Asp.Net

Ch.5 :: Working With XML

Introduction to XML :

- XML stands for Extensible Markup Language.
- XML is a markup language much like HTML.
- XML responsible for managing the HTML standard.
- XML was designed to carry data, not to display data.

Introduction to XML :

- XML tags are not predefined. You must define your own tags.
- XML is designed to be self-descriptive.
- XML is a document-processing standard.
- XML is choice for dynamically generated content, including non-static web pages.
- XML is used generally to create simple and short database.

Difference between XML and HTML

- XML stands for Extended Markup Language.
- HTML stands for Hyper Text Markup Language.
- XML is used for database creation.
- HTML is used for Designing the website.
- XML is case sensitive.
- HTML is not case sensitive.
- XML does require that that tags opened are required to be closed.
- HTML tags does not require to close.
- XML tags are defined by the programmer.
- HTML tags are predefined by HTML standards.

XML Parser (ચકાસણી) :

- As we know that in XML we can define our own tags.
- As we define our own tag we have our own set of rules.
- Now we need to check whether the tags which we define are following the standards provided by XML or not, to fulfill the task we can use XML Parser as a small and auto-integrated software.

XML Parser (ચકાસણી) :

- XML Parser is small software which would read XML file for two purpose.
  - ❑ It checks whether the XML file is well formed and verified or not.
  - ❑ It also makes sure that XML file meets the defined structures, validation and constraints or not.

Rules for Creating XML file :

- XML file begin with information about the version of the XML file. For Visual Studio it would be UTF-8.
- Every XML document can have only one root element. All other elements should be inside the main element.
- Every Start tag which we write must have an end tag.
- Start tag are defined by using **< >** symbol and end tag is defined by **</ >** symbol.
  - For example **<start>**…… **</start>**
- End tag cannot contain any information. It should just contain the tag name preceded by **/** symbol.

Rules for Creating XML file :

- If tag does not contain any data then it also acts as end tag.
  - For example **<employee id="1" />**
- All the attributes value must be defined between double quotes.
  - For example **<employee id="1" />**
- All the tags must be properly nested in XML file.
- XML is case sensitive language. So tag **<Employee>** and **<employee>** both are different tag in XML
- We can specify any comments in XML by using starting **<!….** And ending with **>**

XML file Example :

```
<? xml version=“1.0” encoding=“utf-8” ?>
   <Classes>
       <Class>
             <ID>1</ID>
             <Name>Monarch Sankul</Name>
             <City>Lathi</City>
       </Class>
       <Class>
             <ID>2</ID>
             <Name>Monarch Computer</Name>
             <City>Babra</City>
       </Class>
   </Classes>
```

XML namespace and classes

- .NET provides a namespace which is used to work with XML file called **System.XML** namespace.
- Classes

Class	**Description**
XmlDocument	It is used to create XML file.
XmlDataDocument	It is used to create data based XML document
XmlNode	It is used to XML node element for XML file.

XML namespace and classes

Class	Description
XmlComment	It is used to provide comment in XML document.
XmlElement	It is used to create XML element in XML file.
XmlAttribut	It is used to create attribute for XML element.
XmlNodeList	It is used to create and store list of node collection elements in XML file.

Writing DataSet to XML :

- We can write data from DataSet or DataTable in XML file using tow methods.
  - WriteXML( ) method
  - WriteXmlSchema( ) method
- WriteXML( ) method :
  - This method is used to read data from DataSet and write data as XML data.
  - When we write a data from DataSet we must follow following rules.
    - All tables of DataSet are treated as Main Elements of XML file.

Writing DataSet to XML :

- All rows of tables are treated as Sub Element of XML file.
- All Columns of tables are treated as Data Element which contain the actual data of table.
- Also there is only one root element as XML rule.

Writing DataSet to XML :

- WriteXmlSchema( ) method :
  - It is a small bit of XML Schema Definition (XSD) file of XML basic part.
  - XML schema is a file which contain rules for defining XML element and their data.
  - If we use only WriteXML( ) method, the only data would be written from dataset to XML file. But with WriteXmlSchema( ) method, the schema information would be write with the XML file.

Def. Demonstrate use of Write XML

- Step-1
  - Add a new website with a webpage.
  - Rename the webpage as writeXML.aspx
- Step-2
  - Add a button with ID=btnWriteXML & Text="WriteXML"

Def. Demonstrate use of Write XML

- Step-3
  - Add following code in button click event.

```
SqlConnection con = new
   SqlConnection(@ “Connectionstring”);
con.Open();
SqlDataAdapter da = new
   SqlDataAdapter("Select * from
   employee", con);
DataSet ds = new DataSet();
da.Fill(ds);
```

Def. Demonstrate use of Write XML

- Step-3 Continue…

```
string st = Server.MapPath(@"~\");
ds.WriteXml(st + "\\Employee.xml");
Response.Write("Data Written! Success");
```

- Step-5
  - Run and check the folder.

Reading DataSet from XML :

- We can read XML file and store the information in form of DataSet or DataTable by using two methods.
- These methods can be given as:
  - ReadXml( ) method
    - It is used to read XML data from XML file to a DataSet or DataTable.
    - When we read DataSet or DataTable it follows following rules:
      - All main elements under root element are treated as different tables
      - It would automatically store data in XML way

Reading DataSet from XML :

- ReadXmlSchema( ) method
  - It is used to read XML Schema Definition file from XML files.
  - XML Schema is a file which has rules for defining XML element and their data.

Def. Demonstrate use of ReadXML

- Step-1
  - Add a new website with a webpage.
  - Rename the webpage as ReadXML.aspx
- Step-2
  - Add a button with ID=btnReadXML & Text="ReadXML"

Def. Demonstrate use of ReadXML

- Step-3
  - In button click event add following code

```
DataSet ds = new DataSet();
ds.ReadXml(MapPath("Employee.xml"));
GridView1.DataSource=ds;
GridView1.DataBind();
```

- Step-4
  - Run, click on the button and check.

Web Services :

- Web Services are one of the new strongest features that are introduced by Microsoft.
- It is wet based functionality provided by ASP.NET.
- Web Services can be defined as group of web methods where each web method provides some specific functionality .
- We were create simple methods in web page now we will create web methods.
- Web Methods are somewhat same as methods that we use.

Web Services :

- The main feature of Web Service is that it can be used by different applications.
- Web Services are platform independent.
- Web Services are a group of web methods.
- The application which provides web services can be called Web Service Client.
- At a time only one method would be used from the group of method available.

Growing use of Web Services :

- Code Sharing is one of the most important features of any programming language.
- Code Sharing can be done among the system only. If change in hardware, operating system and programming language it does not affect.
- Web Services are platform independent.
- Because of this feature Web Services are becoming popular.

Growing use of Web Services :

- For example,
  - If we have return some code in C# now we want to share that code in J#. Basically It is not possible.
  - By placing the code under Web Service, we can use this code with J# easily because web services are platform independent.

Protocols used in Web Services :

- We already know that Web Services are platform independent.
- Protocols are used which works behind each web service.
- Various protocols that works behind Web Services can be listed as.
  - HTTP (**H**yper **T**ext **T**ransfer **P**rotocol)
  - XML (**E**xtensible **M**arkup **L**anguage)
  - SOAP (**S**imple **O**bject **A**ccess **P**rotocol)
  - WSDL(Web Services Description Language)
  - UDDI (Universal Description Discovery and Integration)

Protocols used in Web Services :

- XML (Extensible Markup Language) :
- The basic thing that works behind any web service is XML and HTTP.
- HTTP is a standard protocol which is used for sharing information over the internet.
- XML is a standard language which is used for sharing information among different platforms as it is platform independent.

Protocols used in Web Services :

- Web method that we create are same like normal methods which have arguments and which also returns values.
- These arguments and return values are given in form of XML which is platform independent.
- As the data is shared via XML thus it makes the data also platform independent.
- XML provides standard data representation format which can be understood by any platform.

Protocols used in Web Services :

SOAP (Simple Object Access Protocol) :

- XML serves as the base for any web service.
- Web Service parameters and returns values which are written in XML. These values needed to be passed trough internet. For this SOAP is used.
- SOAP is XML based protocol which allows different applications to share information over HTTP.
- Using SOAP the XML data will be transferred from one location to another.

Protocols used in Web Services :

- SOAP is standard communication protocol that is used for interchanging information in distributed environment in structured format.
- Whenever a client application makes a request for a Web method, SOAP packet will be created.
- This packet contains the name of the Web method to be accessed and invoked.
- It also contains the parameters passed to web method in an XML format.

Protocols used in Web Services :

- When the SOAP packet comes at a Web Server side, then the Web Method name and its parameters are extracted from the packet.
- Then the method is invoked, executed and then return value if available is sent back to client using SOAP packet.

WSDL (Web Services Description Language)

- After defining the Web Services, the Web Services should be defined in standard format so that it can be accessed by any one.
- The developer who is using Web Service must know the parameters, return values and body of the Web Service.
- This information should be available in standard format so that it can be shared among different platforms.
- This is achieved by using XML based description language called WSDL.

WSDL (Web Services Description Language)

- WSDL is a language that is used to describe Web Service and its web methods.
- It is a markup language which describes the Web Services and its web methods.

- WSDL contains following information related to Web Services :
  - Information related to all the Web Services which are specified or created under particular web site.
  - Information related to the purpose of the Web Services.
  - Information related to the types of parameters and return values for each web method under Web Service.

- Information related to format in which each web service method can be accessed.
- Information related to URL to which a web service can be accessed.
- We can say that WSDL is used to describe and locate any Web Service.

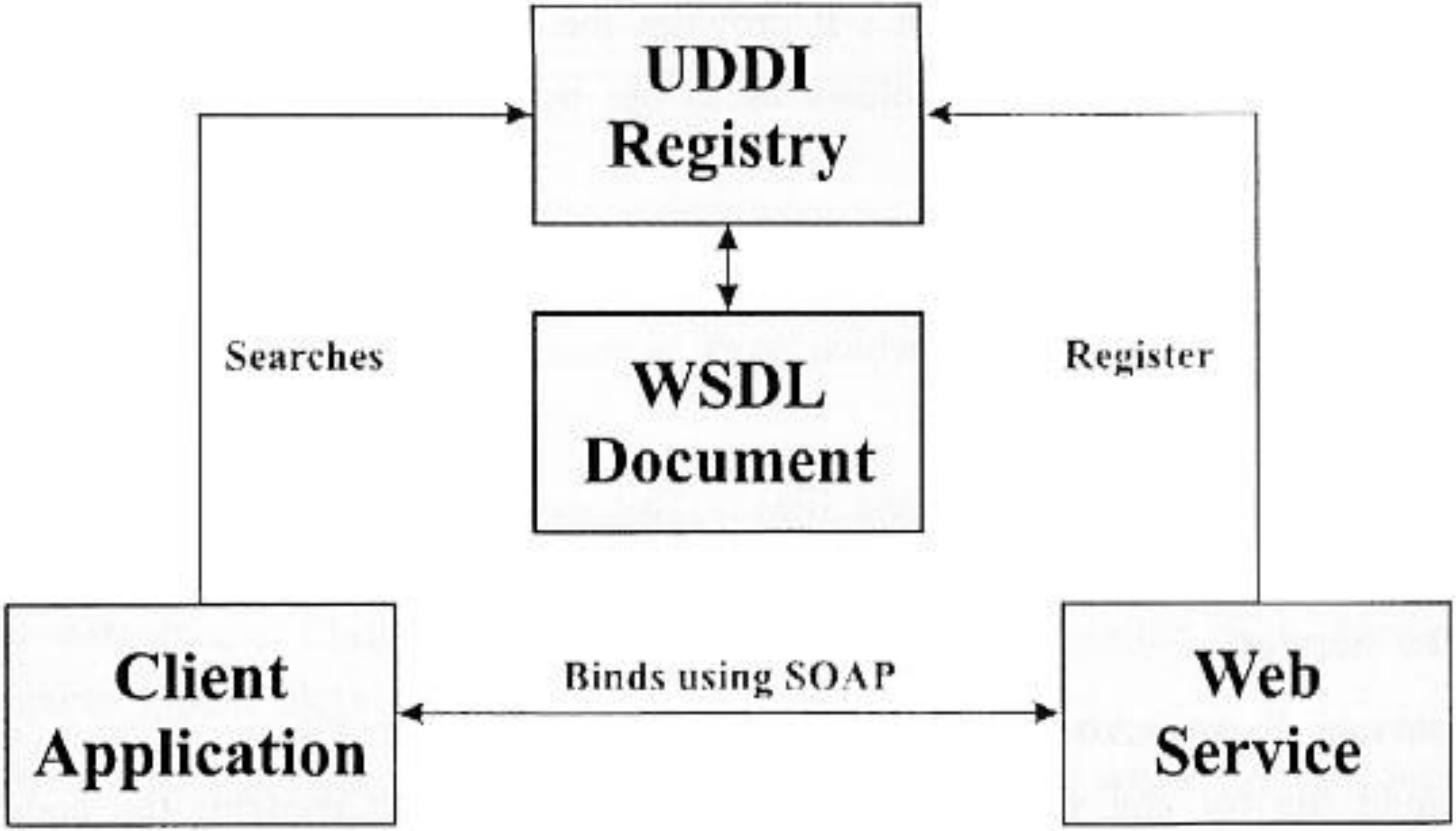
UDDI (Universal Description Discovery and Integration)

- UDDI is a directory which is used by different companies for registering and searches any Web Services available.
- It provides a standard mechanism which is used for registering and discovering a Web Service.
- In simple words, a UDDI is a place for storing information related Web Services.
- Whenever a Client needs to use a Web Service, first need to find the web service and information related to Web Service.

UDDI (Universal Description Discovery and Integration)

- The information related to Web Service is available in WSDL.
- Whenever we register Web Services under UDDI, it is registered and the URL of web service would be stored under UDDI, which can be accessed by any Client.
- UDDI uses SOAP for communication.

UDDI


UDDI Registry
Searches
WSDL Document
Register
Client Application
Binds using SOAP
Web Service

Working of Web Services :

- When any Web Service is created it is first registered in UDDI.
- UDDI is the registry which stores all the web services.
- UDDI gives the location of Web Services in WSDL document format.
- WSDL is document which describes web services.
- Location of WSDL is given by using UDDI.

Working of Web Services :

- WSDL is the document which has information about the Web Service and all its web methods.
- After getting details about the Web service, the Client application is bounded with the Web Service using SOAP.
- SOAP is communication protocol that is used to access Web service and its web methods.

Namespace used for Web Services :

- Web Services uses two namespace for creating and using Web Services.
- These namespace can be given as :

System.Web.Services :

  - It is used for creating a web service.

- ❑ The class which we create should inherit this WebService class.
- ❑ System.Web.Services is a namespace which has WebService class which allows us to inherit the WebService class in our class.

Namespace used for Web Services :

- System.Web.Services.Protocol :
  - It provides the facility to use the various protocols like SOAP.
- This namespace allows us to use protocols in order to use web service defined.

Web Service Files :

- There are two types of files which work accessing and creating Web Services. They are :

  .asmx file (Web Service Class file) :

  - This is the class file which contains Web Service.
  - Web Service Classes are stored in **.asmx** file.
  - This file is the entry to any Web Service.

Web Service Files :

- **.asmx.cs** / **.asmx.vb** file (Web Service Code Behind File) :
  - This is the code behind file for any web service and its web methods.
  - It contains the code for various web method created in web service.

Def. Create Web Service…

- We can easily create new web service using following steps.
  - Create a New **WEB Service** using
    - File – New – Web Site…
      - ASP.NET Web Service
  - After creation of web service it will automatically create two file called **Service.asmx** and **Service.as**
  - **Service.as** would be created inside the App_Code folder.
- Now remove the default method HelloWord from the Service as page.

Create Web Service…

- Now run the application
  - It will display link of created web method.
  - When we click the web method it will execute the code in its own environment and provide proper output.
  - If web service is tested then we can create its DLL.
- How to create a DLL(Dynamic Linked Library)
  - To use created web service in future we can create its DLL.
  - To create a DLL just right click on project and select the Publish WebSite option and then give proper path for DLL file.

Create Web Service…

- After creating the DLL file we can see that DLL in the BIN folder with
  - **FileName**.compiled
  - **FileName**.dll
- This file we have to used whenever we want to use a web service.
- We have to add reference for this file in the Web Application in which we are using.
- The reference for the file can be added by using Add Reference option which will appear on the Right Click of the web application.

Create Web Service…

- After adding Web Service DLL in current new web application use this web service as your requirement.

Asp.Net

05. Asp.NET Application Configuration and Deployment of Application

Web Application Configuration in ASP.Net

- Introduction :
  - ASP.NET provides a security model which makes easy to protect our web application.
  - Different configuration files provided by ASP.NET help us to make the web application more secure.
  - All the files which are used for storing configuration information are based on XML.

ASP.NET Configuration :

- ASP.NET configuration is done using XML file format.
- As XML is a platform independent so it could be applied to any machine and any platform.
- ASP.NET configurations are created as soon as we create our web application.
- We can also add configuration even after the creation of web application.

ASP.NET Configuration :

- Some of the benefits of XML based configuration files are as follows :
  - XML is a platform independent so it can be stored and retrieved in any platform.
  - When any of configuration settings are changed we do not need to restart Web Server again and again. ASP.NET configures changes automatically and applies it to Web Application.
  - Configuration files are in XML format so it can be opened with any simple text editor. We do not need special tools for configuring Web file.

ASP.NET Configuration :

- Configuration can be done at different levels. We can do configuration as follows :
  - Configure IIS to configure all Web applications under a machine.
  - Configure ASP.NET web application and all its web page under it. Configure individual web page of ASP.NET application.
  - If we have installed IIS. It provides MMC (Microsoft Management Console) interface to manage Web Sites that are stored under IIS. It also known as ISM (Internet Service manager)

Configuration Files :

- Configuration Files are used to configure websites.
- Configuration can be done at any time i.e.
  - Before execute the website,
  - During website execution or
  - Even after website is deployed on the server.
- There are two types of Configuration files which are used to configure either all web sites in a machine or individual web applications. They are
  - Machine.Config
  - Web.Config

Configuration Files :

- Machine.Config :
  - As the name suggest this file is used to configure all web application stored in our machine.
  - This is global configuration file for all web application stored in particular machine.
  - Every machines having ASP.NET installed, It has one Machine.Config file.
  - Machine.Config file is located in :

C:\Windows\Microsoft.Net\Framework\version\ConfigDirectory

  - The Machine.Config file provides global settings for all sites running on the machine.

Configuration Files :

- Web.Config :
  - Each and every web application has its own Web.Config file which can configure entire web application or a particular web page of the web application.
  - The Web.Config file is generated from Machine.Config file initially. Initial Web.Config file is stored at location :

C:\WINDOWS\Microsoft.NET\Framework\version\CONFIG directory

  - Each web application has its own Web.Config file.
  - If there are multiple folders in a web application, then each folder has its own set of Web.Config file.

Common Configuration Settings :

- Some of the common settings that can be applied to a website can be given as :
  - Tracing WebSite
  - Customizing Errors
  - Authentication and Authorization
  - Enabling Role Manager
  - Session Configuration
  - Trust Levels
  - Web Service Configuration
  - Caching

Common Configuration Settings :

- Tracing Website :
  - Tracing a web site allows us to check for any type of bugs that our application may have.
  - With tracing, we can maintain a log file of the activities done which provides us additional information which can be used for many purposes.
  - The log files can be used to find-out problems that may occur during web application usage.

Common Configuration Settings :

- Customizing Errors :
  - When we access different Web sites, different errors may occur.
  - We might not be aware with all errors which we are getting. Some unexpected screens may come which may make web site users uncomfortable.
  - Many websites may generate own error page. We can customize the display of the error page which we are getting.
  - It would make the user comfortable and would redirect the user to specific error page which is designed by developer.

Common Configuration Settings :

- Authentication and Authorization :
  - These are most used settings for any type of web site.
  - Authenticated users are requirement of the most web sites.
  - To allow authenticated users only, we must Authorize the users in such a way that the unauthorized users who visit our web sites should not be able to harm our information.
  - This can be done by combining Authentication and Authorization.

Common Configuration Settings :

- Enabling Role Manager :
  - Assignment of role is important part of Web Site management.
  - It is used along with the features of Authentication and Authorization.
  - This feature enables us to define special features for all the users.
  - For example, if want the administrator to access a web page, then we can use this feature to generate a role which would provide administrator with the right to use specific web page.

Common Configuration Settings :

- Session Configuration :
  - We can configure the session related settings under this configuration.
  - Under this setting, we can specify the settings like timeout period, session mode, cookie based session or cookieless session, SqlCommandTimeout, Network Timeout etc.
- Trust Levels :
  - There are different users available. All the users do not have all rights.
  - Different types of users may have different trust levels.

Common Configuration Settings :

- Trust Levels :
  - For example,
    - Administrator can access all the pages of the web sites while normal user cannot do so different users may have different trust levels which are defined by using such setting.
- Web Service Configuration :
  - This setting are related to web services.
  - We can configure web services by using this setting.

Common Configuration Settings :

- Caching :
  - Caching is somewhat a new concept, which is introduced in ASP.NET.
  - Caching means storing the data in cache memory.
  - This stored data in memory can be accessed in faster way as the data is stored in memory
  - By caching we can store the output or data in cache memory depending on the requirement of user.

Tracing :

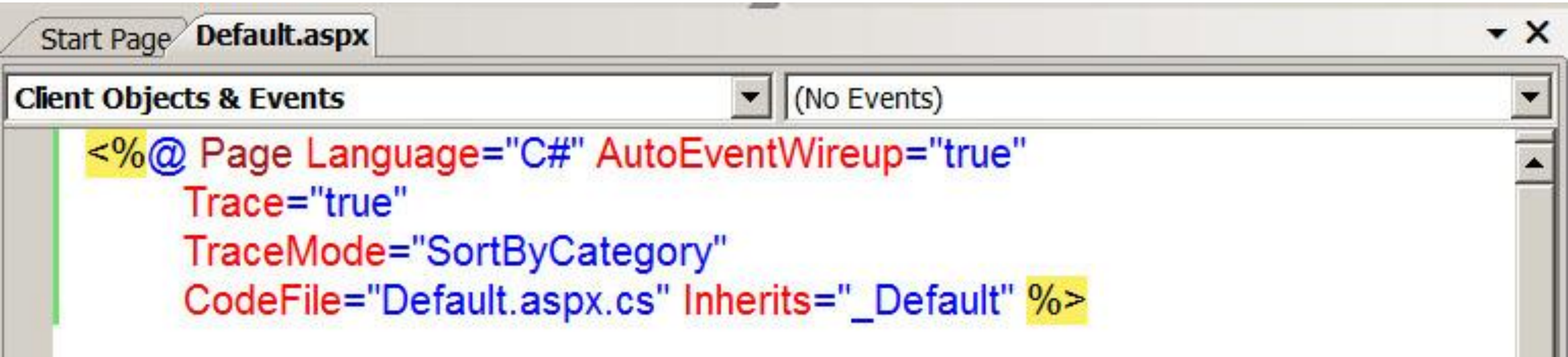
- Trace means monitor something.
- Tracing is a way of monitoring how our web applications execute.
- Whenever we execute web site or web page within web sites, if any problems comes then it is difficult to find the actual source of the problem.
- For finding the actual problem, you need to trace the web site or web page within web sites.
- With tracing we can record some of the execution of program and program flow so that it does not affect the output of the web page.

Tracing :

- Tracing is way of testing web site in order to get perfect output as per user’s requirement which fulfils user’s need.
- Tracing is not enabled automatically, we have to enable it.
- Tracing can be done at two levels:
  - Page Level Tracing :
    - As the name suggest, Page Level Tracing, allows us to trace web page that we want to trace.
    - It is used when we want to trace particular web page.

Page Level Tracing :

- Page Level Tracing is done when we want to trace selected web pages from web sites. It allows us to trace a particular web page.
- We can enable or disable page level tracing using the trace attribute in page directive.
- It can be enabled by writing the following code in the page Directive:



```
<%@ Page Language="C#" AutoEventWireup="true"
    Trace="true"
    TraceMode="SortByCategory"
    CodeFile="Default.aspx.cs" Inherits="_Default" %>
```

Page Level Tracing :

- There are two attributes available in the page directive for enabling tracing at page level.
- Trace = “True”
  - This property enables tracing at page level. By default this property is disabled for all web pages.
  - We can enable this property by using trace property of the Page Directive.

Page Level Tracing :

- TraceMode=“SortByCategory” || “SortByTime”
  - This property also enables tracing at page level.
  - We can sort the details of the page while tracing either by category or by time.
  - We can write the trace information by using Trace.Write() method or by using Trace.Warn() method.

Application Level Tracing :

- Application level tracing is done when we wan to trace all the web pages of Web application.
- After we enable the application level tracing we do not need to enable individual page level tracing.
- We can enable Application Level Tracing from Web.Config file.
- Tracing information is specified in <configuration> tag inside <system.web> tag of Web.Config file.

Application Level Tracing :

- Web.Config can be specified as follows:

```
<configuration>
  <system.web>
    <trace enable=“true” requestLimit=“20”
           localOnly=“false”
           traceMode=“SortByCategory”
           pageOutput=“true”/>
  </system.web>
</configuration>
```

Application Level Tracing :

- When we enable application level tracing, ASP.NET provides information of the all request which occurs within the web application.
- We need to specify the request limit which will collect the number of request that will be displayed on each page.
- This can be done by using the requestLimit property of the trace tag.

Application Level Tracing :

- By default, when we enable application level tracing, it would be visible only on the local web server which we are using.
- When we are working on remote based web server, if we want to enable application level tracing we need to set the LocalOnly property as false.
- It means that when we are working on remote based client, application level information would be visible on client machine.

Application Level Tracing :

- We can get the information on the screen by enabling the pageOutput property to ture.
- By default this property is set as false so we are not able to view the trace information on the screen.
- For viewing the trace information we need to go to the trace viewer and then write the Trace.axd on the current URL of browser.