PV247 – Development I

Introduction to ASP.NET and related technologies

Kentico

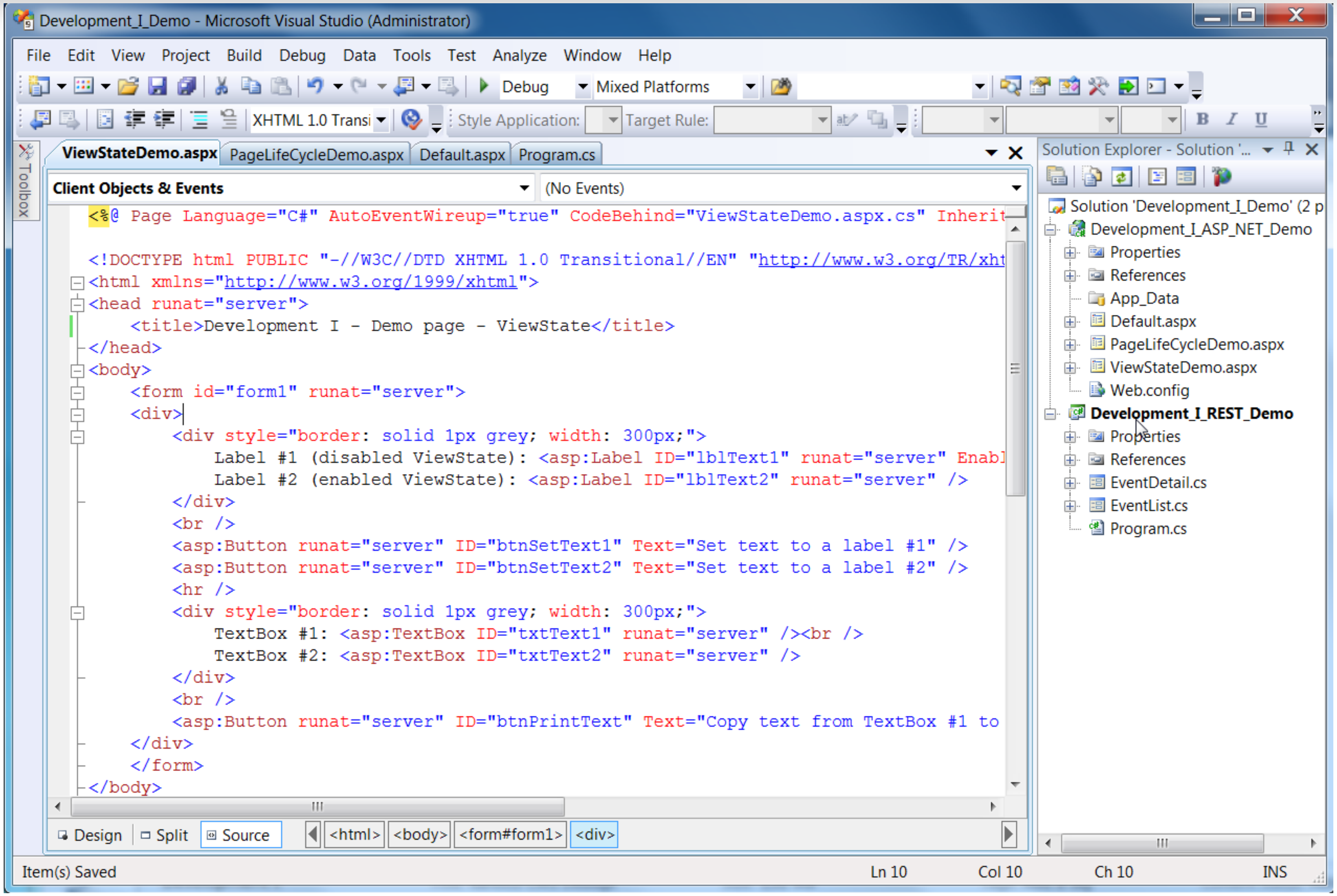
Overview

- **ASP.NET Basics**
  - What is ASP.NET?
  - What is a request?
  - How does ASP.NET deal with stateless http?
  - ASP.NET request/page/control life cycle
  - REST Services
- **Kentico CMS Platform Basics**
  - CMS.IO namespace

What is ASP.NET?

- **Active Server Pages .NET**
- Platform for creating dynamic web applications
- You can use any .NET language as a code-behind
- Development of ASP.NET WebForms applications can be similar to the development of WinForms applicatons. Similar, not same!

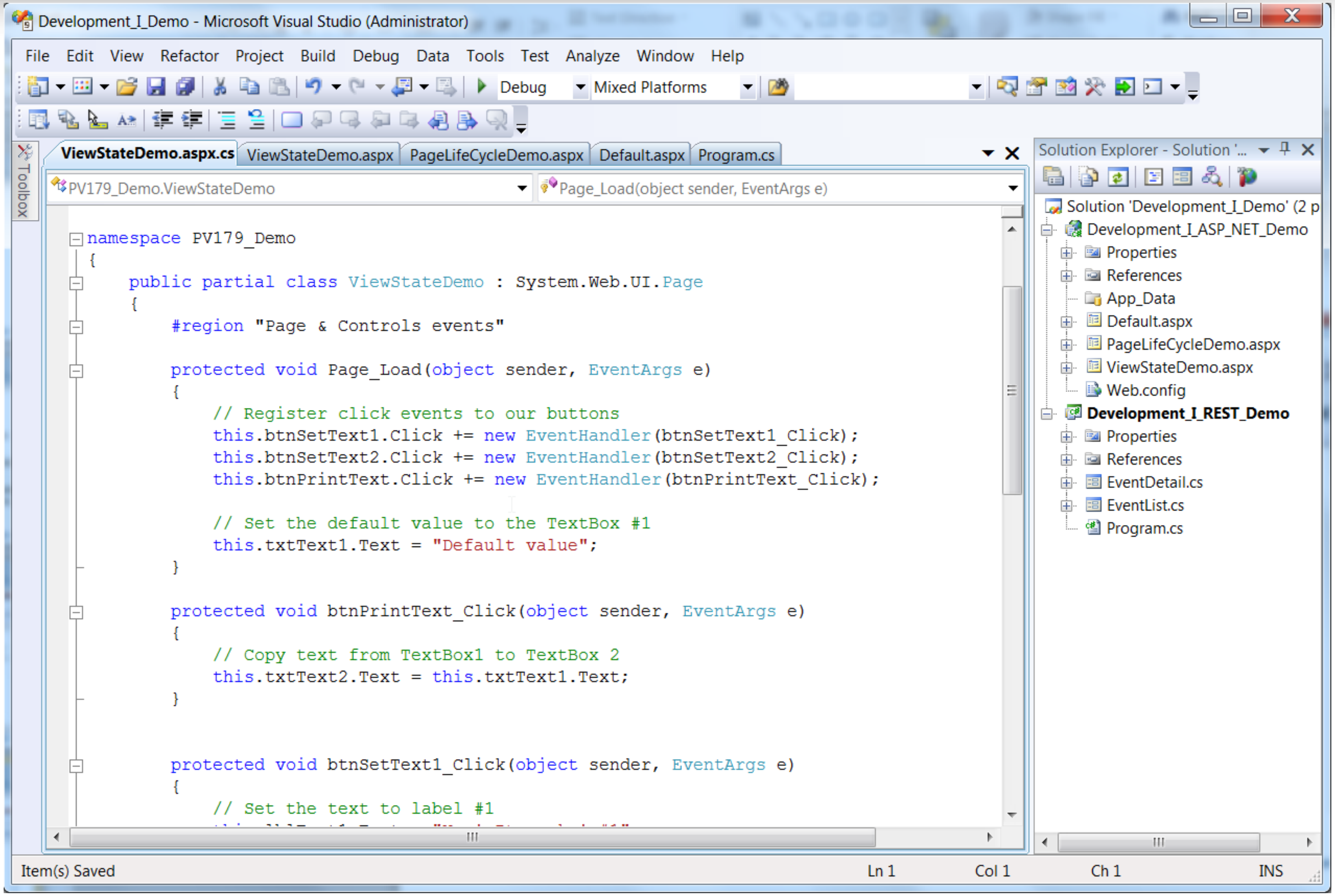
Example of ASP.NET page



```
<%@ Page Language="C#" AutoEventWireup="true" CodeBehind="ViewStateDemo.aspx.cs" Inherit

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org/TR/xht
<html xmlns="http://www.w3.org/1999/xhtml">
<head runat="server">
    <title>Development I - Demo page - ViewState</title>
</head>
<body>
    <form id="form1" runat="server">
    <div>
        <div style="border: solid 1px grey; width: 300px;">
            Label #1 (disabled ViewState): <asp:Label ID="lblText1" runat="server" Enabl
            Label #2 (enabled ViewState): <asp:Label ID="lblText2" runat="server" />
        </div>
        <br />
        <asp:Button runat="server" ID="btnSetText1" Text="Set text to a label #1" />
        <asp:Button runat="server" ID="btnSetText2" Text="Set text to a label #2" />
        <hr />
        <div style="border: solid 1px grey; width: 300px;">
            TextBox #1: <asp:TextBox ID="txtText1" runat="server" /><br />
            TextBox #2: <asp:TextBox ID="txtText2" runat="server" />
        </div>
        <br />
        <asp:Button runat="server" ID="btnPrintText" Text="Copy text from TextBox #1 to
    </div>
    </form>
</body>
```

Kentico

ASP.NET page – code behind



Development_I_Demo - Microsoft Visual Studio (Administrator)

File Edit View Refactor Project Build Debug Data Tools Test Analyze Window Help
Debug
Mixed Platforms
Toolbox
ViewStateDemo.aspx.cs ViewStateDemo.aspx PageLifeCycleDemo.aspx Default.aspx Program.cs
PV179_Demo.ViewStateDemo
Page_Load(object sender, EventArgs e)
Solution Explorer - Solution '...

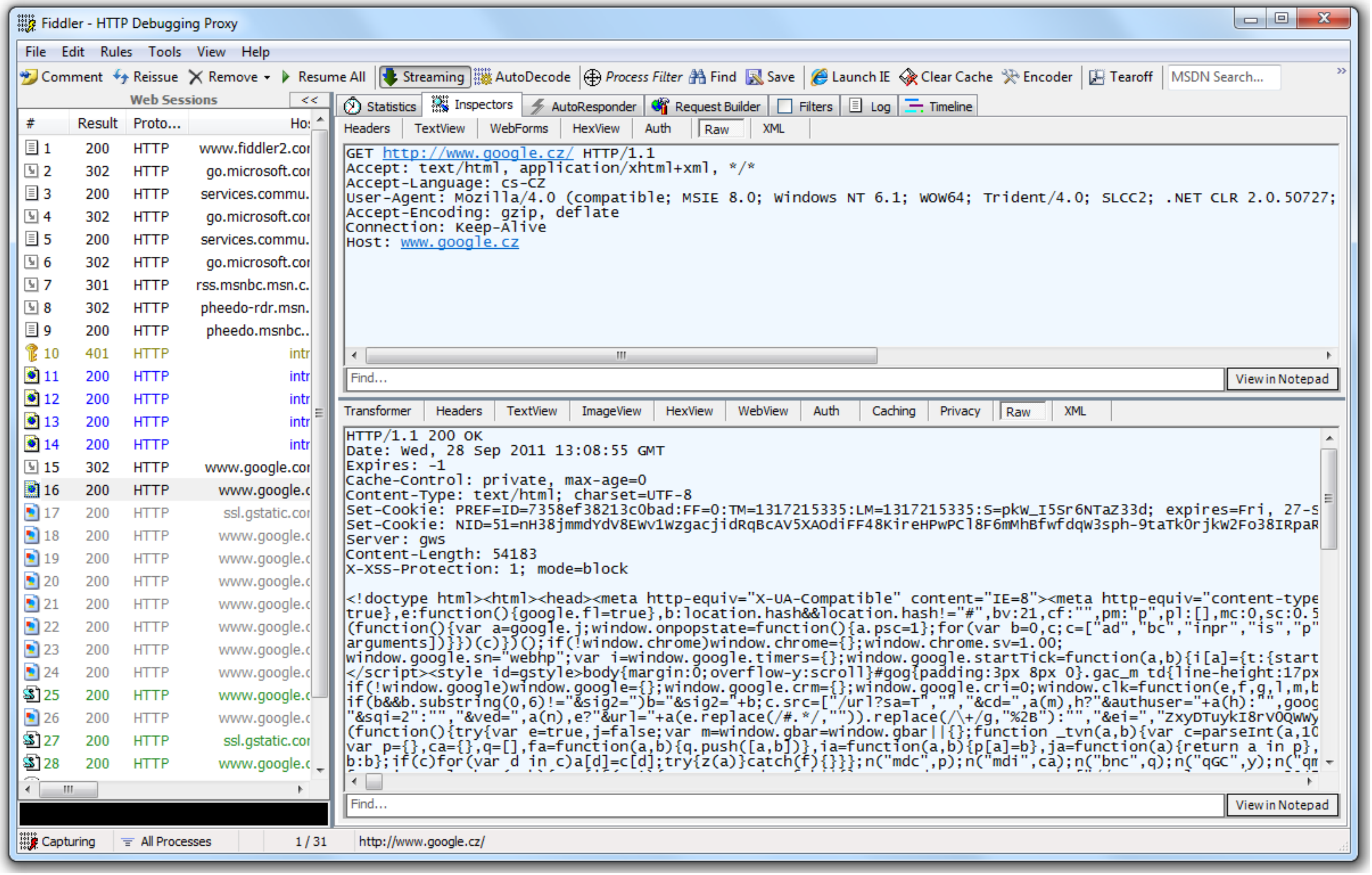
```
namespace PV179_Demo
{
    public partial class ViewStateDemo : System.Web.UI.Page
    {
        #region "Page & Controls events"

        protected void Page_Load(object sender, EventArgs e)
        {
            // Register click events to our buttons
            this.btnSetText1.Click += new EventHandler(btnSetText1_Click);
            this.btnSetText2.Click += new EventHandler(btnSetText2_Click);
            this.btnPrintText.Click += new EventHandler(btnPrintText_Click);

            // Set the default value to the TextBox #1
            this.txtText1.Text = "Default value";
        }

        protected void btnPrintText_Click(object sender, EventArgs e)
        {
            // Copy text from TextBox1 to TextBox 2
            this.txtText2.Text = this.txtText1.Text;
        }


        protected void btnSetText1_Click(object sender, EventArgs e)
        {
            // Set the text to label #1
```

Solution 'Development_I_Demo' (2 p
- Development_I_ASP_NET_Demo
  - Properties
  - References
  - App_Data
  - Default.aspx
  - PageLifeCycleDemo.aspx
  - ViewStateDemo.aspx
  - Web.config
- **Development_I_REST_Demo**
  - Properties
  - References
  - EventDetail.cs
  - EventList.cs
  - Program.cs

Item(s) Saved Ln 1 Col 1 Ch 1 INS

Kentico

What is a request?



Fiddler - HTTP Debugging Proxy
File Edit Rules Tools View Help
Comment Reissue Remove Resume All Streaming AutoDecode Process Filter Find Save Launch IE Clear Cache Encoder Tearoff MSDN Search...
Web Sessions
Statistics Inspectors AutoResponder Request Builder Filters Log Timeline
Headers TextView WebForms HexView Auth Raw XML
Find... View in Notepad
Transformer Headers TextView ImageView HexView WebView Auth Caching Privacy Raw XML
Capturing All Processes 1 / 31 http://www.google.cz/

#	Result	Proto...	Ho:
1	200	HTTP	www.fiddler2.cor
2	302	HTTP	go.microsoft.cor
3	200	HTTP	services.commu.
4	302	HTTP	go.microsoft.cor
5	200	HTTP	services.commu.
6	302	HTTP	go.microsoft.cor
7	301	HTTP	rss.msnbc.msn.c.
8	302	HTTP	pheedo-rdr.msn.
9	200	HTTP	pheedo.msnbc..
10	401	HTTP	intr
11	200	HTTP	intr
12	200	HTTP	intr
13	200	HTTP	intr
14	200	HTTP	intr
15	302	HTTP	www.google.cor
16	200	HTTP	www.google.c
17	200	HTTP	ssl.gstatic.cor
18	200	HTTP	www.google.c
19	200	HTTP	www.google.c
20	200	HTTP	www.google.c
21	200	HTTP	www.google.c
22	200	HTTP	www.google.c
23	200	HTTP	www.google.c
24	200	HTTP	www.google.c
25	200	HTTP	www.google.c
26	200	HTTP	www.google.c
27	200	HTTP	ssl.gstatic.cor
28	200	HTTP	www.google.c

```
GET http://www.google.cz/ HTTP/1.1
Accept: text/html, application/xhtml+xml, */*
Accept-Language: cs-CZ
User-Agent: Mozilla/4.0 (compatible; MSIE 8.0; Windows NT 6.1; WOW64; Trident/4.0; SLCC2; .NET CLR 2.0.50727;
Accept-Encoding: gzip, deflate
Connection: Keep-Alive
Host: www.google.cz
```

```
HTTP/1.1 200 OK
Date: Wed, 28 Sep 2011 13:08:55 GMT
Expires: -1
Cache-Control: private, max-age=0
Content-Type: text/html; charset=UTF-8
Set-Cookie: PREF=ID=7358ef38213c0bad:FF=0:TM=1317215335:LM=1317215335:S=pkW_I5Sr6NTaZ33d; expires=Fri, 27-S
Set-Cookie: NID=51=nH38jmmdYdV8EWv1WzgacjidRqBCAV5XAOdiFF48KireHPwPCl8F6mMhBfwfdqW3sph-9taTkOrjkW2Fo38IRpaR
Server: gws
Content-Length: 54183
X-XSS-Protection: 1; mode=block

<!doctype html><html><head><meta http-equiv="X-UA-Compatible" content="IE=8"><meta http-equiv="content-type
true},e:function(){google.fl=true},b:location.hash&&location.hash!="#",bv:21,cf:"",pm:"p",pl:[],mc:0,sc:0.5
(function(){var a=google.j;window.onpopstate=function(){a.psc=1};for(var b=0,c;c=["ad","bc","inpr","is","p"
arguments])}})(c)})();if(!window.chrome)window.chrome={};window.chrome.sv=1.00;
window.google.sn="webhp";var i=window.google.timers={};window.google.startTick=function(a,b){i[a]={t:{start
</script><style id=gstyle>body{margin:0;overflow-y:scroll}#gog{padding:3px 8px 0}.gac_m td{line-height:17px
if(!window.google)window.google={};window.google.crm={};window.google.cri=0;window.clk=function(e,f,g,l,m,b
if(b&&b.substring(0,6)!="&sig2=")b="&sig2="+b;c.src=["/url?sa=T","","&cd=",a(m),h?"&authuser="+a(h):"",goog
"&sqi=2":"","&ved=",a(n),e?"&url="+a(e.replace(/#.*/,"")).replace(/\+/g,"%2B"):"","&ei=","ZxyDTuykI8rVOQWWy
(function(){try{var e=true,j=false;var m=window.gbar=window.gbar||{};function _tvn(a,b){var c=parseInt(a,10
var p={},ca={},q=[],fa=function(a,b){q.push([a,b])},ia=function(a,b){p[a]=b},ja=function(a){return a in p},
b:b};if(c)for(var d in c)a[d]=c[d];try{z(a)}catch(f){}}};n("mdc",p);n("mdi",ca);n("bnc",q);n("qGC",y);n("qm
```

Kentico

What is a request?

- Main thing you need to remember about HTTP protocol:

  HTTP is stateless protocol!

- But we need state in dynamic web applications!

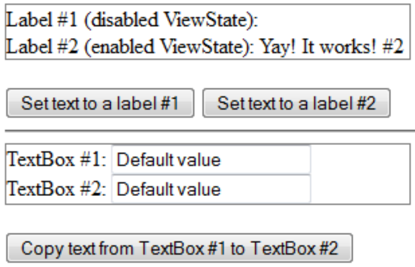
How does ASP.NET deal with stateless http?

- The answer is ... **ViewState**!

- It is a technique used by an ASP.NET Web page to persist changes to the state of a Web Form across postbacks (HTTP POST to the same page that the form is on).

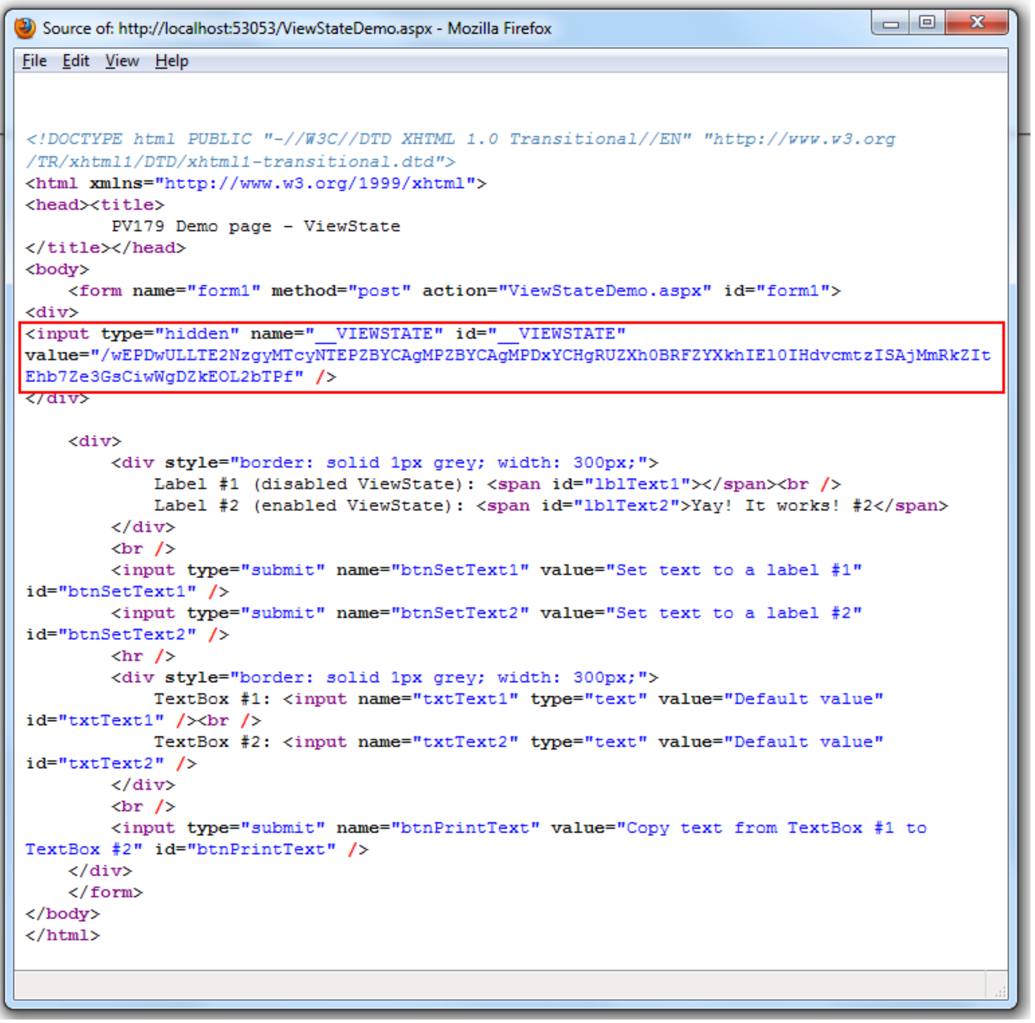
- **Use ViewState carefuly** and only when it‘s really needed! It‘s helpful technique, but it might become too greedy and can cause the application to be less effective.

ViewState – How is it send within requests?



Label #1 (disabled ViewState):
Label #2 (enabled ViewState): Yay! It works! #2

TextBox #1: Default value
TextBox #2: Default value



Source of: http://localhost:53053/ViewStateDemo.aspx - Mozilla Firefox
File Edit View Help

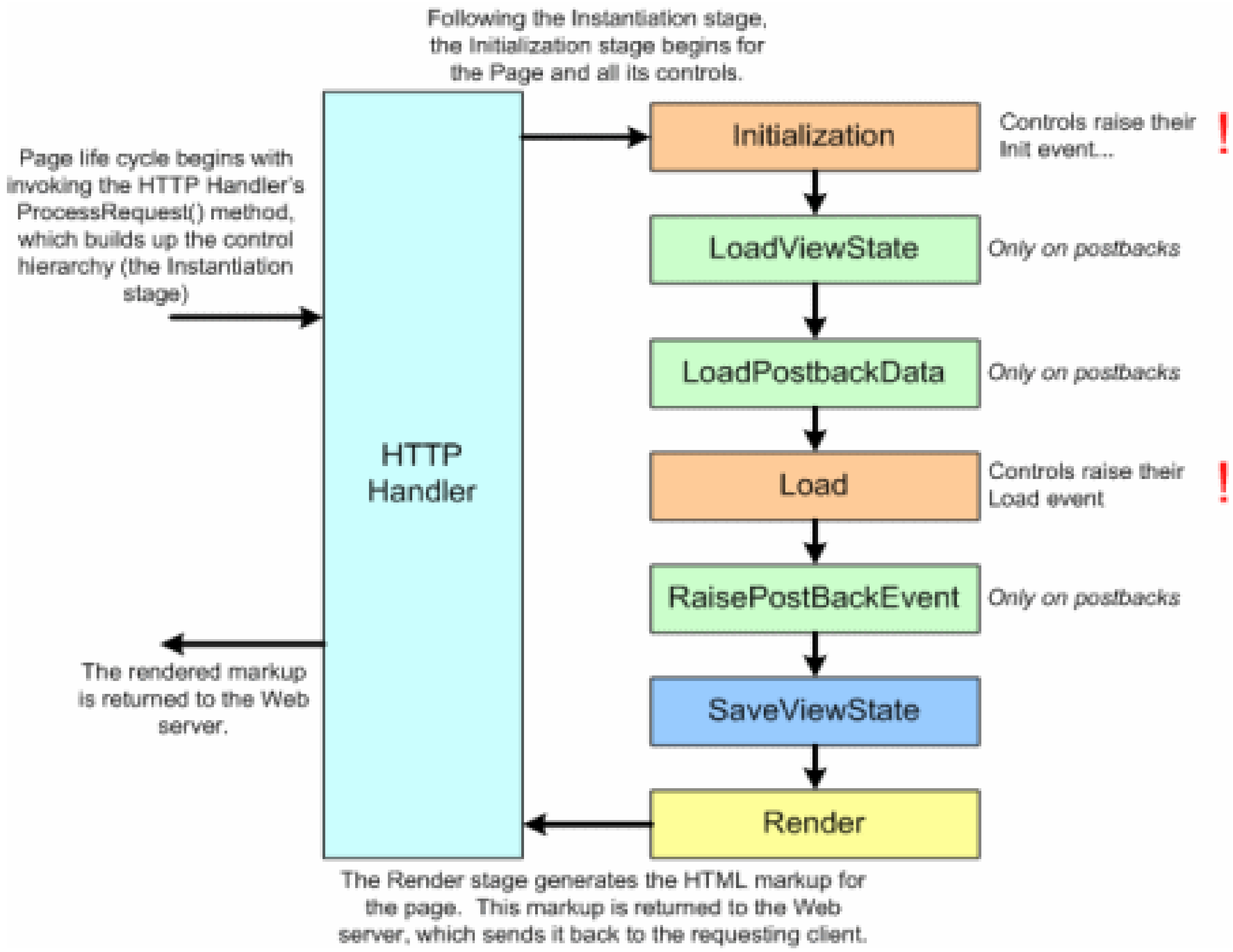
```html
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.org
/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<head><title>
        PV179 Demo page - ViewState
</title></head>
<body>
    <form name="form1" method="post" action="ViewStateDemo.aspx" id="form1">
<div>
<input type="hidden" name="__VIEWSTATE" id="__VIEWSTATE"
value="/wEPDwULLTE2NzgyMTcyNTEPZBYCAgMPZBYCAgMPDxYCHgRUZXh0BRFZYXkhIE10IHdvcmtzISAjMmRkZIt
Ehb7Ze3GsCiwWgDZkEOL2bTPf" />
</div>

    <div>
        <div style="border: solid 1px grey; width: 300px;">
            Label #1 (disabled ViewState): <span id="lblText1"></span><br />
            Label #2 (enabled ViewState): <span id="lblText2">Yay! It works! #2</span>
        </div>
        <br />
        <input type="submit" name="btnSetText1" value="Set text to a label #1"
id="btnSetText1" />
        <input type="submit" name="btnSetText2" value="Set text to a label #2"
id="btnSetText2" />
        <hr />
        <div style="border: solid 1px grey; width: 300px;">
            TextBox #1: <input name="txtText1" type="text" value="Default value"
id="txtText1" /><br />
            TextBox #2: <input name="txtText2" type="text" value="Default value"
id="txtText2" />
        </div>
        <br />
        <input type="submit" name="btnPrintText" value="Copy text from TextBox #1 to
TextBox #2" id="btnPrintText" />
    </div>
    </form>
</body>
</html>
```

ASP.NET page/control life cycle

- To be able to work with ASP.NET pages and controls properly you need to understand the life cycle of these elements.

- Most importat phases of page/control life cycle are:
  - PreInit
  - Init
  - Load
  - PreRender
  - Render

ASP.NET page/control life cycle


Following the Instantiation stage, the Initialization stage begins for the Page and all its controls.
Page life cycle begins with invoking the HTTP Handler's ProcessRequest() method, which builds up the control hierarchy (the Instantiation stage)
HTTP Handler
Initialization
Controls raise their Init event...
LoadViewState
Only on postbacks
LoadPostbackData
Only on postbacks
Load
Controls raise their Load event
RaisePostBackEvent
Only on postbacks
SaveViewState
Render
The rendered markup is returned to the Web server.
The Render stage generates the HTML markup for the page. This markup is returned to the Web server, which sends it back to the requesting client.

Source: http://i.msdn.microsoft.com/dynimg/IC152667.gif

Where to get more information?

- Where to start:
  http://msdn.microsoft.com/en-us/library/ywdtth2f%28v=vs.71%29.aspx

- More about page life cycle:
  http://msdn.microsoft.com/en-us/library/ms178472.aspx

- More details about how the ViewState works:
  http://msdn.microsoft.com/en-us/library/ms972976.aspx

Kentico

Let‘s have a REST!

- **Representational state transfer (REST)** is a style of a service architecture
- Instead of complex technologies (RPC, SOAP, etc.) you use simple HTTP requests which are supported by all the clients!
- Commonly used formates of REST responses are XML, JSON, AtomPub

JSON format

- **JavaScript Object Notation**

```
{
  "firstName": "John",
  "lastName": "Smith",
  "age": 25,
  "address": {
    "streetAddress": "21 2nd Street",
    "city": "New York",
    "state": "NY",
    "postalCode": "10021"
  }
}
```

Let‘s have a REST!

- Conforming to the REST constraints is referred to as being "**RESTful**„ service
  - Access the objects within the system defining clear structure of URLs
  - The structure of URLs should be self-explaining and self-navigating
- Kentico CMS supports RESTful service

Type of requests/responses

- POST http://localhost/KenticoCMS/rest/cms.country HTTP/1.1

  User-Agent: Fiddler
  Authorization: Basic YWRtaW5pc3RyYXRvcjo=
  Host: localhost
  Content-Type: text\xml
  Content-Length: 271

  <data><cms_country><CountryDisplayName>Test Country REST</CountryDisplayName><CountryName>TestCountryREST</CountryName></cms_country></data>

How to request a REST Service

- Create request
- Wait for response
- That‘s all!

- DEMO