HNDIT1012 Visual Application Programming

Week 4

Switch Statement

Switch is a selection statement that chooses a single switch section to execute from a list of candidates based on a pattern match with the match expression.

Eg:

```
int caseSwitch = 1;
switch (caseSwitch)
{
        case 1:
                Console.WriteLine("Case 1");
                break;
        case 2:
                Console.WriteLine("Case 2");
                break;
        default:
                Console.WriteLine("Default case");
                break;
}
```

Example

```
class Switch
{
static void Main()
{
                Console.Write("Enter your selection (1, 2, or 3): ");
                string s = Console.ReadLine();
                int n = Int32.Parse(s);
                switch (n)
                {
                case 1:
                                Console.WriteLine("Current value is 1");
                                break;
                case 2:
                                Console.WriteLine("Current value is 2");
                                break;
                case 3:
                                Console.WriteLine("Current value is 3");
                                break;
                default:
                                Console.WriteLine("Sorry, invalid selection.");
                                break;
                }

Console.WriteLine("Press any key to exit.");
Console.ReadKey();
}
```

Loops / Iterations

Loops are used to execute a block of statements several times. In C# following types of loops are discussed here.

- for loop
- while loop
- do while loop

for loop

- The for statement executes a statement or a block of statements while a specified Boolean expression evaluates to true .
- At any point within the for statement block, you can break out of the loop by using the break statement, or step to the next iteration in the loop by using the continue statement. You can also exit a for loop by the goto, return, or throw statements

Structure of the for statement

- The for statement defines initializer, condition, and iterator sections:

```
for (initializer; condition; iterator)
        body
```

All three sections are optional. The body of the loop is either a statement or a block of statements. The statements in the initializer section are executed only once, before entering the loop.

The condition section, if present, must be a boolean expression. That expression is evaluated before every loop iteration. If the condition section is not present or the boolean expression evaluates to true , the next loop iteration is executed; otherwise, the loop is exited.

The iterator section defines what happens after each iteration of the body of the loop. The iterator section contains zero or statement expressions, separated by commas.

Example

```
for (int i = 0; i < 5; i++)
{
        Console.WriteLine(i);
}
```

Another Example

```
int i;
int j = 10;
for (
        i = 0, Console.WriteLine($"Start: i={i}, j={j}");
        i < j;
        i++, j--, Console.WriteLine($"Step: i={i}, j={j}")
        )
{
        // Body of the loop.
}
```

Initializer Section

condition

Iterator Section

The following example defines the infinite for loop:

```
for ( ; ; )
{
        // Body of the loop.
}
```

Break statement

The break statement terminates the closest enclosing loop or switch statement in which it appears. Control is passed to the statement that follows the terminated statement, if any.

```
for (int i = 1; i <= 100; i++)
{
        if (i == 5)
                {
                        break;
                }
        Console.WriteLine(i);
}
```

Output:

```
1
2
3
4
```

While Loop

- The while statement executes a statement or a block of statements while a specified Boolean expression evaluates to true . Because that expression is evaluated before each execution of the loop, a while loop executes zero or more times. This differs from the do loop, which executes one or more times.
- At any point within the while statement block, you can break out of the loop by using the break statement.
- You can step directly to the evaluation of the while expression by using the continue statement. If the expression evaluates to true , execution continues at the first statement in the loop. Otherwise, execution continues at the first statement after the loop.
- You can also exit a while loop by the goto, return, or throw statements

Example

```
int n = 0;
while (n < 5)
{
        n++;
}
TextBox1.Text= Convert.ToInt32(n);
```

do ... while loop

The while loop tests the condition before executing the code following the while .

The do ... while loop executes the code first, and then checks the condition. The do while loop is shown in the following code:

```
int counter = 0;
do
{
        Console.WriteLine($"Hello World! The counter is {counter}");
        counter++;
} while (counter < 10);
```

Thank You