ASP.NET

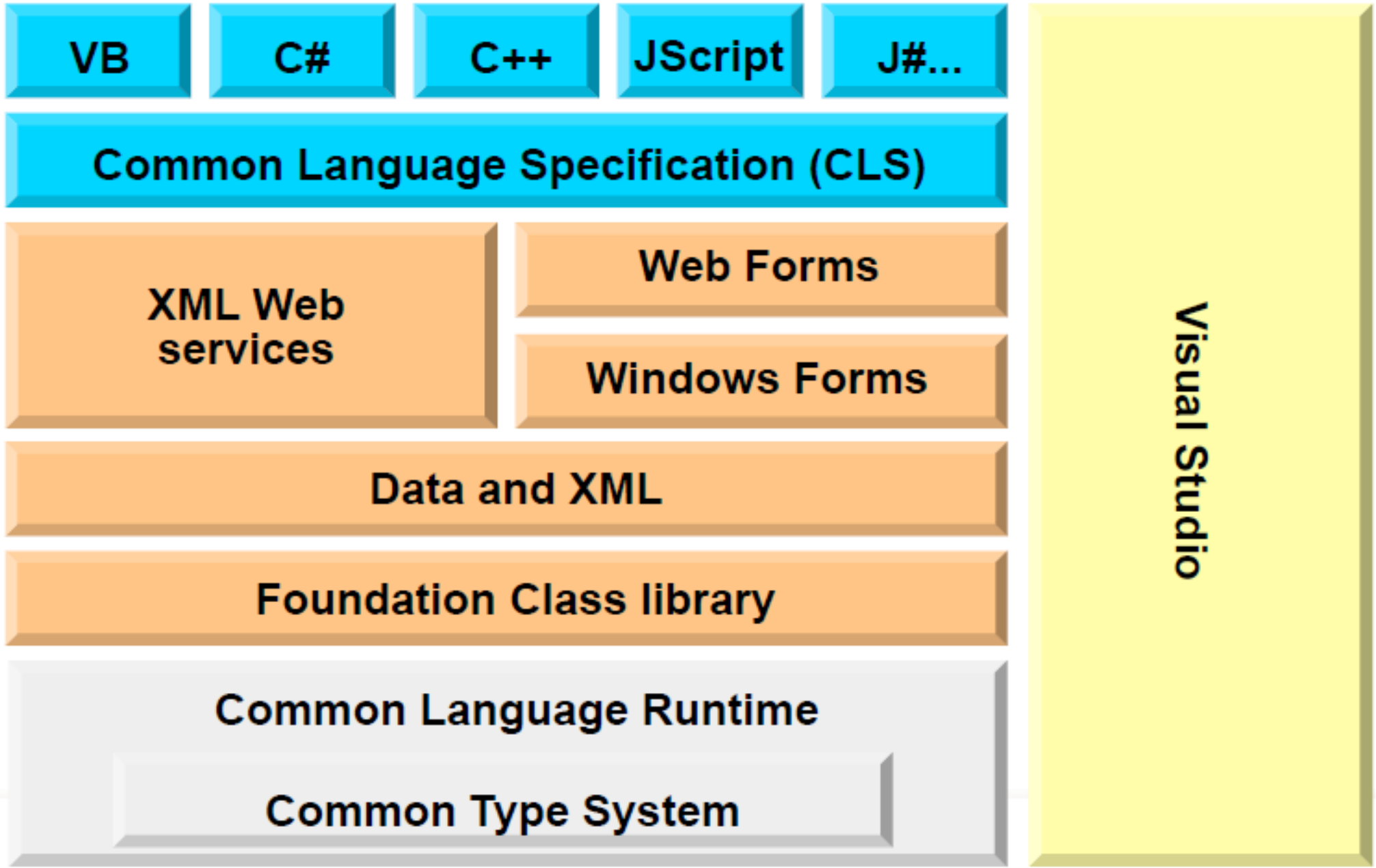
Kısa Geçmiş

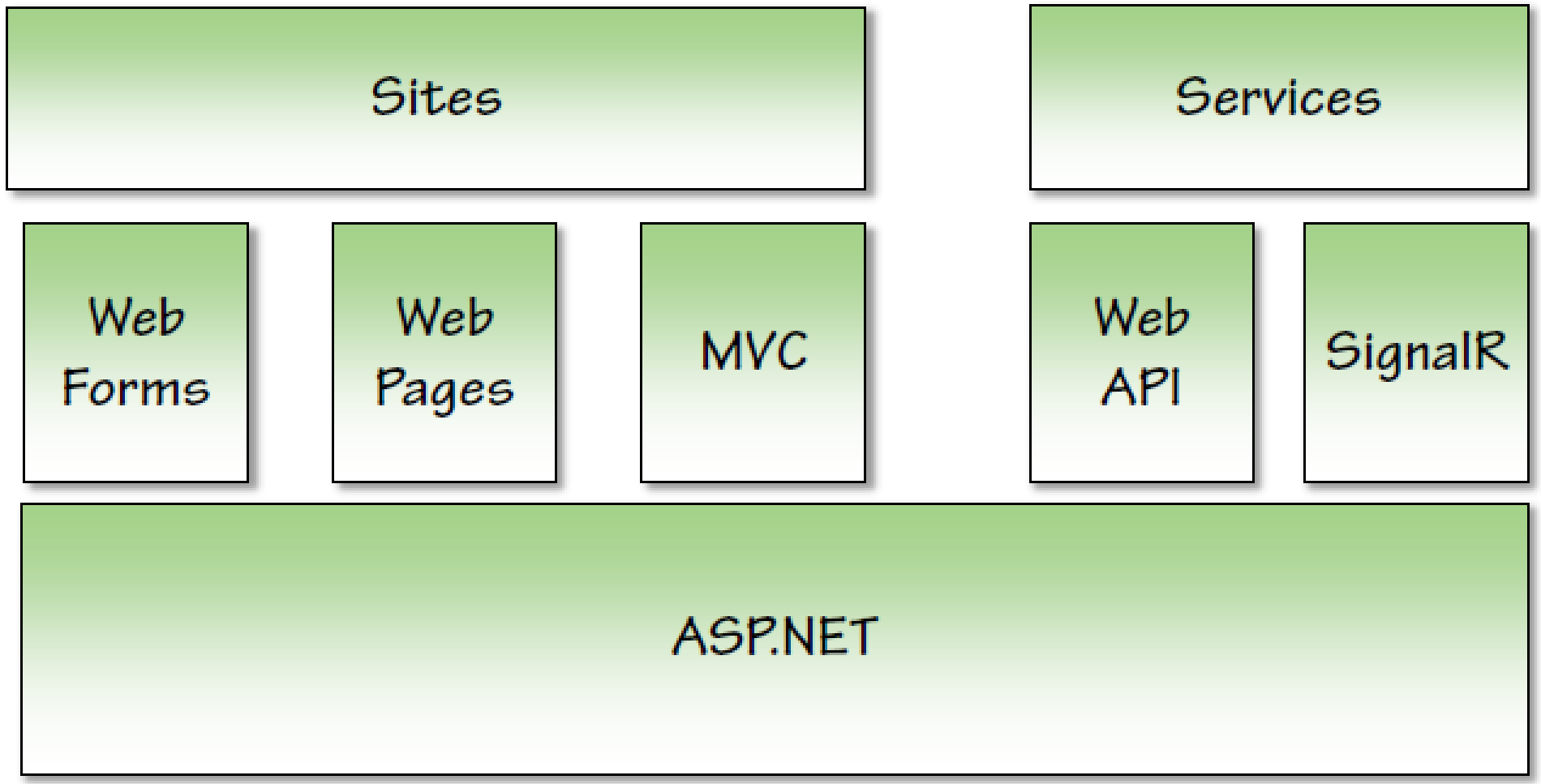
- Klasik ASP ilk defa 1990'ların sonunda tanıtılmıştır.
- ASP.NET 1.0 2002 (Web Forms)
- ASP.NET 2.0 2005 (Data)
- ASP.NET 3.5 ve ASP.NET MVC 2008
- Takip eden iki yıl içinde ikiden fazla versiyon daha yayınlandı
- MS-PL adı altında open source olarak yer aldı
- ASP.NET 4 / ASP.NET MVC 4 Visual Studio 2010 ile birlikte tanıtıldı.
  - İlk defa Razor syntaxı tanıtıldı
- ASP.NET 4.5 / ASP.NET MVC 4.5 Visual Studio 2012 ile birlikte tanıtıldı.
  - ASP.NET Web API-RESTful web servisleri gündeme geldi
- Şubat 2013'de SignalR

One ASP.NET

5 bileşen barındırmaktadır.

- **Web Forms**
- **Model View Controller (MVC)**
- **Web Pages**
- Web API
- SignalR


VB
C#
C++
JScript
J#...
Common Language Specification (CLS)
XML Web services
Web Forms
Windows Forms
Data and XML
Foundation Class library
Common Language Runtime
Common Type System
Visual Studio


Sites
Services
Web Forms
Web Pages
MVC
Web API
SignalR
ASP.NET

Bilinmesi Gerekenler

- C#
- HTML, CSS3
- Javascipt
- LINQ

- Gerekli olan yazılım
- Visual Studio 2013, 2012 veya 2010 (Express sürümleri de olabilir)
- SQL Express

ASP.NET Mimarisi

Tamamıyla .NET kütüphanesinin üzerine inşa edilmiştir.

- Tüm aspx sayfaları sınıflara parse edilip assembly'ler şeklinde derlenirler.
- Derlenen assembly'ler (w3wp.exe)'nin çalışma anındaki ihtiyacına göre yüklenirler ve çalıştırılırlar.
- Sınıflar servis isteklerine göre oluşturulurlar.

W3wp.exe IIS Worker process olup web sunusuna gelen istekleri yönetmekle sorumludur.

Dinamik İçerik

Server-side script block

Interspersed server-side script

Server-side evaluation syntax

MyPage.aspx

```
<%@ Page Language="C#" %>
<script runat="server">
const int _itemCount = 10;

string GetDisplayItem(int n) {
    return "Item #" + n.ToString();
}
</script>

<html xmlns="http://www.w3.org/1999/xhtml" >
<head><title>Simple page</title></head>
<body>
    <h1>Test ASP.NET 2.0 Page</h1>
    <ul>
    <% for (int i=0; i<_itemCount; i++) { %>
    <li><%=GetDisplayItem(i)%></li>
    <% } %>
    </ul>
    <%
    Response.Write("<h2>Total number of items = "
                   + _itemCount.ToString() + "</h2>");
    %>
</body></html>
```

Sunucu Tarafı Kontrolleri

- Bu tarz elemanlar runat="server" attribute ile belirtilirler.
- Böylelikle içiçe geçmiş sunucu tarafı scriptlere (Interspersed server-side scripts) göre daha okunabilir alternatif sunarlar.
- Sınıf tanımlamalarına member variable olarak eklenirler.
- Programsal olarak bu tarz kontrollerin nesne modelleri ile etkileşim mümkündür.
- Aynı sayfaya olan POST işlemlerinde kontrollerin durumları saklanır. Bu işlem "state retention" olarak isimlendirilir.

```
protected override void OnLoad(EventArgs e)
{
  _displayList.Items.Clear();

  for (int i=0; i<_itemCount; i++)
     _displayList.Items.Add(
              new ListItem(GetDisplayItem(i)));

    base.OnLoad(e);
}
```

```
<asp:BulletedList runat="server" ID="_displayList">
    <asp:ListItem>Sample Item 1</asp:ListItem>
    <asp:ListItem>Sample Item 2</asp:ListItem>
    <asp:ListItem>Sample Item 3</asp:ListItem>
</asp:BulletedList>
```

Code-Behind (1.1)

Asp.net 1.1, tek dosya içerisinde kod yazımına alternatif olarak sayfadan miras alma özelliğini getirmiştir.

- "Page" direktifinin "Inherits" niteliği ile bu özelliği destekler.
- Böylelikle tasarımsal görsellik ile kullanıcının yazdığı kodu birbirinden ayırır.
- Arka planda olan kodlar (code-behind) önceden derlenerek /bin klasörüne yerleştirilirler.
- Arka plandaki dosyalar aynı zamanda yine "Page" direktifinin "src" niteliği ile gösterilerek istek geldiğinde derlenebilirler.

CodeBehindV1.aspx.cs

```
namespace PS.AspDotNet {
  public class CodeBehindV1 : Page {
    protected BulletedList _displayList;
    protected HtmlGenericControl _messageH2;

    protected override void OnLoad(EventArgs e) {
      _messageH2.InnerText = "Total number of items = " +
                             _displayItemData.Length.ToString();

      ...
      base.OnLoad(e);
    }
  }
}
```

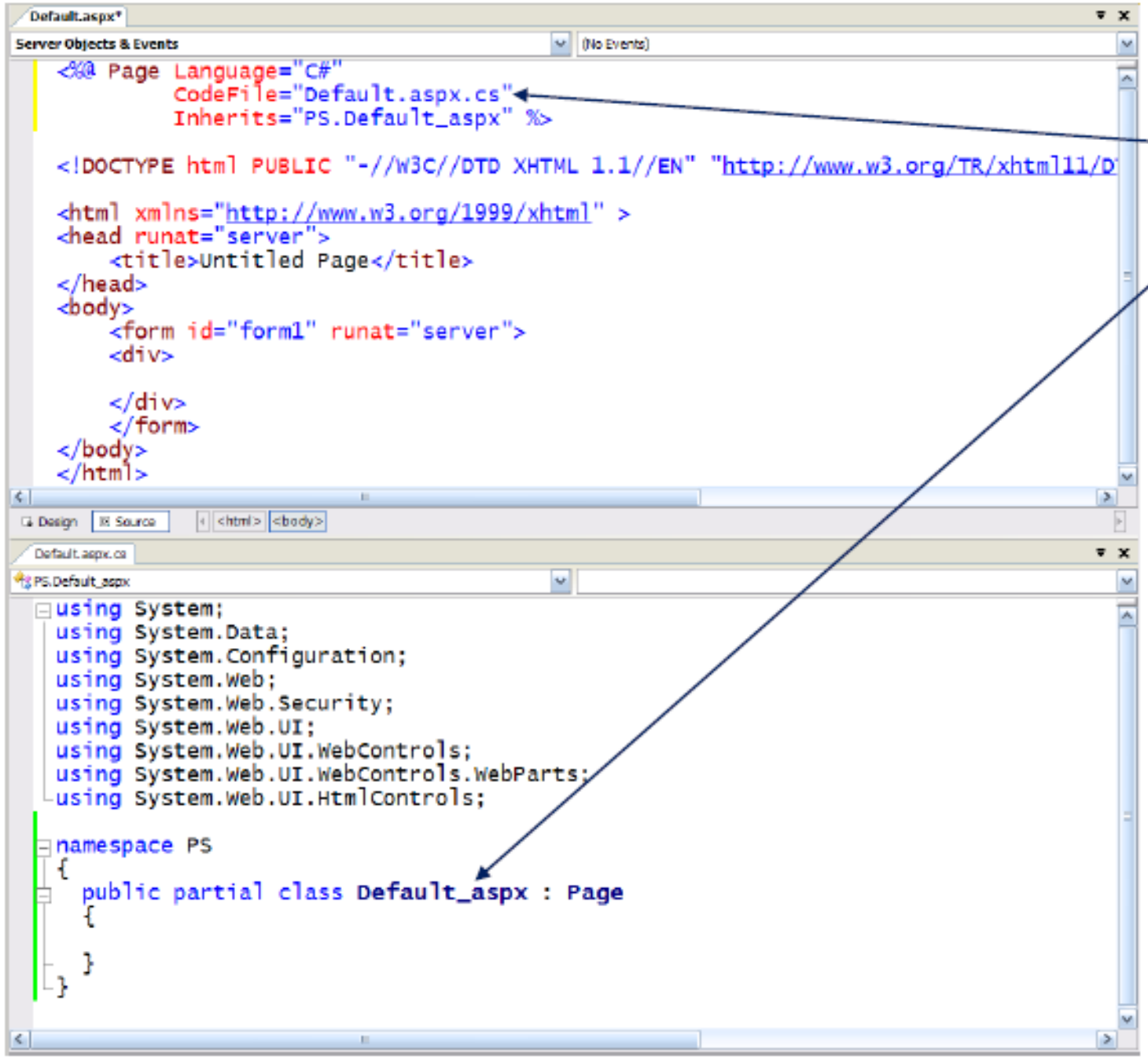
```
<%@ Page Language="C#"
         Src="CodeBehindV1.aspx.cs"
         Inherits="PS.AspDotNet.CodeBehindV1" %>

<html xmlns="http://www.w3.org/1999/xhtml" >
  <head><title>Code behind page 1.1</title></head>
  <body>
   <form id="form1" runat="server">
   ...
   </form>
  </body>
</html>
```

Code-Behind (2.0)

Asp.NET 2.0 da bu modeli destekler.

- Bu modelde yine 1.1'de olduğu gibi ön planda tasarım, deklaratif programlama ile sağlanırken, programcı .NET uyumlu herhangi bir dili kullanarak kod yazabilir.
- 1.1'e ilave olarak "partial" class kullanılır hale gelmiştir.

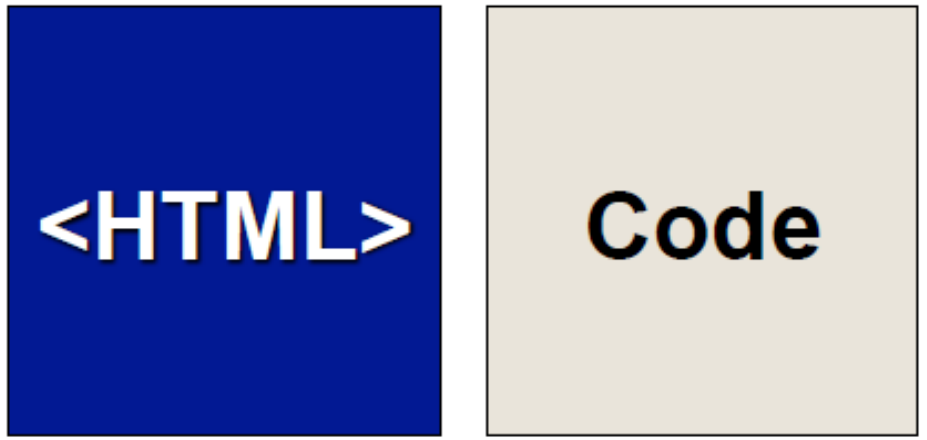


Partial class'lar

Tek Dosya

form.aspx

Ayrık Dosyalar



form.aspx

form.aspx.cs

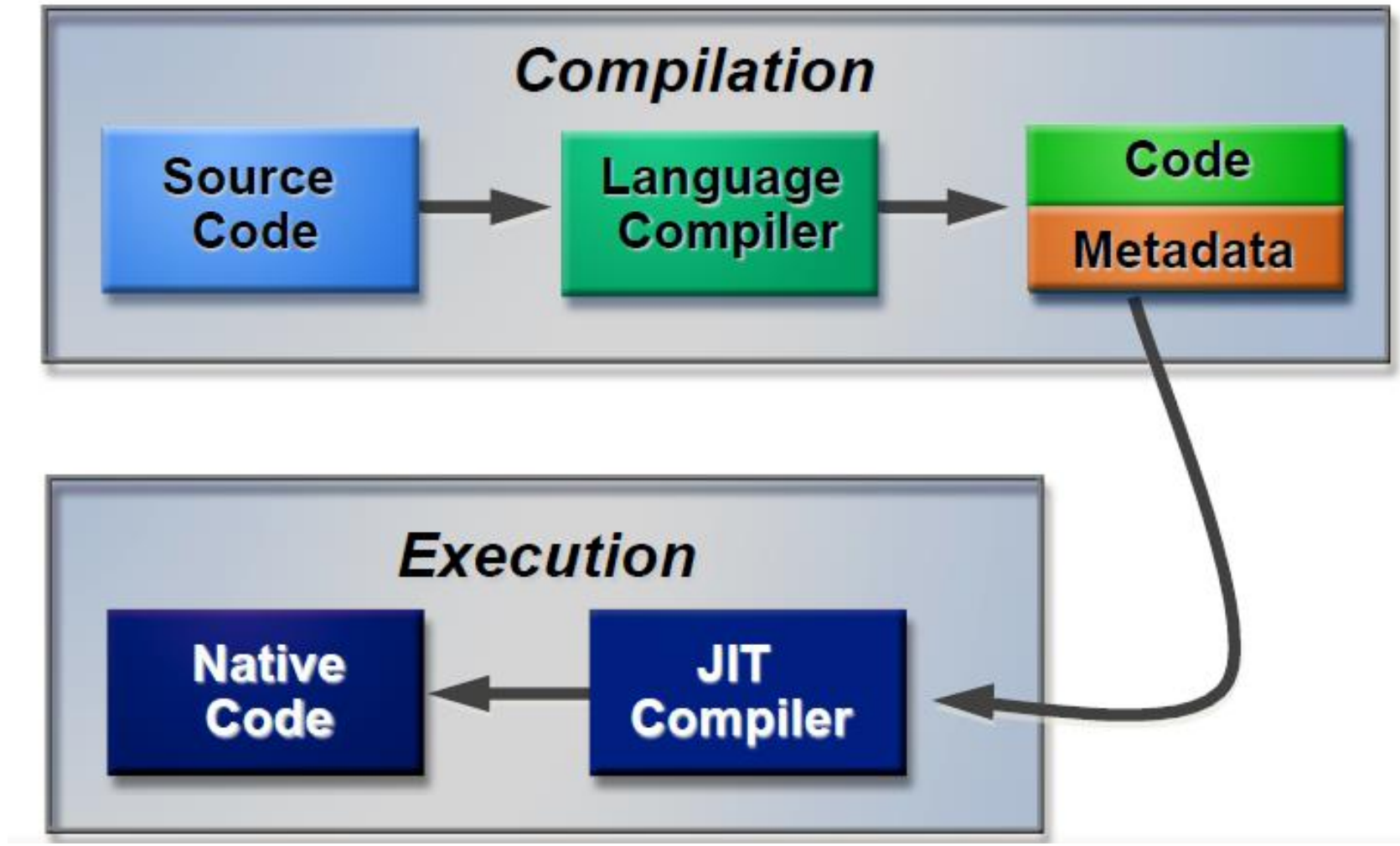
Birleşim

```
namespace PS
{
  public partial class Default_aspx
  {
    private   HtmlHead @__control2;
    protected TextBox  _name;
    protected Button   _enterButton;
    protected Label    _message;
    private   HtmlForm @__control3;
    //...
  }
}
```

```
using System;

namespace PS
{
  public partial class Default_aspx : Page
  {
    void _enterButton_Click(object sender, EventArgs e)
    {
      _message.Text = "Hi " + _name.Text;
    }
  }
}
```

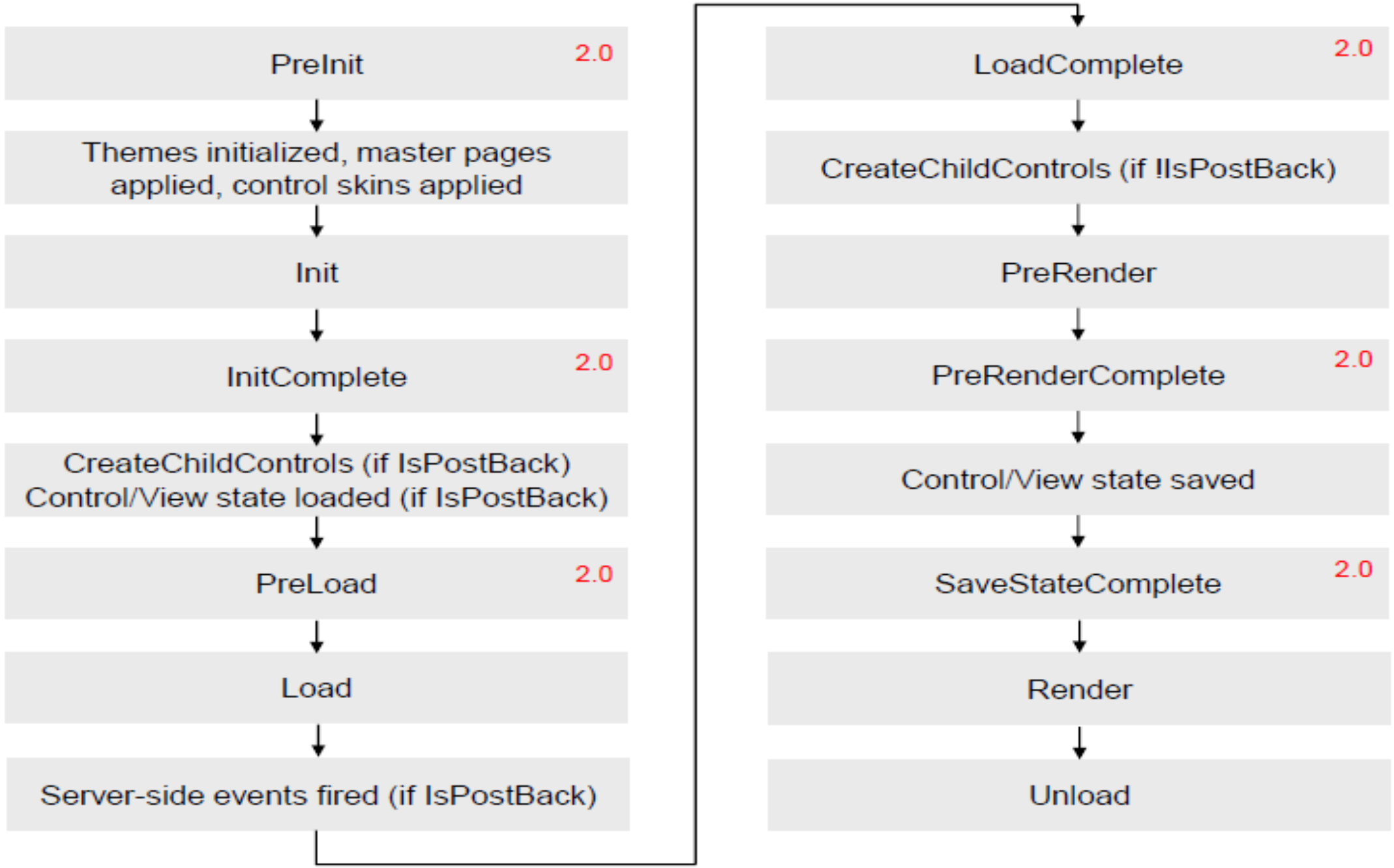
```
namespace ASP
{
    public class Default_aspx : PS.Default_aspx
    {
        //...
    }
}
```


Compilation
Source Code
Language Compiler
Code
Metadata
Execution
Native Code
JIT Compiler

Sayfa Döngüsü

Bir aspx sayfasının istemci-sunucu arasındaki iletişimini parçalar halinde incelemek mümkündür. Her bir alt parça bir olay (event) olarak isimlendirilir. Bunlar:

- PreInit
- InitComplete
- PreLoad
- LoadComplete
- PreRenderComplete
- SaveStateComplete


PreInit
2.0
Themes initialized, master pages applied, control skins applied
Init
InitComplete
2.0
CreateChildControls (if IsPostBack)
Control/View state loaded (if IsPostBack)
PreLoad
2.0
Load
Server-side events fired (if IsPostBack)
LoadComplete
2.0
CreateChildControls (if !IsPostBack)
PreRender
PreRenderComplete
2.0
Control/View state saved
SaveStateComplete
2.0
Render
Unload

Olay Yönetimi (Event Handling)

- Olay yönetimi WinForm'lara oldukça benzer şekilde gerçekleştirilir.

- Olay yönetimi için 3 farklı alternatif izlenebilir.

```
this.Load += new EventHandler(EventPage_Load);
             // or
this.Load += EventPage_Load;

void EventPage_Load(object sender, EventArgs e)
{
}
```

Açık bir şekilde delegate türündeki olaylara ekleme gerçekleştirilebilir. (Explicitly)

Virtual metotlar override edilebilir.

```
protected override void OnLoad(EventArgs e)
{

  base.OnLoad(e);
}
```

```
protected void Page_Load(object sender, EventArgs e)
{
}
```

Kapalı bir şekilde delegate türündeki olaylara ekleme gerçekleştirilir. (Implicitly)

Olay	Metot
Page_PreInit	Page.PreInit
Page_Init	Control.Init
Page_InitComplete	Page.InitComplete
Page_Load	Control.Load
Page_PreLoad	Page.PreLoad
Page_LoadComplete	Page.LoadComplete
Page_PreRenderComplete	Page.PreRenderComplete
Page_DataBind	Control.DataBinding
Page_PreRender	Control.PreRender
Page_SaveStateComplete	Page.SaveStateComplete
Page_Unload	Control.Unload
Page_Error	TemplateControl.Error
Page_AbortTransaction	TemplateControl.AbortTransaction
OnTransactionAbort	TemplateControl.AbortTransaction
Page_CommitTransaction	TemplateControl.CommitTransation
OnTransactionCommit	TemplateControl.CommitTransaction

Olaylar meydana geldiğinde ilgili metotların işletilebilmesi için ilgili öğenin "**AutoEventWireup**" niteliğinin (attribute) "**true**" olması gerekmektedir.

- Pek çok kontrol sunucu tarafındaki olaylarla ilgilenirler.
- Olaylar meydana geldiğinde çalıştırılacak olan metotlar eğer istenirse deklaratif olarak da bağlanabilirler.

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

```
protected void _clickMeButton_Click(object sender, EventArgs e)
{
}
```

Derleme

2.0 versiyonundan itibaren yeni derleme klasörleri entegre edilmiştir. Bunlar:

- App_Code
- App_Browsers
- App_GlobalResources
- App_Local Resources
- App_Themes
- App_WebReferences

Özel Klasörler

Klasör	İçeriği
/App_Browsers	Tarayıcı dosyalarını içerir. (Tarayıcının yapabilecekleri XML dosyası şeklinde tanımlanmıştır.
/App_Code	Kaynak kodları (.cs, .vb, cpp), Web servisi tanımlamalarını (wsdl files), şemaları (xsd files) içerir.
/App_Data	Veritabanı dosyaları, xml ve diğer veri kaynakları içerir.
/App_GlobalResources	Kaynak dosyaları (.resx ve .resources) içerir.
/App_LocalResources	Yerelde bir sayfa veya user controlle ilişkili kaynak dosyaları (.resx ve .resources) içerir.
/App_Themes	.skin, .css, görüntü ve diğer kaynakları içerir.
/App_Webreferences	.wsdl, .xsd içerir.
/Bin	.dll assembly dosyalarını içerir.

Projeler

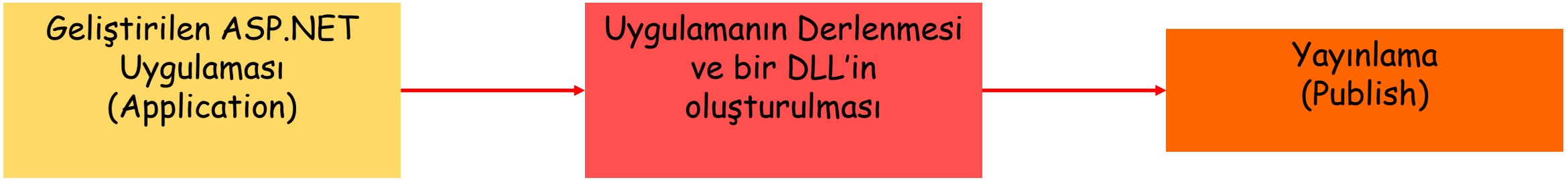
Asp.NET'de ön plana çıkan 2 tür proje bulunmaktadır. Bunlar:

- Web Sitesi Projeleri (Web Site Projects)

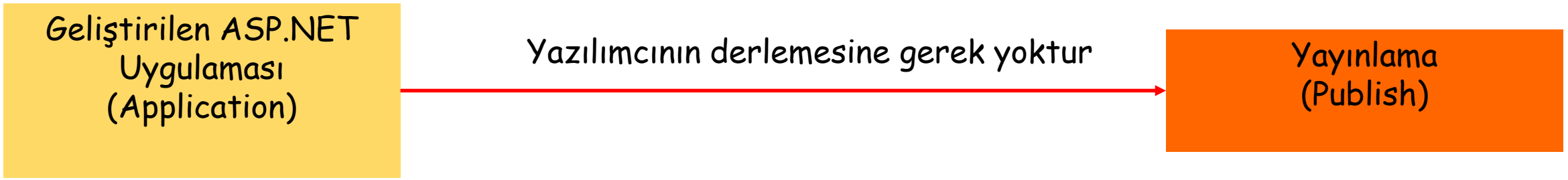
  Bu tarz projeler klasör tabanlı olup derleme işlemi arka planda implicit olarak gerçekleşir. Kullanıcının ayrıyetten derleme işlemi yapmasına gerek yoktur.
- Web Uygulama Projeleri (Web Application Projects)

  Bu proje türünde ise solution içindeki projeler derlenerek bir dll oluştururlur.

ASP.NET Web Application Derleme Modeli



ASP.NET Website Derleme Modeli



ASP.NET
Web Forms

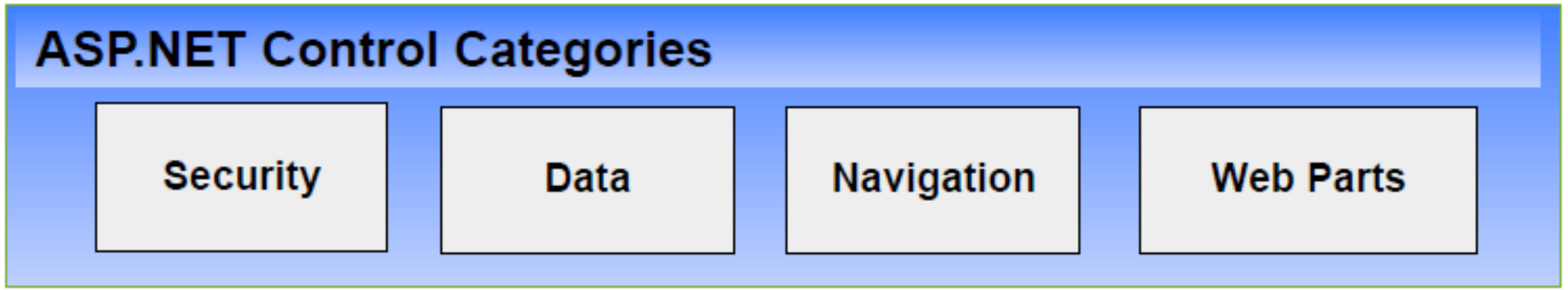
- ASP.NET, Web Form olarak isimlendirilen programlanabilir Web sayfaları sunmaktadır.

- Amaç, sunucuya anlamdırabileceği HTML bilgisini sağlamaktır.

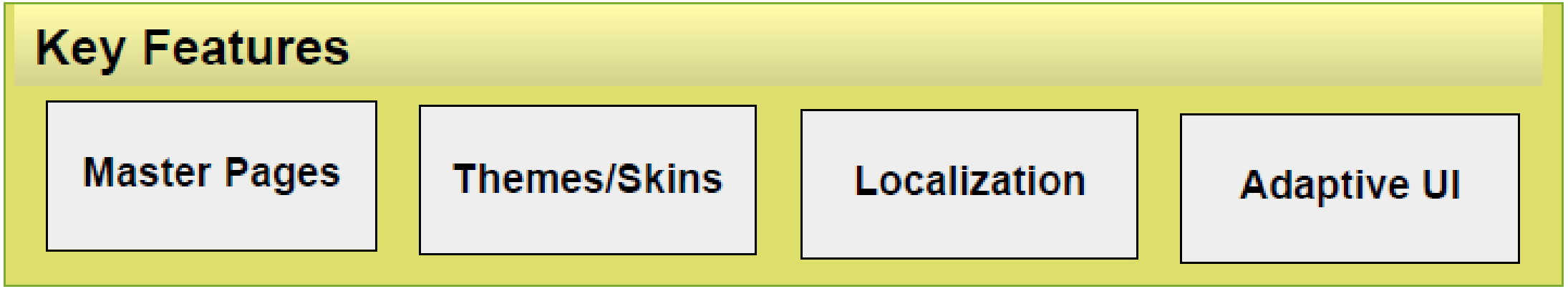
- Nesne yönelimlidir (Object -Oriented)

- aspx sayfası System.Web.UI.Page sınıfından türemiştir.

Web Form Kontrollerinin Kategorilendirilmesi



Web Form'larının Temel Özellikleri



Web Form'u nun içerisinde Neler vardır?

- Direktifler

```
<%@ Page Language="C#" AutoEventWireup="True"%>
```

- Kod Blokları

```
<script language="C#" runat="Server">….</script>
```

- Render Blokları

```
<%=UserDetails%>
```

- Sunucu Kontrolleri (Server Controls)

```
<asp:Label id="lblHelloWorld" runat="server" />
```

- Kullanıcı Kontrolleri (User Controls)

```
<acme:Header id="ucHeader" runat="server" />
```

- ASP.NET ifadeleri (Expressions)

```
<%$ ConnectionStrings: NorthwindConnString %>
```

- Veri Bağlama İfadeleri (Data Binding Expressions)

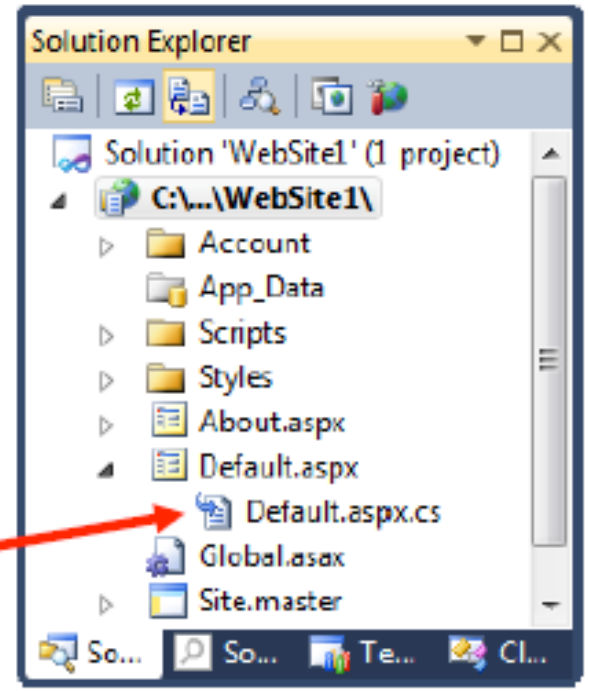
```
<%# Eval("DBFieldName") %>
```

Page Direktifi

- Page direktifi her ASP.NET sayfasının başlangıcına eklenir
  - <%@ Page Language =“C#” %>

- Temel özellikleri
  - Sayfa içerisinde kullanılacak dili belirtir.
  - Scrollbar’ın pozisyonunu korur.
  - Code-behind dosyasının yolunu belirtir.
  - İzleminin (Tracing) olup olmayacağını belirler.
  - Sayfada kullanılan (var ise) Temalar ve Master Page’leri tanımlar.
  - Hata sayfasını (Error Page) tanımlar.

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



- Page direktifinin bazı nitelikleri (attributes) de bulunmaktadır. En sık kullanılanları :

Attribute	Tanım
Async	true değer aldığında sayfa IHttpAsyncHandler arayüzünde türer ve sayfa asynchronous yenetekleri kazanır.
CodeFile	Arka planda çalışacak olan kod dosyasını belirtir.
EnableTheming	Sayfaya tema uygulanıp uygulanmayacağını belirtir.
Language	Sayfada kullanılacak dili (C# & VB) belirler
Trace	Takip (tracing) fonksiyonunun kullanılıp kullanılmayacağını belirtir.
MaintainScrollPostionOnPostBack	Post-back işlemi gerçekleştiğinde scroll position'nın durumu saklayacak bir JavaScript kodu sayfaya eklenir.
Theme	Sayfaya uygulanacak olan temayı belirler.