Esbaide Yaziel Hernández Delgadillo

Programacion visual

Daniel Nuño

PVT14 Eventos

Los delegados aparecen con la palabra clave delegate y ya sea que aparrscan solos o anidados dentro de una clase, esta palabra no siempre significa que esta declarando un tipo de delegado.

Cualquier tipo de delegado se crea tiene los miembros heredados de sus tipos de padres, un constructor con sus parámetros y tres métodos adicionales. Los métodos no se pueden heredar de nada porque la firma varía según sea declarado el delegado.

Los puntos clave de datos en cualquier instancia de delegado en particular son el método al que se refiere el delegado y una referencia para llamar al método en (el *destino*). Para los métodos estáticos, no se requiere un objetivo. El CLR en sí mismo admite otras formas de delegado ligeramente diferentes, donde el primer argumento pasado a un método estático se mantiene dentro del delegado, o el objetivo de un método de instancia se proporciona como un argumento cuando se llama al método. Consulte la documentación para System. Delegateobtener más información sobre esto si está interesado, pero no se preocupe demasiado por eso.

Las instancias delegadas pueden referirse a métodos y objetivos que no son visibles en el punto en que se realiza la llamada. 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.

Tanto el objetivo como el método deben ser accesibles para el código de creación. En otras palabras, si (y solo si) puede llamar a un método en particular sobre un objeto en particular, puede usar ese método y destino para la creación de delegados.

Los delegados se pueden combinar de modo que cuando se llame el delegado, se llame una lista completa de métodos, con diferentes objetivos. Los delegados combinados pueden combinarse entre si, creado efectivamente una gran lista de delegados simples e la manera obvia. La combinación de dos instancias de delegado generalmente 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 combinado de otro, la resta funciona en términos de listas. Los operadores de suma y resta siempre funcionan como parte de la asignación: d1 += d2; es exactamente equivalente d1 = d1+d2; e igualmente para la resta.

Los eventos no son instancias delegadas. Los eventos son pares de métodos, los cuales los métodos corresponden a operaciones de agregar y quitar, las cuales toman u n parámetro de instancia de delegado del mismo tipo.

Los eventos en sí pueden ser declarados de dos maneras. El primero es con métodos explícitos de agregar y quitar, declarados de manera muy similar a las propiedades, pero con la eventpalabra clave. Aquí hay un ejemplo de un evento para el System.EventHandlertipo de delegado.

Los eventos de campos se crea una variable delegada y un evento ambos con el mismo tipo. El acceso al evento esta determinado por la declaración del evento, pero la variable delegada siempre es privada.

Un evento seguro para subprocesos de modo que cuando genera un evento siempre usa el valor mas reciente de la variable delegada, además de asegurarse de que las operaciones de agregar y quitar no interfieran entre sí, debe escribir el cuerpo de las operaciones de agregar y quitar usted mismo.

Para el bloqueo se debe tener en cuenta que se debe asignar el valor actual a una variable local dentro del bloque (esto para obtener el valor más reciente) y luego probarlo para determinar la nulidad y ejecutarlo fuera del bloqueo: mantener el bloqueo mientras se genera el evento es una muy mala idea. La verificación de nulidad se debe a que las variables delegadas se nullproducen cuando no hay instancias de delegado a las que llamar. Una forma de simplificar las cosas es utilizar una instancia de delegado no operativo como la instancia "predeterminada", que nunca se elimina. En ese punto, puede obtener el valor de la variable delegada (dentro de un candado si está seguro para subprocesos) y luego ejecutar la instancia de delegado. Si no hay objetivos delegados "reales" a los que llamar, el objetivo no operativo se ejecutará y eso es todo lo que sucederá.

Delegar instancias: otros métodos

 Los tipos de delegados también pueden permitir un comportamiento asincrónico utilizando el BeginInvoke/ EndInvokepar. Siguen el mismo modelo asíncrona que el resto de .NET, lo que permite proporcionar un controlador de devolución de llamada, junto con un objeto para almacenar información de estado.

Las llamadas a Thread.Sleepson solo para demostrar que la ejecución realmente ocurre en paralelo. La suspensión CountCharacterses tan grande como forzar al grupo de subprocesos del sistema a ejecutar las tareas en dos subprocesos diferentes: el grupo de subprocesos serializa las solicitudes que no tardan mucho para evitar crear más subprocesos de los necesarios.

Las llamadas a EndInvokebloquear hasta que el delegado se haya completado de la misma manera que las llamadas a Thread.Join bloquear hasta que los hilos involucrados hayan terminado. Los IAsyncResultvalores devueltos por las llamadas a BeginInvoke permiten el acceso al estado pasado como último parámetro BeginInvoke.

El hilo principal simplemente inicia las tareas asincrónicas y luego duerme lo suficiente como para dejar que todo el trabajo termine. (Los subprocesos del grupo de subprocesos son subprocesos en segundo plano; sin la Sleepllamada adicional, la aplicación finalizaría antes de que las llamadas del delegado terminen de ejecutarse).

Los delegados proporcionan una manera simple de representar una llamada a un método, potencialmente con un objeto de destino, como un dato que se puede pasar. Son la base de los eventos, que son convenciones efectivas para agregar y eliminar el código del controlador que se invocará en los momentos apropiados.