26

Generics

Generic Classes

- Generic class is a class, which contains one or more "type parameters".
- You must pass any data type (standard data type / structure / class), while creating object for the generic class.

Generic Class - Example

```
class ClassName<T>
{
    public T FieldName;
}
```

Object of Generic Class - Example

```
ClassName<int> referenceVariable = new ClassName<int> ( );
```

- The same field may belong to different data types, w.r.t. different objects of the same class.
- You will decide the data type of the field, while creating the object, rather than while creating field in the class.
- It helps you in code reuse, performance and type-safety.
- You can create your own generic-classes, generic-methods, generic-interfaces and generic-delegates.
- You can create generic collection classes.
    - The .NET framework class library contains many new generic collection classes in System.Collections.Generic namespace.

- The generic type parameter (T) acts as "temporary data type", which represents the actual data type, provided by the user, while creating object.
- You can have multiple "generic type parameters" in the same class (for use for different fields.
- Generics are introduced in C# 2.0.

Generic Constraints

- Generic Constraints are used to specify the types allowed to be accepted in the "generic type parameter".
  - where T : class
  - where T : struct
  - where T : ClassName
  - where T : InterfaceName
  - where T : new( )

Generic Constraints - Example

```
class ClassName<T> where T : class
{
    public T FieldName;
}
```

Object of Generic Class - Example

```
ClassName<int> referenceVariable = new ClassName<int> ( );  //error
```

- Advantage: You can restrict what type of data types (class names) allowed to be passed while creating object.

- In C#, constraints are used to restrict a generics to accept only particular type or its derived types.
- By using 'where' keyword, we can apply constraints on generics.
- You can apply multiple constraints on generic classes or methods based on your requirements.
    - Ex: where T : class where T2 : class

Generic Methods

- Generic Method is a method that has one or more generic parameter(s).
- You can restrict what type of data types to be allowed to be passed to the parameter while calling the method.

Generic Method - Example

```
public void MethodName<T>
{
}
```

Calling Generic Method - Example

```
MethodName<datatype>(valueHere);
```