



ERASMUS+  
Enriching lives, opening minds

**School of Business**

**of the Belarusian State University**

# WEB-technologies



SCHOOL OF  
BUSINESS  
OF BSU

# INTRODUCTION

The World-Wide Web technologies which were originally designed in CERN (European Organization for Nuclear Research) as a means of delivering documents from one scientific establishment to another one are now used as a platform for complex interactive applications which slowly but steadily drive out installable applications.

This process is facilitated with the advantages that WEB applications have:

- anytime access from different devices,
- automatic updating,
- a possibility of integration of different applications written for different platforms

However creating WEB applications requires a bit different approaches than creating traditional installable applications and is for the moment impossible without integrating a wide variety of approaches and technologies.

This course introduces to these different WEB technologies and gives experience in creating WEB applications. Students learn theoretically and get practical skills in markup languages, scripting languages, network protocols, graphics and video-images, event-driven programming, object oriented programming, databases.

This course introduces to these different WEB technologies and gives experience in creating WEB applications. Students learn theoretically and get practical skills in markup languages, scripting languages, network protocols, graphics and video-images, event-driven programming, object oriented programming, databases.

That is, those technologies which allow constructing modern WEB applications.

There is a difficulty with selecting technologies of WEB programming, because this is the field where changes happen most rapidly. In his book “Business @ the Speed of Thought” Bill Gates told that Microsoft updates these technologies almost completely once every 3 years. And the course has to be updated very often – those technologies which are mainframe today may lose their positions within a year.

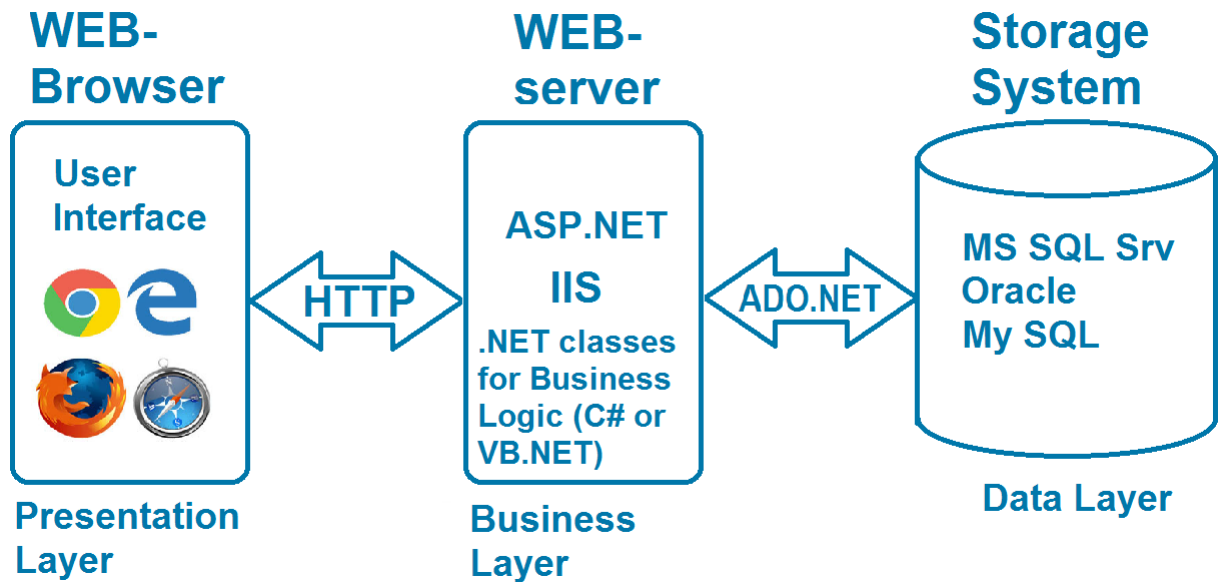
We have found the following way out.

However something remains relatively stable – programming concepts and application architecture.

This is why while answering the question what is more important, knowledge or understanding, we expressly decide for understanding. We follow the principle that understanding of concepts and architecture is more important than specific methods of coding which may undergo changes faster than we would like them to.

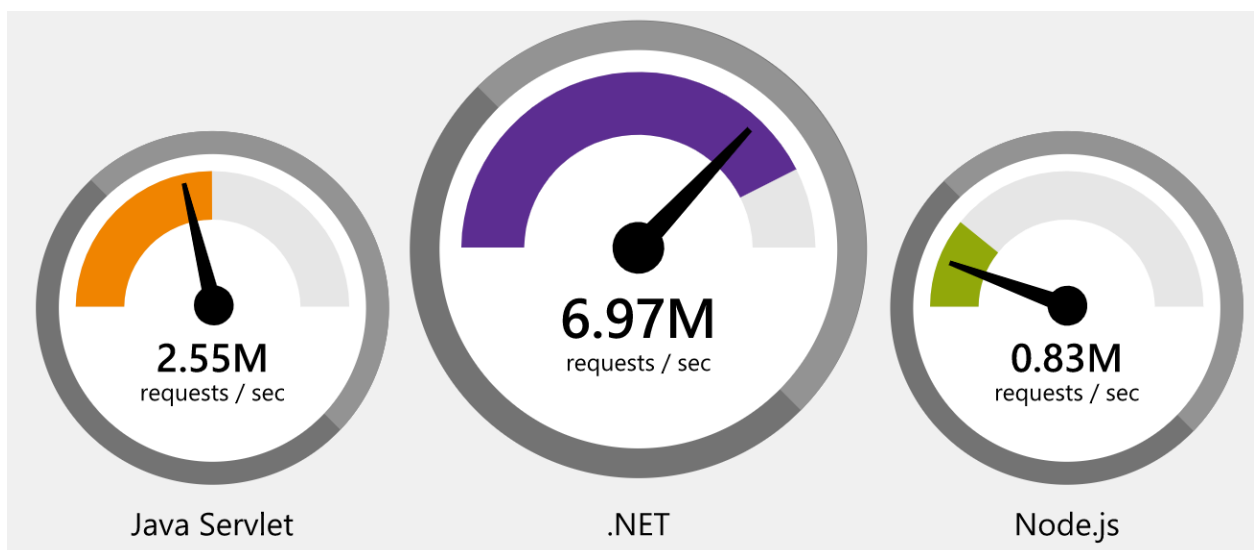
The architecture of a WEB application usually includes the following component parts:

- Application interface in a Web Browser (Front End)
- Server end (Back End)
- Data storage system (Data Storage)



HTML/CSS/JavaScript and Document object Model (DOM) - Document structure Browser software, as well as concepts of Model View Controller, Single page applications (Angular JS and Vue.JS frameworks) are studied at the interface level (Presentation Layer)

The ASP.NET technology based on C# and VB.NET is studied at the server level (Business Layer).



The ADO.NET technology for interoperating with databases is studied at the data level (Data Layer).

Such a selection is conditioned by the fact that today ASP.NET performs faster than any popular WEB framework.

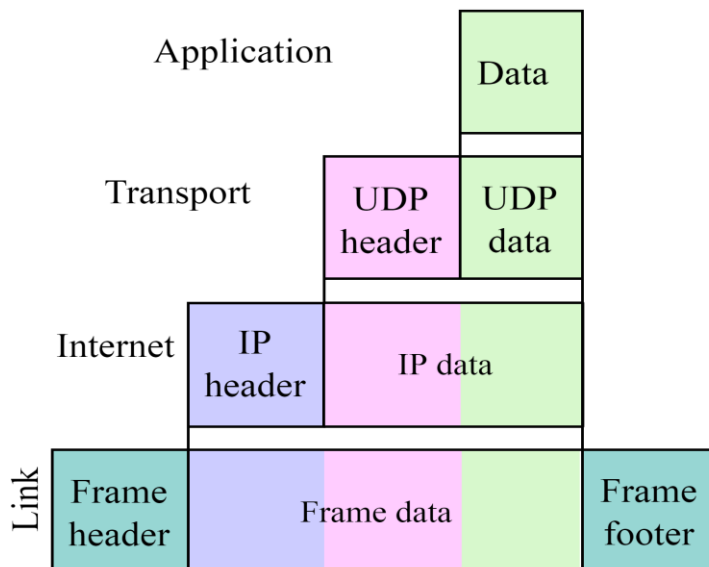
If the situation changes in a year or two and Node.js outperforms ASP.NET in speed or popularity, then we will possibly learn Node.js at the server level.

# DHCP

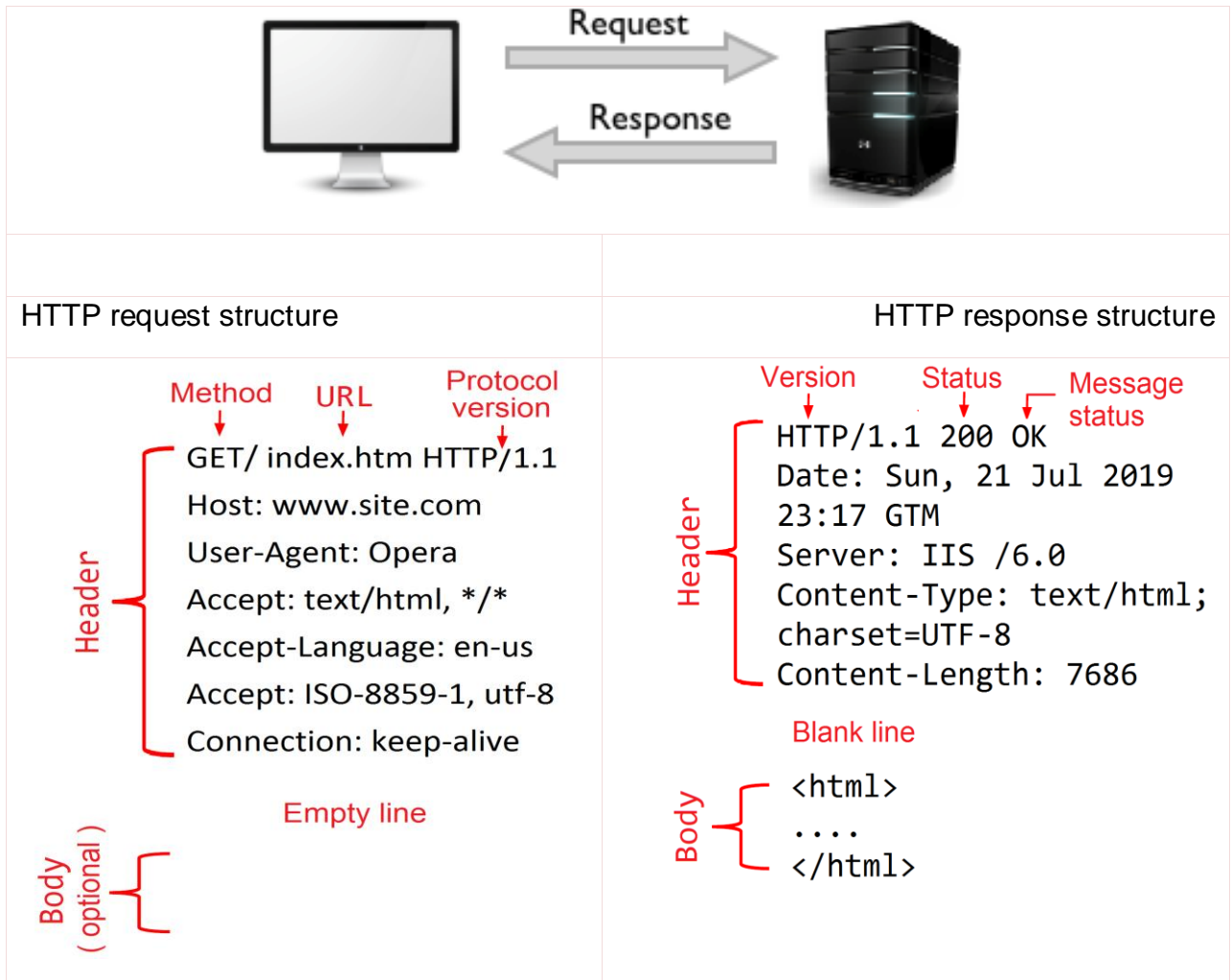
**DHCP** (Dynamic Host Configuration Protocol — a network protocol that allows network devices to automatically obtain the IP address and other parameters necessary to operate on a TCP/IP network.



**TCP/IP** - set of standards and rules defining how data is transmitted by network.







## Status codes

### 1xx: Informational Messages

**100** continue, which means that the client is still sending the rest of the request.

### 2xx: Success Messages

**200** OK

If the client received the code from the 2xx series, then the request was successful..

### 3xx: Redirection

**301** Moved Permanently: The resource can now be found at a different URL.

### 4xx: Client Errors

**404** Not Found. Resource not found on server..

**401** Unauthorized: Authentication is required to complete a request. Information is passed through the Authorization header..

**403** Forbidden: The server did not open access to the resource.

### 5xx: Server Errors

**500** Internal Server Error.

**503 Service Unavailable:** this can happen if an error occurred on the server or it is overloaded.

# Browsers

The purpose of the web browser is to read HTML documents and compose them into visual web pages (with the ability to play music and display video files). The browser does not display HTML tags, but uses tags to interpret page content and format text as needed.

Today there are the following web browsers (in which you need to check your WEB design so that your site is reflected in them correctly):

- Internet Explorer (or Microsoft)
- Firefox (from Mozilla)
- Chrome (from Google)
- Safari (from Apple)
- Opera (Opera from Norway)



# HTML

The Hyper Text Markup Language (HTML) is used to write Web pages intended for publication on the World Wide Web (WWW). It was developed in the late 80s - early 90s. by initiative group at CERN's European Particle Physics Laboratory in Geneva.

HTML is a hypertext markup language. It is the main building material of web pages.

HTML consists of text and tags enclosed in brackets that indicate how to display the text and graphics.

HTML tags are usually used in pairs `<html></html>`.

The first tag is called the opening tag, for example `<html>` `<script>`

The second - closing `</html>` `</script>` - denoting the end of the tag range. In between these tags, web designers can add text, tables, images, etc.

Example of HTML-page:

```
<html>
```

0.html

```
<body>
```

**Head**

```
<h1>Head</h1>
```

Paragraf

```
<p>Paragraf</p>
```

```
</body>
```

```
</html>
```

See more: <http://asp.net.by/en/html/>

# CSS

CSS - Cascading Style Sheets, что означает дословно "каскадные таблицы стилей". Using CSS describes the style of the document or site as a whole, or its individual elements.

Styles are described in the tag.

```
<style></style>
```



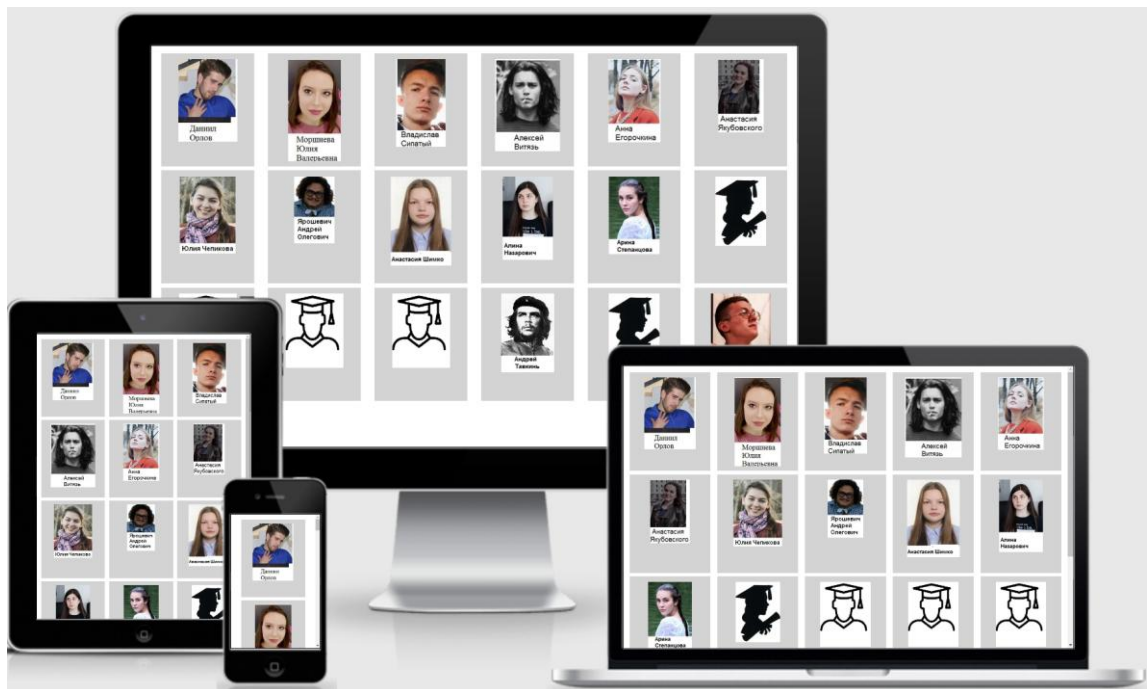
They can be written to a separate file, and then loaded into an html page as follows:

```
<link REL="STYLESHEET" TYPE="text/css" HREF="ss.css">
```

See more: <http://asp.net.by/en/css/>

# Adaptive sites

Adaptive layout - is a layout that can be customized for various screen sizes. Adaptive markup is created using media queries, which appeared in the CSS3 specification and are currently supported by all major browsers.



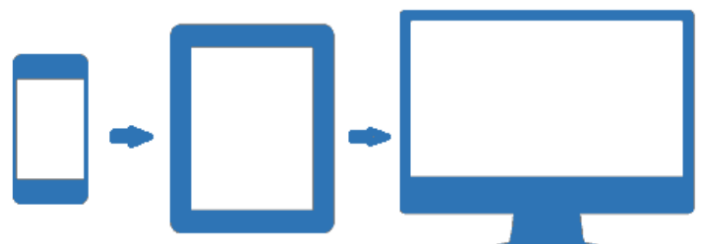
The need for adaptive layout is determined by the fact that at the moment the number of users using mobile traffic has reached 60%. Mobile traffic is becoming more significant and site owners should reckon with these statistics. First of all, the site should be optimized for mobile devices – Mobile First.

Principles of the Mobile First:

1) The most important content should be shown first.

2) Website should be light and load fast

3). Additional information should be loaded at the request of the user.



See more: <http://asp.net.by/en/mobile/>

# CSS FlexBox

They are considered the best way to create an adapted site for a small screen, even when the screen size is unknown and (or) dynamic.



See more: <http://asp.net.by/en/mobile/>

# JavaScript

JavaScript is an object-oriented language designed to create dynamic web pages.

That allows you to create complex dynamic sites.

Here is an example of a simple HTML page with JavaScript code:

```
<html>
<body>
<h1>My 1st JavaScript page</h1>
<p>Click the button to find out the time.
</p>
<p id="date_area"></p>
<button type="button"
  onclick="cmdDate()">
  Show time
  </button>
<script>
  function cmdDate()
```

<http://asp.net.by/js/00.html>

```

{
document.getElementById("date_area")
    .innerHTML = Date();
}
</script>
</body>
</html>

```

ECMAScript was created to standardize JavaScript.

Allows you to use convenient Java-like syntax when defining classes:

```
<script>
```

```

class ACat {
    constructor(n) {
        this.name = n;    //СВОЙСТВО
    }
}

```

<http://asp.net.by/ES6/07.htm>



```
mycat = new ACat("Barsik");
```

```
document.write("My cat's name is " + mycat.name);
```

```
</script>
```

See more: <http://asp.net.by/en/es6/>

# JSON




The mechanism that is used to transfer data from server to client.  
This is a format for exchanging data.

JSON Syntax Rules:

- Data in **name / value** pairs ("lastName": "Musk")

- Data is separated by commas ("firstName":"Elon", "lastName":"Musk")
- Braces hold items  
{"firstName":"Elon", "lastName":"Musk" }
- Square brackets contain arrays {"managers":[.....]}

Example:

| XML  |   | JSON   |
|--|---|--|
| < managers >   |   | {"managers":[  |
| <manager><br><firstName>Bill</firstName><br><lastName>Gates</lastName><br></manager> |    | {<br>"firstName":"Bill",<br>"lastName":"Gates"<br>}, |
| <manager><br><firstName>Mike</firstName><br><lastName>Dell</lastName><br></manager>  |   | {<br>"firstName":"Mike",<br>"lastName":"Dell"<br>},  |
| <manager><br><firstName>Elon</firstName><br><lastName>Musk</lastName><br></manager>  |  | {<br>"firstName":"Elon",<br>"lastName":"Musk"<br>}   |
| </managers>  |   | ]]   |

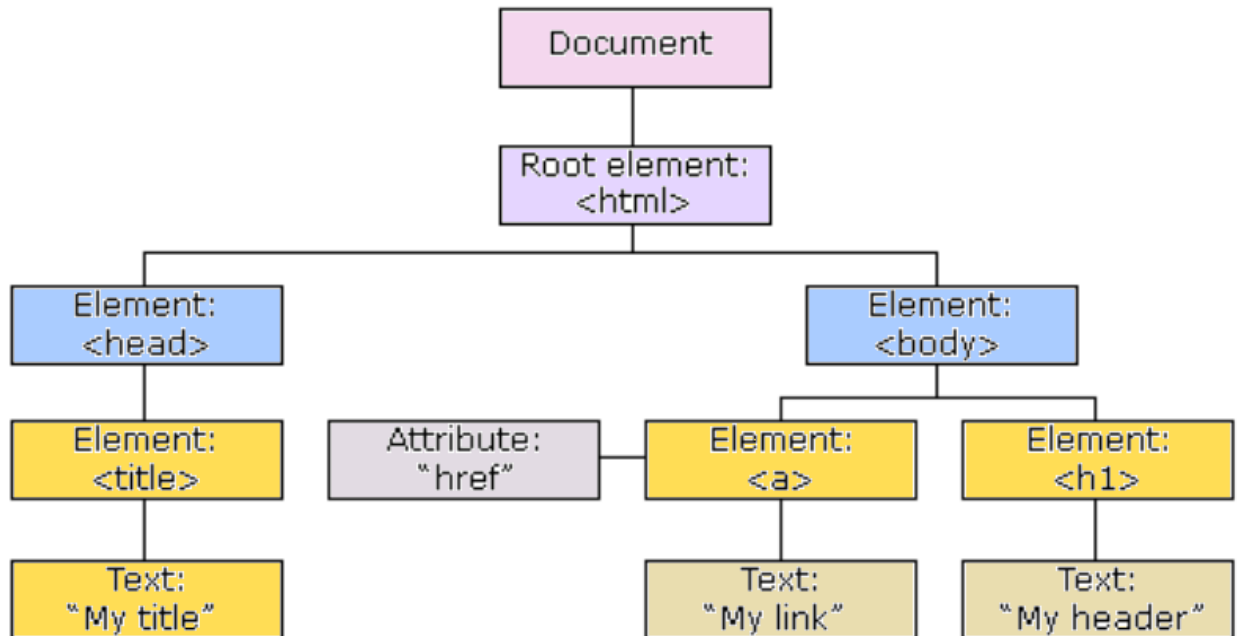
The JSON format is converted to the JavaScript object format using the operator:

```
var myObj = JSON.parse(usManagers);  
myObj.firstName    myObj.lastName
```

See more.: <http://asp.net.by/en/JSON/>

# DOM

DOM - a cross-platform and independent interface that presents an XML or HTML document in a tree structure, in which each node represents an object that is part of a document.



JavaScript can request or modify an HTML document:

```
<html>
<body>
<form name="fr" action="">
```

Name:

```
Name: <input type="text" name="fname" value="Ann" size="2">
<input type="button" value="Go"
onClick='javascript:document.getElementsByName("fname")[0].value="Alex"'> cm.1
onClick='javascript:document.fr.fname.value="Alex"'> cm.2
```

```
</form>
</body>
</html>
```

Name:

For more information about the DOM, see: <http://asp.net.by/en/DOM/>

# AJAX

AJAX, Ajax (Asynchronous Javascript and XML - "Asynchronous JavaScript and XML"). The approach to building interactive user interfaces of web applications, which consists in the ability to download data without reloading the web page.

As a result, when updating data, the web page does not reload completely, and web applications become faster and more convenient.

<http://asp.net.by/ajax/00.html>

Sun Dec 15 2019 00:10:03  
GMT+0200 (Восточная  
Европа, стандартное  
время)

Sun Dec 15 2019 00:10:03  
GMT+0200 (Восточная  
Европа, стандартное  
время)

Text for changing

GO

AJAX -  
мечта  
WEB-

See more.: <http://asp.net.by/en/DOM/>

# jQuery

A set of JavaScript functions that focuses on the interaction of JavaScript and HTML.

The jQuery library helps:

- easy to access any item of the DOM (`$("#h1").show();`)
- access attributes and contents of DOM elements (`(p[0].value="Alex");`)
- manipulate them.

The jQuery library provides a convenient API for working with AJAX.

Example - <http://asp.net.by/jquery/01.html>

```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
</script>
<script>
jQuery (document).ready(function(){
    $("#hide").click(function(){
        $("#h1").hide();
    });
});
```

```

$( "#show" ).click( function() {
    $( "#h1" ).show();
    });
});
</script>
</head>

<body>
<h1>Hello</ h1>
< h1>jQuery</ h1>

<button id="hide">Hide</button>
<button id="show">Show</button>

</body>
</html>

```



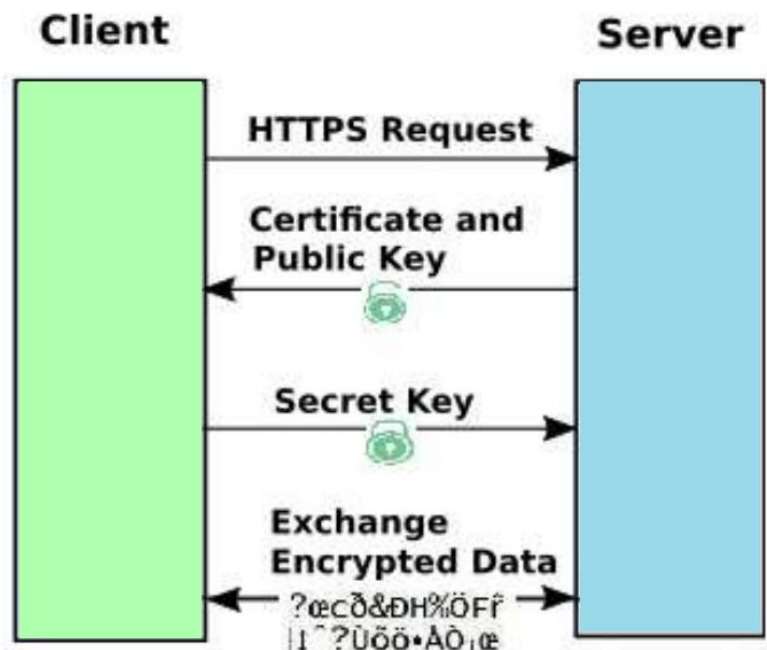
See more.: <http://asp.net.by/en/jQuery/>

# SSL

SSL (Secure Sockets Layer) — standard security technology for establishing encrypted communication between a web server and a browser.

It uses:

- asymmetric cryptography for authentication of exchange keys,
- symmetric encryption to maintain confidentiality



Learn more about SSL here <http://asp.net.by/en/SSL/>

# Bootstrap

Bootstrap (also known as Twitter Bootstrap) is a free set of tools for creating websites and web applications [getbootstrap.com]. Includes HTML and CSS design templates for typography, web forms, buttons, tags, navigation blocks and other web interface components, including JavaScript extensions

See more.: <http://asp.net.by/en/Bootstrap/>

# Angular JS

AngularJS is a JavaScript framework. It can be added to the HTML page in the <script> tag.



AngularJS extends HTML attributes with directives, and associates HTML with expressions.

Example: <http://asp.net.by/Angular/01.html>

```
<!DOCTYPE html>
<html lang="en-US">
<script src="http://ajax.googleapis.com/ajax/libs/angularjs/1.4.8/angular.min.js"></script>
<body>

  <div ng-app="">
    <p>Name :

    <input type="text" ng-model="name">
  </p> <h1>Hello {{name}}</h1>
  </div>
</body>

</html>
```

Input something in the input box:

Name :

**Hello Minsk**

See more.: <http://asp.net.by/en/Angular/>



# React.JS

React (sometimes React.js or ReactJS) is an open source JavaScript library for developing user interfaces.

React is developed and maintained by Facebook, Instagram, and the community of individual developers and corporations.

Example: <http://asp.net.by/React/04.htm>

# 21:15:08

```
<div id="root"></div>

<script type="text/babel">

function tick() {

const element=(<i>{new Date().toLocaleTimeString()}</i>);

ReactDOM.render(element, document.getElementById('root')); }

setInterval(tick, 3000);

</script>
```

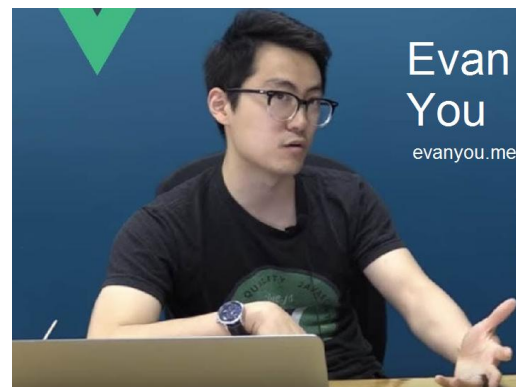
See more.: <http://asp.net.by/en/React/>

# Vue.JS

Vue (pronounced / vju: /, much like view) is a progressive

framework for creating user interfaces.

The creator of Vue.js is [Evan You](#), a former employee of Google and the Meteor Dev Group. He began to develop the framework in 2013, and in February 2014 the first public release took place.



Vue is widely used among Chinese companies, for example: Alibaba, Baidu, Xiaomi, Sina Weibo etc. It is part of the core of Laravel and PageKit.

Recently, the free GitLab repository management system has also switched to Vue.js. Более подробно о Vue.JS

See more.: <http://asp.net.by/en/Vue/>

# BackEnd

The main elements of the classical scheme of the Internet are a WEB browser (Web Client) and a www Server.

Под BackEnd'ом понимают код, выполняемый сервере (или «серверная часть»). Сервер часто физически удален от пользователя.

BackEnd – is the code, executed by the server (or “server part”). The server is often physically removed from the user.

The browser and server interact as follows:

1. The browser forms a request to the server using the HTTP protocol rules.

Typically, the browser requests an HTML page, that is, a text file containing HTML code.

2. The server analyzes the browser request and extracts the required file from the local storage.

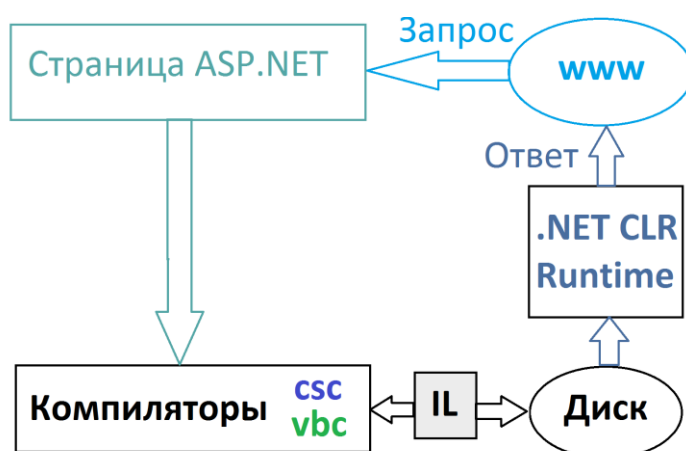
3. The server generates an HTTP response that includes the required information and sends it to the browser via HTTP.

4. The browser converts the HTML code into an image.

In ASP.NET, HTML is generated on the server just before being sent to the client.

To create client-server applications, Microsoft developed ASP.NET technology.

ASP.NET is built on the Common Language Runtime (CLR), which allows programmers to write ASP.NET code using any supported .NET language, such as C # and VB.NET.



An ASP.NET web page or web page, officially known as Web Forms, is the main building block for application development.

Web forms are contained in files with the extension “.aspx”.

See more.: <http://asp.net.by/en/BackEnd/>

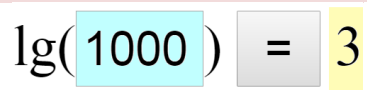
# Web Forms

In ASP.NET Web Forms, server-side code is included directly in the HTML syntax, similar to PHP.

Pages have an extension aspx.

```
<script runat="server">
    protected void ButtonLOG_Click(object sender, EventArgs e)
    {
        var X = int.Parse(TextBox_X.Text);
        LabelResult.Text = System.Math.Log10(X).ToString();
    }
</script>

<html>
<body>
    <form id="form1" runat="server">
        <div>
            lg(<asp:TextBox ID="TextBox_X" runat="server" Width="32px">1000</asp:TextBox>
            <asp:Button ID="ButtonLOG" runat="server" Text="" OnClick="ButtonLOG_Click" />
            <asp:Label ID="LabelResult" runat="server" Text="Label"></asp:Label>
        </div>
    </form>
</body>
</html>
```



It is possible to place the code in a separate file \*.aspx.cs

01.aspx

```
<%@ Page Language="C#"
AutoEventWireup="true" CodeFile="01.aspx.cs"
Inherits="_01" %>

<!DOCTYPE html>

<html>
<body>
    <form id="form1" runat="server">
        <div>
```

01.aspx.cs

```
protected void
    ButtonLOG_Click(object
        sender, EventArgs e)
    {
        var X =
int.Parse(TextBox_X.Text);
        LabelResult.Text =
            Math.Log10(X).
                ToString();
    }
```

```
lg(<asp:TextBox ID="TextBox_X" runat="server" Width="32px">1000</asp:TextBox>
<asp:Button ID="ButtonLOG" runat="server" Text="" OnClick="ButtonLOG_Click" />
<asp:Label ID="LabelResult" runat="server" Text="Label"></asp:Label>
```

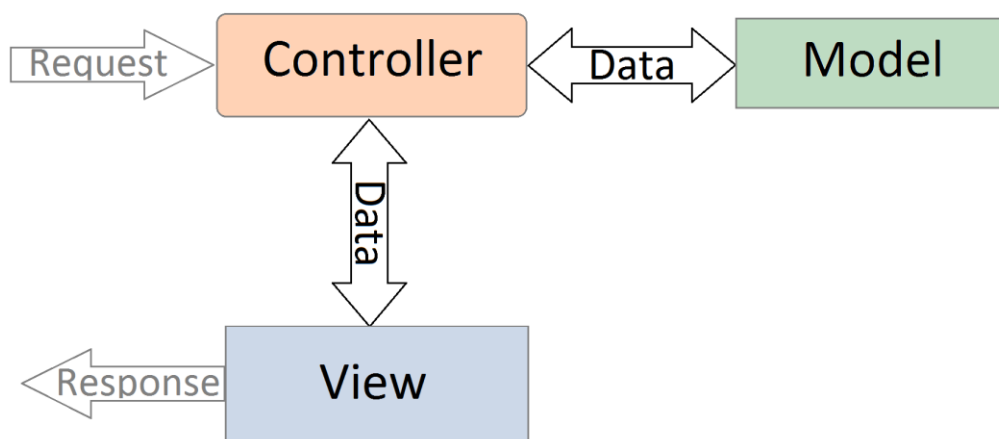
```
</div>  
</form>  
</body>  
</html>
```

See more.: <http://asp.net.by/en/Web%20Forms/>

# MVC

In 2007, Microsoft announced a new web development platform - MVC, based on ASP.NET.

The user's work with the MVC application is as follows: the user takes an action in response to which the MVC application changes its data model and delivers an updated view to the user. Then the cycle repeats. This fits well with the web application schema provided in the form of HTTP request and response sequences.



Model-View-Controller (MVC) — is a scheme for dividing application data, user interface and control logic into three separate components: model, view and controller, so that each component can be modified independently.

The Model provides data and responds to controller commands by changing its state.

The View is responsible for displaying model data to the user, responding to changes in the model.

The Controller interprets the user's actions, alerting the model to changes.

The ASP.NET MVC Framework implements the MVC template while providing a significantly improved division of responsibility. In fact, ASP.NET MVC implements a modern version of MVC, which is particularly suitable for web applications.

See more: <http://asp.net.by/en//MVC/>

# ASP.NET Core

The ASP.NET Core platform is a technology from Microsoft designed to create various kinds of web applications: from small websites to large web portals and web services.

ASP.NET Core is an extension of the ASP.NET platform.

See more: <http://asp.net.by/en/Core/>

# VB.NET

VB.NET or Visual Basic.NET is currently one of the most popular programming languages.

Although it is inferior in popularity to such languages as C++, C#, Java for various reasons, it is not inferior in its capabilities and potential to the above mentioned languages.

One of the main features of VB.NET is its object-oriented nature. VB.NET is a full-fledged object-oriented language. It supports polymorphism, inheritance, static typing and operator overload.

The key difference between VB.NET and the classic Visual Basic is the use of the .NET platform. The VB.NET language was created specifically for the .NET platform.

The environment supports a number of VB.NET, C#, C++, F# languages, as well as various dialects of other languages bound to .NET, for example, Delphi.NET.

The developer can choose the language that suits him best.

This mechanism works thanks to the Common Language Runtime (CLR) environment that underpins the .NET platform. When compiling code in any of these languages, it is



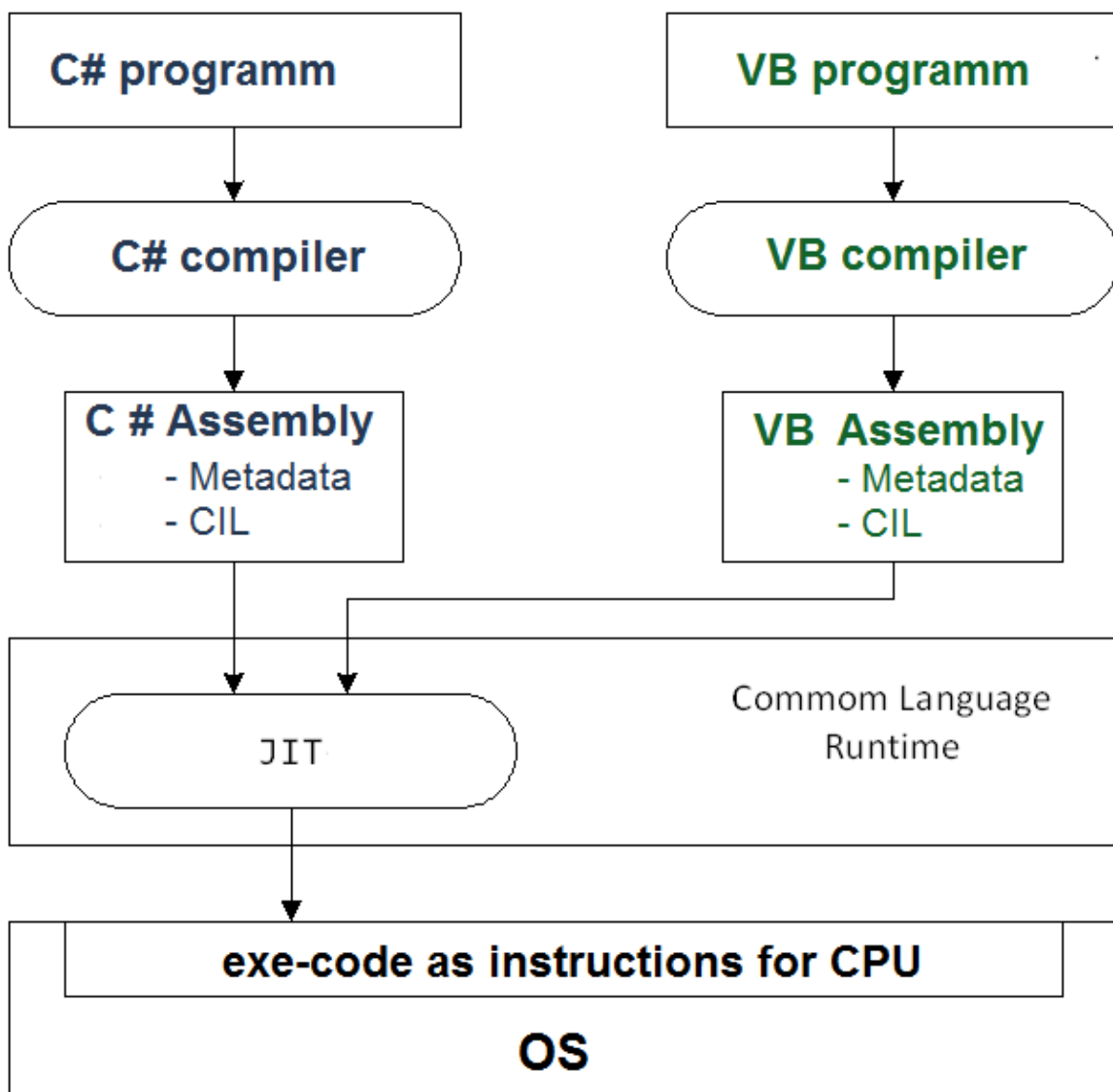
compiled into a Common Intermediate Language (CIL) build. The CIL language is a kind of assembler for the .NET platform.

As it was noted above, the code in VB.NET is compiled into applications or assemblies with .exe or .dll extensions in Common Intermediate Language. Then, when the application is launched, it is compiled (Just-In-Time) into machine code, which is already being executed directly. Since our application can be large and contain a lot of instructions, only the part of the application that is directly accessed will be compiled at one point in time. When accessing another part of the code, it will also be compiled from CIL into machine code. Thus already compiled part of the appendix remains before the end of work of the program. As a result, it increases productivity.

See more: <http://asp.net.by/en/VB.NET/>

## C#

C# (pronounced si sharpe) is an object-oriented programming language developed by Microsoft under the guidance of Anders Halesberg and Scott Wiltaumot as the main application development language for the Microsoft .NET Framework.



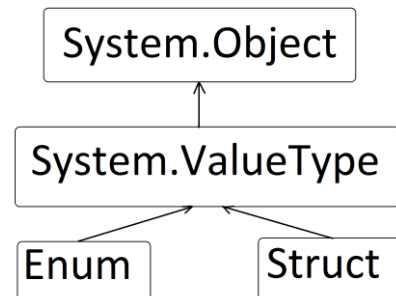
The C # compiler compiles a program written in C # into intermediate code - Common Intermedia Language (IL) or Intermedia Language (IL) - which is a high-level object-oriented assembler.

This code is interpreted by the JIT compiler (Just-in-Time Compiler) in the CPU instruction.

This approach allows you to develop different parts of the project in different languages.

All types in the .NET Framework are inherited (directly or indirectly) from the System.Object class (C# uses the object alias for this type).

The System.ValueType type is the ancestor of all types of values (including numeric types, custom structures and enumerations).

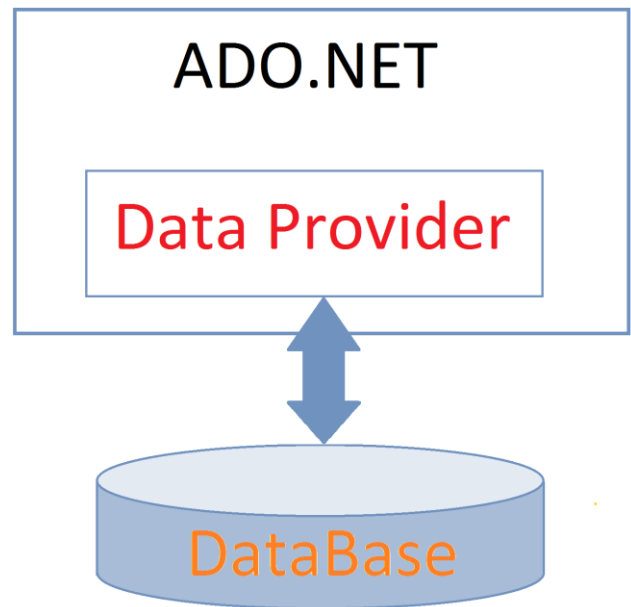


See more.: <http://asp.net.by/en/cs/>

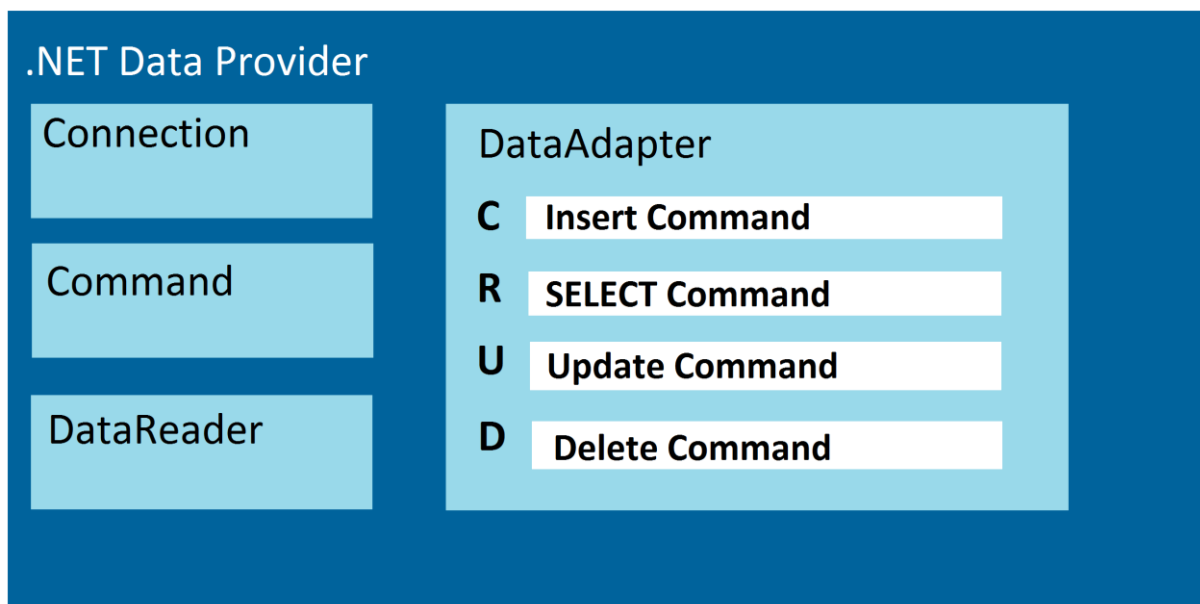
# ADO.NET

ADO.NET - a set of classes that represent the basic functionality for working with databases and other sources that store information (Microsoft SQL Server, Microsoft Access, Microsoft Excel, Microsoft Outlook, Microsoft Exchange, Oracle, OLE DB, ODBC, XML, text files ).

It is possible to work autonomously (disconnected environment) using DataSet objects, which are copies of tables with links.



The DataSet object allows you to work with the database in the disconnected state, and send back the received data using the data adapter.



Access to ADO.NET is possible from C #, VB.NET and from any .NET language.

ADO.NET classes are contained in the System.Data.dll file.

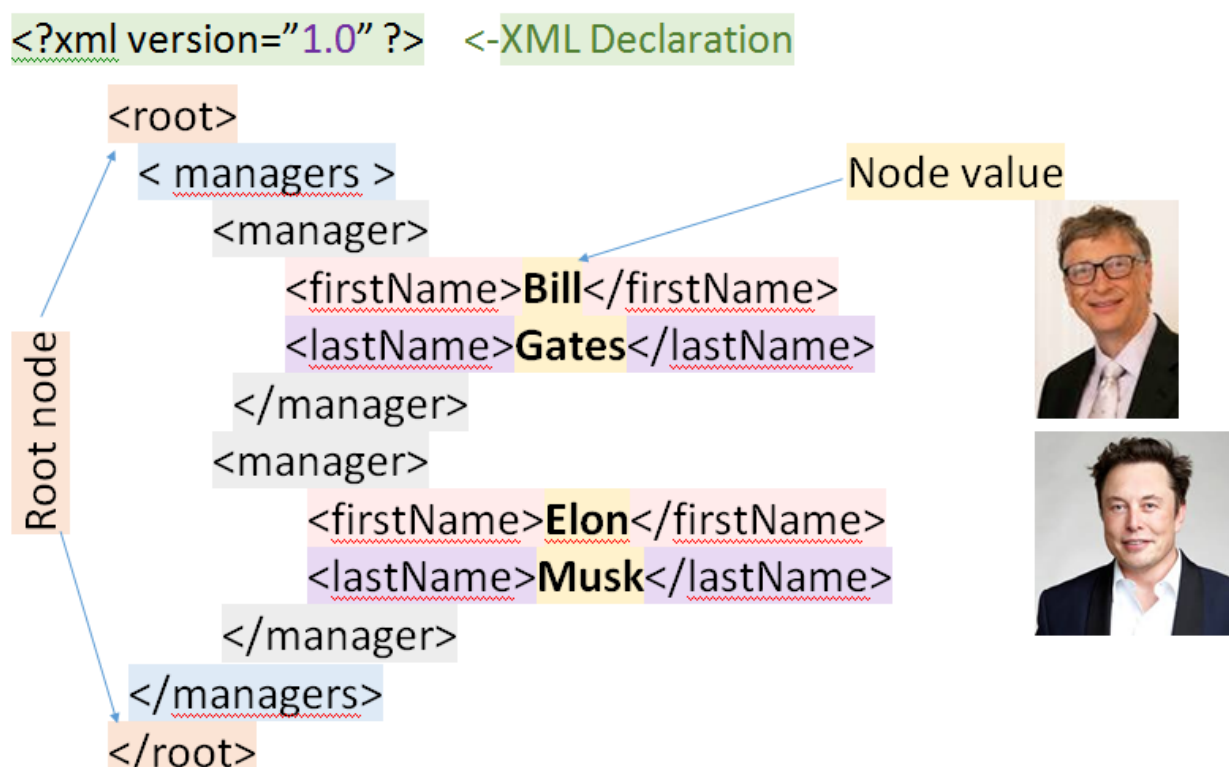
For more on ADO.NET see: <http://asp.net.by/en/ADO.NET/>



# XML

XML (eXtensible Markup Language) is a way of describing structured data. Structured data is data that has a defined set of semantic attributes and can be described hierarchically. XML data is contained in a document that can be a file, stream or other information repository capable of supporting a text format.

Any XML-document is built according to certain rules.

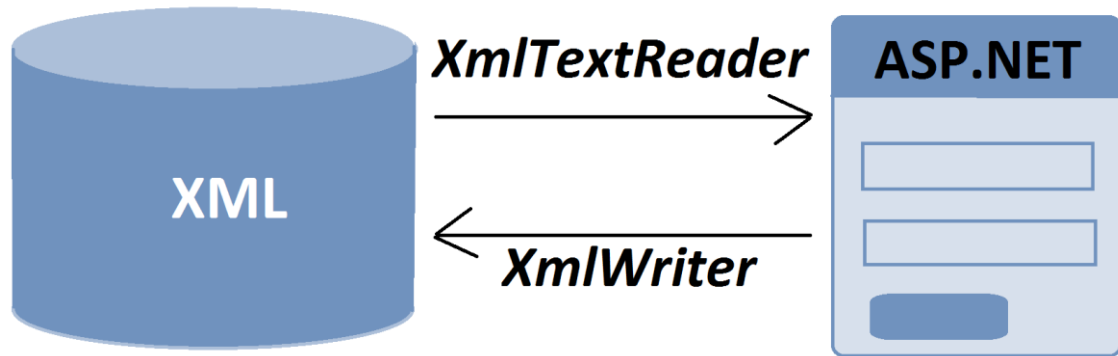


To associate an element with a namespace, a special xmlns attribute is used. Typically, a Uniform Resource Identifier (URI) is used for a namespace identifier to reduce the risk of identity matching between documents.

The System.Xml namespace provides basic functionality for working with XML-files in ASP.NET.

The **XmlWriter** class is a base class that provides a read-only process to generate an XML stream. It provides methods for writing data to an XML document.

**XmlTextReader** is the next class for reading data from an XML document.



WEB.config configuration files use XML.

Site maps for Google are also written in XML.

An example of a fragment of a sitemap SiteMap.XML:

```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
<url>
<loc>http://sunny.gear.host/web-technologies/</loc>
<priority>0.5</priority>
<lastmod>2019-11-10T20:20:33+00:00</lastmod>
<changefreq>daily</changefreq>
</url>
.....
</urlset>
```

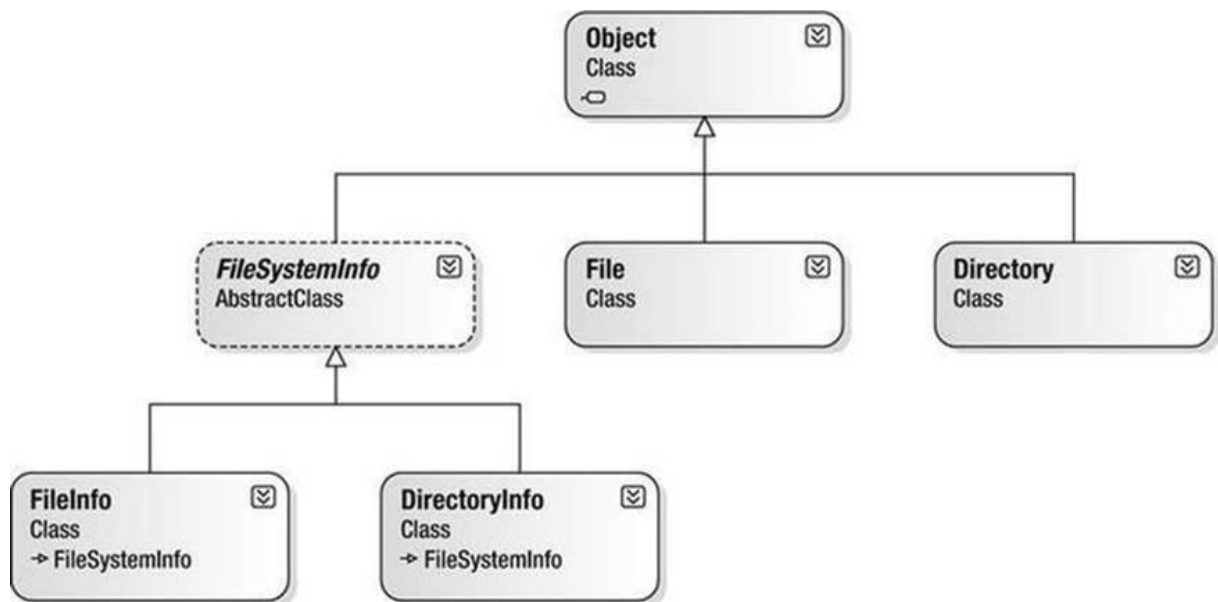
See more: <http://asp.net.by/en/XML/>

# WORKING WITH FILES IN ASP.NET

To perform all these operations, there are several classes located in the "System.IO" namespace.

2 File and FileInfo classes, which are designed to work with the file as part of a file system.

There are several methods that allow you to work with the entire file content.



The Directory, File, DirectoryInfo and FileInfo classes are designed for work with directories and files. The first two classes perform operations using static methods, the second two - using instance methods.

Example.

<http://asp.net.by/Files/Copy/>



```
protected void btnFileCopy_Click(object sender, EventArgs e)
{
    string siteLocation = Request.PhysicalApplicationPath + "\\Files\\Copy";
    string fileToCopy = Path.Combine(siteLocation, "Default.aspx");
```

```

string folderLocation = Path.Combine(siteLocation, "copiedFiles");
string newLocation = Path.Combine(folderLocation, "Default.aspx");
if (Directory.Exists(folderLocation))
{
    if (File.Exists(fileToCopy))
    {
        File.Copy(fileToCopy, newLocation, true);
        Label1.Text = "<font color='green'>File copied to folder</font>";
    }
    else
    {
        Label1.Text = "<font color='red'>No such file</font>";
    }
}
else
{
    Label1.Text = " <font color='red'>No such directory</font>";
}
}

```

The above code checks if the directory exists

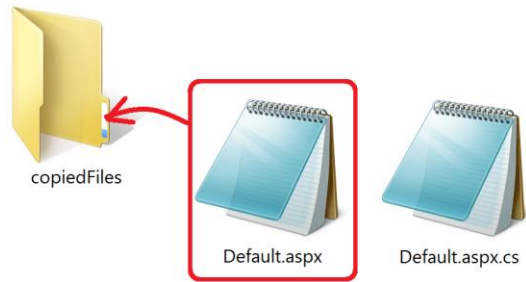
**if (Directory.Exists(folderLocation))**



If so, then we check if the file we are going to copy exists

**if (File.Exists(fileToCopy))**

If everything is in order, then we go ahead and copy the file



**File.Copy(fileToCopy, newLocation, true);**

If the file to be copied is already in the folder, set the override value to **true** will replace him. Default value – **false**.

You will get an error if the file you are copying already exists.

To move a file to a new location, you must use the **Move** method of the File class:

**File.Move(fileToMove, newLocation);**

Everything else is the same:

In the **.Move** method, enter the file you want to move **fileToMove** preceded by a comma.

Then, after the comma, add the new location **newLocation**.

To delete a file from your computer, you can use the **Delete** method of the File class:

**File.Delete(fileToDelete);**

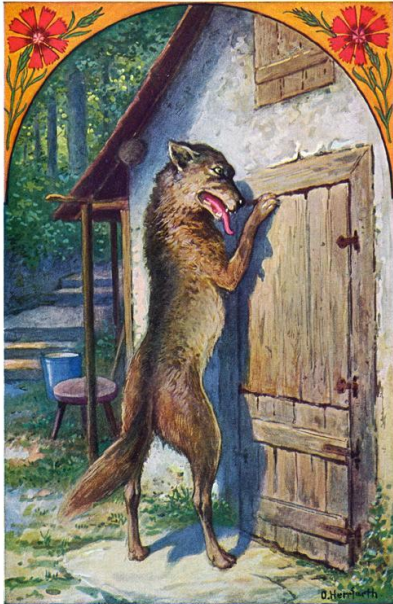
Between the parentheses of Delete, you need to specify **fileToDelete** - which consists of the name and path to the file that you are trying to get rid of. Care must be taken when trying this because File.Delete really deletes the file.

And the deleted file cannot be found in the recycle bin.

See more.: <http://asp.net.by/en/Files/>

# CONFIGURATION AND

# SECURITY



Brüder Grimm · Der Wolf und die sieben Geißlein · O. Herfurth jun.

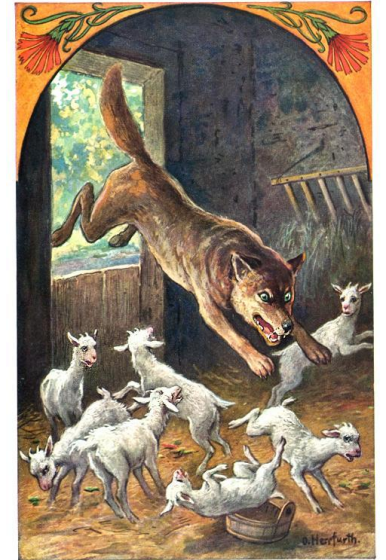
## Authentication

### Authentication

is the process of identifying application users.

### Authorization

is the process of granting access to users based on their identification data.



Brüder Grimm · Der Wolf und die sieben Geißlein · O. Herfurth jun.

## Authorization

ASP.NET together with a web server provides several possible types of authentication:

- Windows (default),
- Forms,
- Passport
- None.

The choice of type determines the mechanism of storage of tokens of the authenticated user. In case of Windows type the marker is placed in a context of a stream of ASP.NET working process. If Forms type is used, the marker is transferred from the client to a server and back in a cookie-file.

You can set authentication requirements for web application clients by adding the

The type of client authentication is defined using the <authentication> element, the mode attribute of which can take one of four values: Windows, Forms, Passport and None. Once you have configured the type of authentication, you need to think about the purposes for which you want to authenticate. Authentication is performed to restrict clients' access to the entire application or parts of it using the <authorization> element.

The <allow> and <deny> sub-elements are added to this element to specify whether or not individual users and roles should be granted access.

A metacharacter \* is used to represent all users, huh? - to represent anonymous users. The following example of a configuration file denies anonymous users access to the site:

```
<configuration>
<system.web>
  <authorization>
    <deny users="*" />
  </authorization>
</system.web>
</configuration>
```

The <allow> and <deny> elements support three attributes: users, roles and verbs. The values of these attributes can be comma separated lists of users, roles and commands. It is possible to define individual security settings for the selected subdirectory of the web application.

In the root, WEB-config should look like this:

```
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms loginUrl="LoginPage.aspx">
        <credentials passwordFormat="Clear">
          <user name="Andrey" password="651652"/>
          <user name="Andy" password="651"/>
        </credentials>
      </forms>
    </authentication>
    <compilation debug="true" targetFramework="4.0"/>
  </system.web>
</configuration>
```

The `LoginPage.aspx` file looks like this:

**Log In**

User Name:

Password:

```
<html>
  <body>
    <form runat="server">
      <asp:Button Text="Log In" OnClick="OnLogin" RunAt="server" />
      User Name: <asp:TextBox ID="UserName" runAt="server" />
```

**Log In**

User Name:

```
<font color="red">
  <asp:Label ID="I_Login" RunAt="server" />
</font>
```

**<= Invalid login**

```
<font color="red">
  <asp:Label ID="I_or" runat="server" Text="">
</font>
```

**<= Invalid password**

Password:

Password:

```
<asp:TextBox ID="Password" TextMode="password"
  RunAt="server" />
<asp:Label ID="I_Password" runat="server"
  Text="" ForeColor="Red"></asp:Label>
```

```
</form>
```

```
<script language="C#" runat="server">
  void OnLogin (Object sender, EventArgs e)
  {
    if (FormsAuthentication.Authenticate(UserName.Text, Password.Text))
      FormsAuthentication.RedirectFromLoginPage(UserName.Text, false);
    else
    {
      I_Login.Text = "<= Invalid login";
      I_or.Text="or";
      I_Password.Text="<= Invalid password";
    }
  }
</script>
```

See more: <http://asp.net.by/en/Config/>





Andrey O. Yaroshevich

**School of Business**

of the **Belarusian State University**

e-mail: [ao@asp.net.by](mailto:ao@asp.net.by)

yaroshevich.com

asp.net.by

