Some more Concepts of DOT NET

Casting Objects - I

```
Student s = new Student(12,"abc"); //Student is a Base class
Sy sy = new Sy(13, "SYStudent",'D'); // sy is a derived class from Student
Console.WriteLine("Type Casting - 1");
Student s1 = new Student(555, "newbase1");
Console.WriteLine("Before casting");
s1.printStudent();
Console.WriteLine("After casting");
s1=sy;
s1.printStudent();
```

Output -1

```
Type Casting - 1
Before casting
ConsoleApplication2.Student
555
newbase1
After casting
ConsoleApplication2.Sy
13
SYStudent
```

Casting Objects - II

```
Console.WriteLine("Type Casting - 3");
  Sy s3 = new Sy(23, "newderived", 'D');
  Student s4 = new Student(54, "newbase");
  Console.WriteLine("Before casting");
  s4.printStudent();
  s4 = (Student)s3;
  Console.WriteLine("After casting");
  s4.printStudent();

  Console.ReadKey();
```

Output- 2

```
 Type Casting - 3
Before casting
ConsoleApplication2.Student
54
newbase
After casting
ConsoleApplication2.Sy
23
newderived
```

Casting Objects - III

```
Console.WriteLine("Type Casting - 3");
Sy sy2 = new Sy(555, "newderived1", 'D');
Console.WriteLine("Before casting");
sy2.printStudent();
Console.WriteLine("After casting");
sy2 = s;
sy2.printStudent();
Console.ReadKey();
```

Output -3

- Compile Error. (Cannot implicitly convert Student to Sy)

Casting Objects - IV

```
Console.WriteLine("Type Casting - 4");
    Student s = new Student(12,"abc");
Sy sy2 = new Sy(555, "newderived1", 'D');
Console.WriteLine("Before casting");
sy2.printStudent();
Console.WriteLine("After casting");
sy2 = (Sy)s;

sy2.printStudent();
Console.ReadKey();
```

Output -4

- Run time Error. (Invalid Cast Exception)

```
static void Main(string[] args)
    {
        sy sy1 = new sy();
        student s1 = sy1;
        s1.stream = "CS";
    }
static void Main(string[] args)
    {
        student s1 = new sy();
        sy sy1 = (sy)s1;
        sy1.stream = "CS";
    }

static void Main(string[] args)
    {
        student s1 = new student();
        sy sy1 = (sy)s1;
    }
```

ASP.NET File Types

File Name

- **Ends with .aspx**

These are ASP.NET web pages. They contain the user interface and, optionally, the underlying application code. Users request or navigate directly to one of these pages to start your web application.

- **Ends with .ascx**

These are ASP.NET user controls. User controls are similar to web pages, except that the user can't access these files directly. Instead, they must be hosted inside an ASP.NET web page. User controls allow you to develop a small piece of user interface and reuse it in as many web forms as you want without repetitive code.

- **web.config**

This is the configuration file for your ASP.NET application. It includes settings for customizing security, state management, memory management, and much more.

- **global.asax**

This is the global application file. You can use this file to define global variables (variables that can be accessed from any web page in the web application) and react to global events (such as when a web application first starts).

- **Ends with .cs**

These are code-behind files that contain C# code. They allow you to separate the application logic from the user interface of a web page. We'll introduce the code-behind model in this chapter and use it extensively in this book.

ASP.NET Folders

- **App_Browsers**

Contains .browser files that ASP.NET uses to identify the browsers that are using your application and determine their capabilities. Usually, browser information is standardized across the entire web server, and you don't need to use this folder.

- **App_Code**

Contains source code files that are dynamically compiled for use in your application.

- **App_GlobalResources**

Stores global resources that are accessible to every page in the web application. This directory is used in localization scenarios, when you need to have a website in more than one language.

- **App_LocalResources**

Serves the same purpose as App_GlobalResources, except these resources are accessible to a specific page only.

- **App_WebReferences**

Stores references to web services, which are remote code routines that a web application can call over a network or the Internet.

- **App_Data**

Stores data, including SQL Server Express database files

- **App_Themes**

Stores the themes that are used to standardize and reuse formatting in your web application.

- **Bin**

Contains all the compiled .NET components (DLLs) that the ASP.NET web application uses. For example, if you develop a custom component for accessing a database