Escuela de Sistemas y Tecnologías

Transparencias de **ANALISTA DE SISTEMAS**

Edición 2017 – Materia: **Diseño de Aplicaciones Web**

TEMA: Eventos y Delegados

Plantel y Contactos

- **Bedelía:**
  - Mail: bedeliasistemas@bios.edu.uy

- **Encargado de Sucursal:**
  - Pablo Castaño
  - Mail: pablocasta@bios.edu.uy

Recursos

- **Recursos Imprescindibles:**
  - Sitio Web de material
    (comunicarse con Bedelía por usuario/contraseña).
  - Transparencias del Curso.
  - Contar con el software necesario

Agenda

- Eventos
- Delegados – Controladores de Eventos
- Eventos en Net Framework

Eventos

Definición (1)

- Un evento es por definición un mensaje que envía un objeto cuando ocurre una acción. La acción puede estar causada por la intervención de un usuario, como un clic, mouseover, etc. o por la lógica del programa que llame a la implementación de un evento.
- El objeto que provoca el evento se definirá como **remitente** del evento. El objeto que captura el evento y responde a él se denominará **receptor** del evento.

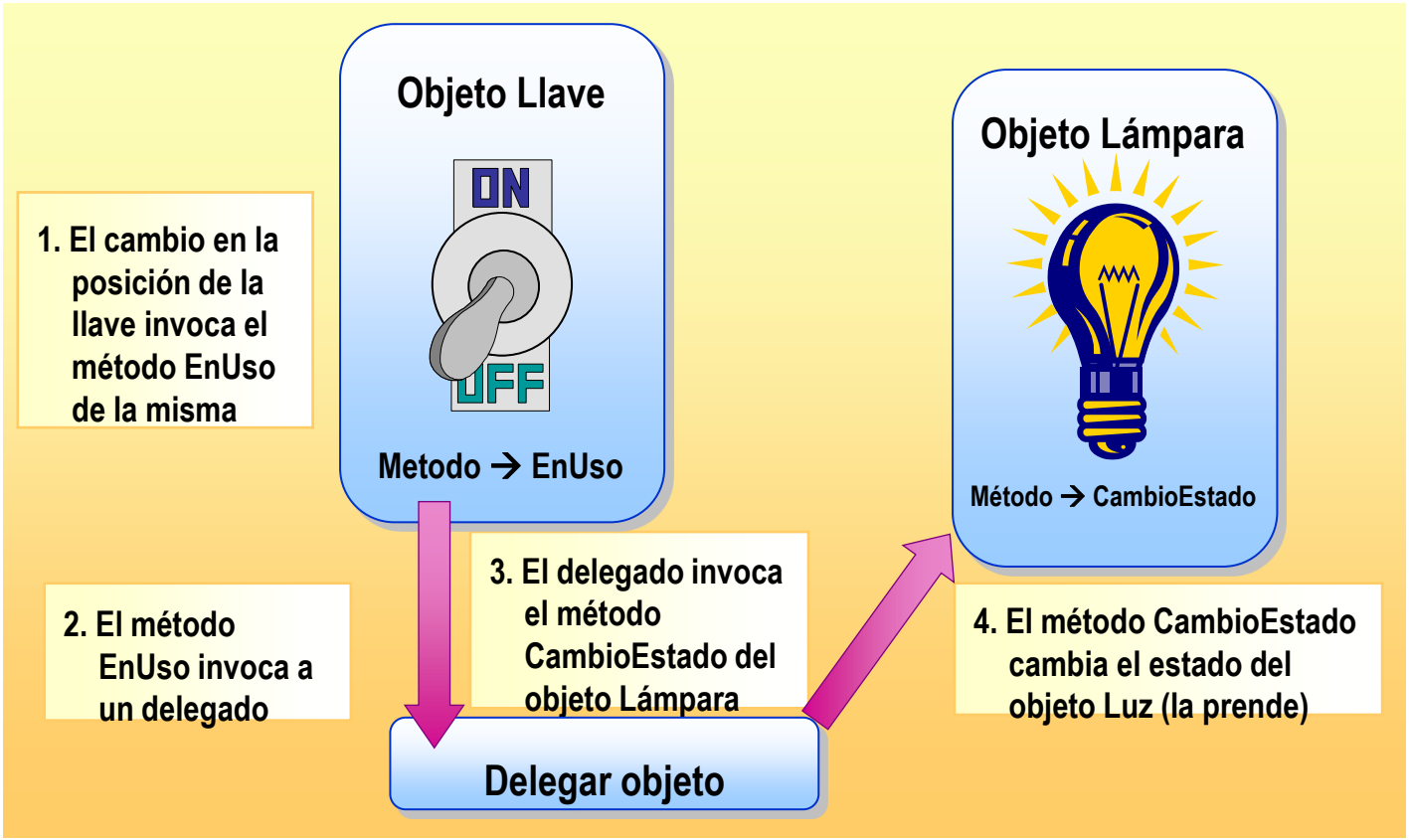
Definición (2)

- En las comunicaciones de eventos, el remitente del evento no sabe qué objeto o método recibirá los eventos que provoca.
- Por lo anterior se necesita un intermediario entre el origen y el receptor. NET Framework define un tipo especial (Delegado) que proporciona la funcionalidad de un puntero a función.
- Por lo tanto, un delegado es un puntero (referencia) a una función (ya sea estática o de instancia).

Delegados
Controladores de Evento

Definición

- Un delegado tiene una firma y mantiene una referencia a un método que coincida con ella.



EventHander (1)

- Representa un método controlador para un evento. No genera datos (notar el retorno tipo **void**)
- **EventHandler** se ubica lógicamente dentro del espacio de nombres **System** y físicamente dentro de la librería `mscorlib.dll`
- Su estructura es:

```
public delegate void EventHandler(
                        Object sender, EventArgs e)
```

EventHander (2)

- Parámetros de un controlador de eventos:
  - **Object sender:** Origen del evento, hace referencia a la instancia (objeto) que provocó el evento.
  - **EventArgs e:** Contiene datos sobre el evento. Por defecto este parámetro simplemente es una instancia de **EventArgs**.

EventHander (3)

- El modelo de eventos del .NET Framework se basa en la existencia de un delegado que conecta un evento a su controlador.
- Para provocar un evento, se requieren dos elementos esenciales:
  - Delegado que identifica el método que proporciona la respuesta al evento.
  - Clase que contiene los datos del evento.

EventHander (4)

- Sin embargo, hay tres elementos interrelacionados que proporcionan la funcionalidad de un evento:
  - Una clase que guarde los datos del evento, denominada ***NombreEvento*EventArgs**. Esta clase debe derivar de **System.EventArgs**;
  - Un delegado para el evento, denominado ***NombreEvento*EventHandler**
  - Una clase que provoca el evento. Esta clase debe proporcionar la declaración de evento y un método que provoca el evento.

Eventos en Net Framework

Modelo Web (1)

- Los eventos asociados a los controles estándar se originan en el cliente (Navegador) pero los controla el código trasero de la página ASP.NET, en el servidor Web.
- Para los eventos producidos en el cliente, el modelo de control de eventos ASP.NET necesita que la información del evento se capture en el cliente, para luego ser trasmitido un mensaje al servidor mediante un envío HTTP.
- ASP.NET controla la tarea de capturar, transmitir e interpretar el evento

Modelo Web (2)

- Al crear controladores de eventos en una página Web ASP.NET, no es necesario saber capturar la información del evento, ni hacer que éste esté disponible para el código.
- Debido a que los eventos requieren un viaje de ida y vuelta al servidor para procesarse, pueden afectar al rendimiento de una página. Por ello los controles de servidor ofrecen un conjunto limitado de eventos: eventos de click del mouse, y eventos de cambio (Changed)

Modelo Web (3)

- En los controles de servidor estándar, los eventos **click** hacen que la página se envíe de vuelta inmediatamente al servidor. Se los conoce como eventos PostBack.
- Sin embargo los eventos **Changed** no ocasionan esto. En su lugar, se generan la próxima vez que tenga lugar una acción de envío. A estos eventos se los denomina Non_PostBack.

Modelo Web (4)

- Los eventos que tienen lugar con frecuencia (y que pueden provocarse sin que el usuario lo sepa como **MouseOver**) no se pueden implementar.
- La propia página ASPX también provoca eventos de ciclo de vida en cada paso del procesamiento.
- A continuación un cuadro especificando los eventos mas importantes del ciclo de vida compartidos por los controles y las paginas web:

Ciclo de Vida Web

Nombre	Descripción
`Init`	Se produce al inicializar el control de servidor, que es el primer paso en su ciclo de vida.
`Load`	Se produce cuando el control de servidor se carga en el objeto Page.
`DataBinding`	Se produce cuando el control se enlaza a un origen de datos.
`PreRender`	Se produce una vez que se carga el objeto Control, pero antes de su depuración.
`UnLoad`	Se produce cuando el control de servidor se descarga de la memoria.
`Disposed`	Se produce cuando un control de usuario se libera de la memoria.