Paradigma de programación por eventos

Darien Parra, Cristian Bello, Samuel Chaves, Santiago Ruiz

Paradigmas de programación

Universidad Sergio Arboleda

Joaquin F. Sanchez

Imagina que estás construyendo una casa de muñecas. En lugar de seguir un plano rígido donde cada habitación se construye en un orden específico, decides construirla de forma más interactiva. A medida que vas armando la casa, vas agregando muebles y decorando cada habitación. Cada vez que agregas algo o cambias algo, ocurre un "evento" (por ejemplo, colocar una silla, abrir una puerta).

La programación por eventos funciona de manera similar. En vez de ejecutar una serie de instrucciones de forma lineal, el programa espera a que ocurran ciertos **"eventos"**. Estos eventos pueden ser acciones del usuario (como hacer clic en un botón), cambios en el sistema (como recibir un mensaje de red) o incluso el paso del tiempo.

Cuando ocurre un evento, el programa ejecuta una función específica llamada **"manejador de eventos"**. Este manejador de eventos contiene el código que se encargará de responder a ese evento en particular. Por ejemplo, si el evento es un clic en un botón, el manejador de eventos podría mostrar una ventana emergente o realizar un cálculo.

Ventajas de la programación por eventos:

Mayor interactividad: Las aplicaciones se sienten más vivas y responsivas, ya que reaccionan de inmediato a las acciones del usuario.
Flexibilidad: Puedes agregar nuevas funcionalidades fácilmente creando nuevos manejadores de eventos.
Reutilización de código: Los manejadores de eventos pueden ser reutilizados en diferentes partes de la aplicación.
Modelo de programación más natural: Se asemeja a cómo interactuamos con el mundo real, donde las cosas suceden en respuesta a eventos.

Desventajas:

Complejidad: Puede ser difícil de entender al principio, especialmente cuando hay muchos eventos ocurriendo simultáneamente.
Dependencia de un bucle de eventos: La mayoría de los lenguajes y frameworks necesitan un bucle que esté constantemente monitoreando los eventos.
Depuración: Los errores pueden ser más difíciles de encontrar, ya que pueden surgir en cualquier momento debido a un evento inesperado.

Ejemplos de uso:

Interfaces gráficas: Casi todas las aplicaciones con una interfaz gráfica utilizan la programación por eventos para responder a los clics del mouse, las pulsaciones de teclado, etc.
Juegos: Los juegos están llenos de eventos, como colisiones, movimientos del jugador, etc.
Aplicaciones de red: Los servidores web y las aplicaciones de chat utilizan eventos para manejar conexiones y datos entrantes.

Ejemplo práctico: Servidor web

Cuando un usuario solicita una página web, se genera un evento de conexión. El servidor escucha este evento y envía la página solicitada. Mientras tanto, el servidor puede seguir atendiendo otras solicitudes.

Tecnologías y herramientas

Sockets: Son la interfaz de programación más común para crear aplicaciones de red basadas en eventos.
Bibliotecas de redes: Existen numerosas bibliotecas que simplifican la programación de redes, como Boost.Asio (C++), Twisted (Python), y Node.js (JavaScript).
Lenguajes con soporte nativo para eventos: Algunos lenguajes, como JavaScript, tienen soporte nativo para programación asíncrona y basada en eventos.