Passing Data from One Page to another One (ASP.NET)

Anubhav Tiwari

*R&D Dept., Syscom Corporation Ltd.

Abstract- This paper talks about
• What are different ways to pass data from one page to another one in ASP.NET?
• How the code is written in each of the ways to pass data?
• Which way is suitable for which scenarios?

Index Terms- Passing data from one page to another, ASP.NET, their suitability, How to write code for them;

I. INTRODUCTION

Dear Readers,

First of all, I would say this paper is just for ASP.net developers and I would like to start with a question. “How many times you faced the problem where you were unable to find which way to adopt to pass the data from one page to another one in asp.net?” Obviously, you would have faced the problem many-a-times and you randomly picks the way whichever strikes in your mind. But before picking a way we must decided and well analyzed that is the way chosen by me is appropriate or not , choosing the incorrect way of passing data may compromise our application performance.

Now in this paper I have consolidated all the ways of passing data and their suitability in the scenarios.
Ways of passing data:

There are so many ways to pass the data from one page to another one which are as follows:

1. Passing data within application:

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Ways of Passing data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Using Session State</td>
</tr>
<tr>
<td>2</td>
<td>Using Public properties of page</td>
</tr>
<tr>
<td>3</td>
<td>Getting control information</td>
</tr>
<tr>
<td>4</td>
<td>Using Cookies</td>
</tr>
<tr>
<td>5</td>
<td>Using Cache</td>
</tr>
<tr>
<td>6</td>
<td>Using Database</td>
</tr>
<tr>
<td>7</td>
<td>Using HTTP context</td>
</tr>
<tr>
<td>8</td>
<td>Using Global Application Variables</td>
</tr>
</tbody>
</table>

2. Passing data across application:

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Ways of Passing data</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Using Web services / WCF services</td>
</tr>
<tr>
<td>2</td>
<td>Using Database</td>
</tr>
<tr>
<td>3</td>
<td>Using Query String</td>
</tr>
<tr>
<td>4</td>
<td>Getting Post information</td>
</tr>
</tbody>
</table>

Passing data within application:

I. Using Session State

Suppose I have an application which consists of two pages **Source.aspx** and **Target.aspx**.

Now I have following data in Source:
- Name
- Age

Now I need to pass this data from Source to Target.
Session is one of other best approach to pass data to another page. Basically the value stored in the Session can be retrieved in any of page for the time span user is logged in that site. Basically this approach is used to identify the user who logged in to the system.

Code Example:

```csharp
protected void btnSubmit_Click(object sender, EventArgs e)
{
    AddDataToSession();
    Response.Redirect("Target.aspx");
}

private void AddDataToSession()
{
    Session.Add("name", txtName.Text);
    Session.Add("age", txtAge.Text);
}
```
II. Using Public properties of page

For accessing the value of the property of the page is another good approach but this approach is applicable only when it is defined, in prior, the page from where it is being redirected. In other words we can say that there is predefined previous page. Also make sure to use Server.Transfer instead Response.Redirect because Response.Redirect treats each page as a separate transaction and does not persist data but Server.Transfer solves this purpose reducing round trips to the page and persisting the data of previous page.

Code Example:

```csharp
public string Name
{
    get { return txtName.Text; }
}

public int Age
{
    get { return int.Parse(txtAge.Text); }
}

protected void btnSubmit_Click(object sender, EventArgs e)
{
    Server.Transfer("Target.aspx");
}
```
III. Getting control information

For accessing the value of the controls of the source page is another good approach but this approach is applicable only when it is defined, in prior, the page from where it is being redirected. In other words we can say that there is predefined previous page.

Also make sure to use Server.Transfer instead Response.Redirect because Response.Redirect treats each page as a separate transaction and does not persist data but Server.Transfer solves this purpose reducing round trips to the page and persisting the data of previous page.

Code Example:

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    GetData();
}

private void GetData()
{
    lblName.Text = PreviousPage.Name;
    lblAge.Text = PreviousPage.Age.ToString();
}

protected void btnSubmit_Click(object sender, EventArgs e)
{
    Server.Transfer("Target.aspx");
}
```
IV. Using Cache

Just like Session, Cache data are stored at server side. There is no guarantee that cached object will be available at any point of time. Cached objects can be dropped automatically depending on what kind of memory situation is there. Whenever you plan to use cached objects, make sure you check for nulls before using it. Here topic cache needs some more discussion but I’m not going in depth which might make the blog unnecessarily un-interesting to reader. So I included only basic example to use the cache.

Code Example:

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    GetData();
}

private void GetData()
{
    lblName.Text = (PreviousPage.FindControl("txtName") as TextBox).Text;
    lblAge.Text = (PreviousPage.FindControl("txtAge") as TextBox).Text;
}

protected void btnSubmit_Click(object sender, EventArgs e)
{
    CreateCache();
    Response.Redirect("Target.aspx");
}

private void CreateCache()
{
    Cache["Name"] = txtName.Text;
    Cache["Age"] = txtAge.Text;
}
```
This the most widely used approach, where we store the data into our database and then retrieve it later on the page wherever it is needed as per our requirement.

**Code Example:**

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    GetData();
}
private void GetData()
{
    lblName.Text = Cache["Name"].ToString();
    lblAge.Text = Cache["Age"].ToString();
}
```

Here In this example I have created one table with two fields Name & Age. I used them for storing the data as shown in the figure above. Then I retrieved these values from the database tables in other page.

Dear readers, to keep the paper concise and precise, we have purposely excluded the code snippet for CRUD (Create, Read, Update, Delete) operation of database.
VI. Using HTTP context

Using HTTP Context is another good approach but this approach is applicable only when a separate round trip to the page is not made, in other words, the page is used for processing the information rather than navigating to another page. So make sure to use Server.Transfer instead of Response.Redirect because Response.Redirect treats each page as a separate transaction and does not persist data but Server.Transfer solves this purpose reducing round trips to the page and persisting the data of previous page.

Code Example:

```csharp
protected void btnSubmit_Click(object sender, EventArgs e)
{
    AddDataToContext();
    Server.Transfer("Target.aspx");
}

private void AddDataToContext()
{
    HttpContext.Current.Items.Add("Name", txtName.Text);
    HttpContext.Current.Items.Add("Age", txtAge.Text);
}
```

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    GetData();
}

private void GetData()
{
    lblName.Text = HttpContext.Current.Items["Name"].ToString();
    lblAge.Text = HttpContext.Current.Items["Age"].ToString();
}
```
VII. Using Global Application Variables

One of the major advantage of using application variable is that the value stored in application variable persist till the down of web server (in session the value persist till the browser closed.)

Code Example:

```csharp
protected void btnSubmit_Click(object sender, EventArgs e)
{
    AddDataToApplication();
    Response.Redirect("Target.aspx");
}

private void AddDataToApplication()
{
    Application["Name"] = txtName.Text;
    Application["Age"] = txtAge.Text;
}

protected void Page_Load(object sender, EventArgs e)
{
    GetData();
}

private void GetData()
{
    lblName.Text = Application["Name"].ToString();
    lblAge.Text = Application["Age"].ToString();
}
```
Passing data across application:

VIII. Using Web Services / WCF Services

Using web service / WCF Service is the most widely used approach for passing data from one page to another in cross application environment. Connectivity between applications is very important. Connectivity in any case is very important for that matter but it specially is very important between applications. Connecting web application used to be a big challenge before the advent of technologies like SOAP (Simple Object Access Protocol). The reason it was so difficult was, there were so many technologies people were working in. The applications were hosted on different types of servers, etc. But these things should not be a barrier to facilitate the communication between applications. The only requirement was to have some standards to follow and standard ways of doing things so that the applications become capable of communicating with other applications irrespective of the technologies used.

IX. Using Database

As this approach is already discussed earlier in the document. Please refer the previous section for the same.

X. Using Query String

Passing data from one page to another one through Query String is one of the good ways when the data is not sensitive and is not too large. The default maximum length of a Query String is 2048 chars. Here is the basic example of passing data through Query String.

Code Example:

Using Query String we can pass this data in the following way:

```csharp
protected void btnSubmit_Click(object sender, EventArgs e)
{
    Response.Redirect("http://Application2/Target.aspx?name=\"+txtName.Text+\"&age=\"+txtAge.Text);
}
```
XI. Getting Post information

We can also get the values using `Request.Form` but the thing that we have to keep in our mind is that the values from the source page should be posted using form element attribute method = “Post”. In other words the source page should use HTTP POST action for redirection.

```csharp
protected void Page_Load(object sender, EventArgs e)
{
    GetQueryString();
}

private void GetQueryString()
{
    lblName.Text = Request.QueryString["name"].ToString();
    lblAge.Text = Request.QueryString["age"].ToString();
}
```

Add below attributes in the page directive:

```
```

Code Example:
### XII. CONCLUSION

Thus we can conclude that there are so many ways to pass the data from one page to another one but we must know their suitability as per need. Using the correct way in correct situation would make our application more performing and reliable. Code of all these ways are very simple and described in the paper.

### ACKNOWLEDGMENT

This work was supported by Syscom Corporation Ltd.

### REFERENCES


### AUTHORS

**First Author** – Anubhav Tiwari, M.C.A, Software Engineer, anubhavtiwari87@gmail.com