

## СЪДЪРЖАНИЕ

### I. Изграждане на веб приложения с ASP.NET.

**1. Въведение**

**2. Уеб форми**

**3. Контроли**

**4. Изпълним код на веб форми и контроли (code-behind)**

**5. Събития**

**6. Валидация на данни**

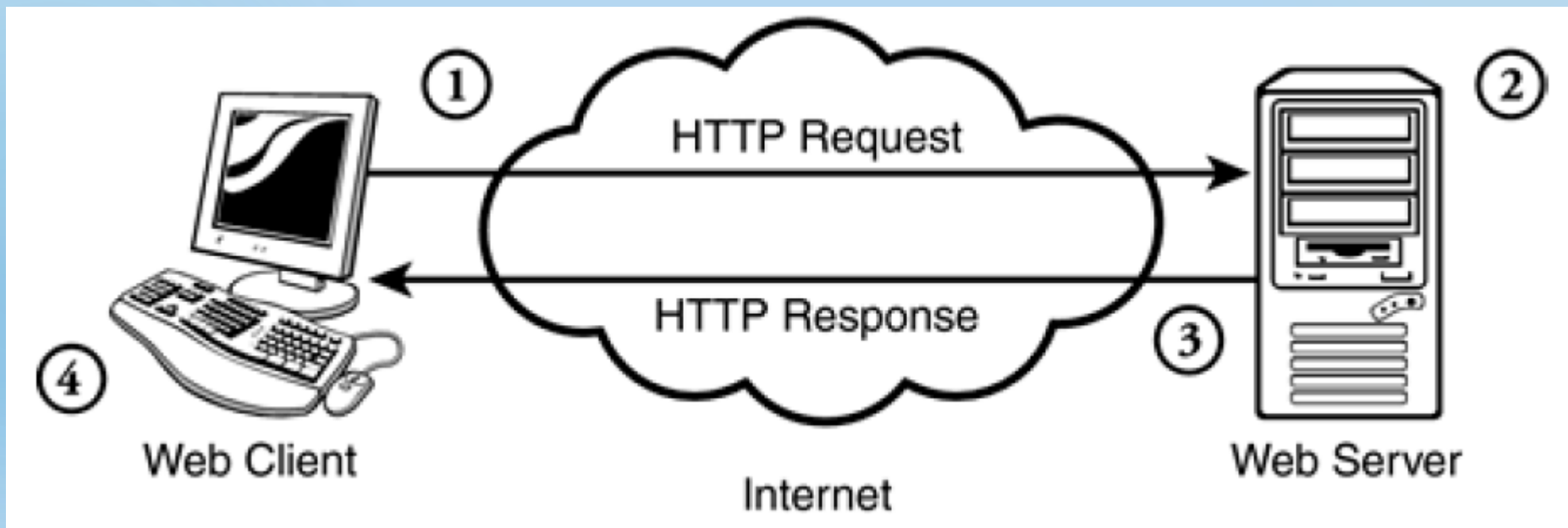
**7. Управление на състоянието**

**8. Оптимизация, конфигурация и разгръщане**

## 1. Въведение

ASP.NET е библиотека за разработка на уеб приложения и уеб услуги, стандартна част от .NET Framework. Тя дава програмен модел и съвкупност от технологии, чрез които можем да изграждаме сложни уеб приложения.

**Изпълнение на ASP.NET уеб приложение** - Уеб приложенията използват модела заявка-отговор (request-response), както е показано на фигурата:



## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

1. Потребителят въвежда в браузъра адрес на страница (URL). Браузърът изпраща HTTP заявка (request) към уеб сървър.
2. Сървърът получава заявката и я обработва. В случая с ASP.NET, IIS намира процес, който може да обработи дадената заявка.
3. Резултатът от вече обработената заявка се изпраща обратно към потребителя/клиента под формата на HTTP отговор (response).
4. Браузърът показва получения отговор като уеб страница.

### Преглед на технологията ASP.NET

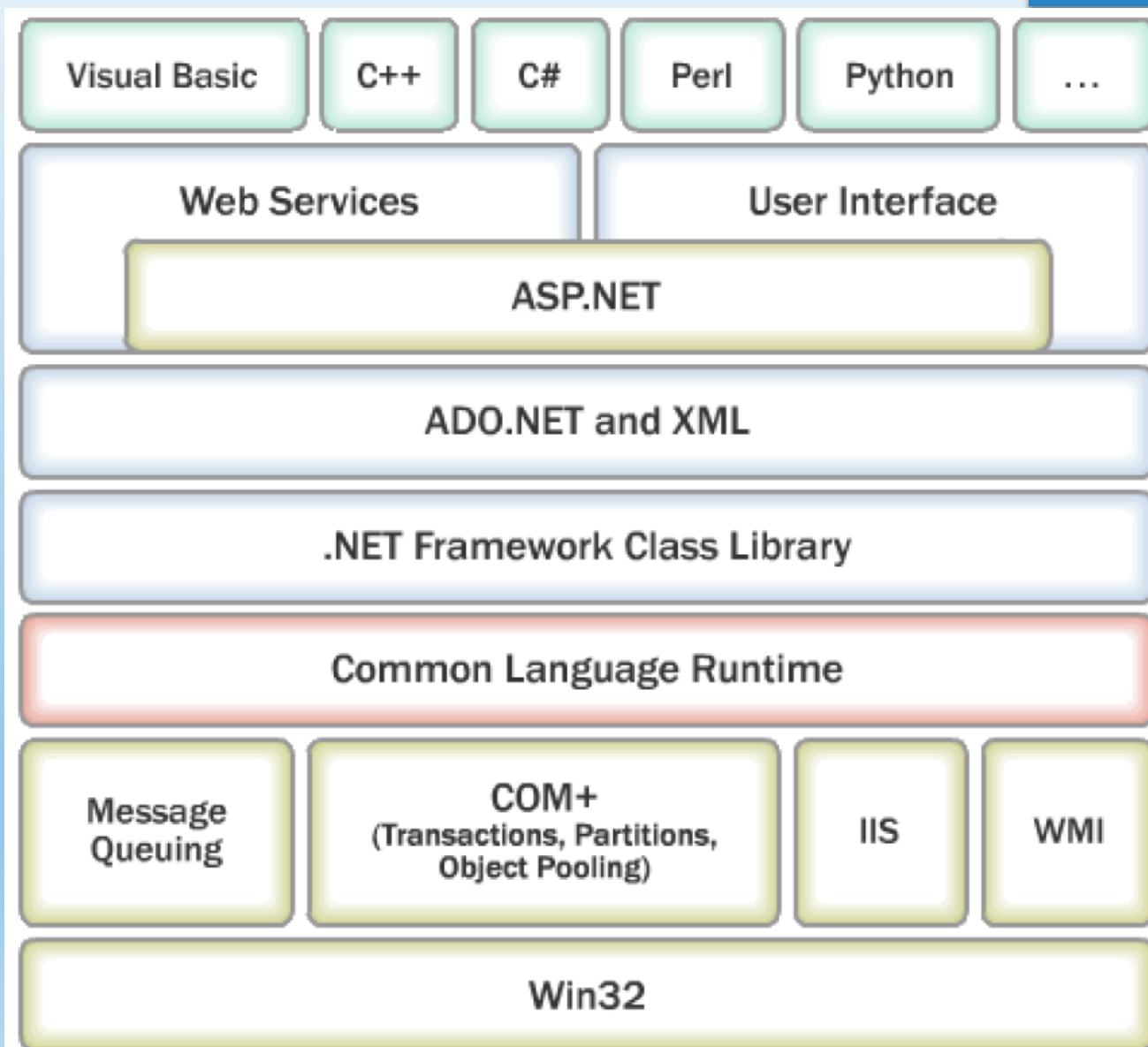
ASP.NET е програмна платформа за разработка на уеб приложения, предоставена от .NET Framework. Тя предлага съвкупност от класове, които работят съвместно, за да обслужват HTTP заявки. Също като класическите ASP (Active Server Pages), ASP.NET се изпълнява на уеб сървър и предоставя възможност за разработка на интерактивни, динамични, персонализирани уеб сайтове, както и на уеб базирани приложения. ASP.NET е също и платформа за разработка и използване на уеб услуги.

# I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

## ASP.NET и .NET Framewrok

На фигурата са показани основните компоненти на .NET Framework, част от които е библиотеката ASP.NET.



## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

### Разлики между ASP и ASP.NET

Разликите между ASP и ASP.NET са значителни. ASP.NET предлага ново ниво на абстракция за разработка на уеб приложения. Ключова характеристика на ASP.NET е възможността за разделяне на кода описващ дизайна от кода, реализиращ логиката на приложенията. ASP.NET приложенията могат да бъдат разработвани с помощта на всички езици за програмиране, които се компилират до MSIL код (C#, VB.NET, J#, Managed C++ и много други).

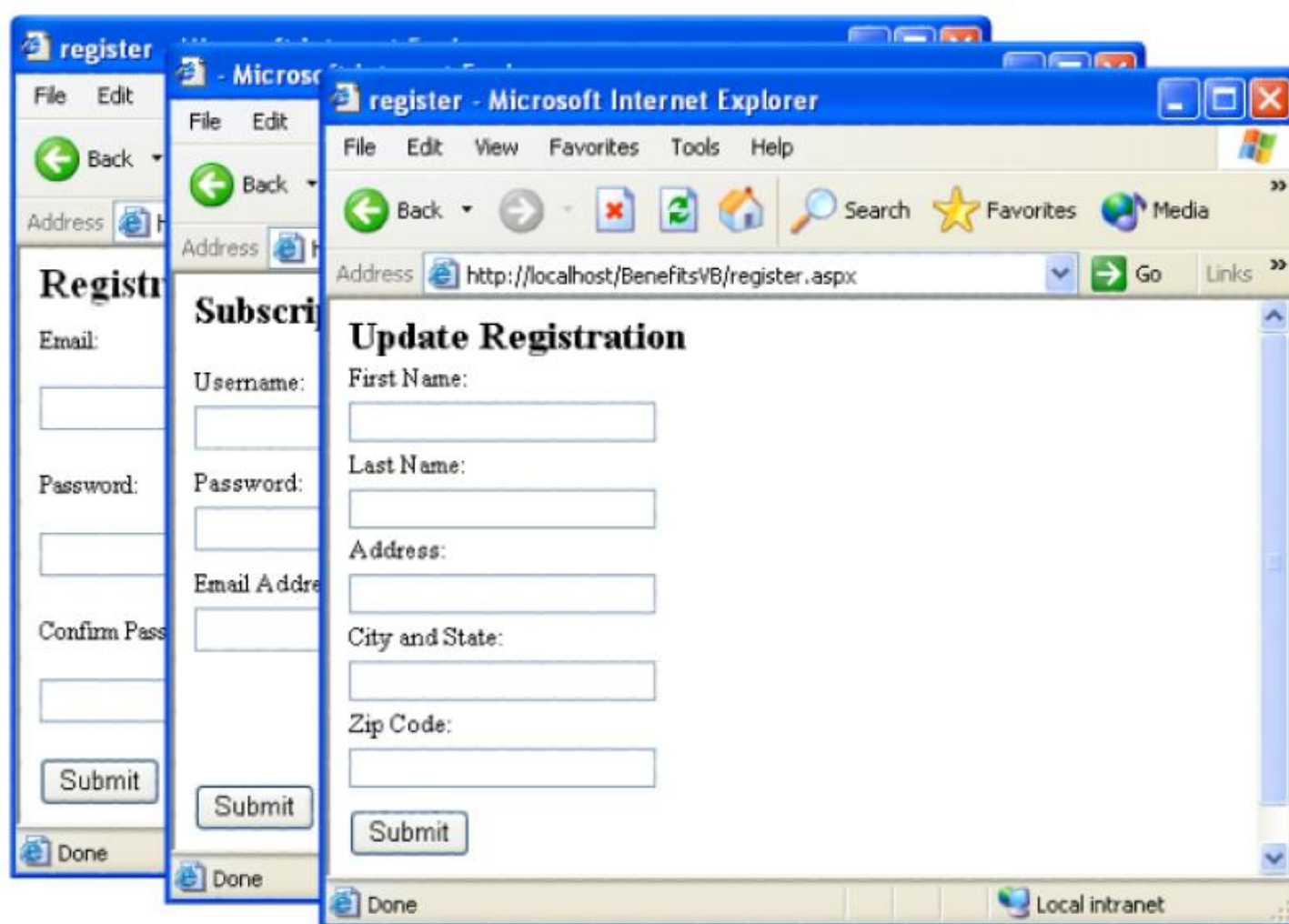
### Фундаменти на ASP.NET

**Основният компонент на ASP.NET е уеб формата** – абстракция на HTML страницата, която интернет потребителите виждат в брауъра си. Замисълът на създателите на ASP.NET е работата с уеб формите да бъде интуитивна и максимално улеснена, както е при Windows Forms формите. ASP.NET предлага едно високо ниво на абстракция, предоставяйки ни богат избор от уеб контроли, подобни на тези в Windows Forms, и намалява нуждата програмиста да работи с чист HTML код.



## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10



The image displays three overlapping web browser windows, likely Microsoft Internet Explorer, showing different pages of an ASP.NET application. The windows are titled 'register', 'register - Microsoft Internet Explorer', and 'register - Microsoft Internet Explorer'. The address bar in the top window shows 'http://localhost/BenefitsVB/register.aspx'. The forms are as follows:

- Register Form (Leftmost window):** Contains fields for 'Email:', 'Password:', and 'Confirm Pass'. A 'Submit' button is at the bottom.
- Subscription Form (Middle window):** Contains fields for 'Username:', 'Password:', and 'Email Address'. A 'Submit' button is at the bottom.
- Update Registration Form (Rightmost window):** Contains fields for 'First Name:', 'Last Name:', 'Address:', 'City and State:', and 'Zip Code'. A 'Submit' button is at the bottom.

Всяко ASP.NET приложение се изгражда от една или повече уеб форми, които могат да взаимодействат помежду си, създавайки интерактивна система.

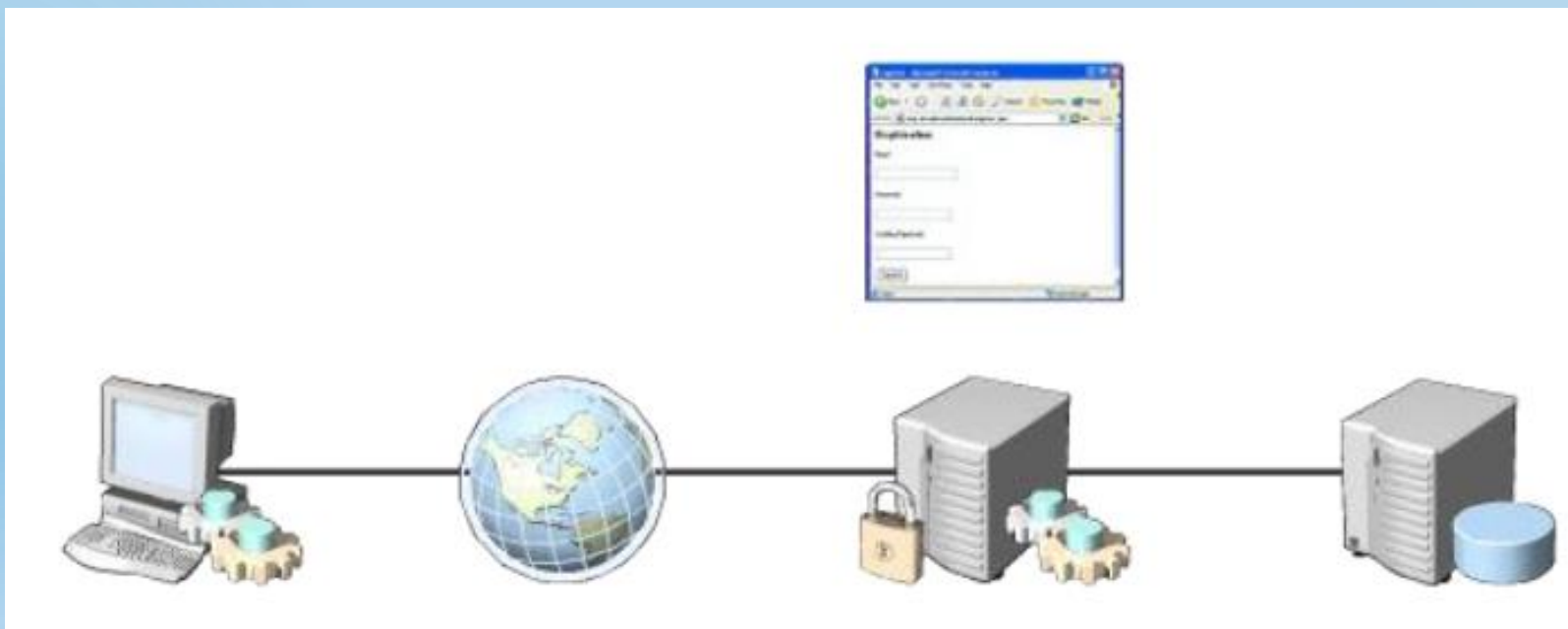
## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

### Как работи ASP.NET?

Традиционните уеб страници могат да изпълняват код на клиента, с който извършват сравнително прости операции.

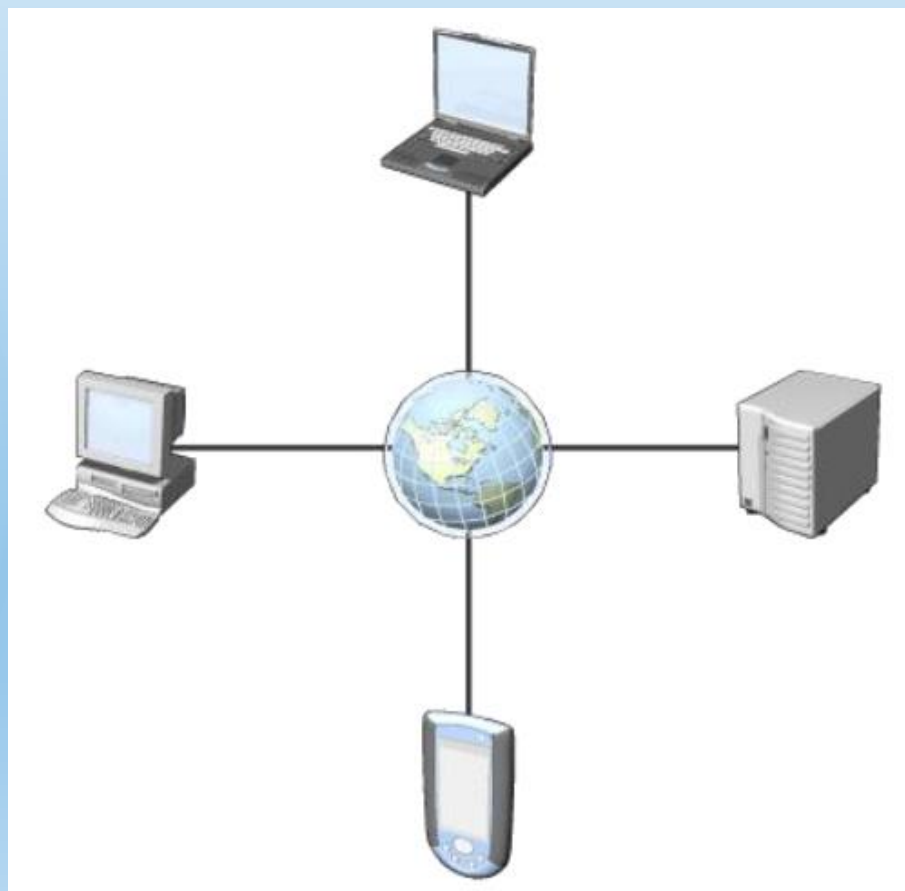
ASP.NET уеб формите могат да изпълняват и код от страна на сървъра (server-side code). С него те генерират HTML код, който да се върне като отговор на заявката. За целта могат да се извършват обработки, изискващи достъп до бази от данни и до ресурсите на самия сървър, генериращи допълнителни уеб форми и други.



## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

Всяка уеб форма в крайна сметка се трансформира в HTML код, пригоден за типа на клиентския браузър. Това позволява улеснена разработка на уеб форми. Те работят практически върху всяко устройство, което разполага с интернет свързаност и уеб браузър.





### **Разделяне на визуализация от бизнес логика**

Една от основните архитектурни цели на ASP.NET е разделянето на визуалната част от бизнес логиката. Тъй като реализацията на потребителския интерфейс и на бизнес логика са до голяма степен, две независими задачи, ASP.NET предоставя модел за разработка, при който те са физически разделени в отделни файлове.

### **Програмирането за клиентския интерфейс (UI) се разделя на две части:**

- За визуализация се използва HTML-подобен код, записан във файл с разширение .aspx.
- Бизнес логиката се дефинира в отделен файл (с разширение .cs за C# или .vb за Visual Basic .NET), съдържащ конкретната имплементация на определен програмен език.

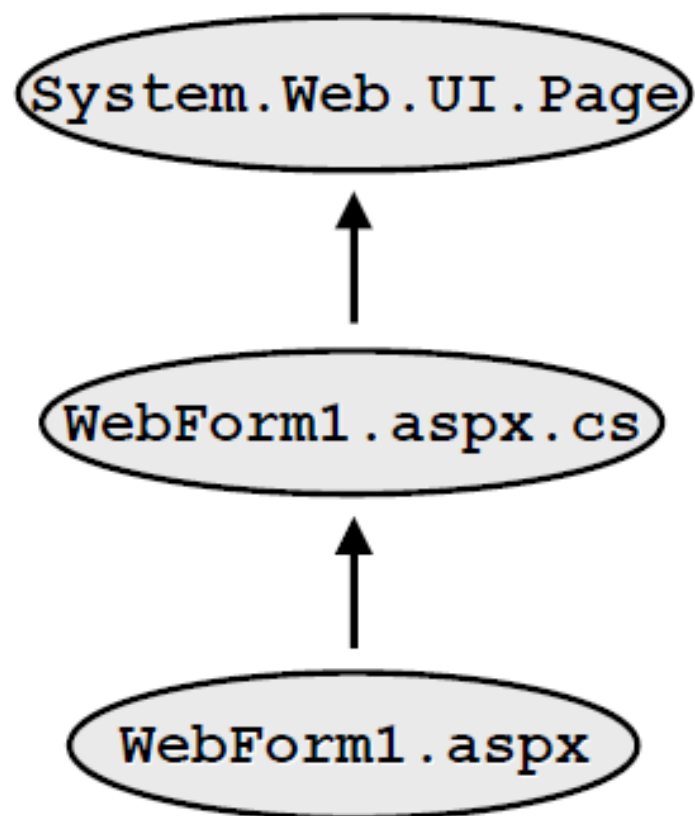
Файлт, съдържащ бизнес логиката, се нарича "Изпълним код на уеб формата" (Code-behind).

Зад всяка уеб форма стои богатият обектен модел на .NET Framework и тя се компилира до клас в асемблито на проекта ни.

## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

Класът, генериран от .aspx файл, се непряк наследник на Page класа. Съществува междинен клас в йерархията, който е за изпълнимия код (code-behind class). В него можем лесно да добавяме методи, обработка на събития и др.




Чрез "изпълнимия код" представянето е разделено от логиката. Това улеснява значително поддръжката на .aspx страниците.


## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET


10


### Компоненти на ASP.NET


Ето кои са основните компоненти, от които се изграждат уеб приложенията, базирани на ASP.NET:

 `Web Forms` – описват интерфейса за ASP.NET приложение.

 `Code-behind` класове – асоциират се с уеб форми и контроли и съдържат server-side код.

 `Web.config` – файл, съдържащ конфигурацията на ASP.NET приложението.

 `Machine.config` – файл с глобални настройки за уеб сървъра.

 `Global.asax` – файл, съдържащ код за прихващане на събития на ниво приложение (application level events).

## I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

### Пример за уеб приложение изградено чрез използване на Visual Studio

Ще създадем примерно уеб приложение чрез Visual Studio.NET, за да разгледаме структурата на директориите и файловете му. Отваряме Visual Studio.NET и създаваме нов уеб проект с име **Demo-1-ExampleOfWebApplication**:

Create a new project

Search for templates (Alt+S) 🔍 [Clear all](#)

Recent project templates

- Console App (.NET Framework) C#
- Windows Forms App (.NET Framework) C#

C# Windows Web

**ASP.NET Web Application (.NET Framework)**  
Project templates for creating ASP.NET applications. You can create ASP.NET Web Forms, MVC, or Web API applications and add many other features in ASP.NET.

C# Windows Cloud Web

Project name

Demo-1-ExampleOfWebApplication

## Create a new ASP.NET Web Application



### Empty

An empty project template for creating ASP.NET applications. This template does not have any content in it.



### Web Forms

A project template for creating ASP.NET Web Forms applications. ASP.NET Web Forms lets you build dynamic websites using a familiar drag-and-drop, event-driven model. A design surface and hundreds of controls and components let you rapidly build sophisticated, powerful UI-driven sites with data access.



### MVC

A project template for creating ASP.NET MVC applications. ASP.NET MVC allows you to build applications using the Model-View-Controller architecture. ASP.NET MVC includes many features that enable fast, test-driven development for creating applications that use the latest standards.



### Web API

A project template for creating RESTful HTTP services that can reach a broad range of clients including browsers and mobile devices.



### Single Page Application

A project template for creating rich client side JavaScript driven HTML5 applications using ASP.NET Web API. Single Page Applications provide a rich user experience which includes client-side interactions using HTML5, CSS3, and JavaScript.

### Authentication

No Authentication

[Change](#)

### Add folders & core references

- ☐ Web Forms
- ☐ MVC
- ☐ Web API

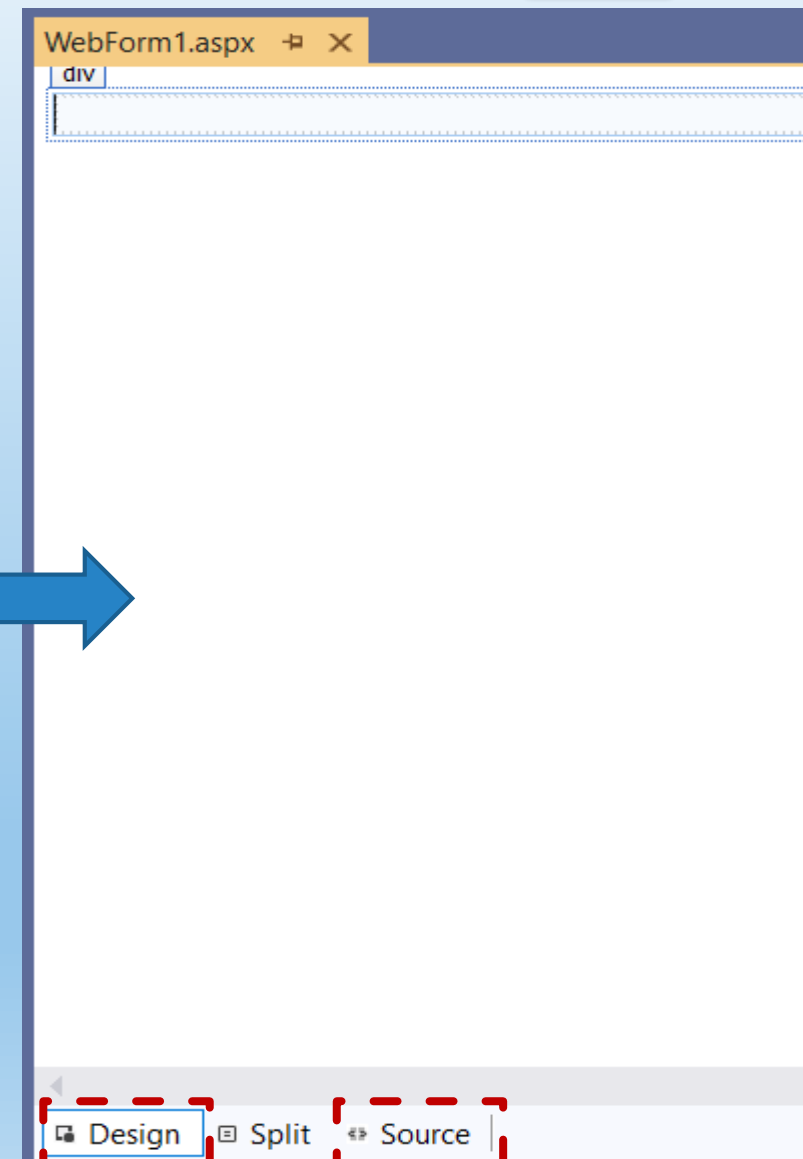
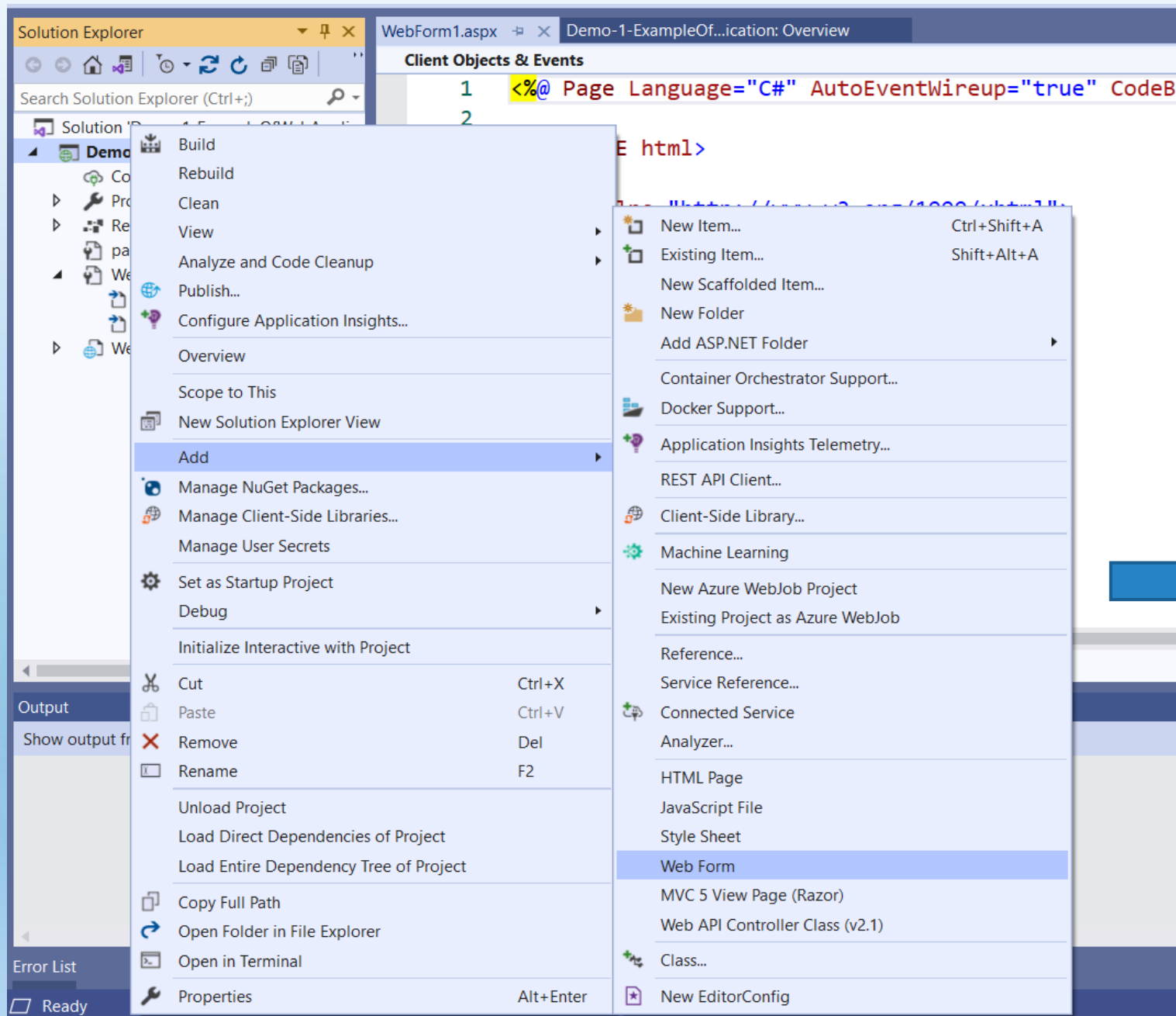
### Advanced

- ☒ Configure for HTTPS
- ☐ Docker support  
(Requires [Docker Desktop](#))
- ☐ Also create a project for unit tests

[Demo-1-ExampleOfWebApplication](#)



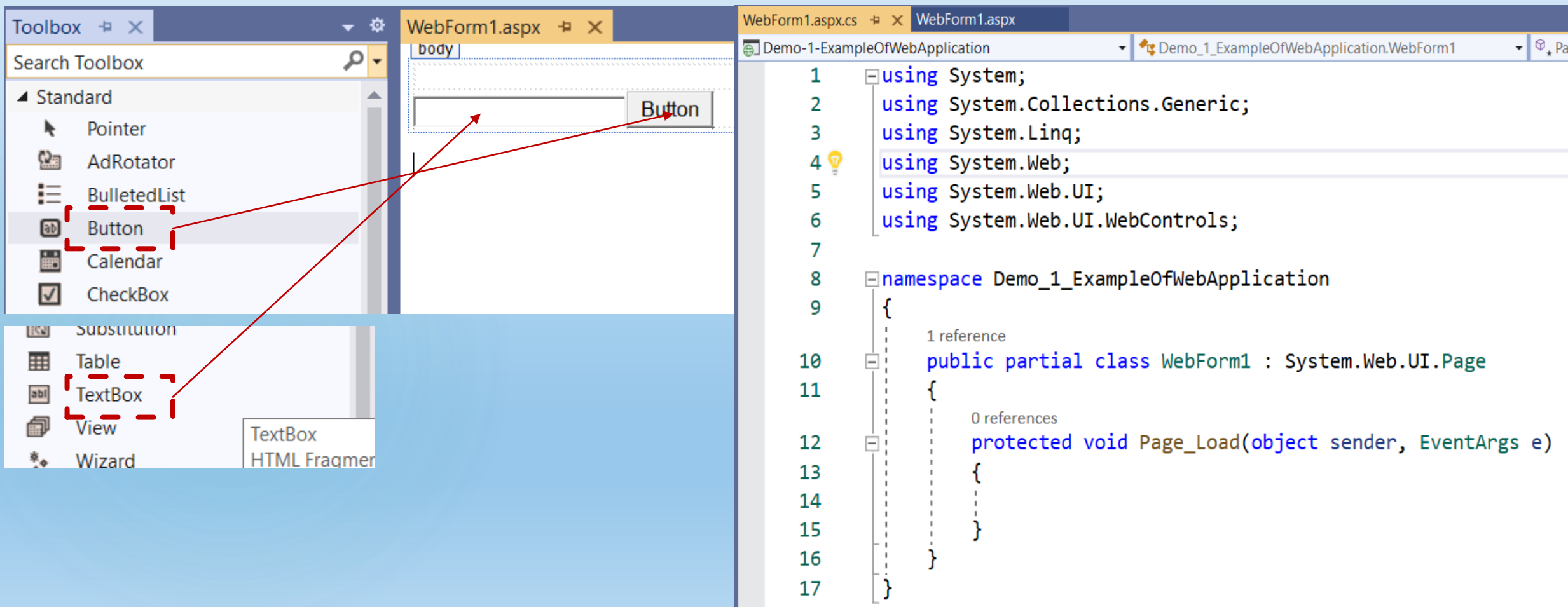
# Добавяне на нова web форма в проекта



# I. ИЗГРАЖДАНЕ НА УЕБ ПРИЛОЖЕНИЯ С ASP.NET

10

Отляво е кутията с инструменти (toolbox), където са контролите. Нека добавим текстово поле и бутон, като привлечим двете контроли върху формата.



The image displays the Visual Studio IDE with two windows open. The left window shows the design view of 'WebForm1.aspx', which contains a single 'body' container. A red dashed box highlights the 'Button' control in the 'Standard' group of the 'Toolbox' on the left. Another red dashed box highlights the 'TextBox' control in the 'WebForms' group. Red arrows point from these controls to their positions on the 'body' of the web form. The right window shows the code view of 'WebForm1.aspx.cs'. The code includes the following using statements and namespace declaration:

```
1 using System;
2 using System.Collections.Generic;
3 using System.Linq;
4 using System.Web;
5 using System.Web.UI;
6 using System.Web.UI.WebControls;

7
8 namespace Demo_1_ExampleOfWebApplication
9 {
10     1 reference
11     public partial class WebForm1 : System.Web.UI.Page
12     {
13         0 references
14         protected void Page_Load(object sender, EventArgs e)
15         {
16             }
17     }
```