Unit 2 : State Management

What is State?

State is an innovative concept in web development. By using states in your web applications, you can save the state of a web application whenever the application makes a round trip to the server.

"The State of a web application stores the changes that are made to the web application."

Why it is required in ASP.NET?

State is an important concept to save the changes made in a web application both at the server side or client side.

State management is highly useful in e-commerce sites. For example, in e-commerce shopping sites it is necessary to store the state for web application. In an e-commerce site, the company have to keep record of the user request i.e. the items selected by the user on different pages of the shopping site.

Types of States

The following are the types of states of a web application in ASP.NET :

1) The Application State
2) The Session State
3) The View State

Client-side State Management

The state of a web application can be saved at the client. If client manages the state of the web application, it is called client-side state management.

There are 3 ways to manage states at client side :

1) HiddenField Control
2) QueryString
3) Cookies

Server-side State Management

The state of a web application can be saved at the server. If server manages the state of the web application, it is called server-side state management.

There are 3 ways to manage states at server side :

1) The Application State Object
2) The Session State Object
3) The Database

The Application State (application specific)

The Application State is used to store the data of the global variables of an ASP.NET web application.

The data in the application state is stored once and read many times.

The application state uses HTTPApplicationState class to store the data throughout the application. You can access the information stored in the application state by using properties of the class.

The data stored in the application state can be access in all the pages of an ASP.NET web application and the data is same for all the users using the web application.

Example :

This example counts the number of times a webpage is loaded using Application state. The count is displayed in a label called Label1.

```
using System;
using System.Web.UI.WebControls;

public partial class Default6 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        int count = 0;
        if (Application["one"] != null)
        {
            count = (int)Application["one"];
        }
        count++;
        Application["one"] = count;
        Label1.Text = "No of times the page is loaded = " + count.ToString();
    }
}
```

Output :

No of times the page is loaded = 3

The Session State (user specific)

The user session means the duration for which a user uses a web site.

Each client accesses the web application maintain a different session with the web server.

There is particular information associated with each of these sessions.

Session state is used to store this information.

The session state is defined in the <SessionState> section of the web.config file and stores the **data specific to a user session** in session variable.

Different sessions are created for each user session.

Also the session variable can be accessed from any page of a web application.

When a user accesses a web page, a **SessionID** is created for the user.

The SessionID is transferred between server and the client over the HTTP protocol using cookies.

Example :

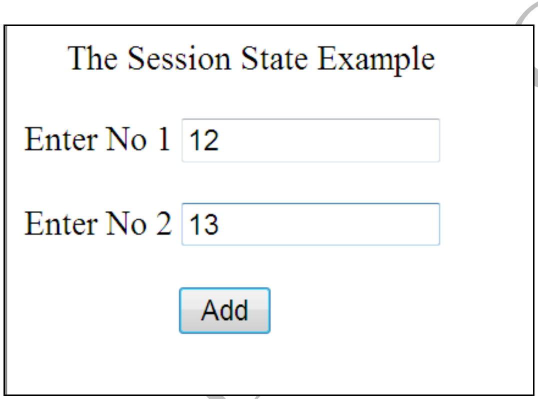
The following example shows the use of Session State.

Default8.aspx

```
using System;
using System.Web.UI.WebControls;

public partial class Default8 : System.Web.UI.Page
{

    protected void Button1_Click(object sender, EventArgs e)
    {
        int ans = Int32.Parse(TextBox1.Text) + Int32.Parse(TextBox2.Text);
        Session["two"] = ans.ToString();
        Response.Redirect("Default9.aspx");
    }
}
```



The Session State Example

Enter No 1 12

Enter No 2 13

Default9.aspx

```
using System;
using System.Web.UI.WebControls;

public partial class Default9 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label2.Text = (string)Session["two"];
    }
}
```

Output : Addition=25

The View State (page specific)

The View State stores the page specific information, when a page is posted back to the server.

When a page is processed, the current state of a web page and controls is stored into a string and saved as a Hiddenfield. Such a state of a web page is called View State.

A view state is defined as a hiddenfield on a webpage.

The "ViewState" property is used to save the view state for each control.

If the “ViewState” property is not used then the value written in different controls such as TextBox are not retained when the web page is reloaded.

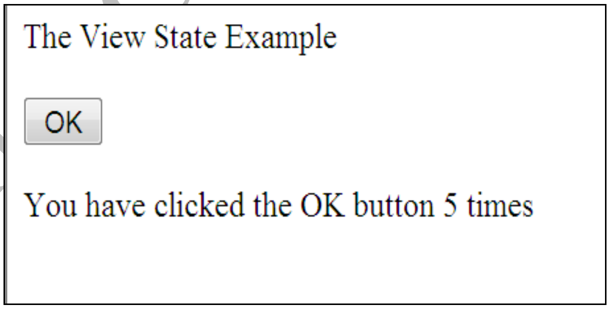
Example :
This example counts and displays the number of times a button is clicked using ViewState.

```
using System;
using System.Web.UI.WebControls;

public partial class Default7 : System.Web.UI.Page
{

    protected void Button1_Click(object sender, EventArgs e)
    {
        int count;
        if (ViewState["one"] == null)
        {
            count = 1;
        }
        else
        {
            count = (int)ViewState["one"] + 1;
        }
        ViewState["one"] = count;
        Label1.Text = "You have clicked the OK button " + count.ToString() + " times";
    }
}
```

Output :


The View State Example

You have clicked the OK button 5 times

Cookies

“A Cookie is a small text file that the browser creates and stores on the hand disk of your machine.”
A cookie is a small amount of data that is stored on the client file system or in the memory of client browser session. Cookies are useful when a user need small amount of data that is persisted for a variable period of time. It can be stored for a browser session or as long as months or even years. Cookies can be temporary (with specific expiration

time and date) or persistent. Cookies are useful for storing small amount of frequently changed information on the client. The cookies contain information that the web application can read whenever the user visits the site. Cookies are associated with a website and not with a particular page. So the browser and server will exchange cookie information. Usually a cookie can have a maximum size of 4KB.

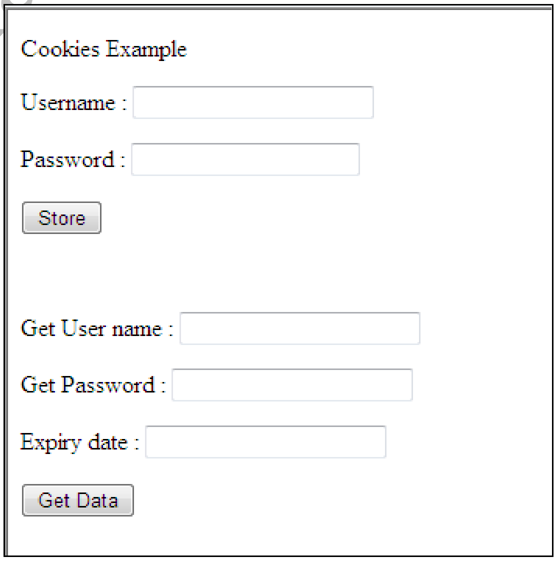
Each cookie must have a unique name. You can also set cookies date and time of expiration. If you do not set the cookie's expiration, the cookie is created but not stored on the user's hard disk.

For example, a site conducting a poll might use a cookie simply as a Boolean value to indicate whether a user's browser has already participated in voting so that the user cannot vote twice.

Example :

The following example shows how to store and retrieve data from cookies.

```
using System;
using System.Web.UI.WebControls;
public partial class _Default : System.Web.UI.Page
{
    protected void Button1_Click(object sender, EventArgs e)
    {
        HttpCookie c = new HttpCookie("Visitor");
        c.Values["uname"] = TextBox1.Text;
        c.Values["pwd"] = TextBox2.Text;
        c.Values["expdt"] = DateTime.Now.ToShortDateString();
        c.Expires = DateTime.Now.AddDays(1);
        Response.Cookies.Add(c);
    }
    protected void Button2_Click(object sender, EventArgs e)
    {
        TextBox3.Text = Request.Cookies["Visitor"].Values["uname"].ToString();
        TextBox4.Text = Request.Cookies["Visitor"].Values["pwd"].ToString();
        TextBox5.Text = Request.Cookies["Visitor"].Values["expdt"].ToString();
    }
}
```



QueryString

"A QueryString is the information that is added at the end of a page's URL".

QueryStrings are used to send information from one page to another page. They are passed along with URL in clear text.

A typical URL with a querystring looks like –

http://www.abc.com/subjects.aspx**?id=111&name=aspnet**

In the URL path given above, the querystring starts with a question mark (?) and have two field/values pairs, one called "id" and other called "name".

We can only pass small amount of data using querystrings.

For example, querystring is an easy way to pass information like product number from one webpage to another.

The maximum size of a querystring is 255 bytes.

Since querystrings are sent in clear text format, we can also encrypt query values.

Advantages :

1) Simple to implement.
2) No server resources required.
3) Wide spread support.

Disadvantages :

1) Limited URL length.
2) Easily modified by the user.
3) It is highly risky according to security point of view because it is human readable.

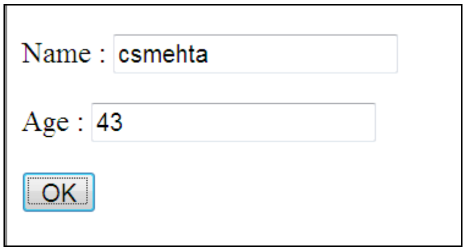
Example :

The following example shows how to pass multiple values to another page using QueryString.

Default4.aspx

```
using System;
using System.Web.UI.WebControls;

public partial class Default4 : System.Web.UI.Page
{
    protected void Button1_Click(object sender, EventArgs e)
    {
        Response.Redirect("Default5.aspx?Name=" + TextBox1.Text + "&Age=" +
TextBox2.Text);
    }
}
```



Name : csmehta

Age : 43

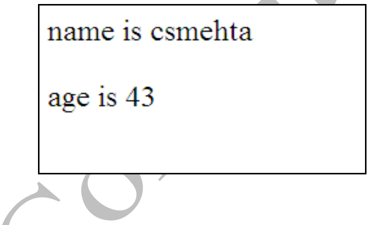
Default5.aspx

```
using System;
using System.Web.UI.WebControls;

public partial class Default5 : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {
        Label1.Text = "name is " + Request.QueryString["Name"];
        Label2.Text = "age is " + Request.QueryString["Age"];

    }
}
```

Output :



name is csmehta

age is 43