

ОСНОВЫ СОЗДАНИЯ ВЕБ-СТРАНИЦ С ПОМОЩЬЮ ТЕХНОЛОГИИ ASP.NET

Технология создания Web-приложений ASP.NET

Web-приложения способны выполнять различные операции, среди которых наиболее типичны:

- 1) принимать данные от пользователя и сохранять их на сервере;
- 2) выполнять для пользователя различные действия: размещать заказы, делать сложные вычисления и извлекать информацию из баз данных, добавлять, удалять, изменять сведения в базах данных;
- 3) опознавать пользователя и отображать соответствующий интерфейс;
- 4) отображать постоянно меняющееся содержимое, оперативную информацию.

Этот перечень далеко не полон. Web-приложения способны решить любые задачи, доступные клиент-серверному приложению.

Существуют две главные особенности разработки Web-приложений:

- 1) Web-приложения исполняются на сервере. Весь программный код исполняется в рамках веб-сервера, а клиенту доставляется уже готовая разметка HTML, которая отображается внутри браузера.
- 2) Web-приложения не хранят состояния о пользователе после того, как сервер обработал его запрос.

При построении любого Web-приложения приходится решать типичные задачи: способы хранения информации о пользователе, организацию сеансов работы пользователя, способы переходов от страницы к странице, механизмы оптимизации эффективности (например, кэширование) и другие. Поскольку набор этих задач является достаточно стандартным и одинаково решается для большинства Web-приложений, то его реализация вынесена в отдельные технологии для разработки Web-приложений. К таким технологиям относятся технология Microsoft ASP.NET, PHP, Ruby on Rails и др. В них, фактически, содержатся все компоненты, необходимые для реализации Web-приложений и учитывающие их специфику.

ASP.NET — это технология от Microsoft, предназначенная для создания серверных Web-приложений. Она входит в состав платформы Microsoft .NET Framework, которая представляет собой набор тесно

связанных технологий, начиная с доступа к базам данных и заканчивая распределенными приложениями.

ASP.NET является частью инфраструктуры .NET Framework и состоит из следующих компонентов:

- 1) инструментов Visual Studio .NET для Web-разработки — графических средств разработки Web-страниц, шаблонов Web-приложений и инструментов для управления проектами и развертывания Web-приложений;
- 2) пространств имен System.Web, которые являются частью .NET Framework и включают классы для работы с элементами, специфичными для Web, такими, как HTTP-запросы и отклики, браузеры и электронная почта;
- 3) серверных элементов управления и HTML-элементов управления — компонентов пользовательского интерфейса, применяемых для приема данных от пользователей и отображения им отклика приложения.

На самом базовом уровне приложение ASP.NET представляет собой комбинацию файлов, страниц, обработчиков, модулей и исполняемого кода, который можно вызывать из виртуального каталога (и его подкаталогов) на веб-сервере.

Каждое Web-приложение, разрабатываемое на основе ASP.NET, состоит из информационной части, программного кода и сведений о конфигурации.

Информационная часть содержит статические и динамические элементы страницы и реализуется в виде Web-форм. Статические элементы представляют собой типичные элементы языка HTML, динамические же komponуются программным кодом приложения во время его выполнения (например, запросы к базе данных).

Программный код реализует логику, определенную в процедурах обработки данных, которые определяют реакцию приложения на запросы пользователя. Программный код исполняется сервером и взаимодействует с динамическими элементами информационной части для формирования отклика приложения.

Сведения о конфигурации представляют собой файлы, содержащие параметры, определяющие способ исполнения приложения на сервере, параметры безопасности, реакцию приложения на возникающие ошибки и т. д.

Основным элементом Web-приложения является Web-форма (или Web-страница), которая, с одной стороны, похожа на Windows-форму, так как позволяет размещать внутри себя различные элементы управления, способные отображать данные и реагировать на действия пользователя, а с другой — представляет собой HTML-страницу, т. к. содержит все ее атрибуты.

При разработке Web-приложений на основе ASP.NET возможны два варианта организации Web-форм.

В первом случае весь код информационной части и программная часть хранятся в одном файле с расширением .aspx. Программный код при этом помещается в так называемые блоки сценариев. При этом сохраняется возможность использования всех принципов современного программирования, таких как реакция на события элементов управления, подпрограммы и т.д. Эту модель целесообразно использовать при создании простых Web-приложений, поскольку в этом случае все хранится в одном пакете.

Во втором случае каждая Web-страница разделяется на две части: Web-форму и файл, содержащий программный код. При этом форма, как и в первом случае, сохраняется в файле с расширением .aspx, а программный код — в файле с расширением .cs. Такая модель обеспечивает лучшую организацию элементов Web-приложения за счет отделения пользовательского интерфейса от программной логики.


ЗАДАНИЯ

Создание ASP.NET-страницы

Задание 1. Создайте простую ASP.NET-страницу.

Ход выполнения:

1. Создайте новый проект **ASP.NET Empty Web Application** с именем **Hello**.
2. Добавьте в проект веб-форму. (Для этого в меню **Project** выберите команду **Add New Item...** В открывшемся диалоговом окне выберите в списке элементов **Web Form**.) Задайте для формы имя **Default.aspx**.
3. Откройте файл **Default.aspx** и просмотрите содержащийся в нем HTML-код.
4. В парный тег `<div></div>` добавьте текстовый абзац произвольного содержания (Например, `<p>Hello, Web!</p>`).

5. Задайте в качестве начальной страницы **Default.aspx**. Для этого в окне *Solution Explorer* в контекстном меню файла Default.aspx выберите пункт **Set As Start Page**.
6. Нажмите сочетание клавиш CTRL+F5, чтобы запустить веб-страницу в браузере. Откроется первая страница.
7. Измените стиль текста. Для этого установите курсор на тег <p> и перейдите в окно **Properties**. Выберите там **Style** и нажмите кнопку  .
8. В появившемся окне задайте новое оформление абзаца.
9. Запустите веб-страницу в браузере (CTRL+F5).

Работа с элементами управления Label и TextBox

Элементы управления – это специализированные объекты, которые можно размещать на формах HTML. Они используются для взаимодействия с пользователем. Для создания пользовательского интерфейса Web-приложения возможно использование как серверных, так и HTML-элементов управления.

Доступ ко всем элементам управления, в том числе и к HTML-элементам, осуществляется посредством панели инструментов Toolbox.

Задание 2. Поработайте с элементом управления Label.

1. Создайте страницу (новую веб-форму) с именем **Text.aspx**.
2. Перейдите в режим **Design** и добавьте элемент управления **Label**.
3. Перейдите в окно *Properties* и в свойстве **Text** введите значение *Элемент управления Label*.
4. Запустите веб-страницу в браузере (CTRL+F5).
5. Перейдите в режим **Source** для веб-формы *Text.aspx*. ASP-код для добавленного элемента управления выглядит следующим образом:

```
<asp:Label ID="Label1" runat="server" Text="Элемент управления Label"></asp:Label>
```

Добавьте в эту строку теги так, чтобы строка приняла вид:

```
<asp:Label ID="Label1" runat="server" Text="<center><b><i>Элемент управления Label</i><b></center>"></asp:Label>
```

6. Запустите веб-страницу в браузере и посмотрите, как изменился внешний вид текста.

Задание 3. Поработайте с элементом управления TextBox.

1. Перейдите в режим **Design** веб-формы *Text.aspx* и добавьте 3 элемента управления **TextBox**.

2. Выделите второй добавленный элемент и перейдите в окно *Properties*. В свойстве **TextMode** установите значение **MultiLine**. Свойство **MultiLine** определяет многострочный способ ввода текста.
3. Для третьего элемента **TextBox** в свойстве **TextMode** установите значение **Password**. Вводимый в этот элемент управления текст будет скрываться. Поля такого типа используются как правило для ввода паролей.
4. Запустите веб-страницу в браузере и попробуйте ввести текст в каждый из трех элементов **TextBox**.

Задание 4. Создайте новую веб-форму *Anketa.aspx* следующего вида:

Анкета

Имя:

Возраст:

E-mail:

О себе:

Переход между страницами. Элементы управления **Button и **HyperLink****


*Элемент управления **Button** (кнопка) позволяет передать на сервер форму, заполненную в окне браузера. Этот элемент управления можно использовать также и для перехода на другую страницу.*

*Ссылку на другую страницу создает и элемент управления **HyperLink**.*

*Существует еще один элемент управления, для которого можно указать страницу для перехода, - **ImageButton**. Для использования его в качестве гиперссылки необходимо указать в свойствах путь к файлу изображения и путь веб-страницы, на которую осуществляется переход.*


Задание 5. Создайте на главной странице проекта ссылку для перехода на веб-страницу Anketa.aspx.

Ход выполнения:

1. Разместите на странице заголовок первого уровня с текстом *Главная страница*.
2. Ниже разместите элемент управления HyperLink. Перейдите в окно *Properties*. В поле **Text** внесите значение **Анкета**. Раскройте свойство **Font** и укажите полужирное начертание (**True** для пункта **Bold**) и подчеркнутый текст (**True** для пункта **Underline**). Выберите цвет и размер текста гиперссылки.
3. В качестве страницы для перехода укажите веб-форму *Anketa.aspx*. Для этого в свойстве **NavigateUrl** впишите **~/Anketa.aspx** или нажмите кнопку  и выберите необходимое имя файла в появившемся диалоговом окне.
4. Запустите веб-страницу в браузере и проверьте работоспособность ссылки. Вернитесь на исходную страницу (*Назад*).
5. Сделайте так, чтобы страница с анкетой открывалась в новой вкладке (новом окне). Для этого в свойстве **Target** выберите **_blank**.

Задание 6. На главной страницы разместите кнопку, при щелчке на которой произойдет переход на страницу Anketa.aspx.

Для этого:

1. Разместите на странице элемент управления Button.
2. Перейдите в окно *Properties*. В поле **Text** внесите значение **Анкета**. Оформите внешний вид кнопки на ваше усмотрение.
3. В свойстве **PostBackUrl** укажите страницу *Anketa.aspx*, выбрав ее в диалоговом окне после нажатия кнопки .

Задание 7. Создайте на странице Anketa.aspx ссылку для перехода на главную страницу.

В качестве ссылки можно использовать изображение. Для этого подберите изображения и поместите его файл во вложенной папке проекта **Hello** с таким же именем. Для свойства **ImageUrl** элемента управления **HyperLink** укажите имя файла изображения с расширением.

Проверка вводимых данных

Важным правилом создания приложений является получение только корректных данных, для чего вводимая информация перед использованием

проверяется. Иными словами, обработке любого ввода извне должен предшествовать шаг валидации. В ASP.NET для этой цели имеются специальные валидационные элементы управления, которые реализуют удобный механизм выполнения типичных задач по проверке данных, включая проверку на допустимость типов, на принадлежность значений заданному диапазону, а также на заполнение обязательных полей.

Каждый валидационный элемент управления содержит ссылку на тот или иной расположенный на странице элемент управления, предназначенный для ввода данных от пользователя. Каждый валидатор выполняет проверку определенного типа. Один элемент управления могут проверять несколько валидаторов.

Валидатор *RegularExpressionValidator* позволяет сравнить вводимый текст с шаблоном, определенным в регулярном выражении, которое устанавливается в свойстве *ValidationExpression*. Регулярные выражения позволяют определять правила, задающие количество символов и их положение в строке.

Задание 8. Осуществите проверку вводимых данных на странице *Anketa.aspx*. Поля Имя, возраст и E-mail обязательны для заполнения. Возраст пользователя может быть в пределах от 8 до 90 лет.

Для выполнения задания проделайте следующие действия:

1. Перейдите в режим **Design** для страницы *Anketa.aspx*.
2. Добавьте на форму кнопку и в свойстве **Text** напишите **OK**.
3. Добавьте рядом с текстовым полем для ввода имени элемент управления **RequiredFieldValidator** (он расположен в группе **Validation**).
4. Перейдите в окно *Properties* и в свойстве **Text** введите символ *, а в свойстве **ForeColor** выберите красный цвет. Это значит, что при нажатии на кнопку **OK** при незаполненном поле для ввода имени рядом с ним появится красная звездочка, сигнализирующая о том, что это поле обязательно для заполнения.
5. В свойстве **ControlToValidate** укажите имя соответствующего элемента управления *TextBox*. (Имя элемента управления можно найти в свойстве **ID**).
6. Прделайте аналогичные действия (3-5) для полей ввода возраста и адреса электронной почты.
7. Добавьте элемент управления **RangeValidator** рядом с полем для ввода возраста. Укажите в качестве элемента, который нужно проверить,

текстовое поле для ввода возраста. В свойстве текст напишите «*Возраст должен быть в пределах 8-90 лет*». В свойстве **MinimumValue** укажите значение **8**, а в свойстве **MaximumValue** – значение **90**. В свойстве **Type** выберите значение **Integer**.

8. Рядом с полем для ввода E-mail добавьте элемент управления **RegularExpressionValidator**. В качестве элемента, который нужно проверить, укажите текстовое поле для ввода e-mail. В свойстве текст напишите «*Неправильный формат ввода e-mail*». В свойстве **ValidationExpression** укажите регулярное выражение для проверки корректности ввода адреса электронной почты.
9. Запустите веб-страницу в браузере и проверьте работоспособность проверки вводимых данных. Для этого вводите различные значения и нажимайте кнопку ОК.

Таблица. Последовательности символов, применяемые в регулярных выражениях

.	Соответствует любому символу.
[aeiou]	Соответствует каждому отдельному символу, входящему во множество.
[^aeiou]	Соответствует каждому отдельному символу, не входящему во множество.
[0-9a-fA-F]	Соответствует любому символу, приведенному в заданных диапазонах (0-9; a-f; A-F).
\w	Соответствует любому алфавитно-цифровому символу из слова, то есть любому алфавитно-цифровому символу или символу подчеркивания.
\W	Соответствует любому символу, не являющемуся алфавитно-цифровым либо символом подчеркивания.
\s	Соответствует любому символу разделителя (пробел, перевод строки, табуляция, перевод строки, новая строка и т. д.).
\S	Соответствует любому непробельному символу.
\d	Соответствует любому десятичному символу.
\D	Соответствует любому недесятичному символу.
*	Ноль или более совпадений
+	Одно или более совпадений
?	Ноль или одно совпадение
{n}	n совпадений
{n,}	n или более совпадений
{n,m}	Количество совпадений от n до m

Результаты проверки вводимой информации

Для отображения итоговой информации о результатах всех проверок вводимой информации необходимо использовать элемент управления

ValidationSummary. Он выводит значение *ErrorMessage* каждого валидатора, для которого эта проверка завершилась неудачей. Итоговая информация может отображаться либо на странице, либо в отдельном окне. Для указания данного режима необходимо установить свойства *ShowMessageBox* и *ShowSummary*. При значении свойства *ShowMessageBox=true* сообщение выводится в отдельном окне, если же значение свойства *ShowSummary=true* - на странице. При отображении итоговой информации на странице возможно установить некоторые дополнительные параметры с помощью свойства *DisplayMode*, а также задать заголовок для итоговой информации с помощью свойства *HeaderText*.

Задание 9. Отобразите итоговую информацию о результатах всех проверок вводимой информации.

Для выполнения задания проделайте следующие действия:

1. Перейдите в режим **Design** для страницы *Anketa.aspx*.
2. Поочередно выделяйте валидационные элементы управления и в свойство **ErrorMessage** каждого из этих элементов впишите сообщения об ошибке.
3. Добавьте на веб-форму элемент управления **ValidationSummary**.
4. Запустите веб-страницу в браузере и проверьте работоспособность этого элемента управления.
5. Измените внешний вид выводимых сообщений об ошибке. Перейдите в окно *Properties* и в свойстве **ForeColor** выберите красный цвет. В свойство **HeaderText** задайте заголовок итоговой информации «*В результате заполнения полей возникли следующие ошибки:*».
6. Запустите веб-страницу в браузере и посмотрите на результат изменений.
7. Выведите сообщение об ошибке в отдельном окне. Для этого в свойстве **ShowMessageBox** выберите **True**, а в свойстве **ShowSummary** – значение **False**, то есть отмените вывод сообщений на странице.

Элементы управления *RadioButton*, *RadioButtonList*, *DropDownList*

Задание 10. Добавьте на веб-форму *Anketa.aspx* новые элементы управления в соответствии с рисунком:

Анкета

Имя:

Возраст:

Пол:



☐ Мужской ☐ Женский

Дата рождения:

E-mail:

О себе:

Для выполнения задания проделайте следующее:

1. Поместите на странице два элемента управления Label с текстом *Пол* и *Дата рождения*.
2. Добавьте элемент управления **RadioButtonList**.
3. Создать элементы для **RadioButtonList** можно одним из следующих способов:
 - а) Нажать на кнопке  элемента управления и в меню выбрать **Edit Items...**
 - б) Перейдите в окно *Properties* и в свойстве **Items** нажать кнопку .
4. Далее в открывшемся диалоговом окне добавьте два элемента, нажав кнопку **Add** два раза. В свойстве **Text** в правой части окна введите значение элементов: *Мужской* и *Женский*.
5. Запустите веб-страницу в браузере и посмотрите, как работает данный элемент управления.
6. Поместите на веб-форму два элемента управления **DropDownList** и один **TextBox**.
7. Чтобы добавить элементы для **DropDownList**, поступите одним из способов, описанных в пункте 3, а затем добавьте элементы аналогично тому, как вы делали это для элемента управления **RadioButtonList**. В

первом элементе управления DropDownList элементами должны быть числа от 1 до 31 (дни месяца), во втором – названия месяцев.

8. Запустите веб-страницу в браузере и посмотрите, как работает данный элемент управления.

Привязка и отображение данных

Большинство элементов управления ASP.NET поддерживают привязку данных. При этом привязка может работать как для данных с одним значением, так и для данных с множественными значениями. Привязка с одним значением означает, что элемент управления может отображать единственное значение, извлекаемое из источника данных.

Привязка с множественными значениями означает, что элемент управления может отображать несколько значений, извлекаемых из источника данных. Элементы управления, поддерживающие привязку с множественными значениями, строятся на основе списков и таблиц. Рассмотрим, как осуществить привязку к данным элементами управления RadioButtonList и DropDownList.

Кроме использованных списковых элементов, ASP.NET содержит также и более сложные элементы, способные привязываться к данным и отображать их. К таким элементам относится, например, GridView.

GridView представляет собой таблицу, способную привязываться к данным и отображать содержимое таблиц. GridView является самым мощным элементом по количеству возможностей. Очень часто его приходится использовать для отображения информации в виде таблицы.

Задание 12. Осуществите привязку элементов управления RadioButtonList, DropDownList и GridView к базе данных.

Ход выполнения:

1. Создайте базу данных произвольного содержания. Поместите файл базы данных в папку проекта.

Для создания базы данных в Visual Studio необходимо создать подключение к базе данных. Для этого откройте окно Server Explorer и щелкните на кнопке Connect to Database (Подключиться к базе данных), на которой изображен шнур электропитания со знаком "плюс" зеленого цвета. Появится диалоговое окно Add Connection (Добавление подключения). В этом окне:



- 1) Выберите в качестве источника данных файл базы данных Microsoft SQL Server.
- 2) Укажите имя для создаваемой базы данных и выберите папку, в которую нужно поместить создаваемый файл.

3) Удостоверьтесь, что выбран переключатель *Use Windows Authentication* (Использовать аутентификацию Windows).

4) Щелкните на кнопке *ОК*. Вскоре отобразится диалоговое окно с запросом о создании новой базы данных. Щелкните в нем на кнопке *Yes* (Да) и в окне *Server Explorer* появится новый элемент. Развернув этот элемент, можно просмотреть различные аспекты только что созданной базы данных.

Для добавления таблицы в базу данных нужно в окне *Server Explorer* развернуть созданную базу данных и щелкнуть правой кнопкой мыши на элементе *Tables* (Таблицы) и выбрать в контекстном меню пункт *Add New Table* (Добавить новую таблицу). Отобразится конструктор для создания новой таблицы.

Для заполнения таблицы данными в окне *Server Explorer* разверните элемент *Tables*, щелкните правой кнопкой мыши на таблице и выберите в контекстном меню пункт *Show Table Data* (Показать данные таблицы). Здесь можно ввести данные вручную.

2. Добавьте к проекту новую страницу.
3. Добавьте на страницу элемент управления **RadioButtonList** и привяжите его к данным. Для этого нажмите на кнопке  элемента управления и в меню выберите пункт **Choose Data Source...**
4. В поле **Select a data source** выберите **<New data source...>** и укажите источник данных.
5. В появившемся диалоговом окне введите имя базы данных и нажмите **Next >**.
6. В новом диалоговом окне выберите в выпадающем списке **Name** название таблицы на ваш выбор, а также отметьте только имя одного поля в области **Columns**. Нажмите кнопку **Next >**, а в следующем окне - **Finish**.
7. Запустите веб-страницу в браузере и посмотрите на результаты.
8. Добавьте на страницу элемент управления **DropDownList** и привяжите его к данным, проделав операции, описанные в пунктах 3-8.
9. Добавьте на страницу элемент управления **GridView** и привяжите его к данным. Для этого нажмите на кнопке  элемента управления и в меню в пункте **Choose Data Source...** выберите **<New data source...>**.
10. В появившемся диалоговом окне введите имя базы данных и нажмите **Next >**.
11. В новом диалоговом окне выберите в выпадающем списке **Name** название таблицы на ваш выбор, а затем отметьте поля в области **Columns**, которые должны отражаться в таблице на странице. Нажмите кнопку **Next>**, а в следующем окне - **Finish**.
12. Запустите веб-страницу в браузере и посмотрите на результаты.