آرایه ها در C#

آرایه ، با مشخص کردن نوع عنصر(Element) ، ابعاد (Dimension) و حد بالا و پایین آن (Upper & Lower)، تعریف میشود. این مشخصات در هر گونه تعریفی که از آرایه ارائه شود صدق میکنند. آرایه ها تنها میتوانند عناصری را در خود نگهداری کنند که از یک نوع تعریف شده باشند.

مقادیر در آرایه از نوع شیئ (Object) هستند. اشیاء آرایه یک مجموعه از آدرس هایی هستند که به مکان ذخیرهء مقادیر عناصر اشاره میکنند.

انواع آرایه از کلاس System.Array مشتق شده اند. این کلاس بدون توجه به عناصر یا بعد آرایه ها، نمایانگر آنهاست. عملیاتی که بر روی آرایه ها تعریف شده شامل موارد زیر است:

- معین کردن و تخصیص حافظه به آرایه براساس اندازه و حد پایین تعریف شده.
- شاخص گذاری (Index) بر آرایه برای خواندن و نوشتن مقادیر .
- محاسبه آدرس یک عنصر از یک آرایه(Managed Pointer).
- امکان بازیابی بعد، محدوده های بالا و پایین و تعداد مقادیر ذخیره شده در آرایه.

آرایه ها در C# ، از نوع شیئ (Object) هستند و مانند C++ و C ، تنها محدوده آدرسهای حافظه ای نیستند. به نمونه ای از تعریف و استفاده از آرایه ها توجه کنید:

```
int[] numbers = { 1, 2, 3, 4, 5 };

int lengthOfNumbers = numbers.Length;
```

یا:

```
class TestArraysClass
{
    static void Main()
    {
        //Declare and initialize an array
        int[,] theArray = new int[5, 10];
        System.Console.WriteLine("The array has {0} dimensions.",
              theArray.Rank);
    }
}
```

آرایه یک ساختار داده ای است که شامل تعدادی متغیر از یک نوع است و همانگونه که دیدید با TYPE تعریف میشود:

type[] arrayName;

یک آرایه دارای مشخصات زیر است:

- میتواند یکی از این سه نوع باشد :یک بعدی ، چند بعدی و دندانه ای.(Jagged)
- مقدار اولیهء آرایه های عددی به صفر و مقدار اولیهء آرایه های مرجعی (Refrence) به null منسوب میشوند.
- یک آرایهء دندانه ای (Jagged) آرایه ای از آرایه هاست و در نتیجه عناصر آن از نوع مرجعی (Refrence) میباشند پس مقدار اولیه آنها null است.
- شاخص آرایه ها از صفر شروع میشود پس عناصر یک آرایهء n بعدی از صفر آغاز میشود تا.n-1
- عناصر آرایه از هر نوعی میتوانند باشند حتی خود آرایه.(Jagged)
- آرایه ها از نوع مرجع هستند (Refrence Type) که از نوع پایه و اصلی و انتزاعی Array مشتق شده اند و میتوان از دستور foreach برای دستیابی به عناصر آرایه استفاده کرد.

یک مثال از سه نوع تعریف آرایه:

```
class TestArraysClass
{
    static void Main()
    {
        //Declare a single-dimensional array
        int[] array1 = new int[5];
        //Declare and set array element values
        int[] array2 = new int[] { 1, 3, 5, 7, 9 };
        //Alternative syntax
        int[] array3 = { 1, 2, 3, 4, 5, 6 };
        //Declare a two dimensional array
        int[,] multiDimensionalArray1 = new int[2, 3];
        //Declare and set array element values
        int[,] multiDimensionalArray2 = { { 1, 2, 3 }, { 4, 5, 6 } };
        //Declare a jagged array
        int[][] jaggedArray = new int[6][];
        //Set the values of the first array in the jagged array
        jaggedArray[0] = new int[4] { 1, 2, 3, 4 };
    }
}
```

قبلا گفتیم که سه نوع آرایه داریم: یک بعدی . چند بعدی و دندانه ای (Jagged)

حالا اولین نوع آرایه یعنی آرایه یک بعدی : (Singel-Dimensional Arrays) همونطور که گفتیم تعریف آرایه به صورت زیر هست:

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

string[] stringArray = new string[6];
```

اپراتور New برای ساختن آرایه و مقداردهی اولیه به اعضای آرایه به مقادیر پیش فرض براساس نوع آرایه بکار میرود . شما میتوانید در حین تعریف آرایه نیز به آن مقداردهی کنید که در این صورت نیازی به مشخص کردن تعداد اعضای آرایه نیست . چون با معرفی اعضا تعداد آنها نیز مشخص میشود

```
int[] array1 = new int[5] { 1, 3, 5, 7, 9 };

string[] weekDays = new string[] { "Sun", "Mon", "Tue", "Wed", "Thu",

                                   "Fri", "Sat" };

int[] array2 = { 1, 3, 5, 7, 9 };

string[] weekDays2 = { "Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat" };
```

میتوانید آرایه را بدون مشخص کردن محدوده آن معرفی کنید . اما در هنگام اختصاص دادن مقادیر به آرایه باید حتما از کلمه New استفاده نمایید:

```
int[] array3;
array3 = new int[] { 1, 3, 5, 7, 9 }; // OK
//array3 = {1, 3, 5, 7, 9}; // Error
SomeType[] array4 = new SomeType[10];
```

نتیجه تعریف یک آرایه به نوع تعریف شده آرایه بستگی دارد که آیا Value type است یا Refrence Type. به عنوان مثال در بالا اگرSomeTYpe از نوع Value Type باشد حاصل این عبارت آرایه ای شامل 10 نمونه از نوع SomeTYpe است. اگر از نوع Refrence Type باشد آرایه شامل 10 عنصر خواهد بود که هرکدام به null ارجاع خواهند داد.

فکر کنم توضیح مختصری در مورد Value Type , Refrence Type ها بد نباشه:

سه گروه type در #c تعریف شده Value Type , Refrence Type , Pointer Type :

متغیر هایی از نوع Value Type اطلاعات و داده ها را ذخیره میکنند در حالیکه متغیر های از نوع Refrence مراجع به داده ها و در واقع آدرس محل داده در حافظه را ذخیره میکنند pointer ها فقط در

حالت Unsafe قابل استفاده هستند. امکان تبدیل متغیری از نوع Value به متغیری از نوع Refrence و برگشت دوباره به Value وجود دارد (با استفاده از Boxing و Unboxing) اما برعکس نه !

نوع دوم آرایه ها , آرایه های چند بعدی هستند(Multidimentional Arrays) :

آرایه ها میتوانند بیش از یک بعد داشته باشند مثلا:

```
int[,] array = new int[4, 2];
int[, ,] array1 = new int[4, 2, 3];
```

که اولین آرایه , مشخص کننده یک آرایه 2بعدی است با 4 سطر و 2 ستون و دومین آرایه یک آرایه 3 بعدی است. میتوانید مقداردهی آرایه را در همان زمان معرفی آن انجام دهید به صورت زیر:

```
int[,] array2D = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 },{ 7, 8 } };
int[, ,] array3D = new int[,,] { { { 1, 2, 3 } }, { { 4, 5, 6 } } };
int[,] array4 = { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } };
```

و مانند ارایه های یک بعدی , میتوانید آرایه را بدون مشخص کردن محدوده آن معرفی کنید . اما در هنگام اختصاص دادن مقادیر به آرایه باید حتما از کلمه New استفاده نمایید:

```
int[,] array5;
array5 = new int[,] { { 1, 2 }, { 3, 4 }, { 5, 6 }, { 7, 8 } }; // OK

//array5 = {{1,2}, {3,4}, {5,6}, {7,8}}; // Error
```

مقداردهی به یک عضو از اعضای آرایه نیز به صورت زیر است:

```
array5[2, 1] = 25;
```

یه نوع از آرایه ها مونده بود که اونم الان میخوام بگم و بحث آرایه ها تموم بشه به امید خدا.... آرایه های دندانه ای یا Jagged Array .

این آرایه ها ، عناصرشون همگی آرایه هستند. یعنی در واقع آرایه ای از آرایه ها هستند. و اندازه و ابعادشون میتونه متفاوت از هم باشه. به یه مثال از تعریف و مقدار دهی آرایه توجه کنید:

```
int[][] jaggedArray = new int[3][];
jaggedArray[0] = new int[5];
jaggedArray[1] = new int[4];
jaggedArray[2] = new int[2];
```

قبل از استفاده از آرایه حتما باید اعضا و عناصرش رو تعریف کرد. در اینجا هر عنصر آرایه یک آرایه تک بعدی از اعداده. عنصر اول آرایه ای با 5 عنصر ، عنصر دوم آرایه ای با 4 عنصر و عنصر آخر آرایه ای با 2 عنصر هست. حتی میتونید همون موقع تعریف آرایه به عناصرش مقدار بدید و دیگه لازم نباشه که اندازه و ابعاد آرایه ای رو که از عناصر آرایه دندانه ایست ، مشخص کنید. مثل مثال زیر :

```
jaggedArray[0] = new int[] { 1, 3, 5, 7, 9 };
jaggedArray[1] = new int[] { 0, 2, 4, 6 };
jaggedArray[2] = new int[] { 11, 22 };
```

یا

```
int[][] jaggedArray2 = new int[][]
{
      new int[] {1,3,5,7,9},
      new int[] {0,2,4,6},
      new int[] {11,22}
};
```

یادتون باشه که فقط کلمه NEW رو فراموش نکنید چون مقدار دهیی بصورت اولیه برای عناصر وجود نداره و باید حتما NEW بشن.

```
int[][] jaggedArray3 =
{
      new int[] {1,3,5,7,9},
      new int[] {0,2,4,6},
      new int[] {11,22}
};
```

چون این آرایه ای از آرایه هاست پس عناصر اون از نوع Reference type هستن.

به هر عضو آرایه هم بصورت منفرد میتونین دسترسی داشته باشین.

```
// Assign 77 to the second element ([1]) of the first array ([0]):
jaggedArray3[0][1] = 77;
// Assign 88 to the second element ([1]) of the third array ([2]):
jaggedArray3[2][1] = 88;
```

میتونین یه آرایه دندانه ای رو با یه آرایه چند بعدی هم ترکیب کنین. مثال زیر یک آرایه دندانه ای یک بعدیست که عناصر آن آرایه هایی 2 بعدی با اندازه های مختلف هستن.

```
int[][,] jaggedArray4 = new int[3][,]
{
      new int[,] { {1,3}, {5,7} },
      new int[,] { {0,2}, {4,6}, {8,10} },
      new int[,] { {11,22}, {99,88}, {0,9} }
};
```

فکر میکنین کد زیر چه مقداری رو از آرایه بالا به ما نشون میده؟

```
System.Console.Write("{0}", jaggedArray4[0][1, 0]);
```

متد Length تعداد آرایه های موجود در آرایه دندانه ای رو به ما نشون میده.

```
System.Console.WriteLine(jaggedArray4.Length);
```

جواب اولی 5 و جواب دومی 3 هستش.

حالا برای اختتام بحث یه مثال میزنیم از آرایه دندانه ای با عناصر مختلف. دقت کنین :

```
    class ArrayTest
    {
        static void Main()
        {
            // Declare the array of two elements:
            int[][] arr = new int[2][];

            // Initialize the elements:
            arr[0] = new int[5] { 1, 3, 5, 7, 9 };
            arr[1] = new int[4] { 2, 4, 6, 8 };

            // Display the array elements:
            for (int i = 0; i < arr.Length; i++)
            {
```

```
            System.Console.Write("Element({0}): ", i);

            for (int j = 0; j < arr[i].Length; j++)
            {
                System.Console.Write("{0}{1}", arr[i][j],
                          j == (arr[i].Length - 1) ? "" : " ");
            }
            System.Console.WriteLine();
        }
    }
```

خروجی این برنامه به این شکله :

Element (0) : 1 3 5 7 9

Element (1) : 2 4 6 8

آدرس وبلاگ : http://Csharpblog.blogfa.com
Email: Csharpblog@yahoo.com
نویسنده: آتاناز

حقوق این مقاله متعلق به وبلاگ http://Csharpblog.Blogfa.com می باشد.