C# - NULLABLES

C# provides a special data types, the **nullable** types, to which you can assign normal range of values as well as null values.

For example, you can store any value from -2,147,483,648 to 2,147,483,647 or null in a Nullable<Int32> variable. Similarly, you can assign true, false, or null in a Nullable<bool> variable. Syntax for declaring a **nullable** type is as follows –

```
< data_type> ? <variable_name> = null;
```

The following example demonstrates use of nullable data types –

Live Demo

```
using System;

namespace CalculatorApplication {
   class NullablesAtShow {
      static void Main(string[] args) {
         int? num1 = null;
         int? num2 = 45;

         double? num3 = new double?();
         double? num4 = 3.14157;

         bool? boolval = new bool?();

         // display the values
         Console.WriteLine("Nullables at Show: {0}, {1}, {2}, {3}", num1, num2, num3, num4);
         Console.WriteLine("A Nullable boolean value: {0}", boolval);
         Console.ReadLine();
      }
   }
}
```

When the above code is compiled and executed, it produces the following result –

```
Nullables at Show: , 45,  , 3.14157
A Nullable boolean value:
```

The Null Coalescing Operator ??

The null coalescing operator is used with the nullable value types and reference types. It is used for converting an operand to the type of another nullable *ornot* value type operand, where an implicit conversion is possible.

If the value of the first operand is null, then the operator returns the value of the second operand, otherwise it returns the value of the first operand. The following example explains this –

Live Demo

```
using System;

namespace CalculatorApplication {
   class NullablesAtShow {
      static void Main(string[] args) {
         double? num1 = null;
         double? num2 = 3.14157;
         double num3;

         num3 = num1 ?? 5.34;
         Console.WriteLine(" Value of num3: {0}", num3);

         num3 = num2 ?? 5.34;
         Console.WriteLine(" Value of num3: {0}", num3);
         Console.ReadLine();
      }
   }
}
```

When the above code is compiled and executed, it produces the following result –

```
Value of num3: 5.34
Value of num3: 3.14157
```