Telerik Academy

ASP.NET Web Forms – Intro

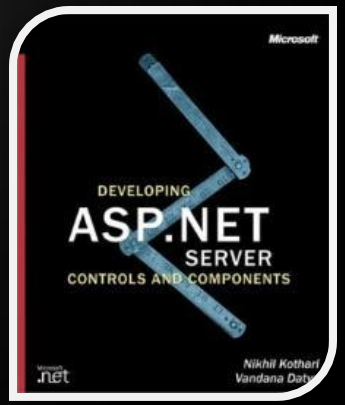
ASP.NET Web Forms

Telerik Software Academy

http://academy.telerik.com

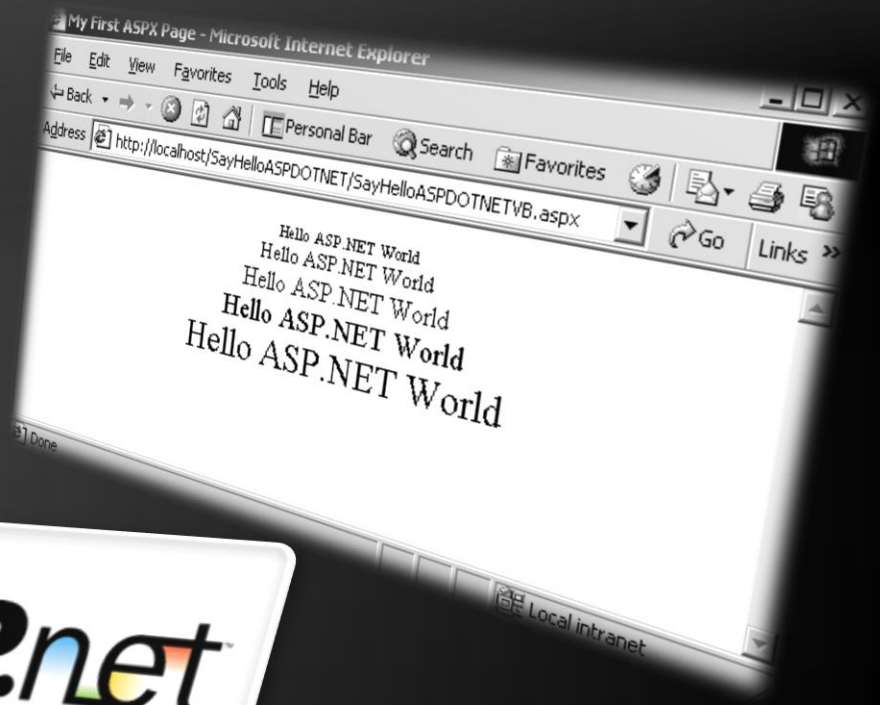
Telerik Academy

Table of Contents



Microsoft® .NET Microsoft® ASP.net 4.5

Telerik Academy



Microsoft ASP.net

Introduction to ASP.NET Web Forms

Telerik Academy

What is ASP.NET Web Forms?

- ASP.NET Web Forms
  - Web application development framework
    - Renders HTML content at the server-side
    - Combines C# with HTML for dynamic content
    - Component-based, event-driven model
  - Created by from Microsoft in 2002
  - Powerful, but complicated, very mature
  - Weak control over the output HTML
- Web Form == composition of nested controls in ASPX page

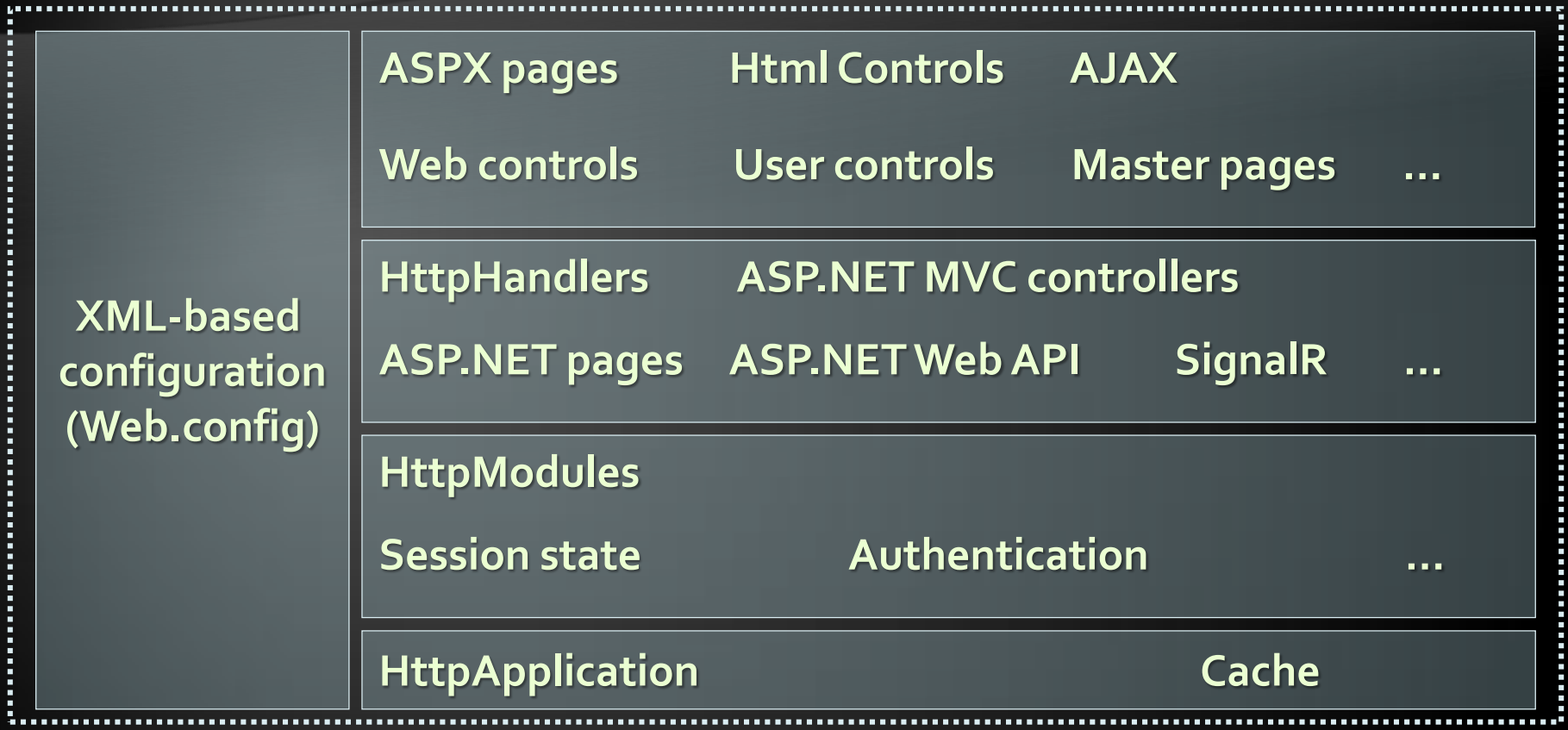
Telerik Academy

ASP.NET Web Forms Benefits

- Separate presentation from code
  - Code behind, unlike PHP (mixed code + HTML)
- Component-based development
  - Event-driven architecture
  - Object-oriented approach
- Code compilation (instead of interpreter)
- Built-in state management (session, app, …)
- Many others (data binding, validation, master pages, user controls, authentication, etc.)

Telerik Academy

ASP.NET Web Forms Architecture



Telerik Academy

HTML vs. ASP.NET Web Forms

- Traditional Web pages (static HTML)
  - Consist of static HTML, images, styles, etc.
  - Execute code on the client side (JavaScript)
  - Need lots of AJAX to become dynamic
- ASP.NET Web Forms
  - Execute code on the server and render HTML
  - Extensive use of server-side components
  - Allow easy integration with databases

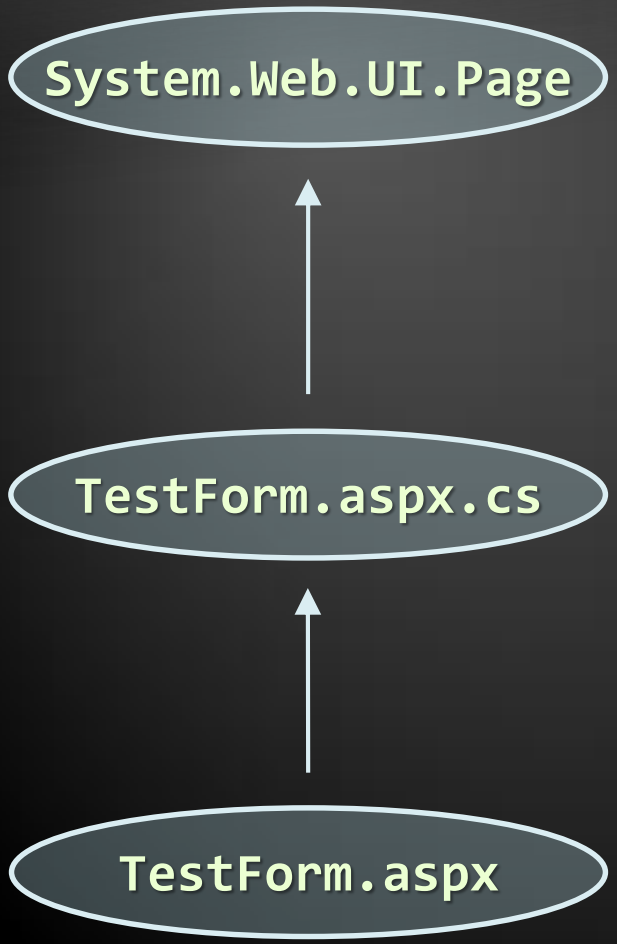
Telerik Academy

ASPX: Separate Visualization from Business Logic

- Old-fashioned Web development mixes HTML and programming code in single file (like PHP)
  - Hard to read, understand and maintain
  - Often leads to spaghetti code
- ASP.NET Web Forms splits the Web pages into two parts:
  - `.aspx` file containing HTML for visualization
  - "Code behind" files (`.aspx.cs`) containing the presentation logic for particular page

Telerik Academy

ASPX: Separate Visualization from Business Logic (2)



- Class generated from the `.aspx` file does not derive directly from `Page` class
- Derives from class defined in the "code behind", where it is easy to add methods, event handlers, etc.
- Using "code behind" separates the presentation logic from UI visualization

Telerik Academy

ASP.NET Web Forms Basic Components

Telerik Academy

Web Forms Basic Components

- Web Forms
  - XML-based files describing the Web UI
- Web Control
  - The smallest part we can use in our Web application (e.g. text box)
- "Code behind"
  - Contains the server-side C# code behind pages
- `Web.config`
  - Contains ASP.NET application configuration

Web Forms

- An ASP.NET Web Form is a programmable Web page (.aspx file)
  - Acts as a Web-based user interface (UI) of the ASP.NET Web Forms applications
  - XML-based language, like XHTML
  - Consists of HTML, C# code and controls
  - Executed at the server-side by ASP.NET
  - Rendered to HTML by the ASP.NET runtime
  - Have complex execution model (many steps)

Telerik Academy

ASP.NET Web Form – Example

```
<%@ Page Language="C#"
  Codebehind="TestWebForm.aspx.cs"
  Inherits="FirstApp.TestWebForm" %>
<!DOCTYPE html>
<html>
  <head><title>First WebForm</title></head>
    <form ID="FormTest" runat="server" …>
      <asp:Label ID="lbl" runat="server">…</asp:Label>
      <asp:TextBox ID="textCustomerName" runat="server"
        Text="Customer Name: ">…</asp:TextBox>
      <asp:Button ID="btn" runat="server" …></asp:Button>
    </form>
</html>
```

Always put ID="…" and runat="server" for the ASP.NET controls!

- The functionality of the Web form is defined by attributes: Page, Codebehind, Inherits

Telerik Academy

ASP.NET Web Controls

- ASP.NET Web controls are the smallest component part
- Deliver fast and easy component-oriented development process
- HTML abstraction, but has server-side properties and events
  - Rendered as HTML (+ CSS + scripts)

```
<asp:Button runat="server" ID="btn"
   Text="Click me!" OnClick="btn_Click" />
```

Web.config

- The configuration file for ASP.NET application
- Text based XML document
- `Web.config` defines:
  - Connection strings to any DB used by app
  - Security settings and membership settings
  - Whether debugging is allowed

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <system.web>
  </system.web>
</configuration>
```

Minimal `Web.config` should look like this

Telerik Academy

Your First ASP.NET Web Forms Application – Sumator

- Steps to create a simple ASP.NET Web application:
  1. Start Visual Studio
  2. Create "New Web Application"
  3. Add two text fields, a button and a label
  4. Handle `Button.Click` and implement logic to sum of the values from the text fields
  5. Display the results in the label

Telerik Academy

ASP.NET Sumator

Live Demo

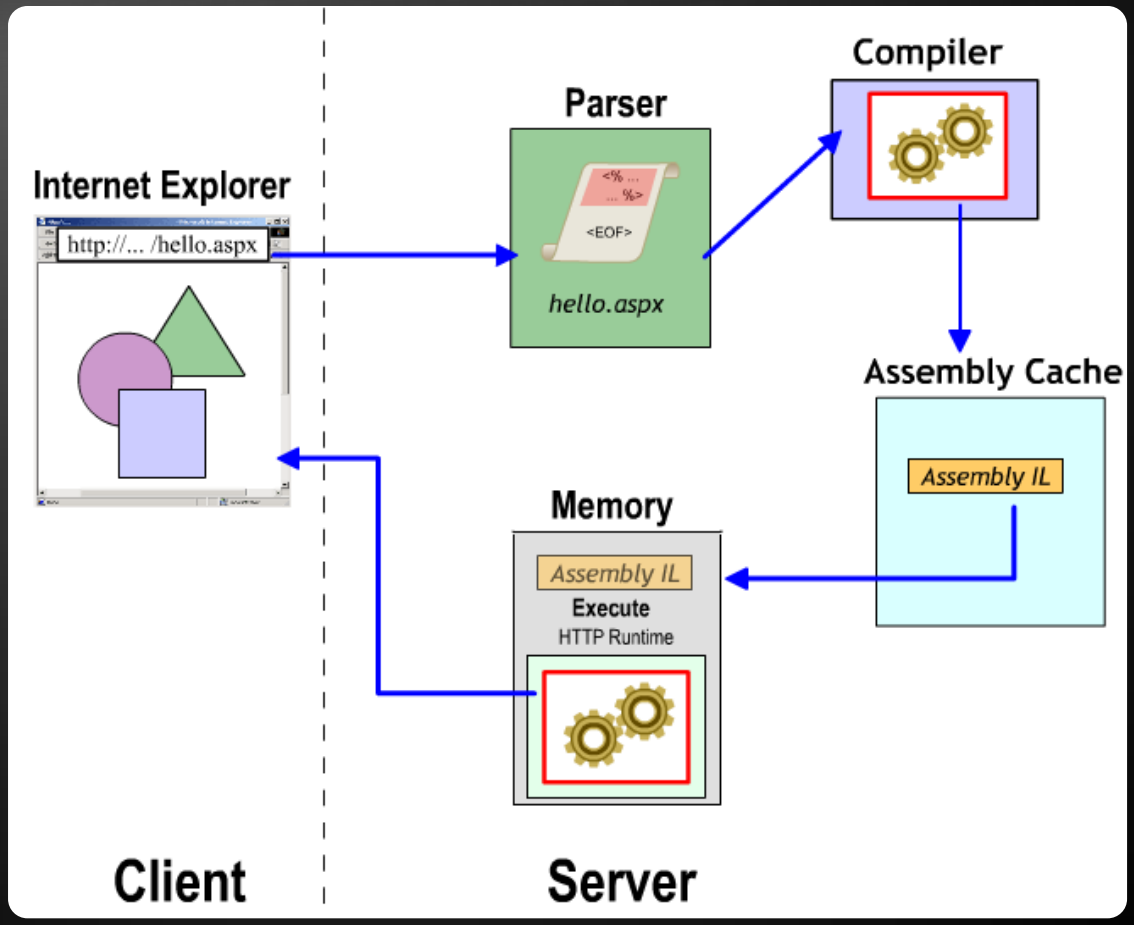
Telerik Academy

ASP.NET Execution Model

Telerik Academy

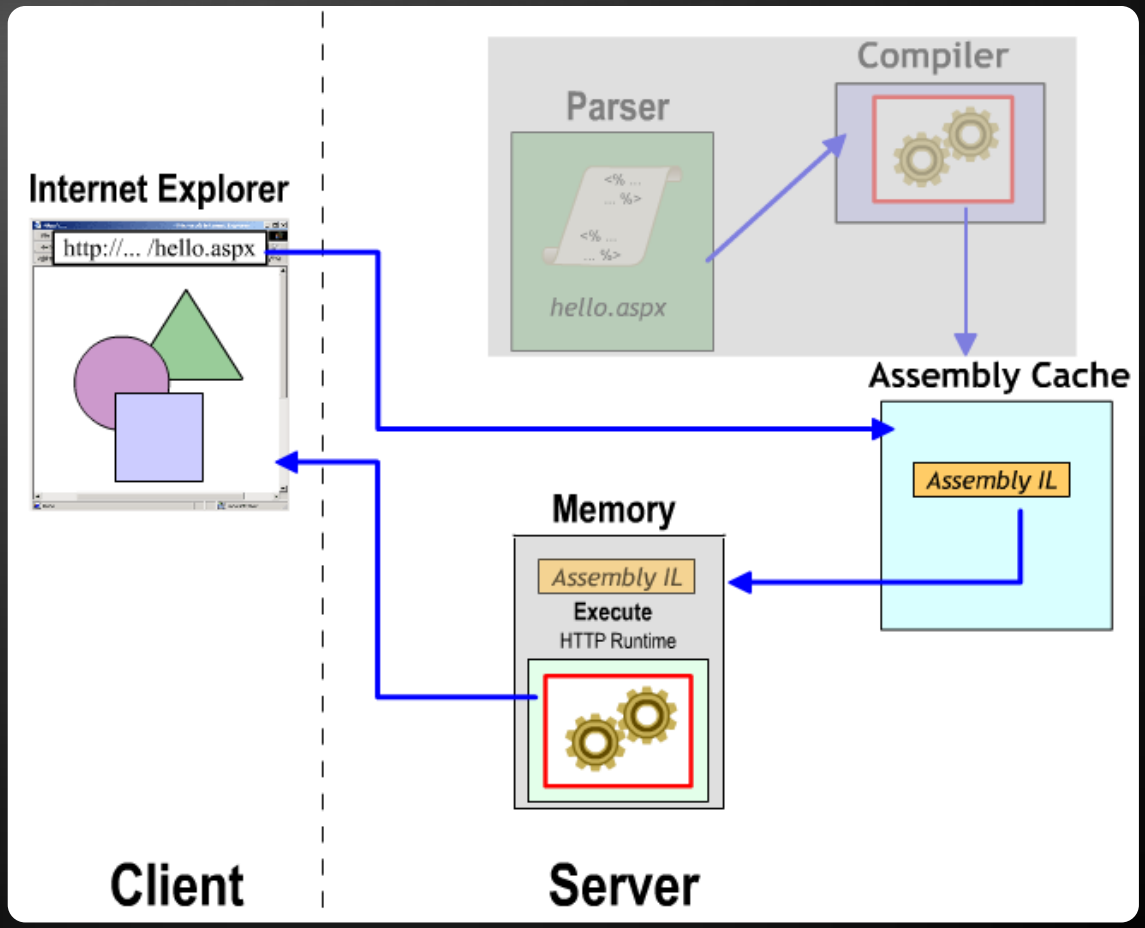
ASP.NET Execution Model

- First call to particular page



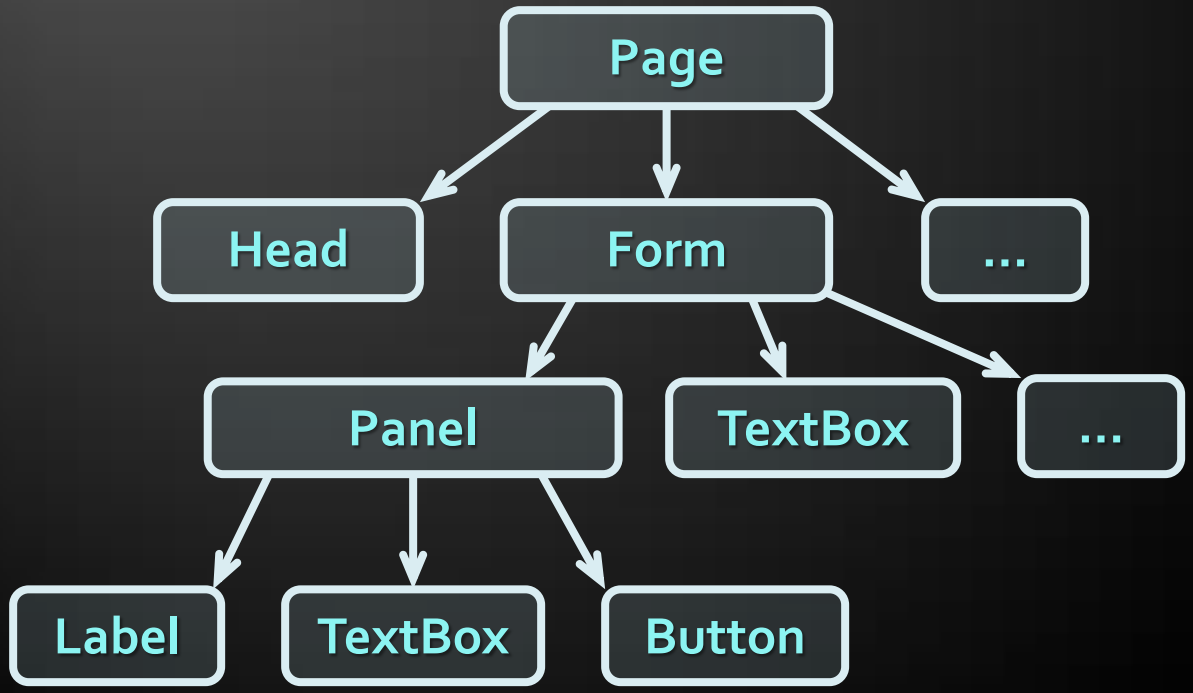
ASP.NET Execution Model (2)

- Any other call after the first



The ASP.NET Controls Tree

- When requested an ASP.NET Web form is parsed by ASP.NET and converted to a tree
  - Also called a controls tree (XML tags are turned into a tree of controls)



Telerik Academy

ASP.NET Page Lifecycle

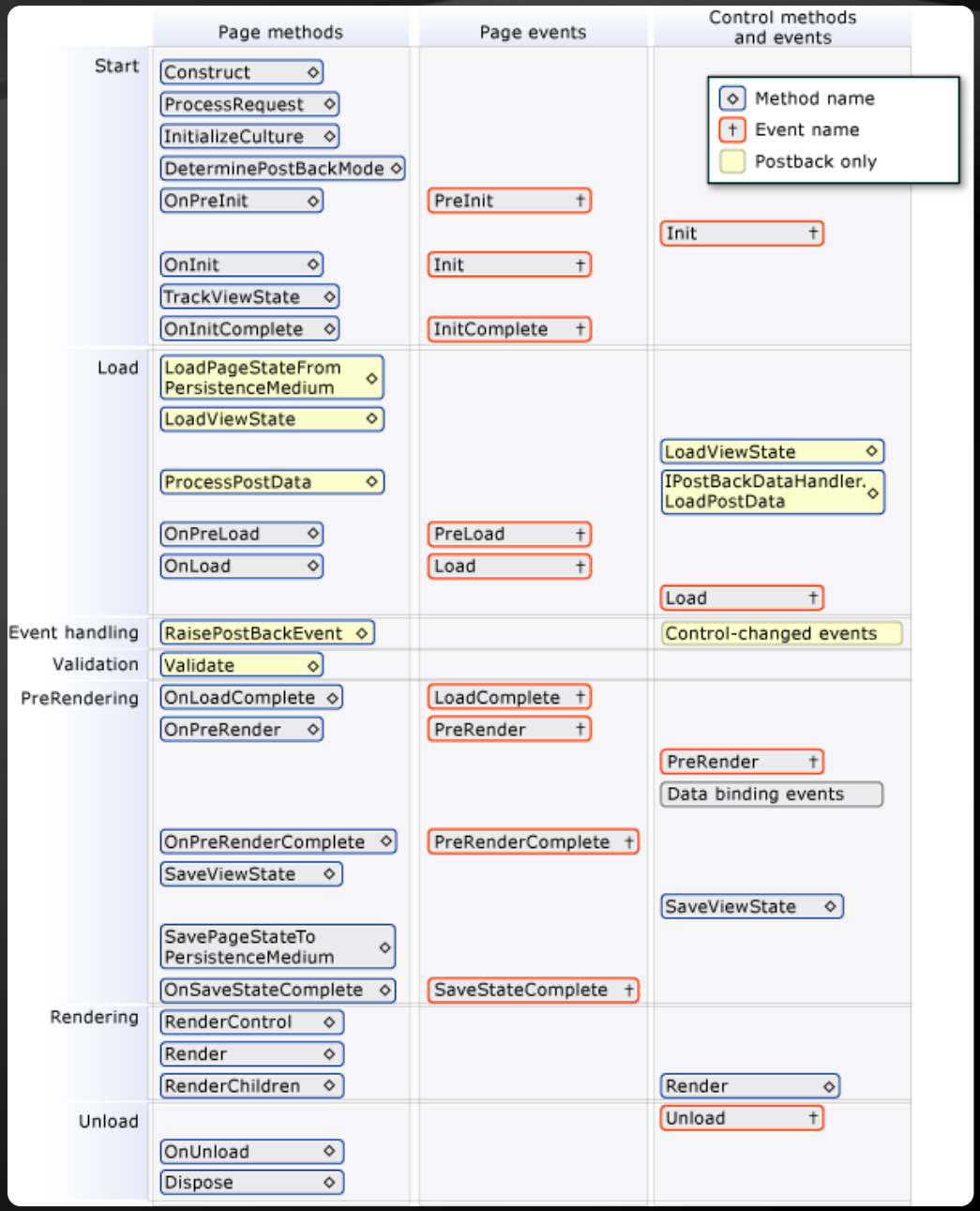
- At each request to a Web Form:
  1. The form is created (a page instance)
  2. The VIEWSTATE is restored (the state of each control, sent from the browser)
  3. The page and control events are executed
  4. The VIEWSTATE is saved in a hidden field
  5. The form is rendered as HTML
  6. The form is unloaded (destroyed)
- Important: page instances do not survive the next HTTP request!

ASP.NET Page Lifecycle Events

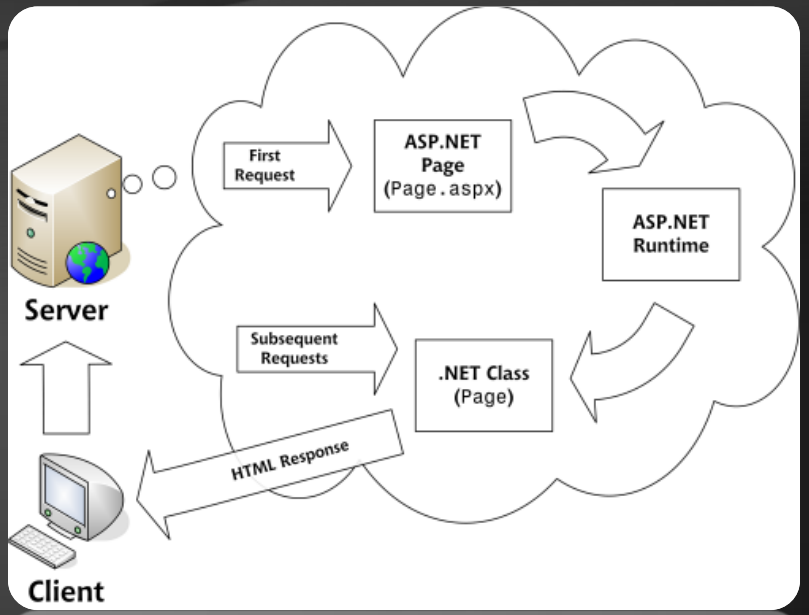
- Web forms and controls have lifecycle events:
  1. `PreInit`
  2. `Init`
  3. `Load` (load data from DB)
  4. (control events are executed here, e.g. `Click`)
  5. `PreRender`
  6. `Render`
  7. `Unload`

Telerik Academy

ASP.NET Page Lifecycle



Telerik Academy



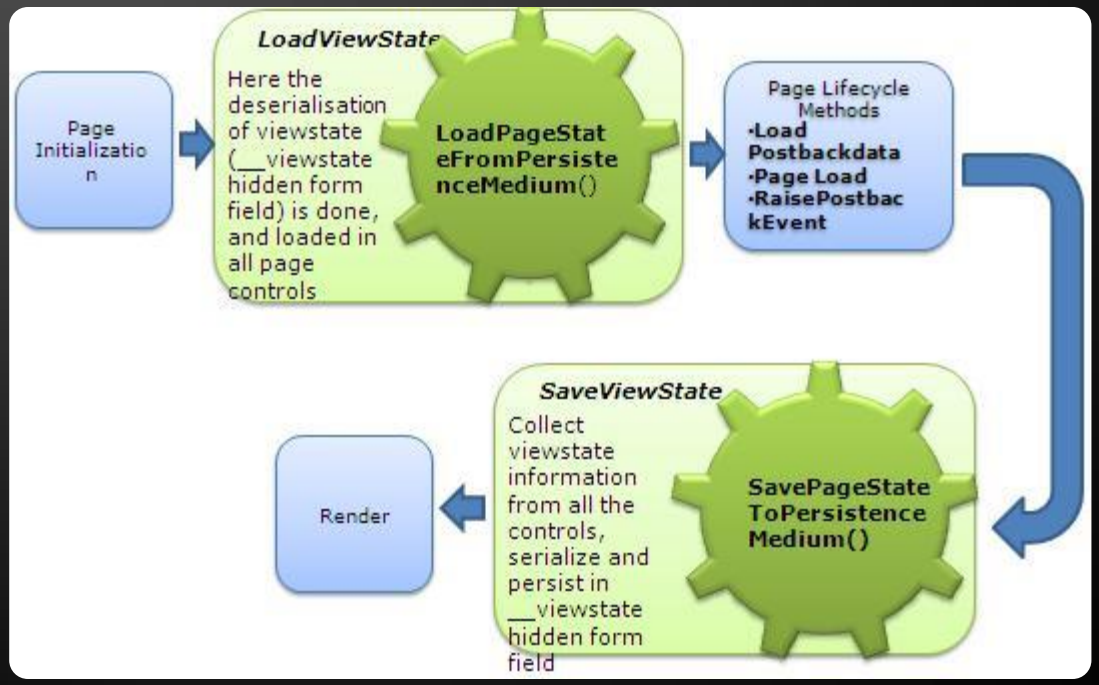
ASP.NET Page Lifecycle

Live Demo

Telerik Academy

Postbacks and VIEWSTATE

What is a Postback? What is a VIEWSTATE? How It Works?



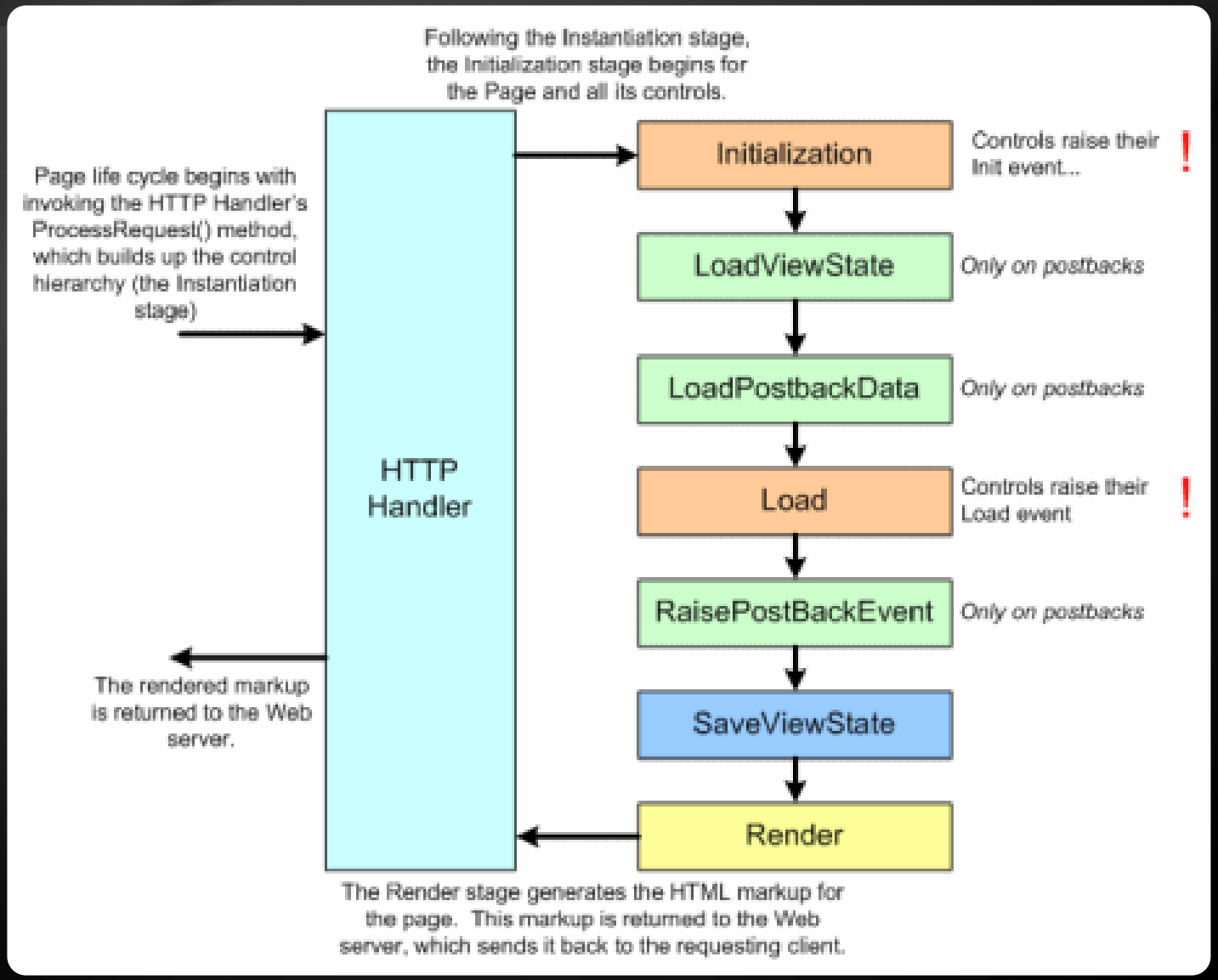
Telerik Academy

Page Postbacks and VIEWSTATE

- VIEWSTATE preserves the state of a Web control into a serialized, encrypted hidden field
- Postback in ASP.NET means submitting a Web form to the server (i.e. HTTP POST request)
  - Executed when a server-side event is raised
    - E.g. a button is clicked or a checkbox is checked
  - VIEWSTATE preserves the page and controls state
    - Almost all control properties: color, position, width, height, etc.
    - The text in the control is posted with the form

Telerik Academy

How Does VIEWSTATE Work?



Telerik Academy

VIEWSTATE

Live Demo

Telerik Academy

ASP.NET Page Directives

ASP.NET Page Directives

- Provide control over many options affecting the compilation and execution of the web form
- Important directives:
  - `@Page` – main directive of the page
  - `@Import` – imports a namespace into the page
  - `@Assembly` – attaches an assembly to the form when it is compiled
  - `@OutputCache` – controls the ability of the forms to use cache
  - `@Register` – registers a user control to be used in a web form

Telerik Academy

The @Page Directive

- Defines a form-specific (`.aspx` file) attributes used by the ASP.NET runtime
- Important attributes:
  - `AutoEventWireup` – auto-bind the controls' events to appropriate methods in the code
  - `Culture` – a culture used at page generation
  - `UICulture` – a culture for the data visualization
  - `Language` – code language (C#, VB.NET, …)
  - `Codebehind` – the code-behind file

The @Page Directive (2)

- Important attributes:
  - `Debug` – whether the page is compiled with debug symbols in it
  - `EnableSessionState` – whether a session is supported
  - `EnableViewState` – whether to use "view state" or not
  - `ErrorPage` – a page to which to redirect in case of unhandled exception

Telerik Academy

Using the @Page Directive

Live Demo

Telerik Academy

ASP.NET Web Forms – Intro

http://academy.telerik.com

Homework

1. Create a simple ASPX page that enters a name and prints "Hello" + name in the page. Use a `TextBox` + `Button` + `Label`.
2. Start Visual Studio and create new Web Forms App. Look at the files generated and tell what's purpose of each file. Explain the "code behind" model. Print "Hello, ASP.NET" from the C# code and from the aspx code. Display the current assembly location by `Assembly.GetExecutingAssembly().Location`.
3. Dump all the events in the page execution lifecycle using appropriate methods or event handlers.

Free Trainings @ Telerik Academy

- "Web Design with HTML 5, CSS 3 and JavaScript" course @ Telerik Academy
  - html5course.telerik.com

- Telerik Software Academy
  - academy.telerik.com

- Telerik Academy @ Facebook
  - facebook.com/TelerikAcademy

- Telerik Software Academy Forums
  - forums.academy.telerik.com