Bab 8

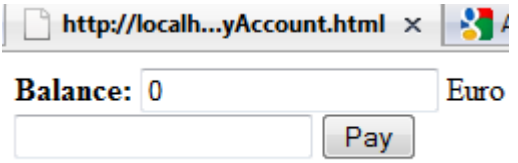
Web Form pada ASP.NET

8.1 Halaman Form HTML

Halaman-halaman web, biasanya menerima input dari pemakai dimana area input ini ditandai dengan tag <form> dan di akhir dengan tag </form>. Didalam tag form, dapat dipakai kode action yang menunjukkan proses selanjutnya dari form tsb ketika di submit. Biasanya action ini berupa program CGI yang berada di web server:

```
...
<body>
<form action="http://foo.com/cgi-bin/myprog" method="post">
<b>Balance:</b>
<input type="text" name="total" readonly value="0"> Euro<br>
<input type="text" name="amount">
<input type="submit" name="ok" value="Pay">
</form>
</body>
```

Kode program tsb dijalankan pada web browser, terdapat dua input dengan sebuah tombol yang dapat diklik sebagai submit ke proses selanjutnya yaitu sebuah program CGI, myprog.



Gambar 8.1 Hasil Kode Form

Proses yang dilakukan server sebagai CGI script myprog adalah

- Membaca *total* dan *amount*
- Mengirimkan kembali text HTML baru dengan nilai baru pada text *total* and *amount*

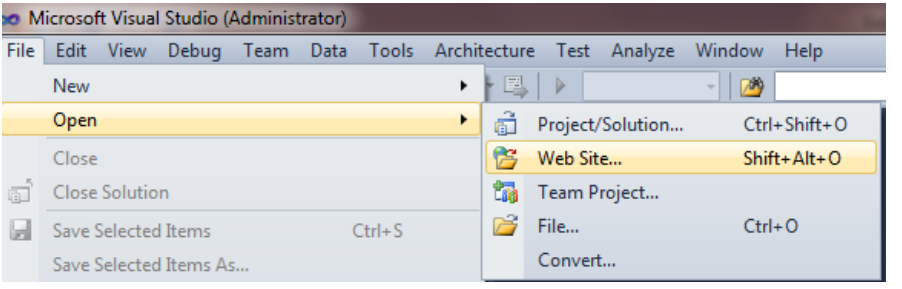
Proses ini memiliki persoalan pada premrograman CGI, ketika halaman di kirimkan kembali, maka state dari text field harus di atur secara manual

8.2 Halaman Form pada ASPX

Beberapa persoalan pada pemrograman yang lama, diatasi oleh ASPX, seperti yang telah dibahas pada bab 7. Cara pemrograman pada ASP.NET dapat melalui penulisan kode text atau memakai tool VS 2010.

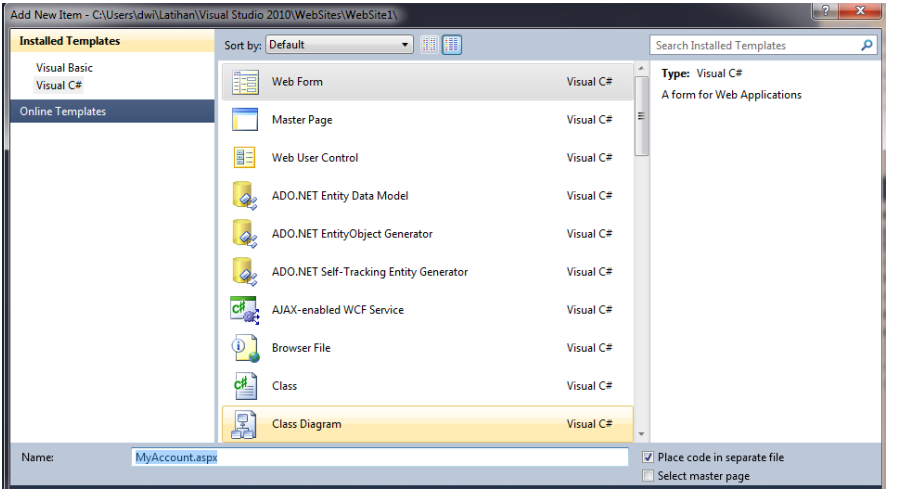
Langkah-langkah untuk menyelesaikan persoalan pada subab 8.1, dengan memakai VS2010:

1. Buka VS2010 dan buat situs baru (atau memakai situs pada bab sebelumnya yaitu bab 7), hal ini terlihat pada gambar 8.2



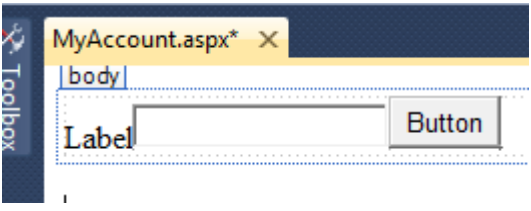
Gambar 8.2 Menu File Baru

2. Buat File program baru, yaitu pilih `File -> New-> File` atau langsung menekan Ctrl-N, beri nama file `MyAccount` seperti terlihat pada gb 8.3, dibawah ini. Jendela yang terbuka adalah tempat untuk memilih template, pastikan memilih web form seperti yang terlihat pada gambar 8.3



Gambar 8.3 Pilihan template

3. Setelah memilih `template` dengan pilihan `webform`, kemudian susun rancangan form dengan tiga komponen, yaitu label, textbox dan button. Pembuatan komponen ini dapat memanfaatkan toolbox, dengan cara drag dan drop ke form yang tersedia, secara sederhana rancangan itu terlihat pada gambar 8.4



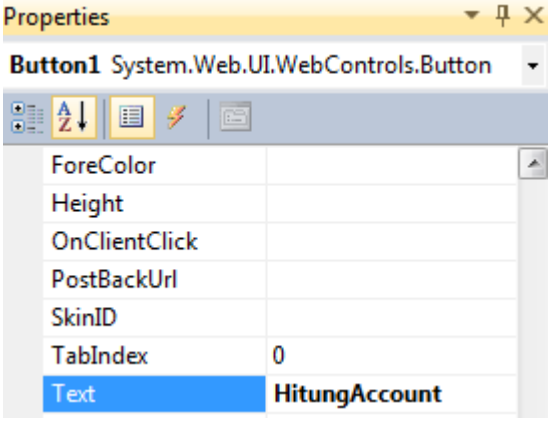
Gambar 8.4 Rancangan MyAccount.aspx

Mode desain pada VS2010 yang terlihat pada gb 8.4 dapat juga diperlihatkan kode programnya seperti terlihat pada script dibawah ini:

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

<!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:Label ID="Label1" runat="server" Text="Label"></asp:Label>
<asp:TextBox ID="TextBox1" runat="server"></asp:TextBox>
<asp:Button ID="Button1" runat="server" Text="Button" />
</div></form>
</body>
</html>
```

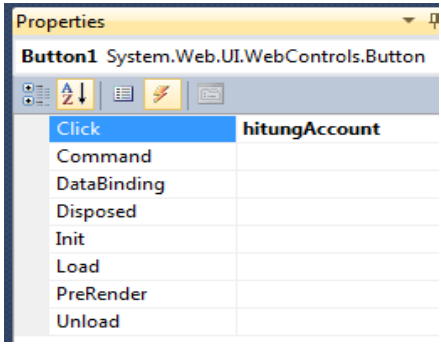
Script Program

4. Setelah desain terbentuk, pada komponen tombol, atur pada jendela properti, utamanya adalah tampilan menjadi hitungAccount seperti terlihat pada gambar 8.5



Gambar 8.5 Jendela Properti Button1

5. Pada jendela properti, tekan tanda petir ‘  ‘ untuk menunjukkan pengaturan aksi, isi aksi jika tombol tsb diklik dengan nama htiungAccount, seperti gambar 8.6 dibawah ini



Gambar 8.6 Jendela Properti *Events* Button1

6. Setelah mengetik nama dari events tsb, maka secara otomatis terbuka halaman kode sumber untuk hitungAccount diats, hal ini ditampilkan seperti gambat 8.7
7. Ketika kode sumber yang berikut dan hapus beberapa pustaka yang tidak diperlukan, sehingga sumber secara lengkap seperti skrip dibawah ini:

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

public partial class MyAccount : System.Web.UI.Page
{
    protected void hitungAccount(object sender, EventArgs e)
    {
        int total = 1000;
        int nilai = Convert.ToInt32(TextBox1.Text);
        Label1.Text = (total + nilai).ToString();

    }
}
```

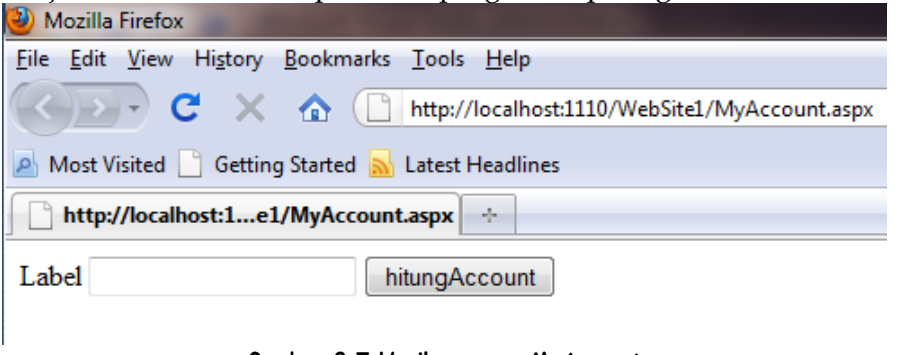
MyAccount.aspx.cs* | MyAccount.aspx*

MyAccount

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.UI;
using System.Web.UI.WebControls;

public partial class MyAccount : System.Web.UI.Page
{
    protected void Page_Load(object sender, EventArgs e)
    {

    }

    protected void hitungAccount(object sender, EventArgs e)
    {

    }
}
```

Gambar 8.7 Jendela Kode Sumber MyAccount.aspx.cs

8. Jalankan program tsb dengan menekan F5, sehingga pada jendela browser tampak hasil program seperti gambar 8.8



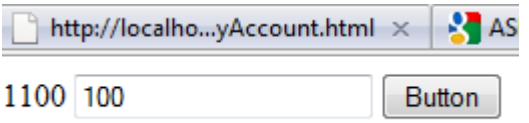
Gambar 8.7 Hasil program MyAccount

Pada tempat script program C#, disitu kode-kode aksi dapat dilakukan, penambahan kode untuk penanganan penekanan tombol tsb, seperti contoh diatas:

```
int total = 1000;
int nilai = Convert.ToInt32(TextBox1.Text);
Label1.Text = (total + nilai).ToString();
```

Program diatas ketika tombol hitungAccount ditekan, maka kendali program memberikan isi variabel total dengan nilai 1000 (baris pertama). Dan akan menngkonversi isi yang ada di textbox ke dalam variabel nilai (baris ke dua). Konversi ini adalah dari string ke bentuk integer. Setelah itu (pada baris ke tiga) total dan nilai di jumlahkan, hasilkan di konversi ke dalam bentuk string kembali dan di-isikan ke label1, sehingga ketika textbox di-isi dengan 100, hasilnya akan bernilai 1100, seperti pada gambar 8.8

Hasil program aspx dijalankan berupa input mirip dengan bambar 8.1, namun lebih teratur seperti terlihat pada gambar 8.6 dibawah ini



Gambar 8.8 Hasil Kode Form Default2.aspx

Pada kode program MyAccount.aspx dengan kode sumber MyAccount.aspx.cs. terhubung dengan statemen

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

Kode program MyAccount.aspx mengembalikan file kode HTML yang dibangkitkan, dimana kode program MyAccount.aspx akan tampak berbeda ketika dibuka sebagai source code, berikut ini hasil dari proses ketika telah dilakukan submit:

Pada kode HTML yang dihasilkan dari pembangkitan *page class*, terlihat sebuah input baru berupan input tipe hidden untuk menjaga state di browser dengan state di server.

```
<!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><title>

</title></head>
<body>
<form method="post" action="MyAccount.aspx" id="form1">
<div class="aspNetHidden">
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwULLTEzMTc4NjY3NTgPZBYCAgMPZBYCAgEPDxYCHgRUZXh0BQQx
MTAwZGRk5gldXhDUi4ePORwnX1cN036dc1HysY2a6u8Kxb6gBoc=" />
</div>
<div class="aspNetHidden">
<input type="hidden" name="__EVENTVALIDATION"
id="__EVENTVALIDATION"
value="/wEWAwLJyejlCwLs0bLrBgKM54rGBpVS7sA8mwURrbquddEbIFAPXss1
mcAE/FFguNqCkHIJ" />
</div><div>
<span id="Label1">1100</span>
<input name="TextBox1" type="text" value="100" id="TextBox1" />
<input type="submit" name="Button1" value="hitungAccount"
id="Button1" />
</div> </form></body></html>
```

8.3 Notasi Umum pada Kontrol Halaman Web

Program ASPX membuat aturan umum untuk mengatur halaman web secara umum seperti dibawah ini

```
<asp:ClassName PropertyName="value" ... Runat="server" />
```

Kode Tag `<asp: … />` adalah tag yang akan diterjemahkan oleh web server sebagai program yang harus diproses dalam framework .NET dimana basisnya adalah kelas kelas yang telah tersedia ataupun kelas yang dibuat oleh pemrograman, kemudian

beberapa pilihan seperti `PropertyName` untuk setting property dari obyek tsb. Kesemuanya kode tag <asp: .. /> dijalankan di sisi server dengan kode `Runat="server"`.

Contoh penulisan sebuah label pada halaman web, kode tag asp adalah seperti dibawah ini:

```
<asp:Label ID="total" Text="Hello" ForeColor="Red"
Runat="server" />
```

Didalam web server, tag tsb akan dibangkitkan menjadi sebuah kelas, yang semua kelas control web berada pada sebuah namespace System.Web.UI. Hasil pembangkitan kelas tsb, berupa kelas seperti dibawah ini:

```
public class Label: WebControl {
       public virtual string ID { get {...} set {...} }
       public virtual string Text { get {...} set {...} }
       public virtual Color ForeColor { get {...} set {...} }
       ...
}
```

Penulisan lain pada tag asp, dapat berupa blok bagian seperti ditunjukkan pada kode program dibawah ini

```
<asp:Label ID="total" ForeColor="Red" Runat="server" >
       Hello
</asp:Label>
```

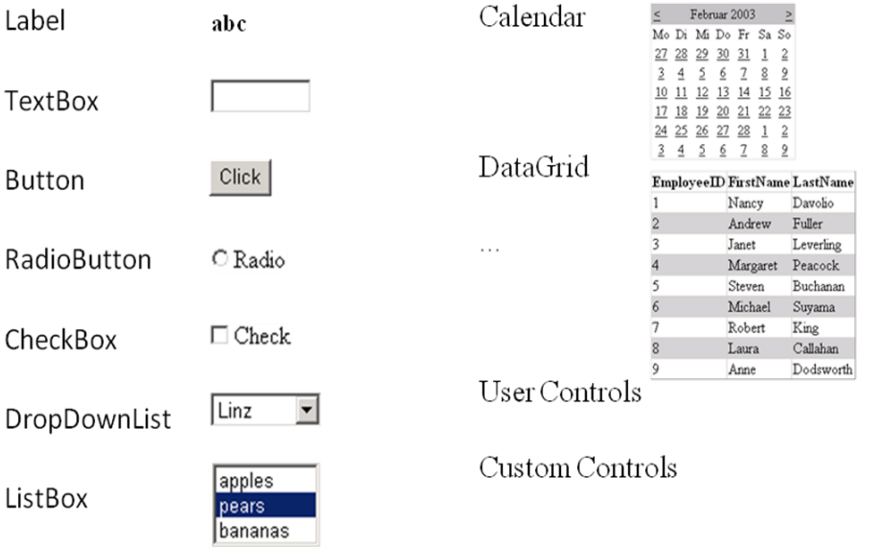
8.4 Keuntungan dari Web Form pada ASPX

Pada program program lama, seperti ASP biasa, web server menerjemahkan file berupa skrip program dijalankan satu persatu, sehingga web server menjadi terbebani dengan metode ini. Ketika web server dipisahkan dengan sebuah framework, maka tugas web server terbagi ke dalam framework yang bertugas untuk mentranslasi sebuah kode menjadi obyek. Cara pandang terhadap sebuah halaman adalan sebuah obyek yang dapat diakses semua

property dan metodenya, seperti `page.IsPostBack, page.User, page.FindControl(), ...`

Tidak hanya halaman saja yang dipandang sebagai obyek, semua elemen GUI adalah obyek yang dapat diakses seperti misalnya obyek amount, dapat diakses metodenya: `amount.Text, amount.Font, amount.Width, ...`

Halaman web dapat pula mengimplementasikan obyek, dengan cara mengkustomisasi elemen GUI, cara yang lain dapat dilakukan pula dengan halaman web yang di akses dengan pustaka .NET, XML, RMI. Gambaran elemen menjadi obyek terletak pada perancangan GUI yang dapat dipilih secara langsung seperti ditunjukkan pada gambar 8.9

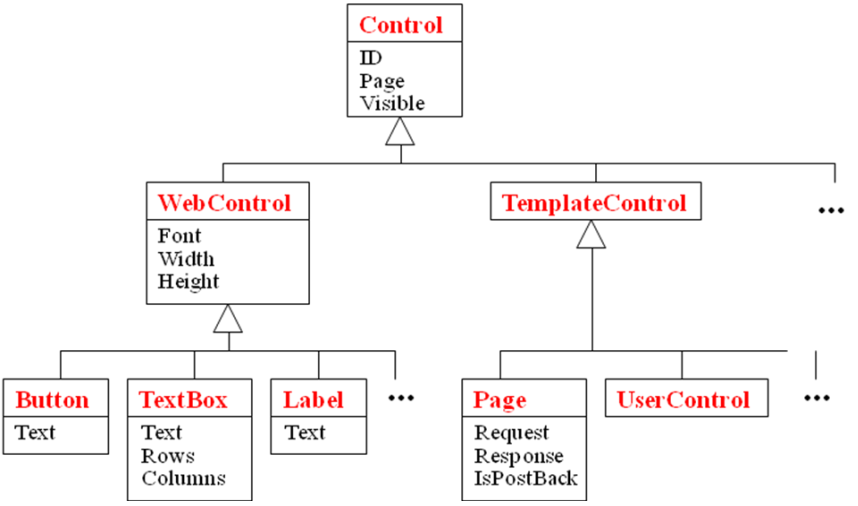


Gambar 8.9 Web Control

Secara umum, web control memiliki hierarki diantara properti dan metodenya, gambar hierarki ini diperlihat pada gb 8.10

Sebuah halaman web adalah tingkat tertinggi dari hierarki yang memiliki ID untuk membedakan dengan halaman web yang lain. Tingkat dibawahnya terdapat isi dari halaman web berupa web control berupa properti `Font, Width dan Height`.

Halaman Web dapat memiki kontrol lain, seperti `button`, `textbox`, `label` dsb. Setiap halaman web memiliki template control yang berupa control untuk halaman berupa `request`, `response` dan `isPostBack`. Selain itu pemrograman dapat menambahkan user control yang lain untuk mengontrol halaman web.



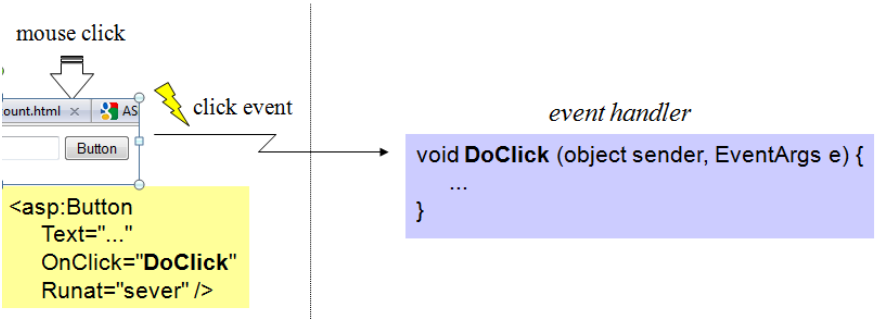
Gambar 8.10 Hierarki Web Control

8.5 Proses Berdasarkan Event

Ketika sebuah obyek diberikan aksi oleh pemakai, maka proses selanjutnya dapat ditelusuri dengan kontrol pada obyek tsb melalui event yang terjadi. Sebagai contoh sebuah tombol di klik, maka event yang dapat ditangani adalah event mouse_click pada sisi browser. Setelah itu browser mengirimkan event tsb ke sisi server dengan memproses event tersebut. Proses ini dapat digambarkan pada gb 8.11.

Terlihat pada gb 8.11 sebuah tombol dengan event `OnClick` yang akan meneruskan proses tsb ke server dengan fungsi yang telah disebutkan pada events tsb yaitu `DoClick`.

Pada sisi server, sebuah fungsi `DoClick` akan dipanggil untuk menerima respon dari events `mouse click` pada sisi brower. Setelah itu state berikutnya dikembalikan lagi ke browser yang memanggil secara defaultnya.



Gambar 8.11 Proses Event mouse click

Secara umum, jenis events pada masing-masing obyek adalah mirip dengan beberapa obyek yang memiliki karaktersitik yang berbeda.

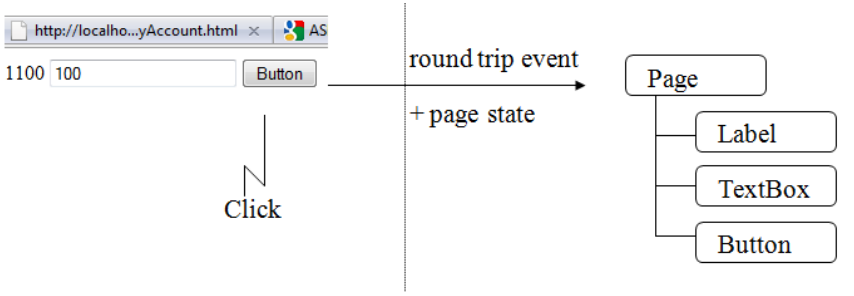
Tabel 8.1 Event Umum

Kontrol	Event	Kapan Event terjadi
all	**Init** **Load** **PreRender** **Unload**	• Ketika kontrol dibuat • data dikirim oleh browser telah masuk kedalam kontrol • sebelum kode HMTL dibangkitkan oleh kontrol ybs • sebelum kontrol dihapus dari memori
Button	**Click**	Ketika tombol di click
TextBox	**TextChanged**	Ketika isi dari textbox berubah

CheckBox	**CheckedChanged**	Ketika state dari sebuah CheckBox mengalami perubahan
ListBox	**SelectedIndexChanged**	Ketika nilai baru dari sebuah list di pilih

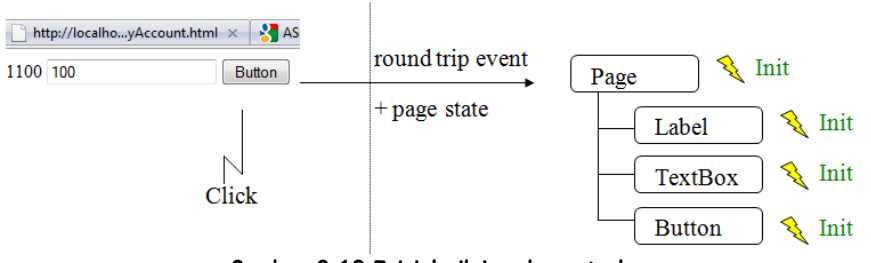
Gambaran proses event yang terjadi pada program subab 8.2 tentang `MyAccount.aspx` adalah

1. Tahap *creation,* Pembuatan Halaman Web, diperlihatkan pada gambar 8.12



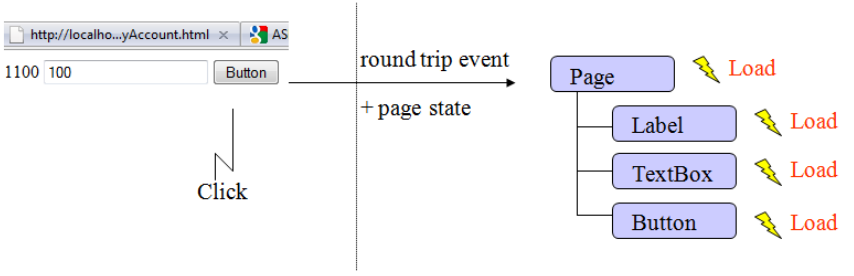
Gambar 8.12 Proses Event Program Default2.aspx

2. Tahap *initial,* Inisialisi disebabkan tombol diklik, sehingga semua kontrol dan page-nya di inisial awal pada sisi server, terlihat seperti gambar 8.13, pada tahap ini, event initial di kerjakan.



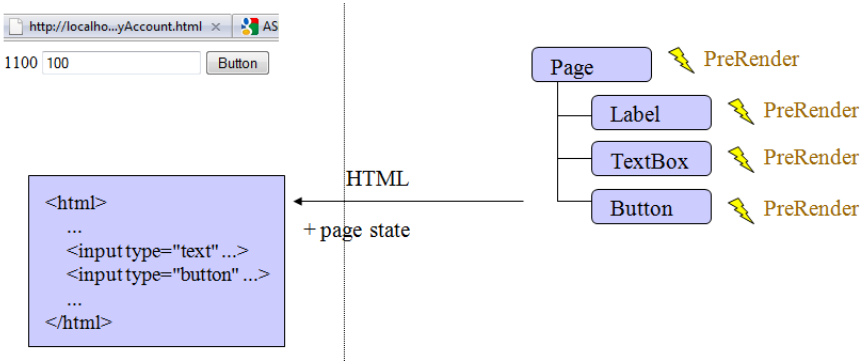
Gambar 8.13 Inisial nilai pada control

3. Tahap *loading,* proses memasukkan nila yang diberikan oleh pemakai ke kontrol pada tahap awal, hal ini membangkitkan event loading dikerjakan, terlihat gambar tahap ini pada gb 8.14



Gambar 8.14 Inisial nilai pada control

4. Tahap *action,* proses interaksi antara pemakai dengan browser terjadi pada tahap *action*, seperti `handle event(s), (Click, TextChanged, ...)`
5. Tahap *rendering,* memanggil metode Render dari semua kontrol yang terlibat dan menghasilkan kode HTML. Tahap ini didahului dengan membangkitkan *events prerender* gambaran tahap action dilihat pada gb 8.15



Gambar 8.15 Tahap rendering