Materia: Programación visual III.

Alumno: Juan Pablo Garcia Benitez.

Profesor: Nuño Ramírez Daniel A..

Resumen

27 de noviembre 2019

Delegados y Eventos

Es una lucha de diferencias ya que a la gente le cuesta diferencias uno del otro el cual la gente entra en debate, y C# no ayuda en mucho ya que los eventos son muy similares a los delegados.

Los delegados proporcionan otras características, pero la tiene la capacidad de realizar llamadas con una firma particular es la razón por cual existe del concepto de delegado. Los delegados tienen una referencia a un método y una referencia al objeto de destino al que se debe llamar el método.

Se debe de tener en cuenta que la palabra delegate no siempre significa que se está declarando un tipo de delegado. Se utiliza la misma palabra clave al crear instancias del tipo delegado utilizando métodos anónimos.

Cualquier tipo de delegado se crea tiene los miembros heredados el cual es de sus tipos de padres, un constructor con parámetros, los cuales va a ser constituido por de object e IntPtry tres métodos adicionales: Invoke, BeginInvokey EndInvoke.

Lo que tenemos que tener en cuenta es que las instancias delegadas estas pueden referirse a métodos y objetivos que normalmente no serían visibles en el punto en que se realiza la llamada. Por ejemplo, se puede usar un método privado para crear una instancia de delegado, y luego la instancia de delegado se puede devolver de un miembro público. Alternativamente, el objetivo de una instancia puede ser un objeto del que la persona pueda que llama eventualmente no sabe nada. mientras tanto el objetivo como el método deben ser accesibles para el código de creación. Y esto se puede llamar a un método en particular sobre un objeto en particular, el cual usar ese método y destino para la creación de delegados. Los derechos de acceso se ignoran efectivamente en el momento de la llamada.

Los delegados se pueden combinar de modo que cuando llame al delegado, se llame a una lista completa de métodos, potencialmente con diferentes objetivos. Cuando dije antes que un delegado contenía un objetivo y un método, eso fue una ligera simplificación. Eso es lo que contiene una instancia de delegado que representa un método. En aras de la claridad, me referiré a tales instancias de delegado como delegados simples.

Es significativo alcanzar que las instancias delegadas constantemente son inmutables. Cualquier objeto que los combine crea una nueva instancia de delegado para implicar la nueva lista de objetivos / métodos para llamar. Esto es como las cadenas: si llama, String.PadLeft por muestra, en realidad no cambia la cadena a la que la llama, solo devuelve una nueva cadena con el relleno adecuado.

La mezcla de entrambas instancias de delegado habitualmente se realiza utilizando el operador de suma, como si las instancias de delegado fueran cadenas o números. Restar uno de otro generalmente se hace con el operador de resta. Tenga en cuenta que cuando resta un delegado mezclado de otro, la resta funciona en términos de listas. Si la lista para restar no se encuentra en la lista original, el resultado es solo la lista original.

Los eventos son pares de métodos, adecuadamente decorados en IL para unirlos y realizar que los idiomas sepan que los métodos representan eventos. Los métodos corresponden a operaciones de añadir y quitar, cada una de las cuales toma un parámetro de instancia de delegado del mismo tipo. Lo que haga con esas operaciones depende de usted, sino el uso típico es sumar o eliminar el delegado de una lista de controladores para el evento.

Sin embargo, sería muy raro no recordar el valor de esta manera, hay momentos en que no desea proteger un evento con una simple variable delegada. Por ejemplo, en situaciones en que hay muchos eventos, pero es posible que solo unos pocos estén suscritos, podría poseer un mapa de alguna clave que describa el evento para el delegado que lo maneja al presente.

Puede utilizar un solo candado para todos sus eventos, o incluido para otras cosas, depende de su contexto. Tenga en cuenta que debe fijar el valor actual a una variable local adentro del bloqueo (para obtener el valor más reciente) y pronto probarlo para establecer la nulidad y ejecutarlo fuera del bloqueo.

La comprobación de nulidad se debe a que las variables delegadas se null producen cuando no hay instancias de delegado a las que llamar. Una modo de facilitar las cosas es utilizar una instancia de delegado no operativo como la instancia "predeterminada", que jamás se elimina.