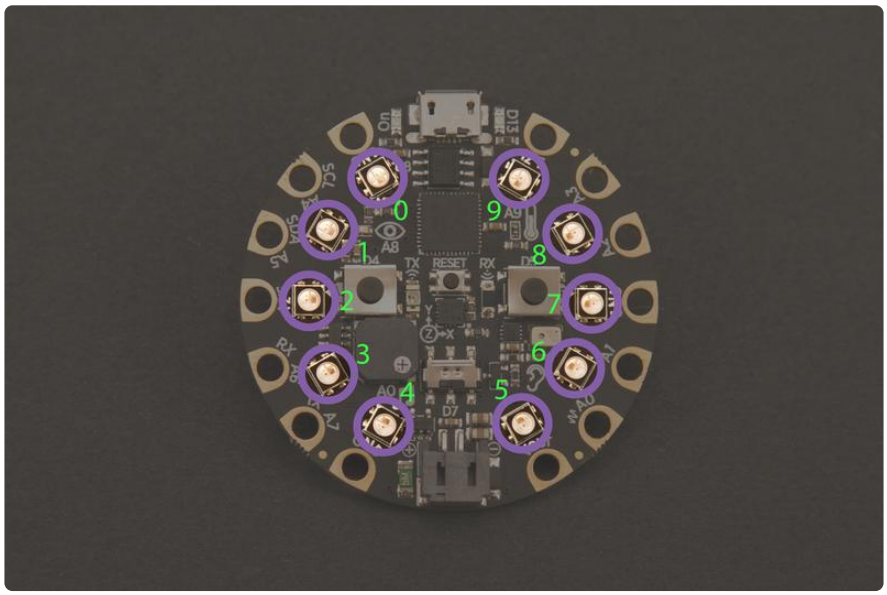
Parlante

El parlante se encuentra del lado inferior izquierdo, cerca de las notas musicales. Vamos a usarlos para generar los tonos, utilizando los pads táctiles.



NeoPixeles

Hay 10 NeoPixeles en un anillo, justo adentro del anillo de pads. Vamos a prender estos LEDs mientras tocamos nuestros tonos.



A continuación, vas a instalar CircuitPython en tu Circuit Playground Express, y a iniciarte con CircuitPython.

Imprimiendo "¡Hola Mundo!"

CircuitPython es una versión de Python diseñada para correr en pequeñas computadoras llamadas microcontroladoras. La Circuit Playground Express una una microcontroladora compatible con CircuitPython. CircuitPython está diseñado de forma que comenzar en electrónica y programación sea muy simple.

CircuitPython busca a un archivo llamado code.py, y ejecuta su código automáticamente. Esto hace que comenzar sea muy directo. CircuitPython te permite conectarte al REPL el cual es una sencilla forma de ver tu código funcionando en vivo mientras salvas tus archivos. También te da una forma para escribir líneas de código que se ejecutan inmediatamente.

En esta parte de la guía, vamos a cubrir como instalar CircuitPython en tu Circuit Playground Express y ¡vas a escribir un poco de código en CircuitPython tu mismo!

Instalando CircuitPython en la Circuit Playground Express

Instalar CircuitPython es muy sencillo. Siga las instrucciones que puedes encontrar en la guía de la Adafruit Circuit Playground Express, sobre CircuitPython (). Incluye algunos pasos sencillos para que arranques con CircuitPython. Una vez que has terminado de instalarlo, puedes regresar aquí para continuar.

¡Felicitaciones al instalar CircuitPython en tu Circuit Playground Express! ¡Muy bien!

La Librería para la Circuit Playground

La libería para la Circuit Playground y todas las librerías de las que depende, se han integrado en CircuitPython para la CircuitPlayground Express. ¡Esto significa que no necesitas cargar ninguna librería para usarlo!

El REPL Serial

CircuitPython envía datos a la computadora conectada a la Circuit Playground Express. Puedes ver la salida de prints, y errores que puedas encontrarte de camino, conectándote al REPL Serial.

Para inicial con el REPL, revisa el tutorial aquí: Consola Serial (REPL) (). Hay enlaces en la guía que cubren como realizarlo en Mac, Linux y Windows.

Mientras que este proyecto se puede completar sin utilizar la REPL, el código está escrito de forma que te da retroalimentación por medio de la consola serial. ¡Puede ser muy útil para depurar problemas!

Desde las Bases de CircuitPython

Una vez que tengas la consola serial, presiona `Ctrl+C` y luego cualquier otra tecla para entar al REPL. Debes ver un mensaje como este que te indica que está lista para recibir información:

```
>>>
```

Esto es un prompt de Python. Aquí puedes escribir código de Python para que sea ejecutado

Cualquier programador en cualquier lenguaje de programación, comienza con un pedazo de código de dice "Hola, mundo" (en inglés "Hello, World."). Vamos a decirle Hola a algo más. A la par del prompt, escriba:

```
print("¡Hola, CircuitPython!")
```

Esto nos retorna:

```
¡Hola, CircuitPython!
```

¡Bienvenido al mundo de la programación!

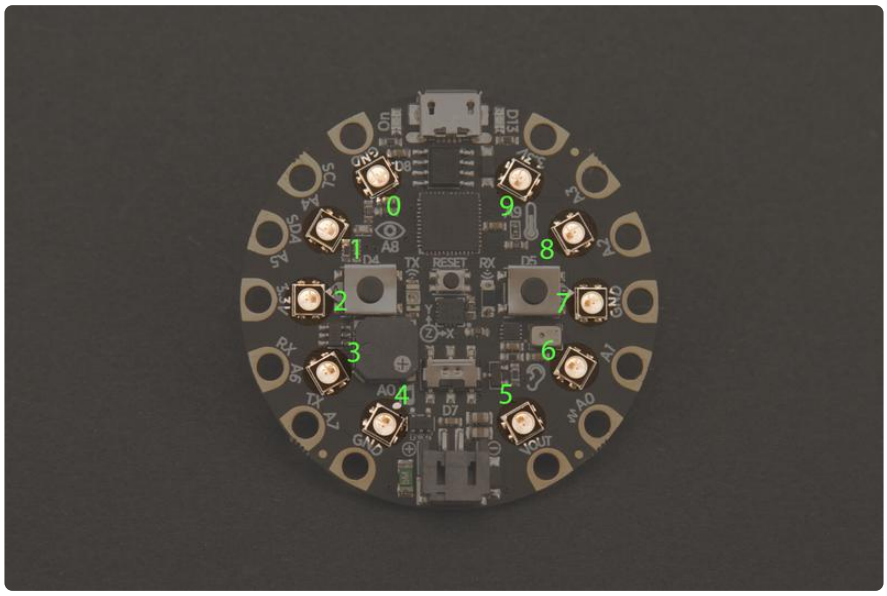
Lecturas Adicionales

Para aprender más, pueder leer la sección Introducción a CircuitPython, de la guía de Circuit Playground Express () section. La guía cubre todas las bases y de da bastante ejemplos para probarlos y aprender de ellos. ¡Revísalos!

Prendiendo sus Luces

La Circuit Playground Express tiene 10 NeoPixeles en un anillo. Están numerados del 0 al 9, comenzando por el pixel superior izquierdo, a la par del puerto micro USB, y van contra las manecillas del reloj alrededor de la tarjeta. Cada pixel puede ser programado de forma individual para que muestre cualquier color, o se pueden programar en conjunto.

Cada LED NeoPixel contiene 3 colores: rojo, verde y azul (o red, green, blue). A estos colores se les llama colectivamente RGB. Cuando trabajamos con luces de colores, cada color se crea de una combinación de estos colores, a diferentes niveles. Si solo el rojo está prendido, vas a ver rojo. Si el rojo y azul están prendidos con la misma intensidad, vas a ver un morado. Si los tres están prendidos con la misma intensidad, vas a ver blanco. Nosotros vamos a utilizar este proceso para agregar un color del piano para cada nota de nuestro piano.



Vamos a comenzar prendiendo el primer NeoPixel. Lo vamos a prender de color azul. Descargue el siguiente archivo. Cámbiele el nombre a code.py. Cópielo a tu CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    cp.pixels[0] = (0, 0, 3)
```

De recompensa por tu trabajo ahora tienes un pixel azul en tu CPX.

Ahora revisemos el código.

La primera línea tiene dos partes. La primera parte, es el módulo. Esto puede ser diferente para diferentes tipos de hardware. Ya que estamos usando la Circuit Playground Express, ¡no ocupamos realizar ningún cambio! La segunda parte de la primera línea, importa la información del módulo que se necesita para realizar su función. Esta línea va a ser parte de todo el código que vamos a utilizar para este proyecto. Es necesaria para que la tarjeta pueda entender el resto del código que vamos a escribir.

La segunda línea le dice a tu CPX que prenda el pixel 0, y que lo ponga de color azul. Los colores se envían usando RGB, con un rango de 0 a 255. Esta es la combinación de números RGB que cambian el color que se muestra, dependiendo de la proporción de estos tres números. Entre más alto el número, más brillante este color individual será. Los tres números, separados por coma, se llama una tupla. La razón para que tengamos dos paréntesis es que la función de `pixels()` espera recibir una sola pieza de información, que es la tupla completa de `(R, G, B)`.

Si cambias el `[0]` a otro número del 0 al 9, puedes prender cualquiera de los NeoPixeles. Si cambias lo números en `((0, 0, 3))` a cualquier número del 0 al 255, vas a cambiar el color y el brillo que muestra cada NeoPixel.

Puedes incluir más de un NeoPixel, agregando otra línea de código con el número del pixel deseado. El siguiente código va a prender también el pixel opuesto. Trata de editar tu archivo code.py para agregarle la línea adicional de código.

```
from adafruit_circuitplayground.express import cpx

cpx.pixels[0] = ((0, 0, 3))
cpx.pixels[5] = ((3, 0, 0))
```

Puedes agregar más pixeles, o cambiar los colores de los que ya tienes prendidos. ¡Diviértete jugando con esto!

Prendiéndolas

Para este proyecto, vamos a prender a todos los NeoPixeles del mismo color. Vamos a utilizar un color diferente para cada tono. Por ahora, vamos a comenzar prendiéndolos todos de color azul.

Si has realizado algún cambio en tu archivo code.py que deseas preservar, cambiar el nombre del archivo primero. Luego, descarga el siguiente archivo. Renombra el archivo como code.py y lo copias a tu CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    cp.pixels.fill((0, 0, 3))
```

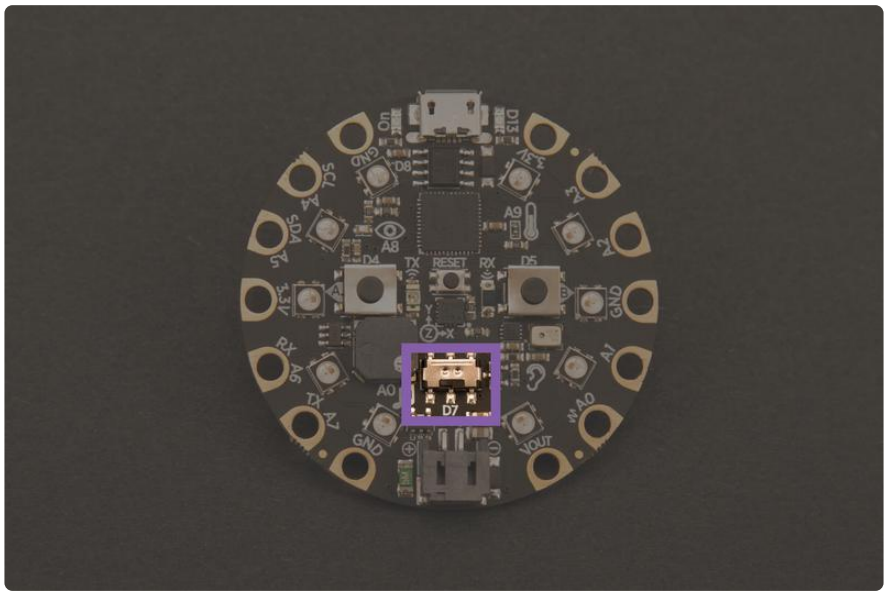
La diferencia entre este código y el primer código que vimos, es que en lugar de especificar un NeoPixel de forma individual por su número, vamos a programarlos todos del mismoc color, utilizando `cpx.pixels.fill`.

De nuevo, puedes experimentar con diferentes colores, cambiando `((0, 0, 3))` a números entre 0 y 255.

Ahora, vamos a aprender como utilizar el interruptor deslizante.

Jugando con el Interruptor

Resulta que si le decimos a la Circuit Playground Express que utilice el tacto capacitivo para tocar un sonido, va a hacer este sonido, todas las veces que lo toquemos. ¡Esto es excelente para este proyecto! Sin embargo, resulta que puede ser frustrante si simplemente estás moviendo la tarjeta o si la quieres desconectar, porque seguiría sonando. Para evitar este problema, vamos a utilizar el interruptor táctil para tener la opción de hacer callar los tonos.



Para comenzar, desliza el interruptor hacia la izquierda. Esto sería la posición de apagado, que vamos a usar para callar nuestros sonidos. Cuando se encuentra en esta posición, la tarjeta lo va a leer como "`True`" (o Verdadero).

Cambia el nombre de tu archivo code.py actual si deseas guardar cambios que hayas realizado. Descarga el siguiente archivo. Cambia su nombre a code.py y lo copias a tu CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT
```

```
from adafruit_circuitplayground import cp

while True:
    if cp.switch:
        print("Slide switch off!")
    else:
        print("Slide switch on!")
```

De nuevo hemos importado el módulo del archivo de librería con el código de la primera línea. Luego tenemos algo nuevo: un ciclo de tipo `while`. Cuando decimos `while True:` lo que significa es "Ejecute para siempre".

`while True:` crea un ciclo. Cuando hay un ciclo, el código va a pasar para siempre por el código que se encuentra dentro del ciclo. Todo el código que está identado bajo `while True:` es el código que se encuentra "dentro" del ciclo.

Dentro de nuestro ciclo, tenemos un `if` . Un `if` lo que dice, "si este evento está sucediendo, entonces haga eso". En nuestro código dice, que si el interruptor está a la izquiera o en `True`, imprima "Slide switch off!" (o "¡Interruptor deslizante apagado!").

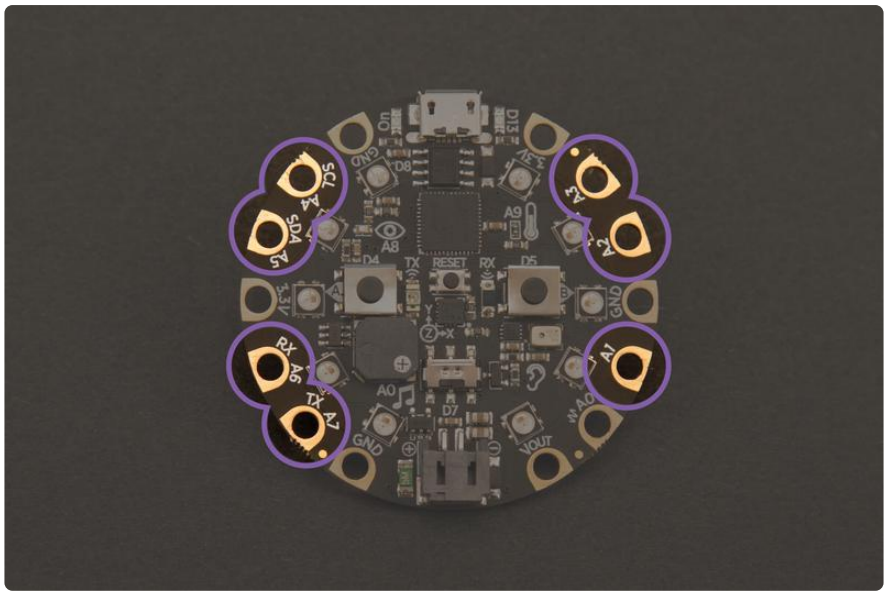
A esto le sigue un `else` . Un `else` lo que dice es "De otra forma, haga esto otro". Un `else` normalmente sigue a un `if`. Juntos lo que dicen es "Si esto pasa, haga esto, pero si no es el caso, haga esto otro". Nuestro código dice que cuando el interruptor esté al lado derecho o en `False`, que imprima "Slide switch on!" (o "¡Interruptor deslizante prendido!")

Tanto el `True` como el `False` es como el interruptor deslizante sabe donde se encuentra, y no necesariamente refleja el propósito que tienes para él.

Ahora, vamos a aprender como utilizar los pads de toque capacitivo, y como agregarles sonido.

Tocando tonos

Esta sección de la guía, nos proporciona capacidades táctiles y de sonido a nuestro proyecto. Primero, vamos a aprender como utilizar los pads de toque capacitivo en la Circuit Playground Express. Le vamos a agregar a cada uno una respuesta que imprime, para saber que está funcionando.



Vamos a comenzar con el pad táctil A1.

Renombra tu archivo code.py actual si deseas guardar algún cambio que hayas realizado. Luego descargas el siguiente archivo. Le cambias el nombre a code.py y lo copias a tu CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    if cp.touch_A1:
        print('Touched 1!')
```

Ahora vamos a revisar el código.

En este código creamos un ciclo. La tarjeta está revisando constantemente a ver si has tocado A1, y te imprime un mensaje en la consola serial REPL, cuando lo haces. El código básicamente se lee como "si ud toca A1, imprima 'Touched 1!' ", y no hace falta más.

El siguiente archivo incluye el resto de pads capacitivos. Descarga el archivo. Cambia su nombre a code.py, y luego lo copias hacia la CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    if cp.touch_A1:
        print('Touched 1!')
```

```
    elif cp.touch_A2:
        print('Touched 2!')
    elif cp.touch_A3:
        print('Touched 3!')
    elif cp.touch_A4:
        print('Touched 4!')
    elif cp.touch_A5:
        print('Touched 5!')
    elif cp.touch_A6:
        print('Touched 6!')
    elif cp.touch_A7:
        print('Touched 7!')
```

Ahora puedes tocar cualquier de los pads capacitivos, y la consola serial del REPL te va a avisar cual fue el que tocaste. ¡Has la prueba!

Vamos a ver una diferencia en el código.

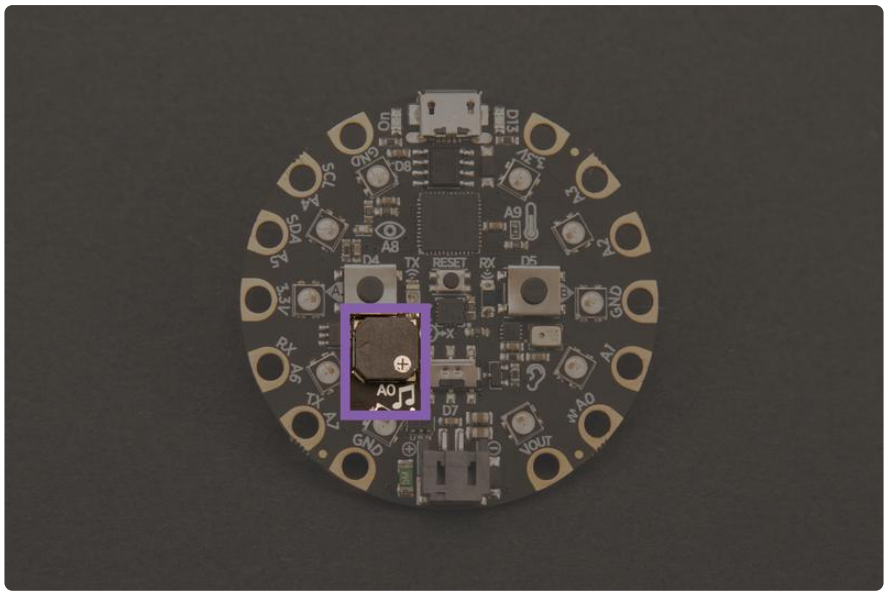
```
while True:
    if cpx.touch_A1:
        print('touched 1!')
    elif cpx.touch_A2:
        print('touched 2!')
```

Hasta el momento hemos utilizado ciclos de tipo `while`, condicionales de tipo `if`, y `else`. Este código incluye una condicional de tipo `elif`. Un `elif` es la combinación de un `else` y un `if`, lo que básicamente significa "De otra forma, siempre y cuando".

Hay 7 pads táctiles. La tarjeta necesita saber cual es la que estás tocando, y dar una respuesta basado en cual pad estás tocando. El código dice "Si estás tocando A1, vas a imprimir'`Touched 1!`', de otra forma, si estás tocando A2, vas a imprimir '`Touched 2!`', de otra forma, si estás tocando A3, vas a imprimir '`Touched 3!`'" y así para cada uno de los pads táctiles. No podemos utilizar un `else` en este código, porque no es un simple escenario de si sí o no. Es más granular porque tiene 7 estados, uno para cada uno de los pad táctiles.

Haciendo Ruido

Ahora, vamos a aprender como tocamos tonos con el parlante integrado.



Hay dos maneras para que suenen tonos con el parlante integrado. Una es usar `play_tone`, el cual toca un tono particular por una duración dada. La otra forma es utilizando `start_tone` y `stop_tone`, las que requieren que le pongas un disparador, como un botón (¡o un pad táctil!), para tocar un tono durante la duración del evento.

Vamos a comenzar realizando un tono sencillo. Descargue el archivo. Cambie su nombre a code.py y luego realice una copia hacia la CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

cp.play_tone(440, 1)
```

En el primer ejemplo, usamos `play_tone`, el cual necesita que se le diga una frecuencia (en Hz) y una duración (en segundos). Le hemos dicho una frecuencia de 440 Hz (un La central) y durante un segundo.

En este caso, el tono suena apenas el código se comienza a ejecutar, dado que no hay otro código que tenga que correr cuando la tarjeta inicia. Este código no repite el tono, solo lo toca una vez. Usted puede salvar de nuevo su code.py, o recargar el REPL para hacerla que toque el tono de nuevo.

Puedes cambiar tanto la frecuencia como la duración para cambiar el tono que tocas y la duración por la que va a sonar. Si necesitas tonos específicos, puedes buscar un generador de tonos en línea. ¡Diviértete con esto!

Start Tone, Stop Tone

Para nuestro piano de tonos, vamos a utilizar `start_tone` y `stop_tone`. Para este ejemplo, vamos a utilizar los pads A1 y A2 de la Circuit Playground Express.

Recuerda, si has realizado cambios que desees preservar, renombra tu archivo code.py a otro nombre. Descarga el siguiente archivo, cambia su nombre por code.py y luego lo copias a la CPX.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    if cp.touch_A1:
        cp.start_tone(262)
    elif cp.touch_A2:
        cp.start_tone(294)
    else:
        cp.stop_tone()
```

Este segundo ejemplo utiliza `start_tone` (o iniciar tono), el cual necesita que le des una frecuencia (en Hz), y luego `stop_tone` (o detener tono) se encarga de se detenga.

Si tocas A1, vas a escuchar un tono hasta que dejes de tocar A1. Si tocas A2, vas a escuchar un tono hasta que dejes de tocar A2. De nuevo hemos utilizado ciclos tipo `while`, y condiciones `if`, y `elif` y esta vez hemos agregado también un `else`. Este código básicamente significa: "Mientras esté tocando A1, toque cierto tono, o de lo contrario detenga el tono. De otra forma, mientras esté tocando A2, toque cierto tono, o de lo contrario detenga el tono". El condicional `else` aplica para ambos el `if` y el `elif` de forma separada.

¡Esta es la base del piano de toque capacitivo! Ya casi llegamos. ¡En la siguiente sección, vamos a juntar todo lo que hemos aprendido en el proyecto final!

Piano de Limones

¡Ahora si vamos a juntar lo que hemos venido aprendiendo!

Asegúrate de salvar tu code.py si has realizado cambios que deseas preservar. Descarga el siguiente archivo, cambia su nombre a code.py y lo salvas hacia tu Circuit Playground Express.

```
# SPDX-FileCopyrightText: 2017 Kattni Rembor for Adafruit Industries
#
# SPDX-License-Identifier: MIT

from adafruit_circuitplayground import cp

while True:
    if cp.switch:
        print("Slide switch off!")
        cp.pixels.fill((0, 0, 0))
        cp.stop_tone()
        continue
    if cp.touch_A4:
        print('Touched A4!')
        cp.pixels.fill((15, 0, 0))
        cp.start_tone(262)
    elif cp.touch_A5:
        print('Touched A5!')
        cp.pixels.fill((15, 5, 0))
        cp.start_tone(294)
    elif cp.touch_A6:
        print('Touched A6!')
        cp.pixels.fill((15, 15, 0))
        cp.start_tone(330)
    elif cp.touch_A7:
        print('Touched A7!')
        cp.pixels.fill((0, 15, 0))
        cp.start_tone(349)
    elif cp.touch_A1:
        print('Touched A1!')
        cp.pixels.fill((0, 15, 15))
        cp.start_tone(392)
    elif cp.touch_A2 and not cp.touch_A3:
        print('Touched A2!')
        cp.pixels.fill((0, 0, 15))
        cp.start_tone(440)
    elif cp.touch_A3 and not cp.touch_A2:
        print('Touched A3!')
        cp.pixels.fill((5, 0, 15))
        cp.start_tone(494)
    elif cp.touch_A2 and cp.touch_A3:
        print('Touched "8"!')
        cp.pixels.fill((15, 0, 15))
        cp.start_tone(523)
    else:
        cp.pixels.fill((0, 0, 0))
        cp.stop_tone()
```

Estudiemos el código:

Lo primero que hacemos es configurar el interruptor deslizante para que el proyecto se pueda activar y desactivar.

```
while True:
    if cpx.switch:
        print("Slide switch off!")
        cpx.pixels.fill((0, 0, 0))
        cpx.stop_tone()
        continue
```

Esto es al inicio del código. Todo el código de este proyecto se encuentra dentro de este ciclo tipo `while`. Lo primero que el código hace dentro de este ciclo, es revisar

si el interruptor se encuentra al lado izquierdo. Para propósitos de este proyecto, eso significa "apagado". Si está apagado, el código va a imprimir "`Slide switch off!`", apaga los LEDs, y deja de tocar cualquier sonido. Luego, el `continue` se asegura que el código no se detenga ahí, sino que continúe a las dos secciones a continuación.

Luego, vamos a programar para que cada pad táctil toque un tono diferente, y que muestre un color diferente en los NeoPixeles. Vamos a dejarle los prints, para tener algo de retroalimentación en el REPL.

Un piano toca notas de la más baja hacia la izquierda, hacia la más alta a la derecha. Para permitirle a nuestras "teclas" que estén en la configuración correcta, vamos a utilizar cables de lagarto con punta de pin, para que nos permita acomodarlos como teclas, y vamos a tener que hacer un par de cosas fuera de orden. Vamos a programar los pads táctiles en el mismo orden que vamos colocando los limones, comenzando desde la izquiera. Iniciemos por A4:

```
    if cpx.touch_A4:
        print('Touched A4!')
        cpx.pixels.fill((15, 0, 0))
        cpx.start_tone(262)
```

Si tocas a A4, el código imprime "`Touched A4!`" en el REPL, prende las luces de color rojo, y toca un tono de 262 Hz (lo cual es un Do medio). Esta es la primera "tecla" de nuestro piano. A5, A6, A7 y A1 básicamente hacen lo mismo, pero les cambiamos el tono y el color. Aquí usamos condicionales tipo `elif` en lugar de condicionales tipo `if`.

Hay 7 pads táctileles. Vamos a incluir 8 tonos para crear una octava completa. Para simular este pad número 8, vas a tocar A2 y A3 al mismo tiempo. De esta forma, tenemos que escribir el código que haga la diferencia entre A2, entre A3 y entre los dos al mismo tiempo.

Miremos:

```
    elif cpx.touch_A2 and not cpx.touch_A3:
        print('Touched A2!')
        cpx.pixels.fill((0, 0, 15))
        cpx.start_tone(440)
    elif cpx.touch_A3 and not cpx.touch_A2:
        print('Touched A3!')
        cpx.pixels.fill((5, 0, 15))
        cpx.start_tone(494)
```

Si tocas A2 pero no A3, el código imprime: "`Touched A2!`", y muestra el color dado y toca el tono dado. Si tocas A3 pero no A2, el código imprime: "`Touched A2!`", y muestra el color dado y toca el tono dado.

El pad simulado funciona de esta forma:

```
    elif cpx.touch_A2 and cpx.touch_A3:
        print('Touched "8"!')
        cpx.pixels.fill((15, 0, 15))
        cpx.start_tone(523)
```

Si estás tocando A2 y A3, el código imprime "`Touched "8"!`", despliega el color final, y suena el tono final.

La última pieza del código es donde le decimos que si no está tocando nada, apague las luces, y deje de tocar tonos.

```
    else:
        cpx.stop_tone()
        cpx.pixels.fill((0, 0, 0))
```

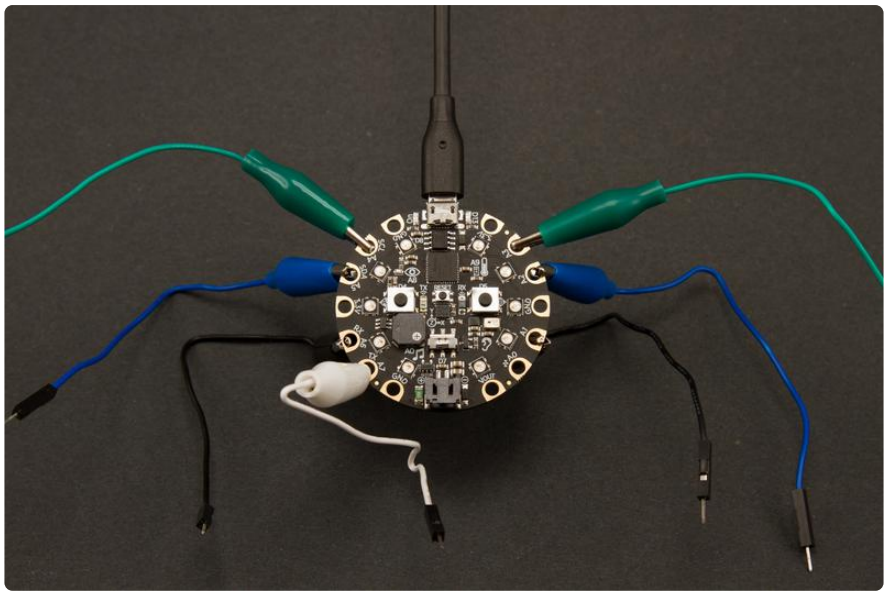
Ahora, ¡si tocas los pads, vas a escuchar los tonos y a ver los colores según se asignaron!

Armando nuestro Piano de Limones

¡Ahora vamos a agregarle nuestras teclas de frutas!

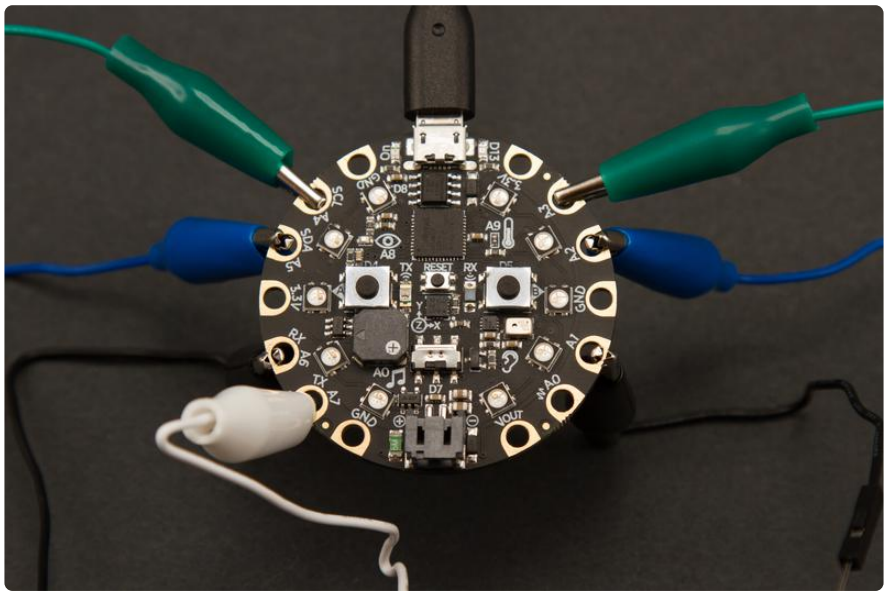
Apaga tu Circuit Playground Express. Conecta un cable de lagarto a cada uno de los pads.

Los pads de toque capacitivo se calibran al inicio, así que si dejaste prendida tu CPX, es mejor que presiones el botón de reinicio luego de conectar las frutas.



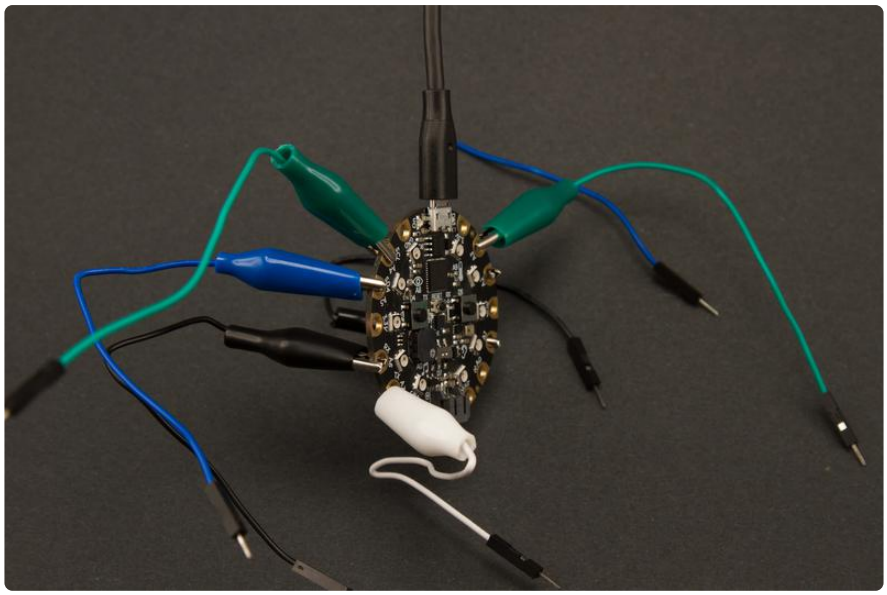
Las vas a querer acomodar de forma que no se toquen. Puedes doblar ligeramente los cables para convencerlos que se queden donde los necesitas. Si los cables se tocan, puede que parezca como si los estuvieran tocando.

El acomodarlos de esta forma, permite que no se toquen:

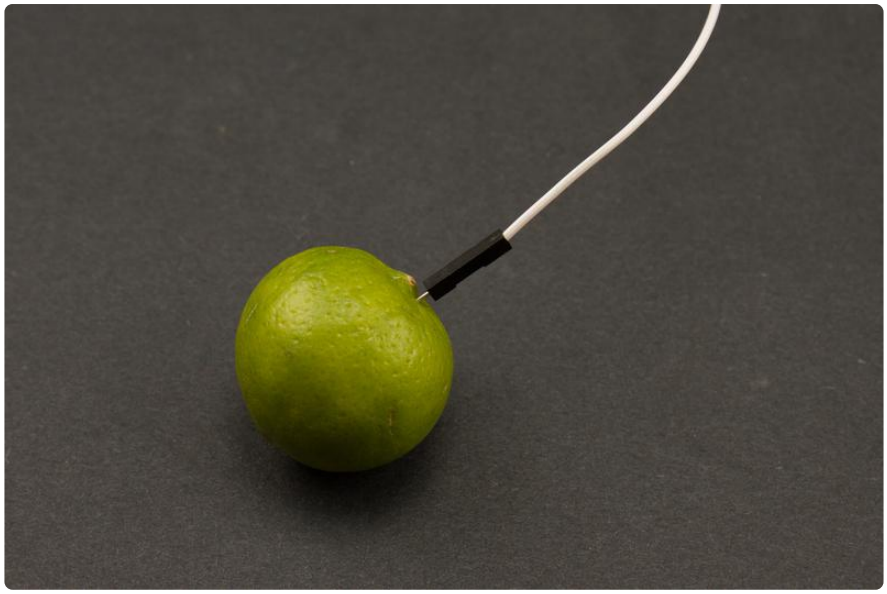


Los clips se conectan de forma que permitan abanicarse, y para evitar interferencia de los otros clips.

Note como el clip que se conectó a A7, ha sido doblado dos veces para lograr acortar su tamaño.

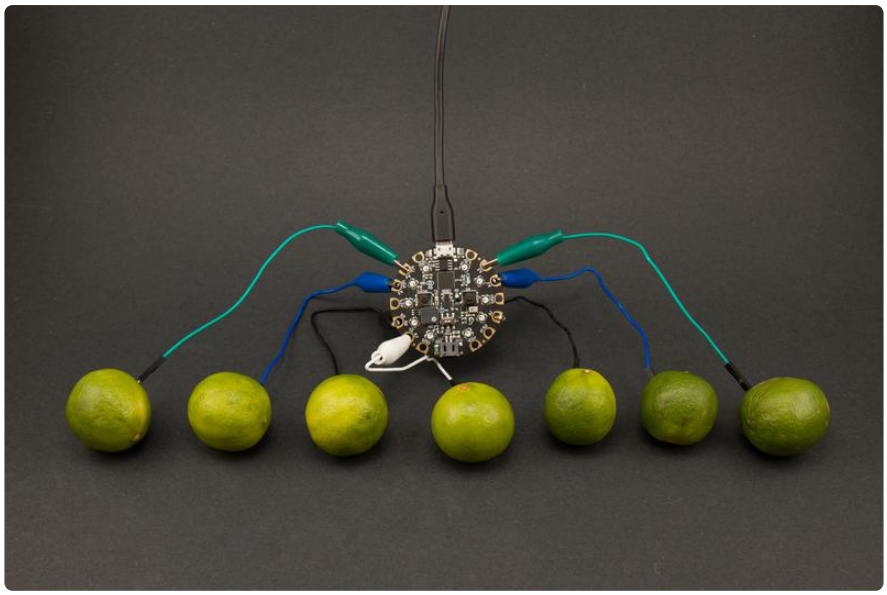


Ahora vamos a conectar las frutas. Con limas, es más sencillo porque puedes pinchar su piel con facilidad utilizando una punta de pin.



Ahora acomoda las limas en una línea, pero no muy cerca la una de la otra. Al igual que con los cables, el poner las limas muy cerca, puede causar interferencia.

Conecte su Circuit Playground Express. Puede que necesites presionar el botón de reset luego de conectarla, si está conectada a batería.



¡Ahora es tiempo de tocar! Este video muestra una canción siendo tocada en el Piano de Limones. ¡Que te diviertas tocando!