

Desarrollador de Aplicaciones Web

Programación Web III



Departamento de Ingeniería e Investigaciones Tecnológicas

Pasaje de Datos

Ing. Mariano Juiz
Ing. Matias Paz Wasiuchnik
Ing. Pablo Nicolás Sanchez

Agenda

1. Introducción
2. Pasar Datos Controller => Vista
 1. ViewData
 2. ViewBag
 3. ViewModels
3. TempData
4. Variables de Sesión
5. Variables de Aplicación
6. Cookies

Introducción : HTTP Request y Response

Páginas web

Cliente

http://www.unlam.edu.ar

Internet DNS



IP= 200.47.130.101 Puerto: 80



HTTP Request

Servidor



www.unlam.edu.ar

IP = 200.47.130.101

inicio.html

```
<html>
```

```
<body>
```

.....

```
</body>
```

```
</html>
```

HTTP Response

ViewData

- ViewData is used to pass data from controller to view
- It is derived from ViewDataDictionary class
- It is available for the current request only
- Requires typecasting for complex data type and checks for null values to avoid error
- If redirection occurs, then its value becomes null

```
//Codigo Controller
public ActionResult Index()
{
    List<string> empleado = new List<string>();
    empleado.Add("Marita");
    empleado.Add("Pedro");
    empleado.Add("Juan");

    ViewData["Empleado"] = empleado;
    return View();
}
//codigo vista
<ul>
    @foreach (var empleado in ViewData["Empleado"] as List<string>)
    {
        <li>@empleado</li>
    }
</ul>
```

ViewData

- ViewBag is also used to pass data from the controller to the respective view
- ViewBag is a dynamic property that takes advantage of the new dynamic features in C# 4.0
- It is also available for the current request only
- If redirection occurs, then its value becomes null
- Doesn't require typecasting for complex data type

//Controller Code

```
public ActionResult Index()
{
    List<string> empleado = new List<string>();
    empleado.Add("Juana");
    empleado.Add("Pedro");
    empleado.Add("Joaquin");

    ViewBag.Empleado = empleado;
    return View();
}
```

//page code

```
<ul>
    @foreach (var empleado in ViewBag.Empleado)
    {
        <li>@empleado</li>
    }
</ul>
```

TempData

- TempData is derived from TempDataDictionary class
- TempData is used to pass data from the current request to the next request
- It keeps the information for the time of an HTTP Request. This means only from one page to another. It helps to maintain the data when we move from one controller to another controller or from one action to another action
- It requires typecasting for complex data type and checks for null values to avoid error. Generally, it is used to store only one time messages like the error messages and validation messages

```
//Controller Code
public ActionResult Index()
{
    List<string> Student = new List<string>();
    Student.Add("Jignesh");
    Student.Add("Tejas");
    Student.Add("Rakesh");

    TempData["Student"] = Student;
    return View();
}

//page code
<ul>
    <% foreach (var student in TempData["Student"] as
List<string>)
    { %>
        <li><%= student%> </li>
    <% } %>
</ul>
```

Variables de Sesión

- Son objetos que se inician cuando el usuario ingresa en la página y finalizan cuando deja el sitio o por timeout
- Son datos que solo visualiza el usuario en cuestión

En Global.asax (se inicializan los objetos)

```
protected void Session_Start(Object sender, EventArgs e)
{
    Session["usuarioID"] = String.Empty;
    Session["sexo"] = String.Empty;
}
```

En cualquier parte de la aplicación web (se setean los objetos)

```
Session["usuarioID"] = "Iquirola";
Session["sexo"] = "M";
```

Recomendaciones

- Encriptar datos antes de guardarlos

Variables de Sesión

Modos

- InProc
- State Server
- SqlServer
- Custom
- Off

Variables de Sesión

Modo InProc

- Es el modo por defecto (y el más óptimo)
- El estado de la sesión se almacena en la memoria del servidor web
- Ofrece el mejor rendimiento y buena seguridad
- No se persiste si se reinicia la aplicación web o a través varios servidores (Web Farm)

```
<sessionState mode="InProc" cookieless="false" timeout="20" />
```

- Cookieless define si almacena el "session Id" en el usuario o es ingresado en la querystring de la url

```
http://www.globons.com/(55mfgh55vgblurtywsityvjq)/Resultado.aspx
```

Recomendaciones

- No guardar muchos datos porque ocupan memoria de servidor
- Encriptar los datos antes de guardarlos

Variables de Sesión

StateServer

- El estado de la sesión se almacena en un servicio llamado ASP.NET State Service (aspnet_state.exe)
- Se envían datos por el protocolo HTTP sobre un puerto TCP
- Persiste aunque se reinicie la aplicación o a través de varios servidores (Web Farm)
- Ofrece menor rendimiento que el modo InProc, pero mayor fiabilidad y escalabilidad

```
<sessionState mode="StateServer" cookieless="false"  
stateConnectionString="tcpip=myserver:42424" timeout="20" />
```

Recomendaciones

- No detener el servicio (se pierden los datos de la sesión)
- Encriptar los datos antes de guardarlos

Variables de Sesión

SqlServer

- El estado de la sesión se almacena en una base de datos de SQL Server, brindando mayor estabilidad y escalabilidad
- Persiste aunque se reinicie la aplicación o a través de varios servidores (Web Farm)
- En las mismas condiciones de Hardware, ofrece menor rendimiento que **State Server** pero ofrece una mejor integridad de los datos y reporting.

```
<sessionState mode="SqlServer"
sqlConnectionString="data source=127.0.0.1;user
id=sa; password=" cookieless="false" timeout="20" />
```

Recomendaciones

- Crear la base de datos "ASPState" usando el script "InstallState.sql" (ubicado en la carpeta *WinDir\Microsoft.Net\Framework\Version*)
- Encriptar los datos antes de guardarlos

Variables de Sesión

Custom

- Permite especificar un proveedor de almacenamiento de la sesión customizado
 - Ej: <http://www.codeproject.com/KB/session/sessiontool.aspx>
- Es necesario implementarlo

```
<sessionState mode="StateServer" cookieless="false"  
stateConnectionString="tcpip=myserver:42424" timeout="20" />
```

Recomendaciones

- Encriptar los datos antes de guardarlos

Off

- Deshabilita el estado de la sesión.
- Si la aplicación web no usa sesión, se mejora el rendimiento.

Variables de Aplicación

- Son objetos que se inician con la Aplicación Web y persisten hasta detenerla
- Son datos que visualizan todos los usuarios de la aplicación web

En Global.asax (se inicializan los objetos)

```
protected void Application_Start(Object sender, EventArgs e)
{
    Application["localidad"] = "San Justo";
    Application["universidad"] = "UNLAM";
}
```

En cualquier parte de la aplicación web

```
Response.Write(Application["localidad"].ToString());
```

Variables de Aplicación

Recomendaciones

- Gestionar la concurrencia:
 - Application.Lock antes de actualizar
 - Application.Unlock después de actualizar
- ¡Cuidado con el rendimiento!
 - Los bloqueos pueden ralentizar
 - No se comparte entre distintos servidores
- Utilizar cuando son datos para todos los usuarios (provincias, localidades, etc.)

Cookies

- Se guarda información en el disco rígido del cliente
- Están asociadas a un sitio web y no a una página en particular
- Son útiles para mantener la continuidad en una aplicación Web

```
Response.Cookies["usuario"]["usuarioId"] = "jquirola";  
Response.Cookies["usuario"]["ultimaVisita"] = DateTime.Now.ToString();  
Response.Cookies["usuario"].Expires = DateTime.Now.AddDays(1);
```

```
lblUsuarioId.Text = Request.Cookies["usuario"]["usuarioId"];
```

Recomendaciones

- Encriptar los datos antes de guardarlos
- Almacenar datos no susceptibles y no invalidantes para la aplicación web
- No depender de los datos guardados como cookies porque el cliente los puede tener inhabilitados o pueden ser borrados
- Almacenar poca información (tamaño máximo de 4096 bytes, 20 cookies por sitio y/o 300 cookies en total)

Desarrollador de Aplicaciones Web

Programación Web III

2do Cuatrimestre (2015)



Departamento de Ingeniería e Investigaciones Tecnológicas

Muchas gracias

Ing. Mariano Juiz
Ing. Matias Paz Wasiuchnik
Ing. Pablo Nicolás Sanchez