Bab 13

State Management

Sebuah instan dari kelas halaman Web yang dibuat setiap kali halaman tersebut diposting ke server. Dalam pemrograman Web tradisional, ini biasanya berarti bahwa semua informasi yang terkait dengan halaman dan kontrol pada halaman tsb akan hilang dengan setiap kali terjadi perpindahan halaman web. Misalnya, jika pengguna memasukkan informasi ke dalam kotak teks, informasi isian dari pengguna akan hilang.

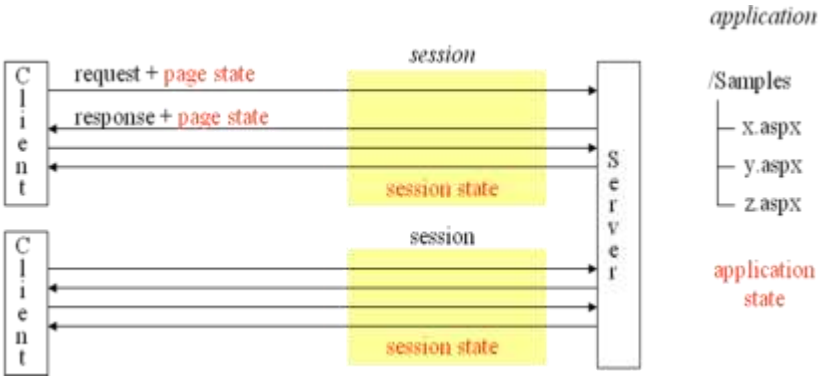
13.1. Tiga Jenis State

Secara umum pemrograman mengatasi *state management* dengan tiga jenis state, yaitu state page, state session dan state aplikasi.

- Page State atau disebut pula sebagai *View State* sebagai state halaman adalah sebuah kondisi yang menyimpan berbagai informasi di tiap-tiap halaman, contohnya isi dari textbox, status checkbox dsb.
- Session State disebut pula penyimpan informasi di tiap sesi untuk masing-masing pengguna pada waktu tertentu, contohnya adalah informasi keranjang belanja dari sebuah toko online, contoh yang lain adalah penyimpan alamat email dari pengguna.
- Application state sebagai penyimpan informasi pada level aplikasi, dimana definis aplikasi adalah semua file yang

berektensi *.aspx yang berada pada folder (tempat) yang sama sebagai bagian dari virtual direktori.

Gambaran ketiga state diatas, diperlihatkan pada gb 13.1 dibawah ini:



Gambar 13.1 Jenis State

13.1.1. Cara AksesTiga Jenis State

Cara mengakses sebuah state bergantung pada jenis state yang dipakai:

- Page state

```
penulisan: ViewState["counter"] = counterVal;
pembacaan: int counterVal = (int) ViewState["counter"];
```

- Session state

```
penulisan: Session["cart"] = shoppingCart;
pembacaan: DataTable shoppingCart =
                    (DataTable) Session["cart"];
```

- Application state

```
penulisan: Application["database"] = databaseName;
pembacaan: string databaseName =
             (string) Application["databaseName"];
```

13.1.2. Kelas Page

Sebuah view state, session state dan aplikasi state merupakan suatu properti dalam kelas page, seperti yang ditunjukkan pada kelas dibawah ini:

```
public class Page: TemplateControl {
       //--- properties
       public  ValidatorCollection Validators { get; }
       public bool IsValid { get; }
       public bool IsPostBack { get; }
       public virtual string TemplateSourceDirectory { get; }
       public HttpApplicationState Application { get; }
       public virtual HttpSessionState Session { get; }
       public HttpRequest Request { get; }
       public HttpResponse Response { get; }
       ...
       //--- methods
       public string MapPath(string virtualPath);
       public virtual void Validate();
       ...
}
```

Sebuah kelas page adalah kelas yang diinstan sebagai obyek untuk masing-masing halaman, dengan beberapa atribut:

- IsValid = bernilai True, jika tidak ada validator pada halaman yang melaporkan kesalahan
- IsPostBack = bernilai True, menunjukkan kondisi halaman sebagai bagian dari *round-trip* yaitu komunikasi antara klien dan server, bernilai false, ketika pertama kali sebuah halaman di *request.*
- TemplateSourceDirectory = penunjukan tempat virtual direktori saat ini.
- Application and Session = nilai-nilai state saat ini
- Request and Response = Penanganan atas permintaan klien dan jawaban dari server.
- **MapPath**(virtPath) = pemetaan virtual direktori ke alamat fisik
- **Validate**() = suatu metode yang berfungsi sebagai eksekusi dari semua validator pada halaman tsb.

Penanganan atas permintaan klien berupa *request* merupakan turunan dari sebuah kelas `HttpRequest` , secara umum kelas ini terdiri dari atribut yang menyimpan data yang dikirimkan oleh klien beserta identitas lain, selengkapnya kelas tsb diberikan dibawah ini:

```
public class HttpRequest {
       public string UserHostName { get; }
       public string UserHostAddress { get; }
       public string HttpMethod { get; }
       public HttpBrowserCapabilities Browser { get; }
       public NameValueCollection Form { get; }
       public NameValueCollection QueryString { get; }
       public NameValueCollection Cookies { get; }
       public NameValueCollection ServerVariables { get; }
       ...
}
```

13.1.3. Kelas HttpRequest

Kelas ini memungkinkan ASP.NET untuk membaca nilai-nilai HTTP yang dikirim oleh klien selama permintaan Web. Pada kelas HttpRequest terdapat atribut informasi tentang pengguna, antara lain:

- UserHostName : nama domain dari klien
- UserHostAddress :alamat / nomor IP klien

Pemakaian untuk menampilkan informasi yang ada pada klien ditunjukkan pada program dibawah ini:

```
<body>
<%= "address = " + Request.UserHostAddress %><br>
<%= "method = " + Request.HttpMethod %><br>
<%= "browser = " + Request.Browser.Browser %><br>
<%= "version = " + Request.Browser.Version %><br>
<%= "supports JS = " + Request.Browser.JavaScript %><br>
<%= "server = " + Request.ServerVariables["SERVER_SOFTWARE"] %>
</body>
```

Pemakaian kelas HttpRequest untuk menangani data yang dikirmkan oleh pengguna ke dalam server dengan memakai koleksi

QueryString, selengkapnya ditunjukkan pada kode program dibawah ini:
Kode program sisi klien:

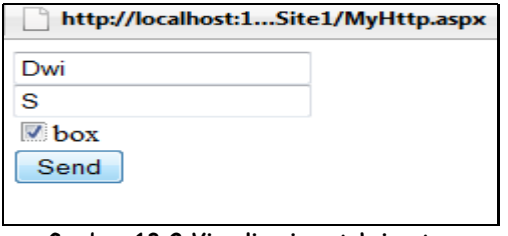
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="MyHttp.aspx.cs" Inherits="MyHttp" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
 <form id="form1" runat="server">
    <div>
    <asp:TextBox ID="text1" Runat="server" /><br>
    <asp:TextBox ID="text2" Runat="server" /><br>
    <asp:CheckBox ID="checkbox" Text="box" Runat="server"
/><br>
    <asp:Button ID="button" Text="Send" OnClick="DoClick"
Runat="server" />
    <asp:Label ID="lab" Runat="server" />
   </div>
 </form>
</body>
</html>
```

Kode program sisi server:

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

public partial class MyHttp : System.Web.UI.Page
{
    protected void DoClick(object sender, EventArgs e)
    {
        lab.Text = "Query string<br>";
        foreach (string par in Request.QueryString.Keys)
            lab.Text += par + " = " + Request.QueryString[par]
                     + "<br>";
        lab.Text += "<br>Form parameters<br>";
```

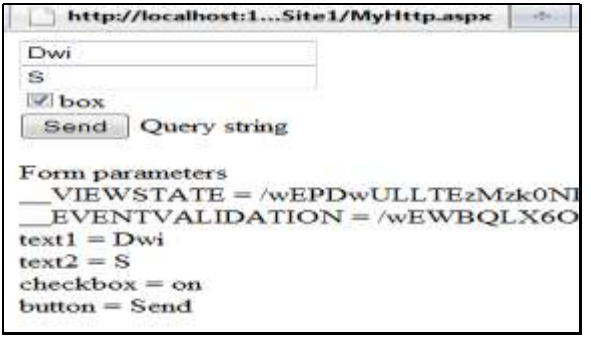
```
        foreach (string par in Request.Form.Keys)
            lab.Text += par + " = " + Request.Form[par] +
                    "<br>";
    }

}
```

Pada sisi klien yang memanggil program tsb akan tampak pada browser dua input textbox, sebuah checkbox dan tombol, seperti yang terlihat pada gambar 13.2 dibawah ini:



Gambar 13.2 Visualisasi contoh inputan

:



Gambar 13.3 Hasil Visualisasi contoh inputan

Hasil dari program, ketika tombol *send* ditekan /diklik dengan terlebih dahulu textbox diisi dan checkbox dicentang, maka pada sisi server data tsb ditampung dalam kelas HttpRequest, khususnya data yang dikirimkan tersimpan pada koleksi QueryString, sehingga server dapat mengambil data-data tsb.

Hasilnya adalah sebuah informasi yang ditampilkan dalam gambar 13.4

13.1.4. Kelas HttpRespons

Kelas HttpResponse adalah kelas yang membungkus informasi yang dilakukan server sebagai tanggapan terhadap sebuah aksi. Secara umum kelas HttpResponse di tunjukkan dibawah ini:

```
public class HttpResponse {
//--- properties
public string ContentType { get; set; }
public TextWriter Output { get; }
public int StatusCode { get; set; }
public HttpCookieCollection Cookies { get; set; }
...
//--- methods
public void Write(string s); // various overloaded versions
public void Redirect(string newURL);
       ...
}
```

Kelas HttpResponse memiliki atribut atau properti yang berisi informasi server:

- ContentType : tipe response, contoh MIME (text/html)
- Output : sebagai obyek yang akan dituliskan di klien
- StatusCode: kode status penanganan server, contoh 200 for "ok" atau 404 for "page not found"

Contoh pemakaian kelas HttpResponse untuk menuliskan kembali ke halaman klien sebagai pengalihan ke halaman tertentu.

Kode program MyHttp2.aspx:

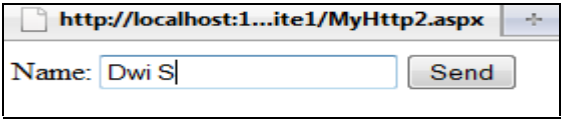
```
<%@ Page Language="C#" AutoEventWireup="true"
CodeFile="MyHttp2.aspx.cs" Inherits="MyHttp2" %>

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title></title>
</head>
<body>
```

```
    <form id="form1" runat="server">
    <div>
    Name: <asp:TextBox ID="name" Runat="server" />
    <asp:Button ID="Button1" Text="Send" OnClick="DoClick"
Runat="server" />
    </div></form>
</body></html>
```

Kode program MyHttp2.aspx.cs:

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

public partial class MyHttp2 : System.Web.UI.Page
{    protected void DoClick(object sender, EventArgs e)
    {Response.Redirect("MyWelcome.aspx?name=" + name.Text);}
}
```

Ketika program dijalankan, maka browser akan menampilkan sebuah isian textbox dengan serbuah tombol , seperti yang terlihat pada gambar 13.4 dibawah ini:

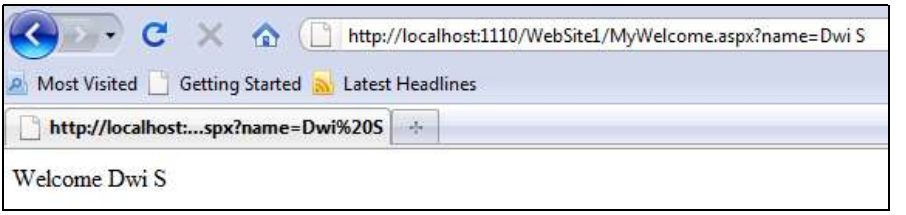


Gambar 13.4 Visualisasi test1.aspx

Pada program test1.aspx, terdapat sebuah aksi ketika tombol *send* ditekan, yakni sebuah response ke halaman Welcome.aspx. Kode program file tsb adalah sbb:
Kode program Welcome.aspx:

```
Welcome <%= Request.QueryString["name"] %> !
```

Hasil dari aksi sebagai response dari penekanan tombol oleh klien, halaman klien akan berganti menuju halaman Welcome.aspx dengan paramater data dari halaman test1.asp. Hal ini terlihat pada gambar 13.5



Gambar 13.5 Visualisasi Welcome.aspx

13.2. Manajemen State Secara Lengkap

Secara lengkap, untuk mengatasi keterbatasan yang melekat pada pemrograman Web tradisional, ASP.NET mencakup beberapa pilihan yang dapat membantu untuk mempertahankan data pada kedua basis per-halaman dan aplikasi secara luas. Fitur-fitur ini yang disediakan adalah sebagai berikut:

- View state
- Control state
- Hidden fields
- Cookies
- Query strings
- Application state
- Session state
- Profile Properties

Suatu *View state, control state, hidden fields, cookies*, dan *query strings* kesemuanya melibatkan penyimpanan data pada klien dengan berbagai cara. Namun, kesemuanya properti *application state, session state, and profile* menyimpan data semua dalam memori di server. Setiap opsi memiliki kelebihan dan kekurangan yang berbeda, tergantung pada skenario.

13.2.1. Pilihan Client-Based State Management

Bagian berikut menjelaskan pilihan bagi manajemen state yang melibatkan baik menyimpan informasi dalam halaman atau pada komputer klien. Untuk pilihan ini, tidak ada informasi yang dikelola di server.

13.2.1.1 View State

Sebuah properti *viewstate* menyediakan kamus objek untuk mempertahankan nilai antara beberapa permintaan untuk halaman yang sama. Metode ini adalah metode default pada halaman yang menggunakan *preserve* halaman dan nilai-nilai kontrol.

Ketika halaman ini diproses, keadaan saat ini halaman dan kontrol akan diacak menjadi string dan disimpan di halaman sebagai *fiekd* yang tersembunyi, atau beberapa field tersembunyi jika jumlah data yang disimpan dalam kondisi tampilan properti melebihi nilai yang ditentukan dalam properti MaxPageStateFieldLength. Ketika halaman ini diposting kembali ke server, halaman tersebut mem-parsing string *view state* yang telah di inisialisasi pada halaman dan mengembalikan informasi properti halaman. Data juga dapat disimpan dalam *views state*

13.2.1.2 Control State

Kadang-kadang diperlukan untuk menyimpan data control state agar sebuah kontrol bekerja dengan baik. Misalnya, jika sebuah kontrol kustom yang memiliki tab yang berbeda menunjukkan informasi yang berbeda pula, agar kontrol yang bekerja seperti yang diharapkan, kontrol perlu tahu mana tab yang dipilih.

Properti *viewstate* bisa digunakan untuk tujuan ini, namun sering *view state* dimatikan pada tingkat halaman oleh pengembang / programmer untuk tujuan tertentu, hal ini menyebabkan menggangu kerja dari kontrol ybs. Untuk mengatasi hal ini, framework ASP.NET memperkenalkan suatu fitur yang disebut *control state*. Suatu properti *control state* memungkinkan untuk mempertahankan properti informasi yang spesifik untuk sebuah kontrol, dimana *control state* tidak dapat dimatikan seperti halnya properti *viewstate.*

13.2.1.3 Hidden Fields

ASP.NET memungkinkan untuk menyimpan informasi dalam kontrol HiddenField, dimana informasi ini akan disembunyikan dari pengguna, tidak terlihat pada kode HTML standar. Sebuah *field* tersembunyi tidak menjadikan terlihat dalam browser. Ketika halaman di *submit* ke server, isi field tersembunyi dikirim dalam bentuk koleksi HTTP bersama dengan nilai-nilai kontrol yang lain. Sebuah *field* tersembunyi bertindak sebagai repositori untuk informasi pada halaman-spesifik.

Sebuah HiddenField menyimpan variabel tunggal dalam properti *value* dan harus secara eksplisit ditambahkan ke halaman.

13.2.1.4 Cookies

Suatu Cookie adalah sejumlah kecil data yang disimpan baik dalam sebuah file teks di sistem berkas klien atau di memori pada sesi browser klien. Suatu Cookies berisi informasi situs tertentu dimana server akan mengirimkan ke klien bersama dengan output halaman. Cookies dapat bersifat sementara (dengan waktu atau tanggal tertentu) atau bersifat persisten.

Pemakaian cookies biasanya untuk menyimpan informasi tentang informasi klien tertentu, sesi, atau aplikasi. Semua Cookie disimpan pada perangkat klien, dan ketika browser meminta halaman, klien mengirimkan informasi dalam cookie bersama dengan informasi permintaan. Server dapat membaca cookie dan mengekstrak nilainya. Suatu penggunaan yang khas pada cookie adalah pemakaian untuk menyimpan tanda (mungkin terenkripsi), tanda ini menunjukkan bahwa pengguna telah disahkan dalam aplikasi.

13.2.1.5 Query Strings

Sebuah string query adalah informasi yang ditambahkan ke akhir URL halaman. Sebuah string query khas mungkin terlihat seperti contoh berikut:

```
http://freak.com/listwidgets.aspx?category=basic&price=100
```

Di jalur URL di atas, query string dimulai dengan tanda tanya (?) Dan termasuk dua pasangan atribut / nilai, satu yang disebut "category" dan yang disebut lainnya "price"

Sebuah Query string menyediakan cara sederhana tetapi terbatas untuk menjaga informasi state. Sebuah Query String adalah cara mudah untuk melewatkan informasi dari satu halaman ke halaman lainnya, seperti melewati sejumlah produk dari satu halaman ke halaman lain di mana ia akan diproses. Namun, beberapa browser dan perangkat klien memaksakan batas 2.083-karakter pada panjang URL.

13.2.2. Pilihan Server-Based State Management

ASP.NET menawarkan berbagai cara untuk menjaga informasi state yang terdapat di server, selain informasi yng disimpan pada klien. Dengan manajemen state berbasis server, pekerjaan dapat dikurangi, tidak hanya menumpuk pada sisi klien, namun dapat menggunakan sumber daya yang ada di server.

13.2.2.1 Application State

ASP.NET memungkinkan Anda untuk menyimpan nilai-nilai menggunakan state aplikasi - yang merupakan turunan dari kelas HttpApplicationState - untuk setiap aplikasi Web yang aktif. Sebuah *Application state* adalah mekanisme penyimpanan global yang dapat diakses dari semua halaman dalam aplikasi Web. Dengan demikian, *Application state* berguna untuk menyimpan informasi yang perlu dijaga antara klient dan server terhadap sebuah permintaan halaman.

Sebuah *Application state* disimpan dalam sebuah kunci / nilai kamus yang dibuat pada setiap permintaan untuk URL tertentu. Penyimpanan lewat *Application state* dapat menambahkan informasi tentang aplikasi khusus dengan sebuah struktur

informasi untuk menyimpan di antara permintaan halaman. Setelah menambahkan informasi aplikasi-aplikasi spesifik untuk suatu state, server akan mengelola state tsb.

13.2.2.2. Session State

ASP.NET memungkinkan untuk menyimpan nilai dengan menggunakan state sesi - yang merupakan turunan dari kelas HttpSessionState - untuk setiap sesi Web-aplikasi yang aktif.

Sebuah *Session State* mirip dengan *Application state*, perbedaannya terletak pada scoped. Jika pengguna yang berbeda yang menggunakan sebuah aplikasi, setiap sesi pengguna akan memiliki *Session State* yang berbeda. Selain itu, jika pengguna meninggalkan aplikasi dan kemudian kembali lagi, maka sesi pengguna yang kedua akan memiliki sesi yang berbeda dari yang pertama.

Sebuah *Session State* disusun sebagai kunci / nilai kamus untuk informasi sesi yang khusus untuk menyimpan data yang harus dijaga antara klient dan server pada permintaan halaman.

Pemakaian *Session State* untuk menyelesaikan tugas-tugas berikut:

- Secara unik mengidentifikasi browser atau permintaan klien sebagai suatu map ke sebuah sesi individu pada server.
- Menyimpan data sesi yang khusus di server untuk digunakan di beberapa browser atau permintaan klien dalam sesi yang sama.
- Mendapatkan *events* terhadap session management yang sesuai. Selain itu, dapat pula menulis kode aplikasi dengan memanfaatkan peristiwa ini.

13.2.2.3 Profile Properties

ASP.NET menyediakan fitur yang disebut properti profil, yang memungkinkan untuk menyimpan data pengguna tertentu. Fitur ini mirip dengan *Session State*, kecuali bahwa data profil ini tidak hilang ketika sesi pengguna berakhir. Properti profil menggunakan profil ASP.NET, yang disimpan dalam format persisten dan terkait dengan pengguna individu. Profil ASP.NET memungkinkan untuk dengan mudah data ataupun mengatur informasi pengguna tanpa anda harus membuat dan memelihara database sendiri. Selain itu, profil membuat informasi pengguna yang tersedia menggunakan API sebagai *strongly typed* yang dapat diakses dari mana saja dalam aplikasi. Data dapat disimpan sebagai objek dari setiap tipe dalam profil. Fitur profil ASP.NET menyediakan sistem penyimpanan generik yang memungkinkan untuk menentukan dan mempertahankan hampir semua jenis data

Untuk menggunakan properti profil, terlebih dahulu harus mengkonfigurasi penyedia profil. ASP.NET

Pada ASP.NET mencakup kelas SqlProfileProvider yang memungkinkan untuk menyimpan data profil dalam database SQL, selain itu, juga dapat membuat profil penyedia kelas terhadap data sendiri yang menyimpan data profil dalam format kustom dan mekanisme penyimpanan kustom seperti file XML, atau ke layanan Web. Karena data yang ditempatkan dalam properti profil tidak disimpan dalam memori aplikasi, namun disimpan di Web Server, dalam hal ini adalah Internet Information Services (IIS).