State Management

R.J. Glotzbach

State Management

- Storing information between client requests
- HTTP is effectively stateless
- In the past – cookies have been used
  - ASP 3.0 Session required the use of cookie technology to track users
  - Cookie technology generally not supported on wireless devices

ASP.NET States

- Session state
- Hidden variables
- ViewState
- Application state

Session state

- Allows you to maintain the variables and objects for a client over the course of multiple requests and responses.

Hidden variables

- Allows you to persist objects between server round-trips by posting the data to the client as hidden fields.

ViewState

- Allows you to maintain the values of a mobile web form page on the server.
- The runtime stores this information in an instance of the System.Web.UI.StateBag class, which is itself stored in the Session.
- The server sends some information to the client.

Application state

- Allows you to maintain the variables and objects of an application over multiple requests by multiple clients.

States

- Session, hidden variables, and ViewState require that the server be able to identify the client in order to track multiple requests/responses.

- Some unique token must be passed to the client with each response, which the client returns to the server with each request.

Munged URLs

- ASP.NET uses ***munged URLs***.
- Munged URLs are URLs that the runtime modifies to contain a unique session ID.

http://128.210.136.249/rjglotzbach/Lab10/(lp20nq55utgompmw3ezp4b45)/Lab10.aspx

Munged URLs

- When the web server receives the request, it parses the session ID from the munged URL.
- The runtime then uses the session ID the same way it would use a session ID obtained from a cookie.

Munged URLs

- The runtime does not automatically use munged URLs.
- Disabling cookies forces the use of munged URLs
- Disable cookies by setting the cookieless attribute of the sessionState element within the web.config file.

sessionState

- Fragment from a web.config file:

```
<sessionState
  mode=“inProc”
  stateConnectionString=“tcpip=127.0.0.1:42424”
  cookieless=“true”
  timeout=“20”
/>
```

Disadvantages of munged URLs?

- Some browsers can experience difficulties dealing with relative URLs after being redirected to a munged URL.

- The MobilePage class and DeviceAdapter class include helper methods that allow you to convert URLs to rooted URLs.

Disadvantages of munged URLs?

- ASP supports cookies but not munged URLs.
- If backwards-compatibility is an issue, do not use munged URLs.
- Thirdly, many wireless browsers support URL lengths much shorter than those supported by desktop browsers.

Hidden Variables

- In HTML, similar to:
  - Input tags with a hidden type

- In WML, similar to:
  - Setting variables in the browser’s cache, then posting all variables to the server when the user completes the form.

- In .NET
  - The MobilePage class provides a HiddenVariables property that stores name-value pairs.

Hidden Variables

- In CodeBehind somewhere:

```
private void Command1_Click(…)
{
   HiddenVariables.Add(TextBoxName.ID, TextBoxName.Text);
}
```

ViewState

- ASP.NET gives the user the impression that the runtime maintains the state of pages over several server round-trips.
- The pages do not really exist over multiple requests.
- Instead, the runtime saves the properties of the page and each server control’s ViewState to an instance of the StateBag class.
- For each request, the runtime automatically reconstructs the page using the property values persisted in the StateBag instance.

ViewState

- The ViewState property has the scope of the current MobilePage object (which means the current aspx and any CodeBehind module).

```
ViewState[“DLValue”] = value;
```

ViewState

- Disabling ViewState for a single control

```
<mobile:Label
  id=“label1”
  runat=“server”
  EnableViewState=“False” />
```

ViewState

- Disabling ViewState for an entire page

```
<%@ Page language=“C#”
  CodeBehind=“test.aspx.cs”
  Inherits=“test”
  EnableViewState=“False” %>
```

Application State

- In ASP.NET, an application is the total of all files that the runtime can invoke or run within the scope of a virtual directory and its subdirectories.
- At times, you may want to initialize variables and objects that have application scope.
- Use System.Web.HttpApplicationState class

Application State

- Inside the global.asax.cs file

```
protected void Application_Start(…)
{
   String AppStartTime = DateTime.Now.ToLongTimeString();
   Application[“AppStartTime”] = AppStartTime;
}
```