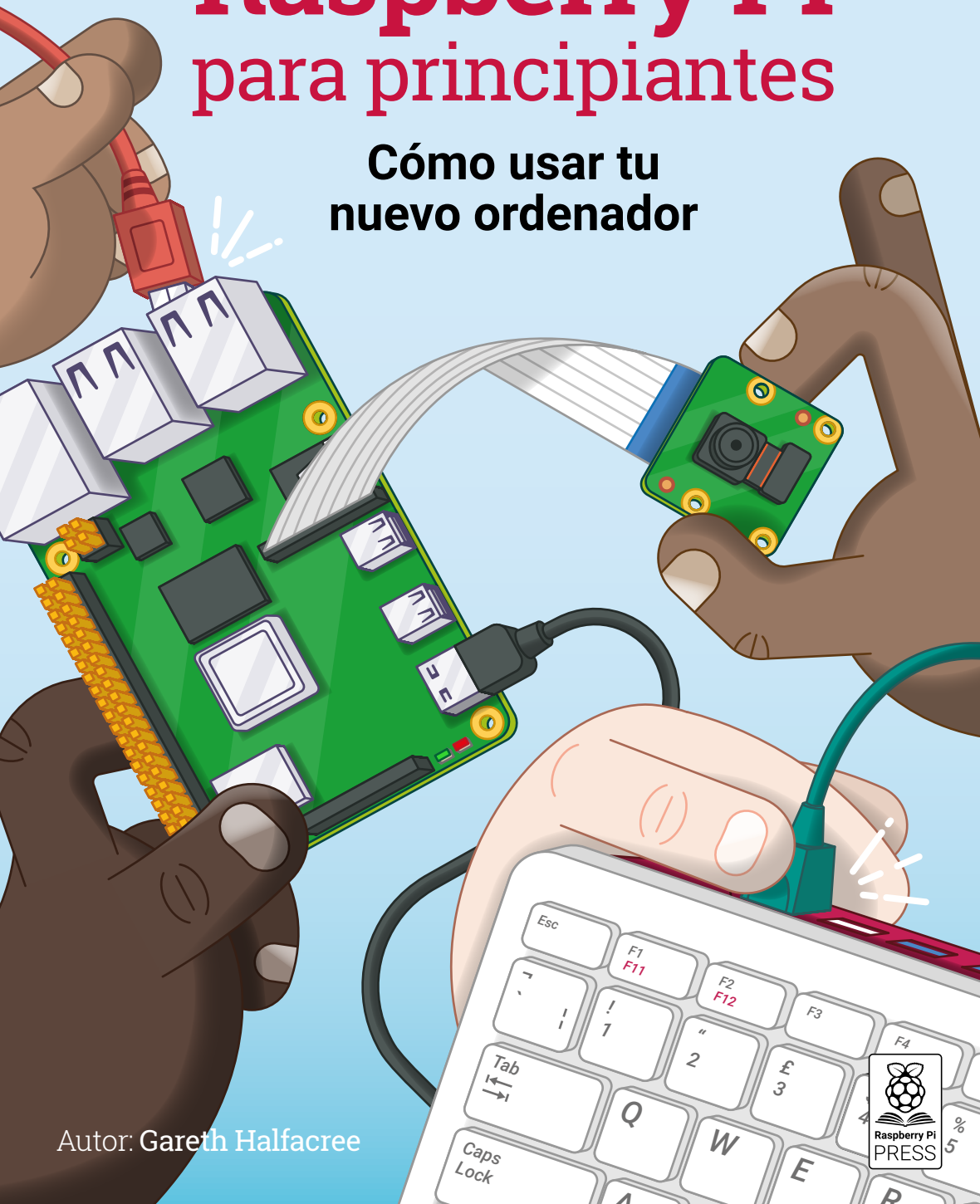
4ª Edición

TOTALMENTE ACTUALIZADO PARA **RASPBERRY PI 400**

LA GUÍA OFICIAL DE Raspberry Pi para principiantes

Cómo usar tu nuevo ordenador



Autor: Gareth Halfacree

LA GUÍA OFICIAL DE Raspberry Pi para principiantes

Cómo usar tu nuevo ordenador

Primera publicación en 2020: Raspberry Pi Trading Ltd, Maurice Wilkes Building,
St. John's Innovation Park, Cowley Road, Cambridge, CB4 0DS, Reino Unido

Director de publicación: Russell Barnes • Editor: Phil King
Diseño: Critical Media • Ilustraciones: Sam Alder
CEO: Eben Upton

ISBN: 978-1-912047-94-9

Bienvenida - Guía oficial de Raspberry Pi para principiantes

Creemos que Raspberry Pi te va a encantar. Este pequeñísimo y asequible ordenador cuesta menos que la mayoría de los videojuegos, pero se puede usar para aprender a codificar, construir robots y crear todo tipo de proyectos extraños y maravillosos.

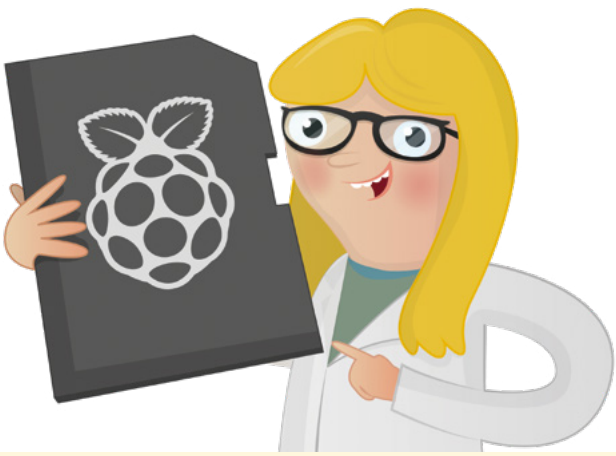
Raspberry Pi es capaz de hacer todo lo habitual con un ordenador: navegar por Internet, jugar, ver películas y escuchar música. Pero Raspberry Pi es mucho más que un ordenador moderno.

Con Raspberry Pi puedes llegar al núcleo de un ordenador. Te permite configurar tu propio sistema operativo y puedes conectar los cables y circuitos directamente a los pines de la placa. Se diseñó para enseñar a la gente joven a programar en lenguajes como Scratch y Python, y los principales lenguajes de programación se incluyen en el sistema operativo oficial.

El mundo necesita programadores más que nunca y Raspberry Pi ha despertado la pasión por la informática y la tecnología en una nueva generación.

Gente de todas las edades usa Raspberry Pi para crear proyectos fascinantes: desde consolas de juego vintage a estaciones meteorológicas conectadas a Internet.

Así que si quieres crear juegos, construir robots o trabajar en todo tipo de proyectos increíbles, esta guía es lo que necesitas para ayudarte a empezar.



El autor

Gareth Halfacree es un escritor y periodista autónomo, especializado en temas tecnológicos, y previamente trabajó como administrador de sistemas en el sector educativo. Es un entusiasta del software y el hardware de código abierto, y fue uno de los primeros en adoptar la plataforma Raspberry Pi, sobre cuyas capacidades y flexibilidad ha escrito varias publicaciones. En Twitter se encuentra en **@ghalfacree** y su sitio web es **freelance.halfacree.co.uk**.

Índice

APÉNDICES

Capítulo 1

Introducción a Raspberry Pi

Familiarízate con tu nuevo ordenador del tamaño de una tarjeta de crédito, mediante un recorrido guiado de Raspberry Pi. Descubre sus numerosos componentes y lo que hacen

Raspberry Pi es un dispositivo excepcional: un ordenador totalmente funcional en un formato pequeño y de bajo coste. Tanto si quieres un dispositivo para navegar en Internet como si es para jugar, o si quieres aprender a escribir tus propios programas o crear tus propios circuitos y dispositivos físicos, Raspberry Pi (y su increíble comunidad) te ayuda en cada paso.

Raspberry Pi es lo que se conoce como *ordenador de una sola placa*, que es exactamente lo que su nombre indica: como un ordenador de sobremesa, un portátil o un smartphone, pero construido sobre una única *placa de circuito impreso*. Como la mayoría de los ordenadores de una sola placa, Raspberry Pi es pequeño —más o menos del tamaño de una tarjeta de crédito— pero eso no impide que sea potente: un Raspberry Pi puede hacer cualquier cosa que haga un ordenador más grande y de mayor consumo, aunque puede que no lo haga tan rápido.

La familia Raspberry Pi nació del deseo de fomentar la educación informática práctica en todo el mundo. Sus creadores unieron sus esfuerzos para establecer la Raspberry Pi Foundation, organización sin ánimo de lucro, sin saber lo popular que iba a hacerse: los pocos miles de unidades fabricadas en 2012 para tantear el terreno se vendieron inmediatamente, y desde entonces se han despachado millones de unidades en todo el mundo. Estas placas se han adoptado en hogares, aulas, oficinas, centros de datos, fábricas, e incluso en embarcaciones dirigidas automáticamente y en globos espaciales.

Desde el Model B original se han lanzado varios modelos de Raspberry Pi, cada uno con especificaciones mejoradas o características específicas para un tipo de uso particular. La familia Raspberry Pi Zero, por ejemplo, es una versión diminuta de la Raspberry Pi de tamaño completo que omite algunas características —en concreto los múltiples puertos USB y el puerto de red con cable— para reducir el formato y los requisitos energéticos.

Pero todos los modelos de Raspberry Pi tienen una cosa en común: son *compatibles*, lo que significa que el software escrito para un modelo funcionará en cualquier otro modelo. Incluso es posible utilizar la versión más reciente del sistema operativo de Raspberry Pi y ejecutarla en un prototipo original del Model B prelanzamiento. Si bien es cierto que el funcionamiento será más lento, el caso es que podrá ejecutarse.

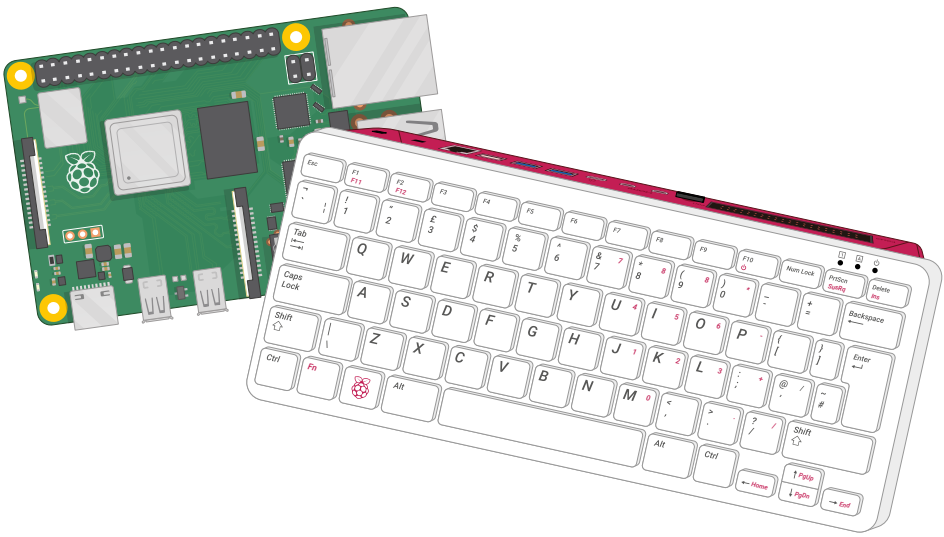
En esta guía vas a conocer Raspberry Pi 4 Model B y Raspberry Pi 400, las versiones más recientes y más potentes de Raspberry Pi. Y lo que aprendas podrás aplicarlo fácilmente a otros modelos de la familia Raspberry Pi, así que no te preocupes si la versión que utilizas no es una de esas.

RASPBERRY PI 400

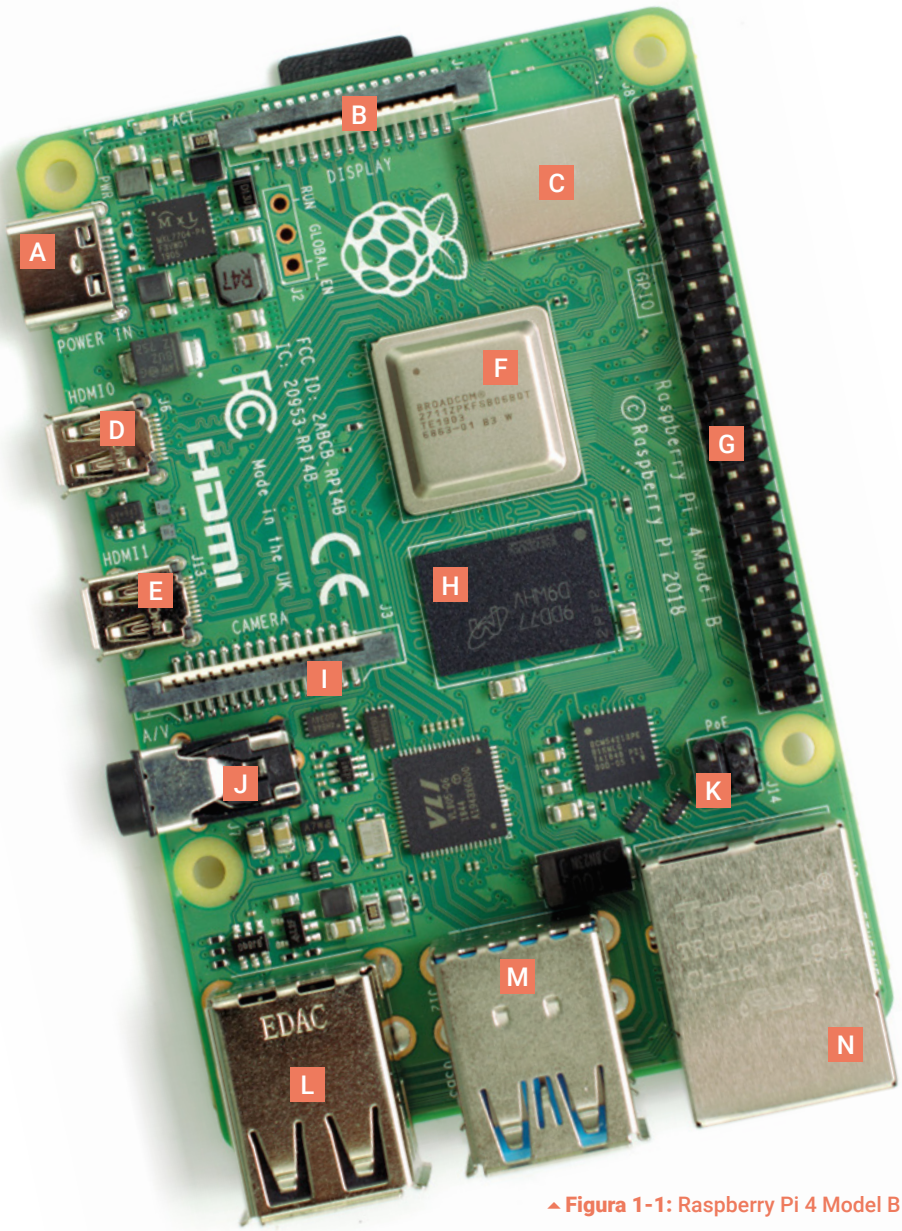
Si tienes un Raspberry Pi 400, la placa de circuito está integrada en la carcasa del teclado. Aquí encontrarás información sobre todos los componentes que hacen que Raspberry Pi funcione. O ve a la página 20 para un recorrido guiado del dispositivo.

Recorrido guiado de Raspberry Pi

A diferencia de un ordenador tradicional, que esconde sus componentes internos en una carcasa, un Raspberry Pi estándar tiene todos sus componentes y puertos a la vista, aunque si lo prefieres puedes comprar una carcasa como protección añadida. La visibilidad la convierte en una gran herramienta para aprender qué hacen las distintas partes de un ordenador, y también facilita el aprendizaje de la ubicación de cada cosa cuando hay que conectar los distintos extras, denominados *periféricos*.



- A Entrada de alimentación USB tipo C
- B Puerto de pantalla DSI
- C Conexión inalámbrica / Bluetooth
- D Micro-HDMI 0
- E Micro-HDMI 1
- F Sistema en chip
- G GPIO
- H RAM
- I Puerto de cámara CSI
- J AV de 3,5 mm
- K PoE
- L USB 2.0
- M USB 3.0
- N Puerto Ethernet



▲ **Figura 1-1:** Raspberry Pi 4 Model B

La **Figura 1-1** muestra un Raspberry Pi 4 Model B visto desde arriba. Cuando uses un Raspberry Pi siguiendo esta guía, intenta mantenerlo orientado igual que en la foto; si no, podría confundirte en secciones como la de GPIO (en el **Capítulo 6, Informática física con Scratch y Python**).

Aunque parezca que hay multitud de cosas apiñadas en la minúscula placa, un Raspberry Pi es muy simple de entender, empezando por sus *componentes*, que son los que hacen que el dispositivo funcione.

Los componentes de Raspberry Pi

Como cualquier ordenador, el Raspberry Pi consta de varios componentes, cada uno de los cuales cumple su propio papel para el funcionamiento. El primero, y posiblemente el más importante, se encuentra justo encima del punto central de la parte superior de la placa (**Figura 1-2**), cubierto con una tapa metálica: es el *sistema en chip* (o SoC, del inglés system on a chip).

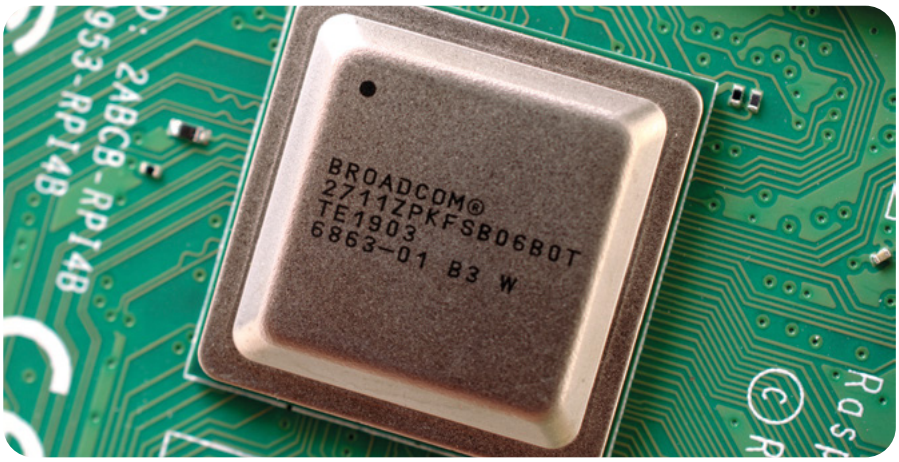


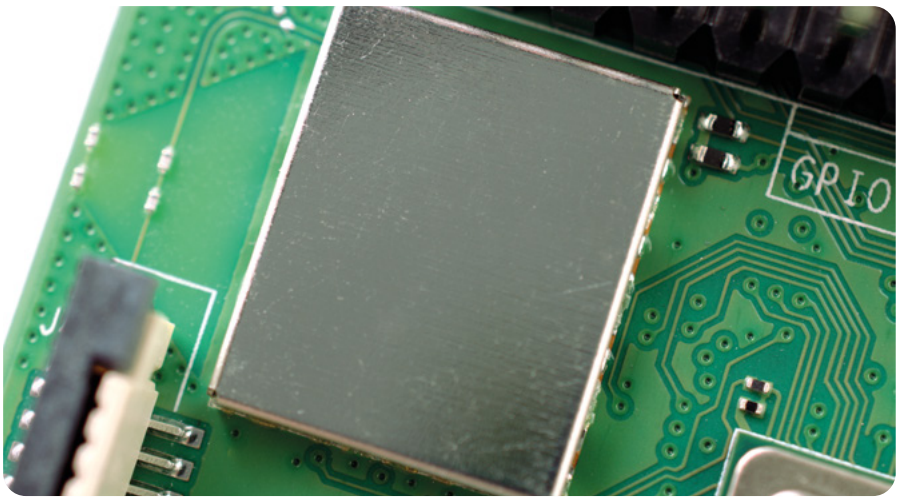
Figura 1-2: Sistema en chip (SoC) de Raspberry Pi

El nombre sistema en chip hace referencia a lo que encontrarías si quitaras la tapa metálica: un chip de silicio, conocido como *circuito integrado*, que contiene la mayor parte del sistema Raspberry Pi. Esto incluye la *unidad central de procesamiento* (CPU), comúnmente considerada como el "cerebro" de un ordenador, y la *unidad de procesamiento gráfico* (GPU), que se encarga del aspecto visual de las cosas.

Pero como un cerebro no es nada si no tiene memoria, justo al lado del SoC encontrarás exactamente eso: otro chip, que parece un pequeño cuadrado de plástico negro (**Figura 1-3** a continuación). Esta es la *memoria de acceso aleatorio (RAM)* de Raspberry Pi. Cuando trabajas con Raspberry Pi, la memoria RAM es la que contiene lo que estás haciendo, que no se guarda en la tarjeta microSD hasta que guardes tu trabajo. Juntos, estos componentes forman las memorias volátil y no volátil de Raspberry Pi: la RAM volátil pierde su contenido cada vez que Raspberry Pi se apaga, mientras que la tarjeta microSD no volátil conserva lo que contiene.

▲ **Figura 1-3:** Memoria de acceso aleatorio (RAM) de Raspberry Pi

En la parte superior derecha de la placa encontrarás otra tapa de metal (**Figura 1-4**) que cubre el componente *radio*, el que permite a Raspberry Pi comunicarse con dispositivos de forma inalámbrica. La propia radio actúa como dos componentes principales: una *radio WiFi* para conectarse a redes informáticas; y una *radio Bluetooth* para conectarse a periféricos como ratones, y para enviar o recibir datos de dispositivos inteligentes cercanos, como sensores o smartphones.



▲ **Figura 1-4:** Módulo de radio de Raspberry Pi

Hay otro chip negro con cobertura de plástico cerca del borde inferior de la placa, justo detrás del conjunto central de puertos USB. Es el *controlador USB*, responsable del funcionamiento de los cuatro puertos USB. Junto a él hay un chip aún más pequeño, el *controlador de red*, que se encarga del puerto de red Ethernet de Raspberry Pi. Un último chip negro, más pequeño que el resto, se encuentra encima del conector de alimentación USB tipo C, en la parte superior izquierda de la placa (**Figura 1-5**). Es el denominado *circuito integrado de gestión de energía (PMIC)* y se encarga de convertir la energía que entra por el micro puerto USB en la energía que Raspberry Pi necesita para funcionar.

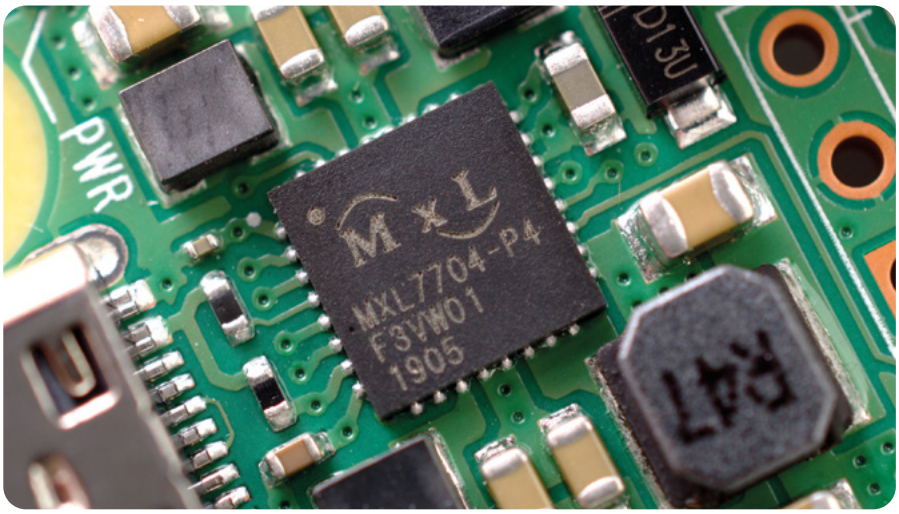


Figura 1-5: Circuito integrado de gestión de energía (PMIC) de Raspberry Pi

No te agobies con toda esta información: para usar Raspberry Pi no es imprescindible saber qué es cada componente o dónde encontrarlo en la placa.

Puertos de Raspberry Pi

Raspberry Pi tiene una serie de puertos, empezando por cuatro *puertos de bus serie universal (USB)* (**Figura 1-6**) en la zona central y derecha del borde inferior. Estos puertos permiten conectar a Raspberry Pi cualquier periférico compatible con USB, desde teclados y ratones a cámaras digitales y unidades flash. Técnicamente hablando, hay dos tipos de puertos USB: los que tienen partes negras en su interior son puertos USB 2.0, basados en la versión 2 del estándar Universal Serial Bus; los que tienen partes azules son puertos USB 3.0, más rápidos, basados en la versión 3 más reciente.

▲ **Figura 1-6:** Puertos USB de Raspberry Pi

A la derecha de los puertos USB hay un *puerto Ethernet*, también conocido como *puerto de red* (**Figura 1-7**). Puedes usar este puerto para conectar Raspberry Pi a una red de ordenadores con cable, usando un cable con un conector RJ45 en su extremo. Si observas minuciosamente el puerto Ethernet, verás dos diodos emisores de luz en la parte inferior: son los LED de estado y te indican que la conexión funciona.

Figura 1-7: Puerto Ethernet de Raspberry Pi

Justo encima del puertos USB, en el borde izquierdo de Raspberry Pi, hay una *toma audiovisual (AV) de 3,5 mm* (**Figura 1-8**). También se conoce como *toma de auriculares* y se puede usar con ese propósito, aunque obtendrás mejor sonido si conectas altavoces amplificados, en lugar de auriculares. Pero la toma tiene una función extra oculta: además de audio, el conector AV de 3,5 mm transmite una señal de vídeo que se puede conectar a televisores, proyectores y otras pantallas que admitan una *señal de video compuesto* usando un cable especial conocido como adaptador *TRRS* (las iniciales inglesas de punta-anillo-anillo-cuerpo).

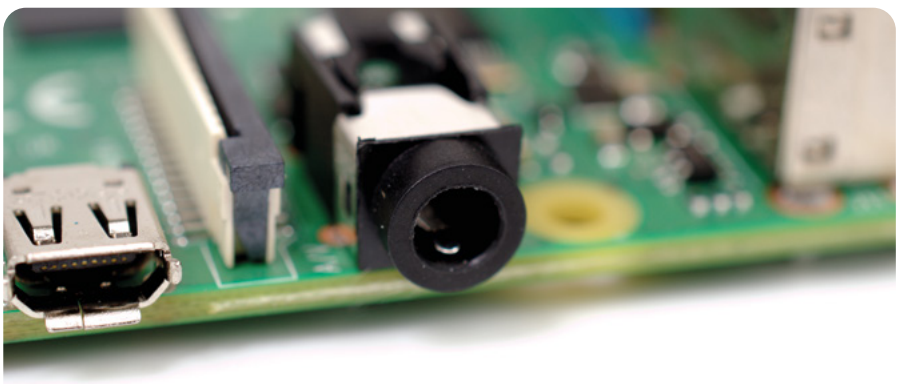


Figura 1-8: Toma AV de 3,5 mm de Raspberry Pi

Directamente encima de la toma AV de 3,5 mm hay un conector de aspecto extraño, con una solapa de plástico de la que se puede tirar hacia arriba: es el *conector de cámara* también denominado *interfaz serie para cámara* (**Figura 1-9**). Este conector te permite usar el módulo de cámara de Raspberry Pi diseñado especialmente (del que aprenderá más cosas en el **Capítulo 8, Cámara de Raspberry Pi**.)

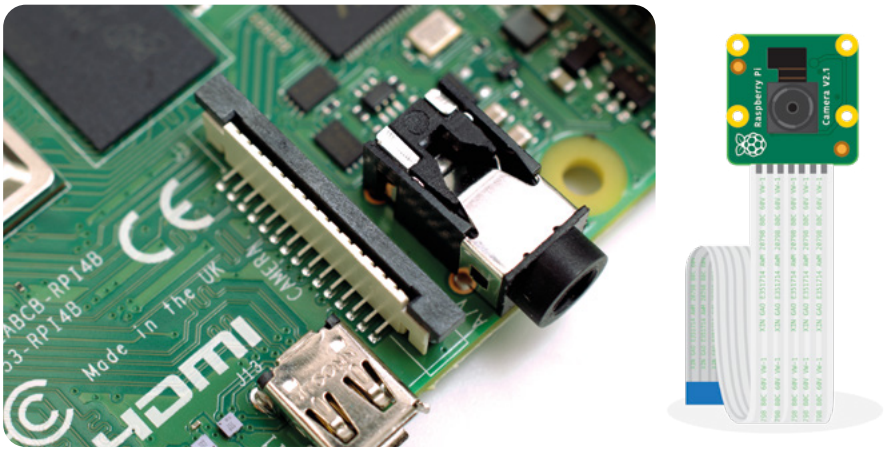


Figura 1-9: Conector de cámara de Raspberry Pi

Por encima, también en el borde izquierdo de la placa, están los *micro puertos de interfaz multimedia de alta definición (micro-HDMI)*, que son una versión más pequeña de los conectores que existen en una consola de juego, un descodificador o un televisor (**Figura 1-10**). La palabra multimedia del nombre indica que transporta señales de audio y vídeo, mientras que "alta definición" hace referencia a la calidad previsible. Usarás esos puertos para conectar Raspberry Pi a uno o dos dispositivos de visualización: un monitor de ordenador, un televisor o un proyector.

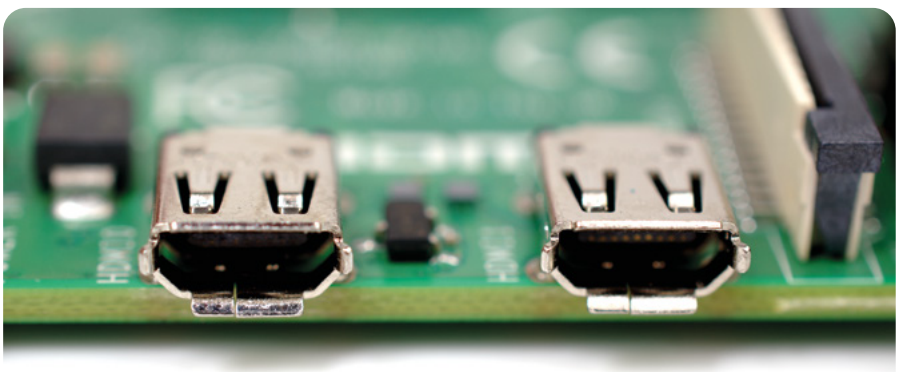


Figura 1-10: Puertos HDMI de Raspberry Pi

Encima de los puertos HDMI hay un *puerto de alimentación USB tipo C* (**Figura 1-11**), que usarás para conectar Raspberry Pi a una fuente de alimentación. El puerto USB tipo C suele ser habitual en smartphones, tablets y otros dispositivos portátiles. Aunque se puede usar un cargador móvil estándar para alimentar Raspberry Pi, los mejores resultados se obtienen con la fuente de alimentación oficial USB tipo C de Raspberry Pi.

Figura 1-11: Puertos de alimentación USB tipo C de Raspberry Pi

En el borde superior de la placa hay otro conector de aspecto extraño (**Figura 1-12**), que a primera vista parece idéntico al conector de cámara. Pero es justo lo contrario: un *conector de pantalla* o *interfaz serie de pantalla (DSI)*, diseñado para usarse con una pantalla de Raspberry Pi táctil (**Figura 1-13** a continuación).

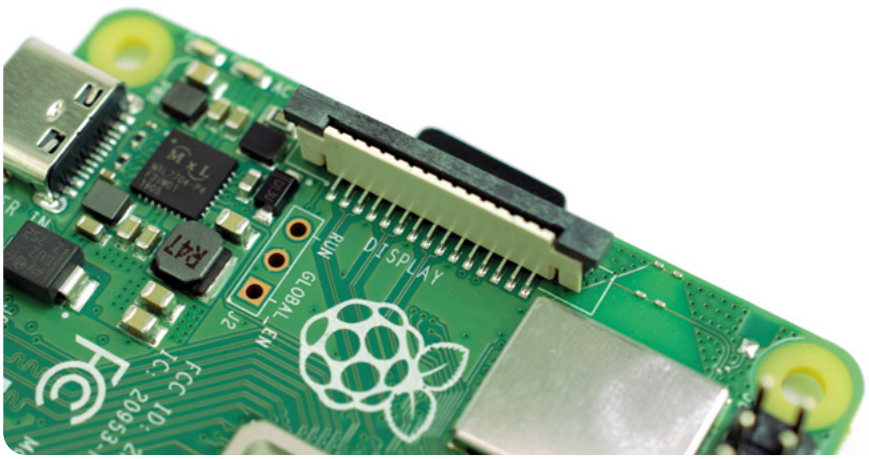
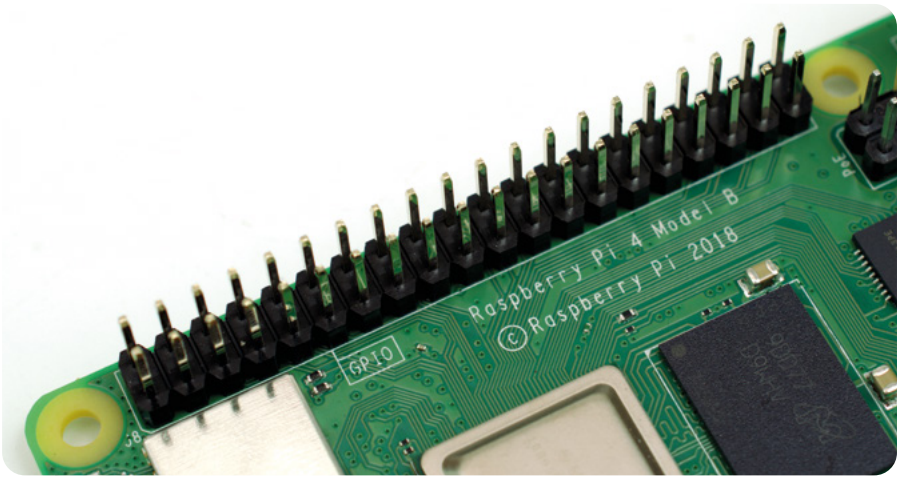


Figura 1-12: Conector de pantalla (DSI) de Raspberry Pi

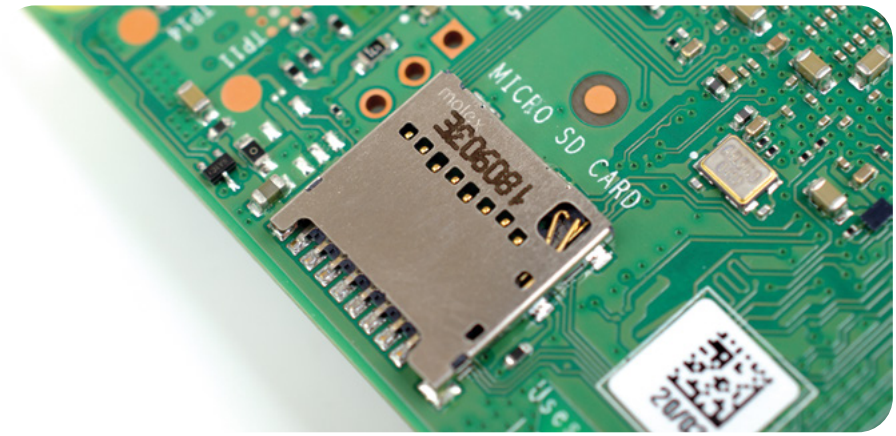
▲ **Figura 1-13:** La pantalla táctil de Raspberry Pi

En el borde derecho de la placa hay 40 pines metálicos, divididos en dos filas de 20 pines cada una (**Figura 1-14**). Se trata del *sistema GPIO (entrada/salida de uso general)*, componente de Raspberry Pi usado para la comunicación con hardware adicional (como LED, botones, sensores de temperatura, joysticks y monitores de pulso). Aprenderás más sobre GPIO en el **Capítulo 6, Informática física con Scratch y Python**. Justo debajo y a la izquierda de este sistema hay otro sistema más pequeño con cuatro pines: se usa para conectar el HAT PoE (de alimentación a través de Ethernet), un complemento opcional que permite a Raspberry Pi recibir alimentación de una conexión de red, en lugar del puerto USB tipo C.



▲ **Figura 1-14:** Sistema GPIO de Raspberry Pi

Hay un último puerto en Raspberry Pi, pero no está a la vista en la parte superior. Dale la vuelta a la placa y verás un *conector de tarjeta microSD* en el lado opuesto al del conector de la pantalla (**Figura 1-15**). Este es el almacén de Raspberry Pi: la tarjeta microSD insertada aquí contiene todos los archivos que guardas, todo el software que instalas y el sistema operativo que hace funcionar a Raspberry Pi.



▲ **Figura 1-15:** Conector de tarjeta microSD de Raspberry Pi

▲ **Figura 1-16:** Raspberry Pi 400 tiene un teclado integrado

Raspberry Pi 400

Raspberry Pi 400 utiliza los mismos componentes que Raspberry Pi 4, ubicados dentro de un teclado. Además de protegerlos, la carcasa del teclado ocupa menos espacio en el escritorio y ayuda a mantener los cables más organizados.

Raspberry Pi 400 tiene los mismos componentes principales que Raspberry Pi 4, incluidos el sistema en chip y la memoria. Aunque no los veas, están ahí. Lo que se puede ver son las partes externas, empezando por el teclado (**Figura 1-16**). Cerca de la esquina derecha hay tres LED: el primero se enciende cuando se pulsa la tecla de bloqueo numérico, que cambia algunas de las teclas para que actúen como un teclado numérico de tamaño normal; el segundo se enciende cuando se pulsa la tecla de bloqueo de mayúsculas, que hace que las teclas de letras sean mayúsculas en lugar de minúsculas; y el último se enciende al encender Raspberry Pi 400.

En la parte posterior de Raspberry Pi 400 (**Figura 1-17**) están los puertos. El de la izquierda, visto desde atrás, es el sistema de entrada/salida de uso general (GPIO). Es el que se ha descrito en la página 17, pero volteado: el primer pin, el 1, está arriba a la derecha y el último, el 40, abajo a la izquierda. Puedes averiguar más sobre GPIO en el **Capítulo 6, Informática física con Scratch y Python**.

Figura 1-17: Los puertos se encuentran en la parte posterior de Raspberry Pi 400

Junto al sistema GPIO está la ranura de la tarjeta microSD, donde reside la tarjeta que hace de almacén de Raspberry Pi 400 para el sistema operativo, las aplicaciones y otros datos. La tarjeta microSD viene preinstalada en Raspberry Pi 400; puedes quitarla presionando suavemente hasta que haga clic y salga a resorte, y tirando de ella. Cuando vuelvas a colocar la tarjeta, asegúrate de que los contactos metálicos brillantes estén orientados hacia abajo; la tarjeta debería deslizarse hasta la posición requerida y encajar con un suave clic.

Los dos puertos siguientes son los puertos micro-HDMI, para la conexión a un monitor, TV u otra pantalla. Al igual que Raspberry Pi 4, Raspberry Pi 400 admite hasta dos pantallas. Junto a estos puertos está el de alimentación USB tipo C, para la conexión a la fuente de alimentación de Raspberry Pi o a una fuente de alimentación USB compatible.

Los dos puertos azules son USB 3.0 y proporcionan una conexión de alta velocidad a dispositivos como unidades SSD (de estado sólido), llaves de memoria, impresoras, etc. El puerto blanco a la derecha de estos es un puerto USB 2.0 de menor velocidad, al que se puede conectar el ratón incluido con Raspberry Pi.

El último puerto es un puerto de red Gigabit Ethernet, que te permite conectar Raspberry Pi 400 a tu red mediante un cable de red como alternativa a la red inalámbrica Wi-Fi incorporada. Encontrarás más información sobre la conexión de Raspberry Pi 400 a una red en el **Capítulo 2, Preparativos para usar Raspberry Pi**.

Capítulo 2

Preparativos para usar Raspberry Pi

Descubre los elementos esenciales que necesitarás para tu Raspberry Pi y cómo conectarlos todos para que funcione

Raspberry Pi se ha diseñado para ser tan rápido y fácil de configurar y usar como sea posible pero, como cualquier ordenador, requiere varios componentes externos, denominados *periféricos*. Aunque es comprensible que al ver la placa de circuitos impresos de Raspberry Pi (muy diferente a los ordenadores con carcasa más habituales) pienses que todo va a ser complicado, no es así. Siguiendo los pasos de esta guía, podrías tenerlo todo a punto para usar Raspberry Pi en menos de diez minutos.

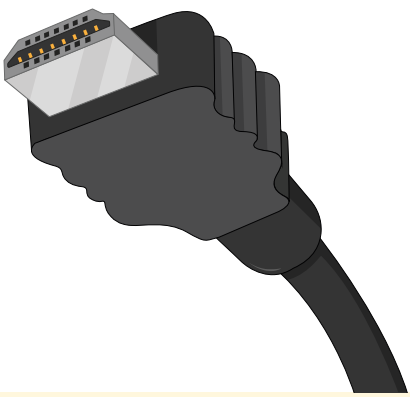
Si has recibido este documento en un kit de escritorio de Raspberry Pi o con un Raspberry Pi 400, ya tendrás casi todo lo necesario para empezar y lo único que debes aportar es un monitor de ordenador o un televisor con una conexión HDMI (el mismo tipo de conector que usan los descodificadores, los reproductores de Blu-ray y las consolas de juego) para ver lo que hace tu Raspberry Pi.

Si has optado por un Raspberry Pi sin accesorios, también necesitarás lo siguiente:

- **Fuente de alimentación USB** – De 5 V y 3 amperios (3 A), con un conector USB tipo C. La fuente de alimentación oficial de Raspberry Pi es la opción recomendada, ya que es capaz de hacer frente a las cambiantes demandas de energía de Raspberry Pi.

- **Tarjeta microSD con NOOBS** – La tarjeta microSD actúa como almacén permanente de Raspberry Pi; todos los archivos que crees y el software que instales, junto con el propio sistema operativo, se almacenan en la tarjeta. Una tarjeta de 8 GB es suficiente para empezar, aunque una de 16 GB ofrece más espacio para progresar. Una tarjeta con NOOBS (New Out-Of-Box Software) preinstalado te ahorrará tiempo. Como alternativa, consulta el **Apéndice A** para ver instrucciones sobre la instalación de un sistema operativo en una tarjeta vacía.

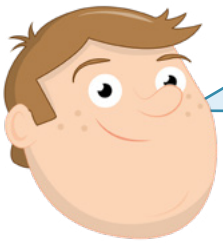
- **Teclado y ratón USB** – El teclado y el ratón te permiten controlar tu Raspberry Pi. Raspberry Pi admite casi cualquier teclado y ratón con cable o inalámbricos dotados de un conector USB, aunque algunos teclados para gaming con luces de colores pueden consumir demasiada energía para un uso fiable.



- **Cable micro-HDMI** – Transmite el sonido y las imágenes de Raspberry Pi a tu TV o monitor. Un extremo del cable tiene un conector micro-HDMI para Raspberry Pi; el otro, un conector HDMI de tamaño normal para la pantalla. También puedes usar un adaptador de micro-HDMI a HDMI y un cable HDMI estándar de tamaño normal. Si utilizas un monitor sin una toma HDMI, puedes comprar adaptadores de micro-HDMI a DVI-D, DisplayPort o VGA. Para conectar un televisor más antiguo que utilice vídeo compuesto o tenga una toma SCART, usa un cable de audio/vídeo TRRS (punta-anillo-cuerpo) de 3,5 mm.

Raspberry Pi es seguro de usar sin una carcasa, siempre y cuando NO se coloque sobre una superficie metálica capaz de conducir electricidad y causar un cortocircuito. Pero el uso de una carcasa puede proporcionar protección adicional. El kit de escritorio incluye la carcasa oficial de Raspberry Pi y hay carcasas de otros fabricantes disponibles a través cualquier distribuidor acreditado.

Si quieres usar Raspberry Pi en una red con cable, en lugar de una inalámbrica (WiFi), también necesitarás un cable de red, que deberías conectar en un extremo del conmutador o el enrutador de la red. Si piensas usar la radio inalámbrica integrada de Raspberry Pi, no necesitarás un cable pero tendrás que saber el nombre y la clave o frase de contraseña de tu red inalámbrica.



INSTALACIÓN DE RASPBERRY PI 400

Las siguientes instrucciones son para instalar Raspberry Pi 4 u otro miembro de la familia Raspberry Pi sin carcasa. Las instrucciones para instalar Raspberry Pi 400 se encuentran en la página 32.

Instalar el hardware

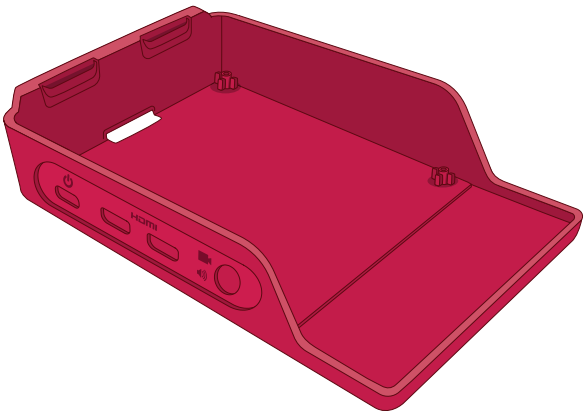
Saca Raspberry Pi del embalaje. Raspberry Pi es un hardware robusto, pero eso no significa que sea indestructible: acostúmbrate a sostener la placa por los bordes, sin poner los dedos en los lados planos; y ten mucho cuidado con los pines metálicos salientes. Si esos pines se doblan, dificultarán el uso de placas adicionales y otros componentes de hardware. Y, en el peor de los casos, incluso podrían causar un cortocircuito que estropearía Raspberry Pi.

Si aún no lo has hecho, consulta el **Capítulo 1, Introducción a Raspberry Pi** para ver dónde están exactamente los distintos puertos y qué hacen.

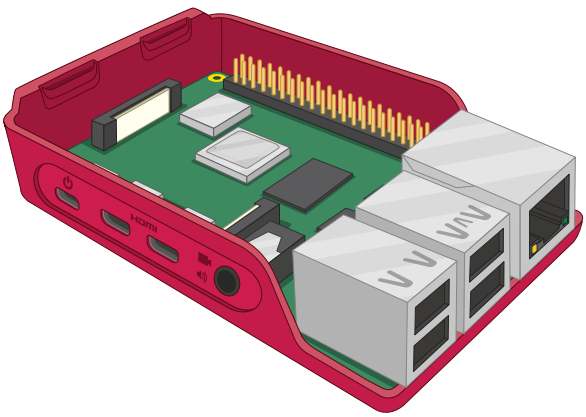
Montar la carcasa

Si vas a instalar Raspberry Pi en una carcasa, ese debería ser tu primer paso. Si utilizas la carcasa oficial de Raspberry Pi, empieza por separar las dos partes que la componen: la base roja y la tapa blanca.

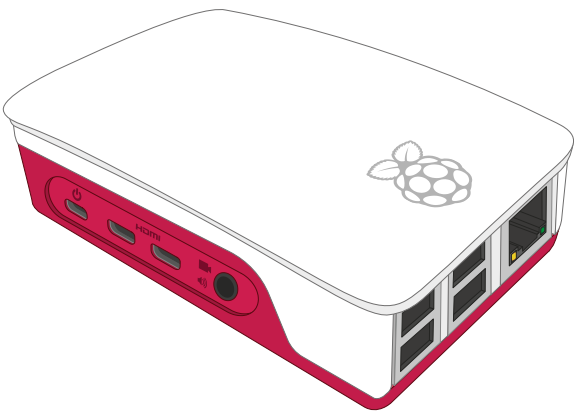
1 **Sujeta la base de manera que el extremo elevado esté a tu izquierda y el extremo bajo a tu derecha.**



2 **Sujetando Raspberry Pi (sin la tarjeta microSD insertada) por sus puertos USB y Ethernet, en un ligero ángulo, encaja los conectores (USB tipo C, 2 × micro HDMI y 3,5 mm) en los orificios correspondientes, en el lateral de la base, y luego baja suavemente el otro lado para que se asiente sobre la superficie.**

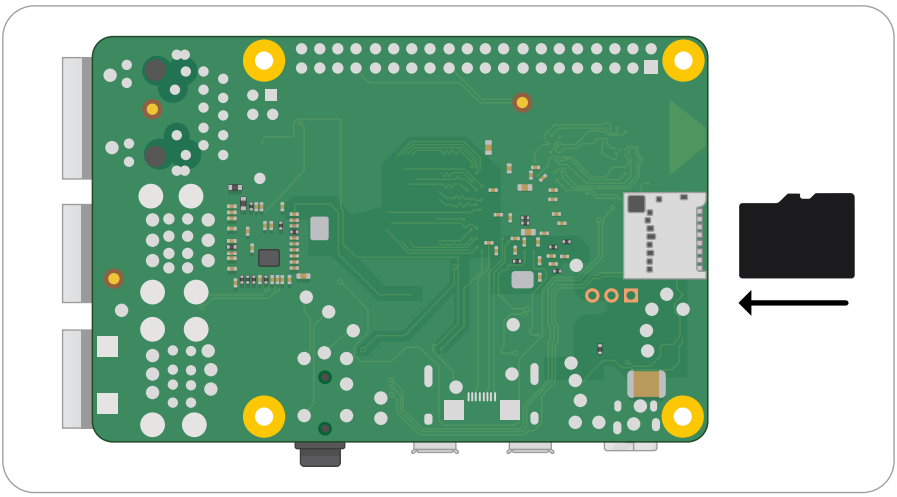


3 **Toma la tapa blanca y coloca los dos clips de la izquierda en los orificios correspondientes en la parte izquierda de la base, sobre la ranura de la tarjeta microSD. Cuando estén en su sitio, presiona sobre el lado derecho (sobre los puertos USB) hasta que oigas un clic.**

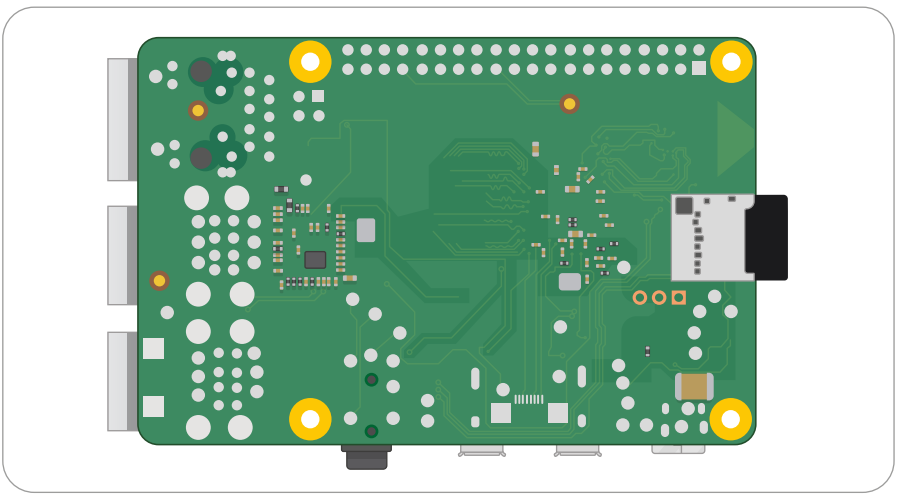


Conectar la tarjeta microSD

Para instalar la tarjeta microSD, que es el *almacén* de Raspberry Pi, gira Raspberry Pi (en la carcasa, si estás usando una) y desliza la tarjeta en la ranura microSD, con la etiqueta hacia el lado opuesto de Raspberry Pi. Solo puede entrar en un sentido y debería encajar en su sitio sin demasiada presión.



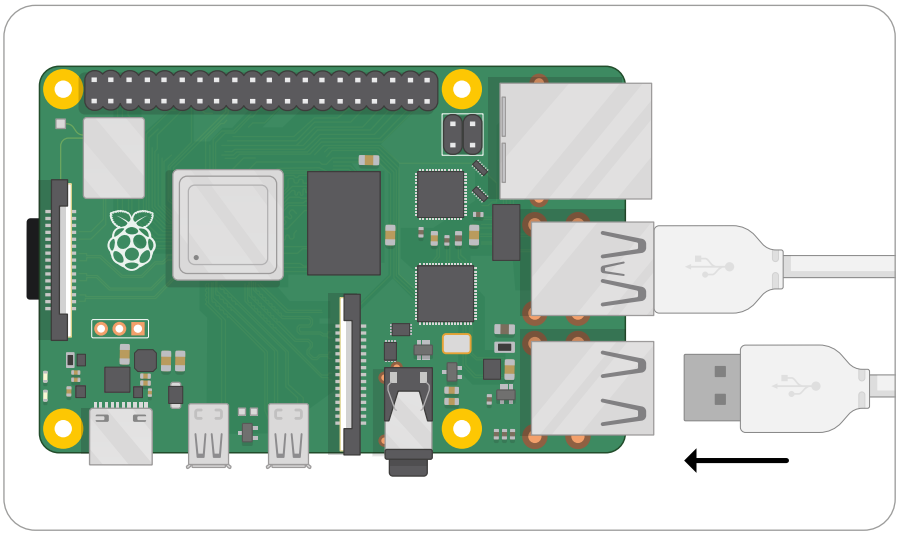
La tarjeta microSD se deslizará en el conector y se detendrá sin un clic.



Si quieres extraerla posteriormente, simplemente sujeta el extremo de la tarjeta y tira de ella suavemente. Si utilizas un modelo antiguo de Raspberry Pi, primero tendrás que empujar un poco la tarjeta para desbloquearla (esa acción no es necesaria con un Raspberry Pi 3 o 4).

Conectar un teclado y un ratón

Conecta el cable USB del teclado a cualquiera de los cuatro puertos USB (2.0 o 3.0) de Raspberry Pi. Si utilizas el teclado oficial de Raspberry Pi, hay un puerto USB en la parte posterior para el ratón. Si utilizas otro teclado, conecta el cable USB de tu ratón a otro puerto USB de Raspberry Pi.



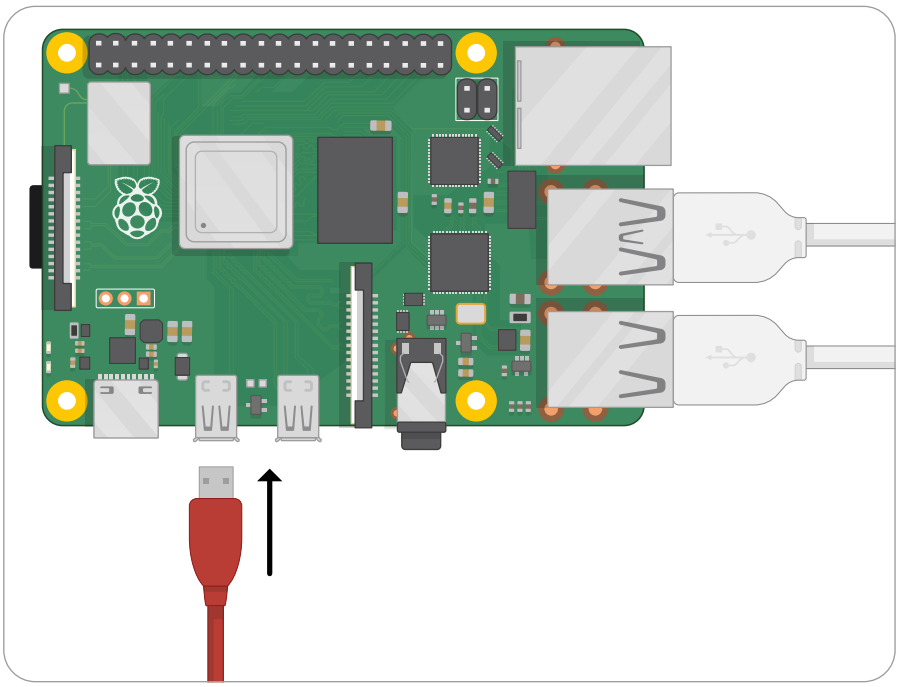
Los conectores USB para el teclado y el ratón deberían deslizarse sin demasiada presión para encajar en su sitio. Si tienes que forzar el conector, es señal de que hay algún problema. ¡Comprueba si la posición del conector USB es correcta!

TECLADO Y RATÓN

El teclado y el ratón actúan son el medio principal para decirle a Raspberry Pi lo que tiene que hacer; en informática, se conocen como *dispositivos de entrada*, a diferencia de la pantalla que es un *dispositivo de salida*.

Conectar una pantalla

Conecta el extremo más pequeño del cable micro-HDMI al puerto micro-HDMI más cercano al puerto USB tipo C de Raspberry Pi. Conecta el otro extremo a tu pantalla. Si la pantalla tiene más de un puerto HDMI, busca un número de puerto junto al conector: tendrás que asignar el televisor a esta entrada para ver la pantalla de Raspberry Pi. Si no ves un número de puerto, no te preocupes: puedes ir pasando de una entrada a otra hasta que encuentres Raspberry Pi.

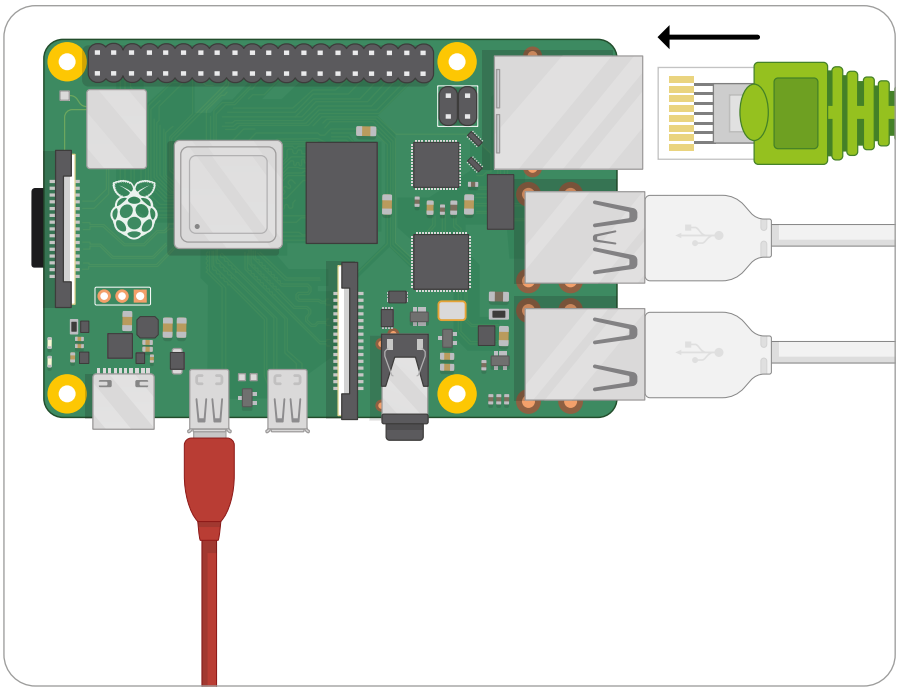


CONEXIÓN DE TV

Si tu televisor o monitor no tiene un conector HDMI, eso no significa que no puedas usar Raspberry Pi. Con cables adaptadores, disponibles en cualquier tienda de electrónica, podrás convertir el puerto micro-HDMI de Raspberry Pi a DVI-D, DisplayPort o VGA para usarlo con monitores de ordenadores más antiguos, que se conectan al puerto micro-HDMI de Raspberry Pi. Luego se utiliza un cable adecuado para conectar el cable adaptador al monitor. Si tu televisor solo tiene una entrada de vídeo compuesto o SCART, puedes comprar cables adaptadores de tipo TRRS y adaptadores de vídeo compuesto a SCART que se conectan a la toma AV de 3,5 mm.

Conectar un cable de red (opcional)

Para conectar Raspberry Pi a una red con cable, usa un cable de red (conocido como cable Ethernet) y conéctalo en el puerto Ethernet de Raspberry Pi, con el clip de plástico hacia abajo. Se debería oír un clic. Si tienes que desconectar el cable, empuja el clip de plástico hacia el conector y retira suavemente el cable.

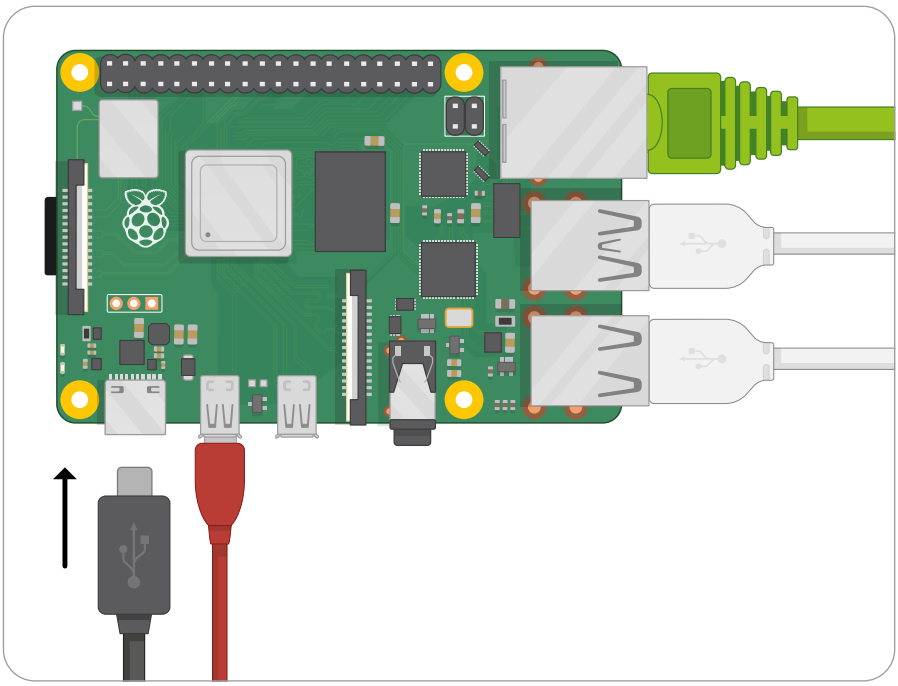


El otro extremo del cable de red debe conectarse a cualquier puerto libre del concentrador, conmutador o enrutador de la red mediante el mismo procedimiento.

Conectar una fuente de alimentación

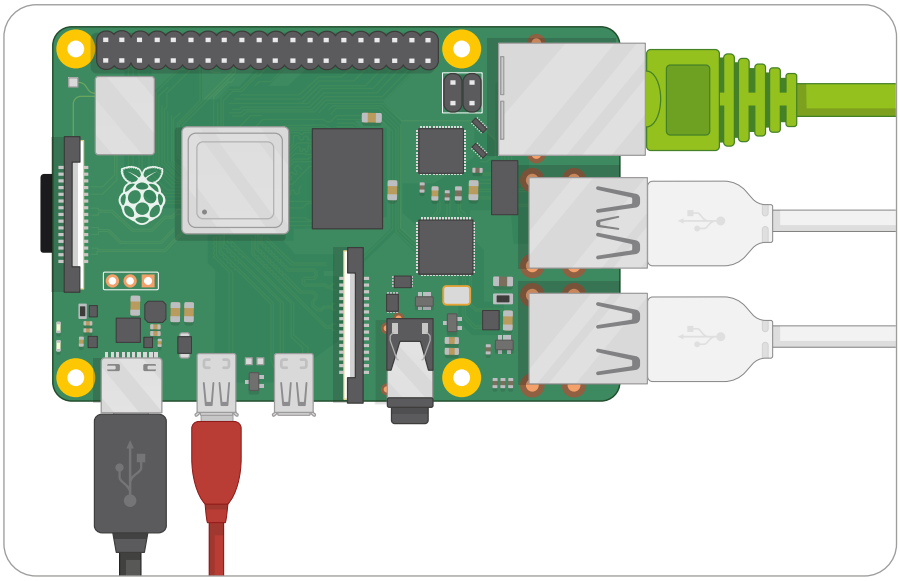
La conexión de Raspberry Pi a una fuente de alimentación es el último paso en el proceso de instalación del hardware, y algo que solo debes hacer cuando tengas todo listo para empezar a configurar el software: Raspberry Pi no tiene un interruptor de encendido y se encenderá en cuanto se conecte a una fuente de alimentación con corriente.

Primero, conecta el extremo USB tipo C del cable de alimentación al conector de alimentación USB tipo C de Raspberry Pi. Puede ir en cualquier dirección y debería encajar fácilmente. Si la fuente de alimentación tiene un cable extraíble, asegúrate de que el otro extremo esté conectado al cuerpo de la fuente de alimentación.

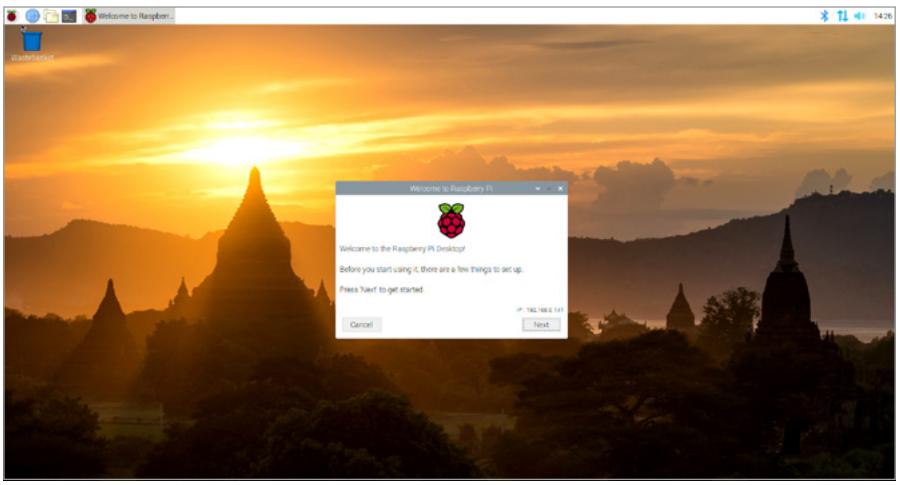


Por último, conecta la fuente de alimentación a una toma de corriente y enciende la toma: Raspberry Pi entrará en funcionamiento inmediatamente.

Enhorabuena: ya has montado tu Raspberry Pi.



Verás brevemente cuatro logotipos de Raspberry Pi en la parte superior izquierda de una pantalla negra y puede que aparezca una pantalla azul a medida que el software se redimensiona para aprovechar al máximo la tarjeta microSD. Si ves una pantalla negra, espera unos minutos; la primera vez que Raspberry Pi se inicia, tiene que realizar algunas tareas en segundo plano. Al cabo de un rato verás el escritorio y el asistente de configuración del sistema operativo Raspberry Pi, como se muestra en la **Figura 2-1**. El sistema operativo está listo para que lo configures, procedimiento que aprenderás en el **Capítulo 3, Uso de Raspberry Pi**.



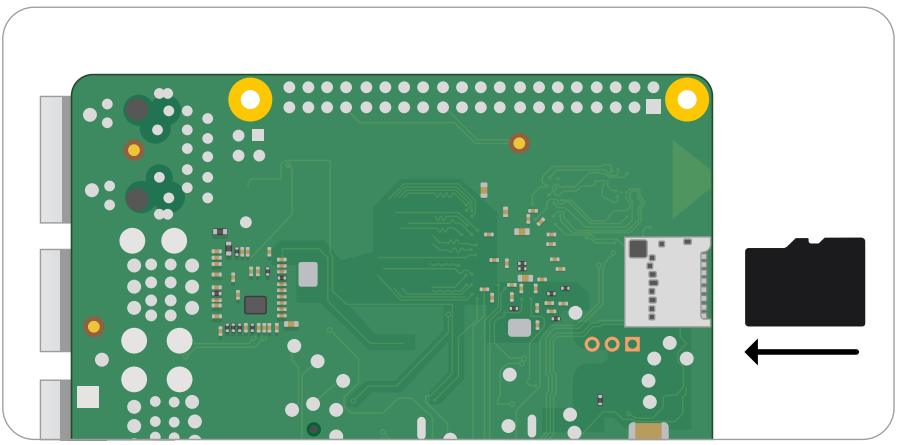
▲ **Figura 2-1:** El escritorio y el asistente de configuración del sistema operativo Raspberry Pi

Instalar Raspberry Pi 400

A diferencia de Raspberry Pi 4, Raspberry Pi 400 viene con un teclado incorporado y la tarjeta microSD ya instalada. Tendrás que conectar algunos cables para empezar, pero solo tardarás unos minutos.

Conectar un ratón

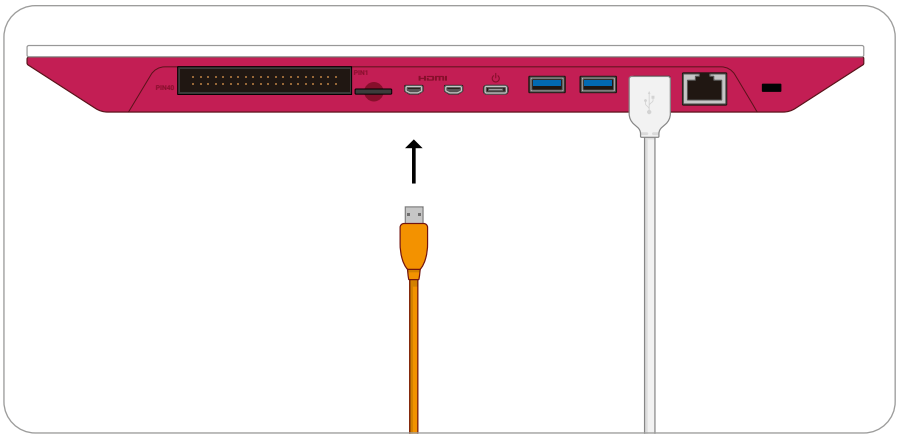
El teclado de Raspberry Pi 400 ya está conectado, solo tienes que añadir el ratón. Toma el cable USB del extremo del ratón e insértalo en cualquiera de los tres puertos USB (2.0 o 3.0) de la parte posterior de Raspberry Pi 400. Si quieres reservar los dos puertos USB 3.0 de alta velocidad para otros accesorios, usa el puerto blanco.



El conector USB debería encajar en su sitio sin demasiada presión. Si tienes que forzar el conector, es señal de que hay algún problema. ¡Comprueba si la posición del conector USB es correcta!

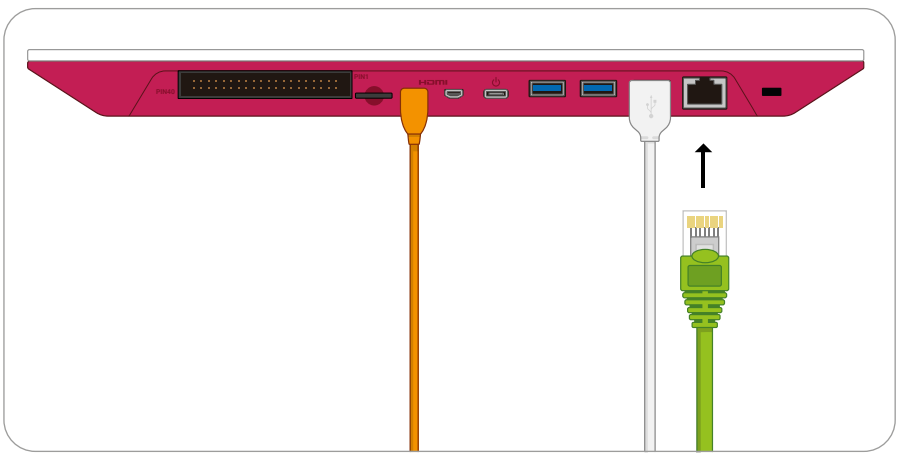
Conectar una pantalla

Conecta el extremo más pequeño del cable micro-HDMI al puerto micro-HDMI más cercano a la ranura microSD de Raspberry Pi 400. Conecta el otro extremo a tu pantalla. Si la pantalla tiene más de un puerto HDMI, busca un número de puerto junto al conector: tendrás que asignar el televisor a esta entrada para ver la pantalla de Raspberry Pi. Si no ves un número de puerto, no te preocupes: puedes ir pasando de una entrada a otra hasta que encuentres Raspberry Pi.



Conectar un cable de red (opcional)

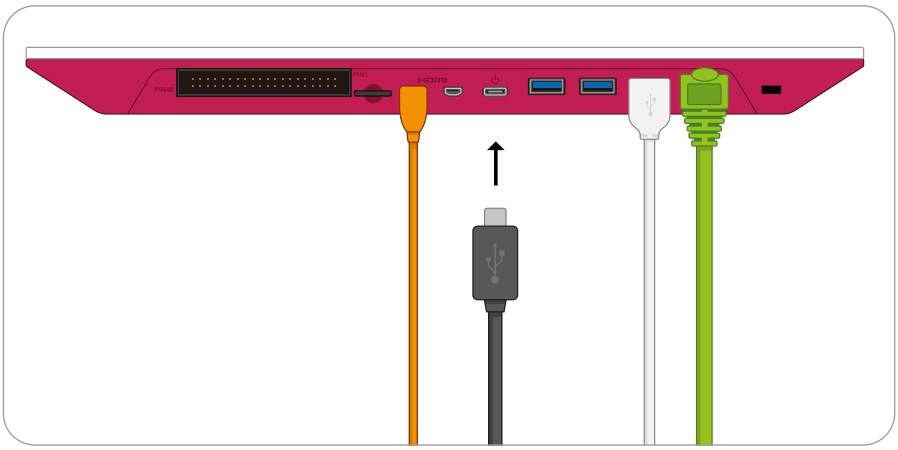
Para conectar Raspberry Pi 400 a una red con cable, usa un cable de red (conocido como cable Ethernet) y conéctalo al puerto Ethernet de Raspberry Pi 400, con el clip de plástico hacia arriba. Se debería oír un clic. Si tienes que desconectar el cable, empuja el clip de plástico hacia el conector y retira suavemente el cable.



El otro extremo del cable de red debe conectarse a cualquier puerto libre del concentrador, conmutador o enrutador de la red mediante el mismo procedimiento.

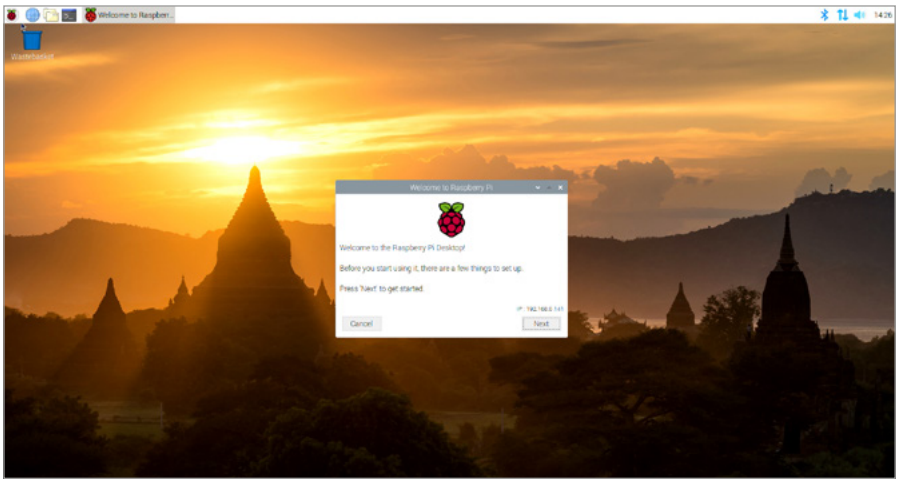
Conectar una fuente de alimentación

La conexión de Raspberry Pi 400 a una fuente de alimentación es el último paso en el proceso de instalación del hardware, y algo que solo debes hacer cuando tengas todo listo para empezar a configurar el software: Raspberry Pi 400 no tiene un interruptor de encendido y se encenderá en cuanto se conecte a una fuente de alimentación con corriente. Primero, conecta el extremo USB tipo C del cable de alimentación al conector de alimentación USB tipo C de Raspberry Pi. Puede ir en cualquier dirección y debería encajar fácilmente. Si la fuente de alimentación tiene un cable extraíble, asegúrate de que el otro extremo esté conectado al cuerpo de la fuente de alimentación.



Por último, conecta la fuente de alimentación a una toma de corriente y enciende la toma: Raspberry Pi 400 entrará en funcionamiento inmediatamente. Enhorabuena: ya has montado tu Raspberry Pi 400.

Verás brevemente cuatro logotipos de Raspberry Pi en la parte superior izquierda de una pantalla negra y puede que aparezca una pantalla azul a medida que el software se redimensiona para aprovechar al máximo la tarjeta microSD. Si ves una pantalla negra, espera unos minutos; la primera vez que Raspberry Pi se inicia, tiene que realizar algunas tareas en segundo plano. Al cabo de un rato verás el escritorio y el asistente de configuración del sistema operativo Raspberry Pi, como se muestra en la **Figura 2-2**. El sistema operativo está listo para que lo configures, procedimiento que aprenderás en el **Capítulo 3, Uso de Raspberry Pi**.

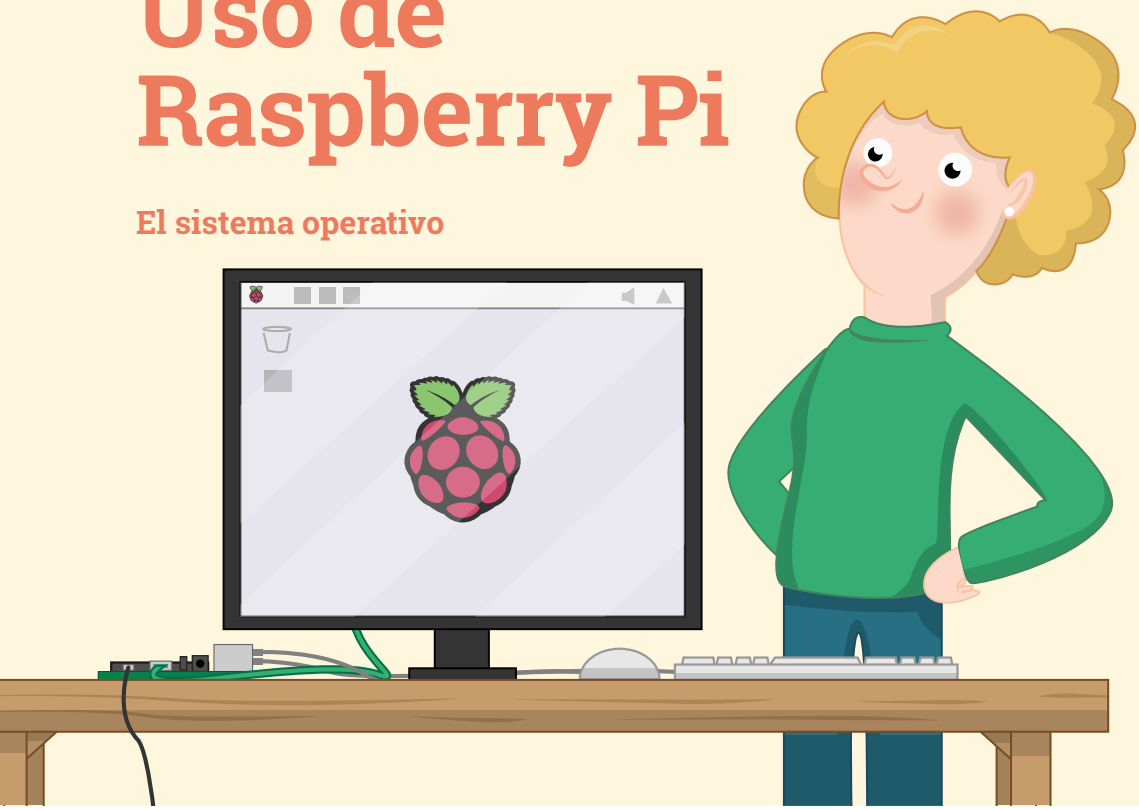


▲ **Figura 2-2:** El escritorio y el asistente de configuración del sistema operativo Raspberry Pi

Capítulo 3

Uso de Raspberry Pi

El sistema operativo



Raspberry Pi es capaz de ejecutar una amplia gama de programas de software y diversos sistemas operativos, que son el software principal que hace que un ordenador funcione. El más popular de estos es el sistema operativo Raspberry Pi OS, que es el oficial de la Fundación Raspberry Pi. Se basa en Debian Linux y se ha adaptado especialmente para Raspberry Pi. Se suministra listo para usar, con varios extras ya preinstalados.

Puede que antes solo hayas usado Microsoft Windows o Apple macOS, pero no te preocupes: Raspberry Pi OS utiliza el mismo tipo de iconos, menús y punteros (WIMP) que Windows y no tardarás en acostumbrarte. El siguiente capítulo te ayudará a empezar y conocerás parte del software integrado.

El asistente de bienvenida

La primera vez que ejecutes Raspberry Pi OS, verás el asistente de bienvenida (**Figura 3-1**), una herramienta muy útil que te guiará para cambiar opciones del sistema (lo que llamamos *configuración*) según cómo y dónde vayas a usar Raspberry Pi.

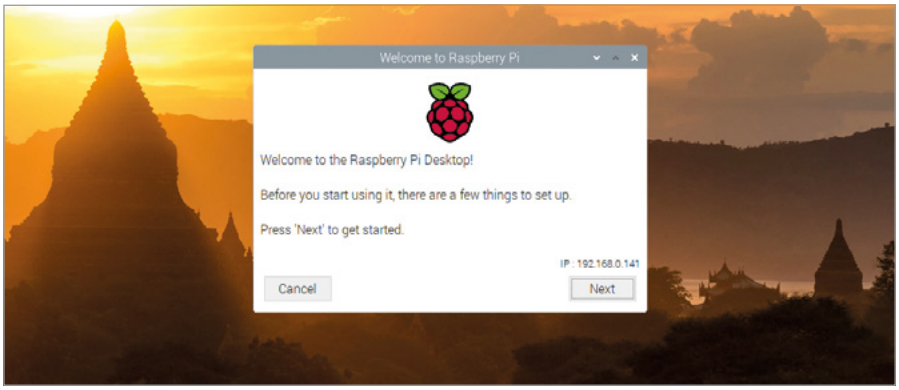
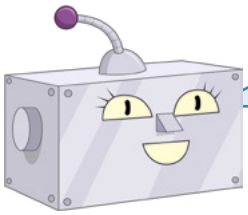


Figura 3-1: El asistente de bienvenida



CIERRE DEL ASISTENTE

Si quieres, puedes cerrar el asistente de bienvenida haciendo clic en el botón Cancelar, pero algunas funciones de Raspberry Pi (como la red inalámbrica) no funcionarán hasta que respondas al menos la primera serie de preguntas.

Haz clic en el botón Next, elige tu país, tu idioma y tu zona horaria haciendo clic en cada cuadro desplegable para seleccionar la opción que quieras en la lista (**Figura 3-2**). Si utilizas un teclado con distribución inglesa (EE.UU.), haz clic en la casilla de verificación para asegurarte de que Raspberry Pi OS use la opción correcta. Si quieres que la interfaz del escritorio y los programas se vean en inglés, sea cual sea tu idioma nativo, haz clic en la casilla "Use English language" para seleccionarla. Cuando termines, haz clic en Next.

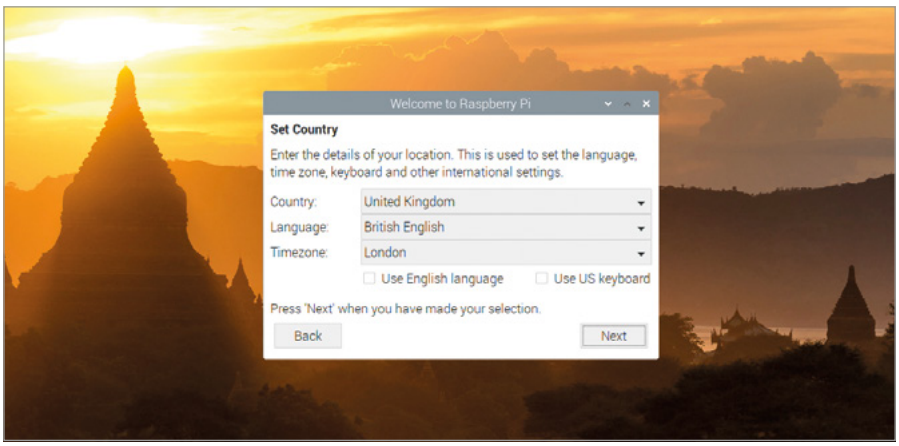


Figura 3-2: Seleccionar un idioma, entre otras opciones

En la siguiente pantalla se te pedirá que cambies la contraseña de usuario de "pi" (la predeterminada es "raspberry"). Por motivos de seguridad, es muy conveniente crear una nueva. Introdúcela en los cuadros (**Figura 3-3**). Cuando tengas la contraseña que quieres, haz clic en Next.

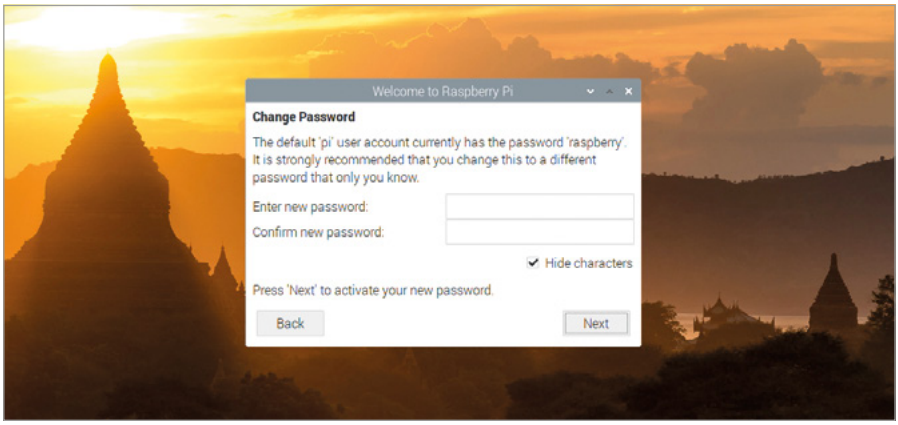


Figura 3-3: Establecer una contraseña nueva

En la siguiente pantalla se puede configurar si quieres tener un borde negro alrededor de la pantalla (**Figura 3-4)**. Si el escritorio de Raspberry Pi ocupa todo el espacio de tu TV o monitor, deja la casilla sin marcar. Si tiene un borde negro y es más pequeño que la TV o el monitor, marca la casilla. Luego haz clic en Next.

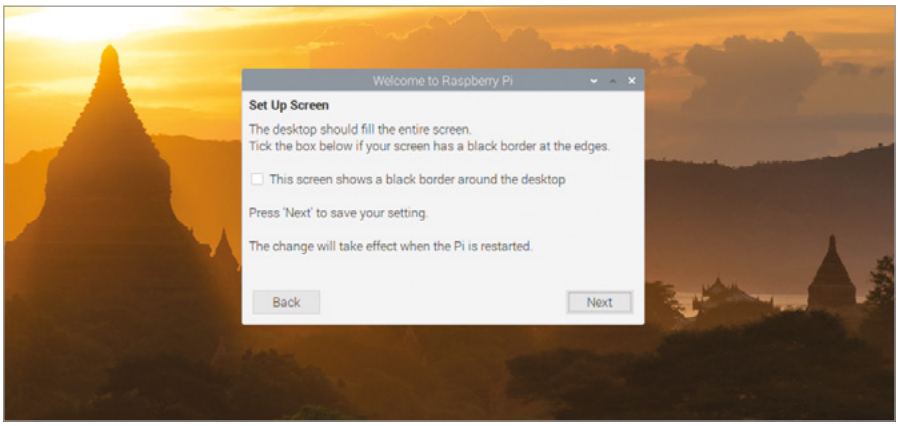


Figura 3-4: Comprobar que no haya un borde negro

En la siguiente pantalla podrás elegir tu red WiFi en una lista (**Figura 3-5**). Recorre la lista de redes usando el ratón o el teclado y cuando encuentres el nombre de tu red, haz clic en ella y luego en Next. Si tu red inalámbrica es segura (debería serlo), se te pedirá tu contraseña (también denominada clave previamente compartida), que suele figurar en una tarjeta con el

RED INALÁMBRICA

La red inalámbrica integrada solo está disponible en Raspberry Pi 3, Pi 4 y Pi Zero W. Si quieres usar otro modelo de Raspberry Pi con una red inalámbrica, necesitarás un adaptador USB para WiFi.

router o en la base de este. Haz clic en Next para conectarte a la red. Si no quieres conectarte a una red inalámbrica, haz clic en Skip.

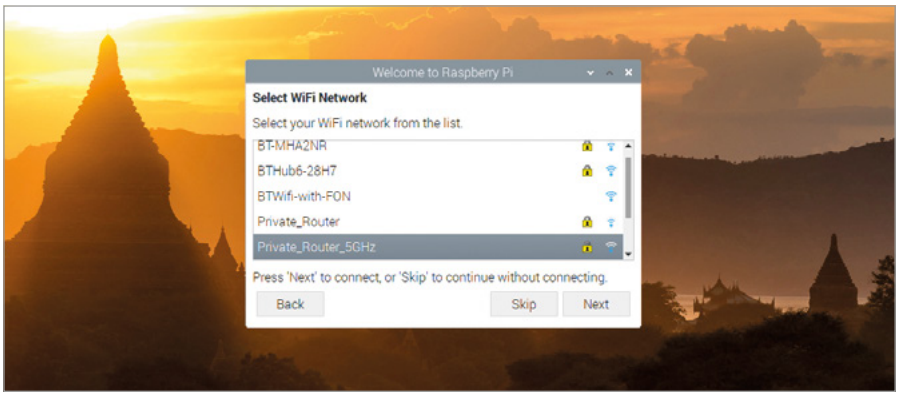


Figura 3-5: Elegir una red inalámbrica

La siguiente pantalla te permitirá buscar e instalar actualizaciones para Raspberry Pi OS y los otros programas de software de Raspberry Pi (**Figura 3-6**). Raspberry Pi OS se actualiza periódicamente para corregir errores, agregar funciones y mejorar el rendimiento. Para instalar las actualizaciones, haz clic en Next. Si no quieres instalarlas, haz clic en Skip. La descarga de las actualizaciones puede tardar varios minutos, así que ten paciencia. Después de instalarse las actualizaciones verás el mensaje "System is up to date". Haz clic en el botón OK.

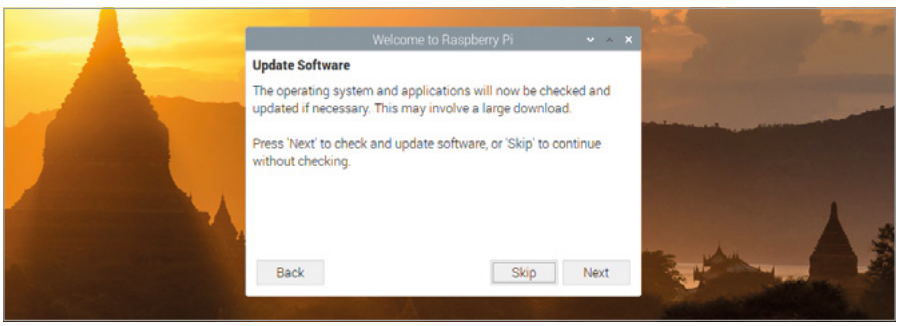
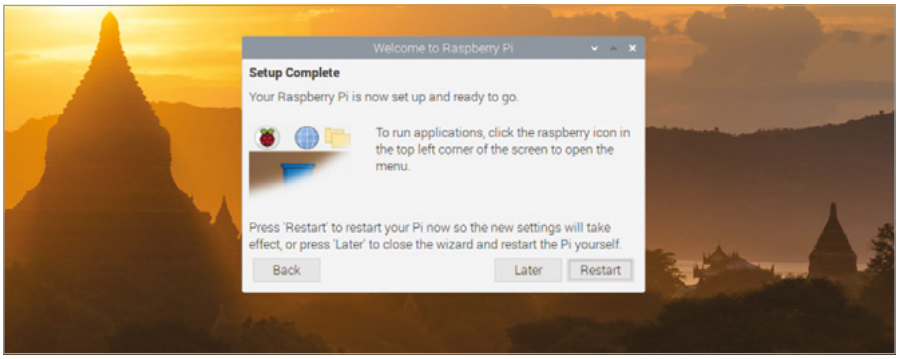


Figura 3-6: Buscar actualizaciones

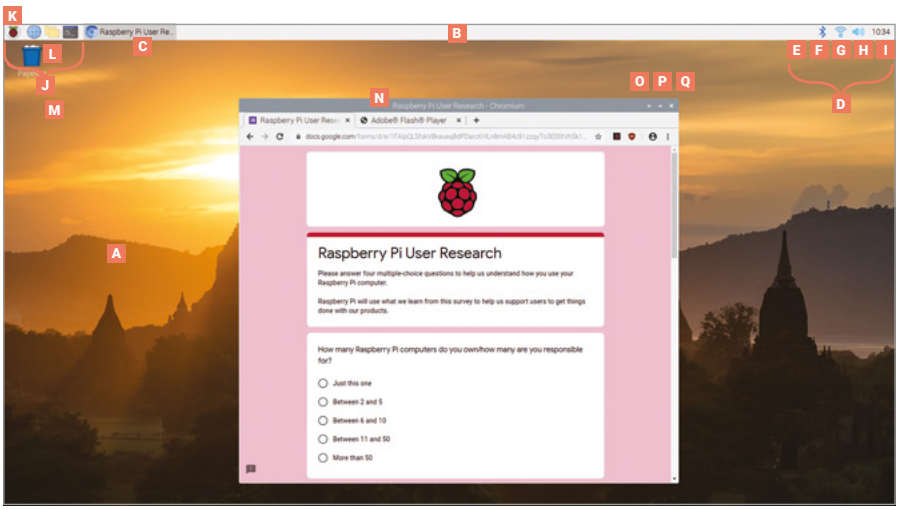
En pantalla final del asistente de bienvenida (**Figura 3-7**) hay una tarea sencilla: ciertos cambios realizados solo entrarán en vigor después de reiniciar Raspberry Pi. Si se te indica, haz clic en el botón Restart. Raspberry Pi se reiniciará. Esta vez no volverá a aparecer el asistente de bienvenida. Ya tienes todo listo para empezar a usar Raspberry Pi.



▲ **Figura 3-7:** Reiniciar Raspberry Pi

El escritorio

La versión del sistema operativo Raspberry Pi OS instalada en la mayoría de las placas de Raspberry Pi se denomina "Raspberry Pi OS con escritorio", en alusión a la principal interfaz gráfica de usuario (**Figura 3-8**). La mayor parte de este escritorio la ocupa una imagen conocida como papel tapiz o fondo de pantalla (**A** en la **Figura 3-8**), sobre la que aparecerán los programas que ejecutes. En la parte superior del escritorio hay una barra de tareas (**B**), para cargar cada uno de los programas, que se identificarán como tareas (**C**) en la barra de tareas.



▲ **Figura 3-8:** El escritorio del sistema operativo Raspberry Pi OS

- **A** Fondo de pantalla
- **B** Barra de tareas
- **C** Tarea
- **D** Bandeja del sistema
- **E** Expulsión de dispositivos
- **F** Icono de Bluetooth
- **G** Icono de red
- **H** Icono de volumen
- **I** Reloj
- **J** Iniciador
- **K** Icono de menús (o Raspberry)
- **L** Icono de papelera
- **M** Icono de unidad extraíble
- **N** Barra de título de la ventana
- **O** Minimizar
- **P** Maximizar
- **Q** Cerrar

En la parte derecha de la barra de menús está la *bandeja del sistema* (**D**). Si tienes conectados a Raspberry Pi dispositivos de *almacenamiento extraíbles* (por ejemplo unidades de memoria flash USB), verás un símbolo de expulsión (**E**); al hacer clic en él podrás extraer de modo seguro ese tipo de dispositivos. En el extremo derecho está el reloj (**I**). Haz clic en él para abrir un calendario digital (**Figura 3-9**).

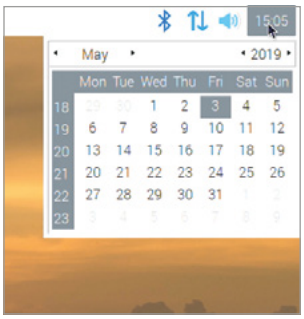


Figura 3-9: El calendario digital

Junto al calendario verás un icono de altavoz (**H**). Haz clic en él con el botón izquierdo del ratón para ajustar el volumen de audio de Raspberry Pi. O utiliza el botón derecho si quieres elegir la salida de audio que debe utilizar Raspberry Pi. Junto al altavoz está el icono de red (**G**). Si tienes conexión a una red inalámbrica, verás la intensidad de la señal de esa red representada por una serie de barras. Si la conexión es a una red con cable, se verán dos flechas. Al hacer clic en el icono de red aparecerá una lista de redes inalámbricas cercanas (**Figura 3-10**). Y si haces clic en el icono de Bluetooth (**F**) adyacente podrás conectarte a un dispositivo Bluetooth cercano.

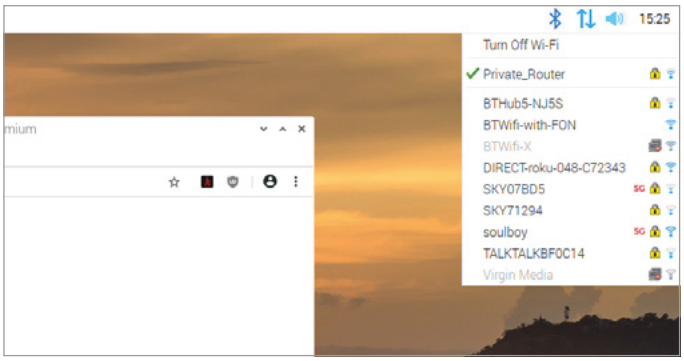


Figura 3-10: Listado de redes inalámbricas

En el lado izquierdo de la barra de menús se encuentra el *iniciador de aplicaciones* (**J**); ahí encontrarás los programas instalados junto con el sistema operativo Raspberry Pi. Algunos de ellos son visibles como iconos de acceso directo y otros están ocultos en el menú, que puedes abrir haciendo clic en el icono de Raspberry (**K**), en el extremo izquierdo, con forma de frambuesa porque eso es lo que significa raspberry en inglés (**Figura 3-11**).

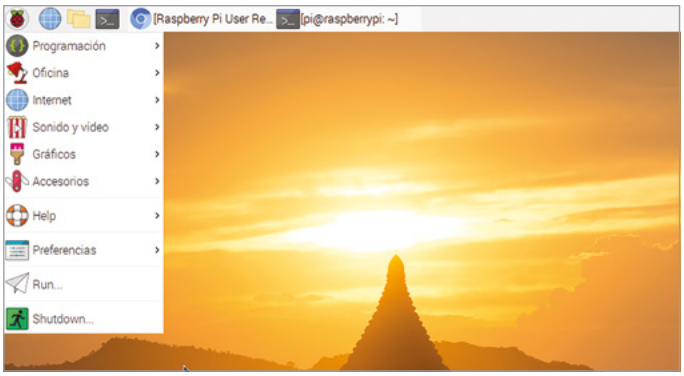


Figura 3-11: El menú del sistema operativo Raspberry Pi OS

Los programas en el menú se dividen en categorías con nombres representativos: la categoría Programación, por ejemplo, contiene software diseñado para ayudarte a escribir tus propios programas (como se explica a partir del **Capítulo 4, Programación con Scratch**) mientras que Juegos te ayudará a pasar el rato. En esta guía no se detallan todos los programas. Experimenta a tu aire para aprender más.

El navegador web Chromium

Para practicar con tu Raspberry Pi, carga el navegador web Chromium: haz clic en el icono de Raspberry, en la esquina superior izquierda, para abrir el menú. Mueve el puntero del ratón para seleccionar la categoría Internet y haz clic en Navegador web Chromium para cargarlo (**Figura 3-12**).

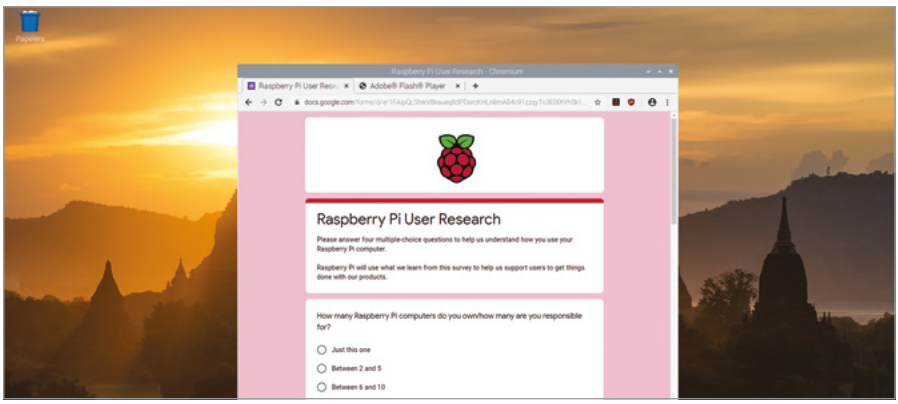


Figura 3-12: El navegador web Chromium

Si has utilizado el navegador Chrome de Google en otro equipo, Chromium te resultará familiar. Al igual que otros navegadores web, Chromium te permite visitar sitios web, reproducir vídeos, jugar y comunicarte con gente de todo el mundo en foros y sitios de chat.

Maximiza la ventana de Chromium para que ocupe más espacio en pantalla. En la parte superior derecha de la barra de título de la ventana de Chromium verás tres iconos (**N**): haz clic en el del medio, el de la flecha que apunta hacia arriba (**P**). Este es el botón para *maximizar*, para que la ventana ocupe toda la pantalla. A la izquierda del icono de maximizar está el de *minimizar* (**O**), que ocultará una ventana hasta que hagas clic en ella en la barra de tareas de la parte superior de la pantalla. La x a la derecha del icono de maximizar es el icono de *cerrar* (**Q**), para cerrar la ventana.

CERRAR Y GUARDAR

No deberías cerrar una ventana antes de haber guardado cualquier trabajo realizado. Aunque muchos programas te advertirán que guardes tu trabajo cuando hagas clic en el botón de cerrar, otros no lo harán.

Haz clic en la barra de direcciones en la parte superior de la ventana de Chromium (el cuadro blanco alargado con el icono de una lupa en la esquina izquierda), escribe **www.raspberrypi.org** y pulsa la tecla **ENTRAR** en el teclado. Así se cargará el sitio web de Raspberry Pi (**Figura 3-13**). En esa barra también puedes escribir texto para hacer búsquedas: prueba con "Raspberry Pi", "Raspberry Pi OS" o "informática para educación".

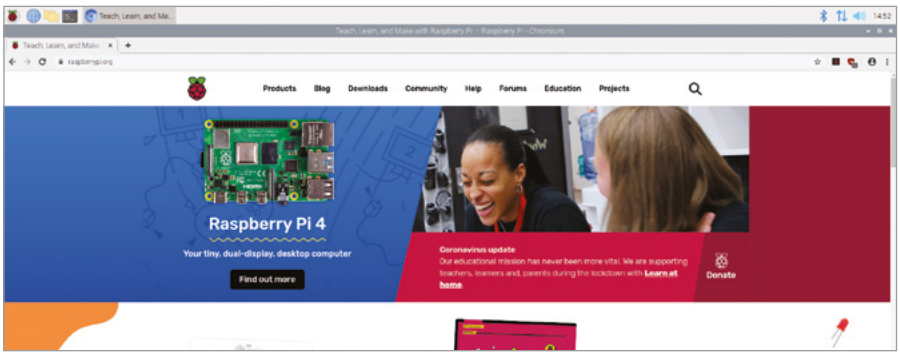


Figura 3-13: Cargar el sitio web de Raspberry Pi en Chromium

Al cargar Chromium por primera vez, podrían aparecer varias *pestañas* en la parte superior de la ventana. Para abrir una pestaña, haz clic en ella. Para cerrar una pestaña sin cerrar el navegador, haz clic en la x de la esquina derecha de esa pestaña. Para abrir una nueva pestaña (algo muy práctico para tener varios sitios web abiertos sin multiplicar innecesariamente el número de ventanas de Chromium) puedes hacer clic en el botón de pestaña situado a la derecha de la última de la lista o mantener pulsada la tecla **CTRL** en el teclado y pulsar la tecla **T** antes de soltar la tecla **CTRL**.

Cuando termines de usar Chromium, haz clic en el botón de cerrar, en la esquina superior derecha de la ventana.

El Gestor de archivos

Todos los archivos que guardes (programas, vídeos, imágenes, etc.) irán a tu *directorio de inicio*. Para verlo, vuelve a hacer clic en el icono de Raspberry para abrir el menú, mueve el puntero del ratón para seleccionar Accesorios y haz clic en Gestor de archivos para cargarlo (**Figura 3-14**).

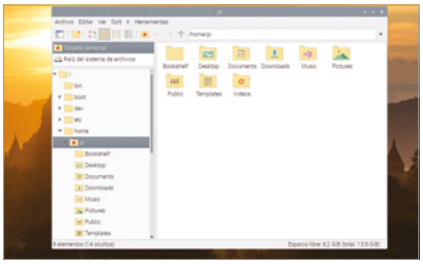


Figura 3-14:
El Gestor de archivos

El Gestor de archivos te permite navegar por los archivos y las carpetas o *directorios* en la tarjeta microSD de Raspberry Pi, así como en cualquier dispositivo de almacenamiento extraíble (por ejemplo, unidades flash USB) que conectes a los puertos USB de Raspberry Pi. Cuando lo abras por primera vez, irá automáticamente a tu directorio de inicio. Ahí encontrarás otras carpetas, denominadas *subdirectorios*, que se organizan por categorías, igual que el menú. Los subdirectorios principales son:

- **Bookshelf:** Contiene copias digitales de libros, revistas y otras publicaciones de Raspberry Pi Press, entre ellas una copia de esta *guía para principiantes*. Puedes leer esas publicaciones y descargar más con la aplicación Bookshelf, que encontrarás en la sección de ayuda del menú.
- **Desktop:** Esta carpeta es lo primero que ves al cargar por primera vez el sistema operativo Raspberry Pi. Si guardas un archivo aquí, aparecerá en el escritorio, lo que facilitará su búsqueda y carga.
- **Documents:** Aquí se guardarán la mayoría de los archivos que crees, ya sean cuentos, recetas, etc.
- **Downloads:** Cuando descargas un archivo de Internet con el navegador web Chromium, se guarda automáticamente en Downloads.
- **Music:** Aquí se puede guardar cualquier música que crees o pongas en Raspberry Pi.
- **Pictures:** Esta carpeta está destinada a imágenes o, en términos técnicos, *archivos de imagen*.
- **Public:** Aunque la mayoría de tus archivos son privados, cualquier archivo que guardes en la carpeta Public estará disponible para otros usuarios de Raspberry Pi, incluso si tienen su propio nombre de usuario y contraseña.

- **Templates:** Esta carpeta contiene todas las plantillas, documentos en blanco con una estructura o un diseño básicos ya existentes, que hayas creado o que instalen tus aplicaciones.
- **Videos:** Una carpeta para vídeos y la primera en la que buscarán la mayoría de los programas de reproducción de vídeo.

La ventana Gestor de archivos se divide en dos paneles: el izquierdo muestra los directorios de tu Raspberry Pi y el derecho los archivos y subdirectorios del directorio que selecciones en el panel izquierdo. Si conectas un dispositivo de almacenamiento extraíble al puerto USB de Raspberry Pi, aparecerá una ventana en la que indicar si quieres abrirlo en el Gestor de archivos (**Figura 3-15**). Si haces clic en Aceptar podrás ver tus archivos y directorios.

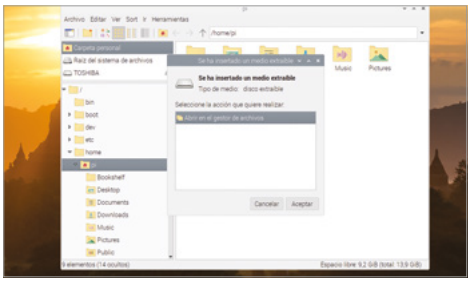


Figura 3-15: Insertar un dispositivo de almacenamiento extraíble

Es fácil copiar archivos entre la tarjeta microSD de Raspberry Pi y un dispositivo extraíble: teniendo abiertos tanto el directorio de inicio como el dispositivo extraíble, en ventanas distintas del Gestor de archivos, mueve el puntero del ratón al archivo que quieras copiar, haz clic y mantén pulsado el botón izquierdo del ratón, desliza el puntero a la otra ventana y suelta el botón (**Figura 3-16** a continuación). Esta es la técnica de *arrastrar y colocar*.

Otro método consiste en hacer clic una vez en el archivo, y luego hacer clic en el menú Editar, luego en Copiar, luego en la otra ventana y ahí, en el menú Editar, en Pegar.

La opción Cortar, también disponible en el menú Editar, es similar, excepto que el archivo de la ubicación original se elimina después de crear la copia. Ambas opciones también se pueden utilizar a través de los métodos abreviados de teclado **CTRL+C** (copiar) o **CTRL+X** (cortar), y pegar con **CTRL+V**.

MÉTODOS ABREVIADOS DE TECLADO

Cuando veas un método abreviado como **CTRL+C**, indica que debes mantener pulsada la primera tecla (**CTRL**), pulsar la segunda tecla (**C**) y luego soltar ambas.

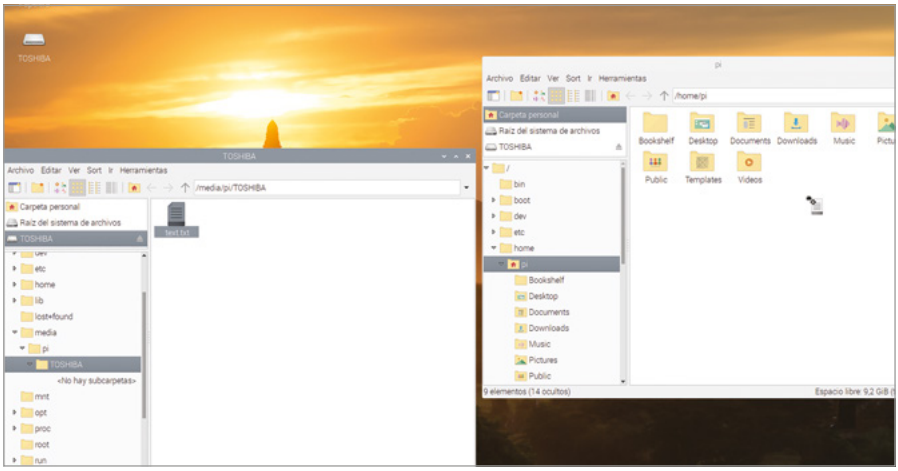


Figura 3-16: Arrastrar y colocar un archivo

Cuando termines de experimentar, cierra el Gestor de archivos haciendo clic en el botón x de la parte superior izquierda de la ventana. Si tienes abierta más de una ventana, ciérralas todas. Si tenías conectado a Raspberry Pi a un dispositivo de almacenamiento extraíble y quieres desconectarlo: haz clic en el botón de expulsión en la parte superior derecha de la pantalla, búscalo en la lista y haz clic en él antes de extraerlo.

EXTRACCIÓN DE DISPOSITIVOS

Utiliza siempre el botón de expulsión antes de quitar un dispositivo de almacenamiento externo. De lo contrario, los archivos que contiene podrían dañarse o quedar inutilizables.

La suite de productividad LibreOffice

Para ver más cosas posibles con Raspberry Pi, haz clic en el icono de menús de Raspberry, mueve el puntero del ratón a Oficina y haz clic en LibreOffice Writer. Se cargará el procesador de texto de LibreOffice (**Figura 3-17**), una popular *suite de productividad*. Si has usado Microsoft Office o Google Docs, has usado un paquete de productividad.

Nota: puede que LibreOffice no esté instalado de modo predeterminado en algunos sistemas Raspberry Pi; en ese caso, usa la herramienta Recommended Software (consulta la página 43) para instalarlo.

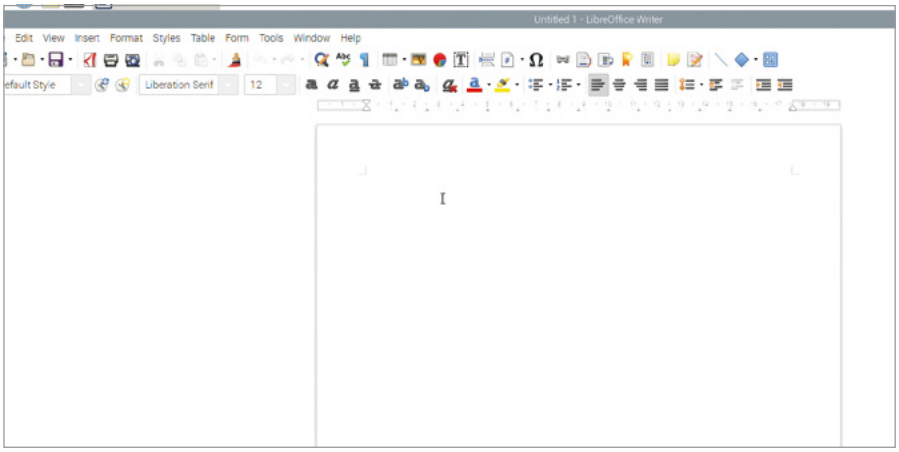


Figura 3-17: El programa LibreOffice Writer

Además de permitirte escribir documentos, un procesador de texto también te permite formatearlos de manera inteligente: puedes cambiar el estilo de letra, el color, el tamaño, añadir efectos e incluso insertar imágenes, gráficos, tablas y otros contenidos. Además puedes comprobar si has cometido errores: las erratas y los problemas gramaticales se resaltan en rojo y en verde respectivamente.

Para empezar, escribe un párrafo sobre lo que has aprendido hasta ahora de Raspberry Pi y su software. Experimenta con los iconos disponibles en la parte superior de la ventana para ver qué hacen: igual te permiten aumentar el tamaño de las letras o cambiar su color. Si tienes dudas de cómo hacerlo, simplemente pasa el puntero del ratón por encima de cada icono para ver una descripción de lo que hace. Cuando quieras, haz clic en el menú File y luego en la opción Save para guardar tu trabajo (**Figura 3-18**). Asígnale un nombre y haz clic en el botón Save.

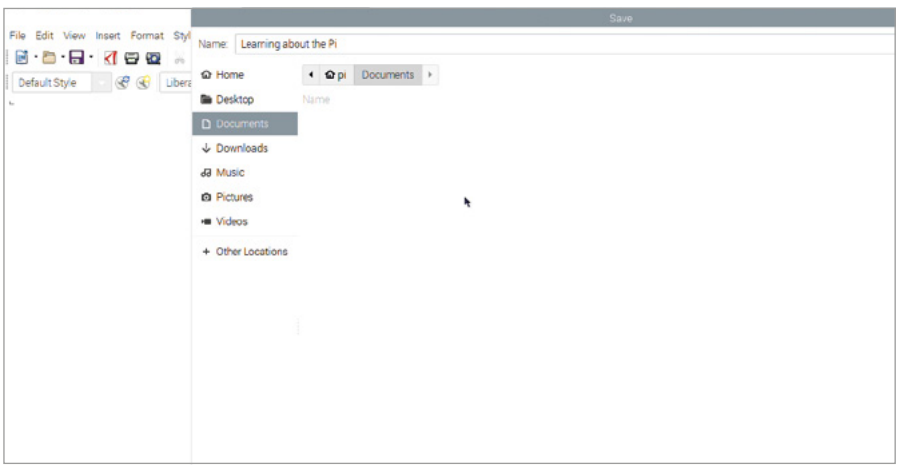


Figura 3-18: Guardar un documento

GUARDA TU TRABAJO

Acostúmbrate a guardar tu trabajo, aunque aún no lo hayas terminado. Te evitará muchos problemas en caso de haber un corte de corriente cuando vas por la mitad.

LibreOffice Writer no es más que uno de los componentes de la suite de productividad LibreOffice. Los otros, disponibles en la misma categoría de menús Oficina que LibreOffice Writer, son:

- **LibreOffice Base:** Una base de datos, para almacenar información, consultarla rápidamente y analizarla.
- **LibreOffice Calc**: Una hoja de cálculo, para gestionar números y crear gráficos.
- **LibreOffice Draw**: Programa de ilustración, una herramienta para imágenes y diagramas.
- **LibreOffice Impress**: Un programa de presentación, para crear diapositivas y hacer presentaciones.
- **LibreOffice Math**: Un editor de fórmulas, para crear fórmulas matemáticas con un formato adecuado que puede utilizarse en otros documentos.

LibreOffice también está disponible para otros ordenadores y sistemas operativos. Si te gusta la experiencia en tu Raspberry Pi, puedes descargarlo gratis desde **libreoffice.org** e instalarlo en cualquier ordenador con Microsoft Windows, Apple macOS o Linux.

Si quieres saber más sobre el uso de LibreOffice, haz clic en el menú de ayuda. Si no, cierra LibreOffice Writer haciendo clic en el botón de la esquina superior derecha de la ventana.

AYUDA

La mayoría de los programas incluyen un menú de ayuda que tiene de todo, desde información sobre qué es el programa hasta guías sobre cómo usarlo. Si algún programa te está resultando muy complicado, busca el menú de ayuda para orientarte.

La herramienta Recommended Software

Aunque el sistema Raspberry Pi se suministra con todo tipo de programas de software ya cargado, es compatible con muchos otros. En la herramienta Recommended Software encontrarás una selección de los mejores de esos programas.

Ten en cuenta que la herramienta Recommended Software necesita una conexión a Internet. Si trabajas con Raspberry Pi conectado, haz clic en el menú de Raspberry, mueve el puntero del ratón a Preferencias y haz clic en Recommended Software. La herramienta se cargará y luego empezará a descargar información sobre el software disponible.

Al cabo de unos segundos aparecerá una lista de paquetes de software compatibles (**Figura 3-19**). Al igual que el software en el menú de Raspberry, se organizan por categorías. Haz clic en una categoría en el panel de la izquierda para ver software de esa categoría. Haz clic en All Programs para ver todo el software.

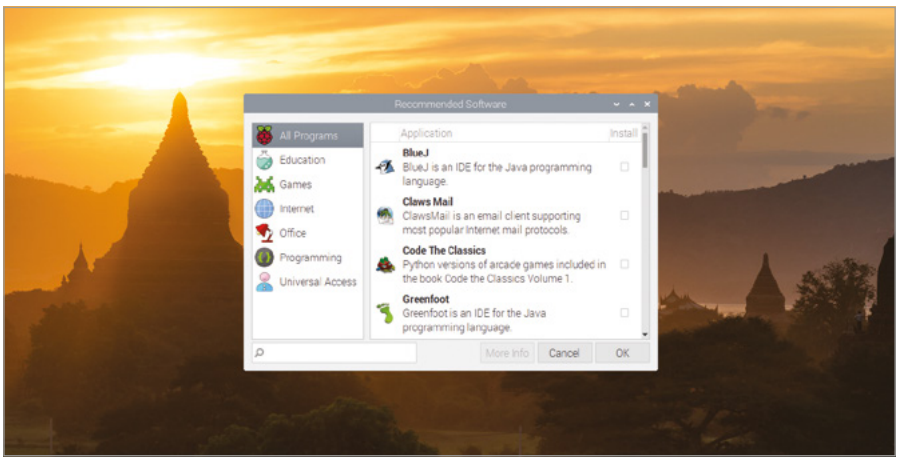


Figura 3-19: La herramienta Recommended Software

Si un programa de software tiene una marca de verificación, indica que ya está instalado en tu Raspberry Pi. Si no, puedes hacer clic en la casilla correspondiente para poner la marca e indicar que debe instalarse. Puedes marcar tantos programas como quieras antes de instalarlos todos en una sola operación. Pero si utilizas una tarjeta microSD más pequeña de lo recomendado, es posible que no tengas espacio para todos.

También puedes desinstalar software mediante el mismo procedimiento: busca un programa que ya tenga una marca en la casilla de verificación y luego haz clic en la marca para eliminarla. Si te equivocas o cambias de opinión, solo tienes que hacer clic de nuevo para volver a poner la marca.

Cuando tengas la selección de software deseada, haz clic en el botón OK para iniciar el proceso de instalación o desinstalación (**Figura 3-20** a continuación). Después de descargar e instalar cualquier software nuevo que hayas elegido, aparecerá un cuadro de diálogo. Haz clic en OK para cerrar la herramienta Recommended Software.

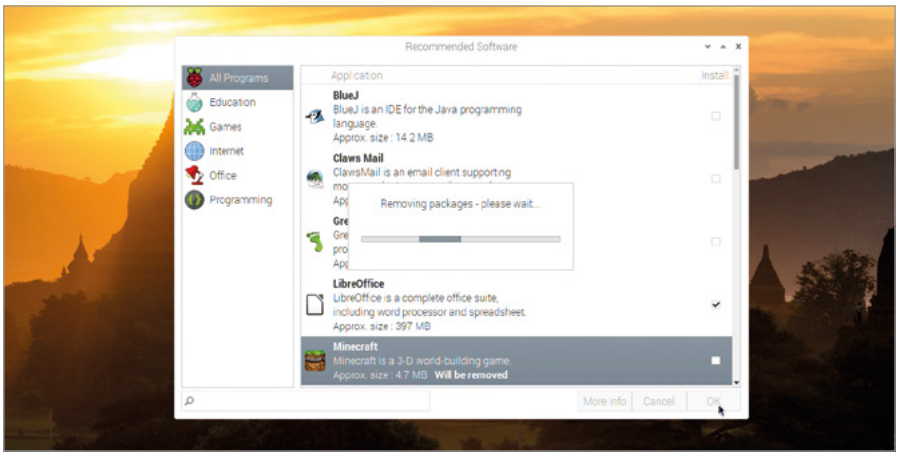


Figura 3-20: Desinstalar software

También en la categoría de preferencias del menú de Raspberry Pi encontrarás la herramienta Add/Remove Software para instalar o desinstalar programas. Te ofrece una amplia selección de programas de software, pero en este caso no han sido aprobados por la Fundación Raspberry Pi.

Herramienta Configuración de Raspberry Pi

El último programa que trataremos en este capítulo es la herramienta Configuración de Raspberry Pi y se parece mucho al asistente de bienvenida que hemos visto al principio: te permite cambiar opciones del sistema operativo Raspberry Pi. Haz clic en el icono de Raspberry, mueve el puntero del ratón para seleccionar la categoría Preferencias y haz clic en Configuración de Raspberry Pi para cargarla (**Figura 3-21**).

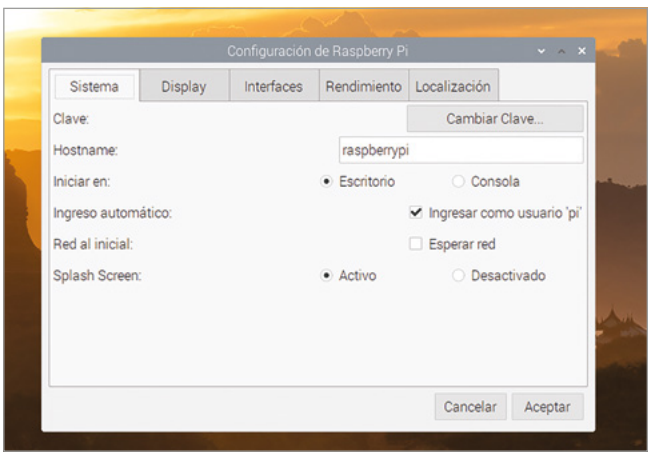


Figura 3-21: Herramienta Configuración de Raspberry Pi

La herramienta se compone de cinco pestañas. La primera pestaña es Sistema, que permite cambiar la contraseña de tu cuenta, definir un nombre de host (el que Raspberry Pi usa en tu red local con cable o inalámbrica) y modificar otras opciones, aunque es probable que la mayoría de ellas no necesiten cambios. Haz clic en la pestaña Display para pasar a la siguiente categoría. Aquí puedes modificar las opciones de pantalla si es preciso para adaptarlas a tu TV o monitor.

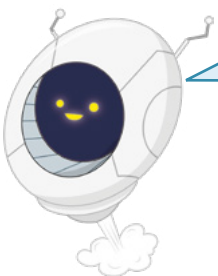
MÁS DETALLES

Esta breve información general es una mera introducción a la herramienta. Encontrarás más detallas sobre cada opción en el **Apéndice E, Herramienta Configuración de Raspberry Pi**.

La pestaña Interfaces contiene varias opciones, todas ellas desactivadas inicialmente. Solo deberías modificarlas si vas a añadir hardware nuevo (por ejemplo, el módulo de cámara de Raspberry Pi) y únicamente en caso de que así lo indique el fabricante del hardware. Las excepciones a esta regla son: SSH, que habilita "Secure Shell" y te permite iniciar sesión en Raspberry Pi desde otro ordenador de tu red mediante un cliente SSH. VNC, que habilita un "ordenador de red virtual" y te permite ver y controlar el escritorio del sistema Raspberry Pi desde otro ordenador de tu red usando un cliente VNC. Y Remote GPIO, que te permite usar pines GPIO de Raspberry Pi (algo que se detallará más adelante, en el **Capítulo 6, Informática física con Scratch y Python**) desde otro ordenador de tu red.

Haz clic en la pestaña Rendimiento para ver la cuarta categoría. Aquí puedes definir la cantidad de memoria usada por la unidad de procesamiento gráfico de Raspberry Pi (GPU) y, para ciertos modelos, incrementar el rendimiento de Raspberry Pi mediante un proceso denominado *overclock*. No obstante, también en este caso es preferible mantener las opciones como están, a menos que sea imprescindible cambiarlas.

Por último, haz clic en la pestaña Localización para ver esta categoría. Aquí puedes cambiar tu configuración regional, que controla cosas como el idioma usado en el sistema operativo Raspberry Pi y cómo se muestran los números, y donde puedes cambiar la zona horaria y la distribución del teclado, y establecer tu país para fines de WiFi. De momento, haz clic en Cancelar para cerrar la herramienta sin realizar ningún cambio.



¡ADVERTENCIA!

Las reglas sobre las frecuencias que puede usar una radio WiFi varían de un país a otro. Si el país WiFi configurado en Raspberry Pi Configuration es distinto del país en que te encuentras, seguramente te costará conectarte a tus redes e incluso podría ser ilegal bajo la legislación de licencias de radio, por lo tanto ¡no lo hagas!

Apagado

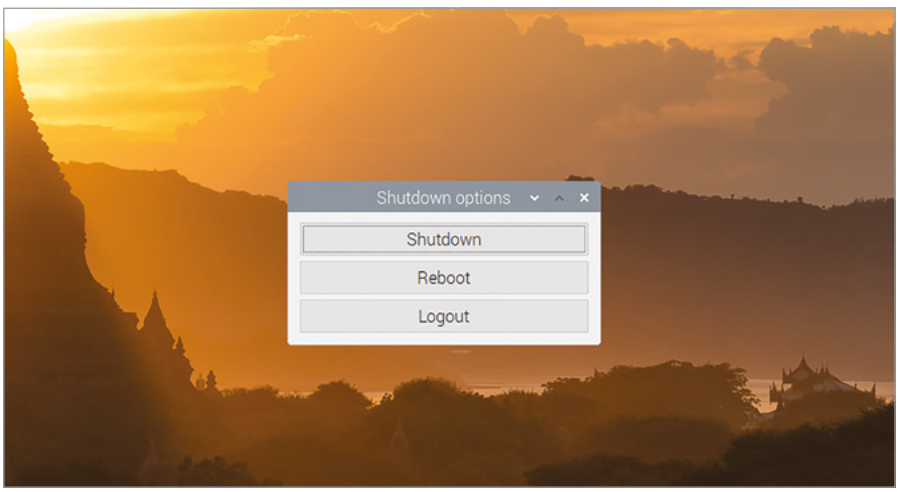
Ahora que ya has explorado el escritorio del sistema Raspberry Pi, es el momento de aprender algo muy importante: cómo apagar correctamente tu Raspberry Pi. Al igual que cualquier ordenador, Raspberry Pi mantiene los archivos en los que estás trabajando en una *memoria volátil*, que se vacía al apagar el sistema. Para los documentos que estás creando, es suficiente con guardar cada uno por separado: así cada archivo se lleva de la memoria volátil a la *memoria no volátil* (la tarjeta microSD), para que no pierdas nada de tu trabajo.

Sin embargo, los documentos en los que estás trabajando no son los únicos archivos abiertos. Durante su ejecución, el propio sistema operativo Raspberry Pi mantiene abiertos una serie de archivos y, si apagas tu Raspberry Pi con estos archivos aún abiertos, el sistema podría dañarse y requerir una reinstalación.

Para evitar que eso suceda, debes asegurarte de indicar al sistema de Raspberry Pi que guarde todos sus archivos y se prepare para la desconexión o *apagado* del sistema operativo.

Haz clic en el icono de Raspberry, en la esquina superior izquierda del escritorio, y luego en Shutdown. Se abrirá una ventana con estas tres opciones (**Figura 3-22**): Shutdown, Reboot y Logout. Shutdown será la que más utilices: al hacer clic en ella, se indica al sistema Raspberry Pi OS que cierre todos los programas de software y los archivos abiertos y luego apague Raspberry Pi. Cuando la pantalla se vuelva negra, espera unos segundos hasta que se apague la luz destellante verde en Raspberry Pi, momento en que será seguro apagar la fuente de alimentación.

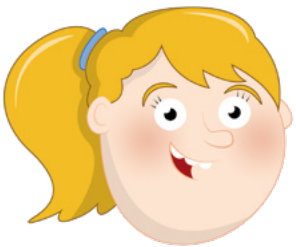
Para volver a encender Raspberry Pi, simplemente desconecta y reconecta el cable de alimentación; o apaga y enciende el interruptor en la toma de pared.



▲ **Figura 3-22:** Apagar Raspberry Pi

El reinicio realiza un proceso de apagado similar, cerrándolo todo, pero en lugar de apagar Raspberry Pi lo reinicia, casi exactamente igual que si lo hubieras apagado y luego desconectado y reconectado el cable. Tendrás que usar la opción Reboot si realizas ciertos cambios que requieren un reinicio del sistema operativo (entre ellos la instalación de algunas actualizaciones en el software principal) o si hay software que ha tenido algún fallo (se ha *bloqueado)* y por ese motivo el sistema de Raspberry Pi está temporalmente inutilizable.

La última opción, Logout, solo es útil si tienes más de una cuenta de usuario en tu Raspberry Pi: para cerrar cualquier programa que tengas abierto e ir a la página de inicio de sesión, en la que se te pedirá que introduzcas un nombre de usuario y una contraseña. Si seleccionas Logout involuntariamente y quieres retroceder, simplemente escribe "pi" como nombre de usuario y la contraseña que creaste en el asistente de bienvenida al principio de este capítulo.



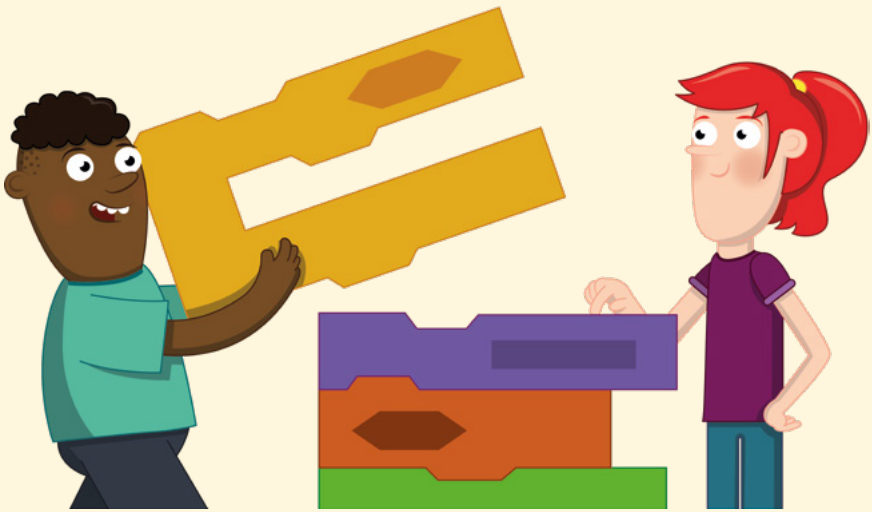
¡ADVERTENCIA!

No tires del cable de Raspberry Pi sin antes apagar el ordenador. Si lo haces podría corromperse el sistema operativo y también podrías perder archivo creados o descargados.

Capítulo 4

Programar con Scratch 3

Aprende a empezar a codificar usando Scratch, el lenguaje de programación basado en bloques

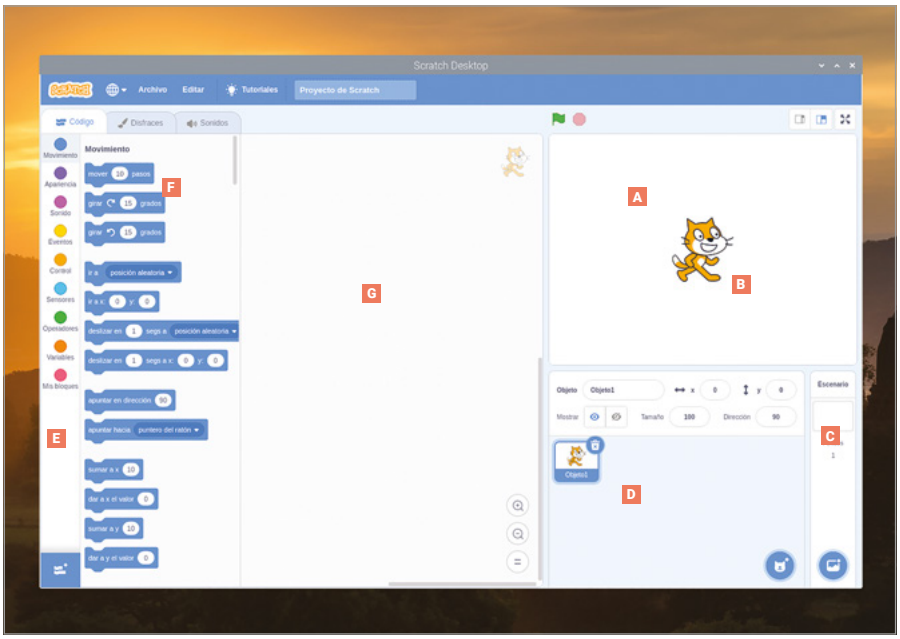


Con Raspberry Pi no solo puedes usar software creado por otros: también puedes crear tu propio software, basándote en tu imaginación. Tanto si tienes experiencia previa creando programas (proceso conocido como programación o codificación) como si no, Raspberry Pi te resultará una gran plataforma de creación y experimentación.

La clave para la codificación en Raspberry Pi es Scratch, un lenguaje de programación visual desarrollado por el Instituto Tecnológico de Massachusetts (MIT). Mientras que con los lenguajes de programación tradicionales tienes que escribir instrucciones de texto para que el ordenador las lleve a cabo (algo así como escribir una receta para hacer una tarta), con Scratch construyes tu programa paso a paso utilizando bloques, que son porciones de código predefinido ocultas tras piezas de rompecabezas codificadas por colores.

Scratch es estupendo como primer lenguaje para codificadores principiantes de cualquier edad. Y, a pesar de su aspecto sencillo, es un entorno de programación potente y totalmente funcional para crear todo tipo de cosas, desde simples juegos y animaciones hasta complejos proyectos de robótica interactiva.

La interfaz de Scratch 3



A **Área de escenario** – Como los actores de una obra de teatro, tus objetos se mueven por el escenario controlados por tu programa.

B **Objeto** – Los personajes u objetos que controlas en un programa de Scratch se denominan objetos y están en el escenario.

C **Controles de escenario** – Sirven para cambiar el escenario, por ejemplo añadiendo tus propias imágenes como fondo.

D **Lista de objetos** – Todos los objetos que hayas creado o cargado en Scratch aparecerán en esta sección de la ventana.

E **Paleta de bloques** – Todos los bloques disponibles para tu programa aparecen en la paleta de bloques, que contiene categorías codificadas por colores.

F **Bloques** – Porciones predefinidas de código de programa que te permiten crear tu programa paso a paso.

G **Área de código** – El área de código es donde se crea tu programa arrastrando y colocando bloques de la paleta de bloques para formar scripts.

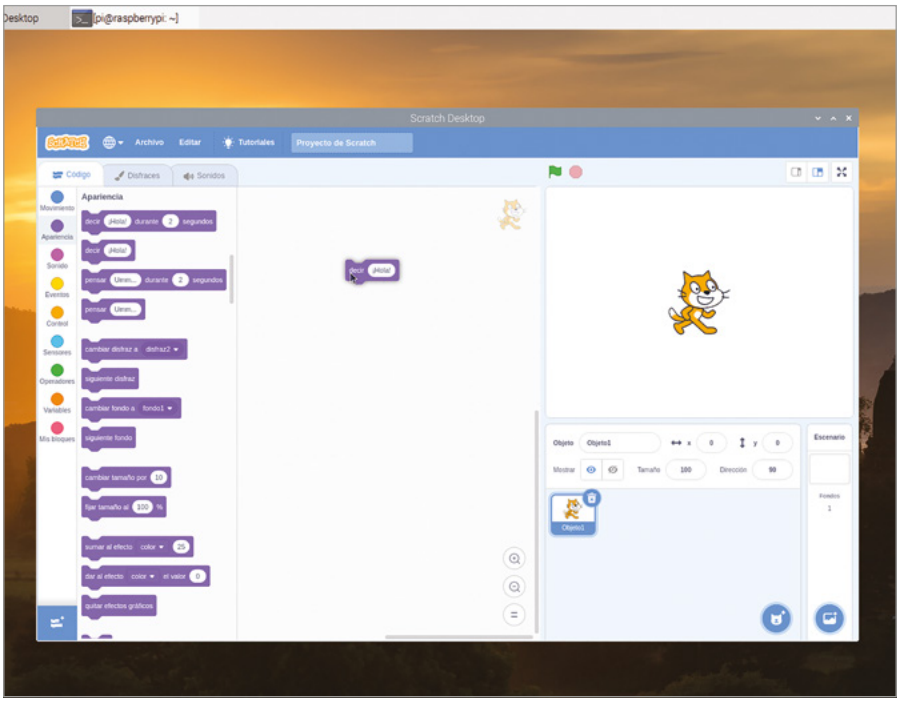
VERSIONES DE SCRATCH

Al redactarse esta guía, el sistema operativo Raspberry Pi se suministraba con tres versiones de Scratch: Scratch 1, 2 y 3, todas ellas incluidas en la sección Programación, en el menú del sistema Raspberry Pi. Este capítulo hace referencia a Scratch 3. Ten en cuenta que Scratch 3 solo funcionará en Raspberry Pi 4. Si prefieres usar Scratch 2, esa versión no funcionará con Raspberry Pi Zero, modelos A, A+, B ni B+.

Tu primer programa con Scratch: ¡Hola mundo!

Scratch 3 se carga en Raspberry Pi igual que cualquier otro programa: haz clic en el icono de Raspberry para cargar el menú del sistema operativo Raspberry Pi, lleva el cursor a la sección Programación y haz clic en Scratch 3. La interfaz de usuario de Scratch 3 tardará solo unos segundos en cargarse.

La mayoría de los lenguajes de programación utilizan instrucciones escritas para decirle al ordenador qué debe hacer, pero Scratch es diferente. Para empezar, haz clic en la categoría Apariencia de la paleta de bloques, en la parte izquierda de la ventana de Scratch. Así se abrirán los bloques de esa categoría, de color morado. Localiza el bloque decir ¡Hola! durante 2 segundos, haz clic en él con el botón izquierdo del ratón y mantenlo pulsado para arrastrar el cursor al área de código, situada en el centro de la ventana de Scratch; luego suelta el botón (**Figura 4-1**).



▲ **Figura 4-1:** Arrastrar y colocar el bloque en el área de código

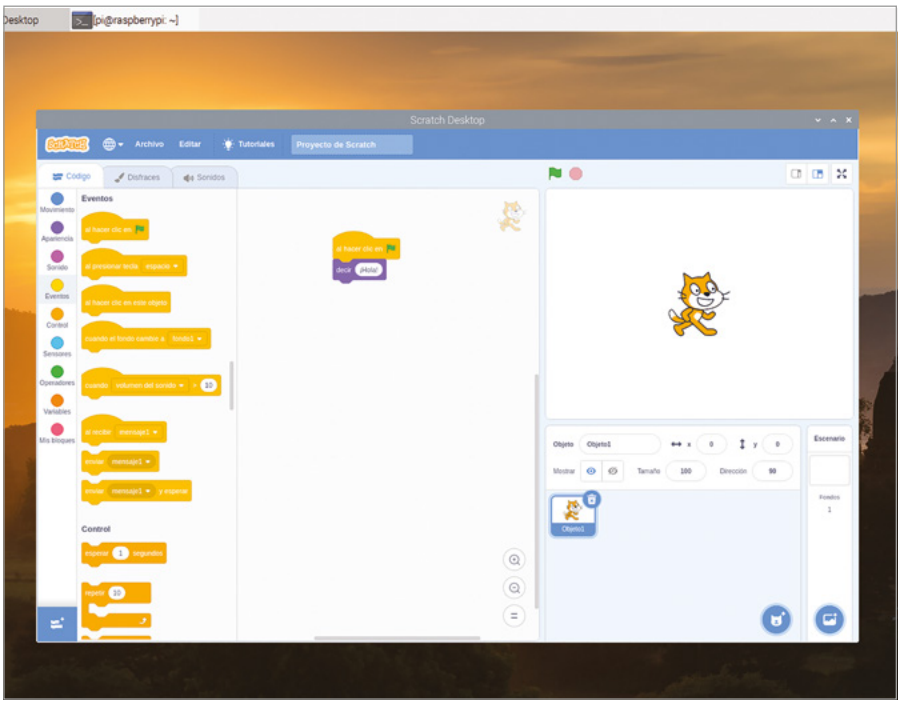
Fíjate en la forma del bloque que acabas de colocar: tiene una hendidura arriba y un saliente en la parte inferior correspondiente. Es como una pieza de un puzle: te indica que el bloque debería tener algo encima y debajo. En este programa, lo que irá encima es un *activador*..

En la paleta de bloques, haz clic en la categoría Eventos (en dorado) y luego haz clic y arrastra al área de código el bloque al hacer clic en ; a este tipo de bloque se le denomina *sombrero*. Colócalo de forma que la parte saliente inferior encaje en la hendidura superior

del bloque 

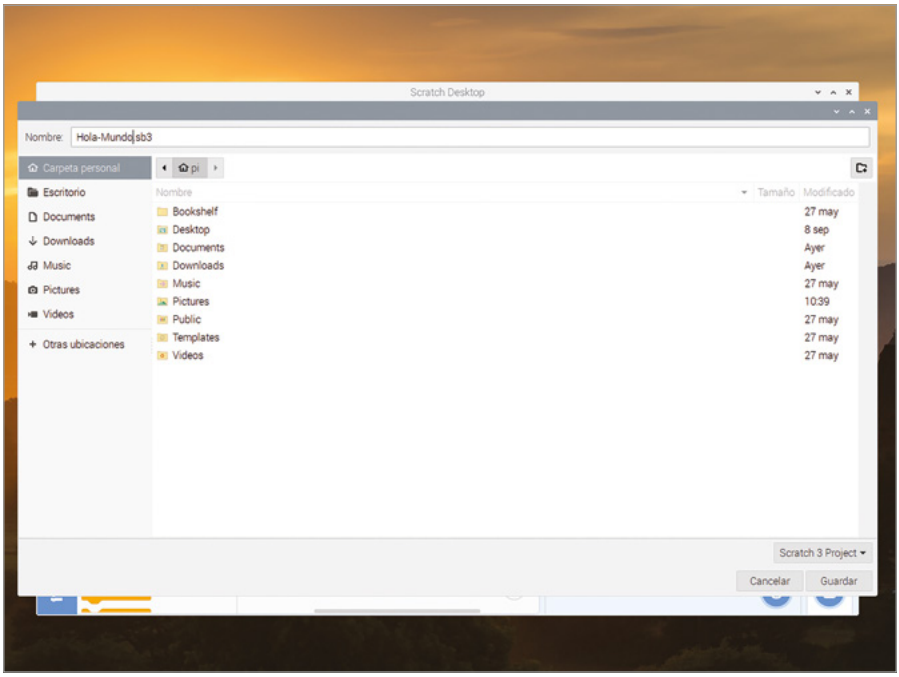
y suelta el botón cuando veas un contorno blanco. No hace falta una gran precisión: en cuanto el bloque esté lo suficientemente cerca, encajará como una pieza de puzle. Si no es así, vuelve a hacer clic en él y mantén la pulsación para ajustarlo.

Ya tienes tu programa completo. Para hacerlo funcionar, o sea para *ejecutarlo*, haz clic en el icono de la bandera verde situada en la esquina superior izquierda del área de escenario. Si todo ha salido bien, el objeto gato te saludará desde el escenario con un alegre "¡Hola!" (**Figura 4-2**), señal de que tu primer programa funciona a la perfección.

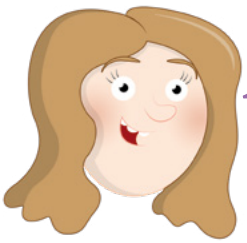


▲ **Figura 4-2:** Haz clic en la bandera verde sobre el escenario y el gato dirá "Hola"

Antes de continuar, asigna un nombre a tu programa y guárdalo. Haz clic en el menú Archivo y luego en "Guardar en tu ordenador". Escribe un nombre y haz clic en el botón Guardar (**Figura 4-3**).



▲ **Figura 4-3:** Guarda tu programa con un nombre fácil de recordar



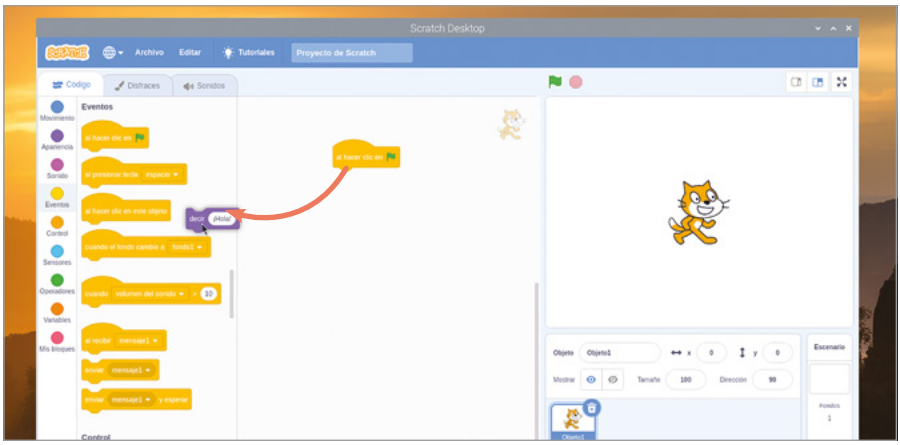
SUGERENCIAS

Algunos bloques de Scratch se pueden modificar. Por ejemplo, haz clic en la palabra "¡Hola!" y escribe otra cosa. Luego vuelve a hacer clic en la bandera verde. Fíjate en lo que ocurre en el escenario.

Pasos siguientes: secuenciación

Tu programa tiene dos bloques pero solo una instrucción real: decir "Hola" cada vez que se hace clic en la bandera y el programa se ejecuta. Para que haga más cosas, debes aprender a *secuenciar*. Los programas informáticos, incluso los más simples, se componen de una lista de instrucciones, igual que en una receta de cocina. Cada instrucción sigue a la instrucción anterior, en una progresión lógica denominada *secuencia lineal*.

Para empezar, haz clic en el bloque decir ¡Hola! y arrástralo del área de código a la paleta de bloques (**Figura 4-4**). Así se quita ese bloque del programa y solo queda el bloque activador, al hacer clic en .



▲ **Figura 4-4:** Para eliminar un bloque, simplemente arrástralo fuera del área de código

Haz clic en la categoría Movimiento en la paleta de bloques, y luego haz clic en el bloque mover 10 pasos, arrástralo y suéltalo bajo el bloque activador en el área de código. Como su nombre indica, esto le dice a tu objeto (el gato) que se mueva un número de pasos en la dirección hacia la que mira actualmente.

Añade más instrucciones a tu programa para crear una secuencia. Haz clic en la paleta Sonido, de color rosa, y luego haz clic en el bloque tocar sonido Miau hasta que termine, arrástralo y colócalo bajo el bloque mover 10 pasos . Continúa: vuelve a hacer clic en la categoría Movimiento y arrastra otro bloque mover 10 pasos bajo el bloque Sonido, pero esta vez haz clic en "10" para seleccionarlo y escribe "-10" para crear un bloque mover -10 pasos .

Haz clic en la bandera verde sobre el escenario para ejecutar el programa. Verás que el gato se mueve hacia la derecha y emite un maullido (para oírlo necesitarás usar auriculares o tener altavoces conectados) y luego regresa a la posición inicial. Vuelve a hacer clic en la bandera y el gato repetirá las acciones.

¡Enhorabuena! Has creado una secuencia de instrucciones que Scratch ejecuta de una en una, de arriba abajo. Aunque Scratch ejecuta las instrucciones de la secuencia de una en una, lo hace rapidísimo: elimina el bloque **tocar sonido Miau hasta que termine** haciendo clic en el bloque inferior **mover -10 pasos** y arrastrándolo para separarlo, arrastrando el bloque **tocar sonido Miau hasta que termine** a la paleta de bloques y sustituyéndolo por el bloque más simple **tocar sonido Miau** antes de arrastrar **mover -10 pasos** a la parte inferior de tu programa.

Haz clic en la bandera verde para volver a ejecutar el programa. Parece que el gato no se mueve, pero la verdad es que retrocede tan rápidamente que parece estar inmóvil. Esto es porque al usar el bloque **tocar sonido Miau** no espera a que el sonido termine de reproducirse antes de proceder con el paso siguiente; porque Raspberry Pi "piensa" con tanta rapidez que la siguiente instrucción se ejecuta antes de que veas al gato moverse. Hay otra forma de arreglarlo, además de usar el bloque **tocar sonido Miau hasta que termine**: haz clic en la categoría Control (de color naranja claro) en la paleta de bloques y luego haz clic en el bloque **esperar 1 segundos** y arrástralo para colocarlo entre el bloque **tocar sonido Miau** y el bloque inferior **mover -10 pasos**.

Haz clic en la bandera verde para ejecutar el programa por última vez y verás que, después de moverse a la derecha, el gato espera un segundo antes de volver a la izquierda. Esto se conoce como *retardo* y es esencial para controlar cuánto tarda en ejecutarse tu secuencia de instrucciones.

RETO: AÑADIR MÁS PASOS

Añade más pasos a tu secuencia y cambia los valores de los pasos que ya tienes. ¿Qué ocurre cuando el número de pasos en un bloque de movimiento no coincide con el número de pasos en otro? ¿Qué ocurre si intentas reproducir un sonido mientras hay otro aún sonando?

Bucles

La secuencia que has creado hasta ahora solo se ejecuta una vez: al hacer clic en la bandera verde, el gato se mueve y maúlla, y luego el programa se detiene hasta que vuelves a hacer clic en la bandera verde. Pero no tiene por qué detenerse, porque Scratch incluye un tipo de bloque Control denominado *bucle*.

Haz clic en la categoría Control en paleta de bloques y localiza el bloque por siempre. Haz clic en él, arrástralo al área de código y suéltalo bajo el bloque al hacer clic en y sobre el primer bloque mover 10 pasos.

Fíjate que el bloque por siempre en forma de C aumenta de tamaño automáticamente para abarcar los otros bloques de la secuencia. Haz clic en la bandera verde para ver el efecto del bloque por siempre: ahora, en lugar de ejecutarse una vez y detenerse, tu programa se ejecuta una y otra vez, indefinidamente. En programación esto se conoce como *bucle infinito*, o sea, un bucle sin fin.

Si te cansa el maullido constante, haz clic en el octógono rojo junto a la bandera verde, sobre el área del escenario, para detener tu programa. Para cambiar el tipo de bucle, haz clic en el primer bloque mover 10 pasos, arrástralo y sácalo junto con los bloques de debajo

del bloque por siempre. Luego colócalos bajo el bloque al hacer clic en . Haz clic en el bloque por siempre y arrástralo a la paleta de bloques para eliminarlo, luego haz clic en el bloque repetir 10 bajo el bloque al hacer clic en para que rodee a los demás bloques.

Haz clic en la bandera verde para ejecutar tu nuevo programa. Al principio, parece que hace lo mismo que tu versión original: repetir la secuencia de instrucciones una y otra vez. Pero esta vez, en lugar de continuar indefinidamente, el bucle terminará después de diez repeticiones. Es lo que se conoce como *bucle definido*: tú defines cuándo debe finalizar. Los bucles son herramientas eficaces y la mayoría de los programas (especialmente los juegos y los programas de detección) utilizan abundantemente bucles infinitos y bucles definidos.

¿QUÉ OCURRE AHORA?

¿Qué ocurre si cambias el número en el bloque de bucle a uno mayor? ¿Y si lo cambias a uno menor? ¿Qué ocurre si escribes 0 en el bloque de bucle?

Variables y condicionales

Los últimos conceptos que es necesario entender antes de empezar a codificar programas reales con Scratch están muy relacionados entre sí: son las *variables* y las *condicionales*. Una variable, como su nombre indica, es un valor que puede variar (es decir, cambiar) con el tiempo y bajo el control del programa. Una variable tiene dos propiedades principales: su nombre y el valor que almacena. Ese valor no tiene por qué ser un número: puede ser texto, verdadero-falso o estar vacío, lo que se denomina *valor nulo*.

Las variables son herramientas potentes. Piensa en las cosas que necesitan seguimiento en un juego: la salud de un personaje, la velocidad del objeto en movimiento, el nivel del juego y la puntuación. Todas esas cosas se siguen como variables.

En primer lugar, haz clic en el menú Archivo y guarda el programa existente haciendo clic en "Guardar en tu ordenador". Si ya has guardado el programa anteriormente, se te preguntará si quieres sobrescribirlo, sustituyendo la copia guardada por tu nueva versión actualizada. A continuación, haz clic en Archivo y luego en Nuevo para iniciar un nuevo proyecto en blanco (haz clic en Aceptar cuando se te pregunte si quieres reemplazar el contenido del proyecto actual). Haz clic en la categoría Variables (en naranja oscuro) en la paleta de bloques y luego haz clic en el botón "Crear una variable". Escribe "bucles" como nombre de la variable (**Figura 4-5**) y haz clic en Aceptar para que aparezca una serie de bloques en la paleta de bloques.

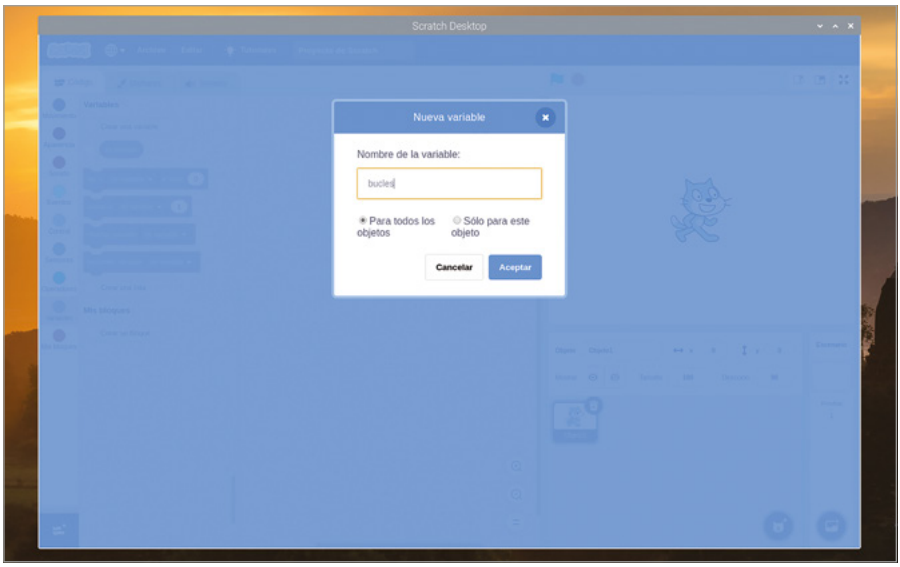


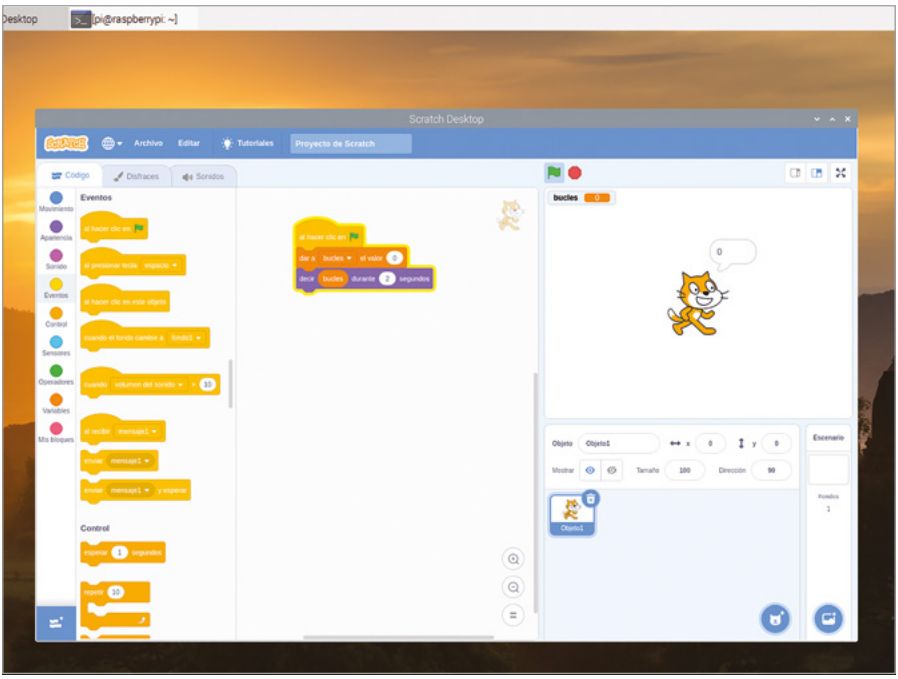
Figura 4-5: Asigna un nombre a tu nueva variable

Haz clic y arrastra el bloque dar a bucles el valor 0 al área de código. Esto indica a tu programa que *inicialice* la variable con un valor de 0. A continuación, haz clic en la categoría Apariencia de la paleta de bloques, arrastra el bloque decir ¡Hola! durante 2 segundos y colócalo debajo del bloque dar a bucles el valor 0.

Como ya hemos visto, los bloques decir ¡Hola! hacen que el gato diga lo que aparece escrito en ellos. En lugar de escribir el mensaje en el bloque, puedes usar una variable. Vuelve a hacer clic en la categoría Variables de la paleta de bloques, haz clic en el bloque redondeado bucles (denominado *bloque informador* y situado al principio de la lista, con una casilla de

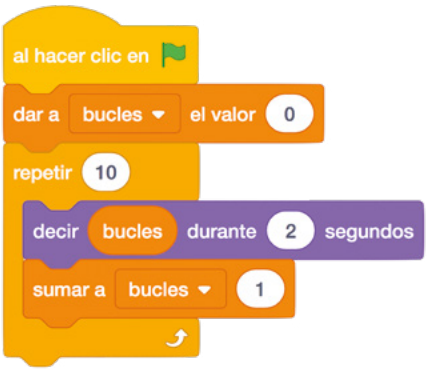
verificación junto a él) y arrástralo a una posición encima de la palabra "Hola" en tu bloque decir ¡Hola! durante 2 segundos. Así se crea un bloque combinado: decir bucles durante 2 segundos.

Haz clic en la categoría Eventos de la paleta de bloques, haz clic en el bloque al hacer clic en y arrástralo para colocarlo encima de tu secuencia de bloques. Haz clic en la bandera verde sobre el área de escenario y verás que el gato dice "0" (**Figura 4-6**), el valor que has asignado a la variable "bucles".



▲ **Figura 4-6:** Esta vez el gato dirá el valor de la variable

Pero las variables son modificables. Haz clic en la categoría Variables de la paleta de bloques, haz clic en el bloque sumar a bucles 1 y arrástralo a la parte inferior de tu secuencia. A continuación, haz clic en la categoría Control, luego en bloque repetir 10 y arrástralo para que empiece directamente debajo de tu bloque dar a bucles el valor 0 y rodee los demás bloques de tu secuencia.



Vuelve a hacer clic en la bandera verde. Esta vez, verás que el gato cuenta de 0 a 9. Esto funciona porque ahora tu programa está cambiando, o *modificando*, la propia variable: cada vez que el bucle se ejecuta, el programa añade una unidad al valor de la variable "bucles" (**Figura 4-7**).

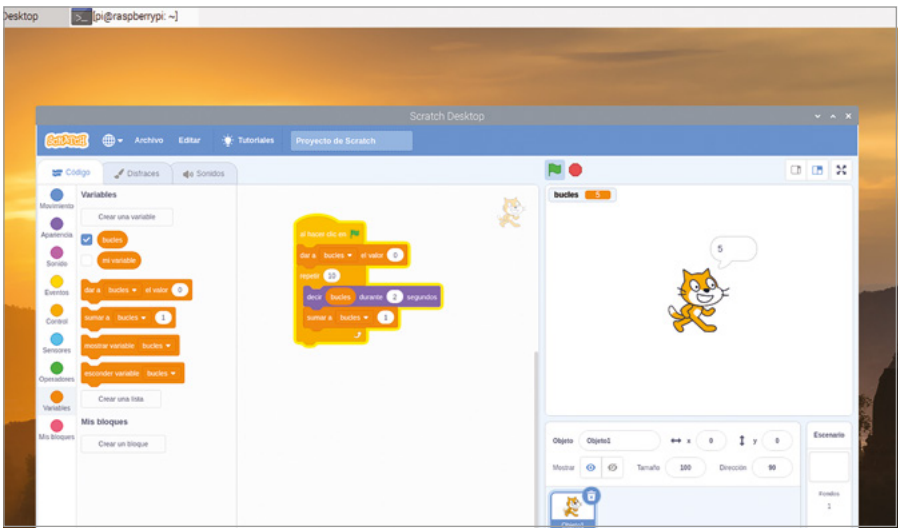
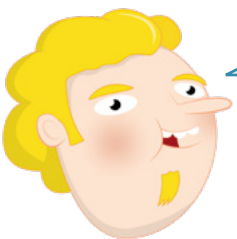


Figura 4-7: Gracias al bucle, ahora el gato cuenta hacia delante

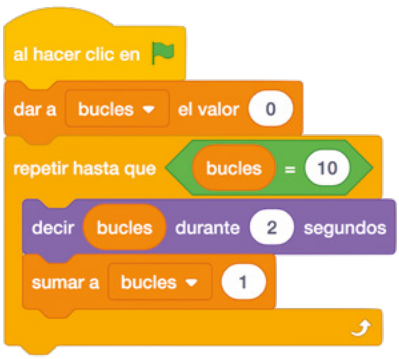


CONTAR DESDE CERO

Aunque el bucle que has creado se ejecuta diez veces, el objeto gato solo cuenta hasta nueve, porque para nuestra variable empezamos con el valor cero. Entre cero y nueve, ambos incluidos, tenemos diez números: por eso el programa se detiene antes de que el gato diga "10". Para cambiar esto, puedes fijar el valor inicial de la variable en 1 en lugar de 0.

Aparte de modificarla, puedes hacer más cosas con una variable. Haz clic y arrastra el bloque decir bucles durante 2 segundos para separarlo del bloque repetir 10 y colócalo debajo del bloque repetir 10. Haz clic y arrastra el bloque repetir 10 a la paleta de bloques para eliminarlo y sustitúyelo por un bloque repetir hasta que, y asegúrate de que el bloque esté conectado a la parte inferior del bloque decir bucles durante 2 segundos y rodee los otros dos bloques de tu secuencia. Haz clic en la categoría Operadores, identificada por el color verde en la paleta de bloques, haz clic en el bloque de forma hexagonal = y colócalo sobre el espacio también hexagonal del bloque repetir hasta que.

Este bloque de tipo Operadores te permite comparar dos valores, incluidos los de variables. Haz clic en la categoría Variables, arrastra el bloque informador bucles al espacio vacío del bloque de Operadores =, haz clic en el espacio que contiene "50" y escribe el número "10".



Haz clic en la bandera verde sobre el área de escenario y verás que el programa funciona igual que antes: el gato cuenta de 0 a 9 (**Figura 4-8**) y luego el programa se detiene.

Esto se debe a que el bloque repetir hasta que funciona exactamente igual que el bloque repetir 10, pero en lugar de contar el número de bucles lo que hace es comparar el valor de la variable "bucles" con el valor que has escrito a la derecha del bloque. Cuando la variable "bucles" llega a 10, el programa se detiene.

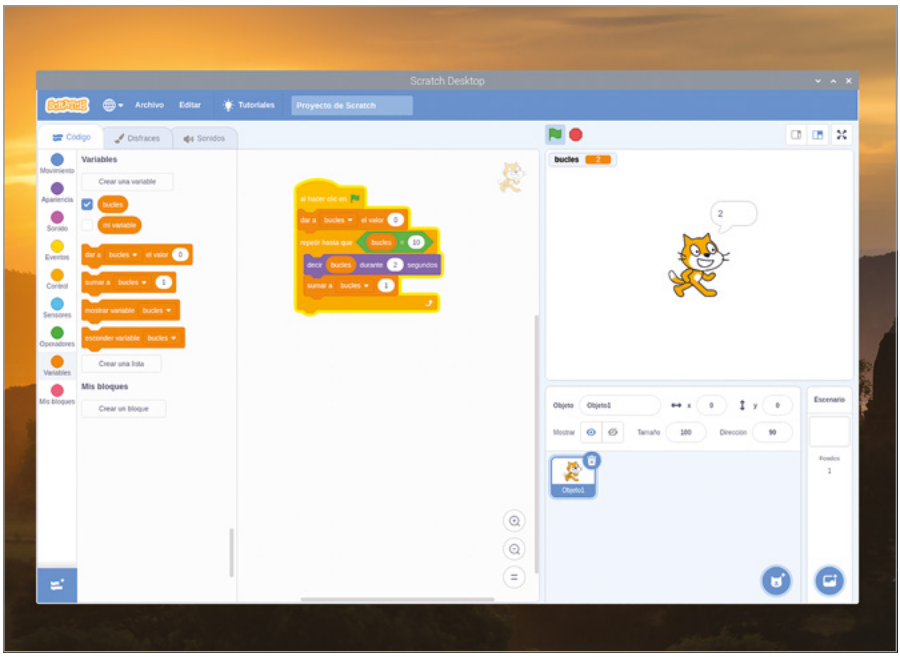


Figura 4-8: Usar un bloque "repetir hasta que" con un operador comparativo

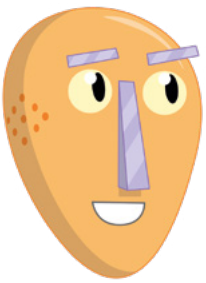
Esto se conoce como *operador comparativo*, porque compara dos valores. Haz clic en la categoría Operadores de la paleta de bloques, y localiza los otros dos bloques con forma hexagonal encima y debajo del que contiene el símbolo "=". Estos también son operadores comparativos: "<" compara dos valores y se activa cuando el valor de la izquierda es menor que el de la derecha, y ">" se activa cuando el valor de la izquierda es mayor que el de la derecha.

Haz clic en la categoría Control de la paleta de bloques, localiza el bloque si entonces, haz clic en él, arrástralo al área de código y colócalo justo debajo del bloque decir bucles durante 2 segundos. Rodeará automáticamente el bloque sumar a bucles 1, por lo que debes hacer clic en ese bloque y arrastrarlo para moverlo, para que se conecte a la parte inferior de tu bloque si entonces. Haz clic en la categoría Apariencia de la paleta de bloques, haz clic en el bloque decir ¡Hola! durante 2 segundos y arrástralo para colocarlo dentro del bloque si entonces. Haz clic en la categoría Operadores de la paleta de bloques, haz clic en el bloque ( ) > ( ) y arrástralo para colocarlo dentro del espacio hexagonal del bloque si entonces.

El bloque si entonces es un bloque condicional, lo que significa que el bloque dentro de él solo se ejecutará cuando se cumpla una condición determinada. Haz clic en la categoría Variables de la paleta de bloques, arrastra el bloque informador bucles hasta el espacio vacío en tu bloque > , haz clic en el espacio que contiene "50" y escribe el número "5". Por último, haz clic en la palabra "¡Hola!" de tu bloque decir ¡Hola! durante 2 segundos y escribe "¡Eso es alto!".

Haz clic en la bandera verde. Al principio, el programa funcionará como antes, con el gato contando de cero en adelante. Al llegar a 6, el primer número mayor que 5, el bloque entonces iniciará la activación y el gato comentará lo altos que empiezan a ser los números (**Figura 4-9**). Y ya está: has aprendido a trabajar con variables y condicionales.

Figura 4-9: El gato hace un comentario al llegar al número 6



RETO: ALTOS Y BAJOS

¿Cómo puedes cambiar el programa para que el objeto gato comente lo bajos que son los números inferiores a 5? ¿Puedes cambiarlo para que comente tanto los números altos como los bajos? Experimenta con el bloque si entonces si no para hacerlo más fácil.

Proyecto 1: Cronómetro de reacción para astronautas

Ahora que entiendes cómo funciona Scratch, es hora de hacer algo un poco más interactivo: un cronómetro de reacción, diseñado en honor de Tim Peake, el astronauta británico de la Agencia Espacial Europea, que fue tripulante de la Estación Espacial Internacional.

Guarda tu programa, si quieres conservarlo, y luego abre un nuevo proyecto haciendo clic en Archivo y Nuevo. Antes de empezar, asígnale un nombre haciendo clic en Archivo y "Guardar en tu ordenador": llámalo "Cronómetro de reacción para astronautas".

Este proyecto se basa en dos imágenes, un fondo y un objeto, que no se incluyen en los recursos de Scratch. Para descargarlas haz clic en el icono de Raspberry para cargar el menú del sistema operativo Raspberry Pi, lleva el cursor a la sección Internet y haz clic en Navegador web Chromium. Cuando se cargue el navegador, escribe **rpf.io/astronaut-backdrop** en la barra de direcciones y pulsa la tecla **ENTRAR**. Haz clic con el botón derecho del ratón en la imagen del espacio y luego en "Guardar imagen como..." y finalmente en el botón Guardar (**Figura 4-10**). Vuelve a hacer clic en la barra de direcciones y escribe **rpf.io/astronaut-sprite**. Luego pulsa la tecla **ENTRAR**.

Figura 4-10: Guardar la imagen de fondo

Haz clic con el botón derecho del ratón en la imagen de Tim Peake, luego haz clic en "Guardar imagen como..." y finalmente en el botón Guardar. Con esas dos imágenes guardadas, ya puedes cerrar Chromium o dejarlo abierto y usar la barra de tareas para volver a Scratch 3.

INTERFAZ DE USUARIO

Si has seguido este capítulo desde el principio, ya te deberías haber familiarizado con la interfaz de usuario de Scratch 3. Las siguientes instrucciones de proyecto requieren que sepas dónde están las cosas; si se te olvida dónde encontrar algo, usa como recordatorio la imagen de la interfaz de usuario al principio de este capítulo.

Haz clic con el botón derecho en el objeto gato en la lista y haz clic en "borrar". Mueve el puntero del ratón sobre el icono Elige un fondo y haz clic en el icono Cargar un fondo en la lista emergente. Localiza el archivo **Space-background.png** en la carpeta de descargas, haz clic en él para seleccionarlo y luego haz clic en Aceptar. El fondo blanco del escenario se sustituirá por la imagen del espacio, y el área del código será reemplazada por el área de fondos (**Figura 4-11**). Aunque puedes dibujar ahí, de momento limítate a hacer clic en la pestaña Código, en la parte superior izquierda de la ventana de Scratch 3.

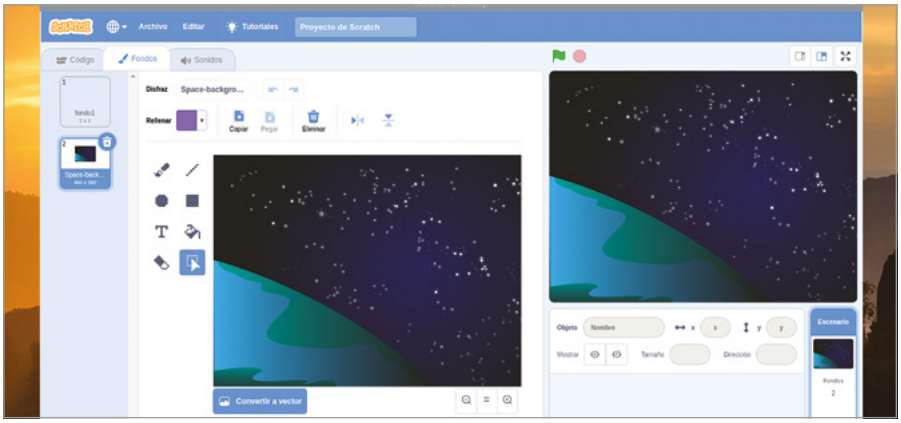


Figura 4-11: El fondo del espacio aparece en el escenario

Carga el nuevo objeto moviendo el puntero del ratón sobre el icono Elige un objeto y haciendo clic en el icono Subir objeto en la parte superior de la lista emergente. Localiza el archivo **Astronaut-Tim.png** en la carpeta de descargas, haz clic en él para seleccionarlo y luego haz clic en Aceptar. El objeto aparece en el escenario automáticamente, pero puede que no esté centrado: haz clic en él, arrástralo con el ratón y colócalo en un punto de la parte central inferior (**Figura 4-12**).

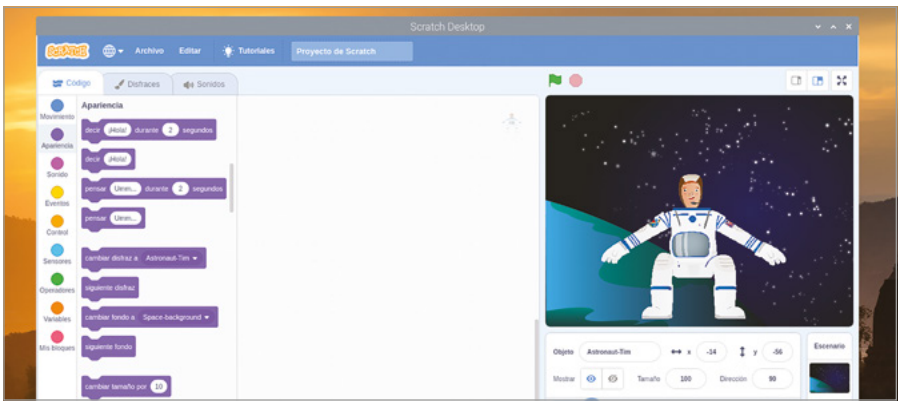
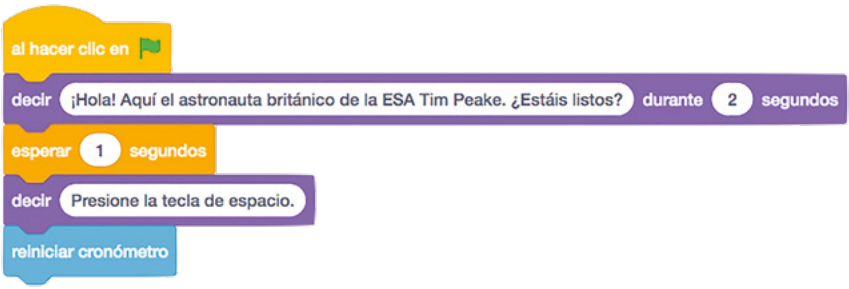


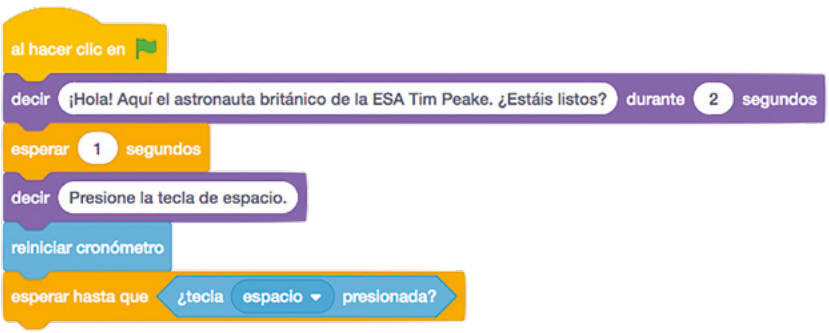
Figura 4-12: Arrastrar el objeto astronauta a la parte inferior del escenario

Con el fondo y el objeto ya en su sitio, ya puedes empezar a crear tu programa. Comienza creando una nueva variable llamada "tiempo", asegurándote de seleccionar la opción "Para todos los objetos" antes de hacer clic en Aceptar. Haz clic en tu objeto (en el escenario o en el panel de objetos) para seleccionarlo y añade al área de código un bloque al hacer clic en de la categoría Eventos. A continuación, añade un bloque decir ¡Hola! durante 2 segundos de la categoría Apariencia y haz clic en él para cambiarlo a "decir ¡Hola! Aquí el astronauta británico de la ESA Tim Peake. ¿Estáis listos?"

Añade un bloque esperar 1 segundos de la categoría Control y luego un bloque decir ¡Hola!. Cambia este bloque a "Presiona la tecla espacio" y añade un bloque reiniciar cronómetro de la categoría Sensores. Esto controla una variable especial integrada en Scratch para cronometrar acciones, y se usará para medir la rapidez con la que reaccionas en el juego.

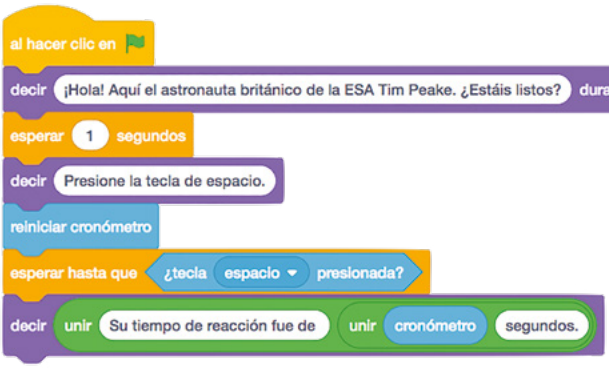


Añade un bloque de la categoría Control esperar hasta que, luego arrastra un bloque de Sensores ¿tecla espacio presionada? a este espacio en blanco. Esto pondrá en pausa el programa hasta que pulses la tecla **ESPACIO** en el teclado, pero el cronómetro seguirá en marcha, para medir exactamente el tiempo desde que se emite el mensaje "Presiona la tecla espacio" hasta que realmente pulsas la tecla **ESPACIO**.



Ahora necesitas que Tim te diga cuánto tiempo has tardado en pulsar la tecla **ESPACIO**, pero de una manera que sea fácil de leer. Para eso necesitas un bloque unir de la categoría Operadores. Con él se usan dos valores, incluidas variables, y se unen uno tras otro, es lo que se denomina *concatenación*.

Empieza con un bloque **decir ¡Hola!** y luego arrastra y coloca un bloque **unir** sobre la palabra "¡Hola!". Haz clic en "manzana" y escribe "Tu tiempo de reacción fue ", asegurándote de añadir un espacio en blanco al final, luego arrastra otro bloque unir sobre la parte superior de "plátano" en el segundo cuadro. Arrastra un bloque informador **cronómetro** de la categoría Sensores al cuadro que ahora es el intermedio y escribe " segundos." en el último cuadro, asegurándote de añadir un espacio en blanco al principio.



Por último, arrastra un bloque de Variables **dar a mi variable el valor 0** hasta el final de tu secuencia. Haz clic en la flecha desplegable junto a "mi variable" y haz clic en "tiempo" en la lista. Luego sustituye el "0" por un bloque informador **cronómetro** de la categoría Sensores. Ya puedes probar tu juego haciendo clic en la bandera verde que está sobre el área del escenario. Prepárate para pulsar la tecla **ESPACIO** en cuanto veas el mensaje "Presiona latecla espacio" (**Figura 4-13**). ¡A ver si eres capaz de superar nuestra puntuación máxima!

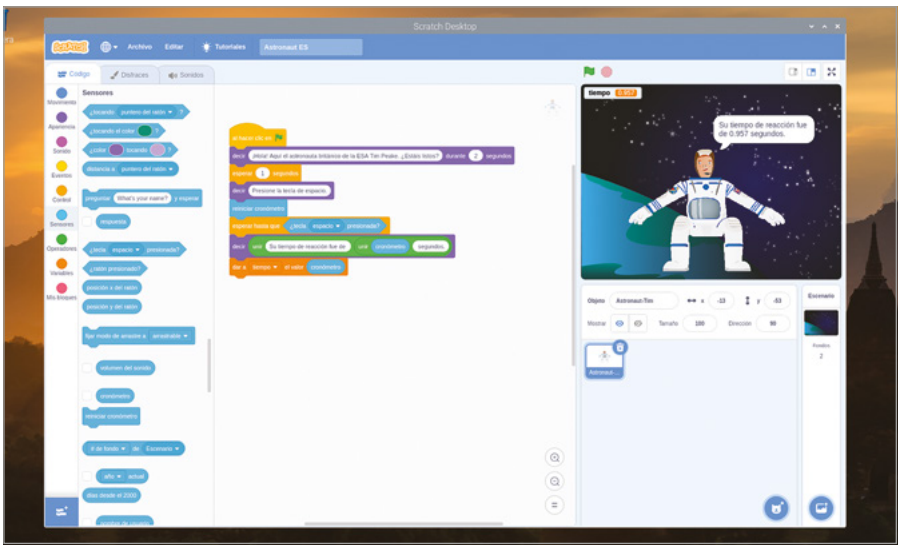
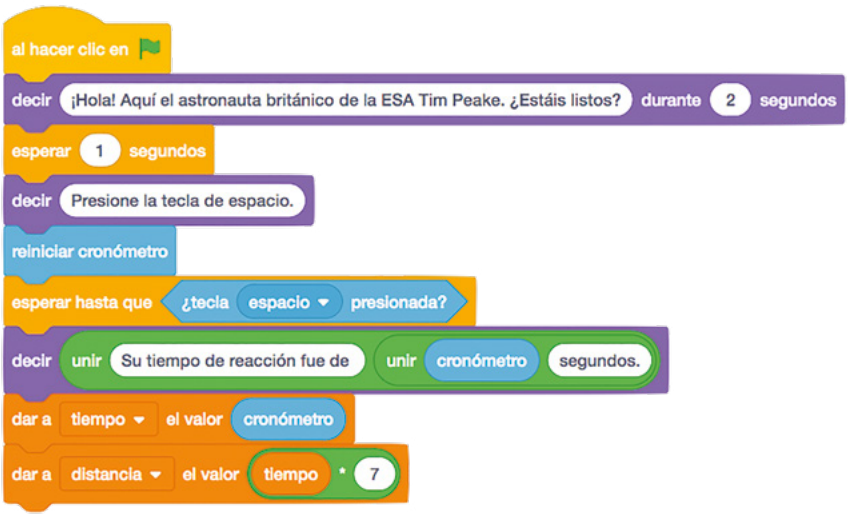


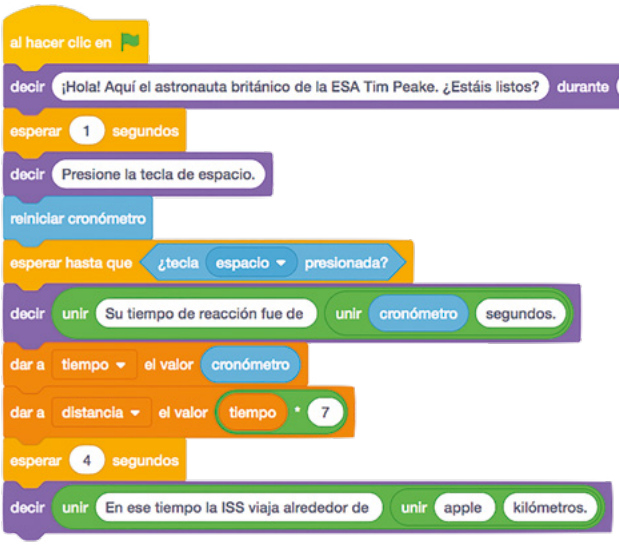
Figura 4-13: ¡A jugar!

Puedes seguir ampliando este proyecto, haciendo que calcule aproximadamente la distancia recorrida por la Estación Espacial Internacional en el tiempo que has tardado en pulsar **ESPACIO**. El cálculo se basa en la velocidad notificada por la estación: 7 kilómetros por segundo. Primero, crea una nueva variable y llámala "distancia". Observarás que los bloques de la categoría Variables cambian automáticamente para mostrar la nueva variable, pero los bloques de tiempo existentes en tu programa siguen siendo los mismos.

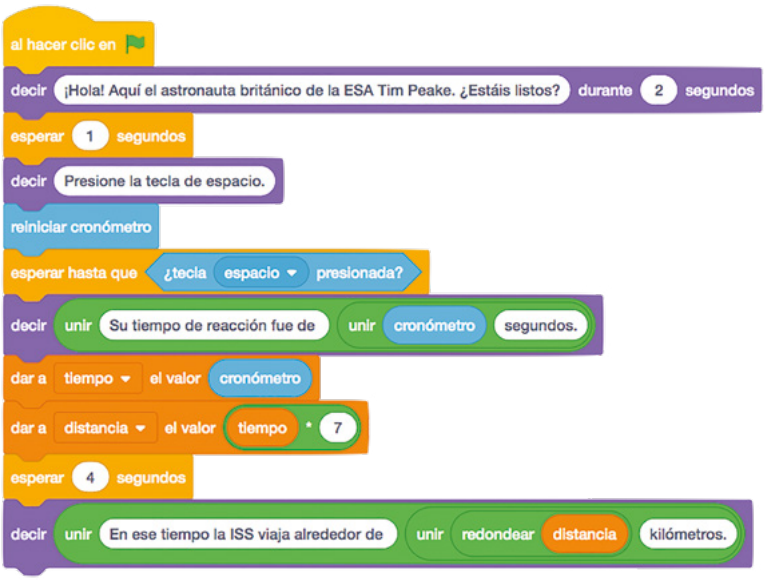
Añade un bloque dar a distancia el valor 0 y arrastra para colocar sobre el "0" un bloque de Operadores * , que indica una multiplicación. Arrastra un bloque informador tiempo sobre el primer espacio en blanco y escribe el número "7" en el segundo espacio. Cuando termines, el bloque combinado dirá dar a distancia el valor tiempo * 7. El tiempo que has tardado en pulsar **ESPACIO** se multiplicará por siete para obtener la distancia en kilómetros recorrida por la Estación Espacial (ISS).



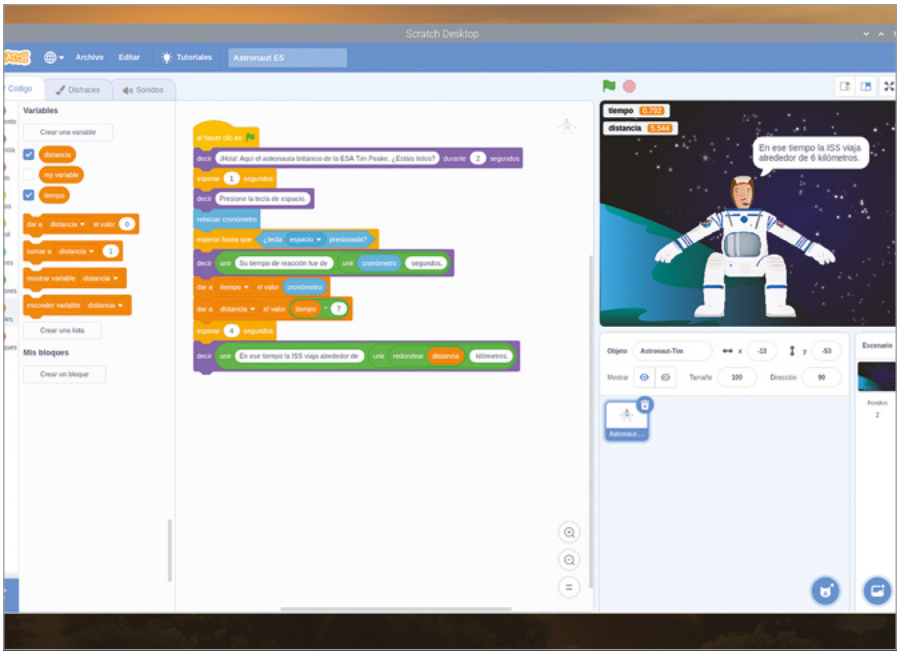
Añade un bloque esperar 1 segundos y cambia el número a "4". Por último, arrastra otro bloque decir ¡Hola! hasta el final de tu secuencia y añade dos bloques unir, como ya has hecho anteriormente. En el primer espacio, sobre "apple" escribe "En ese tiempo la ISS viaja alrededor de ", y recuerda incluir el espacio en blanco al final. En el espacio "banana", escribe "kilómetros", y recuerda incluir el espacio al principio.



Por último, arrastra un bloque redondear de la categoría Operadores hasta el espacio en blanco intermedio y luego arrastra un bloque informador distancia al nuevo espacio en blanco creado. El bloque redondear redondea los números hacia arriba o hacia abajo hasta el entero más cercano, así que en lugar de un número de kilómetros superexacto pero difícil de leer, se obtiene un número entero fácil de leer.



Haz clic en la bandera verde para ejecutar el programa y observa la distancia recorrida por la Estación Espacial en el tiempo que tardas en pulsar la tecla **ESPACIO**. ¡Acuérdate de guardar tu programa al terminar, para poder cargarlo posteriormente sin tener que empezar desde cero!



▲ **Figura 4-14:** Tim te dice la distancia que ha recorrido la Estación Espacial Internacional (ISS)

RETO: ¿QUIÉNES SON RÁPIDOS?

Además de los astronautas, ¿qué otras profesiones necesitan reflejos rapidísimos? ¿Puedes dibujar tus propios objetos y fondos para ilustrar una de esas profesiones?

Proyecto 2: Natación sincronizada

La mayoría de los juegos usan más de un botón y este proyecto lo demuestra ofreciendo un control de dos botones mediante las teclas ← y → del teclado.

PROYECTO ONLINE

Este proyecto también está disponible online en **rpf.io/synchro-swimming**

Crea un proyecto y guárdalo con el nombre "Natación sincronizada". Haz clic en Escenario, en la sección de control de escenario, y luego en la pestaña Fondos, en la parte superior izquierda. Haz clic en el botón Convertir a mapa de bits, situado bajo el fondo. En la paleta Rellenar, elige un color azulado que represente el agua, haz clic en el icono Rellenar y luego en el fondo cuadriculado para rellenarlo de azul (**Figura 4-15**).

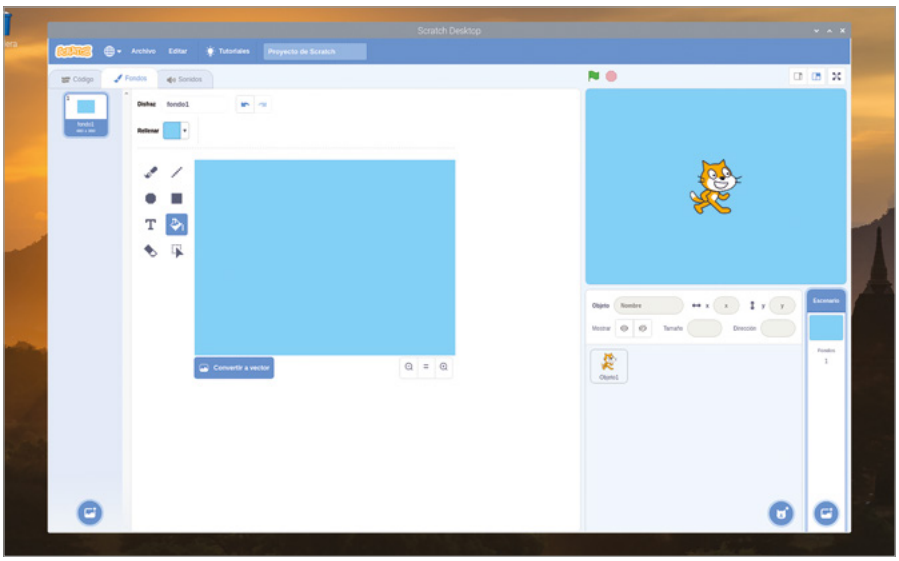


Figura 4-15: Rellenar el fondo en azul

Haz clic con el botón derecho en el objeto gato en la lista y haz clic en "borrar". Haz clic en el icono "Elige un objeto" para ver una lista de los objetos integrados. Haz clic en la categoría Animales y luego en "Cat Flying" (**Figura 4-16**) y Aceptar. Este objeto también es útil para proyectos de natación.

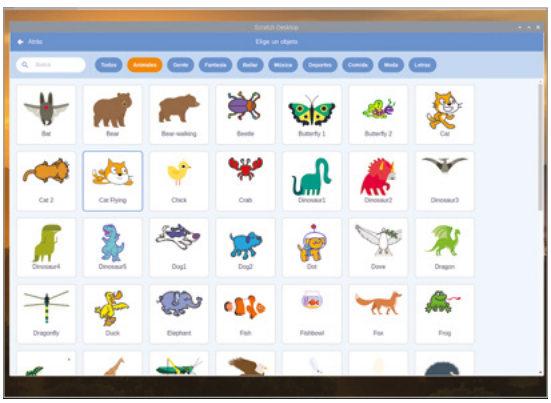


Figura 4-16: Elegir un objeto de la biblioteca

Haz clic en el nuevo objeto y arrastra al área de código dos bloques al presionar tecla espacio de la categoría Eventos. Haz clic en la flecha desplegable junto a la palabra "espacio" en el primer bloque y elige "flecha izquierda" en la lista de opciones. Arrastra un bloque girar 15 grados de la categoría Movimiento y suéltalo bajo el bloque

al presionar tecla flecha izquierda. Luego haz lo mismo con tu segundo bloque de Eventos, pero en este caso elige la opción "flecha derecha" en la lista y usa un bloque de Movimiento girar 15 grados.

Pulsa la tecla ← o → para probar tu programa. Verás que el gato gira en la dirección que elijas en el teclado. En esta ocasión no has tenido que hacer clic en la bandera verde, porque los bloques activadores de la categoría Eventos que has usado están activos en todo momento, incluso cuando el programa no está en plena ejecución.

Repite los pasos anteriores otras dos veces, pero ahora elige la "flecha arriba" y la "flecha abajo" para los bloques activadores de Eventos. Luego elige los bloques de Movimiento mover 10 pasos y mover -10 pasos. Pulsa las teclas de flecha en el teclado y verás que ahora tu gato puede girar y nadar hacia adelante y hacia atrás.

Para hacer el movimiento del gato más realista, puedes cambiar su aspecto, lo que en Scratch se denomina *disfraz*. Haz clic en el objeto gato y luego en la pestaña Disfraces, sobre la paleta de bloques. Haz clic en el disfraz "cat flying-a" y luego en el icono de la X sobre la papelera en la esquina superior derecha, para eliminar el disfraz. A continuación, haz

clic en el disfraz "cat flying-b" y usa el cuadro de nombre de la parte superior para asignar el nombre "derecha" (**Figura 4-17**).

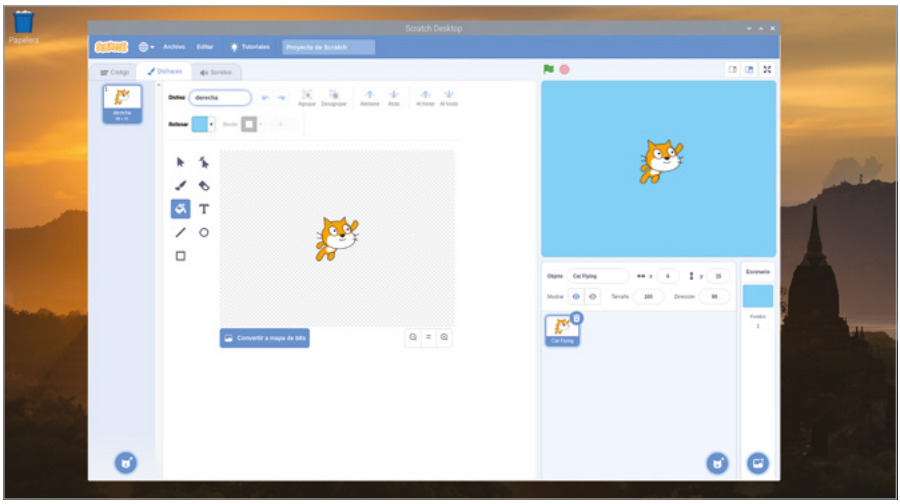


Figura 4-17: Cambiar el nombre de disfraz a "derecha"

Haz clic con el botón derecho en el disfraz que ahora se llama "derecha" y haz clic en "duplicar" para crear una copia. Haz clic en la copia para seleccionarla, luego en el icono Seleccionar y en Voltear horizontalmente, y cambia el nombre del disfraz a "izquierda" (**Figura 4-18**). Al final tendrás dos "disfraces" para tu objeto, reflejo exacto el uno del otro: una imagen se llama "derecha" con el gato mirando hacia la derecha y la otra se llama "izquierda" y el gato mira hacia la izquierda.

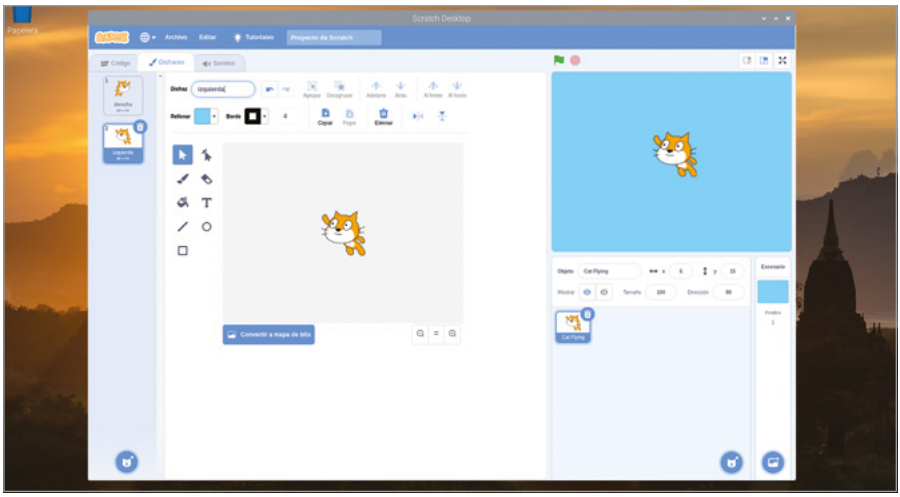


Figura 4-18: Duplicar el disfraz, voltearlo y asignarle el nombre "izquierda"

Haz clic en la pestaña Código sobre el área de disfraces y arrastra dos bloques de Apariencia cambiar disfraz a izquierda bajo tus bloques de eventos de flecha izquierda y flecha derecha, y cambia el situado bajo el bloque de flecha derecha a cambiar disfraz a derecha. Vuelve a probar la teclas de flecha y verás que ahora el gato parece mirar en la dirección en la que nada.

Pero para la natación sincronizada de estilo olímpico necesitamos más nadadores y también tenemos que restablecer la posición del objeto gato. Añade un bloque de Eventos al hacer clic en y debajo de este un bloque de Movimiento ir a x: 0 y: 0 (cambiando los valores según sea necesario) y un bloque de Movimiento apuntar en dirección 90. Ahora, cuando hagas clic en la bandera verde, el gato se moverá al centro del escenario y apuntará a la derecha.

Para crear más nadadores, añade un bloque repetir 6 (debes cambiar el valor predeterminado de 10 a 6) y dentro de él añade un bloque de la categoría Control crear clon de mí mismo. Para que los nadadores no naden todos en la misma dirección, añade un bloque girar 60 grados encima del bloque crear clon pero aún dentro del bloque repetir 6. Haz clic en la bandera verde y prueba las teclas de flecha para ver a tus nadadores en acción.

Para completar el ambiente olímpico, tendrás que añadir música. Haz clic en la pestaña Sonidos situada sobre la paleta de bloques y haz clic en el icono "Elige un sonido"  . Haz clic en la categoría Bucles y busca en la lista (**Figura 4-19**) hasta que encuentres algo que te guste. Nosotros hemos elegido "Dance Around". Haz clic en el botón Aceptar para elegir la música y luego en la pestaña Código para volver a abrir esa área.

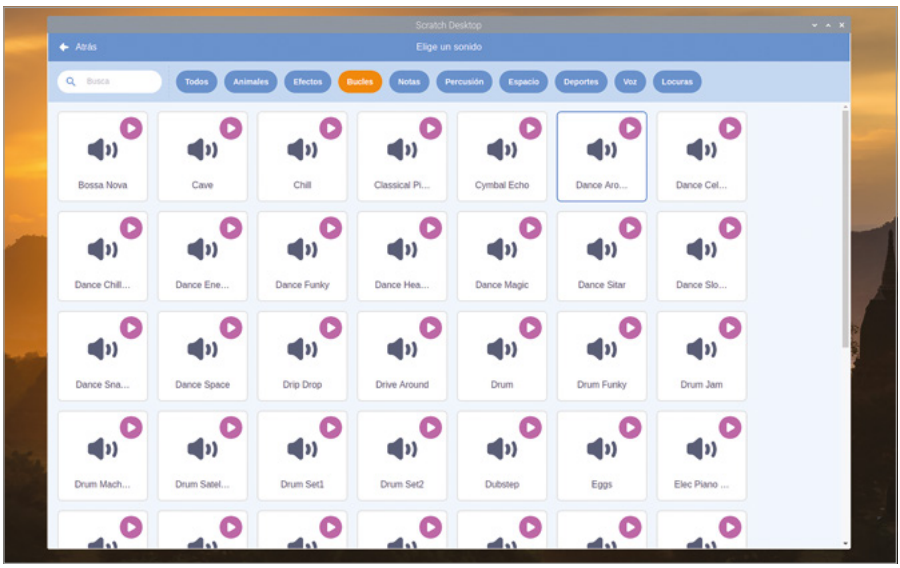
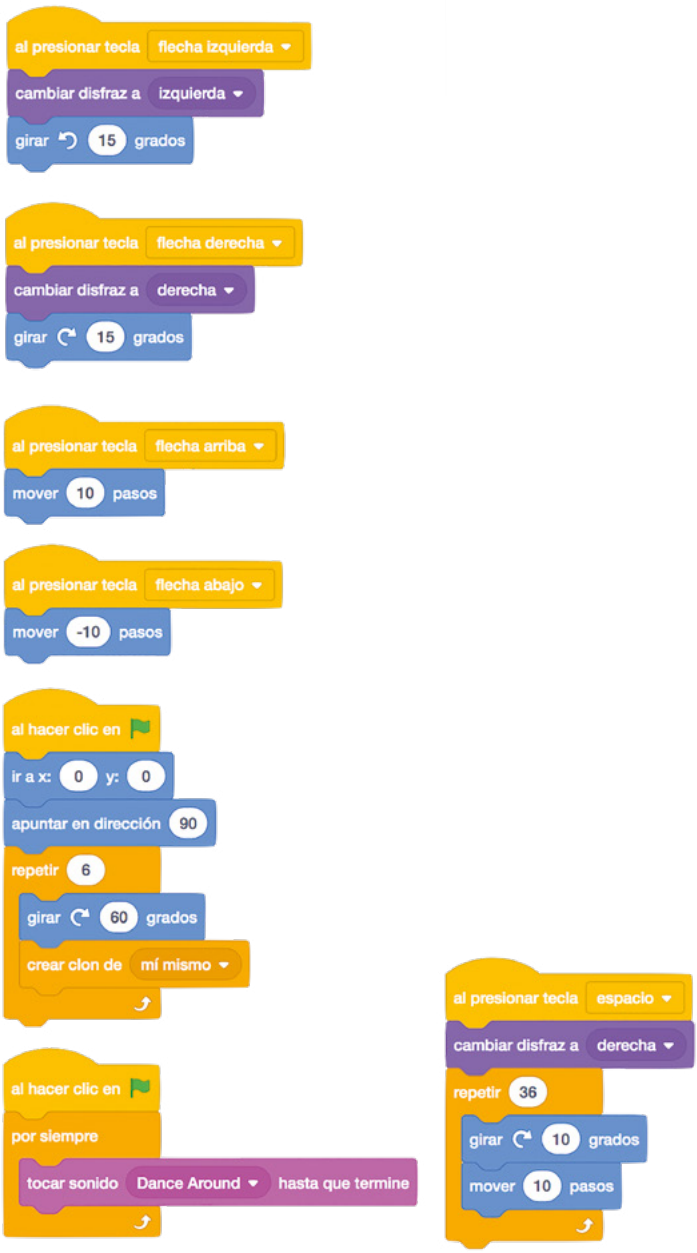


Figura 4-19: Seleccionar un bucle de música en la biblioteca de sonidos

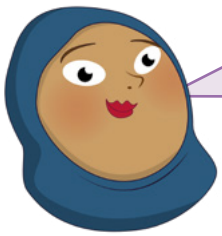
Añade a tu área de código otro bloque de Eventos al hacer clic en y luego añade un bloque de Control por siempre. Dentro del bloque de Control, añade un bloque tocar sonido Dance Around hasta que termine (acuérdate de comprobar el nombre de la pieza musical que elijas) y haz clic en la bandera verde para probar tu nuevo programa. Si quieres detener la música, haz clic en el octógono rojo para detener el programa y silenciar el sonido.

Por último, puedes simular una rutina de baile completa agregando un nuevo activador de eventos a tu programa. Añade un bloque de Eventos al presionar tecla espacio y luego un bloque cambiar disfraz a derecha. Bajo este, añade un bloque repetir 36 (acuérdate de cambiar el valor predeterminado) y dentro de este bloque añade un bloque girar 10 grados y otro mover 10 pasos.



Haz clic en la bandera verde para iniciar el programa y luego pulsa la tecla **ESPACIO** para probar la rutina nueva (**Figura 4-20**, a continuación). Acuérdate de guardar tu programa cuando termines.

▲ **Figura 4-20:** La rutina de natación completa



RETO: TU PROPIA RUTINA

¿Puedes crear tu propia rutina de natación sincronizada usando bucles? ¿Qué tendrías que cambiar si quisieras más o menos nadadores? ¿Puedes añadir varias rutinas de natación que se activen mediante distintas teclas del teclado?

Proyecto 3: Juego de tiro con arco

Como ya tienes bastante experiencia con Scratch, es hora de trabajar en algo un poco más complicado: un juego de tiro con arco, en el que el jugador tiene que acertar en la diana usando un arco y una flecha que se mueven al azar.

PROYECTO ONLINE

Este proyecto también está disponible online en **rpf.io/archery**

Para empezar abre el navegador web Chromium, escribe **rpf.io/p/en/archery-go** y pulsa la tecla **ENTRAR**. Los recursos del juego se descargan como un archivo zip que tendrás que descomprimir (haz clic con el botón derecho del ratón y selecciona Extraer aquí). Vuelve a Scratch 3 y haz clic en el menú Archivo seguido de "Cargar desde tu ordenador". Haz clic en **ArcheryResources.sb3** y

pulsa el botón Abrir. Se te preguntará si quieres reemplazar el contenido de tu proyecto actual: si no has guardado los cambios, haz clic en Cancelar y guárdalos, de lo contrario, haz clic en Aceptar.

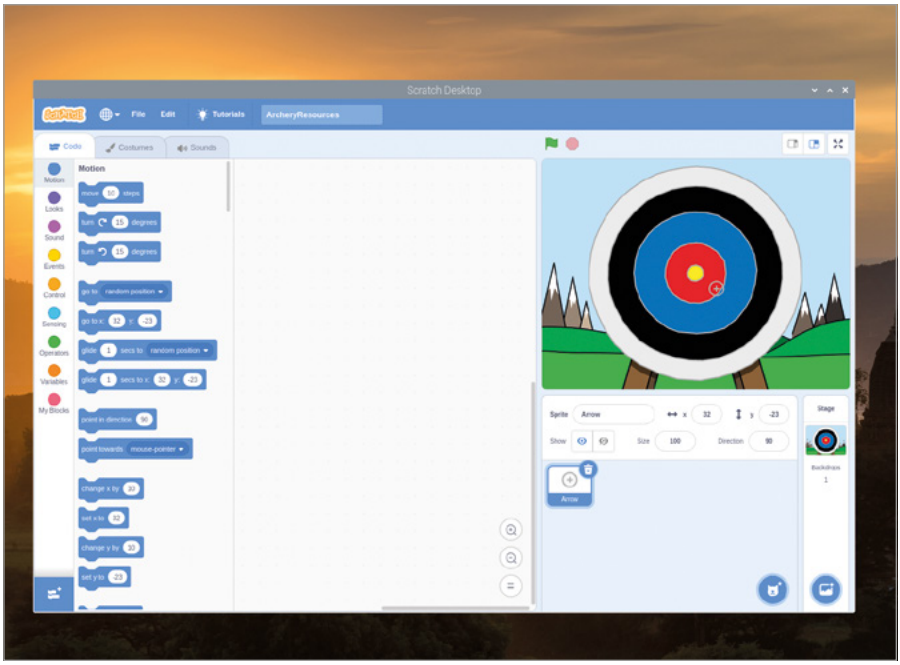


Figura 4-21: Proyecto de recursos cargado para el juego de tiro con arco

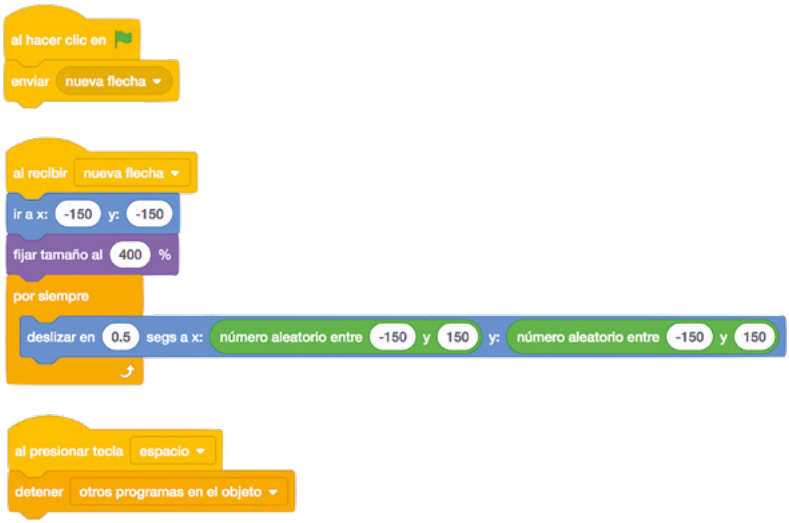
El proyecto que acabas de cargar contiene un fondo y un objeto (**Figura 4-21**), pero no tiene el código requerido para crear un juego: esa es tu misión. Empieza por añadir un bloque al hacer clic en y un bloque enviar mensaje1. Haz clic en la flecha hacia abajo al final del bloque y luego en "Nuevo mensaje", y escribe "nueva flecha" antes de hacer clic en el botón Aceptar. Ahora el bloque dice enviar nueva flecha.

Los mensajes son partes de tu programa que cualquier otra parte de dicho programa puede recibir. Para que se realice una acción, añade un bloque al recibir mensaje1 y repite el cambio anterior para que diga al recibir nueva flecha. Esta vez puedes hacer clic en la flecha abajo y elegir "nueva flecha" en la lista, no hace falta que vuelvas a crear el mensaje.

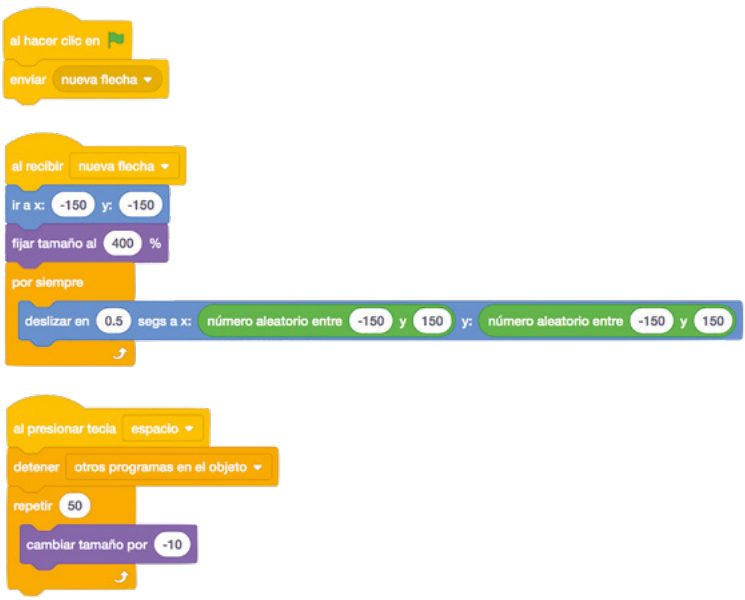
Debajo del bloque al recibir nueva flecha, añade un bloque ir a x: -150 y: -150 y otro fijar tamaño al 400 %. Recuerda que estos no son los valores predeterminados de esos bloques, así que tendrás que cambiarlos después de arrastrarlos al área de código. Haz clic en la bandera verde para ver lo que has hecho hasta ahora: el objeto flecha, que el jugador utiliza para apuntar al objetivo, saltará a la parte inferior izquierda del escenario y se cuadruplicará en tamaño.

Para plantearle un reto al jugador, añade movimiento oscilante al preparar el arco y apuntar. Arrastra un bloque por siempre seguido de uno deslizar 1 segundos a x: -150 y: -150. Edita el primer cuadro blanco para que diga "0.5" en lugar de "1" y coloca un bloque de Operadores número aleatorio entre -150 y 150 en cada uno de los otros dos bloques blancos. Esto significa que la flecha se moverá por el escenario en una dirección aleatoria, recorriendo una distancia aleatoria: eso hace mucho más difícil dar en el blanco.

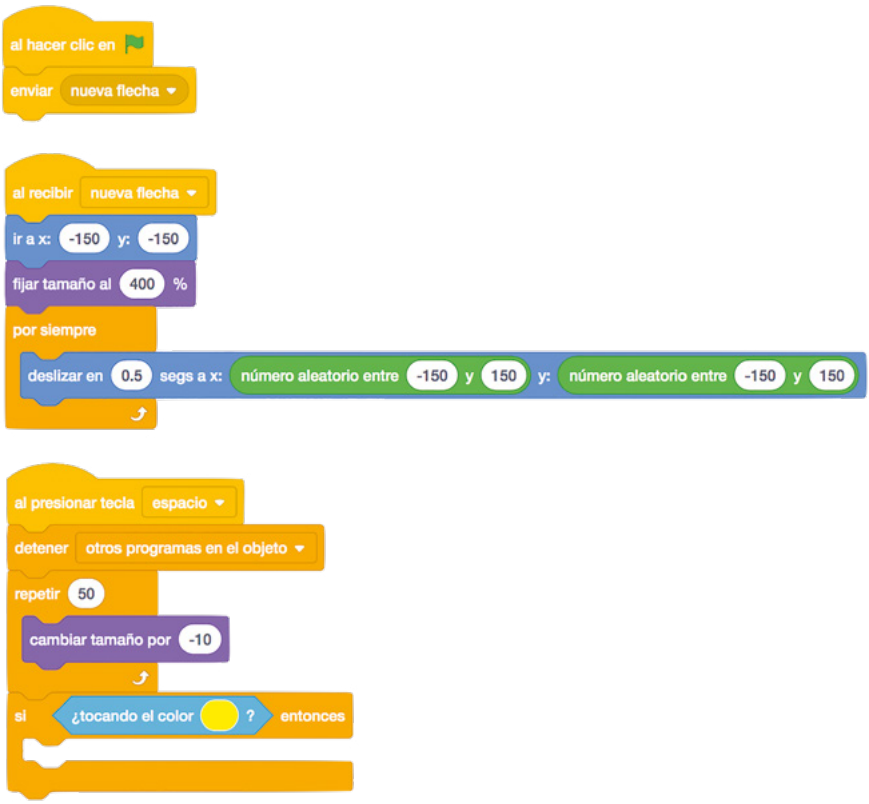
Vuelve a hacer clic en la bandera verde para ver la acción del bloque: tu flecha se mueve sin rumbo por el escenario, cubriendo distintas partes de la diana. Pero de momento no dispones de un modo de disparar la flecha para que dé en la diana. Arrastra al área de código un bloque al presionar tecla espacio seguido de un bloque de Control detener todos. Haz clic en la flecha abajo al final del bloque y cámbiala a detener otros programas en el objeto.



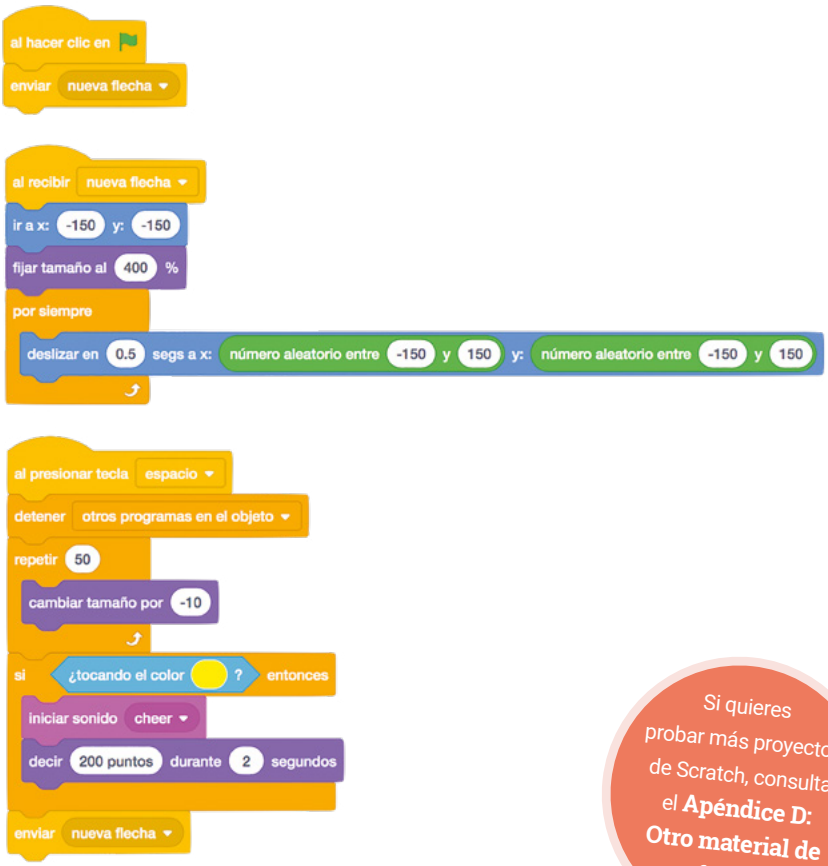
Si has detenido el programa para añadir los bloques nuevos, haz clic en la bandera verde para iniciarlo de nuevo y luego pulsa la tecla **ESPACIO**: verás que el objeto flecha deja de moverse. Por algo se empieza, pero debes hacer que parezca que la flecha vuela hacia la diana. Añade un bloque repetir 50 seguido de uno cambiar tamaño por -10 y haz clic en la bandera verde para volver a probar el juego. Esta vez, la flecha parece volar hacia la diana.



Para hacer el juego más divertido, vas a hacer que registre una puntuación. Desde la misma pila de bloques, añade uno **si entonces** y asegúrate de colocarlo debajo del bloque **repetir 50** y no dentro de él, con un bloque de Sensores **¿tocando el color?** en el hueco hexagonal. Para elegir el color correcto, haz clic en el cuadro de color al final del bloque, luego en el icono del cuentagotas y luego en el centro de la diana amarillo en el escenario.



Para que el jugador sepa cuántos puntos ha obtenido, añade un bloque **iniciar sonido cheer** y uno **decir 200 puntos durante 2 segundos** dentro del bloque **entonces**. Por último, añade un bloque **enviar nueva flecha** al final de la pila de bloques, debajo y fuera del bloque **si entonces**, para darle al jugador otra flecha cada vez que dispare una. Haz clic en la bandera verde para iniciar el juego e intenta dar en la diana amarilla: cuando lo hagas, se te recompensará con una ovación del público y 200 puntos.



Si quieres probar más proyectos de Scratch, consulta el **Apéndice D: Otro material de referencia**

El juego funciona pero es difícil dar en el blanco. Utilizando lo que has aprendido en este capítulo, intenta ampliarlo para otorgar puntos al acertar en otras partes de la diana, aparte del centro: 100 puntos por el círculo rojo, 50 puntos por el azul, etc.

RETO: ¿PUEDES MEJORARLO?

¿Cómo harías el juego más fácil? ¿Cómo lo harías más difícil? ¿Puedes usar variables para que la puntuación del jugador aumente a medida que dispara más flechas? ¿Puedes agregar un cronómetro de cuenta atrás para aumentar la tensión del jugador?

Capítulo 5

Programar con Python

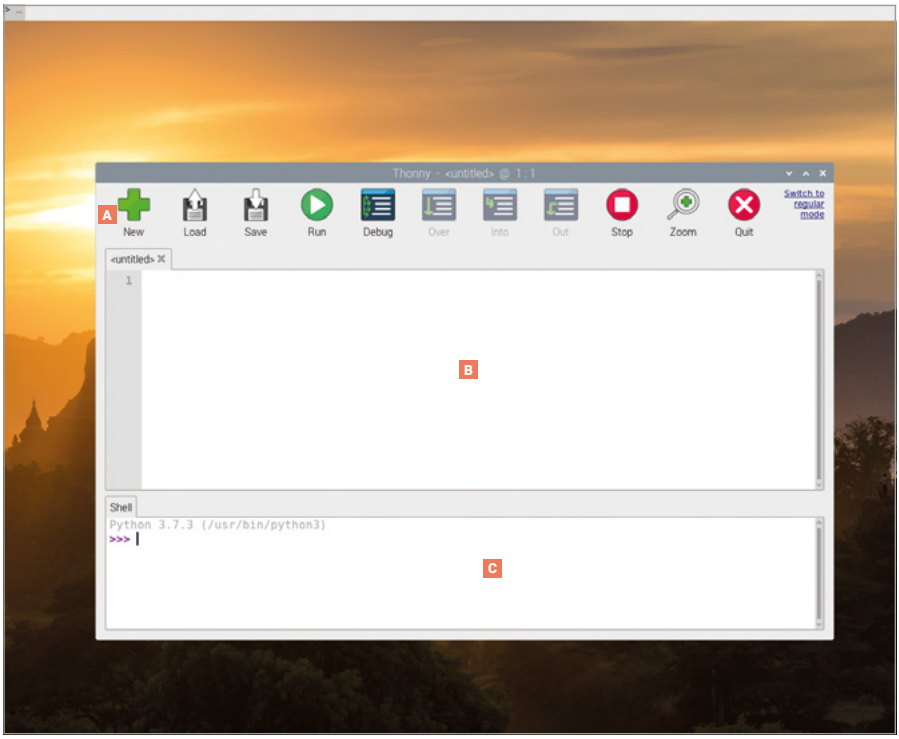
Ahora que ya te has familiarizado con Scratch, te mostraremos cómo hacer una codificación basada en texto con Python

Python, creado por Guido van Rossum, debe su nombre al grupo de humoristas británico Monty Python y desde sus comienzos como proyecto por afición y su lanzamiento al público en 1991 se ha convertido en un apreciadísimo lenguaje de programación para todo tipo de proyectos. A diferencia del entorno visual de Scratch, Python se basa en texto: el programador escribe instrucciones, utilizando un lenguaje simplificado y un formato específico, y luego el ordenador las lleva a cabo.

Python es un gran paso adelante para quienes ya han usado Scratch, con más flexibilidad y un entorno de programación más "tradicional". Pero eso no quiere decir que sea difícil de aprender: con un poco de práctica, cualquiera puede escribir programas para cualquier cosa, ya sean cálculos simples o juegos increíblemente complicados.

Este capítulo reutiliza términos y conceptos ya introducidos en el **Capítulo 4, Programar con Scratch 3**. Si aún no has realizado los ejercicios de ese capítulo, te convendría retroceder y hacerlos antes, para que este te resulte más fácil de seguir.

El entorno Thonny Python IDE



A **Barra de herramientas** – La interfaz del "modo simple" de Thonny utiliza como menú una barra de iconos que te permiten crear, guardar, cargar y ejecutar programas en Python, y probarlos de varias maneras.

B **Área de script** – El área de script es donde se escriben los programas en Python y se divide en un área principal para el programa y un pequeño margen lateral para mostrar los números de línea.

C **Shell de Python** – El Shell Python te permite escribir instrucciones individuales que se ejecutan en cuanto pulsas la tecla **ENTRAR** y te proporciona información sobre la ejecución de los programas.

VERSIONES DE THONNY

Thonny tiene dos versiones de interfaz: La de modo Normal y la de modo Simple, que es la más adecuada para principiantes. Este capítulo utiliza el modo Simple, que se carga de forma predeterminada al abrir Thonny desde la sección de programación del menú de Raspberry.

Tu primer programa en Python: ¡Hola mundo!

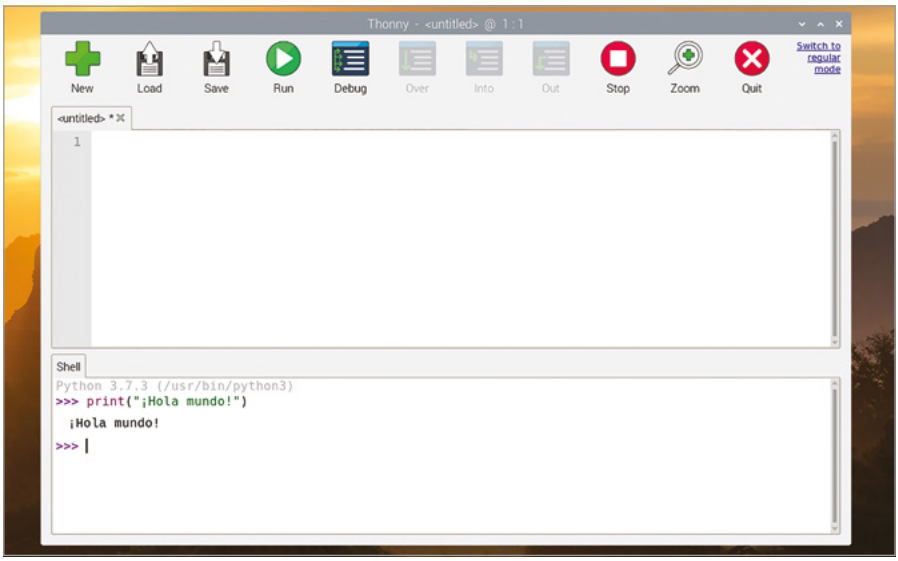
Al igual que los otros programas ya preinstalados en Raspberry Pi, Thonny está disponible en el menú: haz clic en el icono de Raspberry, mueve el cursor a la sección Programación y haz clic en Thonny Python IDE. La interfaz de usuario de Thonny (en modo simple, de forma predeterminada) se cargará al cabo de unos segundos.

Thonny es un paquete de *entorno de desarrollo integrado (IDE)*, algo que suena complicado pero tiene una sencilla explicación: reúne, o *integra*, todas las herramientas que necesitas para escribir, o *desarrollar*, software en una misma interfaz de usuario, o *entorno*. Hay multitud de entornos IDE disponibles: algunos admiten diferentes lenguajes de programación, mientras que otros, como es el caso de Thonny, se centran en un solo lenguaje.

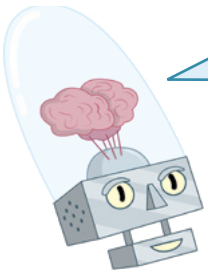
A diferencia de Scratch, que ofrece bloques visuales como base para crear un programa, Python es un lenguaje de programación más tradicional donde todo se escribe. Para empezar a crear tu primer programa, haz clic en el área de shell de Python, en la parte inferior izquierda de la ventana de Thonny y escribe la siguiente instrucción antes de pulsar la tecla **ENTRAR**:

```
print("¡Hola mundo!")
```

Al pulsar **ENTRAR**, verás que tu programa empieza a ejecutarse inmediatamente: Python responderá, en esa misma área de shell, con el mensaje '¡Hola mundo!' (**Figura 5-1**), tal y como lo has indicado. Eso es porque el shell es una línea directa al *intérprete* de Python, cuyo trabajo es examinar tus instrucciones e *interpretar* lo que significan. Esto se conoce como *modo interactivo*, imagínatelo como una conversación cara a cara con alguien: en cuanto terminas de decir algo, la otra persona responde y espera a que tú vuelvas a hablar.



▲ **Figura 5-1:** Python muestra el mensaje '¡Hola mundo!' en el área de shell



ERROR DE SINTAXIS

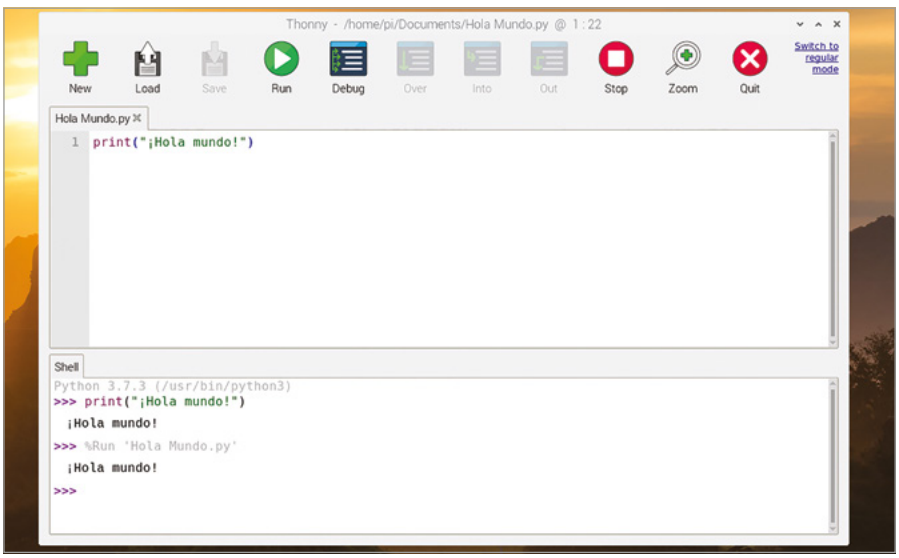
Si tu programa no se ejecuta y muestra un mensaje indicando 'syntax error' en el área de shell, hay un error en algo de lo que has escrito. Python requiere instrucciones escritas de una forma muy específica: si te saltas un paréntesis o una comilla, o te equivocas al escribir 'print' o lo escribes con P mayúscula, o si añades símbolos extra en algún lugar de una instrucción, no funcionará. De ser así, vuelve a escribirla y asegúrate de que coincide con lo indicado en esta guía antes de pulsar la tecla **ENTRAR**.

No es imprescindible que uses Python en modo interactivo. Haz clic en el área de script, en la parte izquierda de la ventana de Thonny, y escribe tu programa de nuevo:

```
print("¡Hola mundo!")
```

Al pulsar la tecla **ENTRAR** esta vez, no ocurre nada, aparte de aparecer una nueva línea vacía en el área de script. Para que esta versión del programa funcione, tendrás que hacer clic en el icono Run de la barra de herramientas de Thonny. Cuando lo hagas, se te pedirá que guardes tu programa: escribe un nombre descriptivo, como 'Hola mundo', y haz clic en el botón Save. Cuando se haya guardado el programa, verás dos mensajes en el área de shell de Python (**Figura 5-2**):

```
>>> %Run 'Hola mundo.py'
    ¡Hola mundo!
```



▲ **Figura 5-2:** Ejecutar tu programa simple

La primera de estas líneas es una instrucción de Thonny diciéndole al intérprete de Python que ejecute el programa que acabas de guardar. La segunda es el resultado del programa, el mensaje que le has dicho a Python que muestre. ¡Enhorabuena! Acabas de escribir y ejecutar tu primer programa en Python, en modo interactivo y en modo script.

RETO: NUEVO MENSAJE

¿Puedes cambiar el mensaje que el programa en Python muestra como resultado? Si quisieras añadir más mensajes, ¿usarías el modo interactivo o el modo script? ¿Qué ocurre si quitas los corchetes o las comillas del programa y luego intentas volver a ejecutarlo?

Pasos siguientes: bucles y sangría de código

Así como Scratch utiliza pilas de bloques similares a las de un rompecabezas para controlar qué partes del programa se conectan con otras, Python tiene su propia manera de controlar la secuencia en la que se ejecutan los programas: la *sangría*. Crea un nuevo programa haciendo clic en el icono New de la barra de herramientas de Thonny. No perderás el programa que has creado antes, sino que Thonny creará una nueva pestaña sobre el área de script. Para empezar escribe:

```
print("¡Empieza el bucle!")
for i in range (10):
```

La primera línea muestra un simple mensaje en el shell, igual que tu programa Hello world. La segunda comienza un bucle *definido* que funciona de la misma manera que en Scratch: se asigna al bucle un contador **`i`** con una cuenta de números (la instrucción **`range`**, para empezar en 0 y contar hacia delante pero sin llegar nunca al número 10). El símbolo de los dos puntos (**`:`**) le dice a Python que la siguiente instrucción debe ser parte del bucle.

En Scratch, las instrucciones que incluir en el bucle están dentro del bloque en forma de C. Python utiliza un enfoque diferente: código con sangría. La siguiente línea comienza con cuatro espacios en blanco, que Thonny debería haber añadido al pulsar **ENTRAR** después de la línea 2:

```
    print("Número de bucle", i)
```

Los espacios en blanco empujan esta línea hacia adentro, con relación a las otras líneas. Esta sangría es la forma en que Python detecta la diferencia entre las instrucciones fuera del bucle y las instrucciones dentro de él; el código sangrado se considera como "anidado".

Verás que al pulsar **ENTRAR** al final de la tercera línea, Thonny automáticamente sangra la siguiente, asumiendo que será parte del bucle. Para quitar esa sangría, simplemente pulsa la tecla **RETROCESO** una vez antes de escribir la cuarta línea:

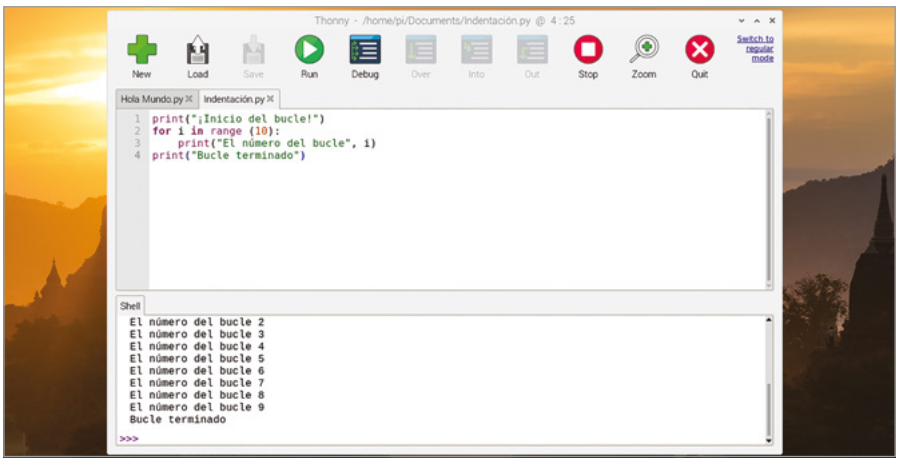
```
print("¡Acabó el bucle!")
```

Ya tienes tu programa completo de cuatro líneas. La primera se encuentra fuera del bucle y solo se ejecutará una vez; la segunda línea establece el bucle; la tercera se encuentra dentro del bucle y se ejecutará una vez con cada repetición del bucle; y la cuarta línea vuelve a estar fuera del bucle.

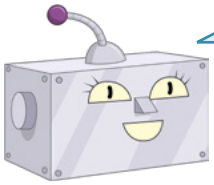
```
print("¡Empieza el bucle!")
for i in range (10):
    print("Número de bucle", i)
print("¡Acabó el bucle!")
```

Haz clic en el icono Run, guarda el programa como **Sangría** y examina el área de shell para ver el resultado **Figura 5-3**):

```
¡Empieza el bucle!
Número de bucle 0
Número de bucle 1
Número de bucle 2
Número de bucle 3
Número de bucle 4
Número de bucle 5
Loop number 6
Número de bucle 7
Número de bucle 8
Número de bucle 9
¡Acabó el bucle!
```



▲ **Figura 5-3:** Ejecutar un bucle



CONTAR DESDE CERO

Python es un lenguaje de índice cero, lo que significa que empieza a contar desde 0, no desde 1, por lo que tu programa muestra los números del 0 al 9 en lugar de hacerlo del 1 al 10. Si quieres, puedes cambiar ese comportamiento cambiando la instrucción range (10) a range (1, 11), o a los números que quieras.

La sangría es un componente importante de Python, y una de las razones más comunes para que un programa no funcione según lo previsto. Cuando investigues problemas en un programa, un proceso conocido como *depuración*, siempre debes comprobar la sangría, especialmente al anidar unos bucles dentro de otros.

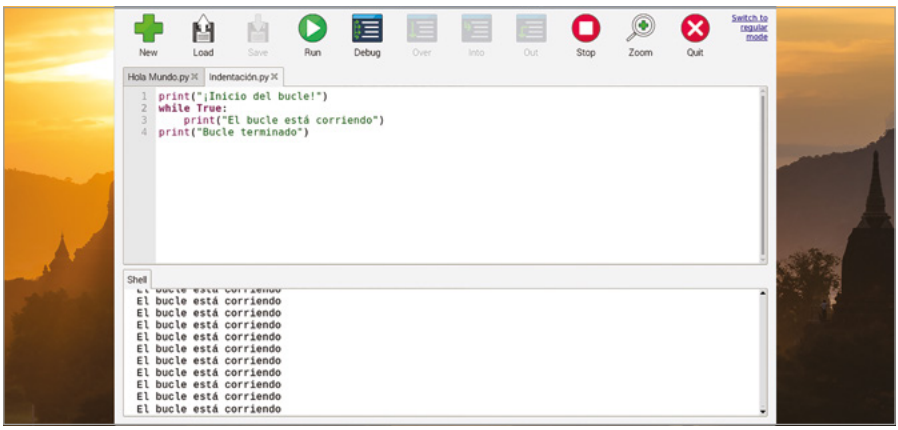
Python también admite el uso de bucles *infinitos*, que se ejecutan indefinidamente. Para cambiar tu programa de un bucle definido a uno infinito, edita la línea 2 para que diga:

```
while True:
```

Si haces clic en el icono Run ahora, verás el error **`name 'i' is not defined`**. Esto se debe a que has borrado la línea con la que se creaba y asignaba un valor a la variable **`i`**. Para corregirlo, simplemente edita la línea 3 para que deje de usar la variable:

```
    print("¡Bucle en ejecución!")
```

Haz clic en el icono Run y, si no te despistas, verás el mensaje de inicio de bucle '¡Empieza el bucle!' seguido de una cadena interminable de mensajes 'Bucle en ejecución' (**Figura 5-4**). El mensaje de fin de bucle, 'Acabó el bucle!', no se mostrará nunca, porque el bucle no tiene fin: cada vez que Python termina de mostrar el mensaje '¡Bucle en ejecución!', vuelve al principio del bucle y lo muestra de nuevo.



▲ **Figura 5-4:** Un bucle infinito se ejecuta constantemente hasta que detienes el programa

Haz clic en el icono Stop de la barra de herramientas de Thonny para decirle al programa que deje de hacer lo que está haciendo: eso se conoce como interrumpir el programa. Verás aparecer un mensaje en el área de shell de Python y el programa se detendrá sin haber llegado a la línea 4.

RETO: BUCLES

¿Puedes volver a convertir el bucle en definido? ¿Puedes añadir un segundo bucle definido al programa? ¿Cómo añadirías un bucle dentro de un bucle, y cómo debería funcionar?

Condicionales y variables

Las variables, como en todos los lenguajes de programación, no se limitan a controlar los bucles. Crea un nuevo programa haciendo clic en el icono New de la barra de herramientas de Thonny y escribe lo siguiente en el área de script:

```
userName = input ("¿Cómo te llamas?")
```

Haz clic en el icono Run, guarda el programa como **Name Test** y fíjate en lo que ocurre en el área de shell: pregunta por tu nombre. Escribe tu nombre en el área de shell y pulsa **ENTRAR**. Como esa es la única instrucción del programa, no pasará más (**Figura 5-5**). Si quieres hacer algo con los datos que has colocado en la variable, tendrás que añadir más líneas a tu programa.

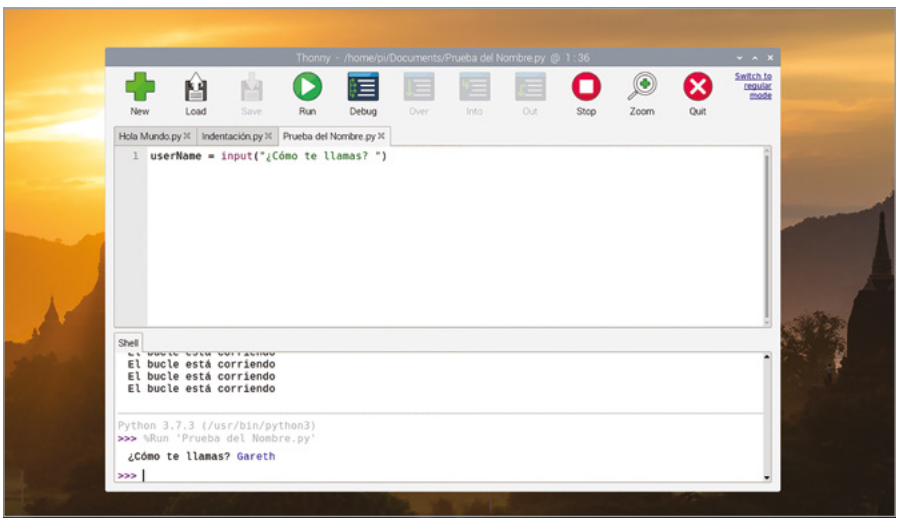


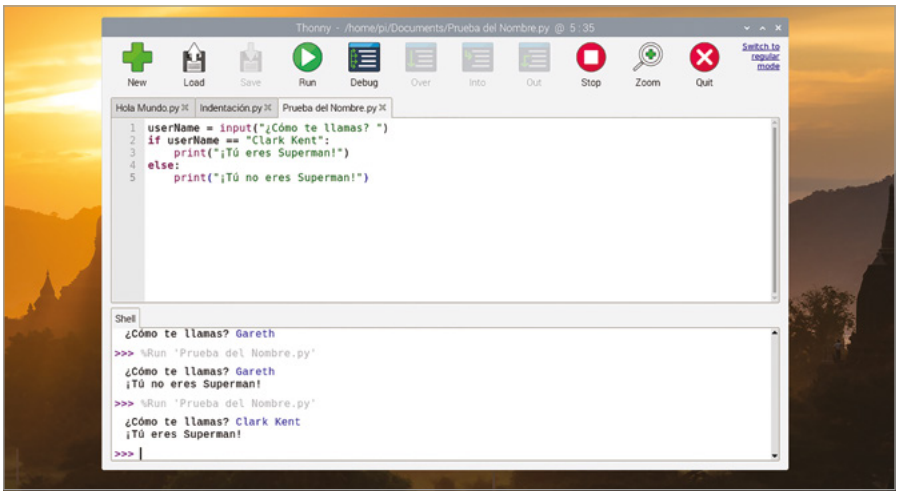
Figura 5-5: La función `input` permite pedir al usuario que introduzca texto

Para que tu programa haga algo útil con el nombre, añade una *instrucción condicional* escribiendo lo siguiente:

```
if userName == "Clark Kent":
    print("¡Eres Superman!")
else:
    print("¡No eres Superman!")
```

Recuerda que cuando Thonny vea que tu código debe tener una sangría, la aplicará automáticamente; pero no sabe cuándo debe dejar de aplicarla, así que tendrás que encargarte de borrar los espacios.

Haz clic en el icono Run y escribe tu nombre en el área de shell. A menos que de verdad te llames Clark Kent, verás el mensaje '¡No eres Superman!'. Vuelve a hacer clic en Run y esta vez escribe el nombre 'Clark Kent' asegurándote de escribirlo exactamente como en el programa, con C y K mayúsculas. Esta vez, el programa reconoce que realmente eres Superman (**Figura 5-6**).



▲ **Figura 5-6:** Deberías estar intentando salvar el mundo

El símbolo **==** indica a Python que haga una comparación directa, para ver si la variable **userName** concuerda con el texto (o *cadena*) en tu programa. Si trabajas con números, hay otras comparaciones que puedes hacer: **>** para ver si un número es mayor que otro número, **<** para ver si es menor, **=>** para ver si es igual o mayor que, **=<** para ver si es igual o menor que. También está **!=**, que significa no igual a y es justo lo contrario de **==**. El nombre técnico de estos símbolos es *operadores de comparación*.

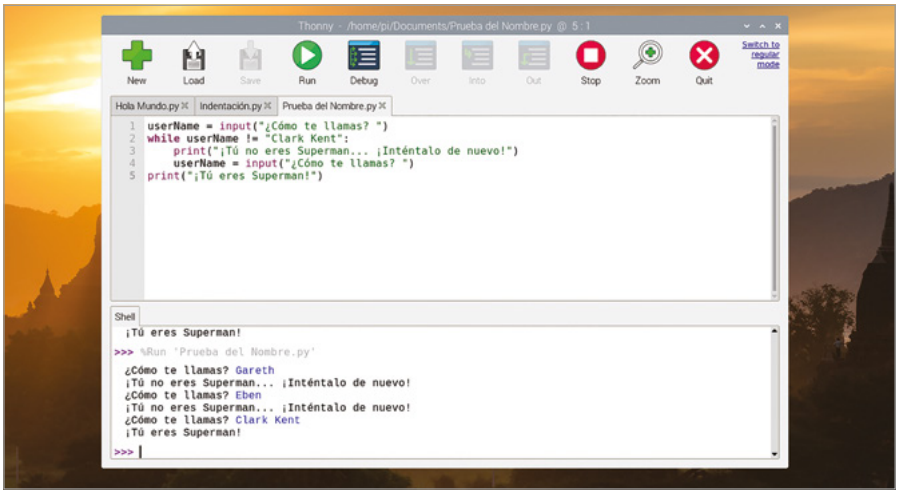
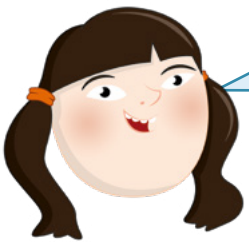


Figura 5-7: El programa insiste hasta que digas que te llamas 'Clark Kent'



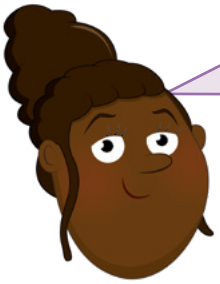
USAR = Y ==

La clave para usar variables es conocer la diferencia entre = y ==. Recuerda: = significa "hacer que esta variable sea igual a este valor", mientras que == significa "comprobar si la variable es igual a este valor". Si te confundes al usarlos, seguro que tus programas no funcionan.

Los operadores de comparación también pueden utilizarse en los bucles. Borra las líneas de la 2 a la 5 y luego escribe lo siguiente en su lugar:

```
while userName != "Clark Kent":
    print("No eres Superman. ¡Vuelve a intentarlo!")
    userName = input ("¿Cómo te llamas? ")
print("¡Eres Superman!")
```

Vuelve a hacer clic en el icono Run. Esta vez, en lugar de abandonar, el programa seguirá preguntando tu nombre hasta que confirmes que eres Superman (**Figura 5-7**, de modo similar a una contraseña muy simple. Para salir del bucle, escribe 'Clark Kent' o haz clic en el icono Stop en la barra de herramientas de Thonny. ¡Enhorabuena! Has aprendido a usar condicionales y operadores de comparación.



RETO: AÑADE PREGUNTAS

¿Puedes cambiar el programa para hacer más de una pregunta, almacenando las respuestas en múltiples variables? ¿Puedes hacer que un programa que utiliza condicionales y operadores de comparación muestre si un número tecleado por el usuario es mayor o menor que 5, como el programa que creaste en el **Capítulo 4, Programar con Scratch**?

Proyecto 1: Copos de nieve con Turtle

Ahora que entiendes cómo funciona Python, puedes experimentar con gráficos para crear un copo de nieve usando una herramienta conocida como *turtle*, o sea, tortuga.

PROYECTO ONLINE

Este proyecto también está disponible online en **rpf.io/turtle-snowflakes**

Las tortugas, en un principio robots físicos con la forma de esos animales, están diseñadas para moverse en línea recta, girar, y levantar y bajar un lápiz; en la versión digital simplemente se empieza o se deja de dibujar una línea al moverse la tortuga. A diferencia de otros lenguajes, entre ellos Logo y sus muchas variantes, Python no tiene integrada una herramienta "tortuga", sino una *biblioteca* de código complementario para lograr el mismo propósito. Las bibliotecas son paquetes de código que añaden nuevas instrucciones con las que expandir las capacidades de Python y se agregan a tus programas mediante un comando de importación.

Crea un programa haciendo clic en el icono New y escribe lo siguiente:

```
import turtle
```

Cuando escribas instrucciones incluidas en una biblioteca, debes usar el nombre de la biblioteca seguido de un punto y luego el nombre de la instrucción. Como puede ser engorroso tener que escribirlo una y otra vez, puedes asignar un nombre de variable más corto: podría ser solo una letra, pero creemos que es una buena idea que sirva también como nombre de mascota para la tortuga. Escribe lo siguiente:

```
pat = turtle.Turtle()
```

Para probar tu programa, tendrás crear alguna tarea para la tortuga. Escribe:

```
pat.forward(100)
```

Haz clic en el icono Run y guarda tu programa como **Turtle Snowflakes**. Cuando el programa se haya guardado, aparecerá una nueva ventana llamada "Turtle Graphics" y verás el resultado: tu tortuga, Pat, se moverá hacia adelante 100 unidades, trazando una línea recta (**Figura 5-8**).

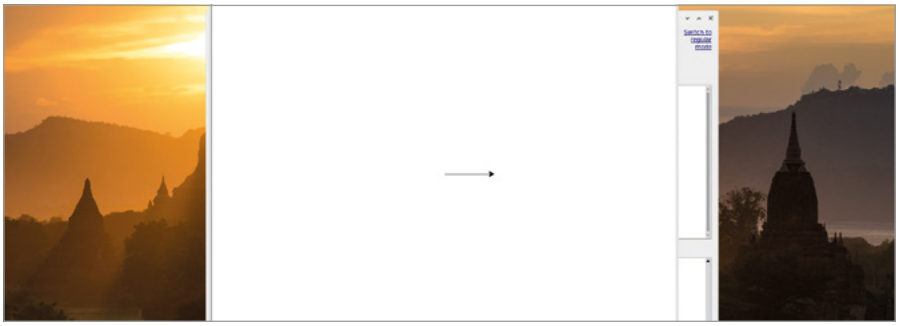


Figura 5-8: La tortuga se mueve hacia adelante para dibujar una línea recta

Vuelve a la ventana principal de Thonny. Si está oculta detrás de la ventana de Turtle Graphics, haz clic en el botón de minimizar en dicha ventana o haz clic en la entrada de Thonny en la barra de tareas en la parte superior de la pantalla. Haz clic en el botón Stop para cerrar la ventana de Turtle Graphics.

Escribir cada instrucción de movimiento a mano sería tedioso, así que borra la línea 3 y crea un bucle para ahorrarte trabajo creando formas:

```
for i in range(2):
    pat.forward(100)
    pat.right(60)
    pat.forward(100)
    pat.right(120)
```

Ejecuta tu programa y Pat dibujará un paralelogramo (**Figura 5-9**).

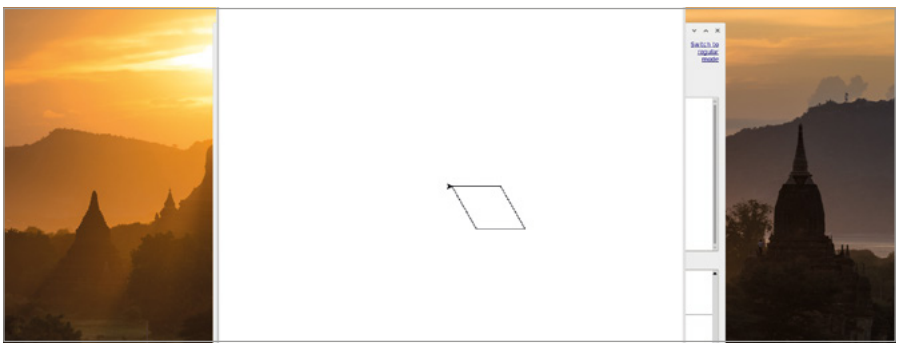


Figura 5-9: Combinando giros y movimientos, puedes dibujar formas

Para convertir el paralelogramo en una forma parecida a un copo de nieve, haz clic en el icono Stop de la ventana principal de Thonny y crea un bucle alrededor del mismo añadiendo la siguiente línea como línea 3:

```
for i in range(10):
```

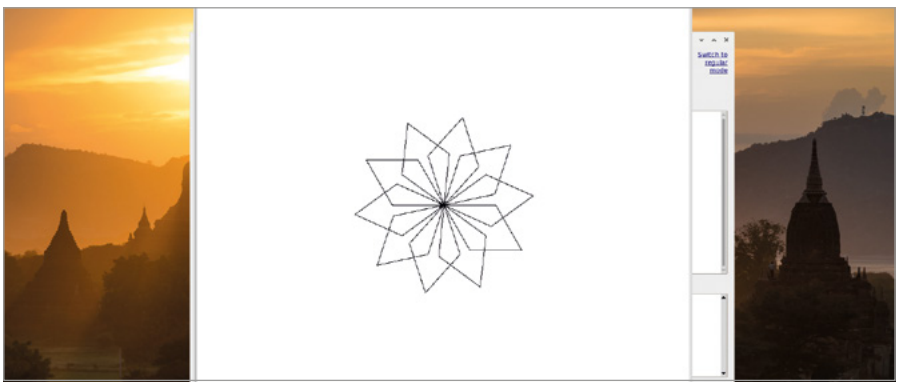
… y añade lo siguiente en la parte inferior de tu programa:

```
    pat.right(36)
```

El programa no se ejecutará tal como está, porque el bucle existente no está sangrado correctamente. Para corregirlo, haz clic al principio de cada línea en el bucle existente (líneas de la 4 a la 8) y pulsa la tecla **ESPACIO** cuatro veces para corregir la sangría. Ahora el programa debería verse así:

```
import turtle
pat = turtle.Turtle()
for i in range(10):
    for i in range(2):
        pat.forward(100)
        pat.right(60)
        pat.forward(100)
        pat.right(120)
    pat.right(36)
```

Haz clic en el icono Run y observa la tortuga: dibujará un paralelogramo, como antes, pero al terminar girará 36 grados y seguirá dibujando hasta que haya diez paralelogramos superpuestos en la pantalla, que formarán una especie de copo de nieve (**Figura 5-10**).



▲ **Figura 5-10:** Repetir la forma para hacer una más compleja

Mientras que una tortuga robótica dibuja en un solo color en un trozo de papel grande, la tortuga simulada de Python puede utilizar una gama de colores. Añade nuevas líneas 3 y 4, empujando hacia abajo las líneas existentes:

```
turtle.Screen().bgcolor("blue")
pat.color("cyan")
```

Ejecuta tu programa de nuevo y verás el efecto del nuevo código: el color de fondo de la ventana de Turtle Graphics ha cambiado a azul, y el copo de nieve es de color azul verdoso o cian (**Figura 5-11**).

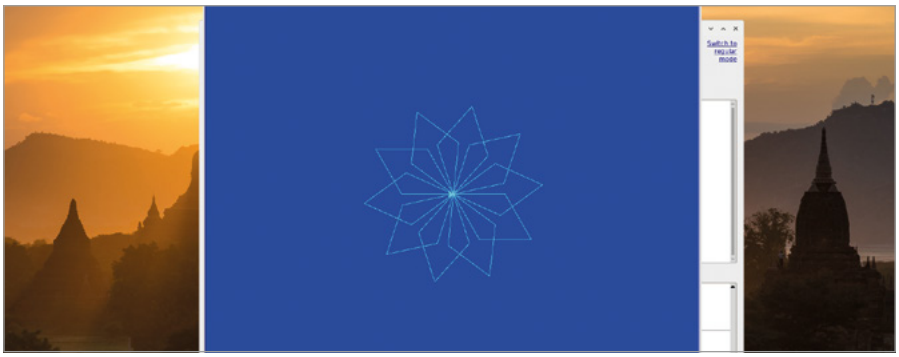


Figura 5-11: Cambiar el fondo y los colores del copo de nieve

También puedes usar colores elegidos al azar en una lista, usando la biblioteca `random`. Vuelve a la parte superior del programa e inserta esto como línea 2:

```
import random
```

Cambia el color de fondo en lo que ahora es la línea 4 de 'azul' a 'gris', luego crea una nueva variable llamada "colores" insertando una nueva línea 5:

```
colours = ["cyan", "purple", "white", "blue"]
```

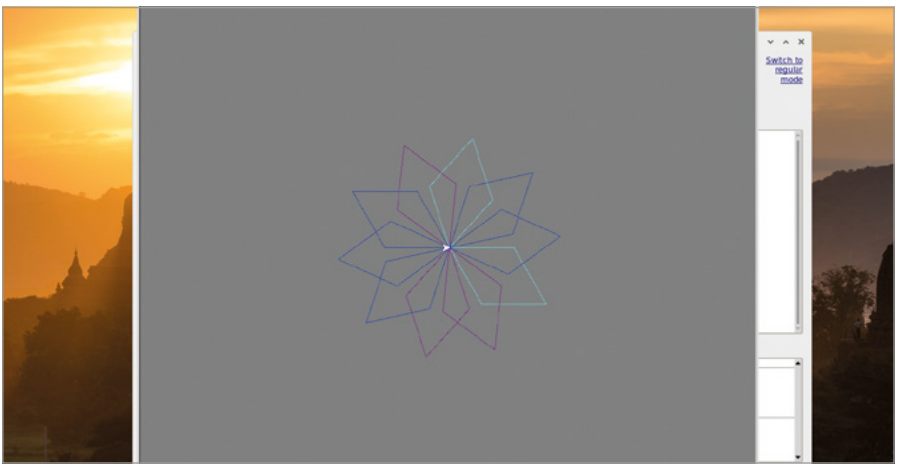
ORTOGRAFÍA DE ESTADOS UNIDOS

Muchos lenguajes de programación usan la ortografía del inglés americano, Python entre ellos: el comando para cambiar el color del lápiz de la tortuga se escribe *color* y no funcionará si lo escribes como en inglés británico como *colour*. Pero las variables se pueden escribir indistintamente: si alguna recibe el nombre *colours*, Python lo entenderá.

Este tipo de variable se denomina lista y está delimitado por corchetes. En este caso, la lista contiene posibles colores para los segmentos de copos de nieve, pero aún así hay que decirle a Python que elija uno cada vez que se repita el bucle. Al final del programa, introduce lo siguiente, asegurándote de que tenga una sangría de cuatro espacios para que forme parte del bucle exterior, al igual que la línea encima de él:

```
    pat.color(random.choice(colours))
```

Haz clic en el icono Run y se volverá a dibujar el copo-estrella-ninja. Pero esta vez Python elegirá un color al azar en tu lista mientras dibuja cada pétalo, dándole al copo de nieve un bonito acabado multicolor (**Figura 5-12**).



▲ **Figura 5-12:** Usar colores aleatorios para los "pétalos"

Para que el copo de nieve se parezca menos a una estrella ninja y más a un copo de nieve real, añade una nueva línea 6, directamente debajo de tu lista **`colours`** y escribe:

```
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
```

Las instrucciones **`penup`** y **`pendown`** moverían un bolígrafo real sobre un papel y lo levantarían si se utilizara un robot tortuga, pero en el mundo virtual simplemente le dicen a la tortuga respectivamente que se detenga y que empiece a dibujar líneas. Pero esta vez en lugar de usar un bucle, vas a crear una *función*, que es un segmento de código que puedes utilizar en cualquier momento, es como crear tu propia instrucción de Python.

Para empezar, borra el código para dibujar los copos mediante paralelogramos, o sea: todo lo que hay entre la instrucción **`pat.color("cyan")`** de la línea 10 y **`pat.right(36)`** en la línea 17, ambas inclusive. Deja la instrucción **`pat.color(random.choice(colours))`** pero añádele un símbolo de almohadilla (**#**) al principio de la línea. Esto se conoce como *marcar con comentarios* una instrucción y significa que Python la pasará por alto. Puedes usar comentarios para añadir explicaciones a tu código, lo que hará que sea mucho más fácil de entender cuando vuelvas a él unos meses después o se lo envíes a otra persona.

Crea tu función, a la que llamarás 'branch' (refiriéndote a las puntas del copo de nieve), escribiendo la siguiente instrucción en la línea 10, debajo de **`pat.pendown():`**

```
def branch():
```

Esto *define* tu función, **`branch`**. Al pulsar la tecla **ENTRAR**, Thonny añadirá automáticamente una sangría a las instrucciones de la función. Escribe lo siguiente, asegurándote de prestar mucha atención a la sangría, porque vas a tener que anidar código con tres niveles de sangría.

```
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
```

Por último, crea un nuevo bucle en la parte inferior del programa —pero encima de la línea de color con comentario— para ejecutar, o *llamar* a tu nueva función:

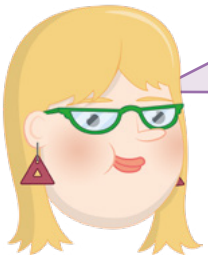
```
for i in range(8):
    branch()
    pat.left(45)
```

Tu programa final debería tener este aspecto:

```
import turtle
import random
pat = turtle.Turtle()
turtle.Screen().bgcolor("grey")
colours = ["cyan", "purple", "white", "blue"]
pat.penup()
pat.forward(90)
pat.left(45)
pat.pendown()
def branch():
    for i in range(3):
        for i in range(3):
            pat.forward(30)
            pat.backward(30)
            pat.right(45)
        pat.left(90)
        pat.backward(30)
        pat.left(45)
    pat.right(90)
    pat.forward(90)
for i in range(8):
    branch()
    pat.left(45)
#    pat.color(random.choice(colours))
```

Haz clic en Run y observa la ventana de gráficos mientras Pat dibuja siguiendo tus instrucciones. ¡Enhorabuena! Ahora tu copo de nieve se parece mucho más a un copo de nieve de verdad (**Figura 5-13**).

Figura 5-13: Las puntas adicionales hacen que parezca un copo de nieve



RETO: ¿QUÉ MÁS?

¿Puedes usar tus instrucciones con comentarios para que las puntas del copo de nieve se dibujen en distintos colores? ¿Puedes crear una función 'snowflake' y usarla para dibujar muchos copos de nieve en la pantalla? ¿Puedes hacer que tu programa cambie aleatoriamente el tamaño y el color de los copos de nieve?

Proyecto 2: Diferencias espantosas

Además de gráficos de tortuga, Python también puede manejar imágenes y sonidos. Y con ellos podrás gastar una broma a tus amigos si creas un juego de "diferencias" que esconde un secreto aterrador, ¡perfecto para Halloween!

PROYECTO ONLINE

Este proyecto también está disponible online en **rpf.io/scary-spot**

Este proyecto necesita dos imágenes —la de las diferencias y una imagen sorpresa "siniestra"— y un archivo de sonido. Haz clic en el icono de Raspberry para cargar el menú del sistema operativo Raspberry Pi, elige la categoría Internet y haz clic en la opción del navegador web Chromium. Cuando se cargue, escribe **rpf.io/spot-pic** en la barra de direcciones y pulsa la tecla **ENTRAR**. Haz clic con el botón derecho del ratón en la imagen y luego haz clic en "Guardar imagen como...", elige la carpeta Home en la lista de la izquierda y haz clic en el botón Guardar. Haz clic de nuevo en la barra de direcciones de Chromium, y luego escribe **rpf.io/scary-pic** y pulsa la tecla **ENTRAR**. Como has hecho antes, haz clic con el botón derecho del ratón en la imagen y luego haz clic en "Guardar imagen como...", elige la carpeta Home en la lista de la izquierda y haz clic en el botón Guardar.

Para el archivo de sonido que necesitas, haz clic de nuevo en la barra de direcciones y escribe **rpf.io/scream** y pulsa la tecla **ENTRAR**. Este archivo, el sonido de un grito para asustar al jugador, se reproducirá automáticamente, pero tendrás que guardarlo para poder usarlo. Haz clic con el botón derecho del ratón en el pequeño reproductor de audio, haz clic en "Guardar como...", elige la carpeta Home y haz clic en Guardar. Ahora puedes cerrar la ventana de Chromium.

Haz clic en el icono New de la barra de herramientas de Thonny para empezar un proyecto nuevo. Como antes, para ampliar las capacidades de Python vas a utilizar una biblioteca, la de Pygame, que está pensada especialmente para los juegos. Escribe lo siguiente:

```
import pygame
```

También necesitarás partes de otras bibliotecas y una subsección de la biblioteca de Pygame. Importa esos elementos escribiendo:

```
from pygame.locals import *
from time import sleep
from random import randrange
```

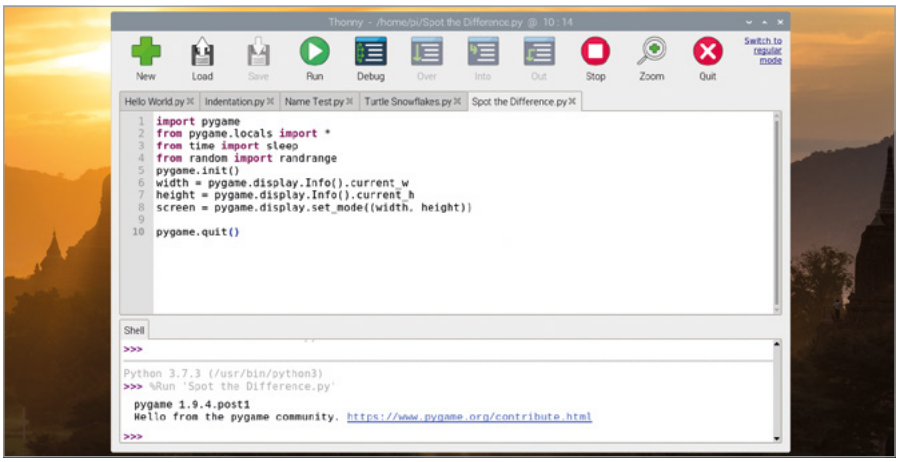
La instrucción **from** es distinta de la instrucción **import** y te permite importar únicamente ciertas secciones de la biblioteca, en lugar de importarla entera. A continuación, tienes que configurar Pygame, lo que se conoce como *inicialización*. Pygame necesita los datos de ancho y alto (*resolución*) del monitor o TV del jugador. Escribe lo siguiente:

```
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
```

El último paso para configurar Pygame es crear su ventana, que Pygame denomina pantalla. Escribe lo siguiente:

```
screen = pygame.display.set_mode((width, height))

pygame.quit()
```

Fíjate en la línea en blanco del medio, ahí es donde irá tu programa. De momento, haz clic en el icono Run, guarda tu programa como **Spot the Difference** y observa: Pygame creará una ventana y la llenará con un fondo negro que desaparecerá casi inmediatamente cuando llegue a la instrucción de salir. Aparte de mostrar un mensaje corto en el shell (**Figura 5-14**), hasta ahora el programa no hace gran cosa.



▲ **Figura 5-14:** Tu programa funciona, pero aún no hace gran cosa

Para mostrar la imagen que contendrá las diferencias, escribe la siguiente línea en el espacio encima de **`pygame.quit()`**:

```
difference = pygame.image.load('spot_the_diff.png')
```

Para asegurarte de que la imagen llena la pantalla, tendrás que modificar valores para ajustarla a la resolución del monitor o el televisor. Escribe lo siguiente:

```
difference = pygame.transform.scale(difference, (width, height))
```

Ahora que la imagen está en la memoria, tienes que decirle a Pygame que la muestre en la pantalla, este es el proceso conocido como *blitting* o *transferencia de bloques de bits*. Escribe lo siguiente:

```
screen.blit(difference, (0, 0))
pygame.display.update()
```

La primera de estas líneas copia la imagen en la pantalla, empezando por la esquina superior izquierda; la segunda le dice a Pygame que redibuje la pantalla. Sin esta segunda línea, aunque la imagen esté en el lugar correcto en la memoria, nunca la verás.

Haz clic en el icono Run y verás la imagen aparecer en pantalla un instante (**Figura 5-15**).

▲ **Figura 5-15:** Tu imagen de diferencias

Para ver la imagen durante más tiempo, añade la siguiente línea justo encima de **`pygame.quit()`**:

```
sleep(3)
```

Vuelve a hacer clic en Run y la imagen permanecerá más tiempo en pantalla. Añade tu imagen sorpresa escribiendo lo siguiente justo debajo de la línea **`pygame.display.update():`**

```
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale(zombie, (width, height))
```

Añade un retardo, para que la imagen del zombi no aparezca inmediatamente:

```
sleep(3)
```

Luego transfiere (blitting) la imagen a la pantalla y actualízala para que la vea el jugador:

```
screen.blit(zombie, (0,0))
pygame.display.update()
```

Haz clic en el icono Run y observa qué ocurre: Pygame cargará tu imagen de diferencias, pero al cabo de tres segundos se sustituirá por la del horrendo zombi (**Figura 5-16**).

Figura 5-16: Para asustar a cualquiera

Pero con el retardo de tres segundos, la cosa es algo predecible. Cambia la línea **`sleep(3)`** encima de **`screen.blit(zombie, (0,0))`** a:

```
sleep(randrange(5, 15))
```

Esto elige al azar un número entre 5 y 15 y aplica al programa un retardo equivalente. A continuación, añade la siguiente línea justo encima de la instrucción **`sleep`**, para cargar el archivo de sonido del grito:

```
scream = pygame.mixer.Sound('scream.wav')
```

Colócate debajo de la instrucción **`sleep`** y escribe lo siguiente en una línea nueva, para que se active justo antes de que el zombi aparezca en pantalla:

```
scream.play()
```

Por último, dile a Pygame que deje de reproducir el sonido escribiendo la siguiente línea justo encima de **pygame.quit()**:

```
scream.stop()
```

Haz clic en el icono Run y admira tu obra: tras unos segundos de inocente diversión buscando las diferencias, aparecerá el zombi y se oirá un grito espeluznante que dará un buen susto a tus amigos. Si la imagen del zombi aparece antes de iniciarse el sonido, puedes compensarlo añadiendo un breve retardo justo después de la instrucción **scream.play()** y antes de **screen.blit**:

```
sleep(0.4)
```

Tu programa final debería tener este aspecto:

```
import pygame
from pygame.locals import *
from time import sleep
from random import randrange
pygame.init()
width = pygame.display.Info().current_w
height = pygame.display.Info().current_h
screen = pygame.display.set_mode((width, height))
difference = pygame.image.load('spot_the_diff.png')
difference = pygame.transform.scale(difference, (width, height))
screen.blit(difference, (0, 0))
pygame.display.update()
zombie = pygame.image.load('scary_face.png')
zombie = pygame.transform.scale (zombie, (width, height))
scream = pygame.mixer.Sound('scream.wav')
sleep(randrange(5, 15))
scream.play()
screen.blit(zombie, (0,0))
pygame.display.update()
sleep(3)
scream.stop()
pygame.quit()
```

Ahora lo único que tienes que hacer es invitar a tus amigos a jugar a buscar las diferencias... con los altavoces encendidos, por supuesto.

RETO: MODIFICACIONES

¿Puedes cambiar las imágenes para adaptar la broma a otros eventos, por ejemplo, la Navidad? ¿Puedes dibujar tus propias imágenes, la de diferencias y la sorpresa, (usando un editor de gráficos como GIMP)? ¿Puedes rastrear el clic del usuario en una diferencia, para hacerlo más convincente?

Proyecto 3: Laberinto RPG

Ahora que ya le has cogido el tranquillo a Python, es hora de usar Pygame para hacer algo un poco más complicado: un juego de laberinto basado en texto totalmente funcional, inspirado en los clásicos juegos de rol. Conocidos como aventuras de texto o ficción interactiva, estos juegos se remontan la época en que los ordenadores no podían gestionar gráficos, pero siguen teniendo aceptación entre quienes argumentan que no hay gráficos más exactos que los de nuestra propia imaginación.

PROYECTO ONLINE

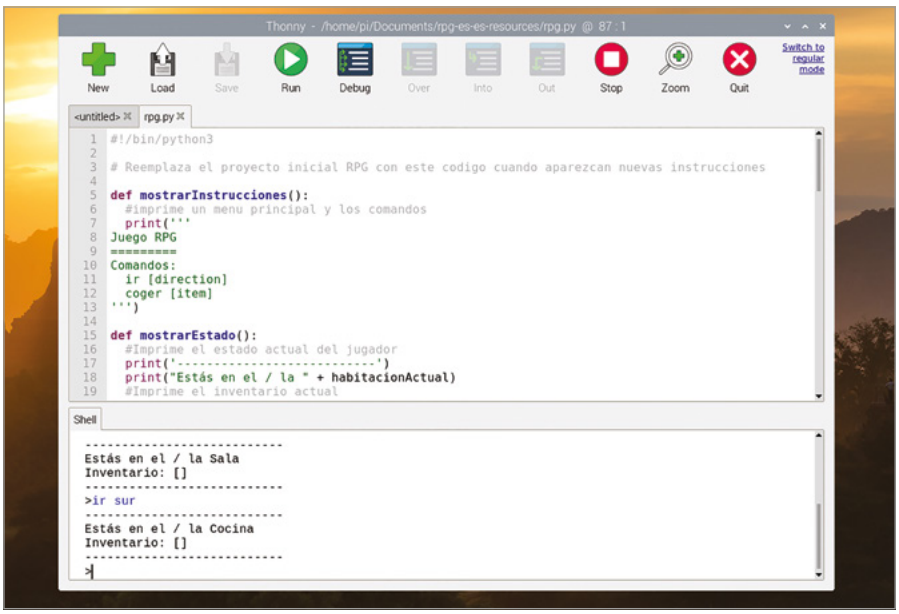
Este proyecto también está disponible online en **rpf.io/python-rpg**

Este programa es bastante más complejo que los otros de este capítulo, así que para facilitarte las cosas, empezarás con una versión parcialmente escrita. Abre el navegador web Chromium y ve a la dirección: **rpf.io/p/es-ES/rpg-go**.

El navegador web Chromium descargará automáticamente el código del programa a tu carpeta de descargas, pero te advertirá que el tipo de archivo (un programa en Python) podría dañar tu ordenador. Dado que has descargado el archivo desde la Fundación Raspberry Pi, que es una fuente de confianza, puedes hacer clic en el botón Keep del mensaje de advertencia que aparece en la parte inferior de la pantalla. Vuelve a Thonny y luego haz clic en el icono Load . Localiza el archivo **rpg-rpg.py** en tu carpeta de descargas y haz clic en el botón Load.

Para empezar, haz clic en el icono Run para familiarizarte con el funcionamiento de una aventura de texto. El resultado del juego aparece en el área de shell, en la parte inferior de la ventana de Thonny. Agranda la ventana de Thonny haciendo clic en el botón de maximizar para que sea más fácil de leer.

Tal y como está ahora, el juego es muy simple: hay dos habitaciones y ningún objeto. El jugador comienza en Sala, la primera de las dos habitaciones. Para ir a la Cocina, simplemente escribe 'ir sur' y pulsa la tecla **ENTRAR** (**Figura 5-17**). Cuando estés en la Cocina, puedes escribir 'ir norte' para volver a la entrada. También puedes escribir 'ir oeste' e 'ir este', pero como no hay habitaciones en esas direcciones el juego mostrará un mensaje de error.



▲ **Figura 5-17:** De momento solo hay dos habitaciones

Baja hasta la línea 29 del programa en el área de script y localiza una variable denominada **`rooms`** (habitaciones). Este tipo de variable se conoce como *diccionario* e informa al juego de las habitaciones, sus salidas y a qué habitación conduce una salida determinada.

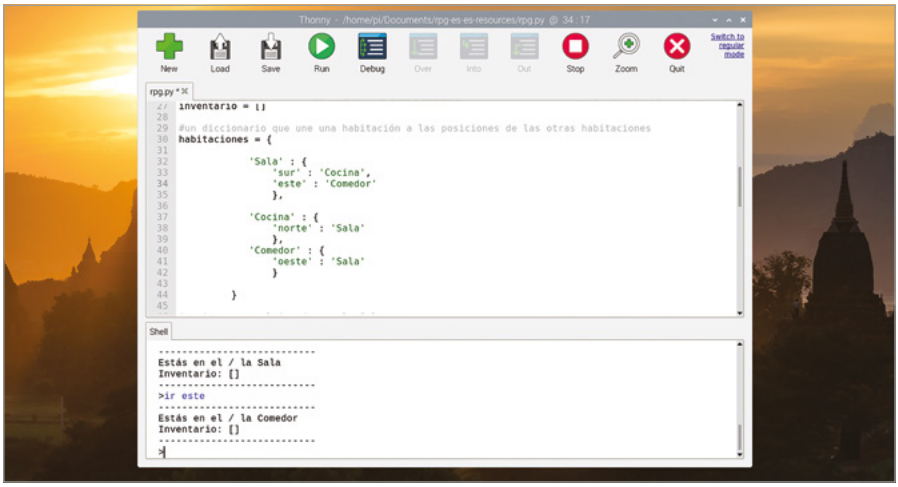
Para hacer el juego más interesante, añade otra habitación: un Comedor, al este de la entrada. Localiza la variable **`rooms`** en el área de scripts y amplíala añadiendo una coma (**`,`**) después de **`}`** en la línea 38. Luego escribe lo siguiente (en un diccionario no es esencial usar la sangría exacta en cada caso):

```
'Comedor' : {
      'oeste' : 'Sala'
   }
```

También necesitarás una nueva puerta en la entrada, porque no se crea una automáticamente. Ve al final de la línea 33, añade una coma y luego la siguiente línea:

```
'este' : 'Comedor'
```

Haz clic en el icono Run y prueba la nueva habitación: escribe 'ir este' en la entrada para pasar al Comedor (**Figura 5-18**, a continuación) y escribe 'ir este' en el comedor para pasar a la Sala. ¡Enhorabuena! Has creado una habitación.



▲ **Figura 5-18:** Has añadido otra habitación

Pero no hay mucho de interés en las habitaciones vacías. Para añadir un elemento a una habitación, tendrás que modificar el diccionario de esa habitación. Detén el programa haciendo clic en el icono Stop. Localiza el diccionario **`Sala`** en el área de scripts y añade una coma al final de la línea **`'este' : 'Comedor'`** antes de pulsar **ENTRAR** y escribir la línea siguiente:

```
'objeto' : 'llave'
```

Vuelve a hacer clic en Run. Esta vez, el juego te dirá que puedes ver tu nuevo objeto: una llave. Escribe 'coger llave' (**Figura 5-19**) para poder cogerla, añadiéndola a la lista de elementos que llevas contigo: tu *inventario*. El inventario va contigo de una habitación a otra.

▲ **Figura 5-19:** La llave se añade al inventario

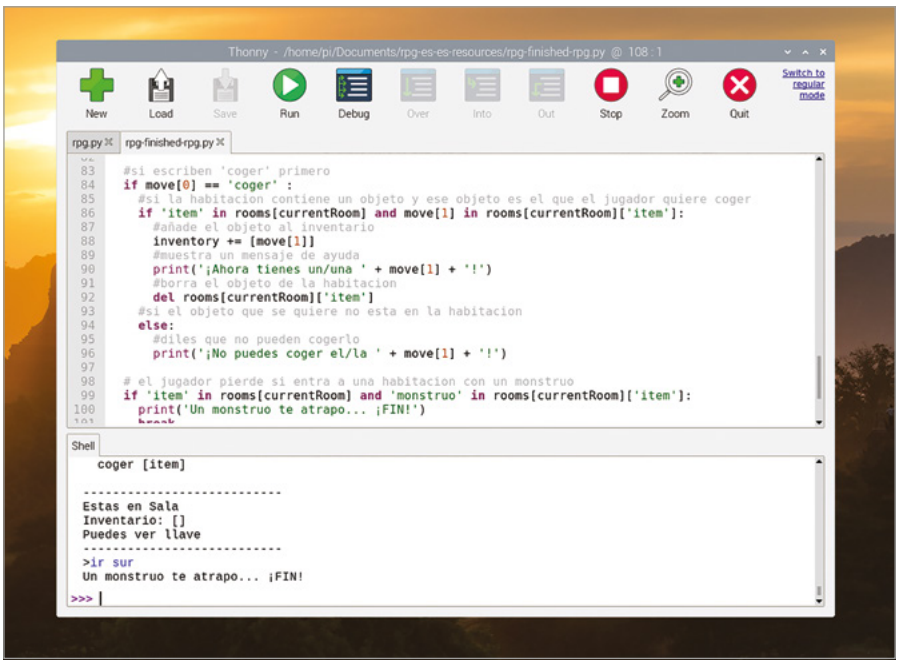
Haz clic en el icono Stop y añade interés al juego añadiendo un monstruo que los jugadores tendrán que evitar. Localiza el diccionario **Cocina** y añade un elemento 'monstruo', igual que has añadido el objeto 'llave'. Acuérdate de poner una coma al final de la línea sobre:

```
'objeto' : 'monstruo'
```

Para que el monstruo pueda atacar al jugador, tendrás que añadir algo de lógica al juego. Desplázate hasta el fondo del programa en el área de script y añade las siguientes líneas, incluyendo el comentario, marcado con un símbolo de almohadilla (indicando que el jugador pierde si entran en la habitación en la que está el monstruo) que te ayudará a entender el programa cuando lo reanudes en otro momento. Asegúrate de sangrar las líneas:

```
    # el jugador pierde si entra en una habitación con un
monstruo
    if 'objeto' in habitaciones[habitacionActual] and 'monstruo'
in habitaciones[habitacionActual]['objeto']:
        print('Te ha pillado el monstruo... JUEGO TERMINADO!')
        break
```

Haz clic en Run y ve a la cocina (**Figura 5-20**). Al monstruo no le va a gustar.



▲ **Figura 5-20:** Las ratas son lo de menos, hay un monstruo en la cocina

Para convertir esta aventura en un juego en toda regla, vas a necesitar más objetos, otra habitación y la posibilidad de "ganar" saliendo de la casa con todos los objetos a salvo en el inventario. Para empezar, añade otra habitación, como hiciste con el comedor. Esta vez vas a añadir un Jardín. Añade una salida del diccionario Comedor y acuérdate de añadir una coma al final de la línea de arriba:

```
        'sur' : 'Jardin'
```

Luego añade la habitación nueva al diccionario **habitaciones** principal y añade la coma después de **}** en la línea de arriba, como has hecho antes:

```
    'Jardin' : {
          'norte' : 'Comedor'
        }
```

Añade un objeto 'poción' al diccionario Dining Room, con la coma necesaria añadida a la línea de arriba:

```
        'objeto' : 'pocion'
```

Por último, ve hasta el final del programa y añade la lógica necesaria para comprobar si el jugador tiene todos los objetos y, de ser así, añade un comentario que diga que ha ganado el juego:

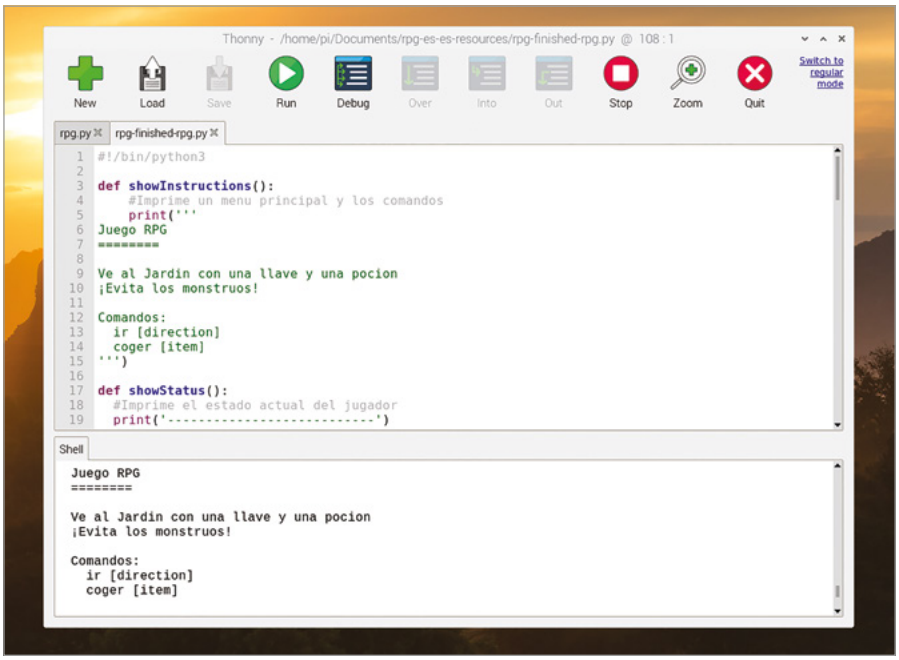
```
   # el jugador gana si llega al jardín con una llave y una poción
      if habitacionActual == 'Jardin' and 'llave' in inventario and
'pocion' in inventario:
          print('Has escapado de la casa... HAS GANADO!')
          break
```

Haz clic en Run e intenta completar el juego cogiendo la llave y la poción antes de ir al jardín. Evita entrar a la cocina, recuerda que ahí está el monstruo.

Como último retoque, añade instrucciones para indicar al jugador cómo completar el juego. Ve a la parte superior del programa, donde se define la función **mostrarInstrucciones()** y añade:

```
Llega la jardín con una llave y una poción.
¡Evita a los monstruos!
```

Ejecuta el juego por última vez y verás que tus nuevas instrucciones aparecen al principio (**Figura 5-21**). ¡Enhorabuena! Has creado un juego de laberinto interactivo basado en texto.



▲ **Figura 5-21:** Ahora el jugador sabe qué tiene que hacer

RETO: EXPANDIR EL JUEGO

¿Puedes añadir más habitaciones para que el juego dure más? ¿Puedes añadir un objeto como protección contra el monstruo? ¿Cómo añadirías un arma para acabar con el monstruo? ¿Puedes añadir habitaciones encima y debajo de las existentes, a las que se acceda por escaleras?

Capítulo 6

Informática física con Scratch y Python

La codificación puede incluir otras cosas, aparte de lo que hagas en la pantalla: también puedes controlar los componentes electrónicos conectados a los pines GPIO de tu Raspberry Pi

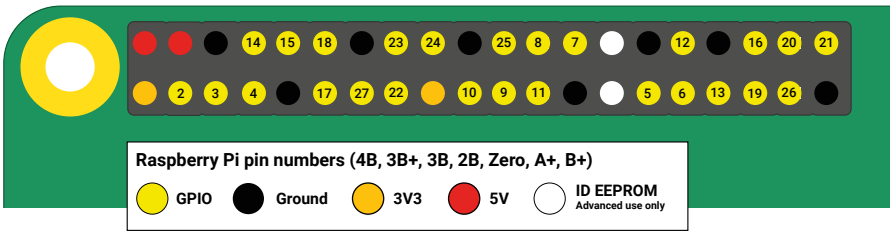
Cuando pensamos en algo como "programar" o "codificar", normalmente lo asociamos a software. Sin embargo, la codificación puede abarcar también el mundo real a través del hardware. Esto se conoce como *informática física*.

Como el nombre sugiere, la informática física está relacionada con el control de las cosas en el mundo real a través de tus programas: hardware, en lugar de software. Cuando elegimos el programa de la lavadora, cambiamos la temperatura en el termostato de casa o pulsamos un botón para cruzar en un semáforo, estamos utilizando informática física.

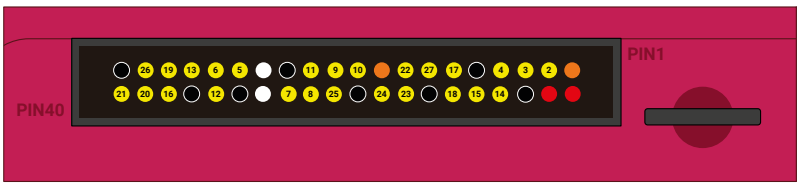
Raspberry Pi es un dispositivo excelente para aprender sobre informática física gracias a un componente: el sistema de pines de *entrada/salida de uso general* o *GPIO*.

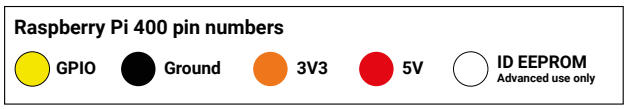
El sistema GPIO

GPIO se encuentra en el borde superior de la placa de circuito de Raspberry Pi, o en la parte posterior de Raspberry Pi 400, y se compone de dos largas filas de pines metálicos. Es lo que te permitirá conectar piezas de hardware, como LED y conmutadores, a Raspberry Pi para controlar los programas que crees. Los pines pueden utilizarse tanto para la entrada como para la salida.



El sistema GPIO de Raspberry Pi se compone de 40 pines macho. Algunos pines están disponibles para proyectos de informática física, otros suministran energía y otros se reservan para la comunicación con hardware adicional, por ejemplo con la placa Sense HAT (consulta el **Capítulo 7**).





Raspberry Pi 400 tiene el mismo sistema GPIO, con los mismos pines, pero está "boca abajo", con relación a otros modelos de Raspberry Pi. Este diagrama presupone que estás viendo el sistema GPIO desde la parte posterior de Raspberry Pi 400. Siempre debes verificar los cables al conectar cualquier cosa al sistema GPIO de Raspberry Pi 400, porque es fácil despistarse, a pesar de las etiquetas "Pin 40" y "Pin 1" de la carcasa.

EXTENSIONES DE GPIO

Aunque el sistema GPIO de Raspberry Pi 400 se puede usar tal cual, puede resultar más fácil usar una extensión. Con una extensión, los pines pasan al lateral de Raspberry Pi 400 y así puedes comprobar y ajustar el cableado sin tener que acceder a la parte posterior.

Entre las extensiones compatibles están la gama Black HAT Hack3r de **pimoroni.com** y Pi T-Cobbler Plus de **adafruit.com**.

Si compras una extensión, debes comprobar siempre su cableado, porque en algunas (por ejemplo, Pi T-Cobbler Plus) la distribución de los pines GPIO es distinta. En caso de duda, sigue siempre las instrucciones del fabricante.

Hay varias categorías de tipos de pines, cada una de las cuales tiene una función concreta:

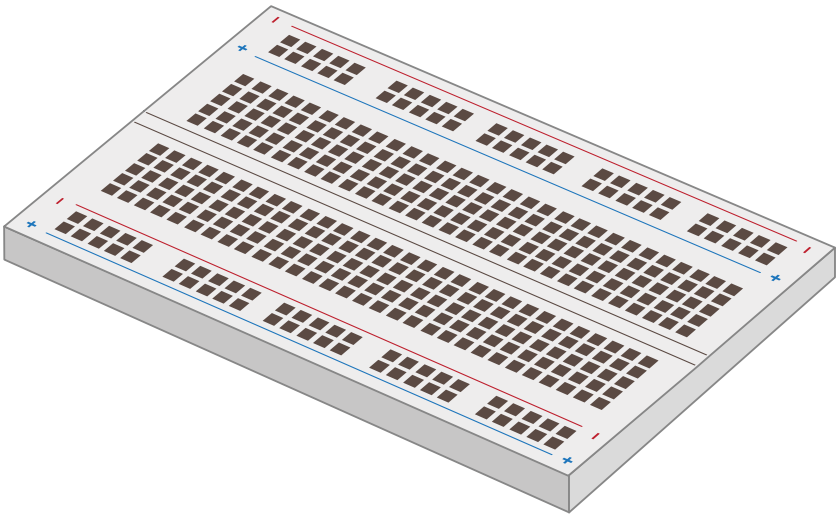
3V3	**3,3 voltios de potencia**	Una fuente de energía de 3,3 V encendida permanentemente, con el mismo voltaje que el de Raspberry Pi internamente
5V	**5 voltios de potencia**	Una fuente de energía de 5 V encendida permanentemente, con el mismo voltaje que Raspberry Pi toma del conector de alimentación micro USB
Ground (GND)	**0 voltios de tierra**	Una conexión a tierra, usada para completar un circuito conectado a la fuente de alimentación
GPIO XX	**Pin 'XX' de entrada/salida de propósito general**	Los pines GPIO disponibles para tus programas, identificados por un número del 2 al 27
ID EEPROM	**Pines reservados para fines especiales**	Los pines se reservan para usarlos con la placa HAT y otros accesorios

¡ADVERTENCIA!

El sistema GPIO de Raspberry Pi ofrece una forma divertida y segura de experimentar con la informática física, pero debes utilizarlo con precaución. Ten cuidado de no doblar los pines al conectar y desconectar el hardware. Nunca conectes dos pines directamente juntos, ni voluntaria ni involuntariamente, a menos que se indique expresamente en las instrucciones de un proyecto: el resultado producido se denomina cortocircuito y, dependiendo de los pines, puede dañar permanentemente tu Raspberry Pi.

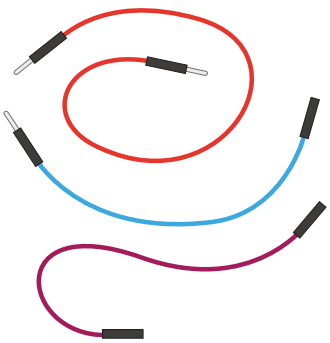
Componentes electrónicos

El sistema GPIO es solo una parte de lo que necesitarás para empezar a trabajar con la informática física; la otra mitad la integran los componentes electrónicos, los dispositivos que controlarás con el sistema GPIO. Hay miles de componentes diferentes disponibles, pero la mayoría de los proyectos de GPIO se realizan utilizando estos de uso común:

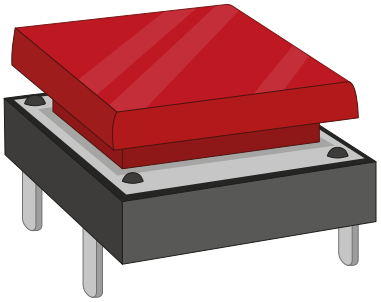


Una *placa de pruebas*, también denominada *placa de inserción*, puede facilitar considerablemente los proyectos de informática física. En lugar de tener un montón de componentes individuales que hace falta conectar con cables, una placa de pruebas permite insertar componentes y conectarlos a través de líneas metálicas ocultas bajo su superficie. Muchas placas de pruebas también incluyen secciones para la distribución de energía y te facilitan la construcción de circuitos. Aunque no sea esencial para empezar con la informática física, una placa de pruebas es muy útil.

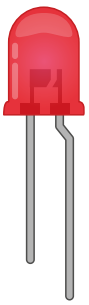
Los cables puente o simplemente *puentes* conectan los componentes a Raspberry Pi y entre sí (en caso de no usar una placa de pruebas). Están disponibles en tres versiones: macho-hembra (M2F), necesarios para conectar una placa de pruebas a los pines GPIO; hembra-hembra (F2F), que se pueden utilizar para conectar componentes individuales entre sí en caso de no utilizar una placa de pruebas; y macho-macho (M2M), para establecer conexiones entre partes de una placa de pruebas. Dependiendo de tu proyecto, podrías necesitar los tres tipos de cables puente. Si usas una placa de pruebas, suele bastar con usar puentes M2F y M2M.



Un *interruptor* o *conmutador momentáneo* es el tipo de interruptor que se usa para controlar una consola de juego. Comúnmente disponible con dos o cuatro patas (ambas opciones funcionarán con Raspberry Pi), es un dispositivo de entrada: puedes decirle a tu programa que detecte cuándo se pulsa para luego realizar una tarea. Otro tipo de interruptor común es el *telerruptor o interruptor de paso*. Un conmutador momentáneo solo está activo mientras lo mantienes pulsado, pero un interruptor de paso (similar a los interruptores de luz comunes) se mantiene activo hasta que vuelves a accionarlo para desactivarlo.

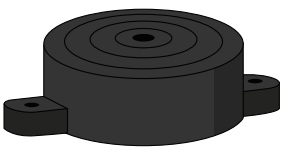


Un *diodo emisor de luz* (*LED*) es un *dispositivo de salida* que controlas directamente desde tu programa. Un LED está iluminado cuando está activo. Los hay por todas partes en nuestras casas: desde los pequeños que avisan cuando la lavadora sigue encendida, hasta los grandes que pueden iluminar nuestras habitaciones. Los LED están disponibles en una amplia gama de formas, colores y tamaños, pero no todos son adecuados para usar con Raspberry Pi: evita los que están diseñados para fuentes de alimentación de 5 V o 12 V.



Las resistencias son componentes que controlan el flujo de *corriente eléctrica* y están disponibles en diferentes valores medidos con una unidad llamada *ohmio* (*Ω*). Cuanto mayor sea el número de ohmios, más resistencia ofrecen. En los proyectos de informática física de Raspberry Pi, su finalidad más común es proteger a los LED para que no carguen demasiada corriente y se dañen o dañen el Raspberry Pi. Por esa razón se deberían usar resistencias de unos 330 Ω, aunque muchos proveedores de componentes electrónicos venden prácticos paquetes que contienen distintos valores de uso común, para más flexibilidad.

Un *zumbador piezoeléctrico*, o simplemente zumbador, es otro dispositivo de salida. Mientras que un LED produce luz, un zumbador produce un ruido, o más concretamente, un zumbido. Dentro de la carcasa de plástico del zumbador hay un par de placas de metal que, cuando están activas, vibran una contra la otra para producir el zumbido. Hay dos tipos de zumbadores: *activos* y *pasivos*. Asegúrate de tener uno activo, ya que son los más simples de usar.



Otros componentes electrónicos comunes son los motores, que necesitan una placa de pruebas de control especial antes de poder conectarse a Raspberry Pi; los sensores infrarrojos que detectan el movimiento; los sensores de temperatura y humedad que pueden utilizarse para predecir el tiempo; y las fotorresistencias (LDR), dispositivos de entrada que funcionan como un LED inverso al detectar luz.

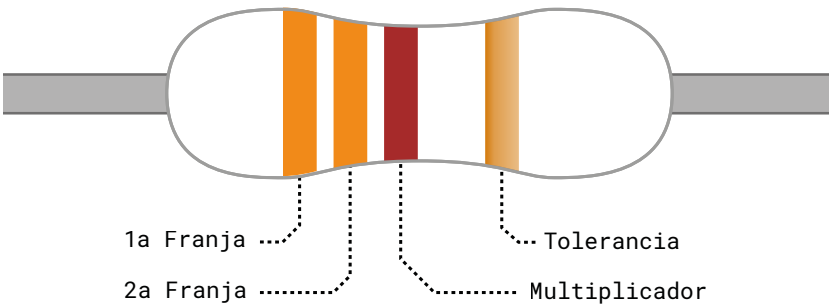
En todo el mundo hay proveedores de componentes para la informática física con Raspberry Pi, ya sea como piezas individuales o en kits que contienen todo lo necesario para empezar. Para encontrar vendedores, visite **rpf.io/products**, haga clic en Raspberry Pi 4 y verá una lista de tiendas online asociadas a Raspberry Pi (approved resellers) de su país o región.

Para completar los proyectos de este capítulo, necesitas como mínimo:

- 3 × LED: rojo, verde y amarillo o ámbar
- 2 × conmutadores momentáneos
- 1 × zumbador activo
- Cables puente macho-hembra (M2F) y hembra-hembra (F2F)
- (Opcional) Una placa de pruebas y cables puente macho-macho (M2M)

Resistencias codificadas con colores

Las resistencias están disponibles en una amplia gama de valores, desde cero (simples cables) a versiones de alta resistencia y gran tamaño. Muy pocas de estas resistencias muestran los valores impresos en números, sino que utilizan un código especial a base de rayas o franjas de color alrededor del cuerpo de la resistencia.



	1a/2a Franja	Multiplicador	Tolerancia
Negro	0	$\times 10^{0}$	-
Brown	1	$\times 10^{1}$	±1%
Rojo	2	$\times 10^{2}$	±2%
Naranja	3	$\times 10^{3}$	-
Amarillo	4	$\times 10^{4}$	-
Verde	5	$\times 10^{5}$	±0.5%
Azul	6	$\times 10^{6}$	±0.25%
Violeta	7	$\times 10^{7}$	±0.1%
Gris	8	$\times 10^{8}$	±0.05%
Blanco	9	$\times 10^{9}$	-
Oro	-	$\times 10^{-1}$	±5%
Plata	-	$\times 10^{-2}$	±10%
Ninguno	-	-	±20%

Para leer el valor de una resistencia, colócala de manera que el grupo de franjas esté a la izquierda y la franja sola a la derecha. Empezando por la primera franja, comprueba ese color en la columna "1ª/2ª franja" de la tabla para obtener el primer dígito y el segundo. Este ejemplo tiene dos franjas anaranjadas, que corresponden a un valor de "3" cada una y hacen un total de "33". Si tu resistencia tiene cuatro franjas agrupadas en lugar de tres, anota el valor de la tercera franja también (para resistencias de cinco/seis franjas, consulta **rpf.io/5-6band**).

Pasa a la última franja del grupo (la tercera o cuarta) y comprueba su color en la columna "Multiplicador". Esto te indica por qué número debes multiplicar tu número actual para obtener el valor real de la resistencia. Este ejemplo tiene una franja marrón, que significa "$\times 10^{1}$". Aunque

parezca confuso, es simplemente una *notación científica*: "×10¹" significa "añade un cero al final de tu número". Si fuera azul, para "×10⁶" significaría "añadir seis ceros al final de tu número".

33, de las franjas anaranjadas, más el cero añadido de la franja marrón nos da 330: ese es el valor de la resistencia, en ohmios. La última franja, a la derecha, es la *tolerancia* de la resistencia. Esto es simplemente la similitud probable con su valor nominal. Las resistencias más baratas pueden tener una franja plateada, indicando que la tolerancia puede ser un 10 % superior o inferior a su calificación. O ninguna franja al final, lo que indica que puede ser un 20 % superior o inferior. Las resistencias más caras tienen una franja gris que indica un margen de error no superior al 0,05 %. Para los proyectos amateur, la precisión no es tan importante: cualquier tolerancia suele funcionar bien.

Si el valor de tu resistencia supera los 1000 ohmios (1000 Ω), normalmente se indica en kilohmios (kΩ). Si supera el millón de ohmios, será en megaohmios (MΩ). Una resistencia de 2200 Ω se escribiría como 2,2 kΩ; una resistencia de 2200000 Ω se escribiría como 2,2 MΩ.

¿LO SABES?

¿De qué color serían las franjas de una resistencia de 100 Ω? ¿De qué color serían las franjas de una resistencia de 2,2 MΩ? Si quieres conseguir las resistencias más baratas, ¿qué color de franja de tolerancia debes buscar?

Tu primer programa de informática física: Hello, LED!

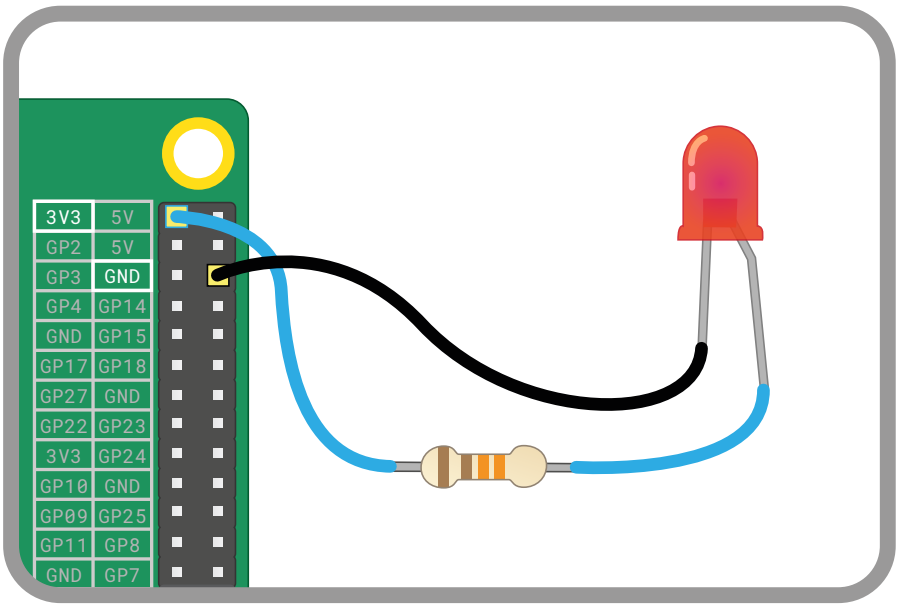
Si mostrar 'Hello, World' en la pantalla es un fantástico primer paso para aprender un lenguaje de programación, hacer que un LED se encienda es el primer paso habitual en el aprendizaje de la informática física. Para este proyecto, necesitarás un LED y una resistencia de 330 ohmios (330 Ω), o tan cerca de 330 Ω como puedas encontrar, además de cables puente hembra-hembra (F2F).

LA RESISTENCIA ES VITAL

La resistencia es un componente vital en este circuito: protege el Raspberry Pi y el LED limitando la cantidad de corriente eléctrica que puede tomar el LED. Sin ella, el LED puede tomar demasiada corriente y fundirse; o se podría fundir el Raspberry Pi. Cuando se usa de esta manera, la resistencia se conoce como *resistencia limitadora de corriente*. El valor exacto de la resistencia que necesitas depende del LED que utilices, pero 330 Ω es un valor adecuado para la mayoría de los LED de uso común. Cuanto más alto sea el valor, más atenuada será la luz del LED. Cuanto más bajo sea el valor, más brillante será el LED.

Nunca conectes un LED a un Raspberry Pi sin una resistencia limitadora de corriente, a menos que sepas que el LED tiene una resistencia incorporada de un valor apropiado.

Empieza por comprobar que tu LED funciona. Gira tu Raspberry Pi para ver el sistema GPIO como dos franjas verticales en la parte derecha. Conecta un extremo de tu resistencia de 330 Ω al primer pin de 3,3 V (identificado como 3V3 en la **Figura 6-1**) usando un puente hembra-hembra. Luego conecta el otro extremo a la pata larga (extremo positivo o ánodo) de tu LED con otro puente hembra-hembra. Usa un último cable puente hembra-hembra y conecta la pata corta (extremo negativo o cátodo) de tu LED al primer pin de tierra (identificado como GND en la **Figura 6-1**).



▲ **Figura 6-1:** Conecta tu LED a estos pines y no te olvides de la resistencia.

Si tu Raspberry Pi está encendido, el LED debería encenderse. De no ser así, comprueba tu circuito: asegúrate de que no has usado un valor de resistencia demasiado alto, que todos los cables están bien conectados y que has elegido los pines GPIO correctos de acuerdo con el diagrama. Comprueba también las patas del LED, ya que los LED solo funcionan en un sentido: con la pata más larga conectada al lado positivo del circuito y la más corta al negativo.

Cuando tu LED funcione, es hora de programarlo. Desconecta el cable puente del pin de 3,3 V (identificado como 3V3 en la **Figura 6-2**) y conéctalo al pin GPIO 25 (identificado como GP25 en la **Figura 6-2**). El LED se apagará, pero eso es normal, no te preocupes.

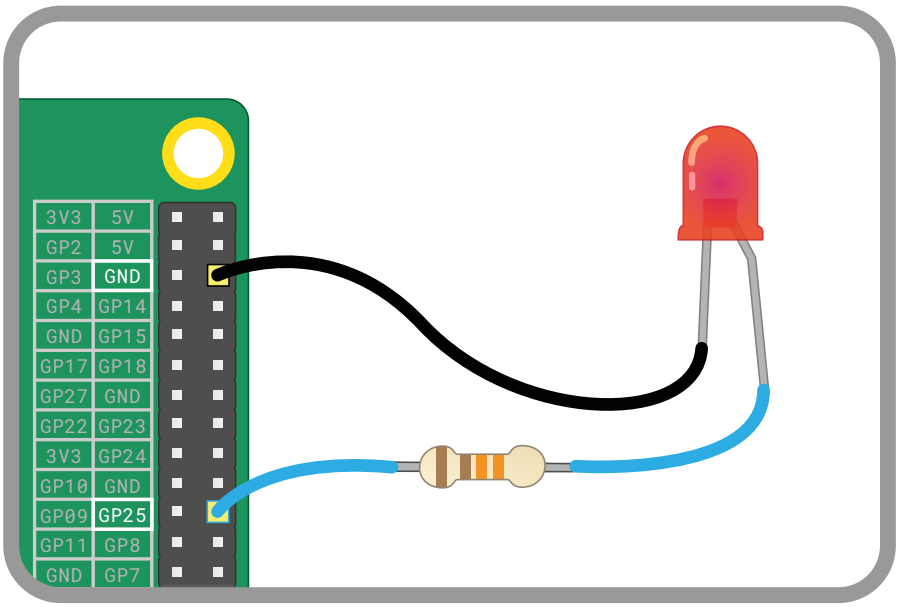


Figura 6-2: Desconecta el cable de 3V3 y conéctalo al pin GPIO 25

Ahora ya puedes empezar a crear un programa con Scratch o Python para encender y apagar tu LED.

CONOCIMIENTOS DE CODIFICACIÓN

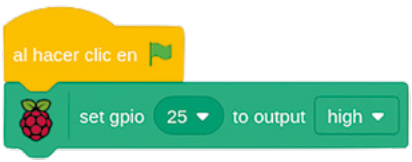
Para realizar los proyectos de este capítulo debes tener algo de experiencia con Scratch 3 y el entorno de desarrollo integrado (IDE) Thonny Python. Si aún no lo has hecho, ve al **Capítulo 4, Programar con Scratch 3** y al **Capítulo 5, Programar con Python** y realiza los proyectos de esos capítulo.

Control de LED en Scratch

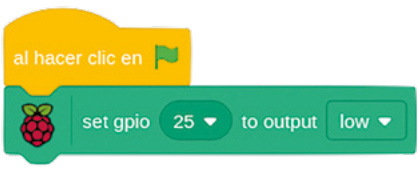
Carga Scratch 3 y haz clic en el icono Añadir extensión . Desplázate hacia abajo para encontrar la extensión GPIO de Raspberry Pi (**Figura 6-3** a continuación) y luego haz clic en ella. Esto carga los bloques necesarios para controlar el sistema GPIO de Raspberry Pi desde Scratch 3. Verás que los nuevos bloques aparecen en la paleta de bloques. Cuando los necesites, estarán disponibles en la categoría GPIO de Raspberry Pi.

Figura 6-3: Añade la extensión GPIO de Raspberry Pi a Scratch 3

Comienza arrastrando un bloque de Eventos al hacer clic en al área de código y luego coloca un bloque set gpio to output high debajo de él. Tendrás que elegir el número del pin que estás usando: haz clic en la flecha para abrir la selección desplegable y haz clic en "25" para decirle a Scratch que estás controlando el pin GPIO 25.

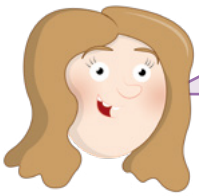


Haz clic en la bandera verde para ejecutar tu programa. Verás encenderse el LED: ¡has programado tu primer proyecto de informática física! Haz clic en el octógono rojo para detener el programa: verás que el LED se mantiene encendido. Eso es porque tu programa solo le ha dicho a Raspberry Pi que encienda el LED, eso es lo que indica "output high" en tu bloque set gpio 25 to output high. Para volver a apagarlo, haz clic en la flecha abajo al final del bloque y elige "low" en la lista.



Vuelve a hacer clic en la bandera verde: ahora tu programa apaga el LED. Para hacer las cosas más interesantes, añade un bloque de control por siempre y un par de bloques esperar 1 segundos para crear un programa que encienda y apague el LED cada segundo.

Haz clic en la bandera verde y observa el LED: se encenderá durante un segundo, se apagará durante un segundo, se encenderá durante un segundo y seguirá repitiendo ese patrón hasta que hagas clic en el octógono rojo para detenerlo. Fíjate en lo que pasa cuando haces clic en el octógono mientras el LED está en estado de encendido o apagado.



RETO: ¿PUEDES MODIFICARLO?

¿Cómo cambiarías el programa para que el LED se mantenga encendido durante más tiempo? ¿Y para que se mantenga apagado durante más tiempo? ¿Cuál es el menor retardo que puedes usar sin dejar de ver cómo se enciende y se apaga el LED?

Control de LED en Python

Carga Thonny desde la sección Programación del menú de Raspberry, luego haz clic en el botón New para iniciar un nuevo proyecto y en Save para guardarlo como **Hello LED**. Para usar los pines GPIO de Python, necesitas una biblioteca llamada GPIO Zero. Para este proyecto, solo se necesita la parte de la biblioteca para trabajar con LED. Importa solo esa sección escribiendo lo siguiente en el área de shell de Python:

```
from gpiozero import LED
```

A continuación, tienes que decirle a GPIO Zero a qué pin GPIO está conectado el LED. Escribe lo siguiente:

```
led = LED(25)
```

Juntas, estas dos líneas permiten que Python controle los LED conectados a los pines GPIO de Raspberry Pi y le dicen qué pin, o pines si tienes más de un LED en tu circuito, debe controlar. Para controlar el LED, escribe lo siguiente:

```
led.on()
```

Para apagar el LED, escribe:

```
led.off()
```

Enhorabuena: ya sabes controlar tus pines GPIO de Raspberry Pi en Python. Vuelve a escribir esas dos instrucciones. Si el LED ya está apagado, **`led.off()`** no hará nada. Lo mismo ocurre si el LED ya está encendido y escribes **`led.on()`**.

Para crear un programa real, escribe lo siguiente en el área de script:

```
from gpiozero import LED
from time import sleep

led = LED(25)

while True:
    led.on()
    sleep(1)
    led.off()
    sleep(1)
```

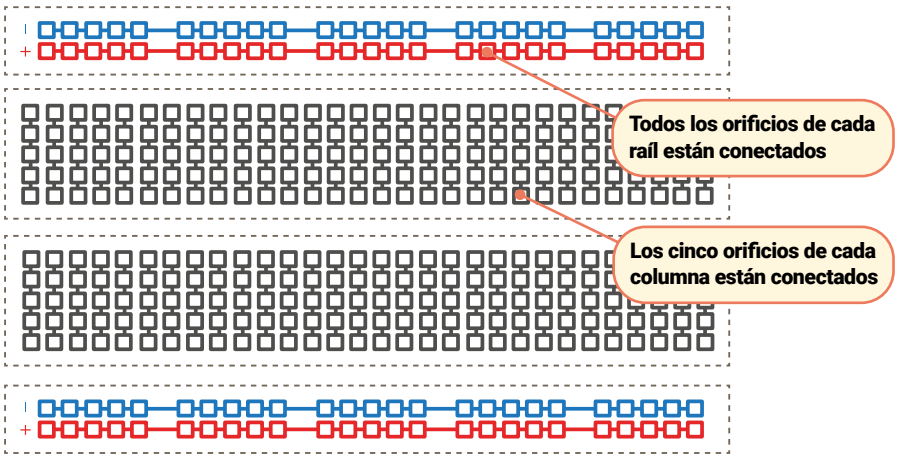
Este programa importa la función de LED desde la biblioteca **`gpiozero`**(GPIO Zero) y la función **`sleep`** de la biblioteca **`time`**. Luego crea un bucle infinito para que el LED se encienda durante un segundo, se apague durante un segundo y repita ese comportamiento. Haz clic en el botón Run para verlo en acción: verás que el LED empieza a parpadear. Al igual que con el programa Scratch, fíjate en lo que ocurre cuando haces clic en el botón de detención mientras el LED está encendido, comparado con lo que ocurre cuando el LED está apagado.

RETO: ILUMINACIÓN PROLONGADA

¿Cómo cambiarías el programa para que el LED se mantenga encendido durante más tiempo? ¿Y para que se mantenga apagado durante más tiempo? ¿Cuál es el menor retardo que puedes usar sin dejar de ver cómo se enciende y se apaga el LED?

Usar una placa de pruebas

Los siguientes proyectos de este capítulo serán mucho más fáciles de completar si se utiliza una placa de pruebas para sujetar los componentes y realizar las conexiones eléctricas.



Una placa de pruebas está cubierta de orificios con separaciones de 2,54 mm, en los que insertar componentes. Debajo de estos orificios hay tiras de metal que actúan como los cables puente que has estado usando. Estos se encuentran en filas a lo largo de la placa. La mayoría de las placas tienen una separación que las divide en dos mitades. Muchas placas de pruebas también tienen letras en la parte superior y números en los laterales. Esto te permite encontrar un orificio concreto: A1 está en la esquina superior izquierda, B1 es el orificio justo a su derecha, mientras que B2 es el orificio situado debajo de B1. A1 está conectado a B1 por las tiras metálicas ocultas, pero ningún orificio 1 está conectado a un orificio 2 a menos que tú añadas un puente.

Las placas de pruebas más grandes también tienen tiras de orificios en los laterales, normalmente identificadas por rayas rojas y negras, o rojas y azules. Estos son los *raíles de energía* y están diseñados para facilitar el cableado: puedes conectar un solo cable del pin de tierra de Raspberry Pi a uno de los raíles (normalmente identificado por una raya azul o negra y un símbolo menos (-) para proporcionar una *base común* para muchos componentes de la placa. Puedes hacer lo mismo si tu circuito necesita 3,3 V o 5 V de energía.

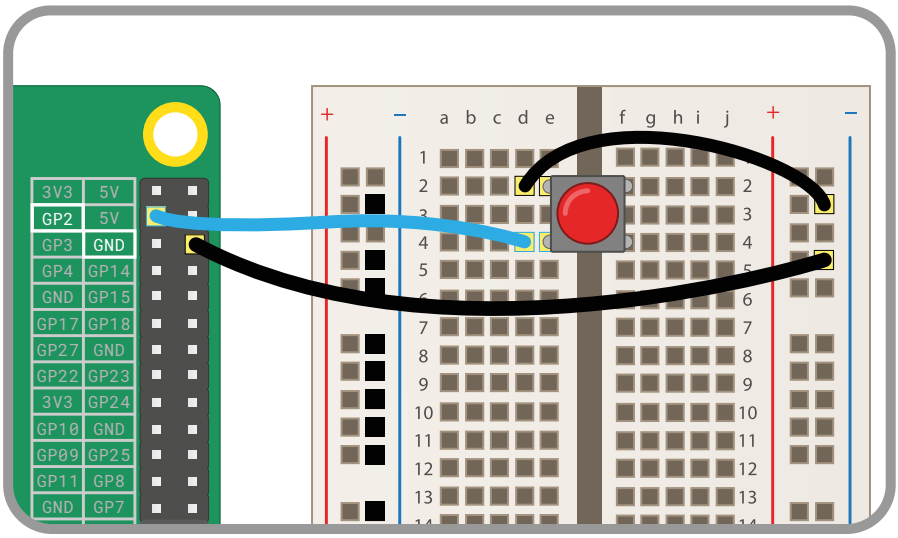
Es fácil añadir componentes electrónicos a una placas de pruebas: solo hay que alinear sus salientes metálicos con los orificios y presionar levemente para que el componente encaje. Si tienes que establecer otras conexiones además de las de la placa de pruebas, puedes usar cables puente macho-macho (M2M): para conexiones entre la placa de pruebas y Raspberry Pi, usa cables puente macho-hembra (M2F).

No intentes insertar más de una pista de componente o puente en un solo orificio de la tabla. Recuerda que los orificios están conectados en columnas, aparte de la división en el medio, por lo que una pista de componente en A1 está conectada eléctricamente a cualquier cosa que añadas a B1, C1, D1 y E1.

Pasos siguientes: leer un botón

Como el sistema GPIO es de "entrada/salida" significa que también puedes usar pines como entradas. Para este proyecto, necesitarás una placa de pruebas, cables puente macho-macho (M2M) y macho-hembra (M2F) y un conmutador momentáneo. Si no tienes una placa de pruebas, puedes usar cables puente hembra-hembra (F2F), pero el botón será mucho más difícil de pulsar, con riesgo de que se rompa el circuito.

Empieza por añadir el conmutador a tu placa de pruebas. Si el conmutador tiene solo dos patas, asegúrate de que estén en diferentes filas numeradas de la placa de pruebas. Si tiene cuatro patas, gíralo para que los lados con patas estén a lo largo de las filas de la placa de pruebas y los lados planos sin patas estén en las partes superior e inferior. Conecta el raíl de tierra de tu placa de pruebas a un pin de tierra de Raspberry Pi (identificado como GND en la **Figura 6-4**) con un cable puente macho-hembra, luego conecta una pata del conmutador al raíl de tierra con un cable puente macho-hembra. Por último, conecta la otra pata (la del mismo lado que la pata que acabas de conectar, si usas un conmutador de cuatro patas) al pin GPIO 2 (identificado como GP2 en la **Figura 6-4**) de Raspberry Pi con un cable puente macho-hembra.



▲ **Figura 6-4:** Conectar un conmutador momentáneo a los pines GPIO

Leer un botón en Scratch

Inicia un nuevo programa de Scratch y arrastra un bloque al hacer clic en al área de código. Conecta un bloque set gpio to input pulled high y selecciona el número 2 en la lista desplegable para que coincida con el pin GPIO que has usado para el conmutador momentáneo.

Si haces clic en la bandera verde ahora, no pasará nada. Eso es porque le has dicho a Scratch que use el pin como entrada, pero no le has dicho qué debe hacer con ella. Arrastra un bloque **por siempre** al final de tu secuencia y luego arrastra un bloque **si no** dentro de él. Encuentra el bloque **gpio is high?**, arrástralo al espacio de forma hexagonal en la parte **si entonces** del bloque y usa la lista desplegable para seleccionar el número 2 e indicarle a GPIO qué pin debe comprobar. Arrastra un bloque **decir ¡Hola! durante 2 segundos** a la parte **si no** del bloque y edítalo para que diga "¡Se ha pulsado el botón!". De momento, deja vacía la parte "si entonces" .

En esa secuencia hay muchas cosas. Para empezar, haz una prueba: haz clic en la bandera verde y luego pulsa el botón de la placa de pruebas. Tu objeto debería decirte que el botón se ha pulsado: ¡has leído una entrada del pin GPIO!

Habrás notado que la parte **si gpio 2 is high? entonces** del bloque está vacía. Y el código que se ejecuta al pulsar el botón está en la parte **si no** del bloque. ¿Te parece confuso? ¿No debería subir el botón al pulsarlo? De hecho, es lo contrario: Los pines GPIO de Raspberry Pi normalmente están subidos, o activados, cuando se configuran como una entrada. Y al pulsarlos, bajan.

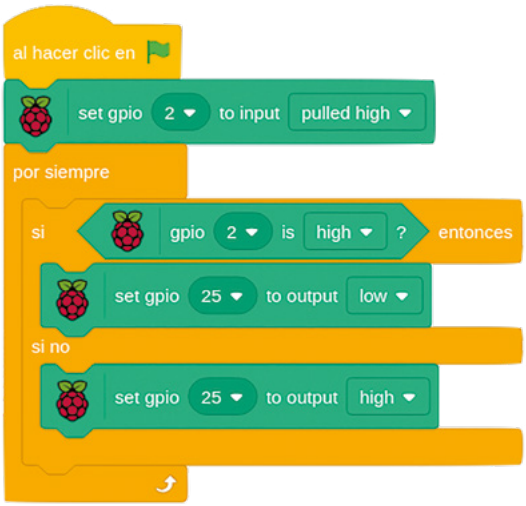
Observa el circuito otra vez: verás que el botón está conectado al pin GPIO 2, que es el que proporciona la parte positiva del circuito, y al pin de tierra. Al pulsar el botón, el voltaje del pin

GPIO se baja a través del pin de tierra, y tu programa Scratch deja de ejecutar el código en tu bloque si gpio 2 is high? entonces, para ejecutar el código en la parte si no del bloque.

Si todo eso suena desconcertante, simplemente recuerda que un botón en un pin de GPIO Raspberry Pi está pulsado cuando el pin baja, no cuando sube.

Para ampliar tu programa, vuelve a añadir el LED y la resistencia al circuito: acuérdate de conectar la resistencia al pin GPIO 25 y la pata larga del LED, y la pata más corta del LED al raíl de tierra de tu placa de pruebas.

Arrastra el bloque decir ¡Se ha pulsado el botón! durante 2 segundos del área de código a la paleta de bloques para eliminarlo y luego sustitúyelo por un bloque set gpio 25 to output high. Recuerda que tendrás que cambiar el número GPIO con la flecha desplegable. Añade un bloque set gpio 25 to output low (acordándote de cambiar los valores) a la parte si gpio 2 is high? entonces actualmente vacía.



Haz clic en la bandera verde y pulsa el botón. El LED permanecerá encendido mientras mantengas el botón pulsado; si lo sueltas, se apagará. Enhorabuena: ya sabes controlar un pin GPIO basado en una entrada de otro.

RETO: MANTÉN LA ILUMINACIÓN

¿Cómo cambiarías el programa para que el LED se mantenga encendido unos segundos, incluso después de soltar el botón? ¿Qué tendrías que cambiar para mantener el LED encendido cuando no estés pulsando el botón y apagado mientras lo pulsas?

Leer un botón en Python

Haz clic en el botón New en Thonny para iniciar un proyecto y el botón Save para guardarlo como **Button Input**. Usar un pin GPIO como entrada para un botón es muy similar a usar un pin como salida para un LED, pero tienes que importar una sección diferente de la biblioteca GPIO Zero. Escribe lo siguiente en el área de script:

```
from gpiozero import Button
button = Button(2)
```

Para que el código se ejecute cuando se pulse el botón, GPIO Zero proporciona la función **`wait_for_press`**. Escribe lo siguiente:

```
button.wait_for_press()
print("You pushed me!")
```

Haz clic en el botón Run y luego pulsa el conmutador momentáneo. Tu mensaje se imprimirá en el shell de Python, en la parte inferior de la ventana de Thonny: has leído una entrada del pin GPIO. Si quieres probar el programa de nuevo, tendrás que volver a hacer clic en el botón Run; como no hay ningún bucle en el programa, este se cierra en cuanto termina de mostrar el mensaje en el shell.

Para ampliar tu programa, vuelve a añadir el LED y la resistencia al circuito (si aún no lo habías hecho): acuérdate de conectar la resistencia al pin GPIO 25 y la pata larga del LED, y la pata más corta del LED al raíl de tierra de tu placa de pruebas.

Para controlar un LED y leer un botón, tendrás que importar las funciones **`Button`** y **`LED`** de la biblioteca GPIO Zero. También necesitarás la función **`sleep`** de la biblioteca **`time`**. Vuelve a la parte superior de tu programa y escribe las dos primeras líneas nuevas:

```
from gpiozero import LED
from time import sleep
```

Debajo de la línea **`button = Button(2)`** escribe:

```
led = LED(25)
```

Borra la línea **`print("¡Me has presionado!")`** y sustitúyela por:

```
led.on()
sleep(3)
led.off()
```

Tu programa final debería tener este aspecto:

```
from gpiozero import LED
from time import sleep
from gpiozero import Button

button = Button(2)
led = LED(25)
button.wait_for_press()
led.on()
sleep(3)
led.off()
```

Pulsa el botón Run y luego pulsa el conmutador momentáneo: el LED se encenderá durante tres segundos, luego se apagará de nuevo y el programa se cerrará. Enhorabuena: ya sabes controlar un LED usando una entrada de botón en Python.

RETO: AÑADE UN BUCLE

¿Cómo añadirías un bucle para que el programa se repita en lugar de cerrarse después de pulsar el botón? ¿Qué tendrías que cambiar para mantener el LED encendido cuando no estés pulsando el botón y apagado mientras lo pulsas?

Haz ruido: cómo controlar un zumbador

Los LED son un excelente dispositivo de salida, pero no sirven de mucho si estás mirando hacia otro lado. La solución: los zumbadores, que hacen un ruido que oirás desde cualquier parte de la habitación. Para este proyecto necesitarás una placa de pruebas, cables puente macho-hembra (M2F) y un zumbador activo. Si no tienes una placa de pruebas, puedes conectar el zumbador usando cables puente hembra-hembra (F2F).

En términos de circuitos y programación, un zumbador activo es igual que un LED. Repite el circuito que hiciste para el LED, pero sustituye el LED por el zumbador activo y no uses la resistencia, ya que el zumbador necesitará más corriente para funcionar. Conecta una pata del zumbador al pin GPIO 15 (identificado como GP15 en la **Figura 6-5**) y la otra al pin de tierra (identificado como GND en el diagrama) usando la placa de pruebas y los cables puente macho-hembra.

Si tu zumbador tiene tres patas, asegúrate de que la marcada con el símbolo menos (-) esté conectada al pin de tierra, y la marcada con "S" o "SIGNAL" esté conectada a GPIO 15. Luego conecta la última pata, que suele ser la del medio, al pin de 3,3 V (identificado como 3V3.)

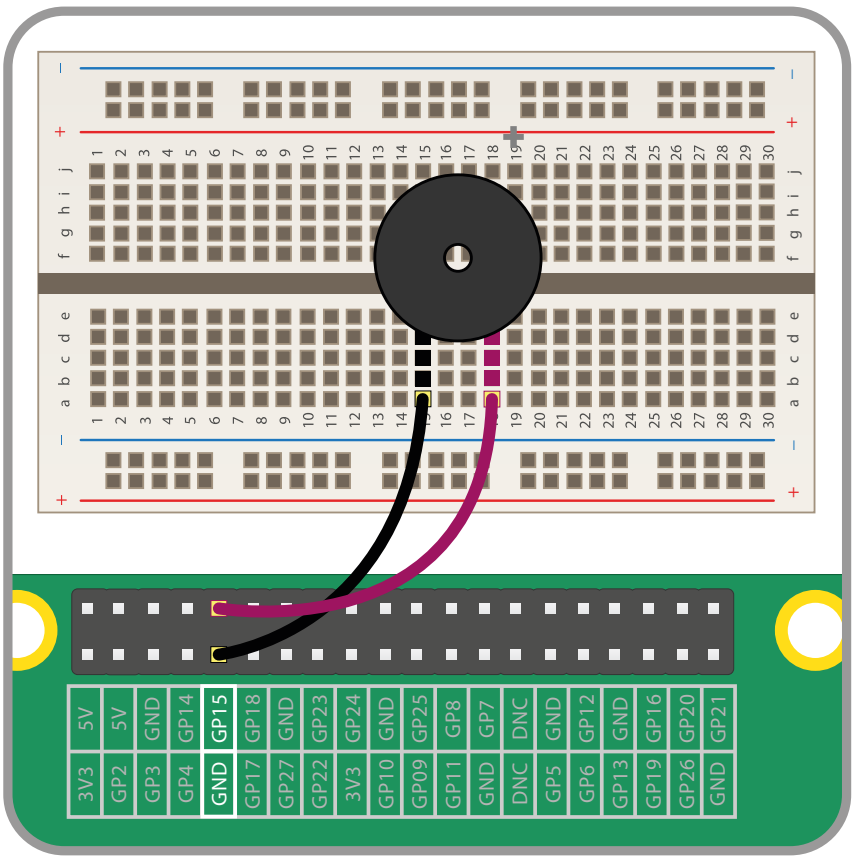


Figura 6-5: Conectar un zumbador a los pines del GPIO

Controlar un zumbador en Scratch

Recrea el programa que utilizaste para hacer que el LED parpadeara, o cárgalo, si lo guardaste antes de crear el proyecto del botón. Usa la lista desplegable de los bloques **set gpio to output high** para seleccionar el número 15, para que Scratch controle el pin GPIO correcto.

Haz clic en la bandera verde y el zumbador empezará a sonar: un segundo sí y un segundo no. Si solo oyes el clic de la acción de zumbido una vez por segundo, es porque se está usando un zumbido pasivo en lugar de uno activo. Cuando un zumbador activo genera la señal de cambio rápida conocida como *oscilación* para hacer que las placas metálicas vibren, un zumbador pasivo necesita una señal oscilante. Cuando simplemente se enciende usando Scratch, las placas solo se mueven una vez y se detienen, con un "clic", hasta la próxima vez que tu programa encienda o apague el pin.

Haz clic en el octógono rojo para detener el zumbador, pero asegúrate de hacerlo cuando no esté haciendo ruido; de lo contrario, el zumbador seguirá sonando hasta que vuelvas a ejecutar el programa.

RETO: CAMBIA EL ZUMBIDO

¿Cómo podrías modificar el programa para que el zumbador suene menos tiempo? ¿Puedes construir un circuito para controlar el zumbador mediante un botón?

Controlar un zumbador en Python

Controlar un zumbador activo mediante la biblioteca GPIO Zero es casi idéntico a controlar un LED, ya que tiene estados de encendido y apagado. Pero necesitas una función diferente, la función **`buzzer`**. Inicia un nuevo proyecto en Thonny, guárdalo como **Buzzer** y escribe lo siguiente:

```
from gpiozero import Buzzer
from time import sleep
```

Al igual que con los LED, GPIO Zero debe saber a qué pin está conectado el zumbador para poder controlarlo. Escribe lo siguiente:

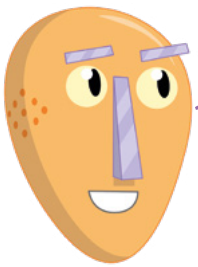
```
buzzer = Buzzer(15)
```

Desde aquí, tu programa es casi idéntico al que escribiste para controlar el LED; la única diferencia (aparte de un número de pin GPIO distinto) es que estás usando **`buzzer`** en lugar de **`led`**. Escribe lo siguiente:

```
while True:
    buzzer.on()
    sleep(1)
    buzzer.off()
    sleep(1)
```

Haz clic en el botón Run y el zumbador empezará a sonar: un segundo sí y un segundo no. Si utilizas un zumbador pasivo en lugar de uno activo, solo oirás un breve clic cada segundo en lugar de un zumbido continuo: esto se debe a que un zumbador pasivo carece de un *oscilador* para crear la señal que cambia rápidamente y hace que vibren las placas dentro del zumbador.

Haz clic en el botón Stop para salir del programa, pero asegúrate de que el zumbador no esté sonando en ese momento, de lo contrario seguirá haciendo ruido hasta que vuelvas a ejecutar el programa.



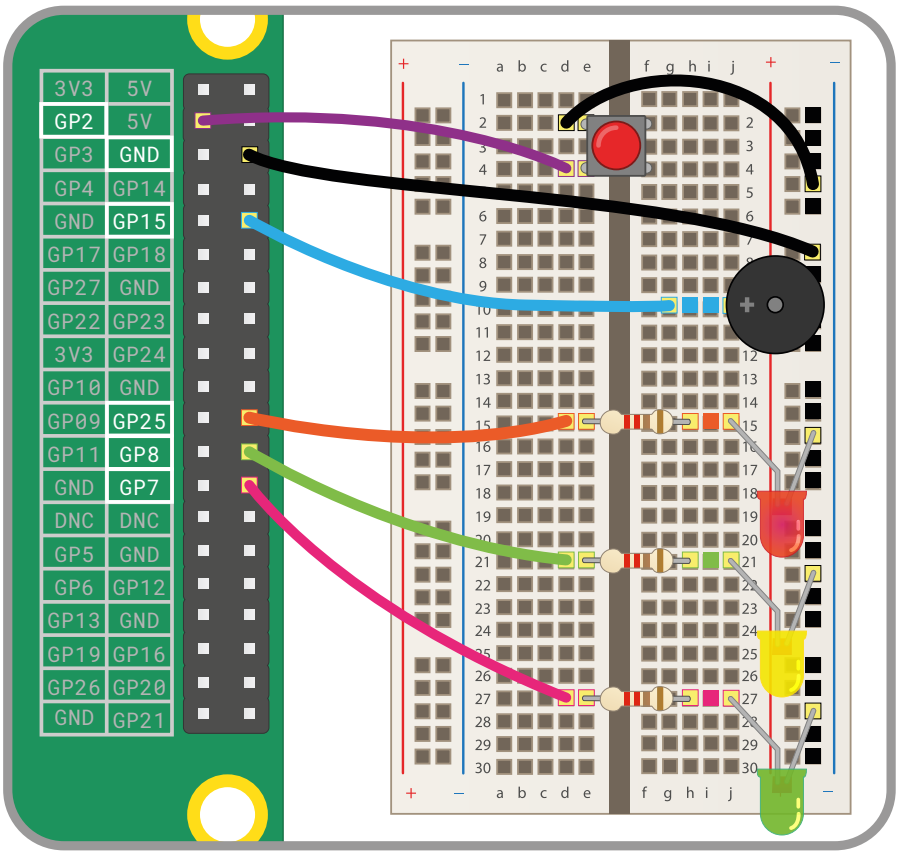
RETO: MEJORA EL ZUMBIDO

¿Cómo podrías modificar el programa para que el zumbador suene menos tiempo? ¿Puedes construir un circuito para controlar el zumbador mediante un botón?

Proyecto de Scratch: Semáforo

Ahora que sabes cómo usar botones, zumbadores y LED como entradas y salidas, puedes empezar a trabajar con un ejemplo de informática del mundo real: un semáforo. Incluirás un botón que pulsar para cruzar la carretera. Para este proyecto, necesitarás una placa de pruebas; un LED rojo, uno amarillo y uno verde; tres resistencias de 330 Ω; un zumbador; un conmutador momentáneo; y una selección de cables puente macho-macho (M2M) y macho-hembra (M2F).

Comienza construyendo el circuito (**Figura 6-6**), conectando el zumbador al pin GPIO 15 (identificado como GP15 en la **Figura 6-6**), el LED rojo al pin GPIO 25 (identificado como GP25), el LED amarillo a GPIO 8 (GP8), el LED verde a GPIO 7 (GP7) y el conmutador a GPIO 2 (GP2). Acuérdate de conectar las resistencias de 330 Ω entre los pines GPIO y las patas largas de los LED, y conecta las segundas patas de todos tus componentes al raíl de tierra de tu placa de pruebas. Por último, conecta el raíl de tierra a un pin de tierra (identificado como GND) en Raspberry Pi para completar el circuito.



▲ **Figura 6-6:** Diagrama de cableado del proyecto del semáforo

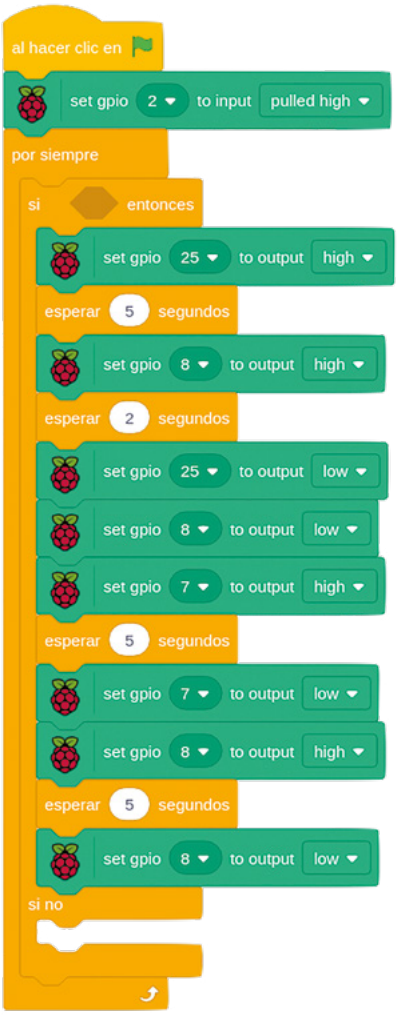
Inicia un nuevo proyecto de Scratch 3 y arrastra un bloque al hacer clic en al área de código. A continuación, tendrás que decirle a Scratch que el pin GPIO 2, que está conectado al conmutador momentáneo de tu circuito, es una entrada (no una salida): arrastra un bloque set gpio to input pulled high de la categoría GPIO de Raspberry Pi de la paleta de bloques y suéltalo bajo tu bloque al hacer clic en. Haz clic en la flecha abajo junto a "0" y selecciona el número 2 de la lista desplegable.

A continuación, tienes que crear tu secuencia de semáforo. Arrastra un bloque por siempre a tu programa, luego llénalo de bloques para encender y apagar los LED del semáforo siguiendo un patrón determinado. Recuerda cómo están conectados los pines GPIO y los componentes: al pin 25 le corresponde el LED rojo, al pin 8 el LED amarillo y al pin 7 el LED verde.

Haz clic en la bandera verde y observa tus LED: primero se encenderá el rojo, luego se encenderán el rojo y el amarillo, luego el verde, luego el amarillo y finalmente la secuencia se repite

una vez más con la luz roja. Este patrón coincide con el utilizado por los semáforos en el Reino Unido; puedes editar la secuencia para que coincida con los patrones de otros países, si lo deseas.

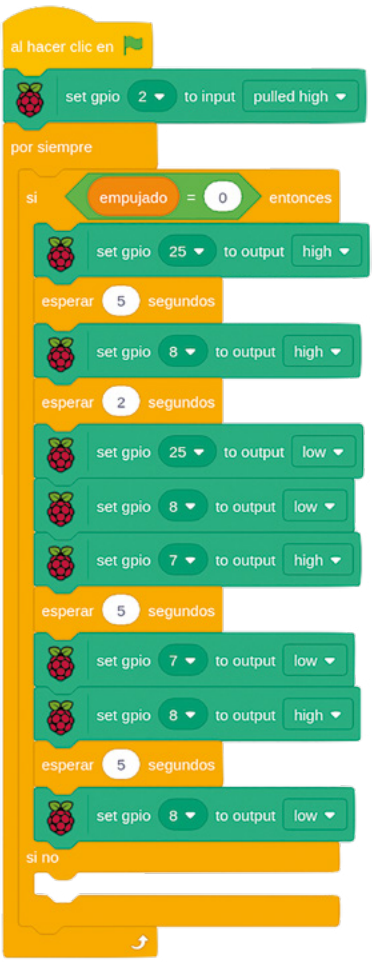
Para simular un paso de peatones, tu programa debe detectar que se pulsa el botón. Haz clic en el octógono rojo para detener el programa, si está en ejecución. Arrastra un bloque **si no** a tu área de script y conéctalo directamente debajo del bloque **por siempre**, con tu secuencia de semáforo en la sección "si entonces". De momento deja vacío el hueco de forma hexagonal.



Un paso de peatones real no cambia la luz a roja en cuanto se pulsa el botón, sino que espera a la siguiente luz roja de la secuencia. Para incorporarlo a tu propio programa, arrastra un bloque **when gpio is low** al área de código y selecciona "2" en la lista desplegable. Arrastra un bloque **dar a empujado el valor 1** debajo de él.

Esta pila de bloques detecta si se pulsa el botón y ajusta la variable "empujado" a 1. Configurar una variable de esta manera te permite guardar la acción de la pulsación del botón, aunque no vayas a hacer nada con ella de momento.

Vuelve a tu pila de bloques original y localiza el bloque si entonces. Arrastra un bloque de Operador = al espacio en blanco del bloque si entonces y luego arrastra un bloque informador empujado al primer espacio en blanco. Escribe "0" sobre el "50" en el lado derecho del bloque.



Haz clic en la bandera verde y observa la secuencia de las luces del semáforo. Pulsa el conmutador momentáneo: al principio parece que no pasa nada, pero cuando la secuencia haya llegado a su fin, con solo el LED amarillo encendido, las luces se apagarán y se mantendrán apagadas, gracias a la variable "empujado".

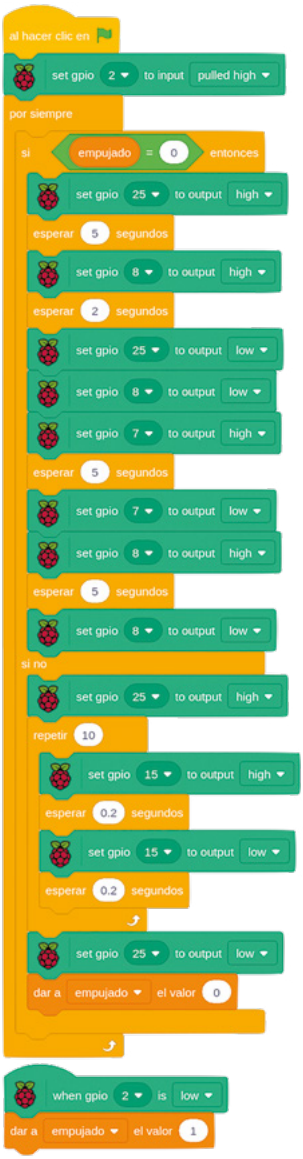
Para terminar, hay que hacer que el botón del paso de peatones haga algo más aparte de apagar las luces. En la pila de bloques principal, localiza el bloque **si no** y arrastra un bloque **set gpio 25 to output high** dentro de él (y acuérdate de cambiar el número de pin GPIO predefinido, para que coincida con el pin al que está conectado tu LED rojo.

Debajo de eso, aún en el bloque **si no**, crea un patrón para el zumbador: arrastra un bloque **repetir 10**, luego llénalo con los bloques **set gpio 15 to output high**, **esperar 0.2 segundos**, **set gpio 15 to output low** y **esperar 0.2 segundos**, cambiando los valores de los pines GPIO para que coincidan con el pin del componente zumbador.

Por último, debajo de la parte inferior del bloque **repetir 10**, pero aún dentro del bloque **si no**, añade un bloque **set gpio 25 to output low** y uno **dar a empujado el valor 0**, el último bloque que restablece la variable que almacena la pulsación del botón, para que la secuencia del zumbador no se repita incesantemente.

Haz clic en la bandera verde y luego pulsa el conmutador en tu placa de pruebas. Después de completarse la secuencia, verás que la luz roja se enciende y el zumbador suena para que los peatones sepan que pueden cruzar de forma segura. Después de un par de segundos, el zumbador se detendrá y la secuencia del semáforo comenzará de nuevo y continuará hasta la próxima vez que pulses el botón.

Enhorabuena: has programado una serie de luces de semáforo completamente funcional, con paso de peatones añadido.



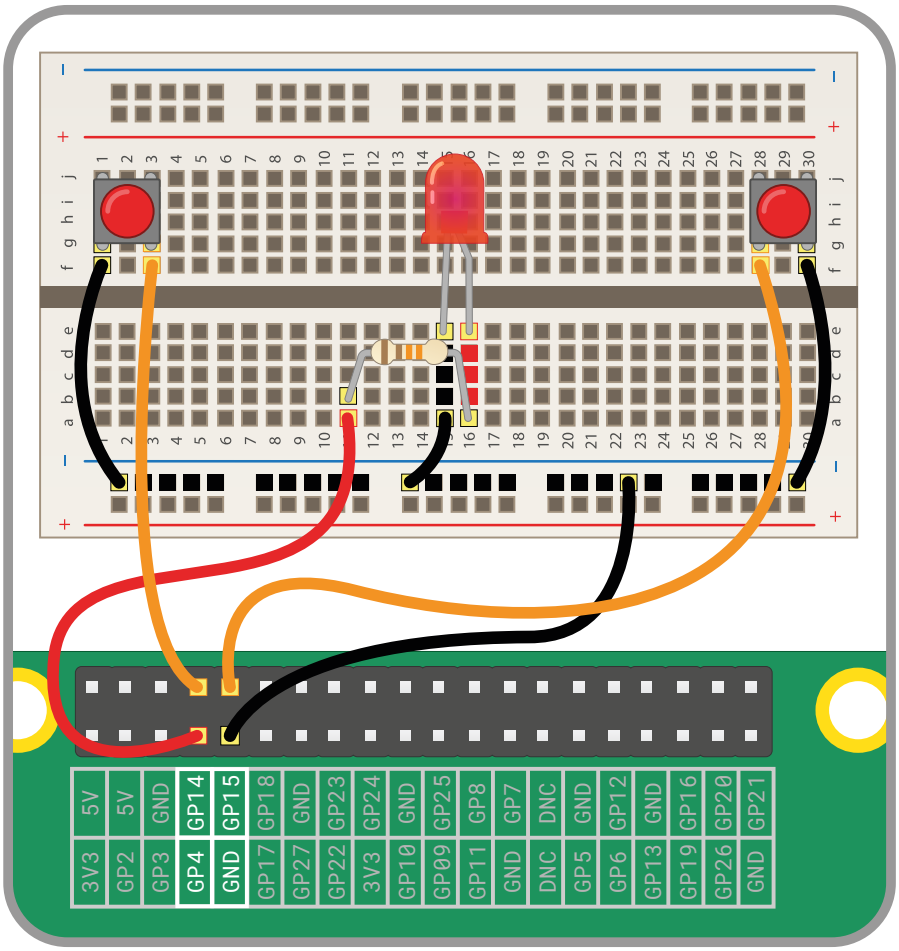
RETO: ¿PUEDES MEJORARLO?

¿Puede cambiar el programa para que un peatón tenga más tiempo para cruzar? ¿Puedes encontrar información sobre los patrones luminosos de los semáforos de otros países y reprogramar tus luces para otro país? ¿Cómo podrías atenuar el brillo de los LED?

Proyecto de Python: Juego de reacción rápida

Ahora que sabes cómo usar botones y LED como entradas y salidas, ya puedes empezar a crear un ejemplo de informática del mundo real: un juego de reacción para dos jugadores, diseñado para ver quién reacciona con más rapidez. Para este proyecto necesitarás una placa de pruebas, un LED y una resistencia de 330 Ω, dos conmutadores momentáneos, cables puente macho-hembra (M2F) y algunos puente macho-macho (M2M).

Empieza construyendo el circuito (**Figura 6-7**): conecta el primer conmutador en el lado izquierdo de tu placa de pruebas al pin GPIO 14 (identificado como GP14 en la **Figura 6-7**), el segundo interruptor en el lado derecho de la plAca al pin GPIO 15 (identificado como GP15), la pata más larga del LED a la resistencia de 330 Ω que luego se conecta al pin GPIO 4 pin (identificado como GP4) de Raspberry Pi y las segundas patas de todos los componentes al raí de tierra de la placa. Por último, conecta el raíl de tierra al pin de tierra de Raspberry Pi (identificado como GND).



▲ **Figura 6-7:** Diagrama de cableado para el juego de reacción rápida

Inicia un nuevo proyecto en Thonny y guárdalo como **Reaction Game**. Vas a usar las funciones **`LED`** y **`button`** de la biblioteca GPIO Zero y la función **`sleep`** de la biblioteca time. Pero, en lugar de importar las dos funciones de GPIO Zero a dos líneas distintas, puedes ahorrar tiempo importándolas juntas, usando un símbolo de coma (**`,`**) para separarlas. Escribe lo siguiente en el área de script:

```
from gpiozero import LED, Button
from time import sleep
```

Igual que antes, tendrás que decirle a GPIO Zero a qué pines están conectados los dos botones y el LED. Escribe lo siguiente:

```
led = LED(4)
right_button = Button(15)
left_button = Button(14)
```

Ahora añade instrucciones para encender y apagar el LED, para comprobar si funciona correctamente:

```
led.on()
sleep(5)
led.off()
```

Haz clic en el botón Run: el LED se encenderá durante cinco segundos, luego se apagará y el programa se cerrará. Pero, para un juego de reacción, que el LED se apague siempre al cabo de exactamente 5 segundos es bastante predecible. Añade lo siguiente debajo de la línea **`from time import sleep`**:

```
from random import uniform
```

La biblioteca random permite generar números aleatorios (aquí con una distribución uniforme - consulta **rpf.io/uniform**). Localiza la línea **`sleep(5)`** y cámbiala para que diga:

```
sleep(uniform(5, 10))
```

Vuelve a pulsar el botón Run: esta vez el LED permanecerá encendido durante un número aleatorio de segundos (entre 5 y 10). Cuenta para ver el tiempo que tarda en apagarse el LED y luego haz clic en el botón Run unas cuantas veces más: verás que el tiempo es diferente para cada ejecución, lo que hace que el programa sea menos predecible.

Para convertir los botones en disparadores para cada jugador, tendrás que añadir una función. Ve a la parte inferior de tu programa y escribe lo siguiente:

```
def pressed(button):
    print(str(button.pin.number) + " ganó el juego")
```

Recuerda que Python usa el método de sangría para saber qué líneas son parte de tu función y Thonny sangrará automáticamente la segunda línea. Por último, añade las dos líneas siguientes para detectar cuándo pulsan los botones los jugadores. Esas líneas no deben tener sangría, de lo contrario Python las tratará como parte de la función.

```
right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

Ejecuta el programa y esta vez pulsa uno de los dos botones en cuanto el LED se apague. Verás un mensaje para el primer botón que se pulse, mostrado en el shell de Python, en la parte inferior de la ventana de Thonny. Lo malo es que también verás mensajes cada vez que se pulse cualquiera de los botones. Y además se utiliza el número de pin en lugar de un nombre descriptivo del botón.

Para arreglar eso, primero pide a los jugadores que indiquen sus nombres. Debajo de la línea **`from random import uniform`**, escribe lo siguiente:

```
left_name = input("El nombre del jugador izquierdo es ")
right_name = input("El nombre del jugador derecho es ")
```

Vuelve a la función y sustituye la línea **`print(str(button.pin.number) + " ganó el juego")`** por:

```
    if button.pin.number == 14:
        print (left_name + " ganó el juego")
    else:
        print(right_name + " ganó el juego")
```

Haz clic en el botón Run y escribe los nombres de ambos jugadores en el área de shell de Python. Al pulsar el botón esta vez (lo más rápido posible después de que el LED se apague), verás que se muestra el nombre del jugador en lugar del número del pin.

Para solucionar el problema de que todas las pulsaciones de botón se notifiquen como ganadoras, tendrás que añadir una nueva función de la biblioteca **`exit`** (una biblioteca del sistema). Bajo la última línea **`import`**, escribe lo siguiente:

```
from os import _exit
```

Luego al final de la función, bajo la línea **`print(right_name + " ganó el juego")`**, escribe:

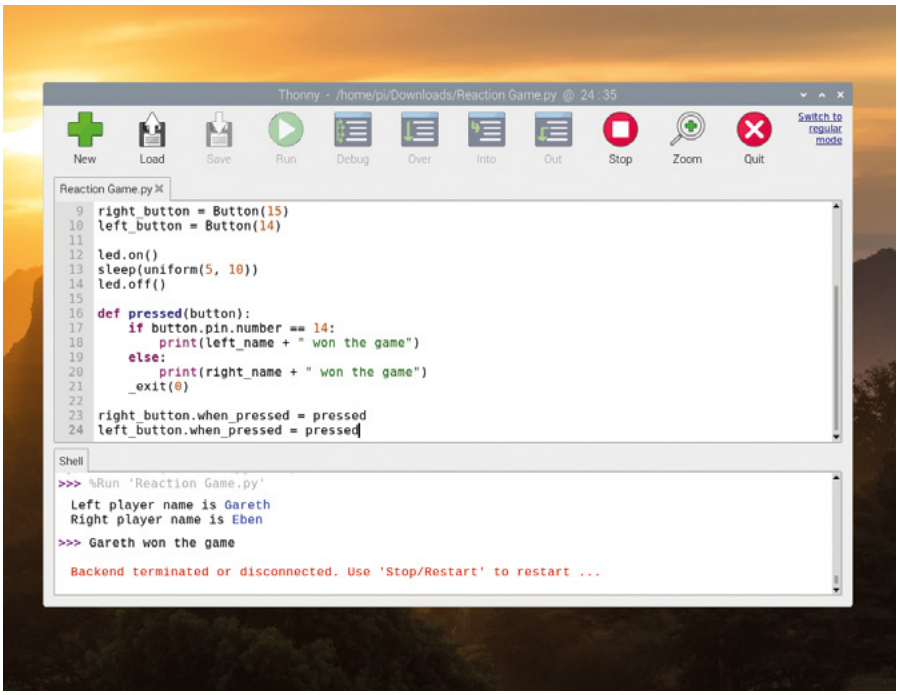
```
    _exit(0)
```

La sangría es importante aquí: **`_exit(0)`** debe tener una sangría de cuatro espacios, alineado con **`else:`** dos línea por encima e **`if`** dos líneas por encima de "else". Esta instrucción le dice a Python que detenga el programa después de pulsarse el primer botón, lo que significa que el jugador cuyo botón se pulse en segundo lugar no obtiene ninguna recompensa por perder.

Tu programa final debería tener este aspecto:

```
from gpiozero import LED, Button
from time import sleep
from random import uniform
from os import _exit

left_name = input("El nombre del jugador izquierdo es ")
right_name = input ("El nombre del jugador derecho es ")
led = LED(4)
right_button = Button(15)
left_button = Button(14)

led.on()
sleep(uniform(5, 10))
led.off()

def pressed(button):
    if button.pin.number == 14:
        print(left_name + " ganó el juego")
    else:
        print(right_name + " ganó el juego")
    _exit(0)

right_button.when_pressed = pressed
left_button.when_pressed = pressed
```

Haz clic en el botón Run, introduce los nombres de los jugadores, espera a que se apague el LED y verás el nombre del jugador ganador. También verás un mensaje de Python: **`Backend terminated or disconnected . Use 'Stop/Restart' to restart ...`**Esto simplemente significa que Python ha recibido el comando **`_exit(0)`** y ha detenido el programa, pero tú tendrás que hacer clic en el icono Stop para cerrarlo completamente y prepararlo para otra ronda (**Figura 6-8**).



▲ **Figura 6-8:** Cuando se decida el ganador, tendrás que detener el programa

Enhorabuena: has creado tu propio juego de informática física.

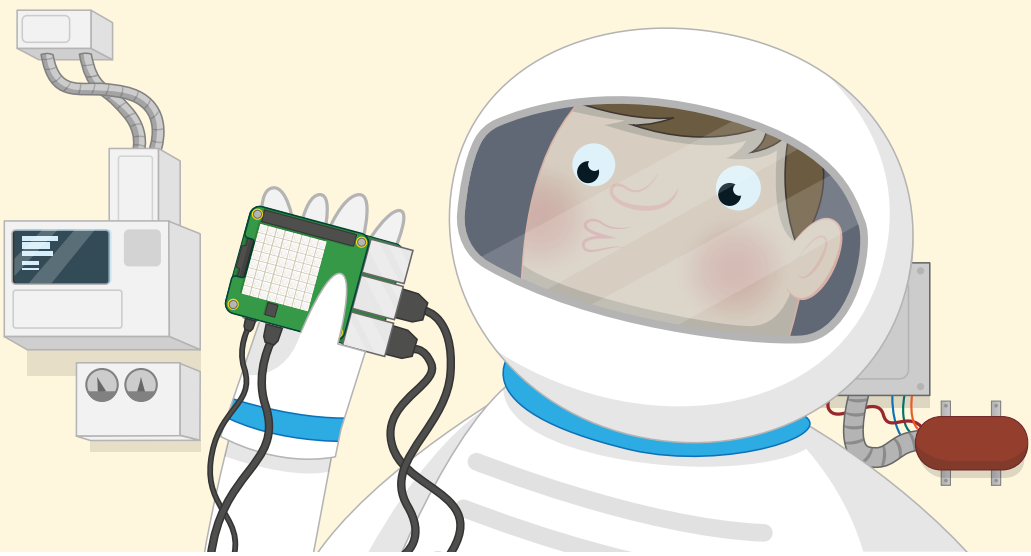
RETO: MEJORA EL JUEGO

¿Puedes añadir un bucle para que el juego se ejecute continuamente? Acuérdate de quitar la instrucción `_exit(0)` antes que nada. ¿Puedes agregar un contador para ver quién está ganando en varias rondas? ¿Y un cronómetro para ver cuánto tiempo tardas en reaccionar al apagarse la luz?

Capítulo 7

Informática física con Sense HAT

Sense HAT es una placa multifuncional complementaria para Raspberry Pi, dotada de sensores y una matriz LED. Se utiliza en la Estación Espacial Internacional.



Raspberry Pi admite un tipo especial de placa adicional conocido como *HAT*, siglas de Hardware Attached on Top. Las placas HAT pueden añadir a Raspberry Pi todo tipo de dispositivos: micrófonos, luces, relés electrónicos, pantallas... pero Sense HAT es la más especial.

La placa Sense HAT se diseñó para la misión espacial Astro Pi. Esta misión fue un proyecto de colaboración entre la Fundación Raspberry Pi, la Agencia Espacial del Reino Unido y la Agencia Espacial Europea, para el que una nave de suministro Cygnus llevó placas Raspberry Pi y placas Sense HAT a la Estación Espacial Internacional. Desde su puesta en órbita sobre la Tierra, las placas Sense HAT (apodadas Ed e Izzy por los astronautas) se han utilizado para ejecutar el código y llevar a cabo experimentos científicos a los que han contribuido escolares de toda Europa.

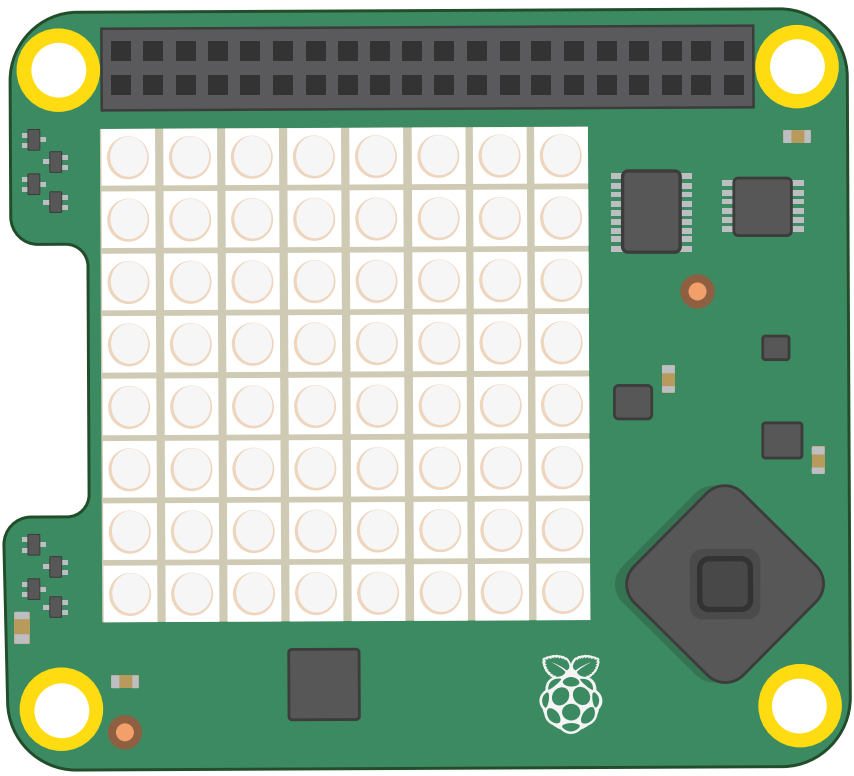
Aunque Ed e Izzy te pillan un poco lejos para poder usarlos, el hardware que utilizan también se encuentra en la Tierra, en todos los distribuidores de Raspberry Pi. Y si no quieres comprar una Sense HAT ahora mismo, ¡puedes simularla en el software!

REALIDAD O SIMULACIÓN

Preferiblemente, este capítulo debería seguirse con una Sense HAT real acoplada a un sistema GPIO de Raspberry Pi. Pero si no tienes una, puedes pasar a la sección titulada "Instalar Sense HAT" y probar los proyectos en el emulador de Sense HAT: el resultado será igual.

La placa Sense HAT

La Sense HAT es un poderoso complemento multifuncional para Raspberry Pi. Además de una matriz 8×8 con 64 LED programables de color rojo, verde y azul (RGB) que pueden controlarse para producir cualquier color de una gama de millones, la Sense HAT incluye un joystick de cinco direcciones y seis sensores integrados.



Sensor giroscópico: Se utiliza para percibir cambios de ángulo a lo largo del tiempo (en términos técnicos se denomina *velocidad angular*) siguiendo la dirección del campo de gravedad de la Tierra, la fuerza que atrae a las cosas hacia el centro de nuestro planeta. O sea, el sensor detecta cuando la Sense HAT gira con relación a la superficie de la Tierra y lo rápido que gira.

Acelerómetro: Es similar al sensor giroscópico, pero en lugar de controlar un ángulo relativo a la gravedad de la Tierra, mide la fuerza de aceleración en múltiples direcciones. Al leer de forma combinada los datos de los dos sensores se puede hallar la dirección a la que apunta una Sense HAT y cómo se está moviendo.

Magnetómetro: Mide la fuerza de un campo magnético y es otro sensor que puede ayudar a rastrear los movimientos de la Sense HAT. Midiendo el campo magnético natural de la Tierra, el magnetómetro puede encontrar la dirección del norte magnético. El mismo sensor puede usarse también para detectar objetos metálicos e incluso campos eléctricos. Los tres sensores están integrados en un solo chip, identificado como ACCEL/GYRO/MAG en el circuito de la Sense HAT.

Sensor de humedad: Mide la cantidad de vapor de agua en el aire, denominada *humedad relativa*. La humedad relativa puede variar desde 0 %, si no hay agua en absoluto, hasta 100 %, si el aire está completamente saturado. Los datos de humedad pueden usarse para detectar si está a punto de llover**.**

Sensor de presión barométrica: también denominado *barómetro*, mide la presión del aire. Aunque la mayoría de la gente sabe que la presión barométrica está relacionada con el pronóstico del tiempo, el barómetro tiene un segundo uso secreto: puede rastrear cuando estamos subiendo o bajando una colina o una montaña, porque el aire se hace menos denso y la presión es menor cuanto más nos alejamos del nivel del mar en nuestro planeta.

Sensor de temperatura: mide lo caliente o frío que es el entorno circundante, aunque también se ve afectado por lo caliente o fría que esté la placa Sense HAT: si usas una carcasa, los datos de la lectura podrían ser más altos de lo previsto. La Sense HAT no tiene un sensor de temperatura específico, sino que utiliza sensores de temperatura incorporados en los sensores de humedad y presión barométrica. Un programa puede usar uno o ambos sensores, tú decides.

SENSE HAT EN RASPBERRY PI 400

Sense HAT es totalmente compatible con Raspberry Pi 400 y se puede insertar directamente en el sistema GPIO de la parte posterior. Pero eso significa que los LED estarán orientados en dirección contraria a tu posición y la placa estará orientada al revés.

Para solucionarlo, necesitarás un cable o una placa de extensión GPIO. Entre las extensiones compatibles está la gama Black HAT Hack3r de Pimoroni. Puedes usar Sense HAT con la placa Black HAT Hack3r o simplemente usar el cable plano de 40 pines incluido como extensión.
En cualquier caso, debes leer siempre las instrucciones del fabricante para asegurarte de que conectas el cable y Sense HAT con la orientación correcta.

Instalar Sense HAT

Si tienes una Sense HAT física, sácala del embalaje y cerciórate de que está completa con todas las piezas: la Sense HAT propiamente dicha, cuatro piezas de metal o plástico denominadas *espaciadores* y ocho tornillos. También puede haber pines de metal en una tira de plástico negro, como los pines GPIO de Raspberry Pi. De ser así, presiona la tira (orientada hacia arriba) sobre la parte inferior de la Sense HAT hasta que oigas un clic.

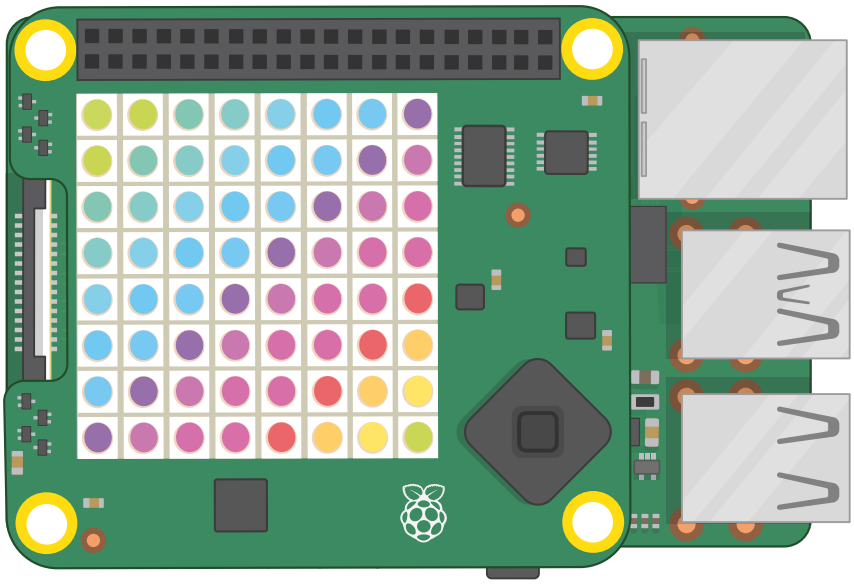
Los espaciadores están diseñados para evitar que la Sense HAT se doble y arquee al usar el joystick. La Sense HAT funcionará aunque no coloques los espaciadores, pero su uso ayudará a proteger de daños a la Sense HAT, a Raspberry Pi y al sistema GPIO.

¡ADVERTENCIA!

Los módulos HAT solo deben conectarse y quitarse del sistema GPIO con Raspberry Pi apagado y desconectado de la fuente de alimentación. La placa Sense HAT debe estar siempre bien asentada al instalarlo. Asegúrate de que los pines estén alineados con los pasadores del sistema GPIO antes de presionar la placa.

Coloca los espaciadores empujando cuatro de los tornillos de abajo arriba en la base de Raspberry Pi, a través de los cuatro orificios de montaje en las esquinas, y atornilla los espaciadores. Presiona la Sense HAT sobre el sistema GPIO de Raspberry Pi, asegurándote de alinearla correctamente con los pines de debajo y de mantenerla lo más plana posible. Por último, coloca los últimos cuatro tornillos en los orificios de montaje de la Sense HAT y en los espaciadores que has colocado previamente. Si la has instalado correctamente, la Sense HAT debe estar plana y nivelada y no debe doblarse ni oscilar cuando acciones el joystick.

Vuelve a encender Raspberry Pi. Verás los LED de la Sense HAT iluminarse con un patrón de arco iris (**Figura 7-1**) y luego se apagarán. Has completado la instalación de la Sense HAT.

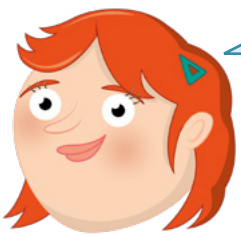


▲ **Figura 7-1:** Patrón luminoso de arco iris con el primer encendido

Si quieres retirar la Sense HAT, simplemente quita los tornillos superiores, levanta la placa (con cuidado de no doblar los pines del sistema GPIO, porque la Sense HAT se ajusta bastante; podrías necesitar un destornillador pequeño) y luego quita los espaciadores de Raspberry Pi.

¡Hola, Sense HAT!

Como en todos los proyectos de programación, lo obvio para empezar con Sense HAT es mediante un mensaje de bienvenida a través de su pantalla LED. Si utilizas el emulador de Sense HAT, cárgalo ahora: haz clic en el icono del menú del sistema operativo Raspberry Pi, elige la categoría Programación y haz clic en la opción del emulador.



EXPERIENCIA EN PROGRAMACIÓN

Este capítulo presupone que el usuario tiene experiencia con Scratch 3 o Python y el entorno de desarrollo integrado (IDE) Thonny, dependiendo de si estás trabajando con los ejemplos de código de Scratch, de Python o de o ambos. Si aún no lo has hecho, ve al **Capítulo 4, Programar con Scratch** o al **Capítulo 5, Programar con Python** y completa primero los proyectos de ese capítulo.

Saludos de Scratch

Carga Scratch 3 desde el menú del sistema operativo Raspberry Pi. Haz clic en el botón Añadir extensión en la parte inferior izquierda de la ventana de Scratch. Haz clic en la extensión Sense HAT de Raspberry Pi (**Figura 7-2**). Esto carga los bloques necesarios para controlar las diversas características de Sense HAT, incluyendo su pantalla LED. Cuando los necesites, los encontrarás en la categoría Sense HAT de Raspberry Pi.

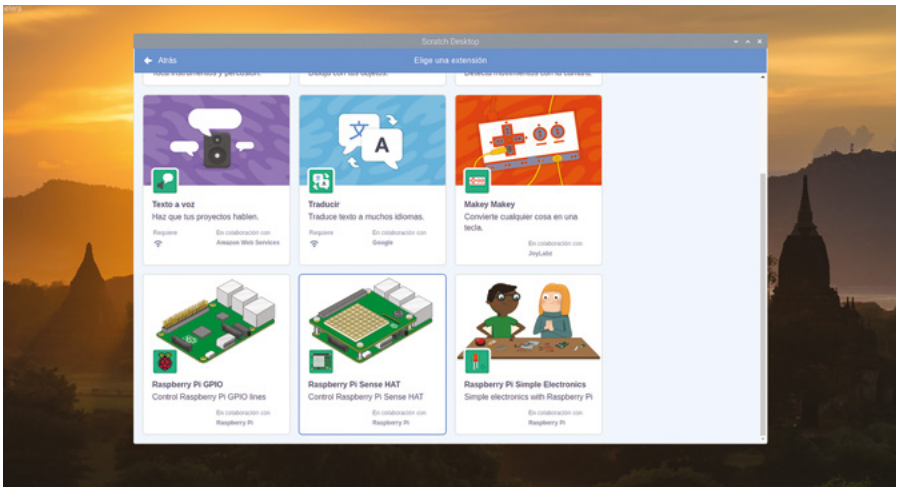


Figura 7-2: Agregar la extensión Sense HAT de Raspberry Pi a Scratch 3

Arrastra un bloque de Eventos al hacer clic en al área de script y luego arrastra un bloque display text ¡Hola! directamente debajo del anterior. Edita el texto para que el bloque diga display text ¡Hola mundo!.

Haz clic en la bandera verde en el área de escenario y observa Sense HAT o su emulador: el mensaje se desplazará lentamente a través de la matriz LED de Sense HAT, iluminando los píxeles de los LED para formar las letras una por una (**Figura 7-3** a continuación). Enhorabuena: tu programa funciona correctamente.

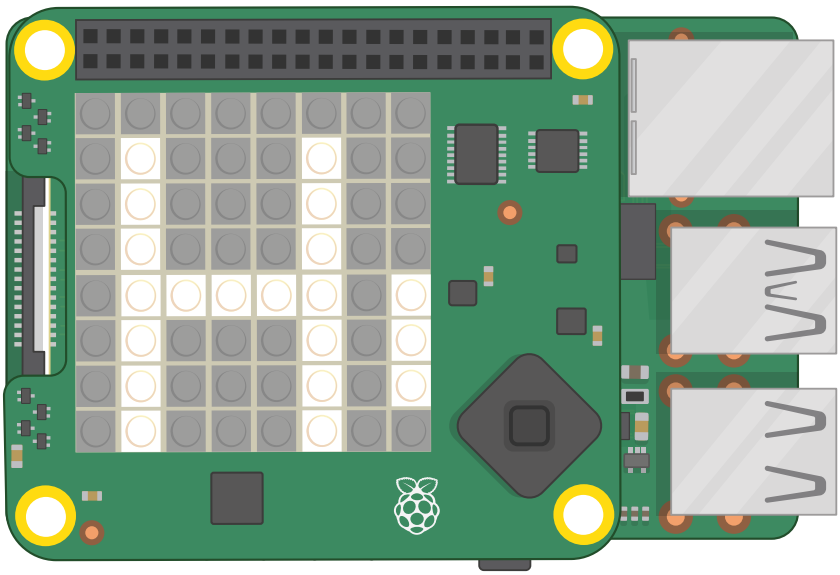


Figura 7-3: El mensaje se desplaza a través de la matriz LED

Ahora que ya sabes crear un mensaje desplazable sencillo, aprenderás a controlar cómo se muestra ese mensaje. Además de poder modificar el mensaje, puedes cambiar la rotación, o sea, cómo aparece el mensaje en Sense HAT. Arrastra un bloque set rotation to 0 degrees desde la paleta de bloques e insértalo debajo de al hacer clic en y encima de display text ¡Hola mundo!. Luego haz clic en la flecha abajo junto a 0 y cambia el valor a 90. Haz clic en la bandera verde y verás el mismo mensaje que antes, pero en lugar de desplazarse de izquierda a derecha se desplazará de abajo arriba (**Figura 7-4**) ¡y tendrás que girar la cabeza, o Sense HAT, para leerlo!

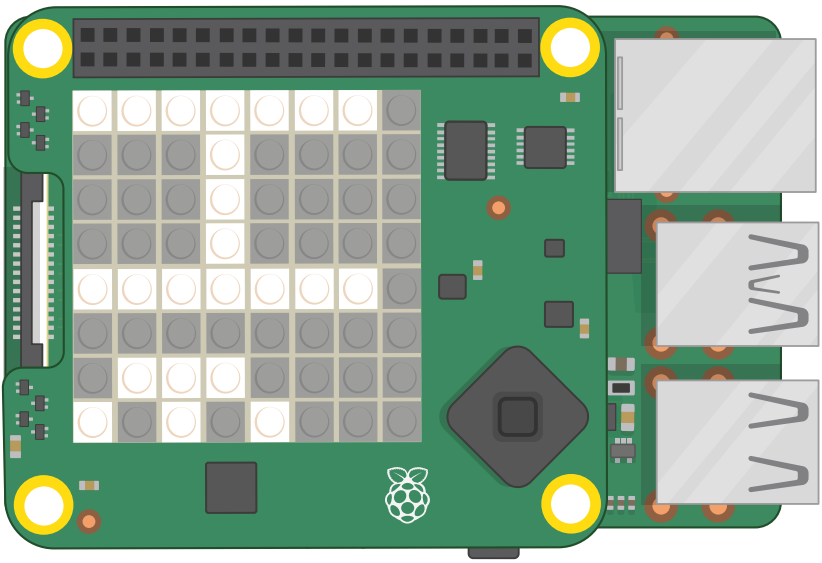


Figura 7-4: Ahora el mensaje se desplaza verticalmente

Vuelve a cambiar el valor de rotación a 0 y luego arrastra un bloque set colour entre set rotation to 0 degrees y display text ¡Hola mundo!. Haz clic en el color al final del bloque para que aparezca el selector de colores de Scratch y localiza un amarillo brillante. Luego haz clic en la bandera verde para ver cómo cambia el resultado de tu programa (**Figura 7-5**).

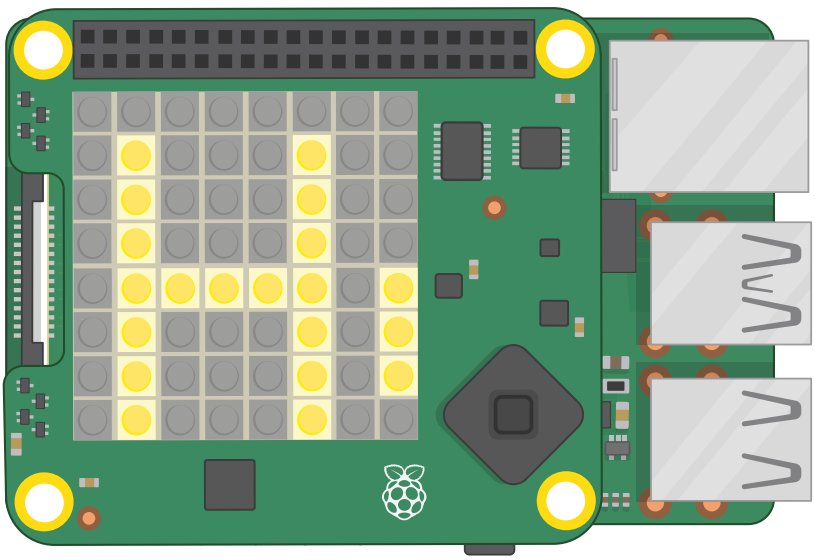
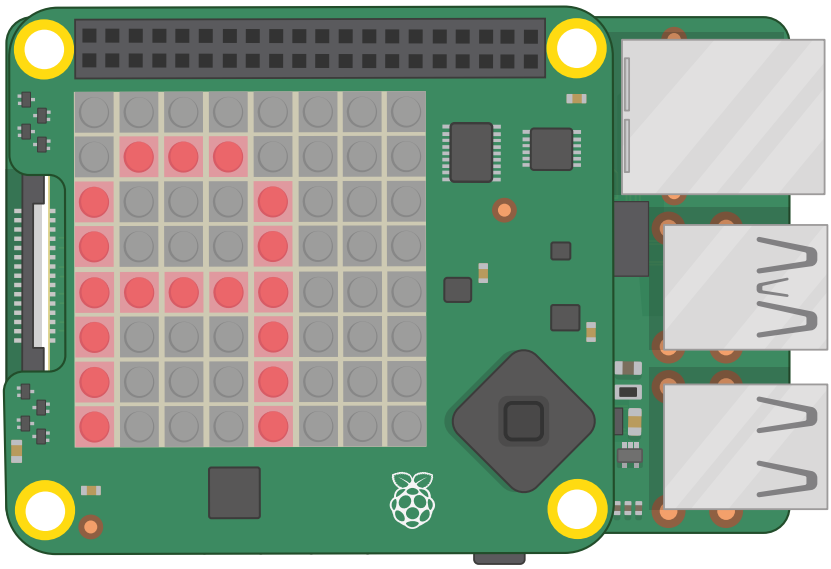


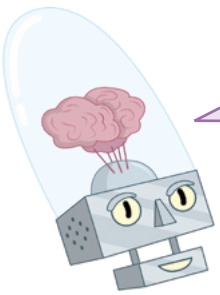
Figura 7-5: Cambiar el color del texto

Por último, arrastra un bloque **set background** entre **set colour to amarillo** y **display text ¡Hola mundo!**, y haz clic en el color para que volver a abrir el selector. Esta vez, la elección del color no afecta a los LED que componen el mensaje, sino a los demás, que son el "fondo". Elige un tono azul y haz clic en la bandera verde: ahora el mensaje será de color amarillo intenso sobre un fondo azul. Prueba con distintos colores para encontrar tu combinación favorita, no todos los colores quedan bien juntos.

Además de desplazar mensajes enteros, puedes mostrar letras individuales. Arrastra el bloque **display text** fuera del área de script para eliminarlo y sustitúyelo arrastrando un bloque **display character A** al lugar que ocupaba el eliminado. Haz clic en la bandera verde y verás la diferencia: este bloque muestra las letras de una en una y cada letra permanece en Sense HAT hasta que tú lo indiques. Este bloque usa los mismos bloques de control de color que el bloque **display text**: cambiar el color de la letra a rojo (**Figura 7-6**).



▲ **Figura 7-6:** Mostrar una sola letra



RETO: REPETIR EL MENSAJE

¿Puedes usar tus conocimientos sobre los bucles para que se repita un mensaje deslizante? ¿Puedes crear un programa que deletree una palabra usando un color distinto para cada letra?

Saludos de Python

Carga Thonny haciendo clic en el icono del menú de Raspberry, eligiendo Programación y haciendo clic en Thonny. Si utilizas el emulador de Sense HAT y lo tapa la ventana de Thonny, mantén pulsado el botón del ratón en la barra de título (de color azul, en la parte superior) y arrastra para moverla por el escritorio hasta que sean visibles ambas ventanas.

CAMBIO DE LÍNEA DE PYTHON

El código de Python escrito para una placa Sense HAT física funciona en el emulador de Sense HAT, y viceversa, con un solo cambio. Si estás usando el emulador de Sense HAT con Python tendrás que cambiar la línea `sense_hat import SenseHat` en todos los programas de este capítulo a `from sense_emu import SenseHat`. Si quieres volver a ejecutarlo en una Sense HAT física, simplemente cambia la línea de nuevo.

Para usar Sense HAT o el emulador en un programa en Python, debes importar la biblioteca de Sense HAT. Escribe lo siguiente en el área de script, acordándote de usar **`sense_emu`** (en lugar de **`sense_hat`**) si estás usando el emulador de Sense HAT:

```
from sense_hat import SenseHat
sense = SenseHat()
```

La biblioteca de Sense HAT tiene una función simple para seleccionar un mensaje, formatearlo para mostrarlo en la pantalla LED y desplazarlo lentamente. Escribe lo siguiente:

```
sense.show_message("¡Hola mundo!")
```

Guarda el programa como **Hello Sense HAT** y haz clic en el botón Run. Verás que el mensaje se desplaza lentamente a través de la matriz LED de Sense HAT, iluminando los píxeles de los LED para formar las letras una por una (**Figura 7-7** a continuación). Enhorabuena: tu programa funciona correctamente.

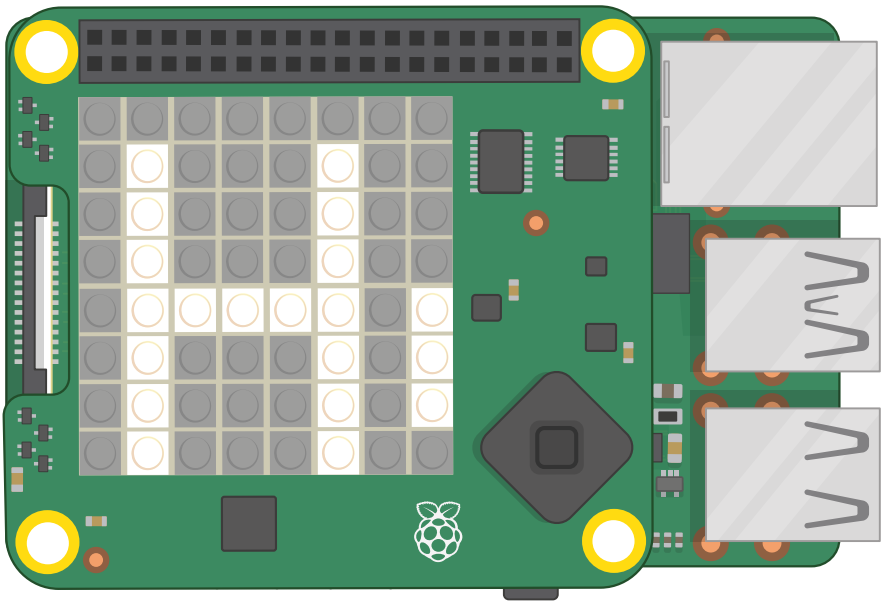


Figura 7-7: Mensaje desplazándose a través de la matriz LED

La función **`show_message()`** tiene aún más posibilidades. Vuelve a tu programa y edita la última línea para que diga:

```
sense.show_message("¡Hola Mundo!", text_colour=(255, 255, 0),
back_colour=(0, 0, 255), scroll_speed=(0.05))
```

Estas instrucciones adicionales, separadas por comas, se denominan *parámetros* y controlan varios aspectos de la función **`show_message()`**. El más simple es **`scroll_speed=()`**, que cambia la rapidez con la que el mensaje se desplaza por la pantalla. Un valor de 0.05 aquí hace que el desplazamiento tenga una velocidad aproximadamente el doble de la habitual. Cuanto más alto sea el número, menor será la velocidad.

Los parámetros **`text_colour=()`** y **`back_colour=()`**, escritos con ortografía del inglés británico, a diferencia de la mayoría de las instrucciones de Python, establecen el color de las letras y el fondo respectivamente. Pero no aceptan nombres de colores: tienes que indicar el color que quieras como un trío de números. El primer número representa la cantidad de rojo en el color (desde 0 para nada de rojo, hasta 255 para la mayor cantidad de rojo posible); el segundo número es la cantidad de verde; y el tercero la cantidad de azul. Juntos se denomina *RGB* (las iniciales en inglés de rojo, verde y azul).

Haz clic en el icono Run y observa Sense HAT: ahora el mensaje se desplaza bastante más rápidamente, con letras amarillas sobre un fondo azul (**Figura 7-8**). Cambia los parámetros para encontrar una combinación de velocidad y color adecuada.

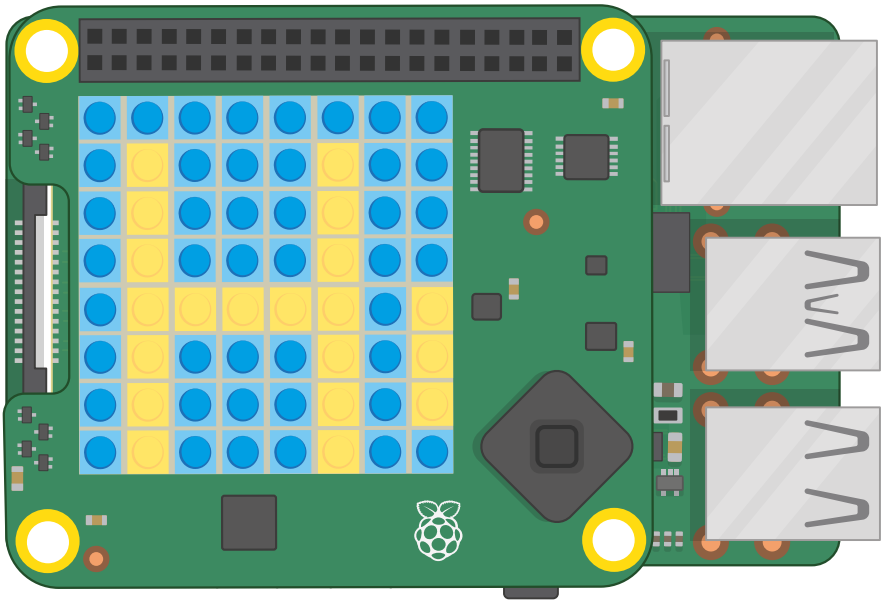


Figura 7-8: Cambiar los colores del mensaje y el fondo

Si quieres usar nombres descriptivos en lugar de valores RGB para definir los colores, tendrás que crear variables. Encima de la línea **`sense.show_message()`**, añade lo siguiente:

```
amarillo = (255, 255, 0)
azul = (0, 0, 255)
```

Vuelve a la línea **`sense.show_message()`** y edítala para que diga:

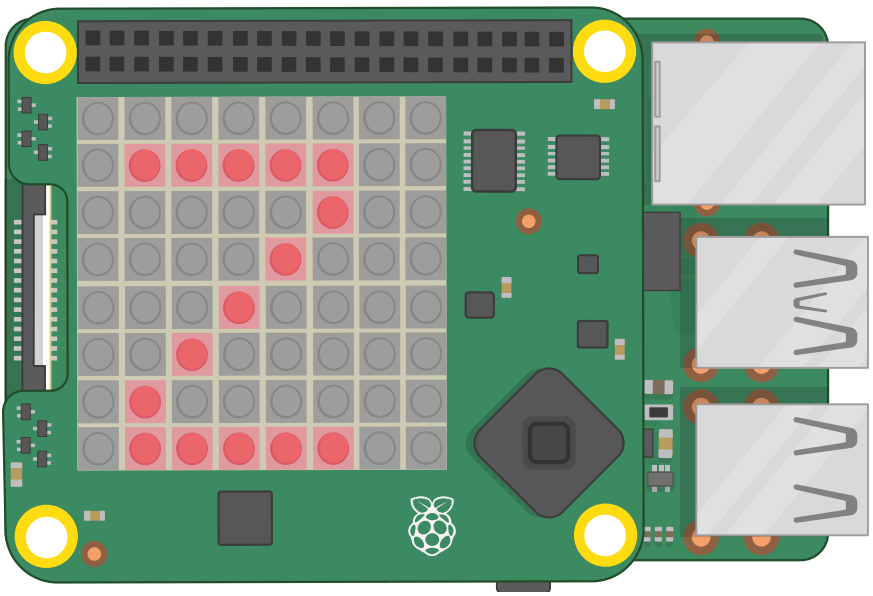
```
sense.show_message("¡Hola Mundo!", text_colour=(amarillo), back_
colour=(azul), scroll_speed=(0.05))
```

Vuelve a hacer clic en el icono Run y verás que nada ha cambiado: tu mensaje sigue siendo amarillo sobre fondo azul. Pero esta vez has usado los nombres de las variables para hacer tu código más legible: en lugar de una cadena de números, el código explica el color configurando. Puedes definir tantos colores como quieras: haz la prueba añadiendo una variable denominada "rojo" con los valores 255, 0 y 0; una variable "blanco" con los valores 255, 255, 255; y una variable "negro" con los valores 0, 0 y 0.

Además de desplazar mensajes enteros, puedes mostrar letras individuales. Borra toda la línea **`sense.show_message()`** y escribe lo siguiente en su lugar:

```
sense.show_letter("Z")
```

Haz clic en Run y verás que la letra "Z" aparece en la pantalla de Sense HAT. La letra permanece donde está, porque las letras individuales, a diferencia de los mensajes, no se desplazan automáticamente. Puedes controlar **`sense.show_letter()`** con los mismos parámetros de color que **`sense.show_message()`**: cambia el color de la letra a rojo (**Figura 7-9**).



▲ **Figura 7-9:** Mostrar una sola letra

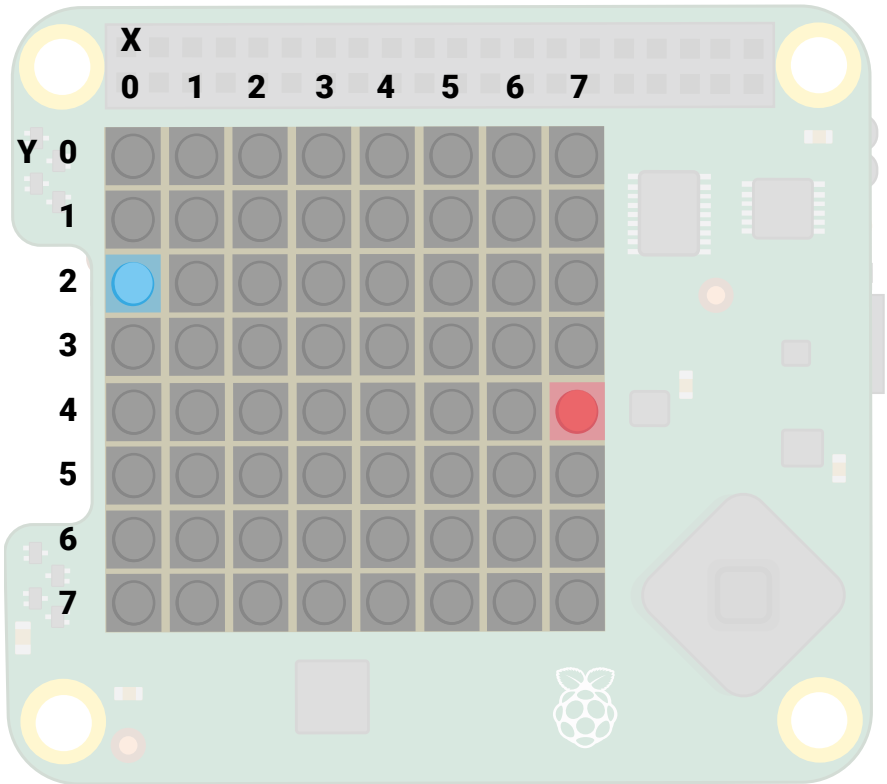
RETO: REPETIR EL MENSAJE

¿Puedes usar tus conocimientos sobre los bucles para que se repita un mensaje deslizante? ¿Puedes crear un programa que deletree una palabra usando un color distinto para cada letra? ¿Con qué rapidez puedes hacer que se deslice un mensaje?

Pasos siguientes: Dibujar con luz

La pantalla LED de Sense HAT no es solo para mensajes: también puedes mostrar imágenes. Cada LED puede usarse como un pixel, abreviatura de *picture element* (elemento de imagen) en una imagen de tu elección, para hacer más interesantes tus programas con imágenes e incluso con animaciones.

Para crear dibujos, tendrás que cambiar los LED individuales. Para hacerlo, debes entender el diseño que hace que la matriz LED de Sense HAT escriba un programa que encienda o apague los LED correctos.



▲ **Figura 7-10:** Sistema de coordenadas de matriz LED

Hay ocho LED en cada fila de la pantalla y ocho en cada columna (**Figura 7-10**). Ten en cuenta que, al contar los LED, deberías empezar desde el 0 y terminar en 7 (como en la mayoría de los lenguajes de programación). El primer LED está en la esquina superior izquierda y el último en la esquina inferior derecha. Usando los números de las filas y columnas, puedes encontrar las *coordenadas* de cualquier LED de la matriz. El LED azul de la matriz mostrada está en las coordenadas 0, 2 y el rojo en las coordenadas 7, 4. El eje X, que atraviesa la matriz en horizontal, es el primero indicado, seguido por el eje Y, de arriba abajo en la matriz.

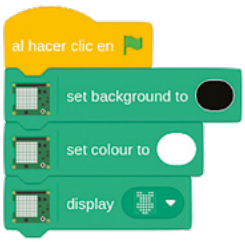
Cuando vayas a incluir dibujos en Sense HAT puede ser útil dibujarlos a mano primero, en papel cuadriculado, o utilizar una hoja de cálculo como LibreOffice Calc.

Imágenes en Scratch

Empieza un nuevo proyecto en Scratch. Guarda tu programa existente, si quieres conservarlo. Si has estado trabajando en los proyectos de este capítulo, Scratch 3 mantendrá cargada la extensión Sense HAT de Raspberry Pi. Si has cerrado y vuelto a abrir Scratch 3 desde tu último proyecto, carga la extensión usando el botón Añadir extensión. Arrastra un bloque de Eventos 

al área de código y luego arrastra un bloque 

y uno set colour directamente debajo. Edita ambos para definir el color de fondo negro y el color de texto blanco. Para crear el color negro, mueve los controles deslizantes de brillo y saturación a 0. Para el blanco desliza el control de brillo a 100 y el de saturación a 0. Tendrás que hacer esto al comienzo de cada programa de Sense HAT, de lo contrario Scratch usará los últimos colores que hayas elegido, aunque fueran para otro programa. Por último, arrastra un bloque display frambuesa al final de tu programa.



Haz clic en la bandera verde: verás que los LED de Sense HAT iluminan una frambuesa (**Figura 7-11**).

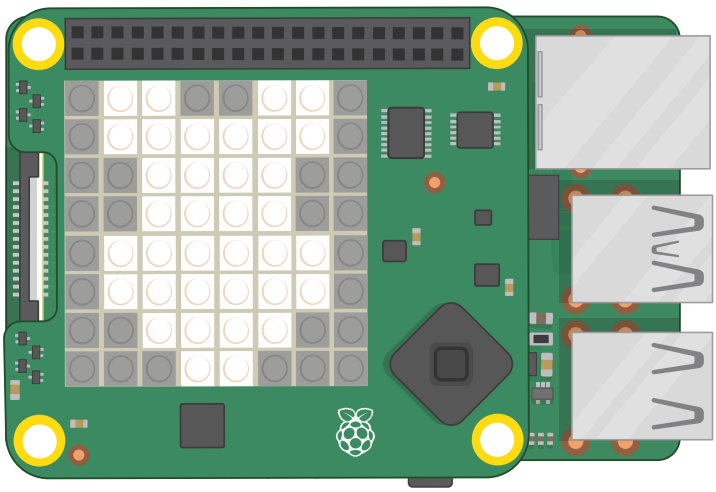
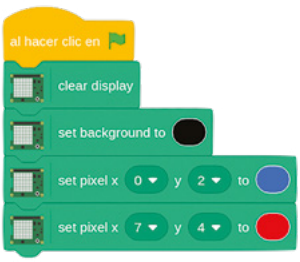


Figura 7-11: No mires directamente a los LED cuando estén iluminados en blanco brillante

La frambuesa predefinida no es la única posibilidad. Haz clic en la flecha abajo junto a la frambuesa para activar el modo de dibujo: puedes hacer clic en cualquier LED del patrón para encenderlo o apagarlo, mientras que los dos botones de la parte inferior sirven para apagar o encender todos los LED. Ahora dibuja tu propio patrón y luego haz clic en la flecha verde para verlo en Sense HAT. Cambia también el color del texto y el color de fondo usando los bloques de arriba.

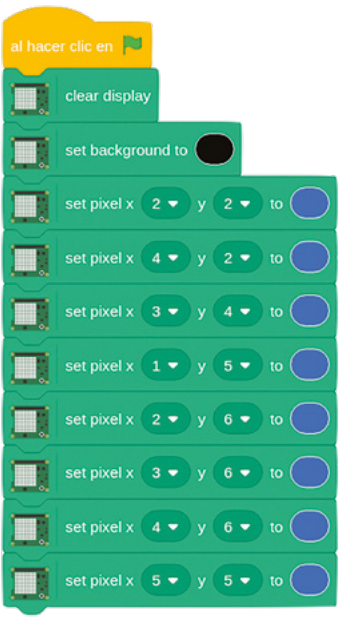
Cuando hayas terminado, arrastra los tres bloques a la paleta de bloques para borrarlos y coloca un bloque clear display debajo de al hacer clic en . Haz clic en la bandera verde y todos los LED se apagarán.

Para crear una imagen, debes ser capaz de controlar píxeles individuales y asignarles distintos colores. Puedes hacerlo encadenando bloques **display frambuesa** editados con bloques **set colour** o puedes gestionar cada píxel individualmente. Para crear tu propia versión del ejemplo de la matriz LED mostrada al principio de esta sección, con dos LED específicamente seleccionados iluminados en rojo y azul, deja el bloque **clear display** en la parte superior del programa y arrastra un bloque **set background** debajo de él. Cambia el bloque **set background** a negro y luego arrastra dos bloques **set pixel x 0 y 0** debajo de él. Finalmente, edita estos bloques de la siguiente manera:

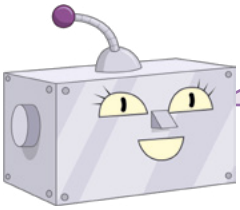


Haz clic en la bandera verde y verás que los LED se encienden igual que en la imagen de la matriz (**Figura 7-10**) en la página 165. Enhorabuena: ¡ya sabes controlar LED individuales!

Edita tus bloques de píxeles existentes como se indica a continuación y arrastra más a la parte inferior, hasta que tengas este programa:



Antes de hacer clic en la bandera verde, intenta adivinar qué imagen va a aparecer en base a las coordenadas de la matriz LED que has usado. Luego ejecuta el programa y comprueba si has acertado.



RETO: NUEVOS DISEÑOS

¿Puedes diseñar más imágenes? Consigue papel cuadriculado y úsalo para delinear tu imagen a mano primero. ¿Puedes crear un dibujo y hacer que cambien los colores?

Imágenes en Python

Inicia un programa nuevo en Thonny y guárdalo como Sense HAT Drawing, luego escribe lo siguiente (acuérdate de usar **`sense_emu`** en lugar de **`sense_hat`** si estás usando el emulador):

```
from sense_hat import SenseHat
sense = SenseHat()
```

Recuerda que necesitas ambas líneas de tu programa para poder usar Sense HAT. Luego escribe:

```
sense.clear(255, 255, 255)
```

Evita mirar directamente a los LED de Sense HAT y haz clic en el icono Run: todos los LED se iluminarán con luz blanca brillante (**Figura 7-12**), por esa razón no deberías mirarlos directamente al ejecutar tu programa.

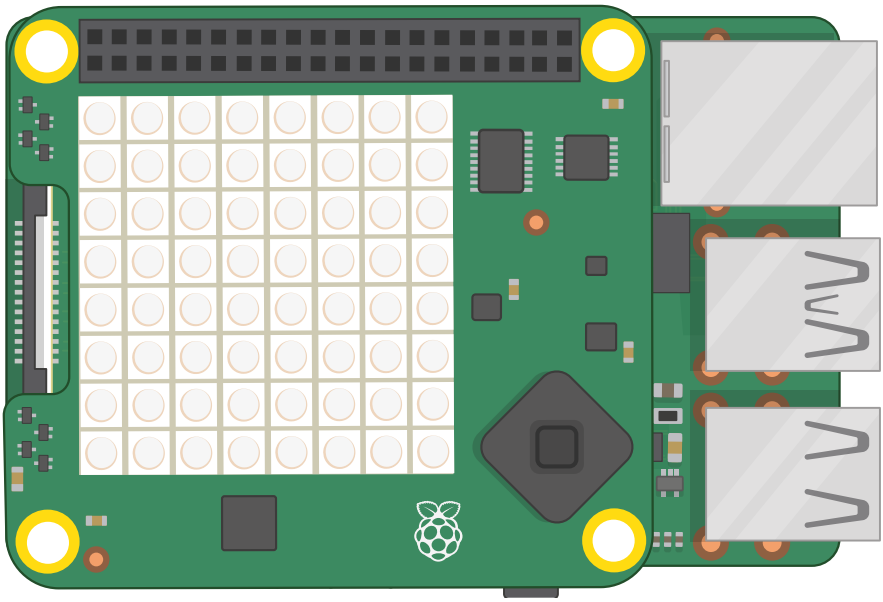


Figura 7-12: No mires directamente a la matriz cuando esté iluminada en blanco brillante

`sense.clear()` tiene la finalidad de borrar los LED de cualquier programación anterior, pero acepta parámetros de color RGB, lo que significa que puedes cambiar la pantalla a cualquier color que desees. Edita la línea a:

```
sense.clear(0, 255, 0)
```

Haz clic en Run y Sense HAT se volverá verde brillante (**Figura 7-13** a continuación). Experimenta con diferentes colores o añade las variables de nombre de color que creaste para tu programa Hello World anterior, para hacer las cosas más fáciles de leer.

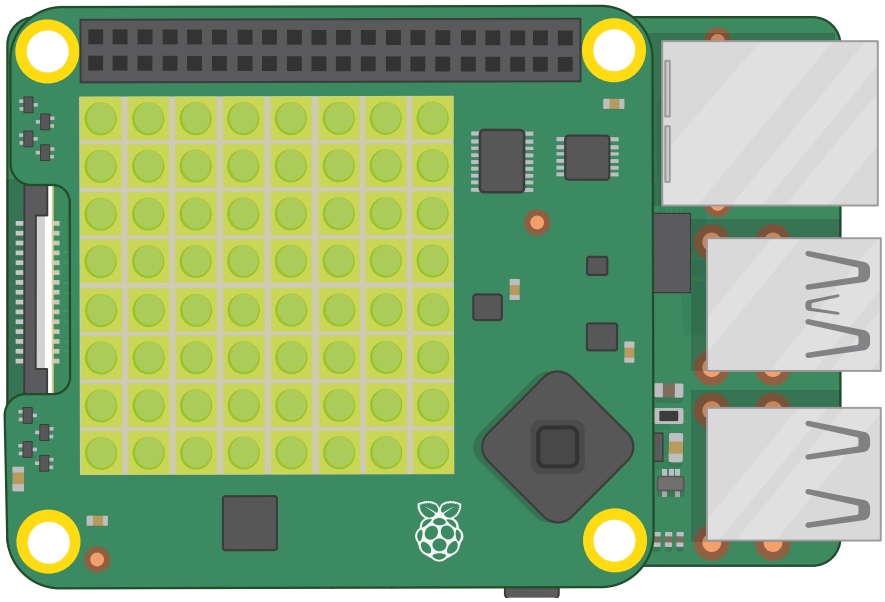


Figura 7-13: La matriz LED iluminada en verde brillante

Para borrar los LED, tendrás que usar los valores RGB para el negro: 0 rojo, 0 azul y 0 verde. Pero hay una manera más fácil. Edita la línea de tu programa para que diga:

```
sense.clear()
```

Sense HAT se oscurecerá. Si la función **`sense.clear()`** no tiene nada entre los paréntesis, lo interpreta como una instrucción para poner todos los LED en negro, o sea, que los apague (**Figura 7-14**). Cuando tengas que borrar todos los LED de tus programas, esa es la función que debes utilizar.

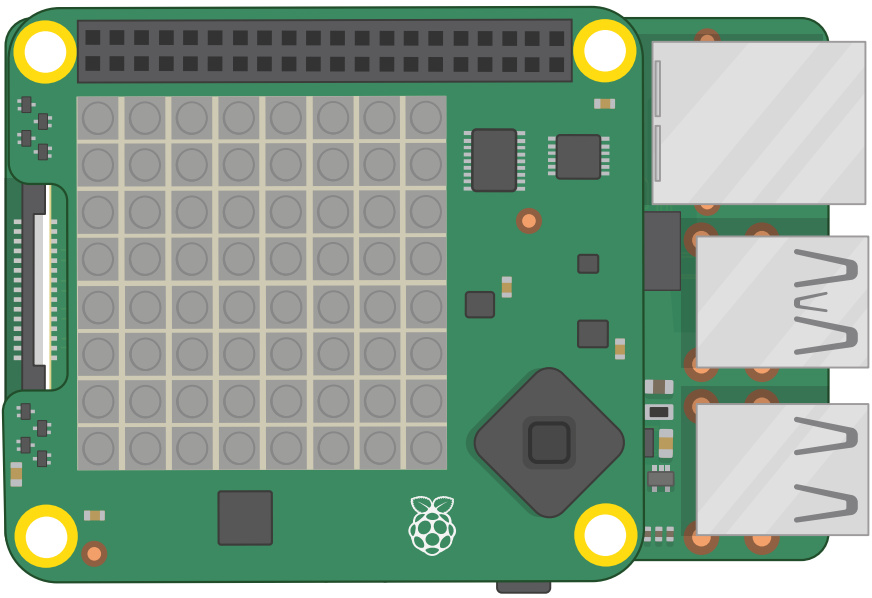


Figura 7-14: Usar la función `sense.clear` para apagar todos los LED

Para crear tu propia versión de la matriz LED mostrada anteriormente en este capítulo, con dos LED específicamente seleccionados iluminados en rojo y azul, añade las siguientes líneas a tu programa después de **`sense.clear()`**:

```
sense.set_pixel(0, 2, (0, 0, 255))
sense.set_pixel(7, 4, (255, 0, 0))
```

El primer par de números son la ubicación del píxel en la matriz, el eje X (transversal) seguido del eje Y (descendente). El segundo par, en su propio conjunto de paréntesis, son los valores RGB para el color de los píxeles. Haz clic en el botón Run y verás el efecto: se encenderán dos de los LED de Sense HAT, como en la **Figura 7-10** de la página 165.

Borra esas dos líneas y escribe lo siguiente:

```
sense.set_pixel(2, 2, (0, 0, 255))
sense.set_pixel(4, 2, (0, 0, 255))
sense.set_pixel(3, 4, (100, 0, 0))
sense.set_pixel(1, 5, (255, 0, 0))
sense.set_pixel(2, 6, (255, 0, 0))
sense.set_pixel(3, 6, (255, 0, 0))
sense.set_pixel(4, 6, (255, 0, 0))
sense.set_pixel(5, 5, (255, 0, 0))
```

Antes de pulsar Run, mira las coordenadas y compáralas con la matriz: ¿puedes adivinar qué imagen van a dibujar esas instrucciones? Haz clic en Run para ver si has acertado.

Crear un dibujo detallado utilizando **`set_pixel()`** es un proceso lento. Para acelerar las cosas, puedes cambiar varios píxeles al mismo tiempo. Borra todas las líneas **`set_pixel()`** y escribe lo siguiente:

```
g = (0, 255, 0)
b = (0, 0, 0)

creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, b, b, g, g, b, b, g,
    g, b, b, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]

sense.set_pixels(creeper_pixels)
```

Hay muchas cosas, pero empieza por hacer clic en Run para ver si reconoces a cierto creeper. Las dos primeras líneas crean dos variables para mantener los colores: verde y negro. Para que el código del dibujo sea más fácil de escribir y leer, las variables son letras simples: "g" para verde (green) y "b" para negro (black).

El siguiente bloque de código crea una variable que contiene los valores de color de los 64 píxeles de la matriz LED, separados por comas y encerrados entre corchetes. Pero, en lugar de números, utiliza las variables de color que creaste anteriormente. Fíjate, recuerda que "g" es para el verde y "b" para el negro y ya sabrás qué imagen va a aparecer (**Figura 7-15**).

Por último, **`sense.set_pixels(creeper_pixels)`** toma esa variable y utiliza la función **`sense.set_pixels()`** para dibujar en toda la matriz simultáneamente. ¡Mucho más fácil que dibujar píxel por píxel!

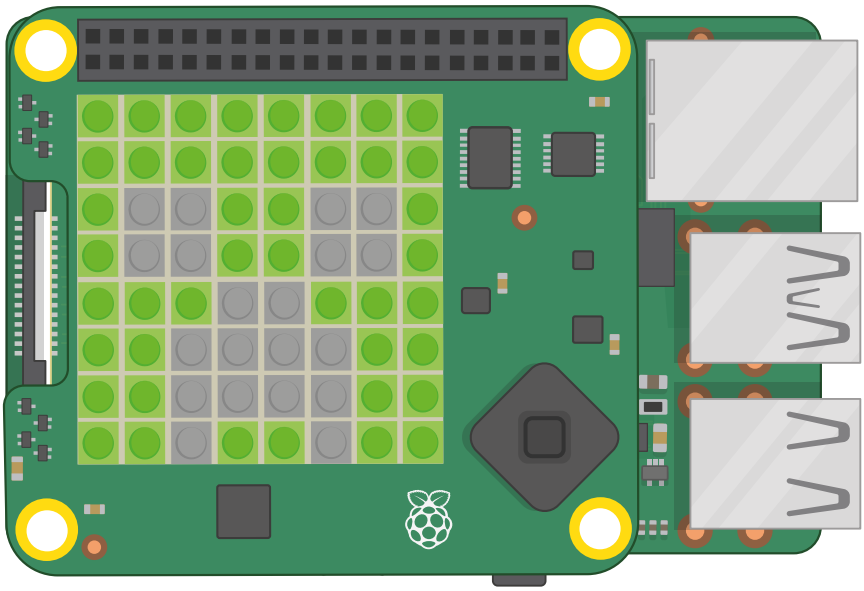


Figura 7-15: Mostrar una imagen en la matriz

También puedes rotar y voltear las imágenes, ya sea para mostrarlas de forma correcta al girar Sense HAT o como una forma de crear animaciones simples a partir de una sola imagen asimétrica.

Empieza editando tu variable **`creeper_pixels`** para cerrar el ojo izquierdo, reemplazando los cuatro píxeles "b", comenzando con los dos primeros en la tercera línea y luego los dos primeros en la cuarta línea, con "g":

```
creeper_pixels = [
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, g, g, g,
    g, g, g, g, g, b, b, g,
    g, g, g, g, g, b, b, g,
    g, g, g, b, b, g, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, b, b, b, g, g,
    g, g, b, g, g, b, g, g
]
```

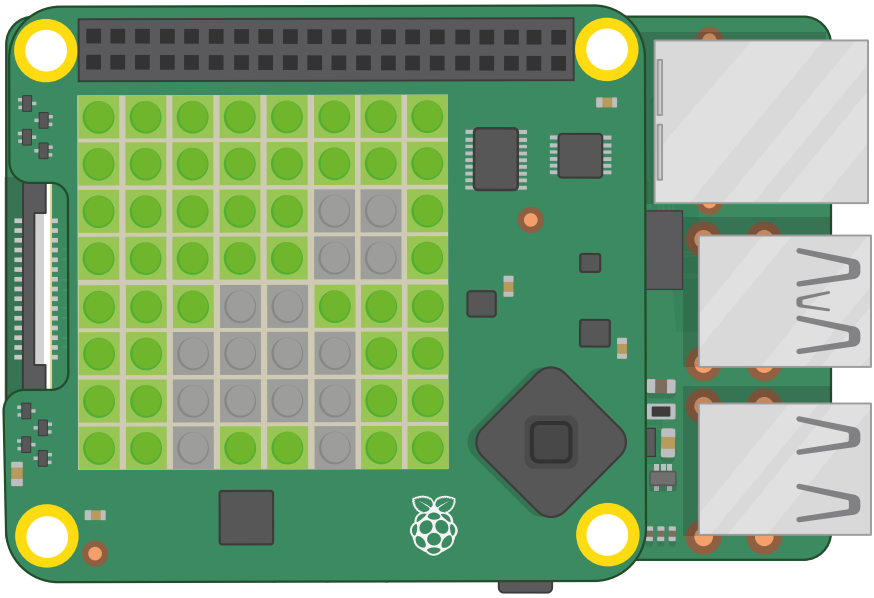
Haz clic en Run y verás que se cierra el ojo izquierdo (**Figura 7-16** a continuación). Para crear una animación, ve a la parte superior de tu programa y añade la línea:

```
from time import sleep
```

Luego ve al final y escribe:

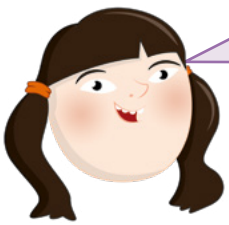
```
while True:
    sleep(1)
    sense.flip_h()
```

Haz clic en Run y observa qué hace el creeper mientras cierra y abre los ojos, primero uno y luego el otro.



▲ **Figura 7-16:** Mostrar una animación simple de dos cuadros

La función **`flip_h()`** voltea una imagen en el eje horizontal. Si quieres voltear una imagen en su eje vertical, utiliza la función **`sense.flip_h()`** con **`sense.flip_v()`**. También puedes rotar una imagen 0, 90, 180 o 270 grados usando **`sense.set_rotation(90)`** y cambiando el número según los grados que quieras rotarla. Prueba eso para que el creeper gire en lugar de guiñar los ojos.

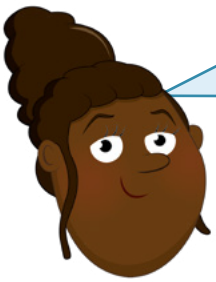


RETO: NUEVOS DISEÑOS

¿Puedes diseñar más imágenes y animaciones? Consigue papel cuadriculado y úsalo para delinear tu imagen a mano primero, para facilitarte la creación de la variable. ¿Puedes crear un dibujo y hacer que cambien los colores? Sugerencia: puedes cambiar las variables después de haberlas usado una vez.

Sensores de tu entorno

El verdadero poder de Sense HAT reside en sus diversos sensores. Esto le permite leer datos de cualquier cosa, desde la temperatura hasta la aceleración, para que los uses en tus programas como creas conveniente.



EMULAR LOS SENSORES

Si utilizas el emulador de Sense HAT, tendrás que habilitar la simulación de sensores inerciales y ambientales: en el emulador, haz clic en Edit, luego en Preferences y luego selecciónalas. En el mismo menú elige "180°..360°|0°..180°" en "Orientation Scale" para asegurarte de que los números del emulador coinciden con los números indicados por Scratch y Python. Luego haz clic en el botón de cierre.

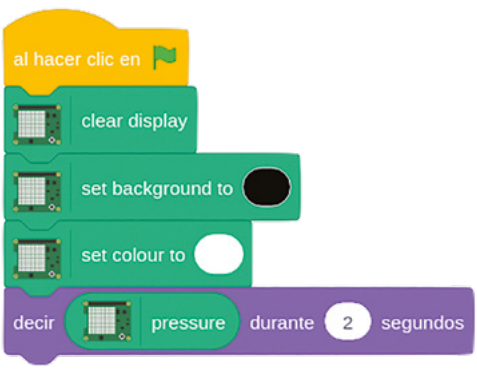
Sensores medioambientales

El sensor de presión barométrica, el de humedad y el de temperatura son sensores ambientales: realizan mediciones del entorno de Sense HAT.

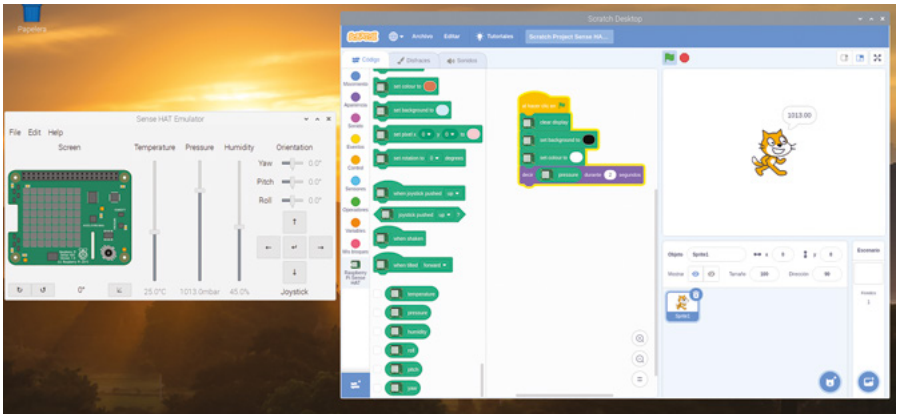
Sensores medioambientales en Scratch

Inicia un programa nuevo en Scratch (guardando el anterior, si quieres) y añade la extensión Sense HAT de Raspberry Pi si aún no está cargada. Arrastra un bloque de Eventos al hacer clic en a tu área de código y luego un bloque clear display debajo y uno set background to negro debajo del anterior. A continuación, añade un bloque set colour to blanco: utiliza los controles deslizantes de brillo y saturación para elegir el color correcto. Siempre conviene hacer esto al comienzo de tus programas, para asegurarte de que Sense HAT no muestre nada de un programa antiguo y cerciorarte de los colores que estás usando.

Arrastra un bloque de Apariencia decir ¡Hola! durante 2 segundos directamente debajo de los bloques existentes. Para hacer una lectura del sensor de presión, encuentra el bloque pressure en la categoría Sense HAT de Raspberry Pi y arrástralo sobre la palabra '¡Hola!' en tu bloque decir ¡Hola! durante 2 segundos.



Haz clic en la bandera verde y el gato de Scratch te indicará el valor leído por el sensor de presión en *milibares*. Al cabo de dos segundos, el mensaje desaparecerá. Sopla sobre Sense HAT (o sube el control de presión en el emulador) y haz clic en la bandera verde para ejecutar el programa de nuevo. Esta vez el valor de lectura visible debería ser más alto (**Figura 7-17** a continuación).

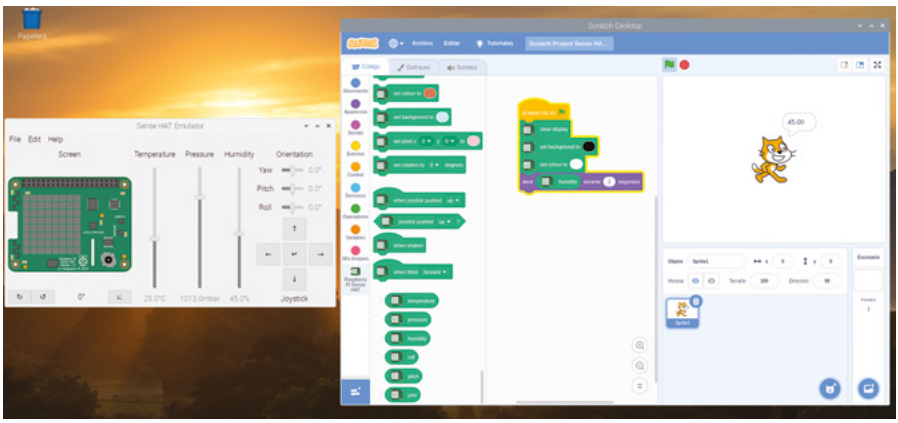


▲ **Figura 7-17:** Lectura del sensor de presión

CAMBIAR VALORES

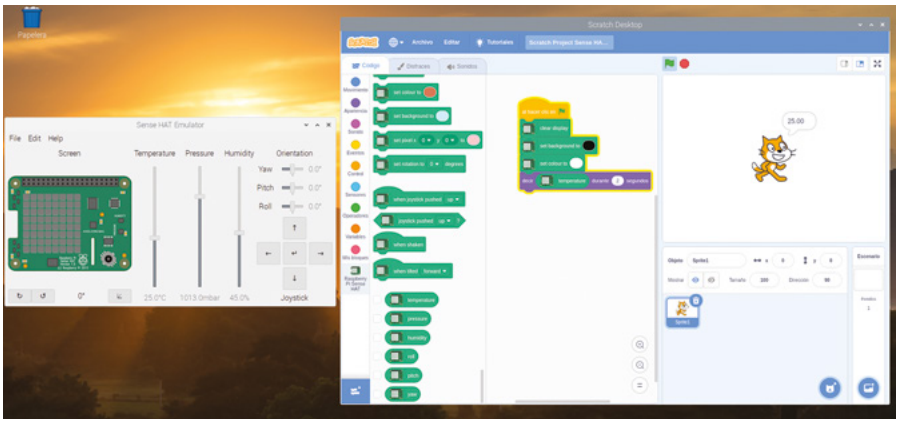
Si utilizas el emulador Sense HAT, puedes cambiar los valores indicados por cada uno de los sensores emulados utilizando sus controles deslizantes y botones. Desliza hacia abajo el sensor de presión y luego haz clic en la bandera verde de nuevo.

Para activar el sensor de humedad, borra el bloque `pressure` y sustitúyelo por `humidity`. Vuelve a ejecutar el programa y verás la humedad relativa actual de la habitación en la que estás. De nuevo, puedes intentar ejecutarlo de nuevo mientras soplas en Sense HAT (o subiendo el control deslizante de humedad en el emulador) para cambiar la lectura (**Figura 7-18**): te sorprenderá lo húmedo que es tu aliento.



▲ **Figura 7-18:** Lectura del sensor de humedad

Para usar el sensor de temperatura, basta con borrar el bloque humidity, sustituirlo por temperature y luego ejecutar el programa de nuevo. Verás una temperatura en grados centígrados (**Figura 7-19**). Pero puede que no sea la temperatura exacta de la habitación: Raspberry Pi genera calor mientras está funcionando y eso calienta la placa Sense HAT y también sus sensores.



▲ **Figura 7-19:** Lectura del sensor de temperatura

RETO: DESPLAZAMIENTO Y BUCLE

¿Puedes cambiar tu programa para que lea cada uno de los sensores por turno, y luego deslizar el resultado a través de la matriz LED en lugar de mostrarlos en el área de escenario? ¿Puedes hacer que tu programa haga un bucle, de modo que muestre constantemente las condiciones ambientales actuales?

Sensores medioambientales en Python

Para empezar a tomar lecturas de los sensores, crea un programa en Thonny y guárdalo como **Sense HAT Sensors**. Escribe lo siguiente en el área de script, como debes hacer cuando usas Sense HAT (y recuerda usar `sense_emu` si estás usando el emulador):

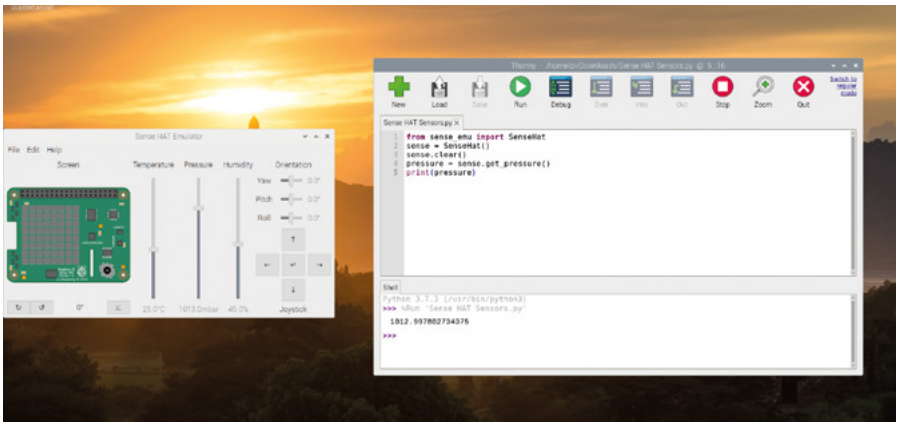
```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Siempre conviene incluir `sense.clear()` al comienzo de tus programas, por si la pantalla de Sense HAT siguiera mostrando algo del último programa ejecutado.

Para tomar una lectura del sensor de presión, escribe:

```
pressure = sense.get_pressure()
print(pressure)
```

Haz clic en Run y verás un número impreso en el shell de Python, en la parte inferior de la ventana de Thonny. Esta es la lectura de la presión del aire detectada por el sensor de presión barométrica, en *milibares* (**Figura 7-20**). Sopla en la placa Sense HAT (o sube el control de presión en el emulador) mientras vuelves a hacer clic en el icono Run. El número leído debería ser mayor esta vez.



▲ **Figura 7-20:** Mostrar una lectura de presión de Sense HAT

CAMBIAR VALORES

Si utilizas el emulador Sense HAT, puedes cambiar los valores indicados por cada uno de los sensores emulados utilizando sus controles deslizantes y botones. Desliza el sensor de presión hacia el botón y luego vuelve a hacer clic en Run.

Para activar el sensor de humedad, borra las dos últimas líneas del código y sustitúyelas por:

```
humidity = sense.get_humidity()
print(humidity)
```

Haz clic en Run y verás otro número impreso en el shell de Python: es la humedad relativa actual de la habitación, en forma de porcentaje. De nuevo, puedes soplar en Sense HAT (o subir el deslizador de humedad del emulador) y verás como aumenta cuando vuelvas a ejecutar tu programa (**Figura 7-21**): te sorprenderá lo húmedo que es tu aliento.

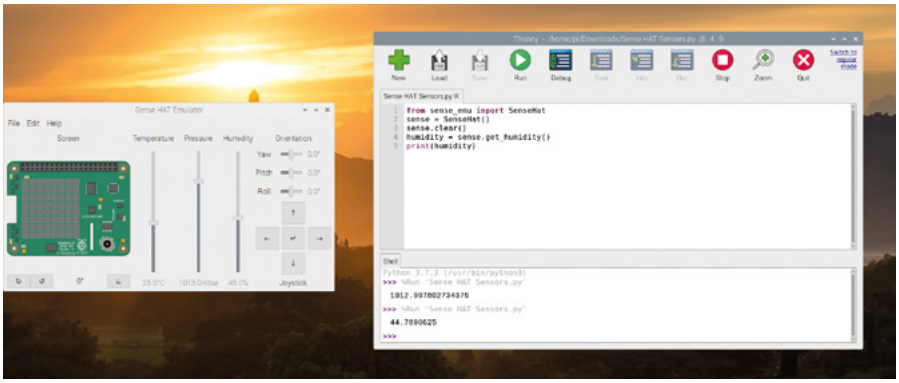


Figura 7-21: Lectura del sensor de temperatura

Para activar el sensor de temperatura, borra las dos últimas líneas del programa y sustitúyelas por:

```
temp = sense.get_temperature()
print(temp)
```

Vuelve a hacer clic en Run y verás una temperatura en grados centígrados (**Figura 7-22**). Pero puede que no sea la temperatura exacta de la habitación: Raspberry Pi genera calor mientras está funcionando y eso calienta la placa Sense HAT y también sus sensores.

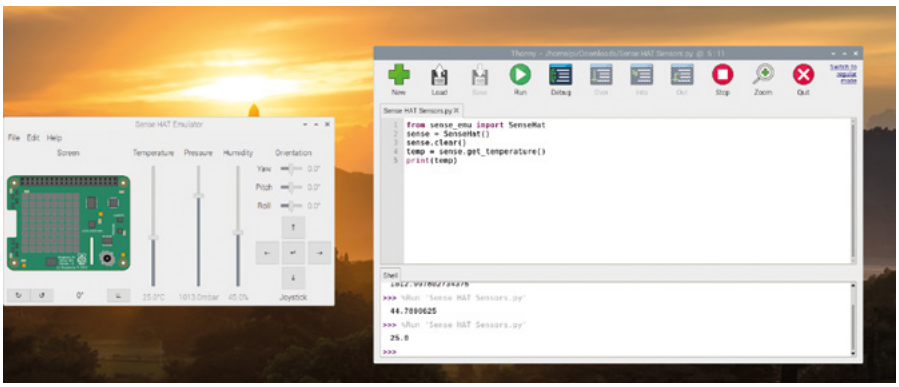


Figura 7-22: Lectura de la temperatura actual

Normalmente Sense HAT indica la temperatura basándose en una lectura del sensor de temperatura incorporado en el sensor de humedad. Si prefieres usar la lectura del sensor de presión, deberías usar **`sense.get_temperature_from_pressure()`**. También es posible combinar las dos lecturas para obtener un promedio, que puede ser más preciso que usar cualquiera de los sensores por separado. Borra las dos últimas líneas de tu programa y escribe:

```
htemp = sense.get_temperature()
ptemp = sense.get_temperature_from_pressure()
temp = (htemp + ptemp) / 2
print(temp)
```

Haz clic en el icono Run y verás un número mostrado en la consola de Python (**Figura 7-23**). Esta vez, se basa en las lecturas de ambos sensores, que has agregado y dividido entre dos (el número de lecturas) para obtener un promedio de ambos. Si estás usando el emulador, los tres métodos —humedad, presión y promedio— mostrarán el mismo número.

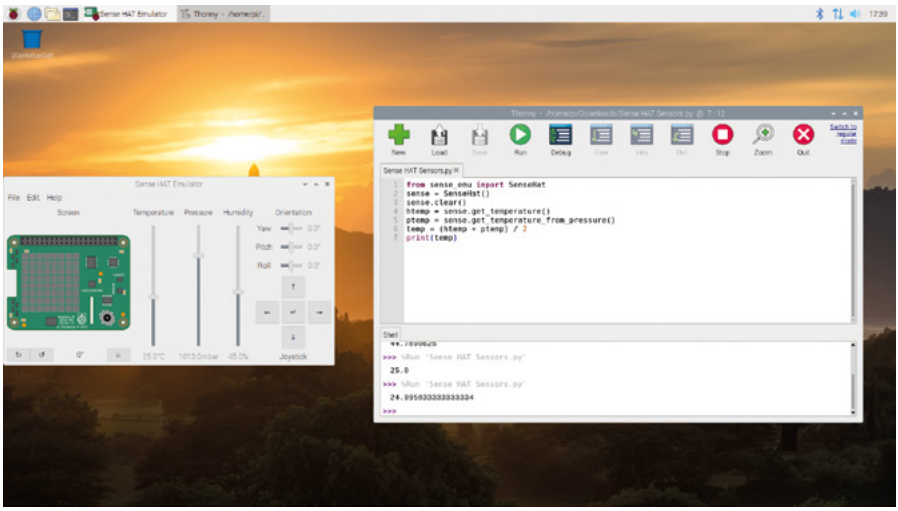


Figura 7-23: Temperatura basada en las lecturas de ambos sensores

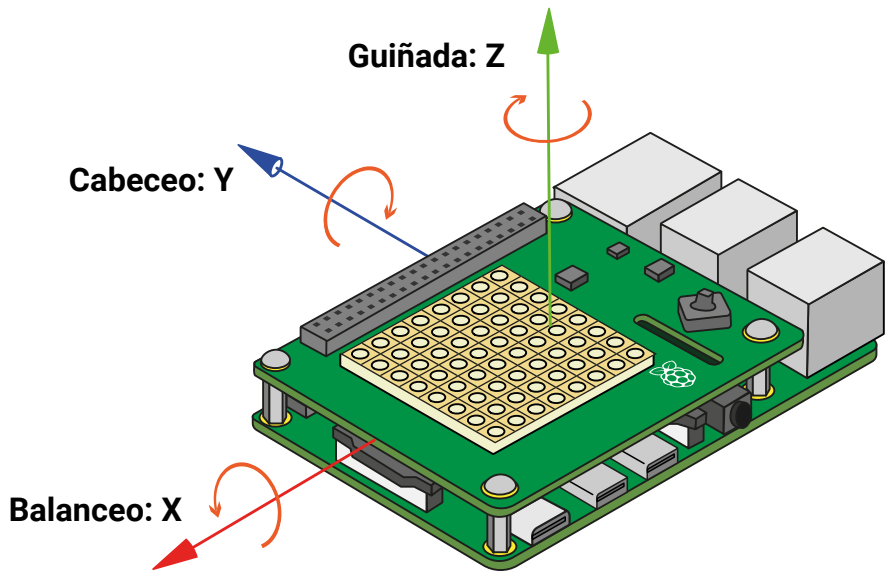
RETO: DESPLAZAMIENTO Y BUCLE

¿Puedes cambiar tu programa para que lea cada uno de los sensores por turno, y luego deslizar el resultado a través de la matriz LED en lugar de mostrarlos en el shell? ¿Puedes hacer que tu programa haga un bucle, de modo que muestre constantemente las condiciones ambientales actuales?

Sensores inerciales

El sensor giroscópico, el acelerómetro y el magnetómetro se combinan para formar lo que se denomina *unidad de medición inercial (IMU)*. Estos sensores realizan mediciones del ambiente circundante igual que lo hacen los sensores ambientales (el magnetómetro, por ejemplo, mide la fuerza del campo magnético) pero se suelen utilizar generalmente para obtener datos sobre el movimiento de la placa Sense HAT propiamente dicha. IMU es la suma de múltiples sensores; algunos lenguajes de programación permiten lecturas con cada sensor de forma independiente, mientras que otros solo ofrecen una lectura combinada.

Pero para entender la IMU, debes entender cómo se mueven las cosas. Sense HAT, así como el Raspberry Pi al que se ha acoplado, puede moverse a lo largo de tres ejes espaciales: de lado a lado en el eje X; adelante y atrás en el eje Y; y arriba y abajo en el eje Z (**Figura 7-24**). También puede rotar sobre esos tres ejes, pero la nomenclatura cambia: a la rotación en el eje X se le llama *balanceo*; a la rotación en el eje Y *cabeceo* y a la rotación en el eje Z *guiñada*. Cuando Sense HAT gira sobre su eje corto, se está ajustando su cabeceo; si la rotación es sobre el eje largo, es el balanceo; y si gira mientras se mantiene plano sobre la mesa, estás ajustando su guiñada. Imagínatelo como si fuera un avión: al despegar, aumenta su cabeceo para ascender; cuando hace un tonel de alerón, está girando sobre su eje de balanceo; y cuando se usa el timón para girar como lo haría un coche, sin balanceo, eso es una guiñada.



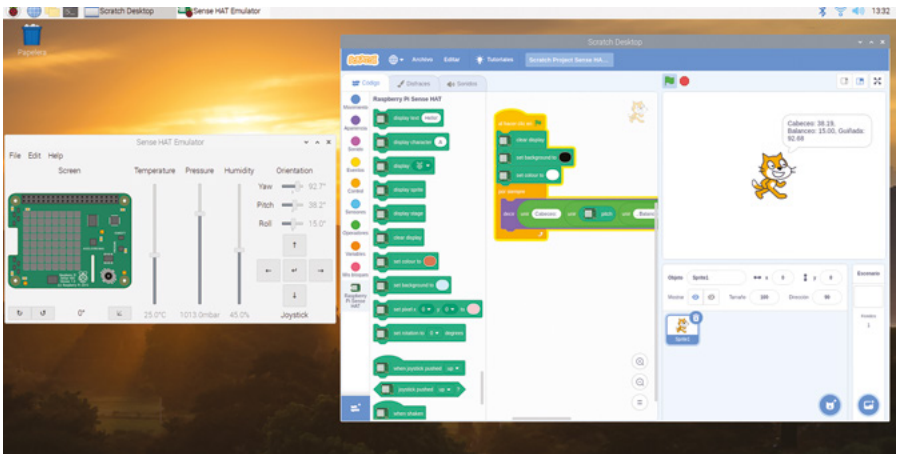
▲ **Figura 7-24:** Los ejes espaciales de IMU de Sense HAT

Sensores inerciales en Scratch

Inicia un nuevo programa en Scratch y carga la extensión Sense HAT de Raspberry Pi, si aún no está cargada. Empieza igual que antes: arrastra un bloque de Eventos al hacer clic en al área de código, luego arrastra un bloque clear display debajo de él, y luego arrastra y edita un bloque set background to negro y uno set colour to blanco.

A continuación, arrastra un bloque por siempre después de todos los bloques existentes y llénalo con un bloque decir ¡Hola!. Para mostrar una lectura de cada uno de los tres ejes de UMI (cabeceo, balanceo y guiñada) tendrás que añadir bloques de operador unir más los correspondientes bloques de Sense HAT. Acuérdate de incluir espacios y comas, para que el resultado sea fácil de leer.

Haz clic en la bandera verde para ejecutar el programa, y mueve Sense HAT y Raspberry Pi, teniendo cuidado de no soltar ningún cable. Al inclinar Sense HAT sobre sus tres ejes, verás que cambian los valores de cabeceo, balanceo y guiñada (**Figura 7-25**).



▲ **Figura 7-25:** Valores de cabeceo, balanceo y guiñada

Sensores inerciales en Python

Crea un programa nuevo en Thonny y guárdalo como **Sense HAT Movement**. Rellena las líneas de salida habituales, recordando usar `sense_emu` si estás usando el emulador de Sense HAT:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

Para usar la información de IMU para calcular la orientación actual del Sense HAT en sus tres ejes, escribe lo siguiente:

```
orientation = sense.get_orientation()
pitch = orientation["pitch"]
roll = orientation["roll"]
yaw = orientation["yaw"]
print("cabeceo {0} balanceo {1} guiñada {2}".format(pitch, roll, yaw))
```

Haz clic en Run y verás las lecturas de la orientación de Sense HAT divididas en los tres ejes (**Figura 7-26**). Haz rotar Sense HAT y haz clic en Run. Los números deberían cambiar para reflejar la nueva orientación.

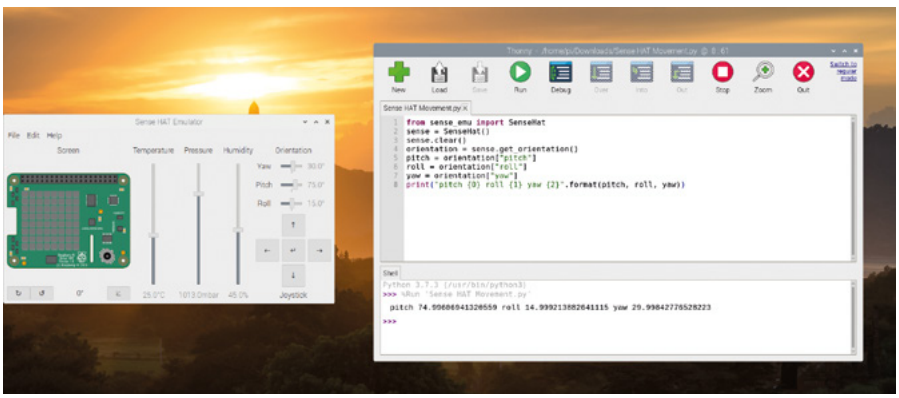


Figura 7-26: Valores de cabeceo, balanceo y guiñada de Sense HAT

IMU puede hacer más cosas aparte de medir la orientación: también puede detectar movimiento. Para obtener lecturas precisas del movimiento, se deben hacer lecturas de IMU frecuentes en un bucle: a diferencia de la orientación, una sola lectura no te dará ninguna información útil para detectar movimiento. Borra todo lo que hay después de `sense.clear()` y escribe el siguiente código:

```
while True:
    acceleration = sense.get_accelerometer_raw()
    x = acceleration["x"]
    y = acceleration["y"]
    z = acceleration["z"]
```

Ahora tienes variables que contienen las lecturas actuales del acelerómetro para los tres ejes espaciales: X, o izquierda y derecha; Y, o adelante y atrás; y Z, o arriba y abajo. Como los números del sensor del acelerómetro pueden ser difíciles de leer, escribe lo siguiente para facilitar su comprensión redondeándolos al número entero más cercano:

```
    x = round(x)
    y = round(y)
    z = round(z)
```

Finalmente, muestra los tres valores escribiendo la siguiente línea:

```
    print("x={0}, y={1}, z={2}".format(x, y, z))
```

Haz clic en Run y verás los valores del acelerómetro mostrados en el área de shell de Python (**Figura 7-27**). A diferencia del programa anterior, se mostrarán continuamente; para detenerlos, haz clic en el botón rojo para detener el programa.

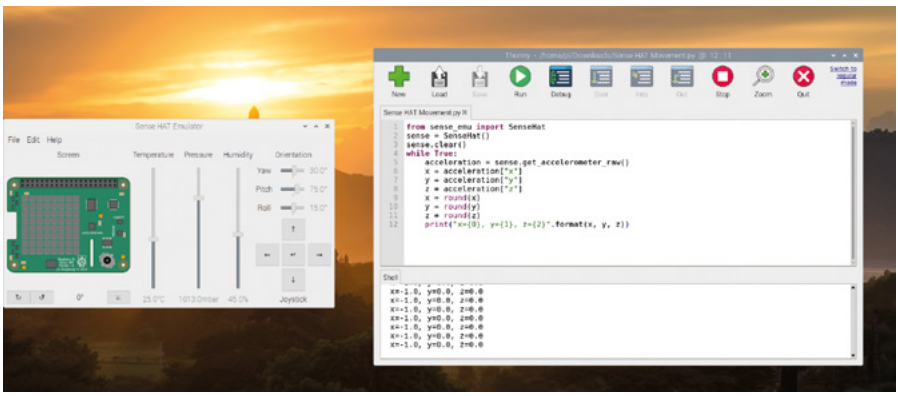


Figura 7-27: Las lecturas del acelerómetro se redondean al número entero más cercano

Habrás notado que el acelerómetro está indicando que uno de los ejes (el eje Z si Raspberry Pi está plano sobre la mesa) tiene un valor de aceleración de 1.0 (una fuerza de 1G), pero Sense HAT no se está moviendo. Eso se debe a que está detectando la atracción gravitatoria de la Tierra: la fuerza que "tira" de Sense HAT hacia el centro de la Tierra, y la razón por la que los objetos en nuestro escritorio caen al suelo si los empujamos.

Con tu programa en ejecución, coge cuidadosamente la placa Sense HAT y Raspberry Pi y gíralos con cuidado, para que no se suelte ninguno de sus cables. Con la red de Raspberry Pi y los puertos USB orientados hacia el suelo, verás que los valores cambian: ahora el eje Z muestra 0G y el eje X 1G. Luego orienta los puertos HDMI y de alimentación hacia el suelo y ahora es el eje Y el que muestra 1G. Si haces lo contrario y tienes el puerto HDMI hacia el techo, verás -1G en el eje Y.

Sabiendo que la fuerza de la gravedad de la Tierra es aproximadamente de 1G, y con tus conocimiento de los ejes espaciales, puedes usar las lecturas del acelerómetro para averiguar la orientación de descenso y también la de ascenso. También te servirán para detectar movimiento: zarandea levemente la placa Sense HAT y Raspberry Pi, y observa los números: cuanto más fuerte sea el zarandeo, mayor será la aceleración.

Al usar **`sense.get_accelerometer_raw()`** le estás diciendo a Sense HAT que apague los otros dos sensores de UMI, el giroscópico y el magnetómetro, y que proporcione únicamente los datos del acelerómetro. Naturalmente, también puedes hacer lo mismo con los otros sensores.

Encuentra la línea **`acceleration = sense.get_accelerometer_raw()`** y cámbiala a:

```
orientation = sense.get_gyroscope_raw()
```

Cambia la palabra **`acceleration`** en las tres líneas debajo de ella a **`orientation`**. Haz clic en Run y verás la orientación de Sense HAT para los tres ejes, redondeado al número entero más cercano. Pero, a diferencia de la última vez que comprobaste la orientación, ahora los datos proceden solo del giroscopio, sin usar el acelerómetro o el magnetómetro. Esto puede ser útil si quieres saber la orientación de una Sense HAT en movimiento colocada en la espalda de un robot, por ejemplo, sin que el movimiento confunda las cosas; o si estás usando una Sense HAT cerca de un campo magnético intenso.

Detén el programa haciendo clic en el botón rojo. Para usar el magnetómetro, borra todo lo que haya en tu programa, excepto las primeras cuatro líneas, y escribe lo siguiente debajo de la línea **`while True`**:

```
north = sense.get_compass()
print(north)
```

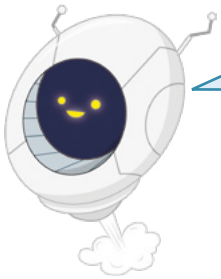
Ejecuta el programa y verás la dirección del norte magnético mostrada repetidamente en el área de shell de Python. Gira cuidadosamente la Sense HAT y verás que el rumbo cambia al cambiar la orientación de la placa con respecto al norte: has construido una brújula. Si tienes un imán (te servirá uno de nevera), muévelo alrededor de Sense HAT para ver el efecto en las lecturas del magnetómetro.

RETO: ROTACIÓN AUTOMÁTICA

Basándote en lo que has aprendido sobre la matriz LED y los sensores de la unidad de medición inercial, ¿puedes escribir un programa que haga girar una imagen dependiendo de la posición de Sense HAT?

Control con el joystick

El joystick de Sense HAT, situado en la esquina inferior derecha, es pequeño pero sorprendentemente eficaz: además de ser capaz de reconocer entradas en cuatro direcciones —arriba, abajo, izquierda y derecha— también tiene una quinta entrada, accionada a presión, como si fuera un conmutador.



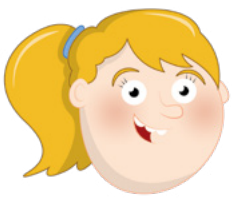
¡ADVERTENCIA!

El joystick de Sense HAT solo debe usarse si has colocado los espaciadores como se describe al principio de este capítulo. Sin los espaciadores, cabe el riesgo de que al empujar hacia abajo el joystick se doble la placa Sense HAT y se dañe tanto la placa como el sistema GPIO de Raspberry Pi.

Control con el joystick en Scratch

Inicia un nuevo programa en Scratch, con la extensión Sense HAT cargada. Igual que antes, arrastra un bloque de Eventos al hacer clic en al área de script, luego arrastra un bloque clear display debajo de él y luego arrastra y edita un bloque set background to negro y uno set colour to blanco.

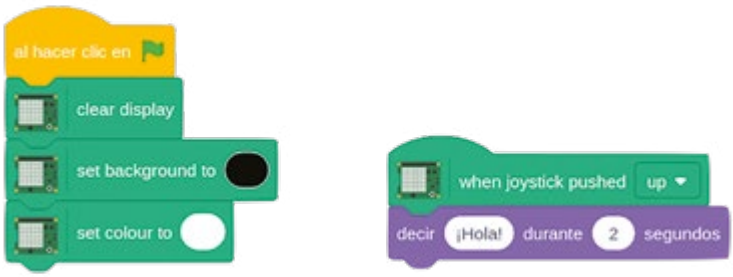
En Scratch, el joystick de Sense HAT se asigna a las teclas de cursor del teclado: empujar el joystick hacia arriba equivale a pulsar la tecla de flecha arriba; empujarlo hacia abajo equivale a pulsar la tecla de flecha abajo; empujarlo hacia la izquierda equivale a la tecla de flecha izquierda; y empujarlo hacia la derecha equivale a la tecla de flecha derecha. Por último, empujar el joystick hacia dentro, como un conmutador, equivale a pulsar la tecla **ENTRAR**.



¡ADVERTENCIA!

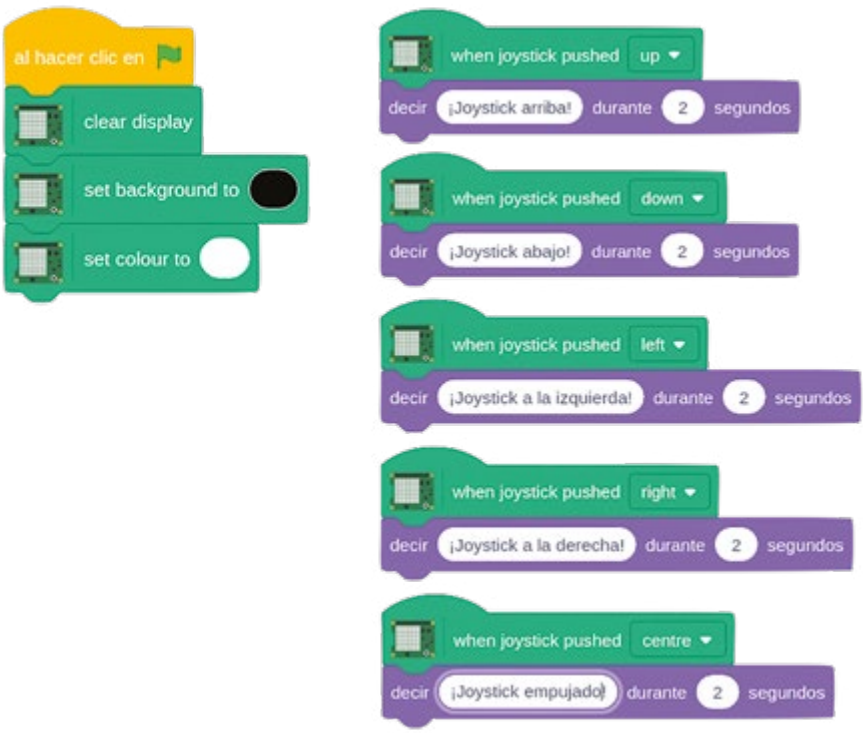
El control mediante joystick solo está disponible con la placa Sense HAT física. Al usar el emulador de Sense HAT, usa las teclas correspondientes de tu teclado para simular las acciones del joystick.

Arrastra un bloque **when joystick pushed up** a tu área de código. Para que haga algo, arrastra un bloque **decir ¡Hola! durante 2 segundos** debajo de él.

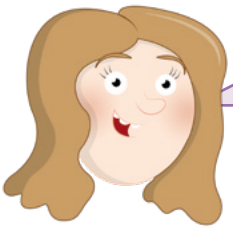


Empuja el joystick hacia arriba y verás que el gato de Scratch dice "¡Hola!"

A continuación, edita el bloque **decir ¡Hola! durante 2 segundos** en un bloque **decir ¡Joystick arriba! durante 2 segundos** y sigue añadiendo bloques de Eventos y Apariencia hasta que tengas algo para cada una de las cinco formas en que se puede accionar el joystick.



Empuja el joystick en varias direcciones para ver aparecer tus mensajes.



RETO FINAL

¿Puedes usar el joystick de Sense HAT para controlar un objeto de Scratch en el área de escenario? ¿Puedes hacer que si el objeto recoge otro objeto, que representa a su vez a un objeto, los LED de Sense HAT muestren un mensaje jovial?

Control con el joystick en Python

Crea un programa nuevo en Thonny y guárdalo como Sense HAT Joystick. Comienza con las tres líneas habituales que configuran Sense HAT y borra la matriz LED:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

A continuación, define un bucle infinito:

```
while True:
```

Luego dile a Python que capte las entradas del joystick de Sense HAT con la siguiente línea, a la que Thonny aplicará sangrado automáticamente:

```
    for event in sense.stick.get_events():
```

Por último, añade la siguiente línea (que también sangrará Thonny) para que realmente ocurra algo cuando se detecte una pulsación del joystick:

```
        print(event.direction, event.action)
```

Haz clic en Run y mueve el joystick en varias direcciones. Verás la dirección que has elegido mostrada en el área de shell de Python: arriba, abajo, izquierda, derecha y en el medio cuando hayas presionado el joystick como un conmutador.

También verás que se te dan dos eventos cada vez que empujas el joystick una vez: un evento, **`pressed`** cuando se empuja por primera vez en una dirección; y uno **`released`** cuando el joystick vuelve al centro. Puedes usar esto en tus programas, por ejemplo, si tienes un personaje en un juego, podrías hacer que empezara a moverse al accionar el joystick en una dirección y que se detuviera al soltar el joystick.

También puedes usar el joystick para activar funciones, sin la limitación de tener que usar un bucle "durante". Borra todo lo que hay debajo de **`sense.clear()`** y escribe lo siguiente:

```
def rojo():
    sense.clear(255, 0, 0)

def azul():
    sense.clear(0, 0, 255)

def verde():
    sense.clear(0, 255, 0)

def amarillo():
    sense.clear(255, 255, 0)
```

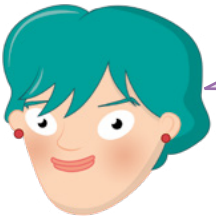
Estas funciones cambian toda la matriz LED de Sense HAT a un solo color: rojo, azul, verde o amarillo. Así será facilísimo comprobar si el programa funciona. Para activarlas, hay que decirle a Python qué función corresponde a cada acción de joystick. Escribe las siguientes líneas:

```
sense.stick.direction_up = rojo
sense.stick.direction_down = azul
sense.stick.direction_left = verde
sense.stick.direction_right = amarillo
sense.stick.direction_middle = sense.clear
```

Para terminar, el programa necesita un bucle infinito, conocido como el *principal* para seguir en ejecución, y por lo tanto, vigilar las acciones de los joysticks, en lugar de simplemente ejecutarse con el código que has escrito y cerrarse. Escribe estas dos líneas:

```
while True:
    pass
```

Haz clic en Run y mueve el joystick: verás que los LED se iluminan a todo color. Para apagar los LED, pulsa el joystick como si fuera un botón: la dirección **`middle`** (medio) se establece para utilizar la función **`sense.clear()`** de apagado de todos los LED. Enhorabuena: ya puedes capturar la acción del joystick.



RETO FINAL

¿Puedes usar lo que has aprendido para dibujar una imagen en la pantalla y luego hacerla girar en cualquier dirección en la que se accione el joystick? ¿Puedes hacer que la entrada del medio alterne varias imágenes?

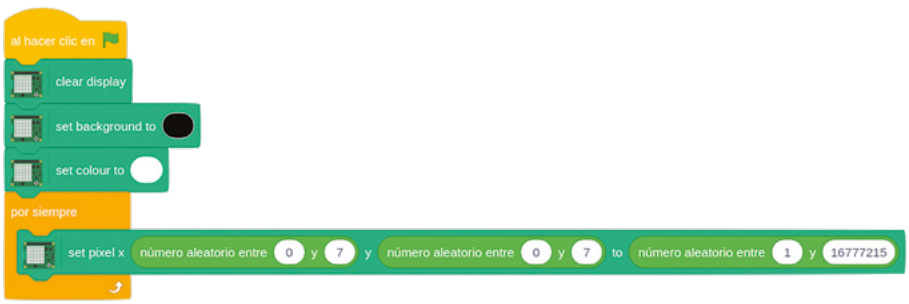
Proyecto de Scratch: Bengala de Sense HAT

Ahora que sabes cómo funciona Sense HAT, es hora de aplicar todo lo que has aprendido para crear una bengala sensible al calor, un dispositivo que prefiere el frío y se ralentiza gradualmente cuanto más calor hace.

Inicia un proyecto de Scratch y añade la extensión Sense HAT de Raspberry Pi si aún no está cargada. Como siempre, empieza con cuatro bloques: al hacer clic en , clear display , set background to negro y set colour to blanco, y ten en cuenta que tendrás que cambiar los colores de la configuración predeterminada.

Para empezar crea una sencilla, pero artística, bengala. Arrastra un bloque por siempre al área de código y llénalo con un bloque set pixel x 0 y 0 to color. En lugar de usar números fijos, rellena cada una de las secciones x e y, y la de color, de ese bloque con un bloque de operadores número aleatorio entre 1 y 10.

Los valores del 1 al 10 no son muy útiles en este caso, así que necesitas editarlos. Los dos primeros números del bloque set pixel son las coordenadas X e Y del píxel en la matriz LED. Eso significa que deben ser números entre 0 y 7: cambia los dos primeros bloques para que digan número aleatorio entre 0 y 7. La siguiente sección es el color que definir para el píxel. Cuando usas el selector de color, el color que eliges se muestra directamente en el área de script. Pero, internamente, los colores están representados por un número y puedes usar ese número directamente. Edita el último bloque aleatorio para que diga número aleatorio entre 0 y 16777215.



Haz clic en la bandera verde y verás que los LED de Sense HAT empiezan a encenderse en colores aleatorios (**Figura 7-28**). Enhorabuena: has creado una bengala electrónica.

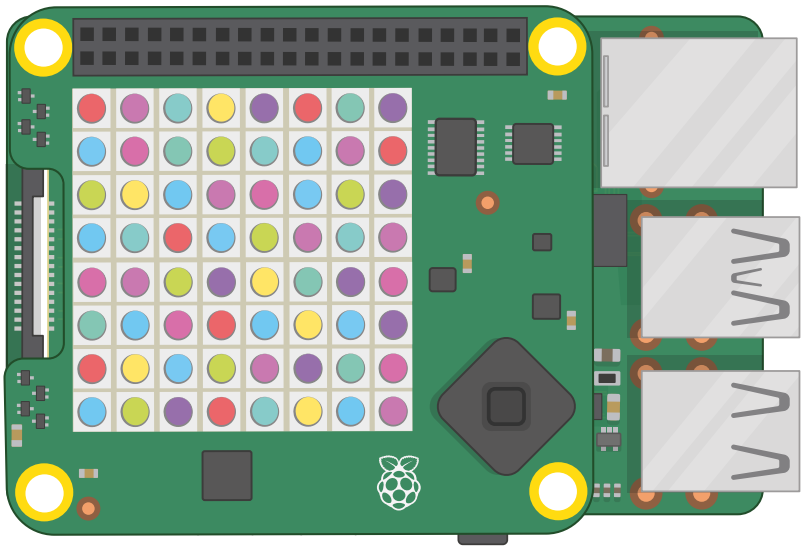


Figura 7-28: Píxeles iluminados en colores aleatorios

De momento, la bengala no es muy interactiva. Para cambiar eso, arrastra un bloque esperar 1 segundos y colócalo debajo del bloque set pixel pero dentro del bloque por siempre. Arrastra un bloque de Operadores ● / ● sobre el 1 y escribe 10 en el segundo espacio. Por último, arrastra un bloque temperature sobre el primer espacio en el bloque Operador de división.

Haz clic en la bandera verde y —a menos que vivas en un sitio muy frío— verás que la chispa es bastante más lenta que antes. Eso es porque has creado un retardo dependiente de la temperatura: ahora, antes de cada bucle, el programa espera un número de segundos igual a *la temperatura actual dividida por 10*. Si la temperatura de la habitación en la que estás es de 20°C, el programa esperará 2 segundos antes de hacer un bucle. Si la temperatura es de 10°C, esperará 1 segundo. Si es inferior a 10°C, esperará menos de un segundo.

Si tu Sense HAT lee una temperatura negativa (por debajo de 0°C, el punto de congelación del agua) intentará esperar menos de 0 segundos. Pero como eso es imposible, a menos que inventes el viaje en el tiempo, verás el mismo efecto que si esperas 0 segundos. Enhorabuena: ya puedes ir pensando en la integración de las diversas funciones de Sense HAT en tus propios programas.

Proyecto de Python: Tricorder de Sense HAT

Ahora que ya sabes cómo funciona la Sense HAT, es hora de aplicar todo lo que has aprendido para crear un tricorder, un dispositivo que como ya sabrán muchos seguidores de cierta serie de ciencia-ficción suministra información captada por varios sensores incorporados.

Inicia un proyecto en Thonny y guárdalo como **Tricorder**. Empieza con las líneas habituales necesarias para crear un programa de Sense HAT:

```
from sense_hat import SenseHat
sense = SenseHat()
sense.clear()
```

A continuación, debes definir las funciones de cada uno de los diversos sensores de Sense HAT. Empieza con la unidad de medición inercial, escribe:

```
def orientation():
    orientation = sense.get_orientation()
    pitch = orientation["pitch"]
    roll = orientation["roll"]
    yaw = orientation["yaw"]
```

Como los resultados del sensor se van a mostrar desplazándolos en los LED, tiene sentido redondearlos para que no aparezcan docenas de decimales. Pero, en lugar de números enteros, redondéalos a un decimal escribiendo lo siguiente:

```
    pitch = round(pitch, 1)
    roll = round(roll, 1)
    yaw = round(yaw, 1)
```

Por último, hay que decirle a Python que desplace los resultados a los LED, para que el tricorder funcione como un dispositivo de mano sin necesidad de conectarlo a un monitor o un televisor:

```
    sense.show_message("Cabeceo {0}, Balanceo {1}, Guiñada {2}".
format(pitch, roll, yaw))
```

Ahora que tienes una función completa para leer y mostrar la orientación con IMU, debes crear funciones similares para cada uno de los otros sensores. Empieza con el sensor de temperatura:

```
def temperature():
    temp = sense.get_temperature()
    temp = round(temp, 1)
    sense.show_message("Temperatura: %s grados centígrados" % temp)
```

Mira la línea que muestra el resultado a los LED: **%s** es lo que se denomina un marcador de posición y se sustituye por el contenido de la variable **temp**. Así puedes formatear bien el resultado con una etiqueta "Temperature:" y una unidad de medida, "degrees Celsius", que harán tu programa mucho más asequible.

A continuación, define una función para el sensor de humedad:

```
def humidity():
    humidity = sense.get_humidity()
    humidity = round(humidity, 1)
    sense.show_message("Humedad: %s por ciento" % humidity)
```

Luego ocúpate del sensor de presión:

```
def pressure():
    pressure = sense.get_pressure()
    pressure = round(pressure, 1)
    sense.show_message("Presión: %s milibares" % pressure)
```

Y finalmente, la lectura de la brújula del magnetómetro:

```
def compass():
    for i in range(0, 10):
        north = sense.get_compass()
    north = round(north, 1)
    sense.show_message("Norte: %s grados" % north)
```

El bucle **for** corto en esta función toma diez lecturas del magnetómetro para asegurarse de que hay datos suficientes para darte un resultado preciso. Si ves que el valor notificado cambia continuamente, amplíalo a 20, 30 o incluso 100 bucles para mejorar aún más la precisión.

Tu programa tiene ahora cinco funciones, cada una de las cuales toma una lectura de uno de los sensores de Sense HAT y las muestra desplazándolas en los LED. Pero necesita una forma de elegir qué sensor usar y el joystick es perfecto para eso.

Escribe lo siguiente:

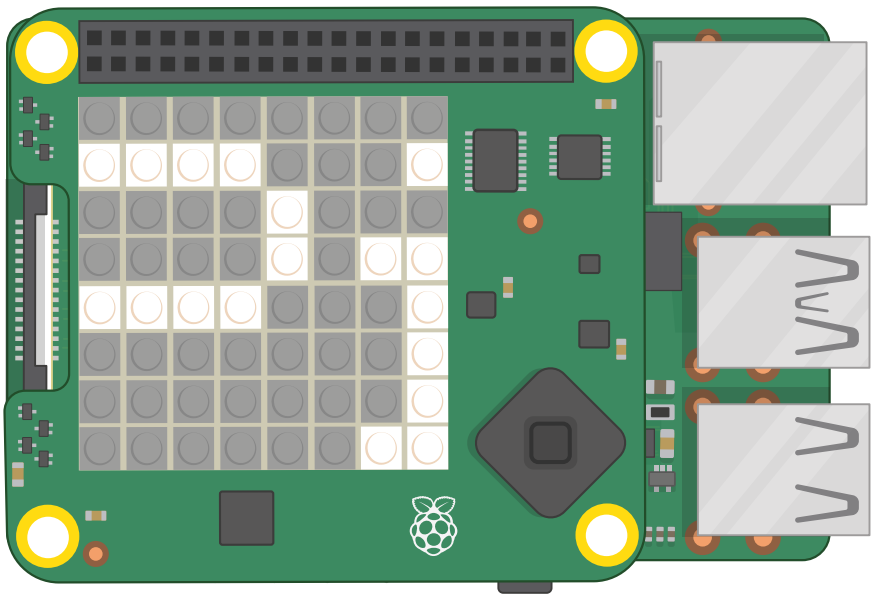
```
sense.stick.direction_up = orientation
sense.stick.direction_right = temperature
sense.stick.direction_down = compass
sense.stick.direction_left = humidity
sense.stick.direction_middle = pressure
```

Estas líneas asignan un sensor a cada una de las cinco direcciones posibles del joystick: arriba lee el sensor de orientación; abajo lee el magnetómetro; izquierda lee el sensor de humedad; derecha el sensor de temperatura; y la pulsación del centro del joystick lee el sensor de presión.

Finalmente, necesitas un bucle principal para que el programa siga captando las acciones del joystick y no se detenga inmediatamente. En la parte inferior de tu programa, escribe lo siguiente:

```
while True:
    pass
```

Haz clic en Run y mueve el joystick para que lea uno de los sensores (**Figura 7-29**). Cuando termine de mostrar el resultado, acciona en otra dirección. Enhorabuena: has creado un tricorder de mano digno de la aprobación de la Federación Unida de Planetas.



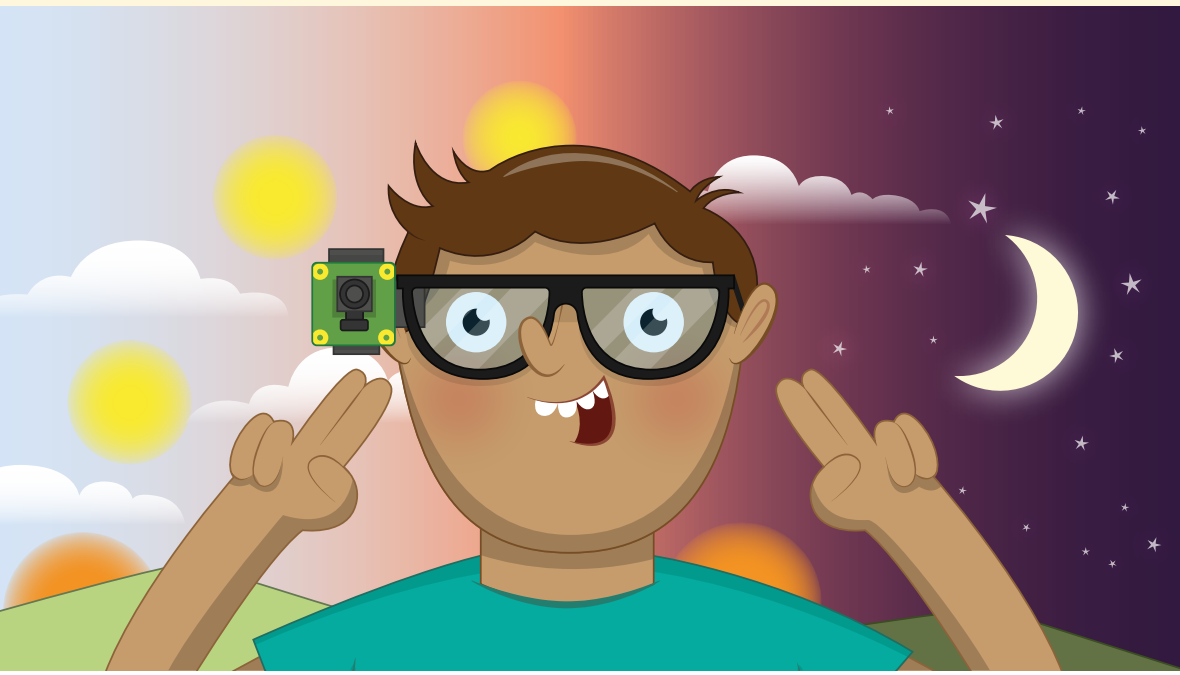
▲ **Figura 7-29:** Cada lectura se desplaza a través de la pantalla

Para ver más proyectos de Sense HAT, sigue los enlaces en el **Apéndice D, Otro material de referencia**.

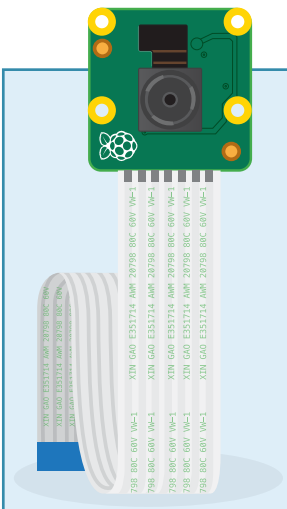
Capítulo 8

Cámara de Raspberry Pi

Al conectar un Camera Module (módulo de cámara) a Raspberry Pi podrás hacer fotos de alta resolución y grabar vídeos, y crear increíbles proyectos



Si alguna vez has querido construir algo capaz de ver, lo que en robótica se denomina *visión artificial*, el Camera Module (módulo de cámara) opcional de Raspberry Pi o la nueva High Quality Camera (cámara de alta calidad) son un gran punto de partida. El Camera Module / HQ Camera es una pequeña placa de circuito cuadrada con un cable plano fino que se conecta al puerto CSI (interfaz serie de cámara) en Raspberry Pi (no disponible en Raspberry Pi 400) y proporciona imágenes fijas de alta resolución y señales de vídeo en movimiento que puede usar tal cual o integrar en tus propios programas.



TIPOS DE CÁMARA

Hay tres tipos de cámara Raspberry Pi: el Camera Module estándar, la versión NoIR y la High Quality (HQ) Camera. Para hacer fotos y vídeos normales en entornos bien iluminados, debes usar el Camera Module estándar. Si quieres usar lentes especiales y obtener una calidad de imagen óptima, usa el HQ Camera. El Camera Module NoIR (que no tiene filtro de infrarrojos o IR) está diseñado para el uso con fuentes de luz infrarroja, para hacer fotos y grabar vídeo en total oscuridad. Si trabajas en un proyecto como por ejemplo una cámara para monitorizar un nido, una cámara de seguridad u otro uso que implique visión nocturna, necesitas la versión NoIR. Pero debes comprar también una fuente de luz infrarroja.

Los Camera Modules de Raspberry Pi estándar y No IR se basan en un sensor de imagen Sony IMX219 de *8 megapíxeles*, lo que significa que puede hacer fotos con hasta 8 millones de píxeles. Lo hace capturando imágenes de hasta 3280 píxeles de ancho por 2464 píxeles de alto. Además de imágenes fijas, el Camera Module puede capturar imágenes de vídeo con resolución Full HD a una velocidad de 30 cuadros por segundo (30 fps). Para obtener un movimiento más fluido o incluso para crear un efecto de cámara lenta, la cámara puede configurarse para capturar a una velocidad de cuadro más alta reduciendo la resolución: 60 fps para imágenes de vídeo de 720p, y hasta 90 fps para imágenes de 480p (VGA).

La High Quality Camera utiliza un sensor Sony IMX477 de 12,3 megapíxeles, que también es más grande que el de los Camera Modules estándar y NoIR, por lo que puede captar más luz y eso permite obtener imágenes de mayor calidad. A diferencia de los Camera Modules, la HQ Camera no incluye una lente, sin la cual no puede hacer fotos ni grabar vídeos. Puedes usar cualquier lente con una montura C o CS. Se pueden usar otras monturas con un adaptador C o CS apropiado. Los detalles de colocación de una lente se explican en el **Apéndice F: Configuración de la High Quality Camera**.

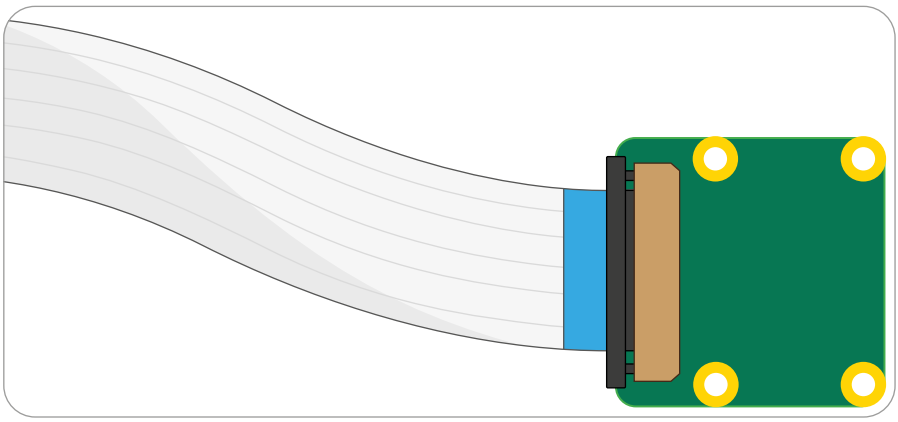
RASPBERRY PI 400

Desafortunadamente, las cámaras de Raspberry Pi no son compatibles con Raspberry Pi 400. Puedes usar cámaras web USB como alternativa, pero no podrás usar las herramientas de software específicas de las cámaras de Raspberry Pi incluidas en el sistema operativo Raspberry Pi OS.

Instalar la cámara

Como cualquier complemento de hardware, el Camera Module o la HQ Camera solo deben conectarse o desconectarse de Raspberry Pi cuando no tenga alimentación y el cable de corriente esté desenchufado. Si tu Raspberry Pi está encendido, elige Shutdown en el menú de Raspberry, espera a que se apague y desenchufa el cable.

En la mayoría de los casos, el cable plano suministrado ya estará conectado al Camera Module o a la HQ Camera. De no ser así, pon la placa de la cámara boca abajo de modo que el sensor quede en la parte inferior y busca un conector de plástico plano. Sujeta los bordes salientes y tira hacia fuera para sacarlo parcialmente. Desliza el cable plano, con los bordes plateados hacia abajo y el plástico azul hacia arriba, por debajo del conector que acabas de sacar y luego vuelve a empujar este hasta que oigas un clic (**Figura 8-1**). Puedes usar cualquiera de los extremos del cable. Si el cable está bien colocado, estará recto y no se saldrá al tirar de él suavemente. De lo contrario, saca el conector e inténtalo de nuevo.



▲ **Figura 8-1:** Conectar el cable plano al módulo de cámara

Instala el otro extremo del cable de la misma manera. Localiza el puerto de la cámara (o CSI) en Raspberry Pi y tira del conector suavemente. Si tu Raspberry Pi está instalado en una carcasa, tal vez sea conveniente sacarlo antes de empezar. Con Raspberry Pi posicionado con el puerto HDMI orientado hacia ti, desliza el cable plano de forma que los bordes plateados estén a tu izquierda y el plástico azul a tu derecha (**Figura 8-2**) y luego pulsa suavemente el conector para que vuelva a encajar en su sitio. Si el cable está bien colocado, estará recto y no se saldrá al tirar de él suavemente. De lo contrario, saca el conector e inténtalo de nuevo.

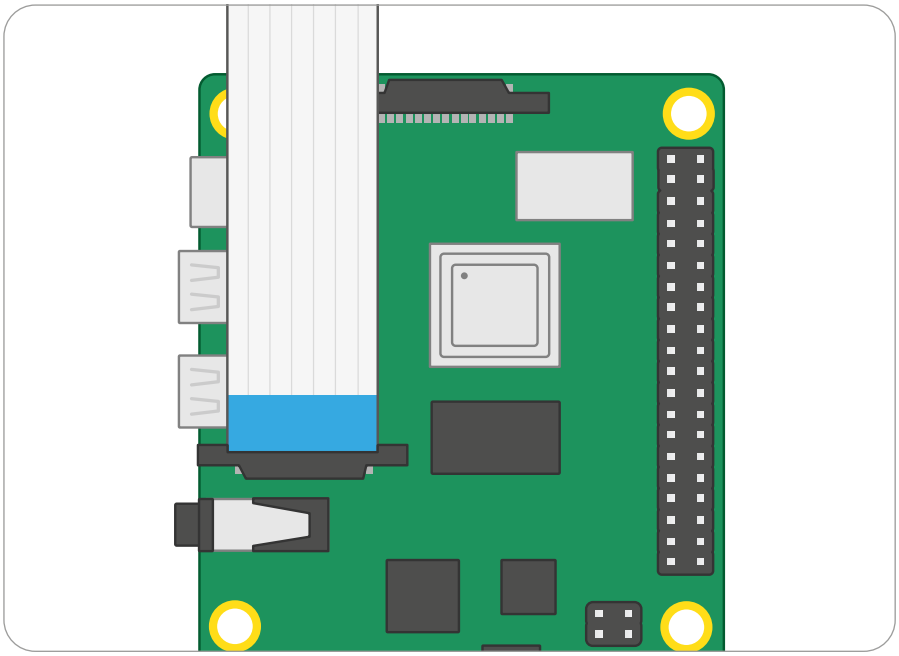


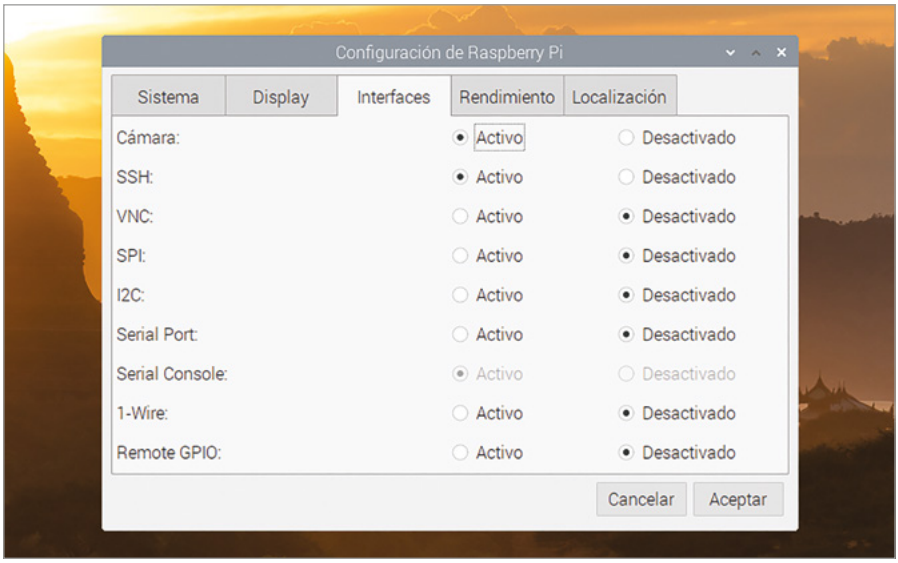
Figura 8-2: Conectar el cable plano al puerto de la cámara/CSI en Raspberry Pi

El Camera Module se suministra con un plástico azul sobre la lente, para protegerla de arañazos durante la fabricación, el transporte y la instalación. Localiza el plástico y tira de él suavemente para quitarlo y poder usar la cámara.

AJUSTAR EL ENFOQUE

El Camera Module se suministra normalmente con una pequeña rueda de plástico para ajustar el enfoque de la lente. Aunque el enfoque ajustado de fábrica suele ser perfecto, si utilizas la cámara para primerísimos planos puedes deslizar la rueda sobre la lente y girarla suavemente para ajustar el enfoque manualmente. Encontrarás información sobre el enfoque de la HQ Camera en el **Apéndice F**.

Vuelve a conectar la fuente de alimentación a Raspberry Pi y espera a que se cargue el sistema operativo Raspberry Pi OS. Antes de poder usar la cámara, tendrás que decirle a Raspberry Pi que tiene una conectada: abre el menú de iconos de Raspberry, elige la categoría Preferencias y haz clic en Configuración de Raspberry Pi. Cuando la herramienta se haya cargado, haz clic en la pestaña Interfaces, busca la entrada de la cámara en la lista y haz clic en el botón de selección redondo a la izquierda de "Activo" para activar el ajuste (**Figura 8-3** a continuación). Haz clic en OK. La herramienta te pedirá que reinicies tu Raspberry Pi. Después del reinicio podrás empezar a usar la cámara.

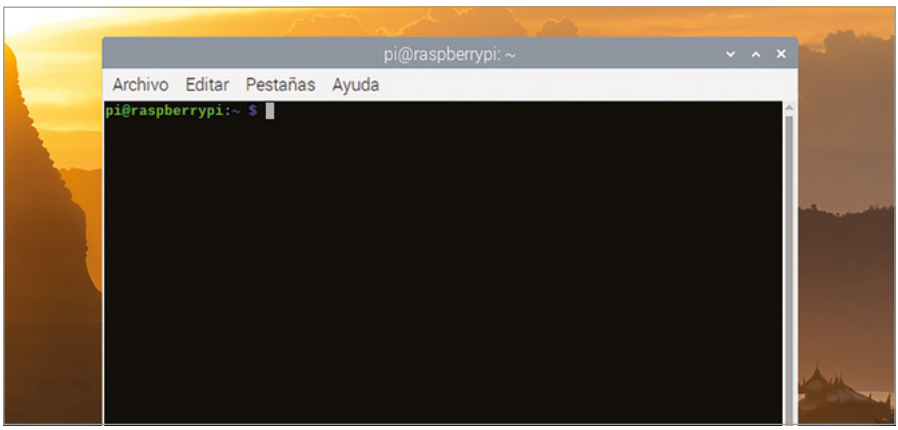


▲ **Figura 8-3:** Debes activar la cámara en la Configuración de Raspberry Pi

Probar la cámara

Para confirmar que la instalación de tu Camera Module o HQ Camera es correcta y que has activado la interfaz en la Configuración de Raspberry Pi, puedes utilizar la herramienta **raspistill**. Con ella y con **raspivid** para vídeos se capturan imágenes de la cámara usando la *interfaz de línea de comandos (CLI)* de Raspberry Pi.

A diferencia de los programas que has usado hasta ahora, no encontrarás raspistill en el menú. Haz clic en el icono de Raspberry Pi para cargar el menú, elige la categoría Accesorios y haz clic en LXTerminal. Aparecerá una ventana negra con texto en verde y azul (**Figura 8-4**): este es el *terminal* que te permite acceder a la interfaz de línea de comandos.



▲ **Figura 8-4:** Abrir una ventana de Terminal para introducir comandos

Para probar la cámara, escribe lo siguiente en el Terminal:

```
raspistill -o test.jpg
```

En cuanto pulses **ENTRAR** verás en pantalla una imagen de lo que ve la cámara (**Figura 8-5**). Esta es la *vista previa dinámica* y será visible durante 5 segundos, a menos que le indiques otra duración a raspistill. Al cabo de 5 segundos, la cámara capturará una sola imagen fija y la guardará en tu carpeta Home con el nombre **test.jpg**. Si quieres capturar otra imagen, vuelve a escribir el mismo comando pero asegúrate de cambiar el nombre del archivo resultante (después de **`-o`**), para evitar sobrescribir la primera imagen.

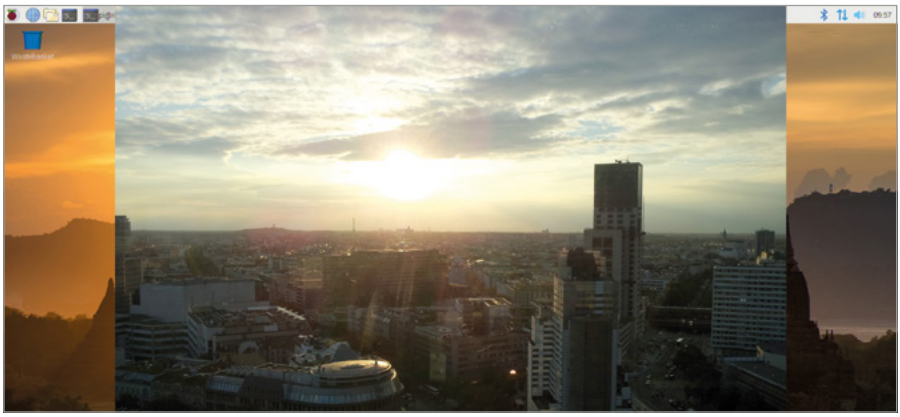


Figura 8-5: Vista previa dinámica de la cámara

Si la vista previa dinámica estaba boca abajo, tendrás que decirle a raspistill que la cámara está girada. El Camera Module está diseñado para que el cable plano salga por el borde inferior. Si sale por los lados o por arriba, como en algunos accesorios de montaje de cámaras de otras marcas, puedes girar la imagen 90, 180 o 270 grados usando el conmutador **`-rot`**. Para una cámara montada con el cable sobresaliendo por la parte superior, usa el comando:

```
raspistill -rot 180 -o test.jpg
```

Si el cable plano sale del borde derecho, usa un valor de rotación de 90 grados. Si sale del borde izquierdo, usa 270 grados. Si la captura original tenía un ángulo incorrecto, haz otra usando **`-rot`** para corregirlo.

Para ver la foto que has hecho, abre el Gestor de archivos en la categoría Accesorios del menú de Raspberry: la imagen que has creado, llamada **test.jpg**, estará en tu carpeta **home/pi**. Localízala en la lista de archivos y haz doble clic en ella para cargarla en un visor de imágenes (**Figura 8-6**). También puedes adjuntar la imagen a correos electrónicos, subirla a sitios web a través del navegador o arrastrarla a un dispositivo de almacenamiento externo.

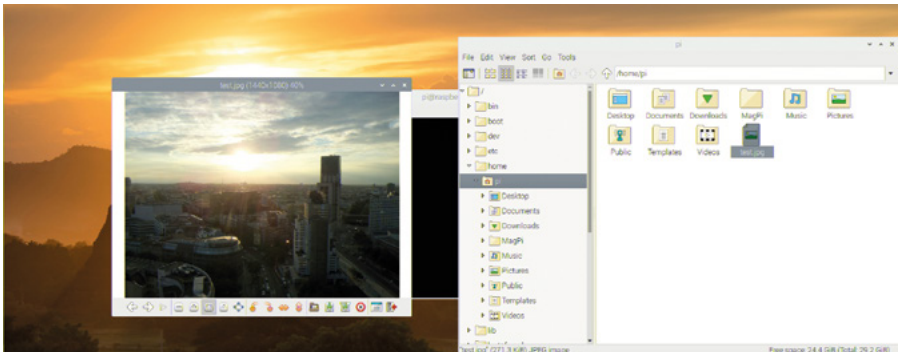
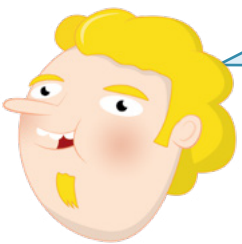


Figura 8-6: Abrir la imagen capturada

picamera

El modo más flexible de controlar el Camera Module o la HQ Camera es usando Python, a través de una práctica biblioteca llamada picamera, que te da un control total de la vista previa de la cámara, las imágenes y las capacidades de captura de vídeo. También te permite integrar las imágenes en tus propios programas y combinarlas con programas que utilizan el módulo GPIO a través de la biblioteca GPIO Zero.



PROGRAMACIÓN CON PYTHON

Los proyectos de este capítulo presuponen que tienes experiencia con el lenguaje de programación Python, Thonny IDE y los pines GPIO de Raspberry Pi. Si aún no lo has hecho, debes completar antes los proyectos descritos en el **Capítulo 5, Programar con Python** y el **Capítulo 6, Informática física con Scratch y Python**.

Si el Terminal aún está abierto ciérralo haciendo clic en la X de la parte superior derecha de la ventana. Luego carga Thonny desde la categoría Programación del menú de Raspberry. Guarda tu nuevo proyecto como **Camera** y luego empieza a importar las bibliotecas que necesita tu programa escribiendo lo siguiente en el área de script:

```
from picamera import PiCamera
from time import sleep
camera = PiCamera()
```

La última línea te permite controlar el Camera Module o la HQ Camera usando la función **camera**. Para empezar, escribe lo siguiente:

```
camera.start_preview()
sleep(10)
camera.stop_preview()
```

Haz clic en Run y el escritorio desaparecerá para dar lugar a una vista previa a pantalla completa de todo lo que ve la cámara (**Figura 8-7**). Anda un poco o mueve la mano frente al objetivo y verás que la imagen en pantalla cambia para mostrarlo. Al cabo de 10 segundos la vista previa se cerrará y tu programa terminará pero, a diferencia de la vista previa de raspistill, no se guardará ninguna imagen.

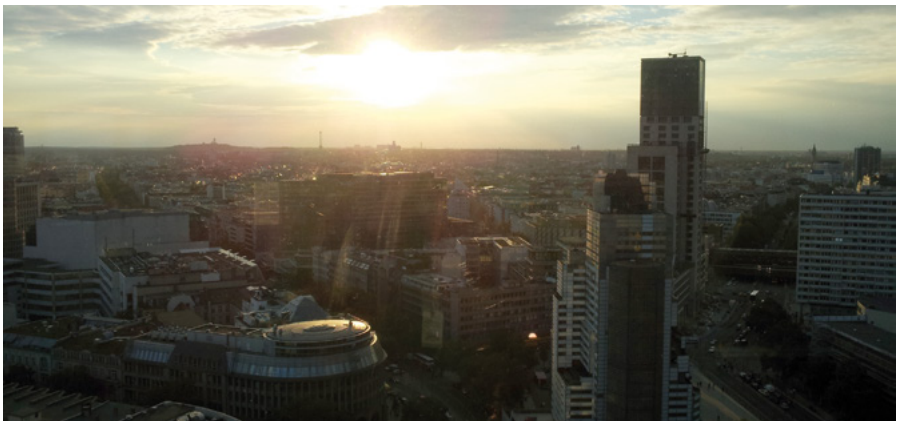


Figura 8-7: Vista previa dinámica a toda pantalla de lo que ve la cámara

Si la vista previa estaba boca abajo, puedes rotar la imagen. Justo debajo de la línea **camera = PiCamera()** escribe:

```
camera.rotation = 180
```

Si la vista previa estaba boca abajo, esa línea hará que las cosas se vean bien de nuevo. Como en el caso de raspistill, **camera.rotation** permite girar la imagen 90, 180 o 270 grados, dependiendo de si el cable sale por el lado derecho, superior o izquierdo del Camera Module. Acuérdate de usar **camera.rotation** al comienzo de cualquier programa que escribas, para que las imágenes o vídeo que captures no salgan al revés.

Capturar imágenes fijas

Para capturar una imagen, en lugar de mostrar una vista previa dinámica, es necesario cambiar el programa. Empieza por reducir el retraso de la vista previa: localiza la línea **`sleep(10)`** y cámbiala a:

```
sleep(5)
```

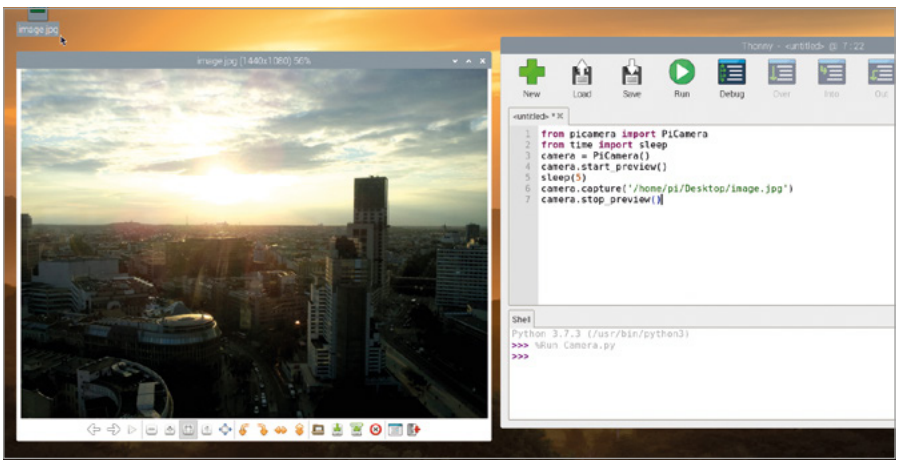
Directamente debajo de esa línea, añade esto:

```
camera.capture('/home/pi/Desktop/image.jpg')
```

AJUSTES

Cuando la cámara está en modo de vista previa, analiza el vídeo para ver si se deben modificar los ajustes para obtener la mejor calidad. Lo notarás si estás en un entorno muy oscuro o muy claro: al principio la vista previa será imposible de ver y se irá aclarando rápidamente. Para dar tiempo a que se ajuste la cámara, añade siempre a tu programa un tiempo de vista previa de al menos 2 segundos antes de capturar una imagen.

La función **`camera.capture`** le dice a Python que guarde una imagen fija y, además del nombre de la imagen, también debe saber en qué carpeta quieres guardarla. En este ejemplo, la vas a guardar en el escritorio, lo encontrarás justo debajo de la papelera. Si la ventana de Thonny se interpone, simplemente haz clic en la barra de título y arrástrala para moverla. Haz doble clic en el archivo para ver la imagen que has capturado (**Figura 8-8**). Enhorabuena: has programado una cámara.



▲ **Figura 8-8:** Abrir la imagen capturada

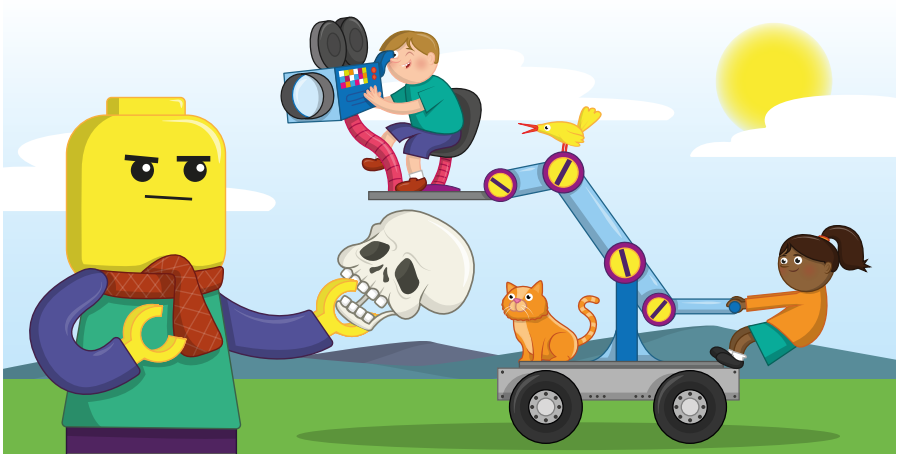
Capturar vídeo en movimiento

Además de capturar imágenes fijas, puedes capturar vídeo. Borra todo lo que hay entre las líneas **`camera.start_preview()`** y **`camera.stop_preview()`** y escribe lo siguiente bajo **`camera.start_preview():`**

```
camera.start_recording('/home/pi/Desktop/video.h264')
sleep(10)
camera.stop_recording()
```

La vista previa de la cámara aparecerá, como antes, pero esta vez también se grabará en un archivo en el escritorio. Espera los 10 segundos que le has indicado a Python previamente y si quieres puedes bailar un poco frente a la cámara para hacer el vídeo más interesante. Cuando se cierre la vista previa, encontrarás tu archivo de vídeo en el escritorio.

Para reproducir el vídeo, haz doble clic en el archivo **video.h264** en el escritorio. El vídeo empezará a reproducirse y, si antes has bailado ante la cámara, también verás tu baile. Cuando el vídeo termine, el software del reproductor se cerrará con un mensaje en el Terminal. Enhorabuena: has aprendido a capturar vídeo usando el Camera Module o la HQ Camera de Raspberry Pi.



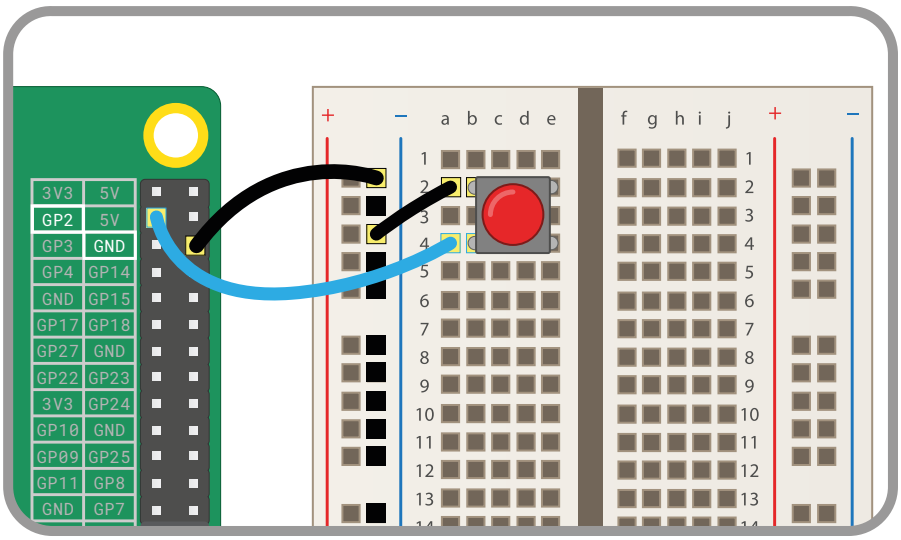
Animación stop motion con un conmutador

Usando lo que has aprendido en este capítulo y lo que aprendiste sobre cómo conectar el hardware al sistema GPIO de Raspberry Pi, en el **Capítulo 6**, **Informática física**, ya puedes ponerte a crear algo especial: tu propio estudio de animación stop motion.

La animación stop motion es el proceso de hacer muchas fotos de objetos estáticos (por ejemplo coches en miniatura o figuras de acción) e ir moviéndolos ligeramente entre foto y foto. Aunque los objetos no se mueven en ninguno de los cuadros, si los muestras uno tras otro con la suficiente rapidez, parecerá que se mueven tan rápido o tan despacio como quieras.

Para este proyecto, necesitarás un conmutador momentáneo, una placa de pruebas, un puente macho-macho (M2M) y un par de puentes macho-hembra (M2F). Si no tienes una placa de pruebas puedes conectar el conmutador usando cables hembra-hembra (F2F), pero será más difícil de pulsar. Si tienes que revisar la información sobre esos componentes, ve al **Capítulo 6, Informática física con Scratch y Python**. También necesitarás objetos que animar: pueden ser cualquier cosa, un trozo de arcilla, un coche de juguete, una figura de acción...

Empieza por crear el circuito: añade el conmutador a la placa de pruebas, conecta el raíl de tierra de la placa de pruebas a un pin de tierra de Raspberry Pi (identificado como GND en la **Figura 8-9**) con un cable puente macho-hembra. Utiliza un puente macho-macho para conectar una pata del conmutador al raíl de tierra en la placa de pruebas, luego un puente macho-hembra para conectar la otra pata del conmutador al pin 2 de GPIO (identificado como GP2 en la **Figura 8-9**).



▲ **Figura 8-9:** Diagrama de cableado para conectar un conmutador a los pines del sistema GPIO

Crea un nuevo proyecto en Thonny y guárdalo como **Stop Motion**. Empieza por importar y configurar las bibliotecas que necesitas para usar la cámara y el puerto GPIO:

```
from picamera import PiCamera
from gpiozero import Button
camera = PiCamera()
button = Button(2)
```

Escribe lo siguiente:

```
camera.start_preview()
button.wait_for_press()
camera.capture('/home/pi/Desktop/image.jpg')
camera.stop_preview()
```

Haz clic en Run y verás una vista previa de lo que esté enfocado por la cámara. La vista previa permanecerá en pantalla hasta que pulses el conmutador momentáneo: púlsalo ahora y la vista previa se cerrará después de que tu programa guarde una imagen en el escritorio. Encuentra la imagen, llamada **image.jpg** y haz doble clic para abrirla y confirmar que el programa funciona.

La animación stop motion implica la creación de muchas imágenes fijas, para dar la impresión de movimiento cuando se muestran todas juntas. Sería un desbarajuste tener todas esas fotos individuales en el escritorio, así que necesitas una carpeta en la que almacenarlas. Haz clic con el botón derecho del ratón en cualquier punto vacío del escritorio y elige New Folder (**Figura 8-10**). Llama a la carpeta **animation**, en minúsculas, y luego haz clic en el botón OK.

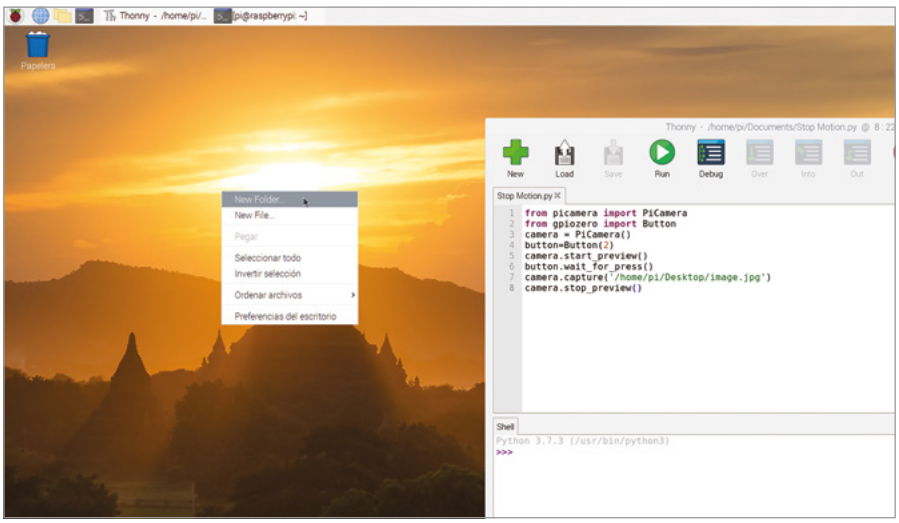


Figura 8-10: Crear una carpeta para las imágenes capturadas

No es práctico tener que reiniciar el programa cada vez que se captura una imagen para la animación, así que debes cambiarlo para que se ejecute en un bucle. A diferencia de los bucles que has creado anteriormente, este debe cerrarse con más precisión. Si no, al detener el programa mientras se muestra la vista previa de la cámara, ya no podrás ver el escritorio. Necesitas usar dos instrucciones especiales: `try` y `except`.

Empieza por borrar todo lo que hay después de **camera.start_preview(),** y escribe:

```
frame = 1
```

Esto crea una nueva variable **frame** que tu programa usará para almacenar el número del cuadro actual. Dentro de poco usarás esto para asegurarte de que guardas un nuevo archivo cada vez. ¡Si no, corres el riesgo de sobrescribir la imagen previa cada vez que pulses el botón!

A continuación, configura el bucle escribiendo:

```
while True:
    try:
```

La nueva instrucción **try** le dice a Python que ejecute cualquier código que haya dentro: ese es el código para capturar imágenes. Escribe:

```
button.wait_for_press()
camera.capture('/home/pi/Desktop/animation/frame%03d.jpg' % frame)
frame += 1
```

En estas tres líneas de código hay un par de trucos ingeniosos. El primero está en el nombre del archivo de captura: **%03d** le dice a Python que tome un número y le añada tantos ceros delante como sea necesario para que tenga tres dígitos en total. Por ejemplo: "1" se convierte en "001", "2" se convierte en "002", y "10" se convierte en "010". Necesitas esto en tu programa para mantener tus archivos en el orden correcto y asegurarte de que no estás sobrescribiendo un archivo que ya has guardado.

% frame al final de esa línea le dice a Python que use el número de la variable frame en el nombre del archivo. Para asegurarte de que cada archivo es único, la última línea (**frame += 1**) incrementa la variable frame en una unidad. La primera vez que pulses el botón, **frame** pasará de 1 a 2; la próxima vez, de 2 a 3; y así sucesivamente.

De momento, tu código no termina impecablemente cuando acabas de hacer fotos. Para que eso suceda, necesitas una instrucción **except** para **try**. Escribe lo siguiente, acordándote de eliminar un nivel de sangría en la primera línea para que Python sepa que no es parte de la sección **try**:

```
except KeyboardInterrupt:
    camera.stop_preview()
    break
```

Tu programa finalizado se verá así:

```
from picamera import PiCamera
from time import sleep
from gpiozero import Button
camera = PiCamera()
button = Button(2)
camera.start_preview()
frame = 1
while True:
    try:
        button.wait_for_press()
        camera.capture('/home/pi/Desktop/animation/frame%03d.jpg'
% frame)
        frame += 1
    except KeyboardInterrupt:
        camera.stop_preview()
        break
```

Haz clic en Run pero, en lugar de pulsar el botón, pulsa **CTRL** y **C** en el teclado. No hace falta que pulses ambas teclas al mismo tiempo: mantén pulsada **CTRL**, pulsa y suelta **C** y luego suelta **CTRL**. Estas dos teclas actúan como una interrupción, diciéndole a Python que deje de hacer lo que está haciendo. Sin la línea **`except KeyboardInterrupt:`** Python se cerraría inmediatamente y dejaría la vista previa de la cámara bloqueando la pantalla. Con esa línea, Python ejecuta cualquier código que esté dentro, en este caso código que le dice que detenga la vista previa de la cámara y se cierre perfectamente.

¡Ya puedes empezar a capturar tu animación stop motion! Coloca el Camera Module o la HQ Camera donde pueda ver los objetos que vas a animar y asegúrate de que no se mueva: si la cámara se mueve, estropea el efecto. Coloca los objetos en sus posiciones iniciales y haz clic en Run para iniciar el programa. Comprueba que todo se vea bien en la vista previa y pulsa el conmutador para capturar tu primer cuadro.

Mueve los objetos ligeramente (cuanto menos los muevas entre cuadros, más fluida será la animación terminada) y vuelve a pulsar el conmutador para capturar otro cuadro. Sigue haciéndolo hasta terminar la animación: cuantos más cuadros captures, más larga será.

Cuando termines, pulsa **CTRL+C** para cerrar tu programa y haz doble clic en la capeta **animation** del escritorio para ver las fotos que has capturado (**Figura 8-11** a continuación). Haz doble clic en cualquier imagen para abrirla y verla con más detalle.

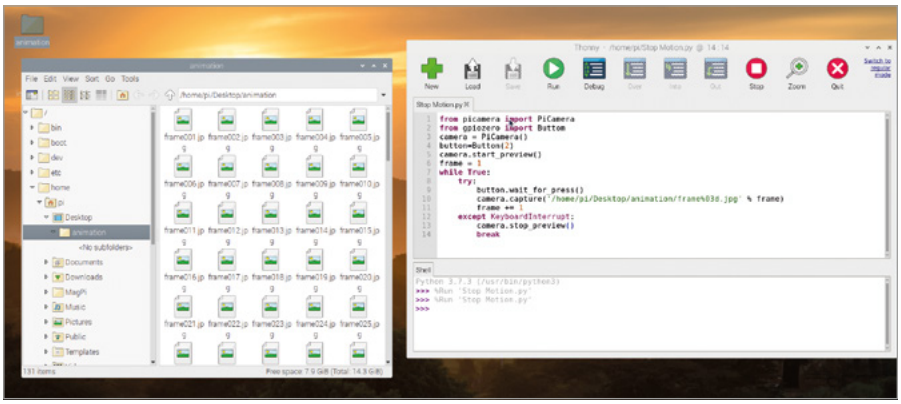


Figura 8-11: Las imágenes capturadas en la carpeta

De momento, lo único que tienes es una carpeta llena de imágenes fijas. Para crear una animación, debes convertirlas en un vídeo. Para ello, haz clic en el icono de Raspberry Pi para cargar el menú, elige Accesorios y haz clic en LXTerminal. Se abrirá una *interfaz de línea de comandos* (algo que se describe más detalladamente en el **Apéndice C**), donde puedes escribir comandos para Raspberry Pi. Cuando el Terminal se cargue, accede a la carpeta que has creado escribiendo:

```
cd Desktop/animation
```

Es importante que la "D" de "Desktop" esté en mayúscula: Raspberry Pi OS *distingue entre mayúsculas y minúsculas*, eso significa que si no escribes un comando o un nombre de carpeta exactamente como se escribió originalmente, ¡no funcionará! Cuando hayas cambiado de carpeta, escribe lo siguiente:

```
ffmpeg -i frame%03d.jpg -r 10 animation.h264
```

Esto utiliza un programa llamado **ffmpeg** para usar las imágenes fijas de la carpeta y convertirlas en un vídeo llamado **animation.h264**. (Nota: si ffmpeg no está disponible, puedes instalarlo con `sudo apt-get install ffmpeg`.) Dependiendo de cuántas fotos hayas hecho, este proceso puede tardar unos minutos. Cuando finalice veras reaparecer el aviso del Terminal.

Para reproducir el vídeo, localiza el archivo **animation.h264** en tu carpeta **animation** y haz doble clic para abrirlo. También puedes reproducirlo desde el Terminal escribiendo:

```
omxplayer animation.h264
```

Una vez cargado el vídeo, entrará en acción tu animación stop motion. Enhorabuena: has convertido tu Raspberry Pi en un estudio de animación.

Si tu animación se mueve con demasiada rapidez o demasiada lentitud, cambia la parte **`-r 10`** del comando **`ffmpeg`** por un número más bajo o más alto: eso es la velocidad de cuadro (cuántas imágenes fijas hay en un segundo de vídeo). Un número bajo hará que la animación se ejecute más lentamente, pero se verá menos fluida. Un número alto producirá más fluidez, pero hará que la animación se ejecute más rápidamente.

Si quieres guardar tu vídeo, asegúrate de arrastrarlo desde el escritorio y soltarlo en la carpeta Videos. De lo contrario, la próxima vez que ejecutes el programa, ¡terminarás sobrescribiendo el archivo!

Ajustes avanzados de la cámara

Si necesitas más control sobre el módulo de Camera Module o la HQ Camera de Raspberry Pi, puedes usar la biblioteca picamera de Python para acceder a varios ajustes. Estos ajustes, junto con sus valores predeterminados, se detallan a continuación y los puedes incluir en tus propios programas.

`camera.awb_mode = 'auto'`
Ajusta el modo de balance de blanco automático de la cámara y ofrece los siguientes modos: **`off`**, **`auto`**, **`sunlight`**, **`cloudy`**, **`shade`**, **`tungsten`**, **`fluorescent`**, **`incandescent`**, **`flash`** y **`horizon`**. Si crees que tus fotos y vídeos son demasiado azulados o amarillentos, prueba un modo diferente.

`camera.brightness = 50`
Ajusta el brillo de la imagen de la cámara: el valor 0 corresponde al más oscuro y el 100 al más brillante.

`camera.color_effects = None`
Cambia el efecto de color que utiliza la cámara actualmente. Normalmente, este ajuste debe mantenerse como está, pero si indicas un par de números puedes alterar la forma en que la cámara graba el color: prueba con **`(128, 128)`** para crear una imagen en blanco y negro.

`camera.contrast = 0`
Define el contraste de la imagen. Un número alto da mayor dramatismo y austeridad. Un número bajo hará que las cosas parezcan más desvaídas. Puedes usar cualquier número entre -100, para el mínimo contraste, y 100 para el máximo.

`camera.crop = (0.0, 0.0, 1.0, 1.0)`
Esto te permite recortar la imagen, quitando porciones de los lados y la parte superior para capturar solo la parte de la imagen que necesitas. Los números representan la coordenada X, la coordenada Y, la anchura y la altura. El ajuste predeterminado captura la imagen completa. Reduce los dos últimos números -0.5 y 0.5: es un buen punto de partida y verás qué efecto tiene este ajuste.

`camera.exposure_compensation = 0`
Establece la *compensación* de exposición de la cámara, permitiéndote controlar manualmente cuánta luz se captura para cada imagen. A diferencia del cambio de brillo, este ajuste controla la cámara propiamente dicha. Los valores válidos están entre -25 para una imagen muy oscura y 25 para una imagen muy brillante.

`camera.exposure_mode = 'auto'`
Establece el *modo de exposición* o la lógica que el módulo de la cámara/la cámara de alta calidad utiliza para decidir cómo exponer una imagen. Los modos posibles son: **`off`**, **`auto`**, **`night`**, **`backlight`**, **`spotlight`**, **`sports`**, **`snow`**, **`beach`**, **`verylong`**, **`fixedfps`**, **`antishake`** y **`fireworks`**.

`camera.framerate = 30`
Establece el número de imágenes capturadas por segundo para crear un vídeo: la *velocidad de cuadro*. Una velocidad de cuadro alta crea un vídeo más fluido, pero ocupa más espacio de almacenamiento. Las velocidades de cuadro más altas requieren una resolución menor para su uso. Eso se puede configurar a través de **`camera.resolution`**.

`camera.hflip = False`
Voltea la imagen de la cámara en el eje horizontal, o X, cuando el valor activo es **`True`**.

`camera.image_effect = 'none'`
Aplica a la secuencia de vídeo uno de los efectos de una gama de efectos de imagen, que será visible en la vista previa y también en las imágenes y los vídeos guardados. Los posibles efectos son: **`blur`**, **`cartoon`**, **`colorbalance`**, **`colorpoint`**, **`colorswap`**, **`deinterlace1`**, **`deinterlace2`**, **`denoise`**, **`emboss`**, **`film`**, **`gpen`**, **`hatch`**, **`negative`**, **`none`**, **`oilpaint`**, **`pastel`**, **`posterise`**, **`saturation`**, **`sketch`**, **`solarize`**, **`washedout`** y **`watercolor`**.

`camera.ISO = 0`
Cambia el ajuste ISO de la cámara, lo que afecta a su sensibilidad a la luz. De forma predeterminada, la cámara lo ajusta automáticamente dependiendo de la luz disponible. Puedes ajustar el valor de ISO usando uno de estos valores: 100, 200, 320, 400, 500, 640, 800. Cuanto más alto sea el valor ISO, mejor será el rendimiento de la cámara en entornos con poca luz, pero más granulada será la imagen o el vídeo que capture.

`camera.meter_mode = 'average'`
Esto controla la forma en que la cámara decide la cantidad de luz disponible al ajustar su exposición. El valor predeterminado es el promedio de la cantidad de luz disponible en todo el cuadro. Otros modos posibles son **`backlit`**, **`matrix`** y **`spot`**.

`camera.resolution = (1920, 1080)`
Establece la resolución de la imagen o el vídeo capturado, representada por dos números para la anchura y la altura. Las resoluciones bajas ocuparán menos espacio de almacenamiento y te permitirán utilizar una mayor velocidad de cuadro. Las resoluciones altas ofrecen mejor calidad pero ocupan más espacio de almacenamiento.

`camera.rotation = 0`
Controla la rotación de la imagen: 0, 90, 180 o 270 grados. Usa este ajuste si no puedes posicionar la cámara de manera que el cable plano salga por la parte inferior.

`camera.saturation = 0`
Controla la saturación de la imagen (o sea, la intensidad de los colores). Los valores posibles van de -100 a 100.

`camera.sharpness = 0`
Controla la nitidez de la imagen. Los valores posibles van de -100 a 100.

`camera.shutter_speed = 0`
Controla la rapidez con la que el obturador se abre y se cierra al capturar imágenes y vídeos. Puedes ajustar la velocidad de obturación manualmente en microsegundos: las velocidades de obturación largas funcionan mejor con poca luz y las más rápidas son mejores para una mayor luminosidad. Por lo general se debería mantener el ajuste automático predeterminado.

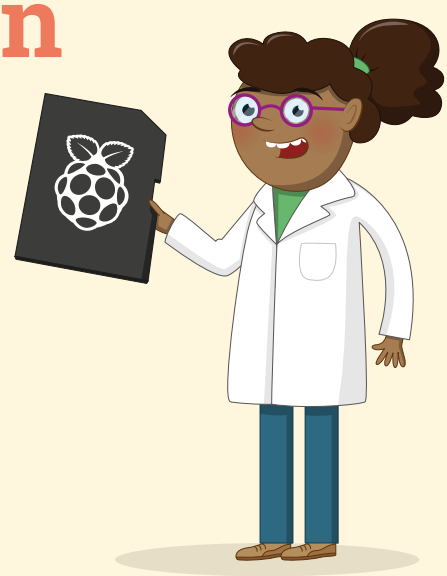
`camera.vflip = False`
Voltea la imagen de la cámara en el eje vertical, o Y, cuando el valor activo es **`True`**.

`camera.video_stabilization = False`
Cuando el ajuste activo es **`True`**, activa la estabilización de vídeo. Solo se necesita si el Camera Module o la HQ Camera se mueve mientras grabas (por ejemplo, si está conectada a un robot o se lleva de un lado a otro) para reducir el temblor del vídeo capturado.

Hay más información sobre estos ajustes, y otros no documentados aquí, en **picamera.readthedocs.io**.

Apéndice A

Instalar un sistema operativo en una tarjeta microSD



Puedes comprar tarjetas microSD con NOOBS (New Out of the Box Software) preinstalado a través de distribuidores acreditados de Raspberry Pi. Con ellas se puede instalar fácilmente el sistema operativo Raspberry Pi (antes conocido como Raspbian) para Raspberry Pi. O puedes usar las siguientes instrucciones para usar Raspberry Pi Imager e instalar un sistema operativo manualmente en tu propia tarjeta microSD vacía (o reutilizada).

> **¡ADVERTENCIA!**
>
> Si has comprado una tarjeta microSD con NOOBS preinstalado, no necesitas hacer nada más que conectarla a tu Raspberry Pi. Estas instrucciones son para tarjetas microSD en blanco o para tarjetas usadas anteriormente en las que se quiere instalar un nuevo sistema operativo. Si estas instrucciones se llevan a cabo en una tarjeta microSD con archivos, se perderán esos archivos, así que asegúrate de hacer una copia de seguridad antes de empezar.

Descargar Raspberry Pi Imager

Basado en Debian, el sistema operativo Raspberry Pi es el sistema oficial de Raspberry Pi. La manera más fácil de instalar el sistema operativo Raspberry Pi OS en una tarjeta microSD para tu Raspberry Pi es con la herramienta Raspberry Pi Imager, descargable desde **rpf.io/downloads**. Ten en cuenta que este método sustituye a la instalación del sistema operativo mediante NOOBS, aunque este último sigue estando disponible en la misma página de descargas.

La aplicación Raspberry Pi Imager está disponible para ordenadores con Windows, macOS y Ubuntu Linux: elige la versión relevante para tu sistema.

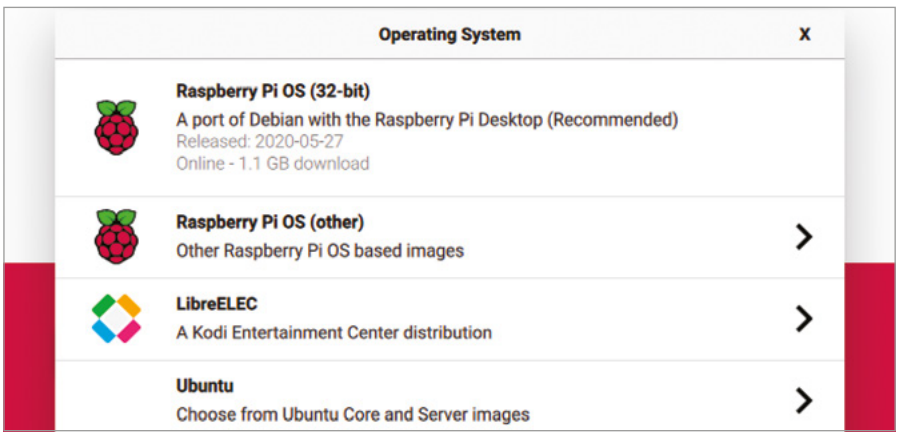
En macOS, haz doble clic en el archivo DMG descargado. Es posible que tengas que cambiar la configuración de seguridad y privacidad para permitir que se ejecuten las aplicaciones descargadas de App Store y desarrolladores identificados. Luego puedes arrastrar el icono de Raspberry Pi Imager a la carpeta de aplicaciones.

En un PC con Windows, haz doble clic en el archivo EXE descargado. Cuando se te indique, selecciona el botón "Sí" para que se ejecute. Luego, haz clic en el botón "Instalar" para iniciar la instalación.

Escribir el sistema operativo en la tarjeta microSD

Conecta tu tarjeta microSD a tu ordenador PC o Mac: necesitarás un adaptador USB de tarjeta microSD a menos que tenga un lector de tarjetas incorporado. Ten en cuenta que no hace falta formatear la tarjeta de antemano.

Abre la aplicación Raspberry Pi Imager. Haz clic en el botón "Choose OS" para seleccionar el sistema operativo que deseas instalar. La opción superior es el sistema operativo estándar Raspberry Pi OS. Si prefieres la versión reducida Lite o la versión completa (con todo el software recomendado preinstalado), selecciona "Raspberry Pi OS (other)". También hay opciones para instalar LibreELEC (elige la versión para tu modelo de Raspberry Pi) y Ubuntu Core o Server.



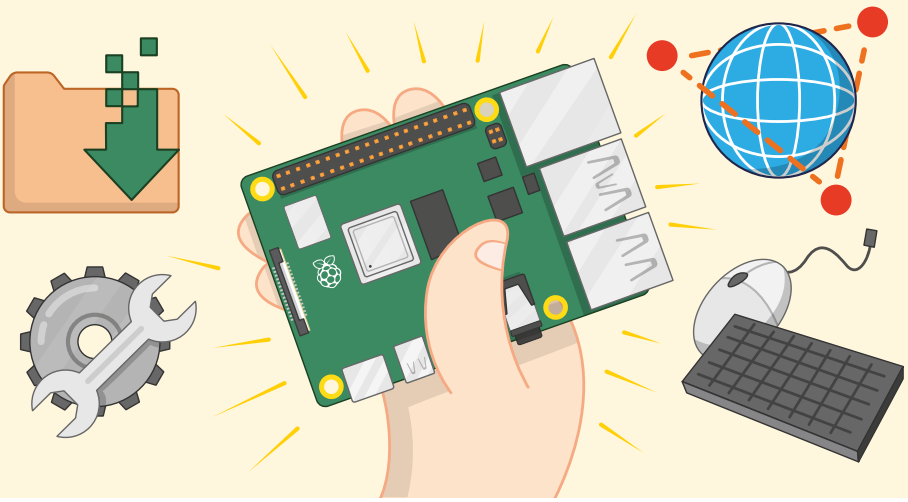
Nota: Si quieres instalar otro sistema operativo diferente, como Lakka, solo tienes que descargar el archivo de imagen de la página web correspondiente y seleccionar la opción "Use Custom" en Raspberry Pi Imager.

Con un sistema operativo seleccionado, haz clic en el botón "Choose SD card" y selecciona tu tarjeta microSD (normalmente sólo habrá una opción).

Por último, haz clic en el botón "Write" y espera mientras la utilidad escribe el sistema operativo seleccionado en tu tarjeta y luego lo verifica. Al terminar, puedes retirar la tarjeta microSD. Luego puedes insertarla en tu Raspberry Pi y arrancarlo en el sistema operativo que acabas de instalar.

Apéndice B

Instalar y desinstalar software



El sistema operativo Raspberry Pi se suministra con una selección de populares paquetes de software elegidos por la Fundación Raspberry Pi, pero no son los únicos que funcionan en un Raspberry Pi. Usando las siguientes instrucciones, puedes buscar software adicional, instalarlo y desinstalarlo de nuevo, para ampliar las capacidades de tu Raspberry Pi.

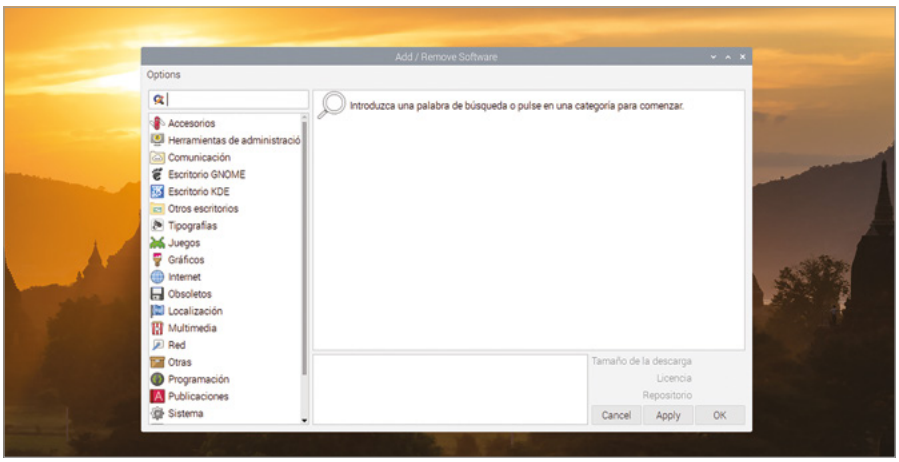
Las instrucciones de este apéndice suplementan a las proporcionadas en el **Capítulo 3, Uso de Raspberry Pi**, en el que se explica cómo utilizar la herramienta Recommended Software. Si aún no has leído ese capítulo, debes hacerlo antes de utilizar los métodos descritos en este apéndice.

CAPACIDAD DE LA TARJETA

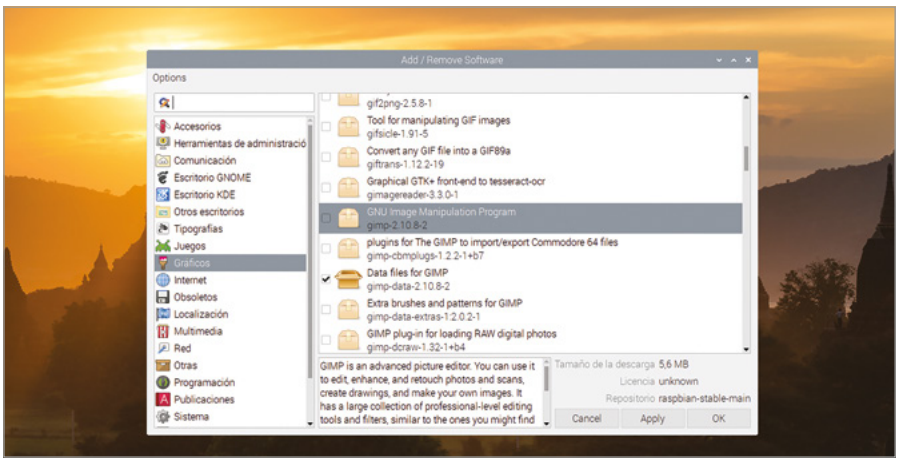
Al añadir más software a tu Raspberry Pi, ocupará espacio en tu tarjeta microSD. Una tarjeta de 16 GB o más te permitirá instalar más software. Para comprobar si la tarjeta que quieres usar es compatible con Raspberry Pi, visita **rpf.io/sdcardlist**.

Ver software disponible

Para ver y buscar en la lista de paquetes de software disponibles para el sistema operativo Raspberry Pi, usando lo que se denomina *repositorios de software*, haz clic en el icono de Raspberry para cargar el menú, selecciona la categoría Preferencias y luego haz clic en Add/Remove Software. Al cabo de unos segundos aparecerá la ventana de la herramienta.



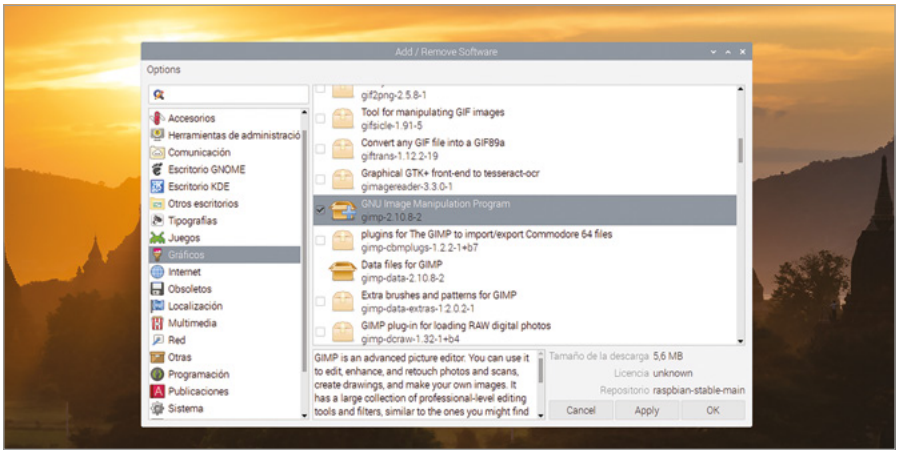
El lado izquierdo de la ventana de agregar o quitar software contiene una lista de categorías, las mismas que se encuentran en el menú principal cuando se hace clic en el icono de Raspberry. Al hacer clic en una de ellas se mostrará una lista de los programas disponibles en esa categoría. También puedes introducir un término de búsqueda en el cuadro de la parte superior izquierda de la ventana, por ejemplo: "text editor" o "game" y ver una lista de paquetes de software de la categoría correspondiente. Al hacer clic en cualquier paquete aparece información adicional sobre el mismo en la parte inferior de la ventana.



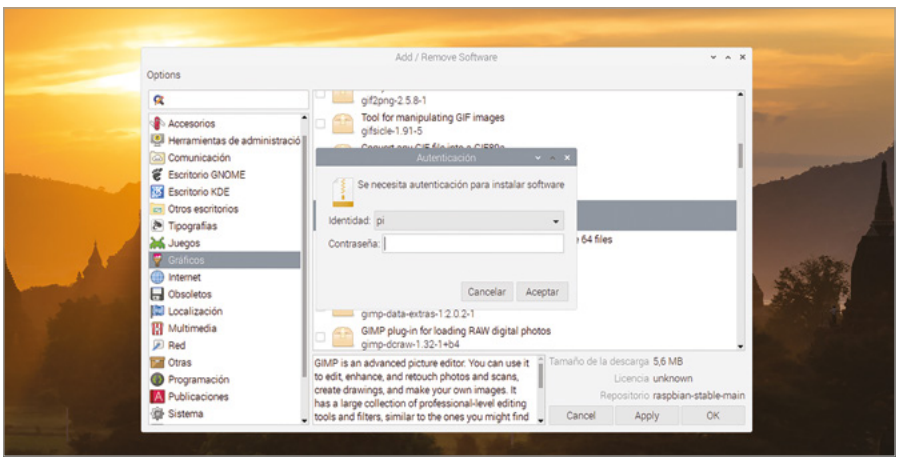
Si la categoría que has elegido tiene muchos paquetes de software disponibles, puede que la herramienta Add/Remove Software tarde un tiempo en terminar de compilar la lista.

Instalar software

Para seleccionar un paquete que instalar, marca la casilla correspondiente haciendo clic en ella. Puedes instalar más de un paquete a la vez: simplemente sigue haciendo clic para añadir más paquetes. El icono junto al paquete se convertirá en una caja abierta con un símbolo "+", para confirmar que se va a instalar.



Cuando hayas seleccionado todo lo que quieras, haz clic en el botón OK o Apply. La única diferencia es que OK cerrará la herramienta cuando se haya instalado el software, mientras que el botón Apply la mantiene abierta. Se te pedirá que introduzcas tu contraseña, para confirmar tu identidad: no queremos que cualquiera pueda añadir o quitar software de tu Raspberry Pi.

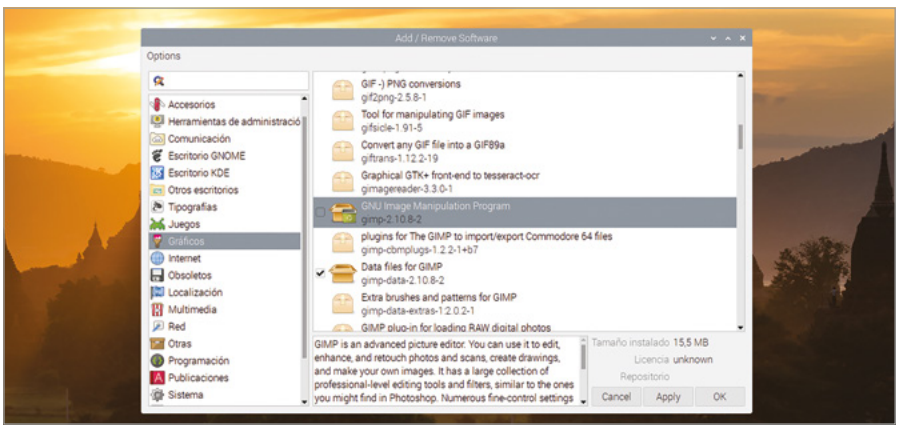


Es posible que al instalar un solo paquete, se instalen otros paquetes con él: son los denominados paquetes de *dependencias*, que el software elegido necesita para funcionar. Pueden ser, por ejemplo, paquetes de efectos de sonido para un juego o una base de datos para un servidor web.

Una vez instalado el software, deberías poder encontrarlo haciendo clic en el icono de Raspberry para cargar el menú y encontrar la categoría del paquete de software. Ten en cuenta que la categoría del menú no siempre es la misma que la categoría de la herramienta Add/Remove Software. Y algunos programas no tienen una entrada en el menú; son lo que se denomina *software de línea de comandos*, que se ejecuta en el Terminal. Para más información sobre la línea de comandos y el Terminal, consulta el **Apéndice C, La interfaz de línea de comandos**.

Desinstalar software

Para eliminar o *desinstalar* un paquete, localízalo en la lista de paquetes (aquí resulta útil la función de búsqueda) y quita la marca de la casilla correspondiente, haciendo clic en ella. Puedes desinstalar más de un paquete a la vez: solo tienes que seguir haciendo clic para eliminar más paquetes. El icono junto al paquete se convertirá en una caja abierta junto a un icono de papelera, para confirmar que se va a desinstalar.



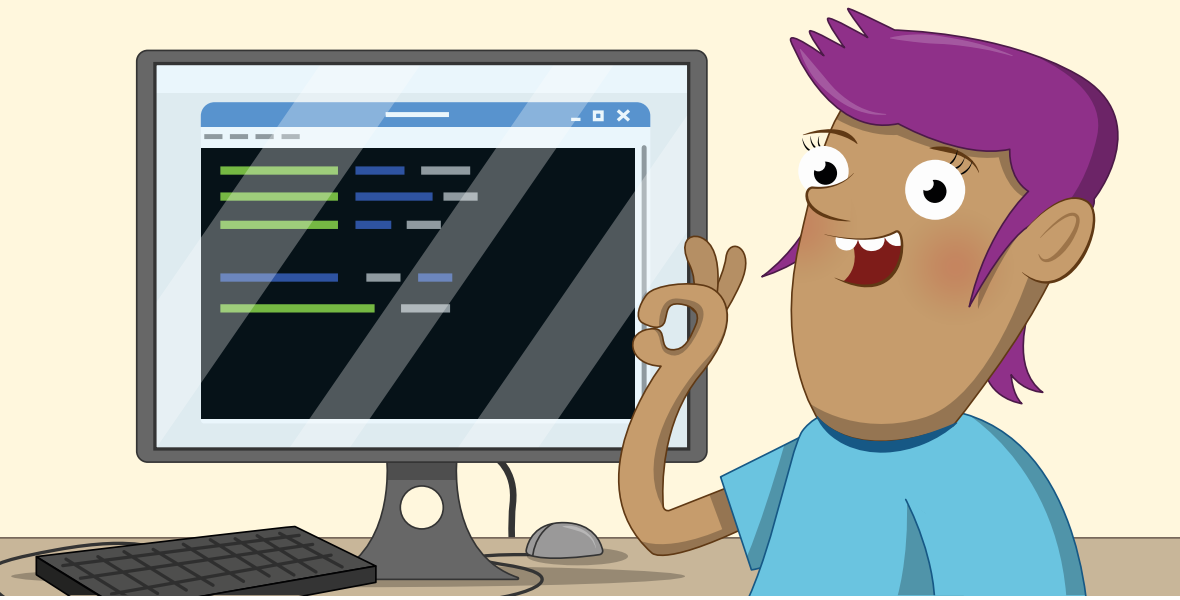
Como antes, puedes hacer clic en OK o en Apply para comenzar a desinstalar los paquetes de software seleccionados. Se te indicará que confirmes tu contraseña, a menos que lo hayas hecho en los últimos minutos, y también que confirmes que deseas eliminar cualquier dependencia relacionada con ese paquete de software. Al terminar la desinstalación, el software desaparecerá del menú de iconos de Raspberry, pero no se eliminarán los archivos que hayas creado con el software (por ejemplo, imágenes para un paquete de gráficos u opciones de juego guardadas).

¡ADVERTENCIA!

Todo el software instalado en el sistema operativo Raspberry Pi aparece en Add/Remove Software, incluido el software necesario para que tu Raspberry Pi funcione. Es posible que al quitar muchos paquetes el escritorio deje de cargarse. Para evitarlo, no desinstales cosas a menos que tengas la certeza de que ya no las necesitas. Si esta advertencia llega demasiado tarde, reinstala el sistema operativo Raspberry Pi siguiendo las instrucciones del **Capítulo 2, Preparativos para usar Raspberry Pi** o las del **Apéndice A**.

Apéndice C

La interfaz de línea de comandos



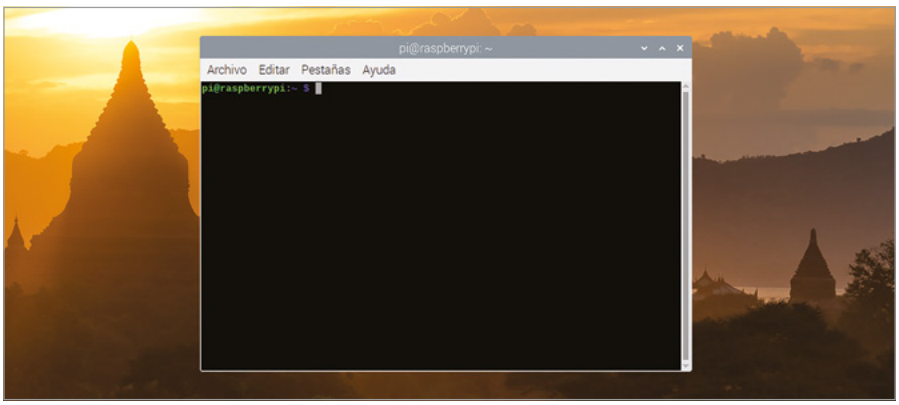
Aunque puedes gestionar la mayoría de los programas en un Raspberry Pi a través del escritorio, para acceder a algunos de ellos hay que usar un método basado en texto, denominado *interfaz de línea de comandos (CLI)*, en una aplicación llamada Terminal. La mayoría de los usuarios no necesitarán usar la CLI, pero para quienes quieran aprender más, este apéndice ofrece una introducción básica.

MÁS INFORMACIÓN

Este apéndice no pretende ser una guía exhaustiva de la interfaz de línea de comandos de Linux. Para obtener más detalles sobre el uso de la CLI, visita **rpf.io/terminal** en un navegador web.

Cargar la aplicación Terminal

El acceso a la CLI es a través del Terminal, un paquete de software que carga lo que técnicamente se conoce como *terminal de teletipo virtual (VTY)*, nombre que se remonta a los primeros tiempos de los ordenadores, cuando los usuarios introducían comandos mediante una gran máquina de escribir electromecánica, en lugar de un teclado y un monitor. Para cargar el paquete Terminal, haz clic en el icono de Raspberry Pi para cargar el menú, elige la categoría Accesorios y haz clic en LXTerminal.



La ventana de Terminal se puede arrastrar por el escritorio, redimensionar, maximizar y minimizar como cualquier otra ventana. También puedes agrandar el texto en ella, si es difícil de ver, o reducirlo para que quepa más en la ventana: haz clic en el menú Edit y elige Zoom In o Zoom Out respectivamente. O mantén pulsada la tecla **CTRL** en el teclado y luego pulsa **+** o **-**.

El símbolo del sistema

Lo primero que se ve en un Terminal es el *símbolo del sistema*, a la espera de tus instrucciones. El símbolo de un Raspberry Pi que ejecuta el sistema operativo Raspberry Pi es:

```
pi@raspberrypi:~ $
```

La primera parte, **`pi`**, es tu nombre de usuario. La segunda, después de **`@`**, es el nombre del ordenador, **`raspberrypi`** de forma predeterminada. Después de "**`:`**" hay una "tilde", **`~`**, que es una referencia abreviada a tu directorio de inicio y representa tu *directorio de trabajo actual*. Por último, el símbolo **`$`** indica que el usuario es un *usuario sin privilegios* por lo que necesitas una contraseña para llevar a cabo tareas como añadir o quitar software.

Navegación

Escribe lo siguiente y pulsa la tecla **ENTRAR**:

```
cd Desktop
```

Verás que el símbolo del sistema cambia inmediatamente a:

```
pi@raspberrypi:~/Desktop $
```

Esto indica que tu directorio de trabajo actual ha cambiado: antes estabas en el directorio de inicio (indicado por **~**) y ahora estás en el subdirectorio **Desktop**, bajo el directorio de inicio. Para eso has usado el comando **cd**, que significa *cambiar de directorio*.

MAYÚSCULAS Y MINÚSCULAS

La interfaz de línea de comandos del sistema operativo Raspberry Pi distingue entre mayúsculas y minúsculas, lo que significa que es importante fijarse en cómo se escriben los comandos o los nombres. Si, al intentar cambiar de directorio, recibes un mensaje indicando que no existe tal archivo o directorio, comprueba que hayas escrito en mayúscula la D al inicio de Desktop.

Hay cuatro maneras de regresar a tu directorio de inicio: pruébalas una por una y regresa de nuevo al subdirectorio **Desktop** entre una prueba y otra. La primera es:

```
cd ..
```

.. es otro método abreviado y esta vez significa "el directorio encima de este", también conocido como el *directorio principal*. Dado que el directorio encima de **Desktop** es tu directorio de inicio, regresas ahí. Vuelve al subdirectorio **Desktop** e intenta el segundo método:

```
cd ~
```

El símbolo **~** utilizado aquí significa "cambiar a mi directorio de inicio". A diferencia de **cd ..**, que solo te lleva al directorio principal de cualquier directorio en el que estés, este comando funciona desde cualquier lugar. Pero hay una forma más fácil:

```
cd
```

Sin proporcionarle el nombre de un directorio, de forma predeterminada **cd** simplemente regresa a tu directorio de inicio. El último método para regresar a tu directorio de inicio es escribiendo:

```
cd /home/pi
```

Esto utiliza una *ruta absoluta*, que funcionará independientemente de cuál sea el directorio de trabajo actual. Al igual que **cd** o **cd ~**, te llevará de nuevo a tu directorio de inicio desde donde te encuentres. Pero, a diferencia de los otros métodos, necesita que proporciones tu nombre de usuario.

Gestionar los archivos

Para practicar con el uso de archivos, cambia a **Desktop** y escribe:

```
touch Test
```

Verás un archivo llamado **Test** en el escritorio. El comando **`touch`** se suele utilizar para actualizar la información de fecha y hora de un archivo. Pero en casos como este, en que no existe un archivo, lo crea.

Escribe lo siguiente:

```
cp Test Test2
```

Verás aparecer en el escritorio otro archivo, **Test2**. Es una *copia* del archivo original e idéntico a él. Bórrala escribiendo:

```
rm Test2
```

Así se *elimina* el archivo y verás que desaparece.

> **¡ADVERTENCIA!**
>
> A diferencia de la eliminación de archivos mediante el Gestor de archivos gráfico, que los almacena en la papelera para su posterior recuperación, los archivos eliminados mediante **`rm`** desaparecen para siempre. ¡Escribe con mucho cuidado!

A continuación, escribe:

```
mv Test Test2
```

Este comando *mueve* el archivo y verás que el archivo **Test** original desaparece y se sustituye por **Test2**. El comando de movimiento **`mv`** se puede usar así para cambiar nombres de archivo.

Pero también necesitarás ver qué archivos contiene un directorio cuando no estés en el escritorio. Escribe:

```
ls
```

Este comando *crea una lista* del contenido del directorio actual, o de cualquier otro que tú indiques. Para aprender más cosas, como la inclusión de archivos ocultos en listas y la visualización del tamaño de los archivos, puedes añadir modificadores:

```
ls -larth
```

Estos modificadores controlan el comando **ls**: **l** organiza el resultado en forma de lista larga en vertical; **a** le indica que muestre todos los archivos y directorios, incluidos los que suelen estar ocultos; **r** invierte la ordenación normal; **t** ordena los elementos por fecha de modificación, lo que combinado con **r** te da los campos más antiguos en la parte superior de la lista, y los más nuevos en la parte inferior. Y **h** usa tamaños de archivo legibles, para hacer que la lista sea más fácil de entender.

Ejecutar programas

Algunos programas solo se pueden ejecutar en la línea de comandos, mientras que otros tienen interfaces tanto gráficas como de línea de comandos. Entre estos últimos está la herramienta Configuración de Raspberry Pi, que normalmente se carga desde el menú de iconos de Raspberry.

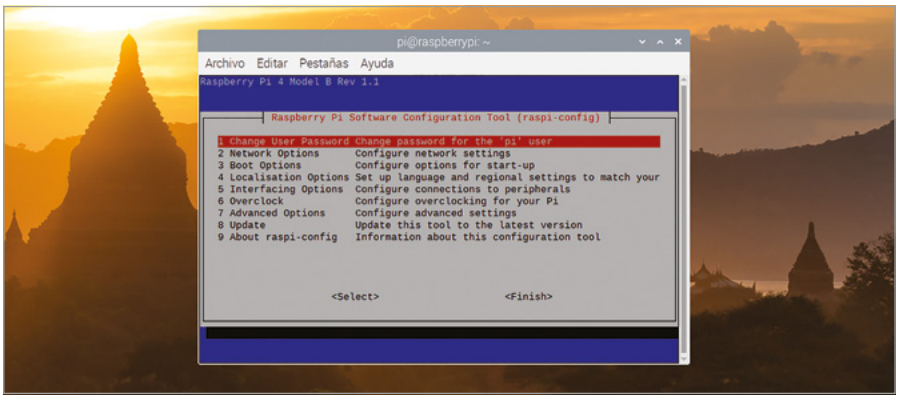
Escribe:

```
raspi-config
```

Un mensaje de error indica que el software solo puede ejecutarse como *raíz*, la cuenta de superusuario en tu Raspberry Pi. También te dirá cómo hacerlo, escribiendo el comando:

```
sudo raspi-config
```

sudo es la parte que indica el *cambio de usuario* y le dice al sistema operativo Raspberry Pi que ejecute el comando como usuario raíz.



Solo tendrás que usar **sudo** cuando un programa requiera *privilegios* "elevados", por ejemplo, para instalar o desinstalar software o para ajustar la configuración del sistema. Un juego, por ejemplo, nunca se debería ejecutar usando **sudo**.

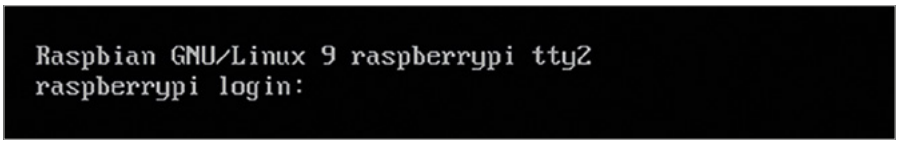
Pulsa dos veces la tecla **TAB** para seleccionar Finish y pulsa **ENTRAR** para salir de la herramienta Configuración de Raspberry Pi y regresar a la interfaz de línea de comandos. Por último, escribe:

```
exit
```

Esto terminará la sesión de la interfaz de línea de comandos y cerrará la aplicación Terminal.

Usar TTY

La aplicación Terminal no es la única forma de usar la interfaz de línea de comandos: también puedes cambiar a uno de los terminales ya en ejecución denominados *teletipos* o *TTY*. Mantén pulsadas las teclas **CTRL** y **ALT** del teclado y pulsa la tecla **F2** para cambiar a "tty2".



Tendrás que volver a iniciar sesión con tu nombre de usuario y contraseña, después de lo cual podrás usar la interfaz de línea de comandos igual que en el Terminal. El uso de estos TTY es útil cuando, por cualquier razón, la interfaz principal del escritorio no funciona.

Para dejar de usar TTY, mantén pulsadas **CTRL+ALT** y luego pulsa **F7**: el escritorio volverá a aparecer. Pulsa **CTRL+ALT+F2** y volverás a cambiar a "tty2": cualquier cosa que estuvieras ejecutando seguirá estando ahí.

Antes de cambiar de nuevo, escribe:

```
exit
```

Luego pulsa **CTRL+ALT+F7** para volver al escritorio. Debes salir antes de cambiar de TTY porque cualquiera con acceso al teclado puede cambiar a un TTY y, si todavía estás conectado, ¡podrán acceder a tu cuenta aunque no sepan tu contraseña!

Enhorabuena: has dado tus primeros pasos con la interfaz de línea de comandos del sistema operativo de Raspberry Pi.

Apéndice D

Otro material de referencia

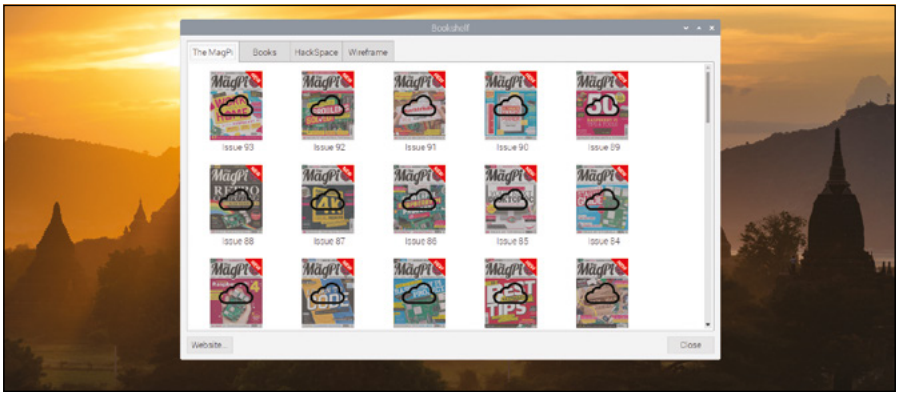
La guía oficial de Raspberry Pi para principiantes tiene la finalidad de ayudarte a empezar a usar tu Raspberry Pi, pero no trata todas las posibilidades que te ofrece. La numerosa comunidad de Raspberry Pi se expande por el mundo y estos ordenadores se utilizan para todo tipo de cosas: juegos, aplicaciones de detección, robótica, inteligencia artificial. Hay multitud de fuentes de inspiración.

En este apéndice se detallan algunas fuentes de ideas de proyectos, planes de lecciones y otros materiales que te servirán para avanzar, ahora que ya te has abierto paso con la *Guía para principiantes*.

Bookshelf

Menú Raspberry > Help > Bookshelf

Bookshelf es una aplicación incluida con el sistema operativo Raspberry Pi OS que te ayuda navegar, descargar y leer versiones digitales de publicaciones de Raspberry Pi Press, incluida esta Guía para principiantes. Para cargarla basta con que hagas clic en el menú de iconos de Raspberry, elijas Help y hagas clic en Bookshelf. Luego explora una serie de revistas y libros que puedes descargar gratuitamente y leer cuando te vaya bien.



El blog de Raspberry Pi

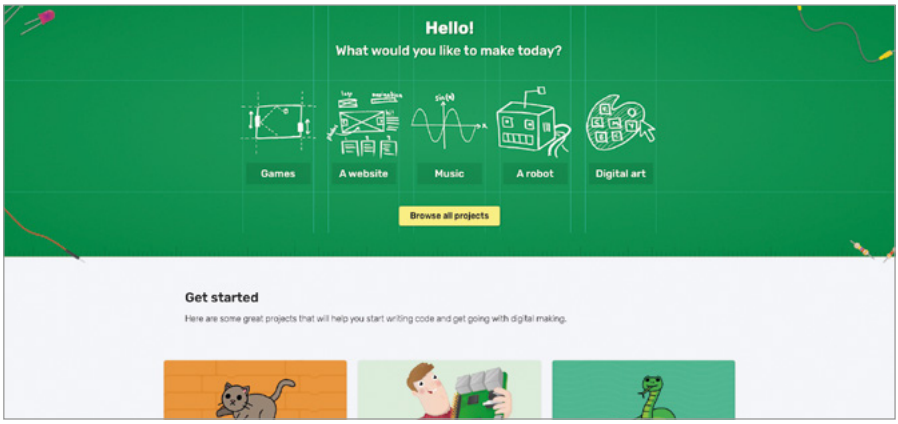
rpf.io/blog

El blog oficial debería ser tu primer recurso para obtener noticias de máxima actualidad sobre todo lo relacionado con Raspberry Pi: desde nuevos lanzamientos de hardware y material educativo hasta resúmenes de los mejores proyectos, campañas e iniciativas de la comunidad. Si quieres estar al día en todo lo relacionado con Raspberry Pi, aquí es donde tienes que mirar.

Proyectos en Raspberry Pi

rpf.io/projects

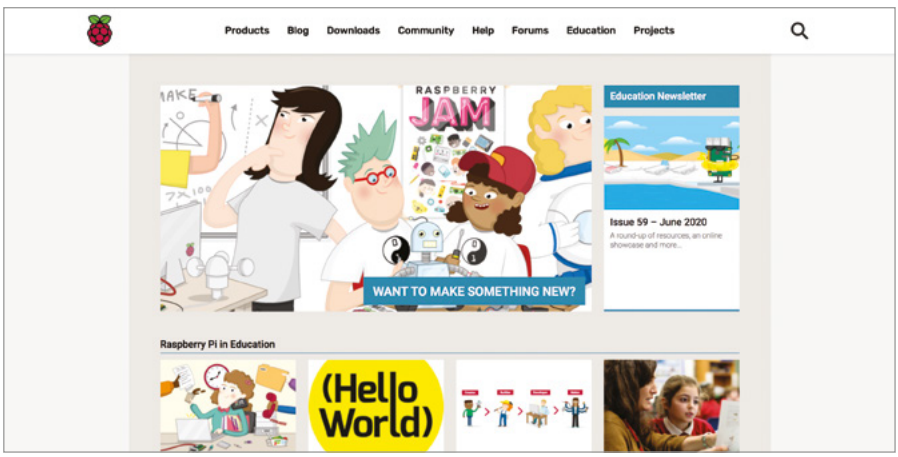
El sitio oficial de proyectos de Raspberry Pi ofrece tutoriales de proyectos paso a paso en una serie de categorías, desde la creación de juegos y música hasta la construcción de tu propio sitio web o un robot programado con Raspberry Pi. La mayoría de los proyectos también están disponibles en diversos idiomas y abarcan una gama de niveles de dificultad adecuados para todo tipo de usuarios, desde los principiantes absolutos a los creadores veteranos.



Educación en Raspberry Pi

rpf.io/education

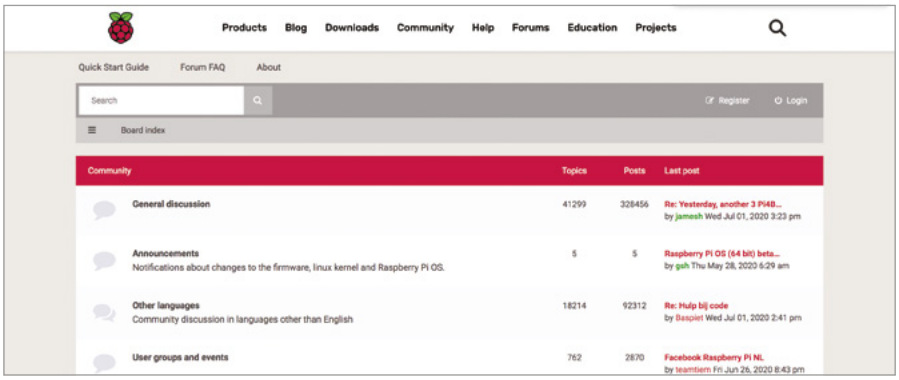
El sitio oficial de educación de Raspberry Pi ofrece boletines de noticias, formación online y proyectos destinados a educadores. También tiene enlaces con recursos adicionales, como el programa de capacitación Picademy, los programas de codificación Code Club y CoderDojo en los que colaboran muchos voluntarios y los eventos mundiales de Raspberry Jam.



Foros de Raspberry Pi

rpf.io/forums

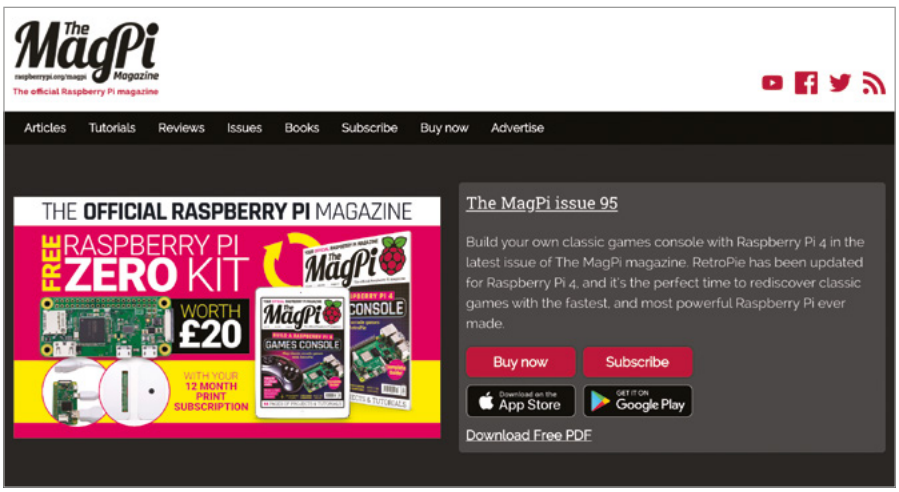
Los foros de Raspberry Pi son el punto de encuentro de los seguidores de Raspberry Pi para hablar de todo tipo de temas, tanto a nivel básico como altamente técnico. Incluso hay un área para charlas de carácter general, sin un tema concreto.



La revista The MagPi

magpi.cc

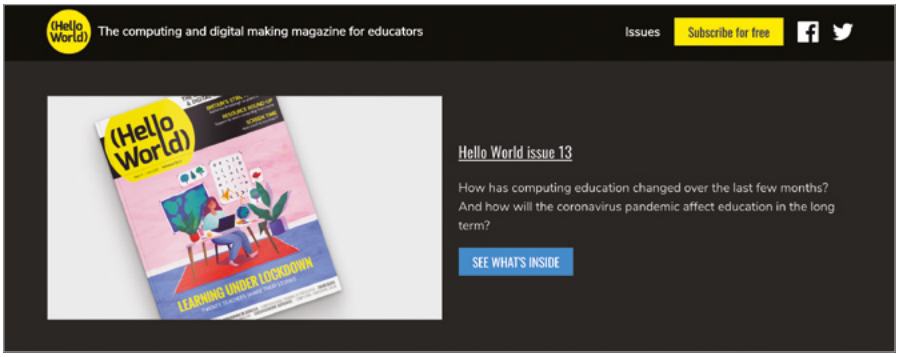
La revista oficial de Raspberry Pi, The MagPi, es una publicación mensual que cubre desde tutoriales y guías a reseñas y noticias, y cuenta con abundantes contribuciones de la comunidad mundial de Raspberry Pi. Hay copias disponibles en quioscos y supermercados selectos y también se pueden descargar copias digitales de forma gratuita bajo la licencia Creative Commons. The MagPi también publica libros y material híbrido sobre diversos temas, que es posible comprar en formato impreso o descargar gratuitamente.



La revista Hello World

helloworld.cc

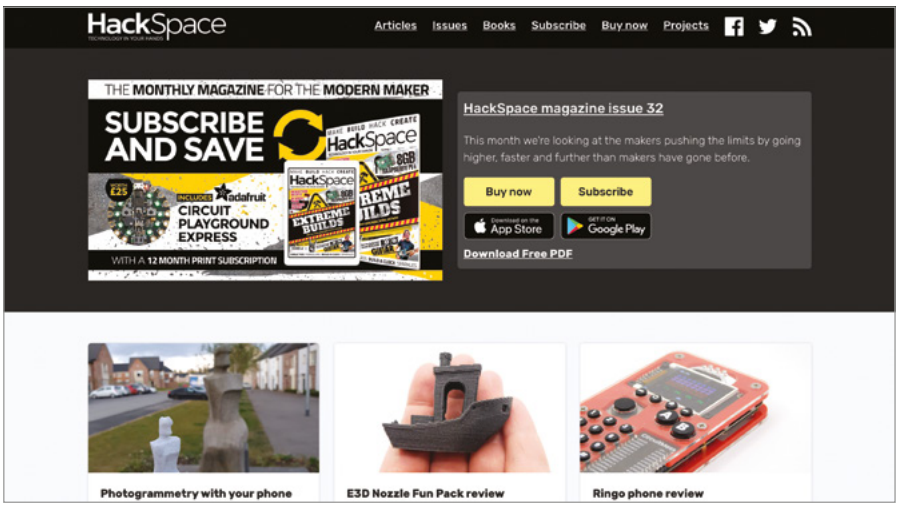
Publicada tres veces al año, Hello World está disponible de forma gratuita en el Reino Unido para profesores, voluntarios y bibliotecarios. Otras personas pueden descargar copias digitales gratuitas bajo la licencia Creative Commons y hay suscripciones a la versión impresa disponibles comercialmente.



La revista HackSpace

hsmag.cc

Dirigida a un público más especializado que The MagPi, la revista HackSpace se centra en la comunidad de creadores, con reseñas de hardware y software, tutoriales y entrevistas. Si te interesa ampliar tus horizontes más allá de Raspberry Pi, la revista HackSpace es un buen punto de partida: se puede encontrar en formato impreso en supermercados y quioscos de prensa o se puede descargar de forma gratuita en formato digital.



Apéndice E

Herramienta Configuración de Raspberry Pi

La herramienta Configuración de Raspberry Pi es un eficaz paquete para realizar diversos ajustes en tu Raspberry Pi, aplicables por ejemplo a interfaces disponibles, programas o el modo de control a través de una red. Como sabemos que puede resultar complicada para los principiantes, este apéndice te guiará a través de cada una de sus secciones, explicando sus propósitos.

Puedes cargar la herramienta Configuración de Raspberry Pi desde el menú de iconos de Raspberry, en la categoría Preferencias. También se puede ejecutar desde la interfaz de línea de comandos o la aplicación Terminal, mediante el comando **`raspi-config`**. La versión de línea de comandos y la versión gráfica son diferentes entre sí: las opciones aparecen en distintas categorías, según la versión que se utilice. Este apéndice se basa en la versión gráfica.

¡ADVERTENCIA!

A menos que tengas la certeza de que debes cambiar un ajuste concreto, es mejor no modificar nada en la herramienta Configuración de Raspberry Pi. Si vas a añadir nuevo hardware a tu Raspberry Pi (por ejemplo, una placa HAT de audio o un módulo de cámara), las instrucciones te dirán qué ajuste debes cambiar. De lo contrario, se debería mantener la configuración predeterminada.

Pestaña Sistema

La pestaña Sistema contiene opciones que controlan varios ajustes del sistema operativo Raspberry Pi.

- **Clave:** Haz clic en el botón "Cambiar Clave..." para establecer una nueva para tu cuenta de usuario actual. La predeterminada es "pi".

- **Hostname:** El nombre con el que un Raspberry Pi se identifica en las redes. Si tienes más de un Raspberry Pi en la misma red, cada uno de ellos debe tener un nombre exclusivo.

- **Iniciar en:** Si la opción seleccionada es "Escritorio" (la predeterminada) se carga el escritorio habitual del sistema operativo Raspberry Pi. Si se selecciona Consola, se carga la interfaz de línea de comandos, que se describe en el **Apéndice C, La interfaz de línea de comandos**.

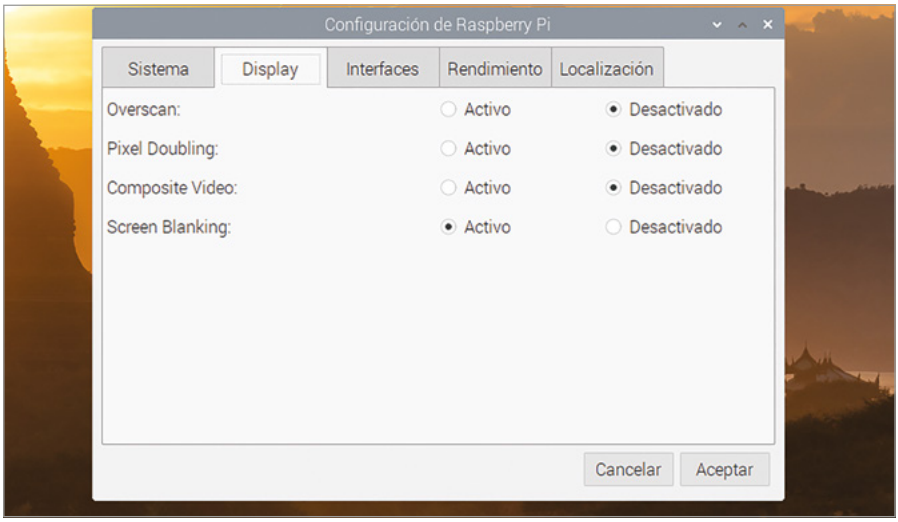
- **Ingreso automático:** Cuando la opción seleccionada es "Ingresar como usuario pi" (opción predeterminada), el sistema operativo Raspberry Pi cargará el escritorio sin necesidad de escribir el nombre de usuario y la clave.

- **Red al iniciar:** Cuando está seleccionada la opción "Esperar red", el sistema operativo Raspberry Pi no se cargará hasta que tenga una conexión de red que funcione.

- **Splash Screen:** Cuando la opción seleccionada es "Activo" (la predeterminada), los mensajes de inicio del sistema operativo Raspberry Pi quedan ocultos tras una pantalla de presentación gráfica.

Pestaña Display

La pestaña "Display" contiene los ajustes que controlan cómo se ve la pantalla.



- **Overscan:** Este ajuste controla si la salida de vídeo en Raspberry Pi incluye o no barras negras alrededor de los bordes, para compensar el marco de muchos televisores. Si ves barras negras, selecciona la opción "Desactivado". Si no, mantén la opción "Activo".

- **Pixel Doubling:**Si utilizas una pantalla de alta resolución pero de tamaño pequeño, puedes activar la función de duplicación de píxeles para que todo lo que aparece en la pantalla sea más grande y fácil de ver.

- **Composite Video:** Este ajuste controla la salida de vídeo compuesto disponible en la toma combinada de audio y vídeo (AV), cuando se utiliza con un adaptador TRRS (punta-anillo-anillo-cuerpo). Si quieres utilizar la salida de vídeo compuesto en lugar de HDMI, selecciona la opción "Activo". Si no, deja el ajuste desactivado.

- **Screen Blanking:** Este ajuste permite activar y desactivar la visualización del contenido de la pantalla (el tiempo de espera (en minutos) tras el cual la pantalla queda "en blanco").

Pestaña Interfaces

La pestaña Interfaces contiene los ajustes que controlan las interfaces de hardware disponibles en Raspberry Pi.

- **Cámara:** Activa o desactiva la interfaz CSI (Camera Serial Interface) de la cámara, para su uso con un Módulo de cámara de Raspberry Pi.

- **SSH:** Activa o desactiva la interfaz SSH (Secure Shell). Permite abrir una interfaz de línea de comandos en Raspberry Pi desde otro ordenador de tu red utilizando un cliente SSH.

- **VNC:** Activa o desactiva la interfaz VNC (Virtual Network Computing). Permite ver el escritorio de Raspberry Pi desde otro ordenador de tu red usando un cliente VNC.

- **SPI:** Activa o desactiva la interfaz SPI (Serial Peripheral Interface), usada para controlar algunos complementos de hardware que se conectan a los pines GPIO.

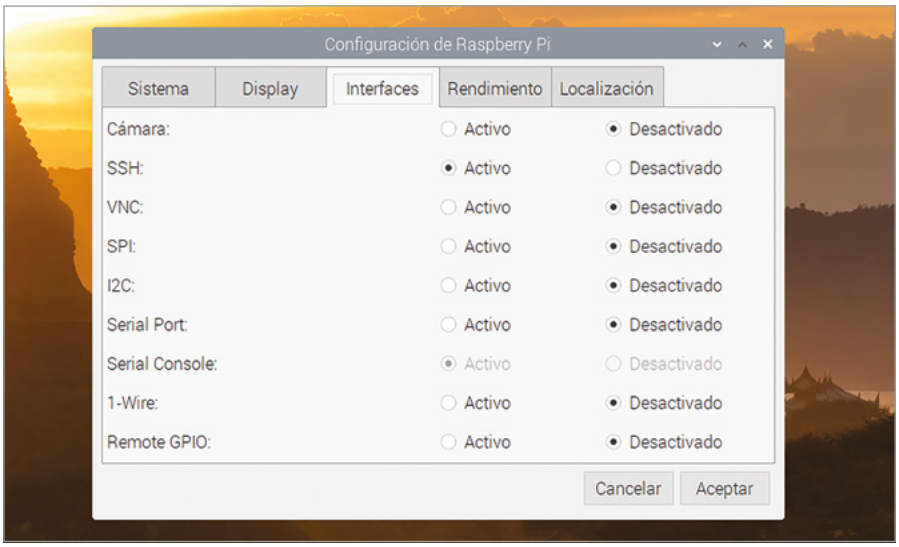
- **I2C:** Activa o desactiva la interfaz I²C (Inter-Integrated Circuit), usada para controlar algunos complementos de hardware que se conectan a los pines GPIO.

- **Serial Port:** Activa o desactiva el puerto serie de Raspberry Pi, disponible en los pines GPIO.

- **Serial Console:** Activa o desactiva la consola serie, una interfaz de línea de comandos disponible en el puerto serie. Esta opción solo está disponible si la opción seleccionada para el ajuste de Serial Port es Activo.

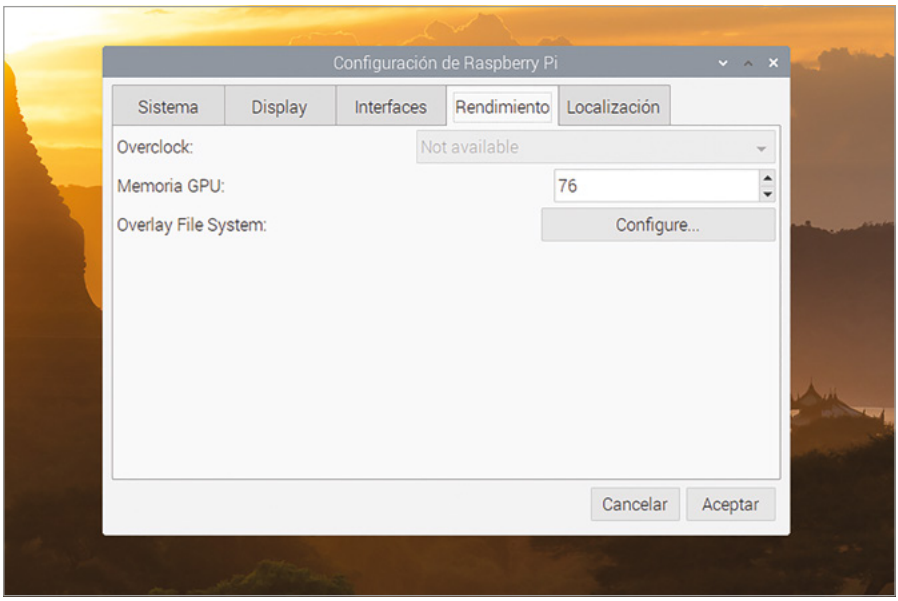
- **1-Wire:** Activa o desactiva la interfaz 1-Wire, usada para controlar algunos complementos de hardware que se conectan a los pines GPIO.

- **Remote GPIO:** Activa o desactiva un servicio de red que permite controlar los pines GPIO de Raspberry Pi desde otro ordenador de tu red usando la biblioteca GPIO Zero. Hay más información sobre el sistema GPIO remoto, disponible en **gpiozero.readthedocs.io**.



Pestaña Rendimiento

La pestaña Rendimiento contiene ajustes que controlan la cantidad de memoria disponible y la velocidad de ejecución del procesador de Raspberry Pi.



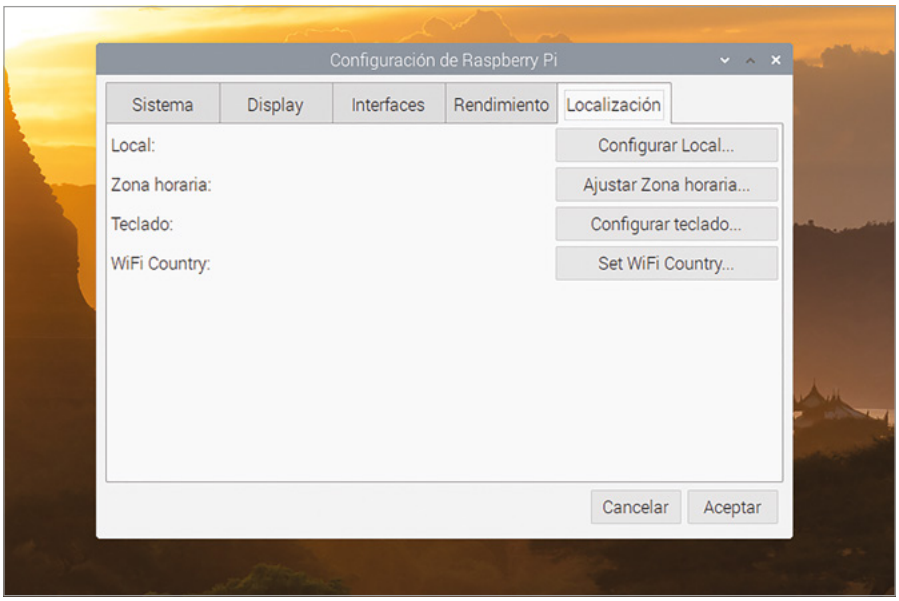
- **Overclock:** Te permite elegir entre una serie de opciones que aumentan el rendimiento de tu Raspberry Pi, a costa de un incremento del uso de energía y del calor generado, y la posibilidad de que se reduzca la vida útil total del ordenador. Este ajuste no está disponible en todos los modelos de Raspberry Pi.

- **Memoria GPU:** Te permite establecer la cantidad de memoria reservada para el uso del procesador de gráficos de Raspberry Pi. Los valores superiores al predeterminado pueden mejorar el rendimiento para tareas complicadas de procesamiento 3D y GPU de propósito general (GPGPU), a costa de reducir la memoria disponible en el sistema operativo Raspberry Pi. Los valores inferiores pueden mejorar el rendimiento para tareas que requieren mucha memoria, a costa de hacer que las funciones de procesamiento 3D, cámara y reproducción de vídeo seleccionadas funcionen más lentamente o no estén disponibles.

- **Overlay File System:** Permite bloquear el sistema de archivos de Raspberry Pi para que los cambios solo se hagan en un disco RAM virtual, en lugar de registrarse en la tarjeta microSD. Por lo tanto, cada vez que se reinicia, el ordenador está en estado limpio.

Pestaña Localización

La pestaña Localización contiene los ajustes que controlan la región en la que Raspberry Pi debe funcionar, entre ellos los ajustes de distribución del teclado.



- **Local:** Te permite elegir tu configuración local, ajustes del sistema entre los que están el idioma, el país y el conjunto de caracteres. Ten en cuenta que el cambio de idioma aquí solo cambiará el idioma mostrado en las aplicaciones para las que haya una traducción disponible.

- **Zona horaria:** Te permite elegir el huso horario de tu ubicación, seleccionando una zona del mundo seguida de la ciudad más cercana a tu ubicación. Si tu Raspberry Pi está conectado a la red pero el reloj muestra la hora equivocada, normalmente se debe a una selección de zona horaria errónea.

- **Teclado:** Te permite elegir el tipo de teclado, el idioma y la distribución. Si tu teclado escribe las letras o símbolos erróneos, puedes corregirlo aquí.

- **WiFi Country:** Te permite seleccionar tu país para fines de reglamento de radiocomunicaciones. Asegúrate de seleccionar el país en el que utilices tu Raspberry Pi: si seleccionas otro país, podría ser imposible conectar con los puntos de acceso WiFi cercanos y puede constituir una infracción de las leyes de radiodifusión. Debes seleccionar un país antes de usar la radio WiFi.

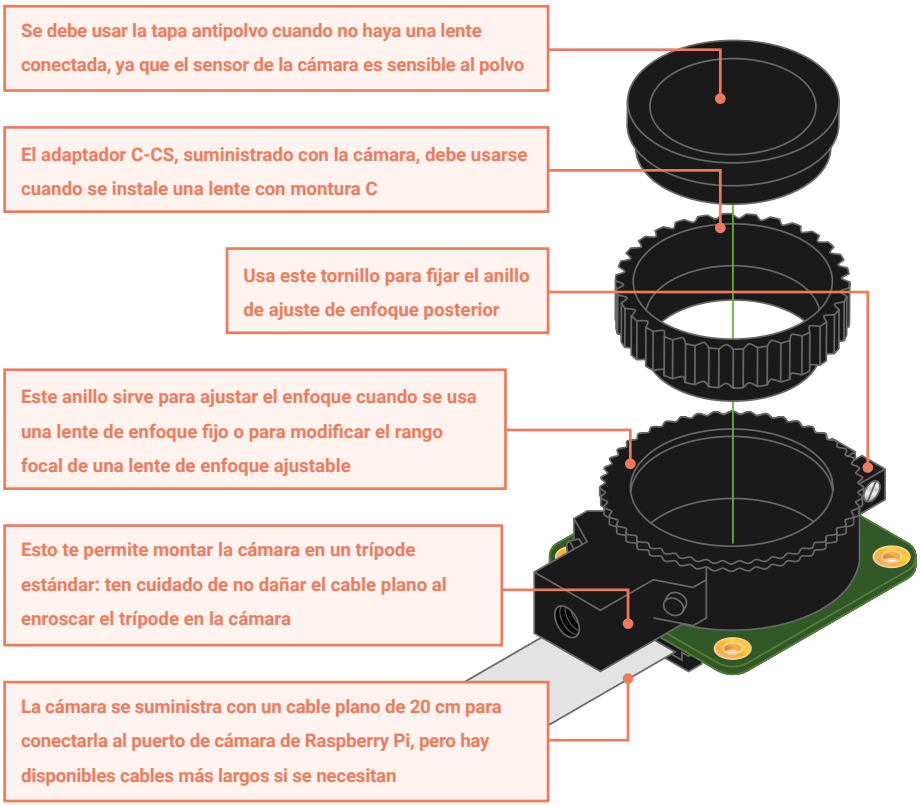
Apéndice F

Configuración de la High Quality Camera

La High Quality Camera (cámara de alta calidad) puede capturar imágenes de mayor resolución que el Módulo de cámara estándar. A diferencia de este último, no tiene una lente ya instalada. Puede utilizarse con cualquier lente estándar con montura C o CS. Para facilitarte las cosas, puedes comprar lentes de 6 mm y 16 mm al comprar la cámara.

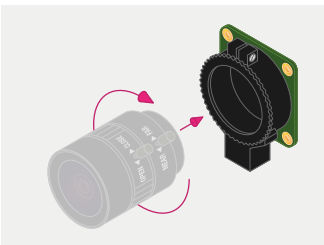
Lente de 6 mm con montura CS

Hay disponible una lente de bajo coste de 6 mm para la HQ Camera. Es una lente adecuada para la fotografía básica. También se puede usar para la macrofotografía, ya que puede enfocar objetos a distancias muy cortas.



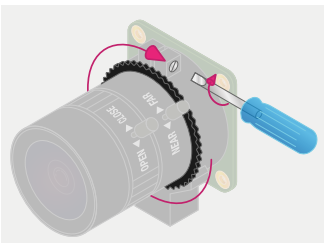
01 Acoplar la lente

La lente de 6 mm es un dispositivo con montura CS, por lo que no necesita el anillo adaptador C-CS (ver diagrama anterior). Si se utiliza el adaptador, no enfocará correctamente. Por lo tanto, quítalo si está puesto. Luego gira la lente en el sentido de las agujas del reloj para que encaje totalmente en el anillo de ajuste de enfoque posterior.



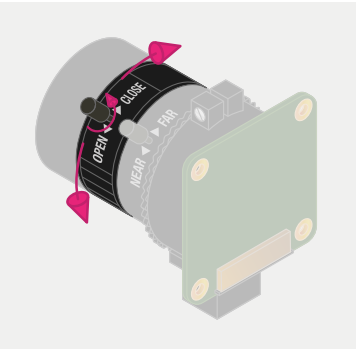
02 Anillo de ajuste de enfoque posterior y tornillo de bloqueo

El anillo de ajuste de enfoque posterior debe enroscarse completamente para que la longitud de enfoque posterior sea lo menor posible. Utiliza el tornillo de bloqueo de enfoque posterior para asegurarte de que no se salga de esta posición al ajustar la apertura o el enfoque.



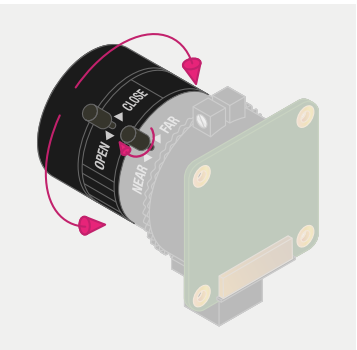
03 Apertura

Para ajustar la apertura, sostén la cámara con la lente orientada en dirección opuesta a ti. Gira el anillo central mientras sostienes firmemente el anillo exterior (el más alejado de la cámara). Gira en el sentido de las agujas del reloj para cerrar la apertura y reducir el brillo de la imagen. Gira en el sentido contrario a las agujas del reloj para abrir la apertura. Cuando el nivel de luz sea el deseado, aprieta el tornillo del lado de la lente para bloquear la apertura.



04 Enfoque

Primero, fija la posición del anillo de enfoque interno, etiquetado como "NEAR ◀▶ FAR", apretando el tornillo. Sostén la cámara con la lente orientada en dirección opuesta a ti. Sostén los dos anillos exteriores de la lente y gíralos en el sentido de las agujas del reloj hasta que la imagen quede enfocada. Para ajustar el enfoque, gira los dos anillos exteriores en el sentido de las agujas del reloj para enfocar un objeto cercano. Gíralos en sentido contrario a las agujas del reloj para enfocar un objeto distante. Es posible que después de eso tengas que volver a ajustar la apertura.

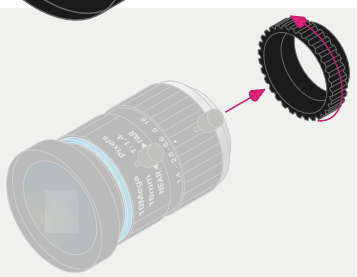


Lente de 16 mm con montura CS

La lente de 16 mm proporciona una imagen de mayor calidad que la de 6 mm. Tiene un ángulo de visión estrecho que es más adecuado para ver objetos distantes.

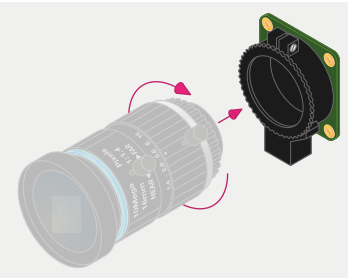
01 Acoplar el adaptador C-CS

El adaptador C-CS suministrado con la HQ Camera debe acoplarse a la lente de 16 mm. La lente es un dispositivo de montaje C, por lo que tiene un enfoque posterior más largo que el de la lente de 6 mm y por lo tanto requiere el adaptador.



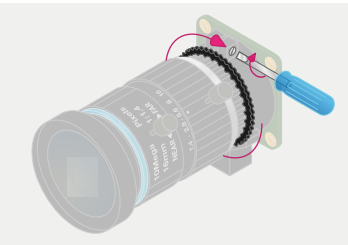
02 Acoplar la lente a la cámara

Gira la lente de 16 mm y el adaptador C-CS en el sentido de las agujas del reloj para que encajen en el anillo de ajuste de enfoque posterior.



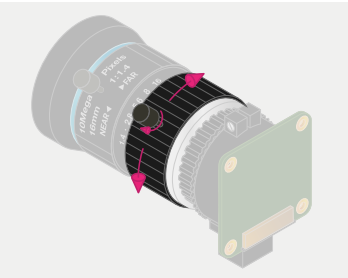
03 Anillo de ajuste de enfoque posterior y tornillo de bloqueo

El anillo de ajuste de enfoque posterior se debe enroscar completamente. Utiliza el tornillo de bloqueo de enfoque posterior para asegurarte de que no se salga de esta posición al ajustar la apertura o el enfoque.



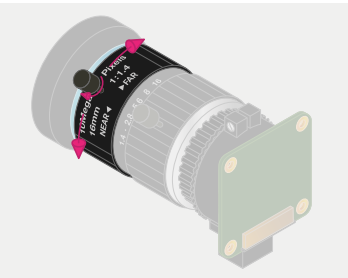
04 Apertura

Para ajustar la apertura, sostén la cámara con la lente orientada en dirección opuesta a ti. Gira el anillo interior (el más cercano a la cámara), mientras sostienes la cámara firmemente. Gira en el sentido de las agujas del reloj para cerrar la apertura y reducir el brillo de la imagen. Gira en el sentido contrario a las agujas del reloj para abrir la apertura. Cuando el nivel de luz sea el deseado, aprieta el tornillo del lado de la lente para bloquear la apertura.



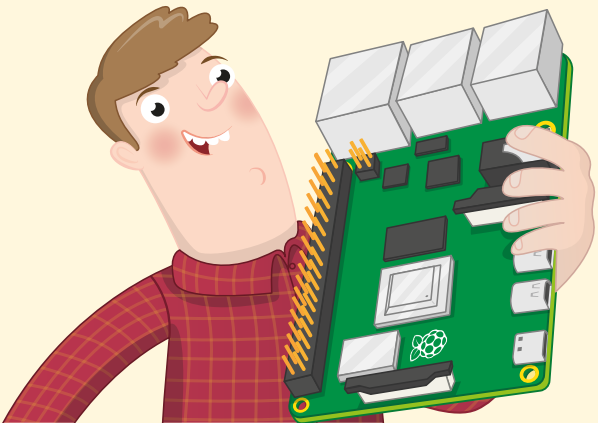
05 Enfoque

Para ajustar el enfoque, sostén la cámara con la lente orientada en dirección opuesta a ti. Gira en sentido contrario a las agujas del reloj el anillo de enfoque (etiquetado "NEAR ◀▶ FAR") para enfocar un objeto cercano. Gíralo en el sentido de las agujas del reloj para enfocar un objeto distante. Es posible que después de eso tengas que volver a ajustar la apertura.



Apéndice G

Especificaciones de Raspberry Pi

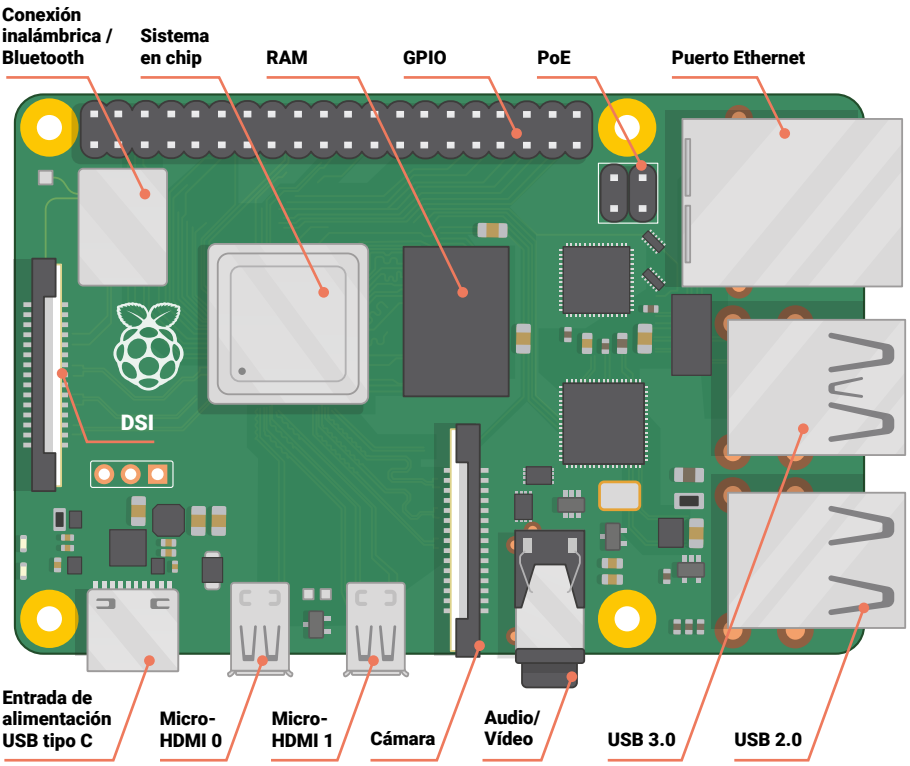


Los diversos componentes y características de un ordenador son lo que se conoce como sus especificaciones y al examinarlas nos proporcionan información útil para comparar dos ordenadores. Estas especificaciones pueden parecer confusas al principio y son de carácter muy técnico; no es necesario que las entiendas para usar un Raspberry Pi, pero las incluimos en esta guía a beneficio de los lectores más curiosos.

El sistema en chip de los ordenadores Raspberry Pi 4 Model B y Raspberry Pi 400 es un Broadcom BCM2711B0, como verás en la tapa metálica si la miras de cerca (en Raspberry Pi 4). Cuenta con una unidad central de procesamiento (CPU) ARM Cortex-A72 de 64 bits con cuatro núcleos, cada uno de ellos a 1,5 o 1,8 GHz (1,5 o 1,8 mil millones de ciclos por segundo); y una unidad de procesamiento gráfico (GPU) Broadcom VideoCore VI (seis) que funciona a 500 MHz (500 millones de ciclos por segundo) para tareas de vídeo y de procesamiento 3D, como las de los juegos.

El sistema en chip está conectado a 2 GB, 4 GB u 8 GB (dos, cuatro u ocho mil millones de bytes) —4 GB en Raspberry Pi 400— de memoria de acceso aleatorio RAM LPDDR4 (Low-Power Double-Data-Rate 4), que funciona a 3200 MHz (tres mil doscientos millones de ciclos por segundo). Esta memoria se comparte entre el procesador central y el procesador de gráficos. La ranura de la tarjeta microSD admite hasta 512 GB (512 mil millones de bytes) de almacenamiento.

El puerto Ethernet admite conexiones de gigabit (1000 Mbps, 1000-Base-T), mientras que la radio admite redes WiFi 802.11ac que funcionan en las bandas de frecuencia de 2,4 GHz y 5 GHz y conexiones Bluetooth 5.0 y BLE (Bluetooth de bajo consumo).



Esta es la lista de especificaciones de Raspberry Pi 4:

- **CPU:** ARM Cortex-A72 de 64 bits y cuatro núcleos a 1,5 GHz
- **GPU:** VideoCore VI a 500 MHz
- **RAM:** 1 GB, 2 GB o 4 GB de LPDDR4
- **Red:** Gigabit Ethernet, doble banda 802.11ac, Bluetooth 5.0, Bluetooth de bajo consumo (BLE)
- **Salidas de audio y vídeo:** Toma analógica AV de 3,5 mm, 2 × micro-HDMI 2.0
- **Conectividad de periféricos:** 2 × puertos USB 2.0, 2 × puertos USB 3.0, interfaz serie de cámara, Interfaz serie de pantalla (DSI)
- **Almacenamiento:** microSD, hasta 512 GB
- **Alimentación:** 5 voltios a 3 amperios a través de USB Tipo C
- **Extras:** sistema GPIO de 40 pines, compatibilidad con Power over Ethernet (con hardware adicional)

Las especificaciones de Raspberry Pi 400 son:

- **CPU:** ARM Cortex-A72 de 64 bits y cuatro núcleos a 1,8 GHz
- **GPU:** VideoCore VI a 500 MHz
- **RAM:** 4 GB de LPDDR4
- **Red:** Gigabit Ethernet, doble banda 802.11ac, Bluetooth 5.0, Bluetooth de bajo consumo (BLE)
- **Salidas de audio y vídeo:** 2 × micro-HDMI 2.0
- **Conectividad de periféricos:** 1 × puerto USB 2.0, 2 × puertos USB 3.0
- **Almacenamiento:** microSD, hasta 512 GB (se suministran 16 GB)
- **Alimentación:** 5 voltios a 3 amperios a través de USB Tipo C
- **Extras:** Sistema GPIO de 40 pines

La operación de este equipo está sujeta

Apéndice H
Raspberry Pi Guía de seguridad y uso

Raspberry Pi

Diseñado y distribuido por
Raspberry Pi Trading Ltd
Maurice Wilkes Building
Cowley Road
Cambridge
CB4 0DS
REINO UNIDO
www.raspberrypi.org

Raspberry Pi Información sobre el cumplimiento normativo y la seguridad

Raspberry Pi 4 Model B
FCC: 2ABCB-RPI4B
IC: 20953-RPI4B

Raspberry Pi 400
FCC: 2ABCB-RPI400
IC: 20953-RPI400

IMPORTANTE: Antes de conectar el dispositivo a una toma de corriente, lee las instrucciones de instalación en **www.raspberrypi.org/safety**

ADVERTENCIA: Cáncer y daños al sistema reproductor - www.P65Warnings.ca.gov.

Toda la información reglamentaria y los certificados se encuentran en **www.raspberrypi.org/compliance**

IFETEL: 2019LAB-ANCE4957
Número de certificado de Raspberry Pi 4 Model B.

Número de registro en TRA
ER73381/19
Número de certificado de Raspberry Pi 4 Model B.

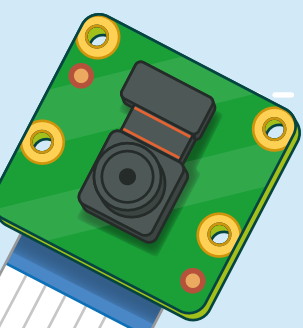
CCAO19LP1120T2
Número de certificado de Raspberry Pi 4 Model B.

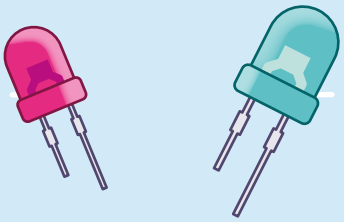
NTC
Tipo aprobado:
N.°: ESD-GEC-1920098C
Número de certificado de Raspberry Pi 4 Model B.

APROBACIÓN TA-2019/750
Número de certificado de Raspberry Pi 4 Model B.

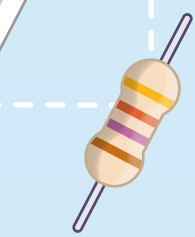
Las marcas comerciales adoptadas HDMI, HDMI High-Definition Multimedia Interface y el logotipo HDMI son marcas comerciales o marcas comerciales registradas de HDMI Licensing Administrator, Inc. en los Estados Unidos y otros países.

La operación de este equipo está sujeta a las siguientes dos condiciones:
(1) es posible que este equipo o dispositivo no cause interferencia perjudicial y
(2) este equipo o dispositivo debe aceptar cualquier interferencia, incluyendo la que pueda causar su operación no deseada.

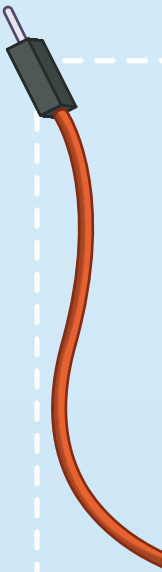


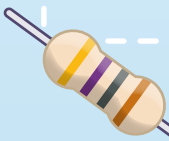


LA GUÍA OFICIAL DE
Raspberry Pi
para principiantes



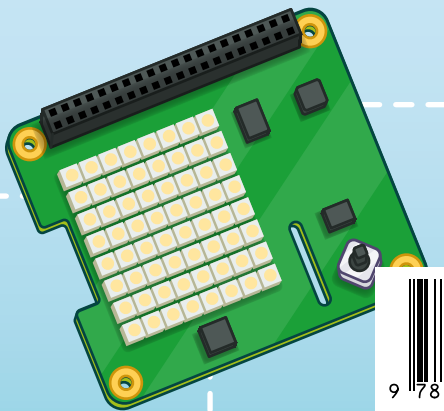
Raspberry Pi es un ordenador pequeño e inteligente de fabricación británica con un gran potencial. Hecho con la misma tecnología que la de un smartphone, Raspberry Pi se ha diseñado para ayudarte a aprender a codificar, descubrir cómo funcionan los ordenadores y construir cosas increíbles. Esta guía tiene el propósito de demostrar lo fácil que es empezar.

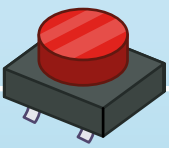




Aprenderás a:

- Configurar tu Raspberry Pi, instalar el sistema operativo y usar este ordenador totalmente funcional.
- Codificar proyectos sencillos paso a paso, usando los lenguajes de programación Scratch 3 y Python.
- Experimentar con la conexión de componentes electrónicos y divertirte creando proyectos estupendos.





raspberrypi.org

9 781912 047949

Raspberry Pi PRESS