زنگ سی‌شارپ – قسمت سیزدهم

نوشته‌ی مسعود درویشیان

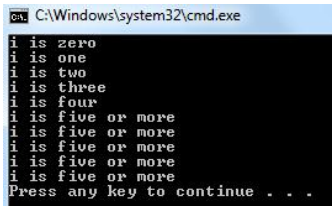
لینک مستقیم این مطلب در وب‌تارگت

در قسمت دوازدهم با دستور switch و کلمات‌کلیدی break و continue آشنا شدید. در این قسمت دستور goto وتوضیحات تکمیلی‌تر دستور switch بیان می‌شود. همان‌طور که در قسمت قبل بیان شد، دستور switch یک متغیر را با چندین مورد مقایسه می کند و آن مورد را که با متغیر مطابقت دارد، انتخاب می کند.

به مثال زیر توجه کنید:

```
using System;
class Example
{
    static void Main()
    {
        int i;
        for (i = 0; i < 10; i++)
            switch (i)
            {
                case 0:
                    Console.WriteLine("i is zero");
                    break;
                case 1:
                    Console.WriteLine("i is one");
                    break;
                case 2:
                    Console.WriteLine("i is two");
                    break;
                case 3:
                    Console.WriteLine("i is three");
                    break;
                case 4:
                    Console.WriteLine("i is four");
                    break;
                default:
                    Console.WriteLine("i is five or more");
                    break;
            }
    }
}
```

خروجی:



همان‌طور که می‌بینید، هربار از طریق حلقه مقدار i به ساختار switch داده شده و case مربوط به آن اجرا می‌شود. هنگامی که مقدار i بیشتر از ۴ است دیگر در هر بار فقط default اجرا می‌شود زیرا case مربوط به آن موجود نیست.

شما می‌توانید هر integer type ای را توسط ساختار switch کنترل کنید، از جمله کاراکتر. به مثال زیر توجه کنید:

```
using System;
class Example
{
    static void Main()
    {
        char ch;

        for (ch = 'A'; ch <= 'E'; ch++)
        {
            switch (ch)
            {
                case 'A':
                    Console.WriteLine("ch is A");
                    break;
                case 'B':
                    Console.WriteLine("ch is B");
                    break;
                case 'C':
                    Console.WriteLine("ch is C");
                    break;
                case 'D':
                    Console.WriteLine("ch is D");
                    break;
                case 'E':
                    Console.WriteLine("ch is E");
                    break;
            }
        }
    }
}
```

خروجی:

```
ch is A
ch is B
ch is C
ch is D
ch is E
```

همان‌طور که می‌بینید در این مثال از default استفاده نشده چراکه این بخش از ساختار switch اختیاری است.

در سی‌شارپ قانونی به اسم no fall-through وجود دارد. طبق این قانون، کامپایلر بعد از statement sequence هر case به سراغ case بعدی نمی‌رود، چراکه این امر برای زبان سی‌شارپ یک خطا محسوب می‌شود و به همین دلیل است که در پایان هرcase از break استفاده می‌کنیم تا کامپایلر به‌کلی از ساختار switch خارج شود و به ادامه‌ی برنامه و خط‌کدها بپردازد. قسمت default نیز نباید fall-through باشد و باید توسط break پایان یابد (از روش‌های دیگر نیز می‌توان قانون no fall-through را رعایت کرد مانند استفاده از goto به‌جای break که در ادامه‌ی این مقاله به شرح آن می‌پردازیم).

شما همچنین می‌توانید چندین case داشته باشید که همگی یک statement sequence دارند:

```
// Empty cases can fall through.
using System;
class Example
{
    static void Main()
    {
        int i;
        for (i = 1; i < 5; i++)
            switch (i)
            {
                case 1:
                case 2:
                case 3: Console.WriteLine("i is 1, 2 or 3");
                    break;
                case 4: Console.WriteLine("i is 4");
                    break;
            }
    }
}
```

خروجی:

```
i is 1, 2 or 3
i is 1, 2 or 3
i is 1, 2 or 3
i is 4
```

در این مثال، اگر i شامل مقادیر ۱، ۲ و ۳ باشد سپس اولین WriteLine() اجرا می‌شود. اگر i برابر با ۴ باشد آن‌گاه دومین WriteLine() اجرا خواهد شد.

The goto

goto یک jump statement غیر شرطی است. هنگامی‌که برنامه به این کلمه می‌رسد، به مکان مشخصی از کد که توسط goto مشخص شده است، پرش می‌کند. goto سال‌ها قبل از چشم برنامه‌نویسان افتاد چراکه موجب می‌شد کدنویسی شما مانند اسپاگتی شود! اگرچه هیچ موقعیت برنامه‌نویسی به‌وجود نمی‌آید که به goto نیاز داشته باشید و در واقع سی‌شارپ برای این‌که یک زبان کامل باشد به goto نیاز ندارد اما هنوز هم به‌ندرت (و در برخی موارد به‌صورت مفید) استفاده می‌شود. با این تفاصیر، اگر از goto به‌صورت هوشمندانه استفاده شود می‌تواند سودمند باشد. نگرانی اصلی برنامه‌نویسان این است که استفاده زیاد از goto باعث شود برنامه ناخوانا و به‌هم ریخته شود اما در برخی از موارد برعکس است و به‌جای به‌هم ریختگی باعث واضح‌تر شدن کد خواهد شد(در هرحال استفاده از این دستور پیشنهاد نمی‌شود مگر به‌صورت هوشمندانه و به‌جا). goto برای انجام عملیات نیاز به یک label دارد. label یکی از شناسه‌های سی‌شارپ است که بعد از آن علامت دونقطه (:) قرار می‌گیرد. label باید در همان بلاک و متدی باشد که goto قرار دارد (توضیح متد را در مقالات بعدی دنبال کنید). به‌عنوان مثال، برنامه زیر یک حلقه است که اعداد ۱ تا ۲۰ را توسط goto و label نمایش می‌دهد:

```
using System;
class Example
{
    static void Main()
    {
        int x = 1;

    Loop1: // this is label

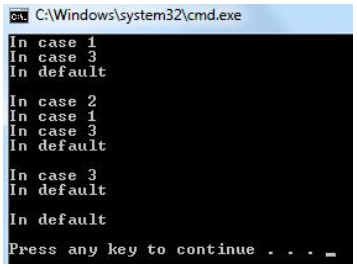
        if (x <= 20)
        {
            Console.WriteLine(x);
            x++;
            goto Loop1; // it goes to the Loop1 label
        }
        Console.WriteLine("Done!");
    }
}
```

goto همچنین می‌تواند در یک ساختار switch به case و default دلخواه پرش کند. در این مورد، case و default نقش label را بازی می‌کنند بنابراین می‌توانند هدف goto قرار گیرند. نکته این‌جاست که goto باید درون switch مربوطه باشد و شما نمی‌توانید از بیرون به درون یک switch پرش کنید.

به مثال زیر که از goto در ساختار switch استفاده می‌کند توجه کنید:

```
// Use goto with a switch.
using System;
class Example
{
    static void Main()
    {
        for (int i = 1; i < 5; i++)
        {
            switch (i)
            {
                case 1:
                    Console.WriteLine("In case 1");
                    goto case 3;
                case 2:
                    Console.WriteLine("In case 2");
                    goto case 1;
                case 3:
                    Console.WriteLine("In case 3");
                    goto default;
                default:
                    Console.WriteLine("In default");
                    break;
            }
            Console.WriteLine();
        }
        // goto case 1; // Error! Can't jump into a switch.
    }
}
```

خروجی:



در این مثال ساختار switch درون یک حلقه‌ی for قرار دارد و در هر دور حلقه، مقدار i به ساختار switch داده می‌شود. توجه کنید که در ساختار switch چگونه goto به case های مختلف و default پرش می‌کند. نکته‌ی دیگر این‌جاست که case ها با break پایان نیافته‌اند (به‌جز default) چراکه استفاده از break در این‌جا بی‌مورد و بی‌تاثیر است زیرا هر case توسط goto به case دیگری فرستاده شده و نهایتاً در قسمت default از ساختار switch خارج می‌شود. همان‌طور که گفته شد، نمی‌توانید از بیرون به درون ساختار switch پرش کنید و اگر علامت کامنت را از ابتدای آخرین خط‌کد برنامه حذف کنید، برنامه کامپایل نمی‌شود. استفاده از goto در ساختار switch در برخی موارد خاص می‌تواند مفید باشد اما پیشنهاد نمی‌شود.

یکی از استفاده‌های مفید goto می‌تواند برای خارج شدن از حلقه‌های تودرتو با تو رفتگی زیاد باشد.

به این مثال دقت کنید:

```
// Demonstrate the goto.
using System;
class Example
{
    static void Main()
    {
        int i = 0, j = 0, k = 0;
        for (i = 0; i < 10; i++)
        {
            for (j = 0; j < 10; j++)
            {
                for (k = 0; k < 10; k++)
                {
                    Console.WriteLine("i, j, k: " + i + " " + j + " " + k);
                    if (k == 3) goto stop;
                }
            }
        }
    stop:
        Console.WriteLine("Stopped! i, j, k: " + i + ", " + j + " " + k);
    }
}
```

خروجی:

```
i, j, k: 0 0 0
i, j, k: 0 0 1
i, j, k: 0 0 2
i, j, k: 0 0 3
Stopped! i, j, k: 0, 0 3
```

حذف goto در این مثال موجب می‌شود برای خاج شدن از سه حلقه که تودرتو هستند، از سه if و break استفاده کنید. در این مثال استفاده از goto کد را ساده‌تر می‌کند. هرچند که این یک مثال ساختگی است، اما می‌توانید موقعیت‌هایی را به‌وجود آورید که استفاده از goto مفید واقع شود. آخرین نکته این است که شما می‌توانید توسط goto به خارج از یک بلاک پرش کنید اما نمی‌توانید به درون یک بلاک پرش کنید.

شاید در فهم این مثال کمی با مشکل برخورده باشید ولی جای نگرانی نیست، چراکه توضیحات کافی در مورد حلقه‌های تودرتو خواهد داده شد و مثال‌ها و تمرینات جالبی را انجام خواهیم داد.

کلیه حقوق مادی و معنوی برای وب‌سایت وب‌تارگت محفوظ است.
استفاده از این مطلب در سایر وب‌سایت‌ها و نشریات چاپی تنها با ذکر و درج لینک منبع مجاز است.