# Programación Basada en Componentes

Mientras que la clásica Programación Orientada a Objetos (POO) puede ser, y es usada, el flujo de trabajo de Unity se construye alrededor de la estructura de componentes—lo cual requiere pensamiento basado en componentes.

## ¿Qué es un componente?

En el mundo de la programación, el concepto de “componentes” y “desacoplamiento” van de la mano. Se puede decir que un componente es una pieza pequeña de una maquina más grande. Cada componente tiene su propio trabajo específico, y pueden (óptimamente) cumplir su tarea o propósito sin la ayuda de fuentes externas. Adicionalmente, los componentes rara vez pertenecen a una sola máquina, y pueden ser unidos con varios sistemas para cumplir con su tarea, pero generar otro resultado. Esto se debe a que los componentes no solo no les importa saber sobre el sistema al que pertenecen, sino que tampoco saben que existe.

Considere un mando de Xbox 360. Tiene 2 sticks análogos, varios botones, gatillos, etc. No solo el mando entero es un componente, sino que también, cada aspecto individual del control es un componente.

El botón X puede: ser presionado; informar que fue presionado; ser liberado; informar que fue liberado. No tiene idea de que hay varios otros botones justo al lado de él, y no le importa.

El mando en sí es un componente, compuesto de otros componentes (todos los botones, joysticks y gatillos), porque puede mandar datos sin importar donde esté conectado, y que tipo de objeto es (Xbox, PC, alguna creación de Arduino u otra cosa). El botón X y el mando en sí, no necesitan saber que juego está jugando, y hará su trabajo sin importarle el receptor de la información que mandan.

La función del mando es unidireccional, y su tarea nunca va a cambiar de receptor a receptor. Esto es lo que lo convierte en un componente exitoso; puede hacer su trabajo él solo, y puede hacer su trabajo conectado a múltiples dispositivos.

## ¿Cómo y por qué Unity favorece el uso de componentes?

Unity se construyó teniendo a los componentes en mente, y lo demuestra. Uno de los aspectos más valorados y distintivos de Unity, es que es un programa muy visual. Unity te deja ver todo en lo que estás trabajando en tiempo real. Esto significa que puedes probar tu proyecto, ver tu proyecto corriendo en una ventana separada, hacer cambios al código y ver esos cambios reflejados en tiempo real. La cantidad de poder que este sistema le da al desarrollador es inmensa. Todo esto fue posible gracias a la arquitectura basada en componentes de Unity.

## Pensando en componentes

La parte más difícil de trabajar con componentes, es aprender como estructurar tus proyectos al usarlo. Para la mayoría de los programadores, esto probablemente significa crear un montón de scripts que harán tareas más pequeñas y específicas.

La comunicación entre scripts también es un obstáculo decente, ya que tendrás más piezas y menos clases gigantes en las que cada objeto conoce a todos los demás. Obviamente, hay maneras de superar este obstáculo, como variables estáticas (para partes clave del juego como jugadores, puntuación, etc.), pero eso raramente funciona para cualquier cosa. También hay métodos avanzados para estructurar correctamente los componentes y mantenerlos desacoplados.

Por suerte, como Unity fue construido pensando en componentes, tiene una variedad de funciones integradas que nos ayudan a lograr esto. Hay funciones para obtener referencias a un componente especifico, para revisar todos los objetos y ver si contienen un componente especifico, etc. Con estas funciones uno puede fácilmente conseguir la información necesaria para crear una comunicación correcta entre componentes.

## Conclusión

Pensar en componentes puede ser difícil, pero definitivamente tiene sus beneficios. Al usar Unity, uno puede elegir el método que prefiera, pero definitivamente, pensar en componentes es la mejor opción.