4. Implementación

Ph.D Priscilla Jiménez P.

CURSO: ITIR554 – Programación basada en eventos
2020 B – Semana 10

Inferfaz Gráfica

Manejo de eventos en WPF

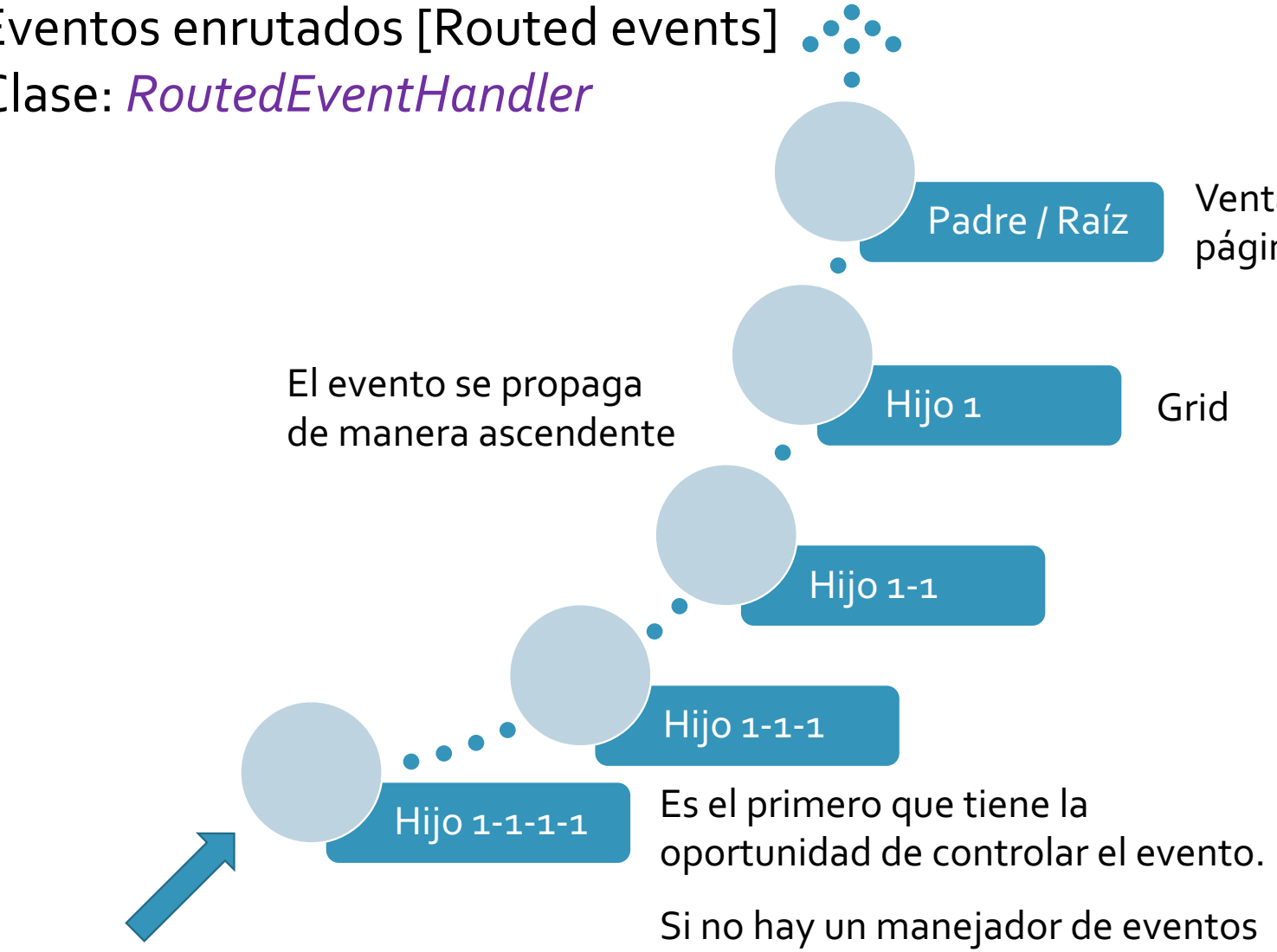
Comandos

Manejo de Eventos en WPF

Los elementos de una aplicación se relacionan entre si a través de un árbol de elementos.

Eventos enrutados [Routed events]

Clase: *RoutedEventHandler*



Eventos Enrutados

Al propagarse hasta la raíz del árbol el árbol de elementos pueden asociar un controlador una sola vez (en el elemento padre o contenedor)

Puede ser respondido por cualquier elemento que este en la ruta en lugar de simplemente por el elemento que lo desencadenó.

No todos los eventos enrutados siguen la misma ruta. Tipos de enrutamiento:

De propagación (Bubbling events)	Directo	De túnel (Tunneling events)

Eventos enrutados

Se propaga	Si va subiendo por el árbol desde el elemento que lo desencadenó.
Se tuneliza	Si se inicia en la raiz y viaja a un elemento hijo del árbol de elementos. Teniendo como meta el evento que lo desencadenó.
Directo	Sólo puede ser respondido por el elemento que lo desencadenó.

Eventos de entrada

Vienen implementados como una pareja de túnel-propagación.

PreviewKeyDown (Evento túnel o vista previa)	KeyDown (Evento de propagación)

Una única acción desencadena los dos eventos.

Primero se desencadena el evento túnel y luego el evento de propagación. Hasta que un elemento marque el evento enrutado como manejado (Handled propiedad de RoutedEventArgs).

Eventos Adjuntos

- Para permitir que los elementos controlen eventos que se declaran en un elemento diferente WPF admite *eventos adjuntos.*
- Por ejemplo, se define el evento en Grid (elemento padre) en lugar del button.
- Para dejar de propagar un evento al padre se debe escribir e.Handled = true; en el manejador del evento hijo.

Commands Ordenes Enrutadas

Acción que la aplicación lleva a cabo a petición del usuario.

Se las define en un lugar y se hace referencia a ellas por medio de barra de menu, barra de herramientas, botones, etc.

Clase: *RoutedCommand*.

Comands: Print, Save, GoToPage, NextTrack, Delete, MoveDown, etc.

5 Categorías: ApplicationCommands, NavigationCommands, MediaCommands, EditingCommands, y ComponentCommands.

La habilitación o deshabilitación de un comando es centralizado.

Se usa para enlazar el comando al código.

- Eventos: PreviewExecuted y Executed
- Métodos: Execute() y CanExecute()

Un CommandBinding es definido en una ventada, elemento, entrada de teclado.

CommandBindings

Usar comandos en WPF

1. Crear un CommandBinding

XAML

```
<Window.CommandBindings>
  <CommandBinding Command="ApplicationCommands.Open"
                  Executed="OpenCmdExecuted"
                  CanExecute="OpenCmdCanExecute"/>
</Window.CommandBindings>
```

C#

```
CommandBinding OpenCmdBinding = new CommandBinding(
    ApplicationCommands.Open,
    OpenCmdExecuted,
    OpenCmdCanExecute);

this.CommandBindings.Add(OpenCmdBinding);
```

2. Se definen el código de los métodos
3. Se enlaza el CommandBinding a un objeto

Cuadros de diálogo

- Objetivos:
  - Mostrar información
  - Recopilar información de los usuarios
- Existen dos tipos:
  - Modales (Modal):
    - Cuando la aplicación necesita datos adicionales de un usuario para continuar.
    - Impide que el usuario active otras ventanas.
  - No Modales (Modeless)
    - No impide que el usuario active otras ventanas.

Ciclo de vida de una ventana

Activate	Activa la ventana situándola en primer plano; se produce el evento "Activated"
Deactivate	Desactiva la ventana; se produce el evento "Deactivated"
Close	Cierra la ventana; se produce el evento "Closing" y poseriormente el evento "Closed" eliminándose los recursos administrados por el objeto "Window"
Show	Muestra una ventana sin impedir que el usuario interactúe con las otras ventanas de la aplicación.
ShowDialog	Muestra una ventana impidiendo que el usuario interactúe con las otras ventanas de la aplicación.
Hide	Oculta una ventana, pero no la cierra; solo cambia el valor de la propiedad "Visibility"

Bibliografía y Recursos útiles

1. Colmenar, Santos, Antonio. *Visual C#: interfaces gráficas y aplicaciones para Internet con WPF, WCF y Silverlight*, RA-MA Editorial, 2014. ProQuest Ebook Central, https://bvirtual.epn.edu.ec:2117/lib/epnsp/detail.action?docID=5759069. (71-91)
2. Ferrer, Martínez, Juan. *Desarrollo de interfaces*, RA-MA Editorial, 2015. ProQuest Ebook Central, https://bvirtual.epn.edu.ec:2117/lib/epnsp/detail.action?docID=5758935. (Capítulo 4)

Bibliografía y Recursos útiles

- https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.applicationcommands?redirectedfrom=MSDN&view=netcore-3.1
- https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.navigationcommands?view=netframework-4.7.2
- https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.mediacommands?view=netframework-4.7.2
- https://docs.microsoft.com/en-us/dotnet/api/system.windows.documents.editingcommands?view=netframework-4.7.2
- https://docs.microsoft.com/en-us/dotnet/api/system.windows.input.componentcommands?view=netframework-4.7.2
- https://docs.microsoft.com/es-es/dotnet/framework/wpf/app-development/dialog-boxes-overview

Bibliografía y Recursos útiles

1. Videos:
   - Eventos WPF:
   - https://www.youtube.com/watch?v=_J9LcsdTwC8