C# For Beginners

Enumerations

Contents:

- The type enum
- The object Enum (with capital E)
- When to use enum

Farid Naisan, Univerisity Lecturer, Malmö University, farid.naisan@mau.se

"enum" is a type

- The type "enum" (with a lowercase 'e') is one of the five core C# types:
  - **class, interface, delegate, struct, enum**
- .NET Framework has hundreds of predefined enums
  - DaysOfWeek, DialogResult, etc are examples.
- Enums are used to group a set of named constants that are related to each other
  - Days of a week, countries of the world.

Example

- An enum is defined in much the same way as a class.
  - An access modifier
  - The keyword enum
  - An enum name
  - Members are separated by a comma
- Comparing to a class definition, the keyword class is replaced by the keyword enum.
- An enum cannot have fields or methods.
  - It holds only a group of constant names with an underlying value.

```
public enum CardSuits
{
    Club,       //=0
    Spade,      //=1
    Diamond,    //=2
    Heart       //=3
}
```

```
public class Car
{
}
```


Member data type and initialization

- The default underlying type for the members is **int** (system.Int32)
- Other integral types can also be used.
  - byte, int, long,uint, ulong,etc

```
public enum SomeBigStuff : long
{


}
```

```
public enum CharCards
{
    Ace  = 1,
    Jack = 10,
    Queen,     // = 11
    King       // = 12
}
```

```
public enum Countries
{
    Afghanistan = 93,
    Sweden = 46,
    UnitedKingdom = 44,
    UnitedStatesOfAmerica = 1
}
```

Declaring and using enums

- A variable of an **enum** type can be declared exactly in the same way as you declare other value types, such int and double.

```
private CharCards fortuneCard = CharCards.Ace;
```

- enums can be nested inside a class as a part of the class when it relates to the class.
- But if an enum is common between different classes, it should be declared public or internal just as a class.
  - Save the enum in its own code file no matter the size.

Conversion between enum and integral types

- An enum value can be converted to its corresponding underlying value and vice-versa.
- Different ways but using type-casting is simple.
- To convert an int variable to its corresponding enum member:

```
Countries country = (Countries)cmbCountry.SelectedIndex;//int to enum

double price = GetCallRatePerMinuteFor(country);
```

Note·countries is an enum, electedIndex is an integer

- To convert an enum variable to the number that it represents:

```
cmbCountry.SelectedIndex = (int)country;          //enum to int
```

Use of enum variables

An effective way of using enums in switch statements!

Make a note of how the enum-members are used in code (Countries·xxx)

```
//Use hard-coded values as an example
public double GetCallRatePerMinuteFor(Countries country)
{
    double vat = 0.0;   //value added tax in percent
    double price = 0.0;  //per minute, excl. vat

    switch (country)
    {
        case Countries.Afghanistan:  //no vat
            price = 0.354;
            break;
        case Countries.Sweden:
        case Countries.UnitedKingdom:
            price = 0.02;
            vat = 25.0;
            break;
        case Countries.UnitedStatesOfAmerica:
            price = 0.01;
            vat = 7.5;
            break;
    }

    price += vat / 100.0;
    return price;
}
```


The object Enum

- .NET Framework provides a ready-to-use object called **Enum (capital E)** that can efficiently be used together with enums to manipulate its values.
- The **Enum** object has several useful methods:
  - GetName
  - GetNames
  - GetValue
  - GetValues
- These methods can be specially useful when working with GUI components.

Example with a combobox

- In the code example here, **cmbNumber** and **cmbSuit** are both **ComboBoxes** (Windows Forms controls), while **CardSuits** is an enum type.

```
private void InitializeGUI()
{
    cmbSuit.DataSource = Enum.GetValues(typeof(CardSuits));
    cmbSuit.SelectedIndex = (int)CardSuits.Heart;

    for (int number = 0; number < 10; number++)
    {
        string strNo = string.Format("{0,2}", number + 1);
        cmbNumber.Items.Add(strNo);
    }
    cmbNumber.Items.AddRange(Enum.GetNames(typeof(CharCards)));
}
```


Summary

- enum is one of the five types that the whole C# language is built on.
- enum is used to group a set of related named constants.
- Variables of an enum type can only hold values (named constants) that are defined as members.
- The underlying data type for the members is int by default.
- If you do not specify a value to each member, the members are initialized to 0, 1, 2, n.
- **Use enums very much**. Make also use of the object **E**num with your enums.