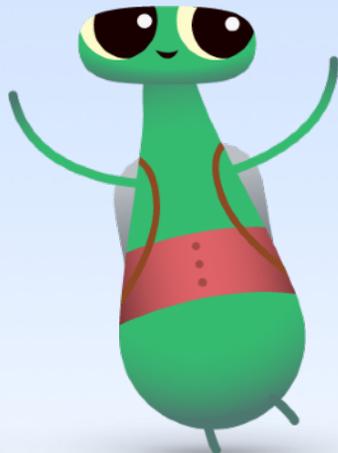




Swift Coding Club

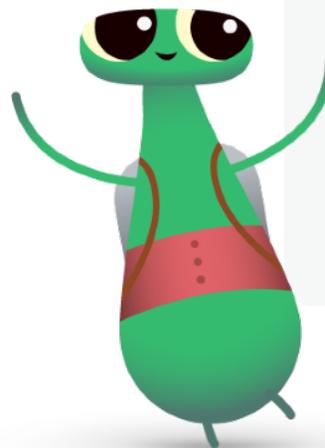


Swift Playgrounds

Actividades de programación

Temas

- | | | |
|---|---|----|
| 1 | Pensar como un ordenador:
Comandos y secuencias | 3 |
| 2 | Pensar de forma eficiente:
Funciones e introducción a los bucles | 5 |
| 3 | Pensar de forma lógica:
Código condicional | 7 |
| 4 | Pensar una y otra vez:
Bucles while | 9 |
| 5 | Pensar en la misma idea:
Algoritmos | 11 |
| 6 | Pensar como un agregador de
noticias (o NewsBot): Variables | 13 |
| 7 | Pensar como un arquitecto:
Tipos | 15 |



¡Te damos la bienvenida al Swift Coding Club!

Estas actividades de programación cubren conceptos de programación fundamentales relacionados con las actividades de diseño de apps. Con estas actividades, además de desarrollar tus habilidades de programación, también empezarás a comprender cómo funcionan las apps. Esto te ayudará a diseñar mejores apps.

Con cada tema, aprenderás sobre un concepto de programación determinado mediante una breve actividad introductoria, y después aplicarás ese concepto para resolver rompecabezas en Swift Playgrounds.

Además, todos los temas incluyen un apartado titulado «Ve más allá» en el que se proponen actividades que te ayudan a ampliar tus conocimientos sobre el concepto en cuestión. Estas actividades adicionales son opcionales y puedes descartarlas, hacer una o las dos. Ten en cuenta que algunas de ellas requieren otro dispositivo, como un robot o un dron. Si dispones de estos dispositivos, te vendrán muy bien para aplicar lo que has aprendido.

¡Que te diviertas!

Pensar como un ordenador: Comandos y secuencias

1



Introducción (15 minutos)

Forma una pareja con un compañero. Un dirigirá y el otro ejecutará. La persona que dirige tiene que pensar qué tiene que hacer la persona que ejecuta. Por ejemplo, dibujar una carita sonriente en la pizarra o dar cinco saltos de tijera. Sin indicar a la persona que ejecuta cuál es la tarea, quien dirige debe proporcionar únicamente indicaciones paso a paso. ¿Ha acertado ha completar la tarea como debía la persona que ejecuta?

La persona que dirigía daba a la persona que ejecutaba una secuencia de comandos, que es lo que tienes que hacer al escribir código.

 Mira este [vídeo](#) sobre los comandos, y [este otro](#) sobre la depuración.

Ahora vuelve a pensar las instrucciones, sobre todo si quien tenía que ejecutar la tarea no lo hizo bien. ¿Faltaba algún paso en las instrucciones? ¿Habría estado más claro si hubieses cambiado el orden de algunos pasos? Este proceso recibe el nombre de «depuración». Cuando programamos, solemos depurar para corregir y mejorar el código que escribimos.

Práctica (30 minutos)

Usa el módulo «Aprende a programar 1» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Qué relación guardan los comandos que has usado en la app con las indicaciones que ha dado la persona que dirigía?

Comando: acción específica que quieres que haga el ordenador.

Secuencia: orden en el que se dan los comandos.

Depuración: proceso de encontrar y corregir errores.

Comandos

Introducción	✓
Dar órdenes con comandos	✓
Añadir un comando nuevo	✓
Activar los interruptores	✓
Práctica de portales	✓
Encontrar y corregir errores	✓
Práctica de depuración de errores	✓
La ruta más corta	✓



Pensar de forma eficiente: Funciones e introducción a los bucles



2

Introducción (5 minutos)

Piensa en unos pasos de baile. Describíos los pasos mutuamente o, mejor aún, pedid a otros que los ejecuten. ¿Ha sido fácil describir los pasos?

En programación, muchas veces resulta más fácil combinar comandos que ya existen que crear un comportamiento nuevo de cero. Este proceso recibe el nombre de «composición». Cuando pones un nombre al comportamiento nuevo para poder utilizarlo más adelante, creas una función. Cuando indicas a un programa que ejecute una función, la estás «llamando». De modo que si alguien dice: «Baila la Macarena», está llamando a la función «Macarena».

 Mira este [vídeo](#) sobre funciones y bucles.

Práctica (40 minutos)

Usa el módulo «Aprende a programar 1» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: En uno de los rompecabezas: ¿cuántos movimientos ha hecho el personaje? ¿Y cuántos comandos has escrito? ¿Cuándo y por qué se deben crear funciones y bucles?

Función: colección de comandos agrupados y con un nombre.

Bucle for: ejecuta un bloque de código un número de veces determinado.



Funciones

Introducción	✓
Componer un comportamiento	✓
Crear una función nueva	✓
Recoger, activar y repetir	
Al otro lado del tablero	
Patrones anidados	✓
Escaleras intercaladas	✓
En busca del tesoro	

Bucles for

Introducción	✓
Utilizar bucles	✓
Bucles por los cuatro costados	✓
Ida y vuelta	
Saltando con ciclos	
Ramificaciones	
Granja de gemas	
Batida a cuatro	

Pensar de forma eficiente: Funciones e introducción a los bucles

Ve más allá

2

Creadores de patrones (1 sesión)

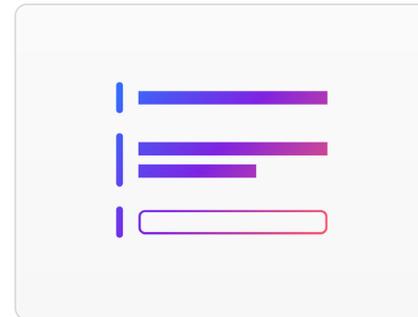
1. Crea un patrón usando una app de dibujo como Art Set o Pages y añade diferentes figuras, objetos y colores. El patrón puede ser todo lo largo que quieras.
2. En la app, escribe el patrón 20 veces con palabras. Por ejemplo: «rojo, amarillo, azul; rojo, amarillo, azul...».
3. Crea un nombre para la parte del patrón que se repite (por ejemplo, rojo, amarillo y azul = colores primarios) y, después, vuelve a escribir el patrón con palabras utilizando su nombre.
4. ¿Te ha resultado más fácil o has podido escribir el patrón en menos pasos?
5. ¿Cuántas veces se repite el patrón «colores primarios»? Ahora describe tu patrón en un paso. ¡Has escrito un bucle for!

Reflexiona: ¿Qué relación guarda esta actividad con la programación?



Entrevista a un robot (1 sesión)

1. Bájate el punto de inicio Respuestas en Swift Playgrounds.



2. En la página Texto, «show» y «ask» son funciones. Toca Ejecutar Código y rellena tu nombre. A continuación, toca Enviar para ver qué ocurre. Las funciones pueden tener un resultado, que es lo que ves en la vista en directo.
3. En la página Tipos, explora distintas funciones «show» y «ask».
4. Por parejas, escribid una serie de funciones «show» y «ask» para que las complete el otro.
5. Usad los resultados de vuestras funciones para escribir una historia de ficción, una entrevista o una pequeña biografía.

Reflexiona: ¿Y si hubieses escrito las funciones «show» y «ask» en una secuencia diferente? ¿Cómo afectaría eso a la historia o entrevista?

Pensar de forma lógica: Código condicional

3



Introducción (10 minutos)

En grupo, jugad un par de rondas de un «Veo, veo» adaptado. En esta adaptación del «Veo, veo», un alumno escoge un objeto y, en lugar de decir la primera letra, describe alguno de sus atributos. El resto de los jugadores tienen que mirar a su alrededor y adivinar qué han visto. El alumno que lo adivine elige el siguiente objeto.

¿Qué decisiones has tenido que tomar? ¿Qué razonamiento has seguido para adivinar el objeto? Para jugar al «Veo, veo», habéis tenido que pensar en términos de condiciones.

 Mira este [vídeo](#) para aprender sobre el código condicional.

Práctica (35 minutos)

Usa el módulo «Aprende a programar 1» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Qué tipos de decisiones ha tomado tu código usando la sentencia if? ¿Cómo has combinado los bucles for y las sentencias if? ¿Por qué?

Condición: cualquier cosa que, tras probarla, da como resultado «true» (verdadero) o «false» (falso).

Código condicional: un bloque de código que se ejecutará si algo es «true» (verdadero).

Booleano: un valor que solo puede ser verdadero (true) o falso (false).

Operador lógico: un símbolo o palabras como «y» (and), «o» (or) y «no» (not).



Código condicional

Introducción	✓
Comprobación de interruptores	✓
Utilizar else if	✓
Código condicional en bucle	✓
Escalada condicional	
Definir funciones más eficaces	✓
Rodado	
Árbol de decisiones	

Operadores lógicos

Introducción	✓
Utilizar el operador NO	✓
Espiral de NO	
Comprueba esto Y aquello	✓
Comprueba esto O aquello	✓
Laberinto lógico	

Pensar de forma lógica: Código condicional

Ve más allá

3

En busca del tesoro (1 sesión)

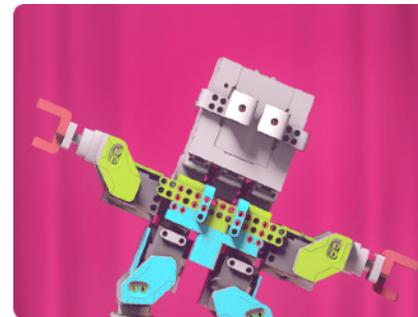
1. Cada estudiante debe escribir un par de condiciones en dos trozos de papel, por ejemplo: «Tiene forma de rectángulo» o «Empieza por C». Después, introduce los trozos de papel en un sombrero.
2. En pequeños grupos, sacad dos preguntas del sombrero y haced entre tres y cinco fotos de objetos de la sala que cumplan esa condición.
3. Crea un álbum de fotos usando una app de presentaciones como Keynote. Crea un apartado para cada condición. No etiquetes las condiciones todavía.
4. Presenta tu álbum de fotos a otro grupo para ver si adivinan las condiciones. Si aciertan, añade a la página del álbum la sentencia condicional (por ejemplo, «Si es azul, haz una foto»).

Reflexiona: ¿Ha habido algún caso en el que resultara difícil valorar si una foto respondía a las condiciones o no? ¿Cómo manejaría un ordenador estos casos?



MeeBot Dances (1 sesión)

1. Si tienes un robot MeeBot de UBTECH, descarga la lección MeeBot Dances en Swift Playgrounds.



2. Recorre la lección mientras ves cómo puedes aplicar funciones, bucles for y código condicional para hacer que MeeBot baile.
3. Crea tu propia coreografía. Incluso puedes elegir tu música en la última página.

Reflexiona: ¿Cómo has usado el código para reflejar el tempo de la música?

Pensar una y otra vez: Bucles while

4



Introducción (5 minutos)

En el tema 2, has aprendido qué son las funciones y los bucles for. ¿Qué habéis bailado? ¿Cómo utilizarías los bucles for para escribir la función del baile?

Ahora imagínate que quieres hacer ese baile en la fiesta de fin de curso del cole. ¿Cómo podrías saber cuándo termina la canción para dejar de bailar?

Usarías un código condicional: si la canción suena, sigue bailando. O aún más claro, un bucle while: mientras suene la canción, baila.

Esto es distinto a un bucle for, que indica a un ordenador que ejecute código un número de veces determinado, por ejemplo, bailar en círculo 10 veces. Un bucle while indica al ordenador que ejecute un bloque de código hasta que algo ocurra. Por ejemplo, bailar en círculo hasta que la música deje de sonar.

 Mira este [vídeo](#) para aprender sobre los bucles while.

Práctica (40 minutos)

Usa el módulo «Aprende a programar 1» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Cuándo has usado bucles for y while? ¿Cómo lo has decidido?

Bucle while: bucle que ejecutará un bloque de código mientras una condición dada sea «true» (verdadera). Cuando la condición sea «false» (falsa), el bucle dejará de ejecutarse.

Bucles while

Introducción	✓
Ejecutar código mientras...	✓
Crea bucles while más inteligentes	✓
Elige la herramienta adecuada	✓
Cuatro por cuatro	
¡Media vuelta!	
Tierra de recompensas	
Anidar bucles	✓
Rectángulos aleatorios	
Siempre llevas razón	



Pensar una y otra vez: Bucles while

Ve más allá

4

El escondite, segunda parte (1 sesión)

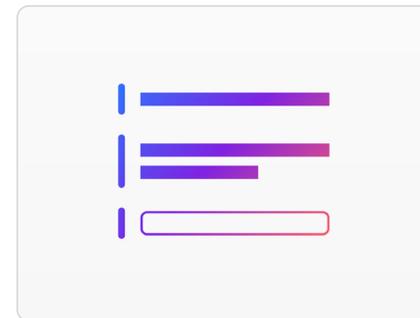
1. Esconde un objeto pequeño en la sala o cerca de ella.
2. Desde un lugar concreto, grábate con la cámara del iPad dando instrucciones a alguien para que encuentre el objeto. Las indicaciones deben comenzar desde tu posición. Utiliza funciones, bucles for y bucles while siempre que puedas.
3. Ahora intercambia vídeos con otro estudiante. Ve su vídeo e intenta encontrar el objeto escondido. ¿Lo has encontrado?
4. Repasad juntos los dos vídeos. Escribe los casos en los que usaste bucles for y while en las indicaciones.

Reflexiona: ¿Te ha facilitado las cosas utilizar bucles en esta ocasión? ¿Ha habido algún momento en el que fuera más difícil?



21 preguntas (1 sesión)

1. Volvamos al punto de inicio «Respuestas».
[Descarga esta versión.](#)



2. La plantilla Respuestas está configurada para que juegues a «21 preguntas». Para empezar, examina el código. ¿Qué hará?
3. El creador elige un objeto o cosa y lo introduce como respuesta en el código. El jugador puede hacer al creador 21 preguntas de sí o no.
4. Después de cada pregunta, el jugador puede introducir su respuesta en la vista en directo para ver si ha acertado. Asegúrate de que el creador entrega el área de juegos con la vista en directo a pantalla completa para que el jugador no pueda ver el código y la respuesta correcta.

Reflexiona: ¿Qué conceptos de programación se han utilizado en esta área de juegos? ¿Cómo se usan?

Pensar en la misma idea: Algoritmos



5

Introducción (5 minutos)

En grupo, escribid una lista de cosas que hacéis habitualmente que requieran varios pasos, como cepillaros los dientes o preparar un bocadillo. Todo esto son algoritmos.

Elige un ejemplo y pide a varios alumnos que den indicaciones de cómo lo hacen. ¿Han dado las mismas indicaciones? ¿En qué se diferencian? ¿Han llegado al mismo resultado al final?

 Ve este [vídeo](#) para aprender sobre los algoritmos.

Práctica (40 minutos)

Usa el módulo «Aprende a programar 1» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Cuántas formas diferentes de resolver cada rompecabezas crees que hay? ¿De quién era el algoritmo más corto? ¿De quién era el más interesante?

Algoritmo: conjunto de reglas o instrucciones paso a paso.

Pseudocódigo: descripción informal de un fragmento de código o de un concepto destinado a la lectura humana.



Algoritmos

Introducción	✓
La regla de la mano derecha	✓
Ajusta tu algoritmo	✓
Conquista un laberinto	✓
¿Hacia dónde girar?	
A derecha e izquierda	

Pensar en la misma idea: Algoritmos

Ve más allá

5

¿Quién es la persona más alta de la clase? (1 sesión)

1. Dividíos en grupos reducidos. Cada grupo tiene que pensar en una forma, o algoritmo, de determinar quién es la persona más alta. ¡No cuenta si ya lo sabéis o si es evidente solo con mirar!
2. Usa tus conocimientos de programación para hallar los pasos del algoritmo. Puedes inventar tu propio pseudocódigo para escribir tu algoritmo con tanta terminología de programación como sea posible.
3. Cada grupo debe presentar su algoritmo. El grupo entero debe ejecutarlo.

Reflexiona: ¿Qué algoritmo parece más eficiente? Si quisieras encontrar al alumno de menor estatura, ¿qué cambiarías en el algoritmo?



Parrot (1 sesión)

1. Si dispones de un dron de Parrot, descarga la lección Parrot en Swift Playgrounds.



2. Usa todas las habilidades de programación que has aprendido hasta ahora para hacer que el dron despegue, aterrice, se mueva en todas direcciones y realice acrobacias.
3. Por último, crea un algoritmo que defina una ruta de vuelo para ir del punto A al punto B.

Reflexiona: ¿Cómo ha afectado la velocidad del dron a tus rutas de vuelo?

Pensar como un agregador de noticias (o NewsBot): Variables



6

Introducción (5 minutos)

Imagínate que te estás mudando a una casa nueva. Probablemente pondrías tus pertenencias en cajas y las etiquetarías con una o dos palabras para describir su contenido. Por ejemplo, si una caja contiene trofeos, le pondrías la etiqueta «Trofeos». Este ejemplo sugiere tres características importantes de un contenedor.

Cuando programamos un ordenador, en lugar de cajas usamos variables. Las variables son parecidas a las cajas: ambas tienen una etiqueta (un nombre) y un contenido (un valor). El valor o contenido puede cambiar, pero la etiqueta o nombre no. Para hallar el valor o contenido de una variable, tenemos que encontrar la variable con su nombre o etiqueta.

 Mira este [vídeo](#) para obtener más información.

Práctica (40 minutos)

Ahora usa «Aprende a programar 2» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Cómo te ha ayudado el uso de variables con la app?

Variable: contenedor con un nombre dado que almacena un valor. El valor puede cambiar a lo largo del tiempo.



Variable

Introducción	✓
Llevar el recuento	✓
Acumular el valor	
Incrementar el valor	✓
En busca de las siete gemas	✓
Tres gemas, cuatro interruptores	
Encontrar valores iguales	✓
Contar interruptores	
Recoger el total	

Pensar como un agregador de noticias (o NewsBot): Variables

Ve más allá

6

Newsbot (1 sesión)

1. En esta actividad vas a crear a NewsBot, un robot capaz de escribir automáticamente un pequeño artículo. En primer lugar, piensa en un evento deportivo o de actualidad y en qué datos puedes necesitar para escribir sobre él.
2. Haz una lista de entre cuatro y seis variables, como el nombre de un equipo, el marcador y la fecha del evento. Escribe un artículo de dos o tres frases usando variables, que NewsBot rellenará para otros eventos similares.
3. Formad parejas. Uno de vosotros proporciona información para cada variable del artículo. Después intercambiad los papeles. ¿Tenéis dos noticias completas con sentido?

Reflexiona: ¿Hay algún nombre de variable que haya funcionado mejor que otro? ¿Por qué sí o por qué no?

VARIABLES:

- teamOneName
- teamTwoName
- teamOneFinalScore
- teamTwoFinalScore
- ballfieldName

STORY:

The teamOneName and the teamTwoName faced off at ballfieldName. The final score was teamOneName teamOneFinalScore to teamTwoName teamTwoFinalScore.

Sphero Arcade (1 sesión)

1. Si dispones de un robot Sphero SPRK+, descarga la lección Sphero Arcade en Swift Playgrounds.



2. Aplica funciones y variables para apuntar, detectar colisiones y, en última instancia, crear tu propia versión del Pong.
3. En la página «Jugar Sphero Pong», analiza el código antes de ejecutarlo o editarlo. ¿Qué hace cada comando?
4. ¿Qué elementos del juego has cambiado usando variables?

Reflexiona: ¿Qué otros juegos podrías personalizar como has hecho con el Pong?

Pensar como un arquitecto: Tipos



7

Introducción (5 minutos)

¿Cuántos tipos de edificios se te ocurren? Elige uno. ¿Qué hace que sea único? En otras palabras, ¿qué características concretas tiene ese tipo de edificio en particular? ¿Qué suele ocurrir en él? A esto lo llamaremos «comportamientos». Por ejemplo, un colegio tiene aulas (características) y el timbre suena entre clase y clase (comportamiento). ¿Todo el mundo está de acuerdo con las propiedades y los comportamientos? ¿Por qué sí o por qué no?

Si utilizamos un programa informático para que nos ayude a construir el edificio, debemos dar indicaciones muy precisas. Tenemos que definir su tipo definiendo las propiedades (a las que llamamos *características*) y los métodos (*comportamientos*).

 Mira este [vídeo](#) para obtener más información.

Práctica (40 minutos)

Ahora usa «Aprende a programar 2» de Swift Playgrounds para completar los rompecabezas con marcas verdes de la lista de la derecha.

Reflexiona: ¿Qué tipos había en la app? ¿Qué has inicializado?

Tipo: agrupación de propiedades (*características*) y métodos (*comportamientos*) de una clase de datos que recibe un cierto nombre.

Inicialización: acción de crear una nueva instancia de un tipo, lo que incluye la definición de los valores iniciales de cualquiera de las propiedades del tipo.



Tipos

Introducción	✓
Desactivar un portal	✓
Activar y desactivar portales	
Elegir el portal correcto	✓
Las esquinas del mundo	
Gemas por doquier	

Inicialización

Introducción	✓
Inicializar al experto	✓
Entrenar al experto	
Usar instancias de distintos tipos	✓
Se necesitan dos	

Pensar como un arquitecto: Tipos

Ve más allá

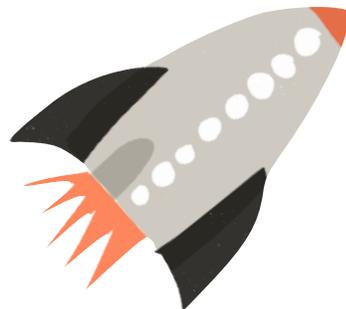
7

Pensar como un arquitecto (1 sesión)

1. Elige un tipo de edificio, real o imaginario.
2. Piensa en cinco o seis variables para describir cómo es tu tipo de edificio. Estas variables no deben contener valores, sino que deben describirlos (por ejemplo, «altura» o «númeroDeVentanas»).
3. Añade valores junto a las variables. Esto describe una instancia específica del tipo de edificio. En la terminología de Swift, estás inicializando una instancia del tipo de edificio.
4. Termina la inicialización usando una app de dibujo como Notas para hacer un boceto del tipo de edificio que refleje las variables y los valores.
5. Busca un compañero e intercambia las listas de variables y valores de vuestros tipos de edificio. Dibujad el tipo de edificio del otro y enseñaos los dibujos que habéis hecho. ¿Se parecen mucho?

Reflexiona: ¿Se te ocurre alguna forma de que los dibujos se parezcan más?

Tipo: casaCohete
númeroDeMotores= 4
altura = 15 metros
númeroDeVentanas = 8
color = gris espacial
formaDeLaPuerta = rectángulo
redondeado



Piedra, papel o tijera (1 sesión)

1. Descarga el reto «Piedra, papel o tijera» en Swift Playgrounds.



2. Mira la lección. ¿Cómo funciona el código? ¿Qué conceptos de programación reconoces?
3. Ahora intenta crear una versión del juego en la que puedas usar tus propias reglas y añadir nuevas acciones. Para ello, tendrás que definir el tipo de juego e inicializarlo junto con las propiedades y métodos correspondientes.

Reflexiona: ¿Cuáles eran las propiedades y los métodos del juego original? ¿Qué propiedades y métodos nuevos has añadido para personalizar el juego?

