C#
Arreglos y colecciones

Tópicos Selectos de Computación I

19 de Septiembre de 2013

Arreglos

- En los arreglos de C#, los índices empiezan en cero. Los arreglos de C# funcionan de forma similar a como lo hacen en la mayoría de los lenguajes populares; existen, sin embargo, algunas diferencias que se deben conocer.

- Cuando se declara un arreglo , los corchetes ([]) deben ir después del tipo, no después del identificador. Colocar los corchetes después del identificador no es sintácticamente válido en C#.

Arreglos

- El tamaño de la matriz no forma parte de su tipo, como ocurre en el lenguaje C. Esto permite declarar una matriz y asignarle cualquier matriz de objetos **int**, independientemente de la longitud de la matriz.

```
int[] numbers;
numbers = new int[10];
 numbers = new int[20];
```

Arreglos

Vectores o matrices unidimensionales:

```
int[] numbers = new int[5];
```

Matrices:

```
string[,] names = new string[5,4];
```

Matrices de matrices (escalonadas):

```
byte[][] scores = new byte[5][];
for (int x = 0; x < scores.Length; x++) {
        scores[x] = new byte[4];
}
```

Matrices

Matriz de tres dimensiones:

```
int[,,] buttons = new int[4,5,3];
```

- Se pueden combinar matrices rectangulares y escalonadas.
- El siguiente código declara una matriz unidimensional que contiene matrices tridimensionales de matrices bidimensionales de tipo **int**:

```
int[][,,][,] numbers;
```

Colecciones

- Los datos estrechamente relacionados se pueden tratar con más eficacia si se agrupan en una colección. En lugar de escribir código independiente para tratar cada objeto individualmente, puede usar el mismo código para procesar todos los elementos de una colección.

- Para administrar una colección, use la clase System.Array y las clases de los espacios de nombres System.Collections, System.Collections.Generic y System.Collections.Concurrent para agregar, quitar y modificar elementos individuales o intervalos de elementos de la colección. También se puede copiar una colección en otra.

Colecciones

- Algunas clases System.Collections tienen capacidades de ordenación y la mayor parte están indizadas.
- La administración de la memoria se controla automáticamente.
- La sincronización proporciona seguridad para los subprocesos cuando se tiene acceso a los miembros de una colección.
- Algunas clases System.Collections pueden generar contenedores que hacen que la colección sea de solo lectura o que tenga un tamaño fijo.
- Cualquier clase System.Collections puede generar su propio enumerador que facilita la iteración por los elementos.

Colecciones

- Algunas clases System.Collections tienen capacidades de ordenación y la mayor parte están indizadas.
- La administración de la memoria se controla automáticamente.
- La sincronización proporciona seguridad para los subprocesos cuando se tiene acceso a los miembros de una colección.
- Algunas clases System.Collections pueden generar contenedores que hacen que la colección sea de solo lectura o que tenga un tamaño fijo.
- Cualquier clase System.Collections puede generar su propio enumerador que facilita la iteración por los elementos.

Colecciones

- En .NET Framework versión 2.0, las clases de colección genéricas proporcionan una nueva funcionalidad y facilitan la creación de colecciones con establecimiento inflexible de tipos. Vea los espacios de nombres System.Collections.Generic y System.Collections.ObjectModel.

- En .NET Framework versión 4, las colecciones del espacio de nombres System.Collections.Concurrent proporcionan operaciones eficaces y seguras para subprocesos con el fin de obtener acceso a los elementos de colección desde varios subprocesos.