ERASMUS+
Enriching lives, opening minds

Беларускі Дзяржаўны Універсітэт

WEB-тэхналогіі

IESED

SCHOOL OF BUSINESS OF BSU

Увядзенне

Тэхналогіі World-Wide Web першапачаткова спраектаваныя ў CERN (European Organization for Nuclear Research), як сродак дастаўкі дакументаў з адной навуковай установы ў іншую, цяпер выкарыстоўваюцца як платформа для складаных інтэрактыўных прыкладанняў, якія павольна, але дакладна выцясняюць усталёўваныя прыкладанні (installable applications).

Гэтаму спрыяюць перавагі, якімі валодаюць WEB-прыкладанні:

- пастаянны доступ з розных прылад,
- аўтаматычнае абнаўленне,
- магчымасць інтэграцыі розных прыкладанняў, напісаных для розных платформаў.

Аднак, стварэнне WEB-прыкладанняў патрабуе некалькі іншых падыходаў, чым стварэнне традыцыйных усталёўваных прыкладанняў і на сённяшні дзень немагчыма без інтэграцыі самых розных падыходаў і тэхналогій.

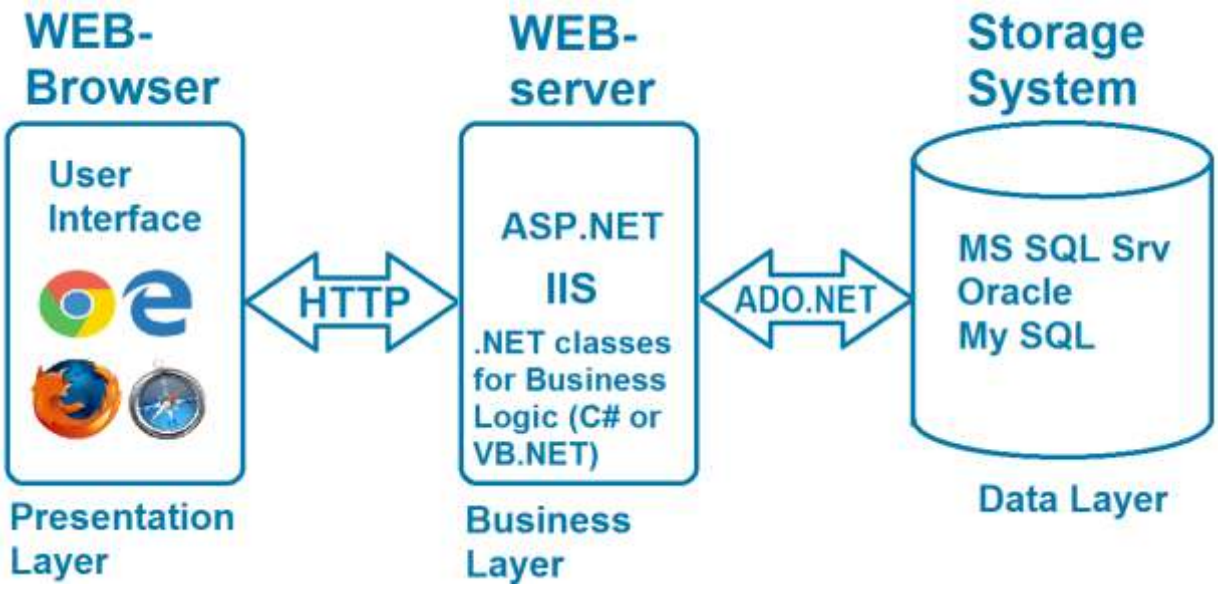
Дадзены курс знаёміць гэтымі рознымі WEB-тэхналогіямі і дае навыкі стварэння WEB-прыкладанняў. Студэнты вывучаюць тэарэтычна і атрымліваюць практычныя навыкі з мовамі разметкі, скрыптавымі мовамі, сеткавымі пратаколамі, графікай, праграмаваннем на аснове падзей, аб'ектна-арыентаваным праграмаваннем, базамі даных.

Г.зн. тыя тэхналогіі, якія дазваляюць будаваць сучасныя WEB-прыкладанні.

Ёсць цяжкасць з выбарам тэхналогій WEB-праграмавання - т.я. у гэтай галіне змены адбываюцца найболей хутка. Біл Гейтс у сваёй кнізе "Business@the Speed of Thought" казаў, што Microsoft практычна цалкам абнаўляе гэтыя тэхналогіі раз у 3 гады. І курс даводзіцца абнаўляць вельмі часта - тыя тэхналогіі, якія з'яўляюцца мэйнфрэймавымі сёння, праз год могуць страціць свае пазіцыі.

Аднак нешта застаецца адносна ўстойлівым - канцэпцыі праграмавання і архітэктура прыкладанняў.

Мы зыходзім з пазіцыі, што разуменне канцэпцый і архітэктуры важней ведаў канкрэтных метадаў кадавання, якія пераживаюць змена можа быць хутчэй, чым хацелася б.



Архітэктура WEB-прыкладанні звычайна ўключаюць наступныя складовыя часткі:

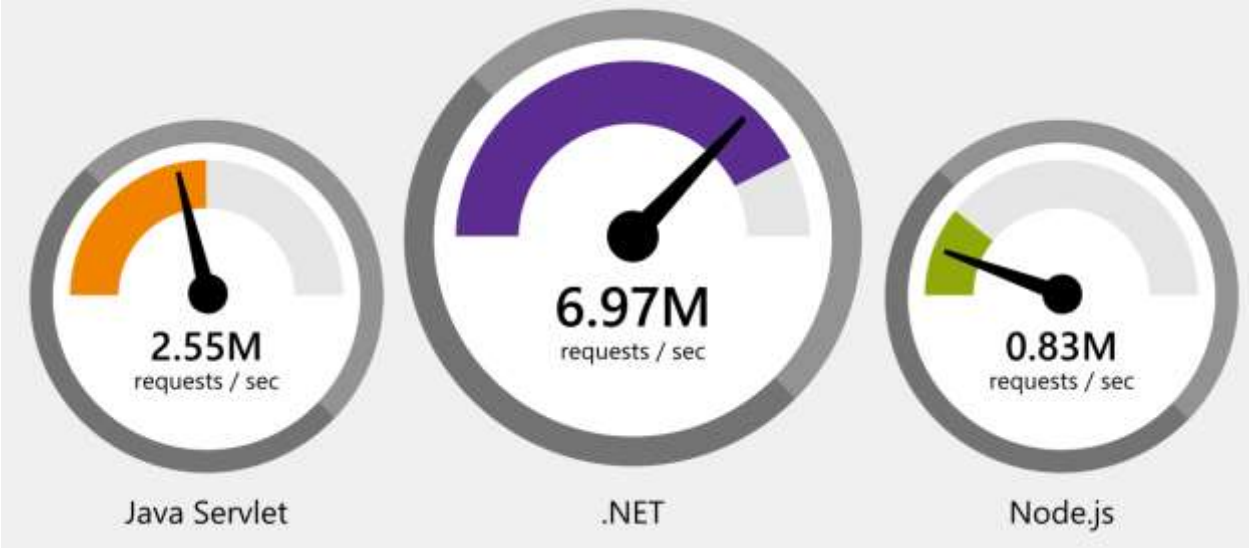
- Інтэрфейс прыкладання ў Web Browser'e (Front End)
- Серверная частка (Back End)
- Сістэма захоўвання дадзеных (Data Storage)

На ўзроўні інтэрфейсу (Presentation Layer) вывучаюцца

HTML/CSS/JavaScript і Document object Model (DOM) - Document structure Browser software а таксама канцэпцыі Model View Controller, Single page applications (фрэймворкі Angular JS і Vue.JS).

На ўзроўні сервера (Business Layer) вывучаецца тэхналогія ASP.NET на базе C# і VB.NET.

На ўзроўні дадзеных (Data Layer) вывучаецца тэхналогія ўзаемадзеяння з БД ADO.NET. Такі выбар абумоўлены тым, што на сённяшні дзень платформа ASP.NET працуе хутчэй, чым любы папулярны вэб-фрэймворк (ASP.NET).



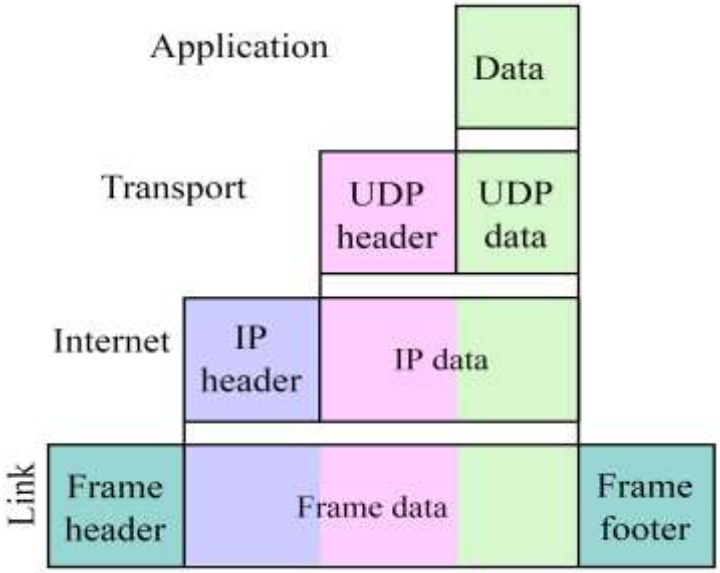
Калі праз год ці 2 сітуацыя зменіцца і Node.js перасягне ASP.NET па хуткасці ці папулярнасці, то на ўзроўні сервера можна будзе вывучаць Node.js.

А пакуль мы прадстаўляем тое, што нам уяўляецца лепшым і найбольш карысным для студэнтаў на сённяшні дзень..

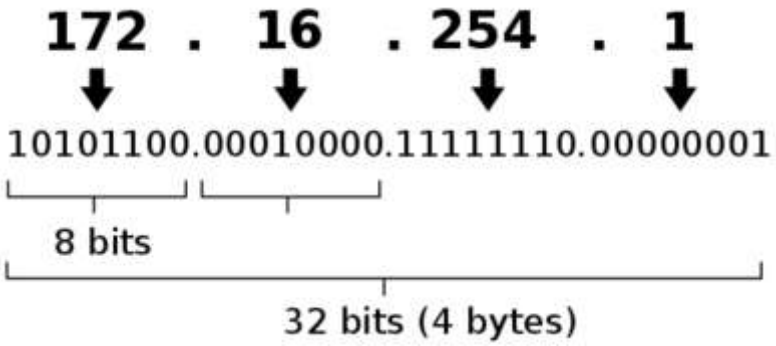
Базавыя канцэпцыі WEB-тэхналогій

DHCP (англ. Dynamic Host Configuration Protocol - пратакол дынамічнай налады вузла) - сеткавы пратакол, які дазваляе сеткавым прыладам аўтаматычна атрымліваць IP-адрас і іншыя параметры, неабходныя для працы ў сетцы TCP/IP

TCP/IP — сукупнасць стандартаў і правіл, якія вызначаюць, як дадзеныя перадаюцца па сетцы.



IP адрес - Internet Protocol Address — унікальны ідэнтыфікатар (адрас) прылады,



падлучанага да лакальнай сеткі ці інтэрнэту. IP-адрас уяўляе сабой 32-бітавае (па версіі IPv4).

Даменнае імя - больш зразумела, чым IP-адрас. Даменныя імёны даюць магчымасць адрасавання інтэрнэт вузлоў у зручнейшай для чалавека форме.

Кампутарная сістэма для атрымання інфармацыі аб IP-вузлах па імі хаста завецца Domain Name System (DNS).

З дапамогай утыліты ping можна пазнаць IP-адрас па даменным імі.

Утыліта tracert (traceroute у Apple) – гэта службовая кампутарная праграма, прызначаная для вызначэння маршрутаў прытрымлівання дадзеных у сетках TCP/IP.

Порт (англ. port) — натуральны лік, які запісваецца ў загалоўках пратаколаў транспартнага ўзроўню. З дапамогай порта вызначаецца атрымальнік пакета ў межах аднаго хаста. Порт паказваецца праз двукроп'е пасля адрасу прылады

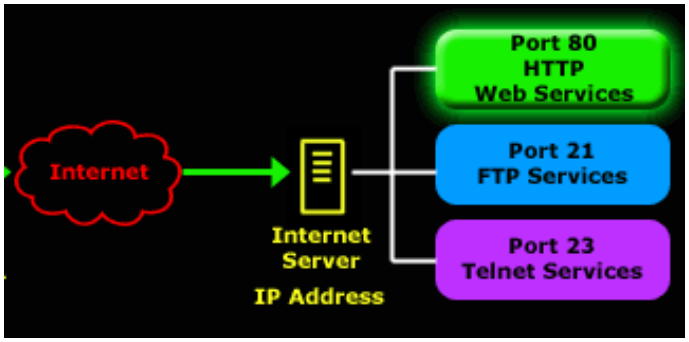
HTTP порт 80,

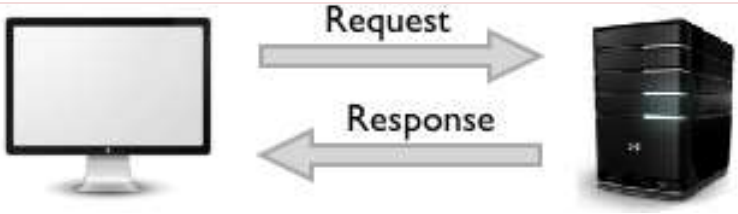
HTTPS — порт **443**.

FTP— порт 21.

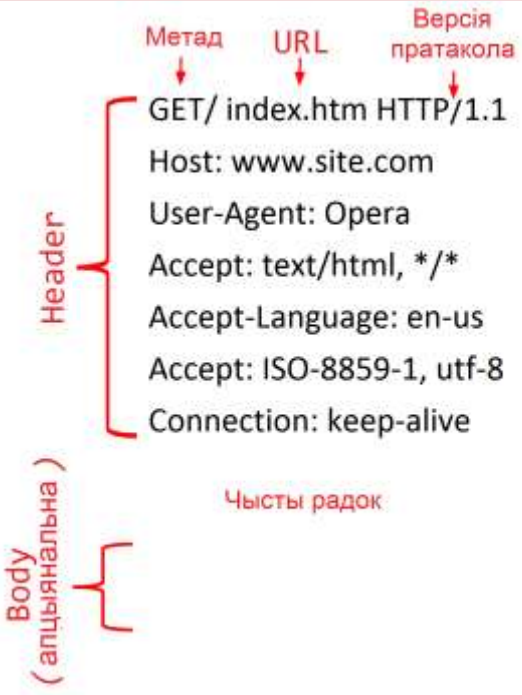
SMTP — порт **25**

SMTP — порт **23**





Структура HTTP-запыту



Структура HTTP-адказу

Коды стану

1xx: Інфармацыйныя паведамленні

100 continue, што азначае, што кліент яшчэ адпраўляе астатнюю частку запыту.

2xx: Паведамленні аб поспеху

200 OK

Калі кліент атрымаў код з серыі 2xx, то запыт сышоў паспяхова.

3xx: Перанакіраванне

301 Moved Permanently: рэсурс зараз можна знайсці па іншым URL адрасе.

4xx: Кліенцкія памылкі

404 Not Found. Рэсурс не знойдзены на серверы.

401 Unauthorized: для совершения запроса нужна аутентификация. Информация передаётся через заголовок Authorization.

403 Forbidden: сервер не открыл доступ к ресурсу.

5xx: Ошибки сервера

500 Internal Server Error.

503 Service Unavailable: это может случиться, если на сервере произошла ошибка или он перегружен.

Браўзэры

Прызначэнне WEB-браўзэра - чытанне дакументаў HTML і кампаноўка іх у візуальныя вэб-старонкі (з магчымасцю прайгравання музыкі і адлюстравання відэафайлаў). Браўзэр не адлюстроўвае HTML тэгаў, але выкарыстоўвае тэгі, каб інтэрпрэтаваць змест старонкі і фарматаваць тэкст патрэбным чынам. На сённяшні дзень існуюць наступныя вэб-браўзэры (у якіх неабходна правяраць Ваш WEB-дызайн, каб Ваш сайт адлюстроўваўся ў іх карэктна):

- Internet Explorer (от Microsoft)
- Firefox (ад Mozilla)
- Chrome (ад Google)
- Safari (ад Apple)
- Opera (Opera з Нарвегіі)

HTML

HTML уяўляе сабой мова гіпертэкставай разметкі. Ён з'яўляецца асноўным будаўнічым матэрыялам WEB-старонак.

HTML складаецца з тэксту і тэгаў, зняволеных у дужкі, якія паказваюць як адлюстроўваць тэкст і графіку.

HTML тэгі звычайна выкарыстоўваюць парамі <html></html>.

Першы тэг называецца адкрывальным тэгам, напрыклад <html> <script>

Другі - зачыняльным </html> </script> - які пазначае канец зоны дзеяння тэга. У прамежку паміж гэтымі тэгамі вэб-дызайнеры могуць дадаць тэкст, табліцы, выявы і г.д.

Прыклад HTML-старонкі:

```
<html>
 <body>
  <h1> Загаловак </h1>
   <p> Параграф </p>
 </body>
</html>
```

0.html

Заголовок

Параграф

Больш падрабязна глядзіце: http://asp.net.by/html/

CSS

CSS - Cascading Style Sheets, што азначае даслоўна "каскадныя табліцы стыляў". З дапамогай CSS апісваецца стыль усяго дакумента ці сайта ў цэлым, альбо яго асобных элементаў.

Стылі апісваюцца ў тэгу

<style>

</style>

Іх можна запісаць у асобны файл і загрузіць яго ў html старонку наступным чынам:

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

Больш падрабязна глядзіце: http://asp.net.by/css/

Адаптыўная разметка

Адаптыўная разметка - гэта такая разметка, якую можна наладзіць пад розныя памеры экранаў. Ажыццяўляецца стварэнне адаптыўнай разметкі з дапамогай медыя запытаў, якія з'явіліся ў спецыфікацыі CSS3 і ў сапраўдны момант падтрымліваюцца ўсімі асноўнымі браўзэрамі.

Неабходнасць адаптыўнай разметкі вызначаецца тым, што ў сапраўдны момант колькасць user'аў, якія выкарыстоўваюць мабільны трафік, дасягнула 60%. Мабільны трафік становіцца больш значным і ўладальнікі сайтаў мусяць лічыцца з гэтай статыстыкай. У першую чаргу сайт павінен быць аптымізаваны пад мабільныя прылады – Mobile First.

Прынцыпамі Mobile First з'яўляюцца:

1) Самае важны кантэнт павінен быць паказаны ў першую чаргу

2) Вэб сайт павінен быць лёгкім і грузіцца хутка

3).Дадатковая інфармацыя павінна грузіцца па патрабаванні user'а.

Больш падрабязна глядзіце: http://asp.net.by/mobile/

CSS FlexBox

Лічацца лепшым сродкам для стварэння адаптаванага сайта для невялікага экрана, нават калі памер экрана невядомы і (ці) дынамічны.

Больш падрабязна глядзіце: http://asp.net.by/mobile/

JavaScript

JavaScript уяўляе сабой аб'ектна-арыентаваная мова, створаная для стварэння дынамічных WEB-старонак.

Што дазваляе ствараць складаныя дынамічныя сайты.

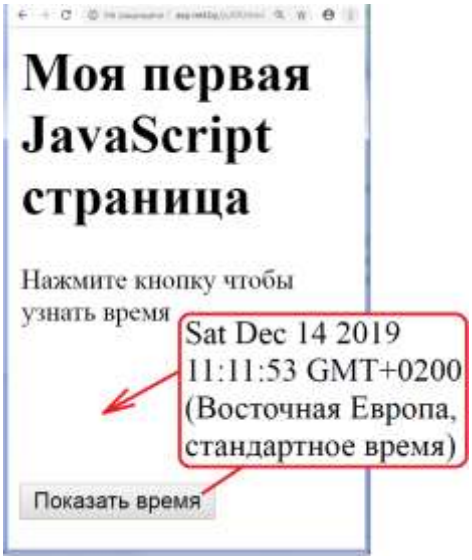
Вось прыклад простай HTML старонкі з JavaScript кодам:

```
<html>
<body>
<h1> Мая першая JavaScript старонка </h1>
<p> Націсніце кнопку каб даведацца які час
```

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

```
</p>
<p id="date_area"></p>
<button type="button"
 onclick="cmdDate()">
Паказаць час</button>
<script>
  function cmdDate()
  {
document.getElementById("date_area")
                              .innerHTML = Date();
  }
 </script>
 </body>
</html>
```

Показать время



ECMAScript быў створаны для стандартызацыі JavaScript.

Дазваляе выкарыстоўваць Java падобны сінтаксіс пры вызначэнні класаў:

```
<script>
class ACat {
  constructor(n) {
    this.name = n;        //свойство
  }
}
mycat = new ACat("Барсік");
document.write("Майго ката клічуць " + mycat.name);
</script>
```

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

Больш падрабязна глядзіце: http://asp.net.by/es6/

JSON

Механізм, які выкарыстоўваецца для перасылкі даных ад сервера да кліента.

Гэта фармат для абмену дадзенымі.

Правілы сінтаксісу JSON

- Дадзеныя ў парах імя / значэнне ("lastName":"**Musk**")
- Дадзеныя падзяляюцца коскамі ("firstName":"**Elon**", "lastName":"**Musk**")
- Фігурныя дужкі вызначаюць прадметы
  { "firstName":"**Elon**", "lastName":"**Musk**" }
- Квадратныя дужкі ўтрымоўваюць масівы {"managers":[.....]}

Прыклад:

XML		**JSON**
< managers >		{"managers":[
<manager> <firstName>**Bill**</firstName> <lastName>**Gates**</lastName> </manager>		{ "firstName":"**Bill**", "lastName":"**Gates**" },
<manager> <firstName>**Mike**</firstName> <lastName>**Dell**</lastName> </manager>		{ "firstName":"**Mike**", "lastName":"**Dell**" },

<manager>
<firstName>**Elon**</firstName>
<lastName>**Musk**</lastName>
</manager>
		{
		"firstName":"**Elon**",
		"lastName":"**Musk**"
		}

</managers>		]}

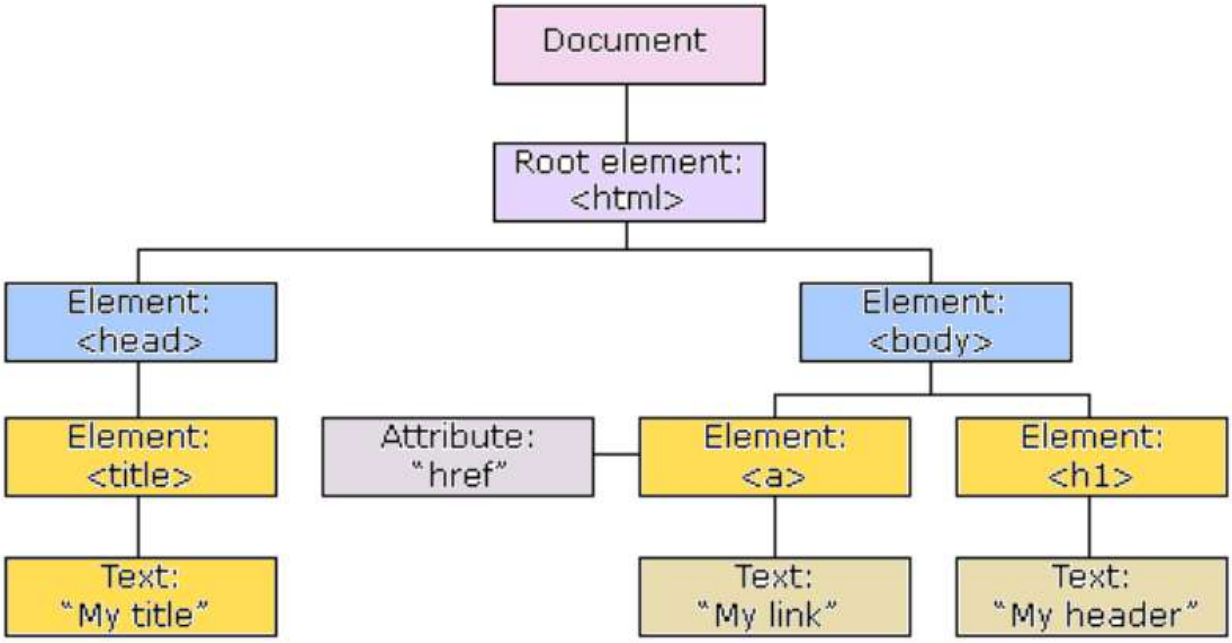
Фармат JSON перакладаецца ў фармат JavaScript object з дапамогай аператара:

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

Больш падрабязна аб фармаце JSON глядзіце: http://asp.net.by/JSON/

DOM

DOM - крос-платформавы і незалежны інтэрфейс, які ўяўляе XML або HTML дакумент у выглядзе дрэвападобнай структуры, у якой кожны вузел уяўляе сабой аб'ект, уяўлялыя сабой частка дакумента.



JavaScript можа запытваць ці змяняць HTML дакумент

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

Name: Ann Go

Name: Alex Go

Больш падрабязна аб DOM глядзіце: http://asp.net.by/**DOM/**

AJAX

AJAX, Ajax (Asynchronous Javascript and XML - "асінхронны JavaScript і XML"). Падыход да пабудовы інтэрактыўных user-інтэрфейсаў вэб-прыкладанняў, які складаецца ў "фонавым" абмене дадзенымі браўзэра з вэб-серверам.

У выніку, пры абнаўленні дадзеных вэб-старонка не перазагружаецца цалкам, і вэб-прыкладанні становяцца хутчэй і зручней

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

Больш падрабязна аб AJAX глядзіце: http://asp.net.by/**AJAX**/

jQuery

Набор функцый JavaScript, які факусуецца на ўзаемадзеянні JavaScript і HTML.

Бібліятэка jQuery дапамагае:

- лёгка атрымліваць доступ да любога элемента DOM *($("h1").show();)*
- звяртацца да атрыбутаў і зместу элементаў DOM *(p[0].value="Alex";)*
- маніпуляваць імі.

Бібліятэка jQuery дае зручны API для працы з AJAX.

Прыклад - http://asp.net.by/jQuery/01.html

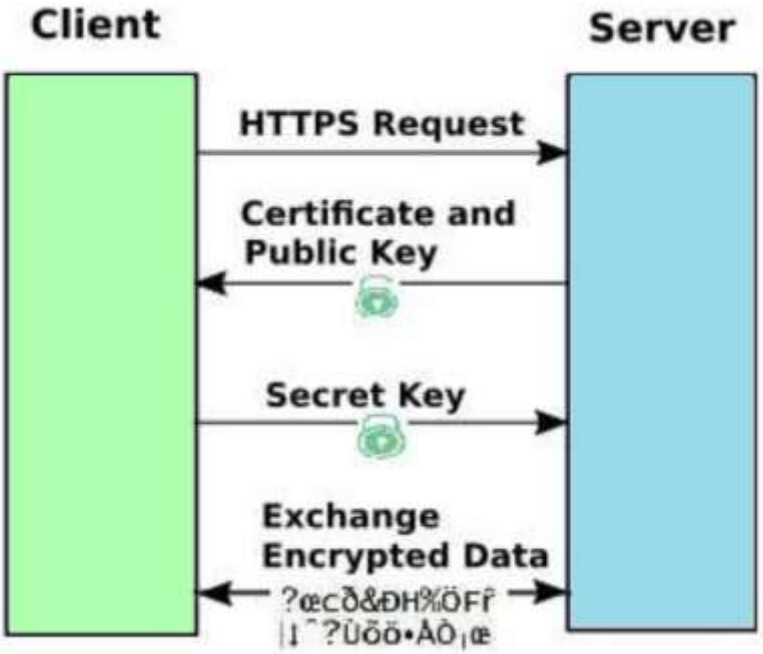
```
<html>
<head>
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js">
</script>
<script>
jQuery (document).ready(function(){
    $("#hide").click(function(){
        $("h1").hide();
    });
```

```
    $("#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>
```

Больш падрабязна аб jQuery гл.: http://asp.net.by/**jQuery/**

SSL

SSL (англ.: Secure Sockets Layer — узровень абароненых сокетаў) — крыптаграфічны пратакол, які мае на ўвазе больш бяспечную сувязь.



Ён выкарыстоўвае:

- асіметрычную крыптаграфію для аўтэнтыфікацыі ключоў абмену,
- сіметрычнае шыфраванне для захавання прыватнасці

Больш падрабязна аб SSL гл.: http://asp.net.by/**SSL**/

Bootstrap

Bootstrap (таксама вядомы як Twitter Bootstrap) - вольны набор інструментаў для стварэння сайтаў і вэбпрыкладанняў [getbootstrap.com]. Уключае ў сябе HTML-і CSS-шаблоны афармлення для друкаркі, вэб-формаў, кнопак, пазнак, блокаў навігацыі і іншых кампанентаў вэб-інтэрфейсу, уключаючы JavaScript-пашырэнні

Больш падрабязна аб Bootstrap гл.: http://asp.net.by/**Bootstrap**/

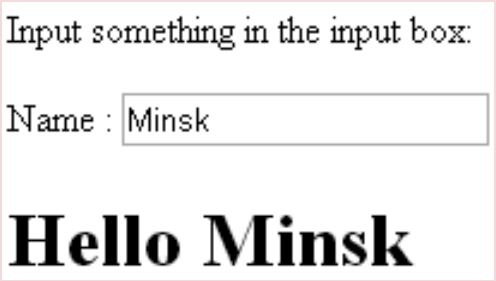
Angular JS

AngularJS – гэта **JavaScript framework**. Ён можа быць дададзены на HTML старонку ў <script> тэг.

AngularJS пашырае HTML аттрыбуты з дапамогай дырэктыў, і звязвае HTML з **выразамі**.

Прыклад 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>
```



Больш падрабязна аб Angular.JS гл.: http://asp.net.by/Angular/

React.JS

React (часам React.js або ReactJS) - JavaScript-бібліятэка з адкрытым зыходным кодам для распрацоўкі usars'кіх інтэрфейсаў.

React распрацоўваецца і падтрымліваецца Facebook, Instagram і супольнасцю асобных распрацоўшчыкаў і карпарацый

Прыклад гл. http://asp.net.by/React/04.htm

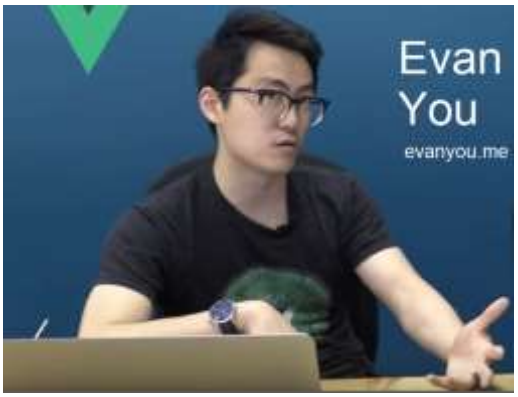

```
<div id="root"></div>

<script type="text/babel">

function tick() {

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

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

setInterval(tick, 3000);

</script>
```

Больш падрабязна аб React.JS гл.: http://asp.net.by/React/

Vue.JS

Vue (вымаўляецца /vjuː/, прыкладна як view) - гэта фрэймворк для стварэння user-інтэрфейсаў.

Стваральнікам Vue.js з'яўляецца Evan You, былы супрацоўнік Google і Meteor Dev Group. Пачаў ён распрацоўваць фрэймворк у 2013-м, а ў лютым 2014-га адбыўся першы публічны рэліз.



Vue шырока выкарыстоўваецца сярод кітайскіх кампаній, напрыклад: Alibaba, Baidu, Xiaomi, Sina Weibo і інш. Ён уваходзіць у ядро Laravel і PageKit.

Нядаўна свабодная сістэма кіравання рэпазітарамі GitLab таксама перайшла на Vue.js.

Больш падрабязна аб Vue.JS гл.: http://asp.net.by/Vue/

BackEnd

Асноўнымі элементамі класічнай схемы працы інтэрнэту з'яўляюцца WEB-браўзэр і www-сервер.

Пад BackEnd'ам разумеюць код, які выконваецца серверы (або «серверная частка»). Сервер часта фізічна выдалены ад user'а.

Браўзэр і сервер узаемадзейнічаюць наступным чынам:

1. Браўзэр фармуе запыт да сервера, выкарыстоўваючы пратакол HTTP.

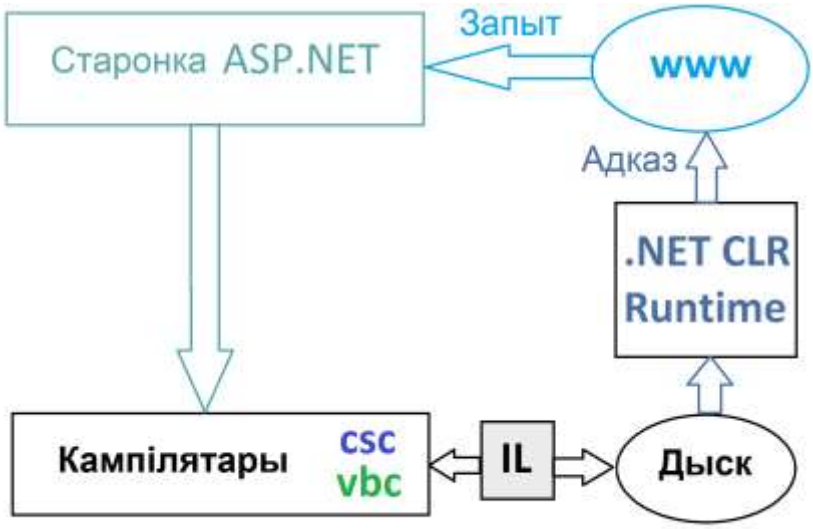
Як правіла, браўзэр запытвае HTML-старонку, гэта значыць тэкставы файл, які змяшчае HTML-код.

2. Сервер аналізуе запыт браўзэра і здабывае з лакальнага сховішча патрабаваны файл.

3. Сервер фармуе HTTP-адказ, які ўключае патрэбную інфармацыю, і адсылае яго браўзэру па пратаколе HTTP.

4. Браўзэр пераўтворыць HTML код у малюнак.

Для стварэння кліент-серверных прыкладанняў фірмай Microsoft была распрацавана тэхналогія ASP.NET. ASP.NET пабудавана на Common Language Runtime (CLR), што дазваляе праграмістам пісаць код ASP.NET з дапамогай любога які падтрымліваецца мова .NET, такія як C # і VB.NET.



ASP.NET вэб-старонка або вэб-

старонка, вядомая афіцыйна як Web Forms, з'яўляецца асноўным будаўнічым блокам для распрацоўкі прыкладанняў.
Вэб-формы змяшчаюцца ў файлах з пашырэннем ".aspx".

Больш падрабязна аб BackEnd гл.: http://asp.net.by/BackEnd/

Web Forms

У ASP.NET Web Forms серверны код уключаецца непасрэдна непасрэдна ў сінтаксіс HTML, аналагічна PHP.

Старонкі маюць пашырэнне 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>
```

lg(1000) = 3

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

Ёсць магчымасць размяшчэння кода ў асобным файле *.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();
    }
```

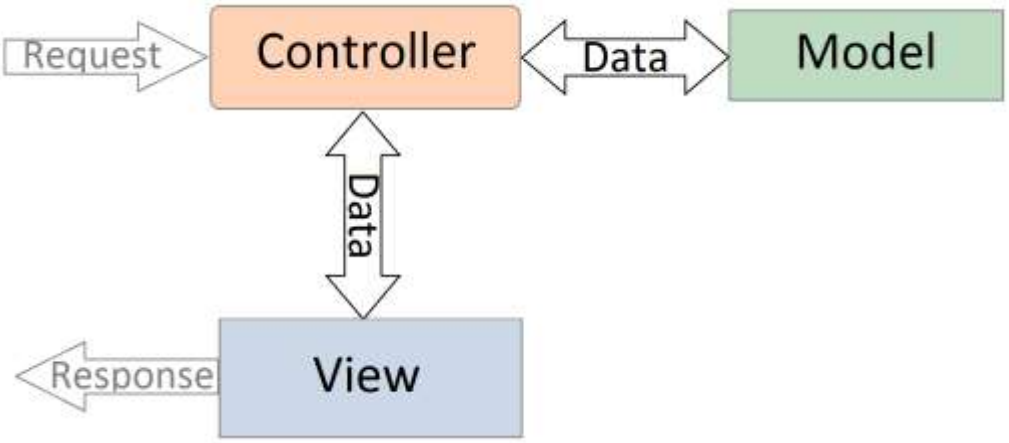
```
lg(<asp:TextBox ID="TextBox_X" runat="server" Width="32px">1000</asp:TextBox>)
<asp:Button ID="ButtonLOG" runat="server" Text="=" OnClick="ButtonLOG_Click" />
<asp:Label ID="LabelResult" runat="server" Text="Label"></asp:Label>
 </div>
</form>
</body>
</html>
```

Больш падрабязна аб Web Forms гл.: http://asp.net.by/Web Forms/

MVC

У 2007 г. у Microsoft анансавалі новую платформу вэб-распрацоўкі – MVC, пабудаваную на аснове ASP.NET.

Праца user'а з дадаткам MVC ажыццяўляецца наступным чынам: user прадпрымае дзеянне, у адказ на якое MVC-дадатак змяняе сваю мадэль дадзеных і дастаўляе абноўленае ўяўленне user'у. Затым цыкл паўтараецца. Гэта добра ўкладваецца ў схему вэб-прыкладанняў, якія прадстаўляюцца ў выглядзе паслядоўнасцяў запытаў і адказаў HTTP.



Model-View-Controller (MVC, "Мадэль-Прадстаўленне-Кантролер", "Мадэль-Від-Кантролер") - схема падзелу дадзеных прыкладання, users'кага інтэрфейсу і кіравальнай логікі на тры асобных кампанента: мадэль, прадстаўленне і кантролер - такім чынам, што мадыфікацыя кожнага кампанента можа ажыццяўляцца незалежна.

Мадэль (Model) падае дадзеныя і рэагуе на каманды кантролера, змяняючы свой стан.

Прадстаўленне (View) адказвае за адлюстраванне дадзеных мадэлі user'у, рэагуючы на змены мадэлі.

Кантролер (Controller) інтэрпрэтуе дзеянні user'а, апавяшчаючы мадэль аб неабходнасці змен.

Больш падрабязна аб MVC гл.: http://asp.net.by/MVC/

VB.NET

VB.NET або Visual Basic.NET у цяперашні час з'яўляецца адной з найбольш папулярных моў праграмавання.

Хоць ён саступае па папулярнасці такім мовам, як C++, C#, Java у сілу розных чыннікаў, аднак па магчымасцях і свайму патэнцыялу вышэй пералічаным мовам мала ў чым саступае.

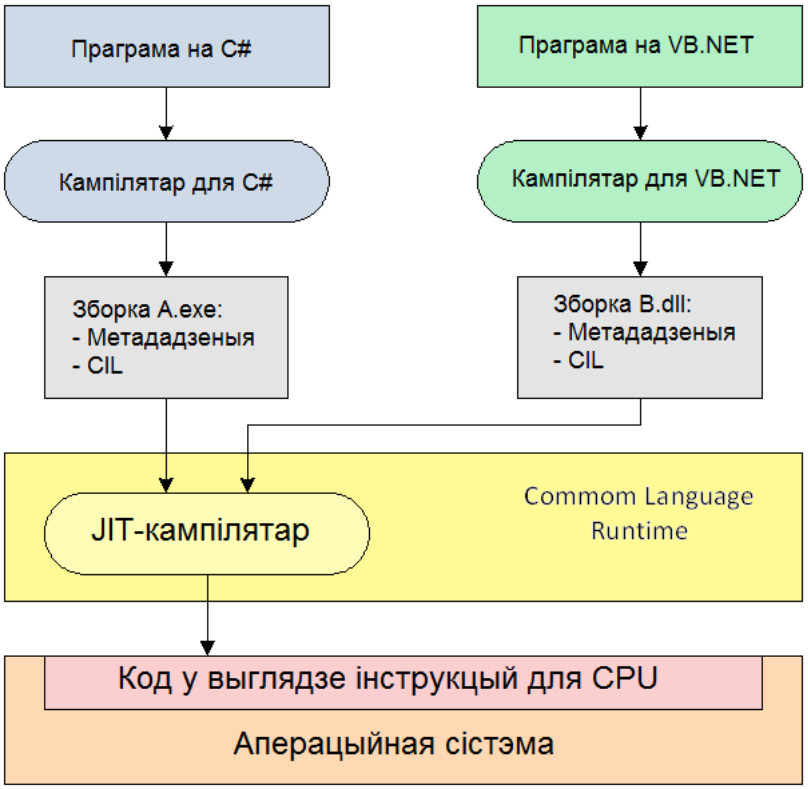
Адной з асноўных асаблівасцяў VB.NET з'яўляецца яго аб'ектна-арыентаванасць. VB.NET - паўнавартасная аб'ектна-арыентаваная мова.

Ён падтрымлівае палімарфізм, атрыманне ў спадчыну, статычную тыпізацыю, перагрузку аператараў.

Больш падрабязна аб VB.NET гл.: http://asp.net.by/VB.NET/

C#

C# (вымаўляецца сі шарп) - аб'ектна-арыентаваная мова праграмавання, распрацаваны кампаніі Microsoft пад кіраўніцтвам Андэрса Хейлсберга і Скота Вільтаумота як асноўная мова распрацоўкі прыкладанняў для платформы Microsoft .NET Framework.



Кампілятар C# кампілюе праграму, напісаную на C#, у прамежкавы код - Common Intermedia Language (IL) або Intermedia Language (IL) - які ўяўляе сабой высокаўзроўневы аб'ектна-арыентаваны асэмблер.

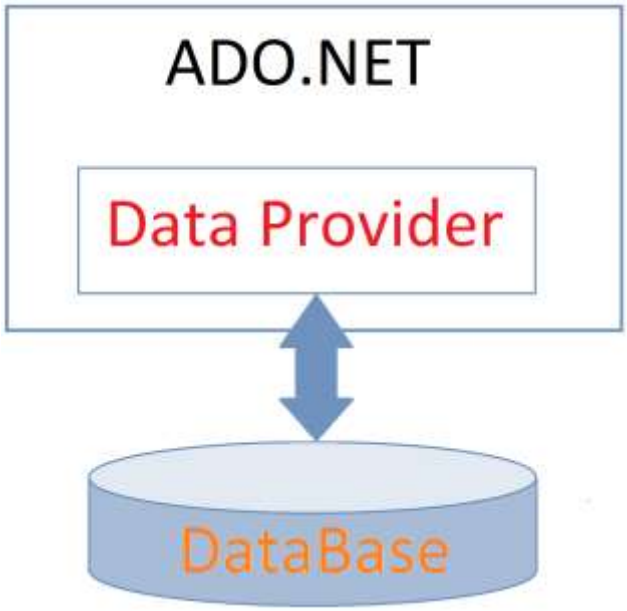
Гэты код інтэрпрэтуецца JIT - кампілятарам (Just-in-Time Compilator) у інструкцыі CPU.

Падобны падыход дазваляе распрацоўваць розныя часткі праекту на розных мовах.

Больш падрабязна аб C# гл.: http://asp.net.by/cs/

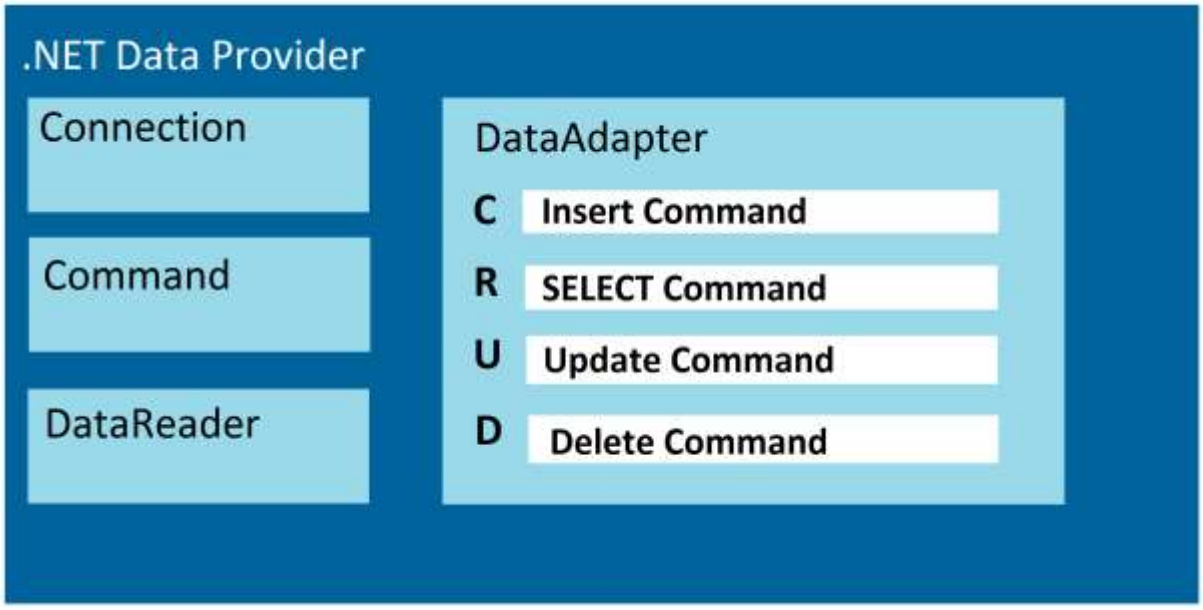
ADO.NET

ADO.NET - набор класаў, якія ўяўляюць сабой базавую функцыянальнасць для працы з базамі дадзеных і іншымі крыніцамі, якія захоўваюць інфармацыю (Microsoft SQL Server, Microsoft Access, Microsoft Excel, Microsoft Outlook, Microsoft Exchange, Oracle, OLE DB, ODBC, XML, тэкставыя файлы).



Маецца магчымасць аўтаномнай працы з дапамогай аб'ектаў DataSet, якія ўяўляюць сабой копіі табліц са сувязямі.

Аб'ект DataSet дазваляе працаваць з базай дадзеных у адключаным стане, і адпраўляць зваротна атрыманыя дадзеныя з дапамогай адаптара дадзеных.



Доступ да ADO.NET магчымы з C#, VB.NET і з любой мовы dot.NET.

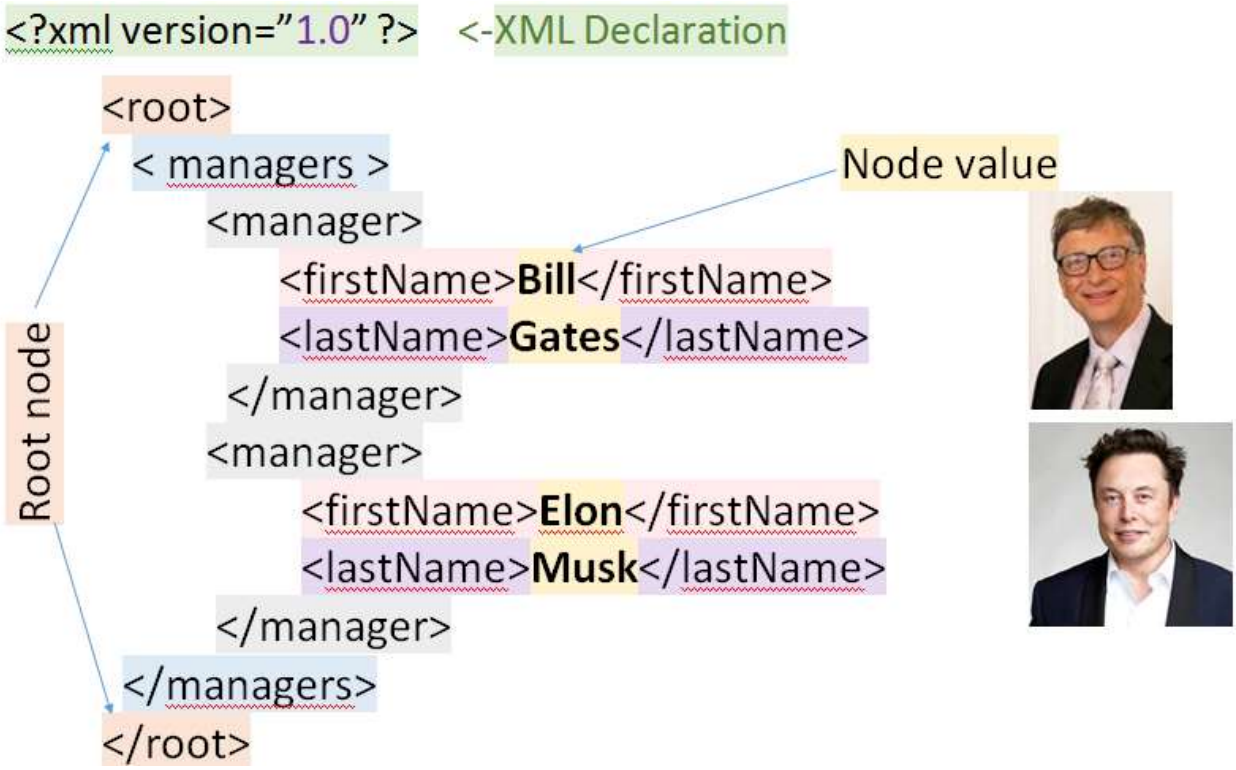
Класы ADO.NET змяшчаюцца ў файле System.Data.dll

Больш падрабязна аб ADO.NET гл.: http://asp.net.by/ADO.NET/

XML

XML (eXtensible Markup Language, якая пашыраецца мова разметкі) - гэта спосаб апісання структураваных дадзеных. Структураванымі дадзенымі завуцца такія дадзеныя, якія валодаюць зададзеным наборам семантычных атрыбутаў і дапушчаюць іерархічнае апісанне. XML-дадзеныя змяшчаюцца ў дакуменце, у ролі якога можа выступаць файл, струмень ці іншае сховішча інфармацыі, здольнае падтрымліваць тэкставы фармат.

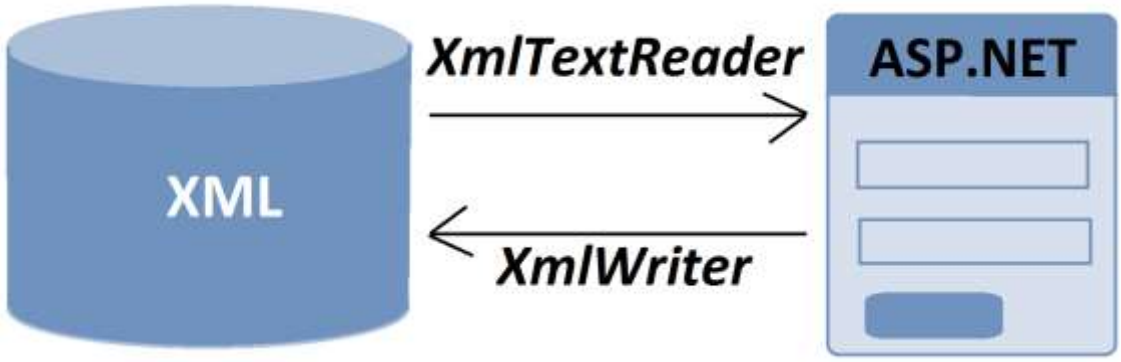
Любы XML-дакумент будуецца па пэўных правілах.



Прастора імёнаў System.Xml забяспечвае базавую функцыянальнасць працы з XML-файламі ў ASP.NET.

Клас XmlWriter - гэта базавы клас, які забяспечвае працэс перасылання толькі для чытання для генерацыі патоку XML. Ён дае метады для запісу дадзеных у дакумент XML.

XmlTextReader - гэта клас для чытання дадзеных з XML-дакумента.



Канфігурацыйныя файлы WEB.config выкарыстоўваюць XML.

Карты сайта для Google таксама пішуцца ў XML.

Прыклад фрагмента карты сайта SiteMap.XML:

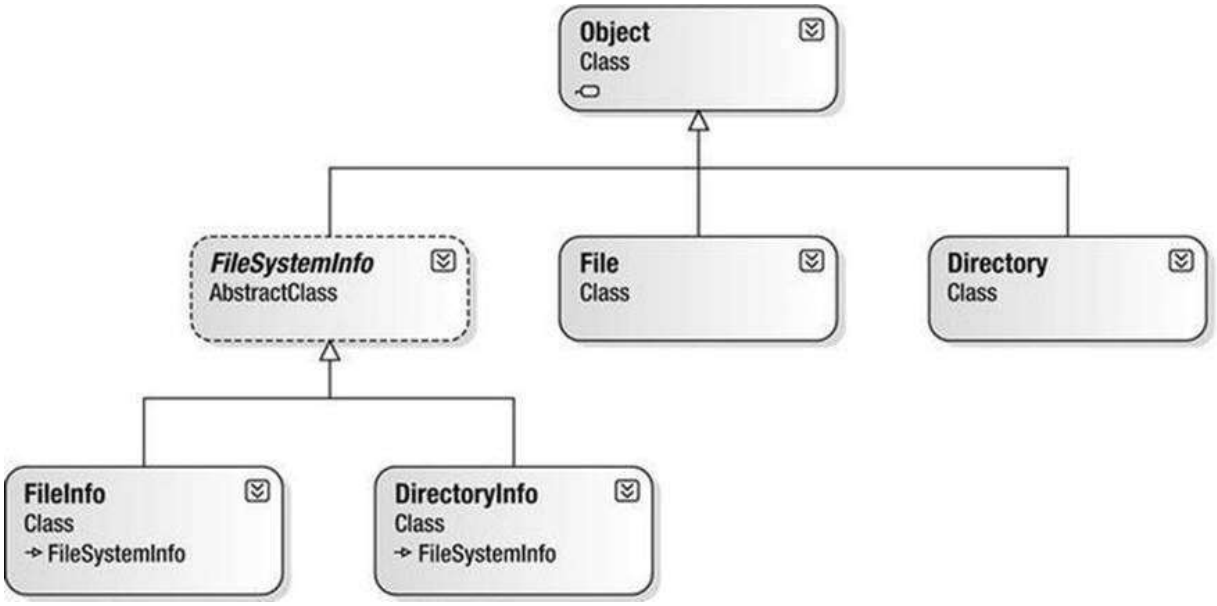
```
<?xml version="1.0" encoding="UTF-8"?>
<urlset xmlns="http://www.sitemaps.org/schemas/sitemap/0.9">
<url>
<loc>http://sunny.gear.host/web-technologies/</loc>
<priority>0.5</priority>
<lastmod>2019-11-10T20:20:33+00:00</lastmod>
<changefreq>daily</changefreq>
</url>
……
</urlset>
```

Больш падрабязна аб працы з XML у ASP.NET гл.: http://asp.net.by/XML/

Праца з файламі ў ASP.NET

Для выканання аперацый з файламі існуе некалькі класаў, размешчаных у прасторы імёнаў "System.IO".

2 класа File і FileInfo, якія прызначаны для працы з файлам, як з часткай файлавай сістэмы. Ёсць некалькі метадаў, якія дазваляюць працаваць са змесцівам файла цалкам.



Класы **Directory**, **File**, **DirectoryInfo** і **FileInfo** прызначаны для працы з каталогамі і файламі.

Першыя два класы выконваюць аперацыі пры дапамозе статычных метадаў, другія два - пры дапамозе экзэмплярных метадаў.

Прыклад працы з файламі ў ASP.NET.

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>";
    }
}
```

Класы **Directory**, **File**, **DirectoryInfo** і **FileInfo** прызначаны для працы з каталогамі і файламі. Першыя два класы выконваюць аперацыі пры дапамозе статычных метадаў, другія два - пры дапамозе экзэмплярных метадаў.

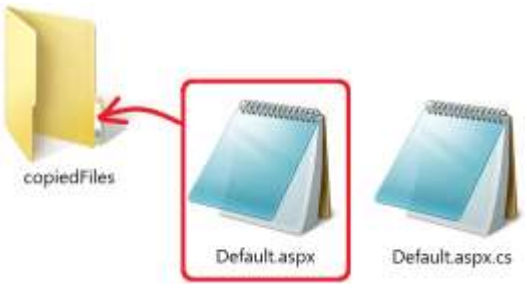
Ніжэй прыведзены код правярае, ці існуе каталог

```
if (Directory.Exists(folderLocation))
```

Калі гэта так, то мы правяраем, ці існуе файл, які мы збіраемся капіяваць

```
if (File.Exists(fileToCopy))
```

Калі ўсё ў парадку, тады мы ідзем далей, і файл капіюецца куды варта




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

Калі файл, які капіюецца, ужо знаходзіцца ў дырэкторыі, усталёўка значэння перавызначэння на **true** заменіць яго. Значэнне па змаўчанні (default) - **false**, і вы атрымаеце памылку (error), калі які такі файл ужо існуе.

Каб перамясціць файл у новае месца, вы павінны выкарыстоўваць метад Move класа File:

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

Больш падрабязна аб працы з файламі гл.: http://asp.net.by/Files/

Канфігурацыя і бяспека

Аўтэнтыфікацыя

Аўтэнтыфікацыяй (authentication) называецца працэс ідэнтыфікацыі user’аў.

Аўтарызацыяй (authorization) называецца працэс прадастаўлення доступу user'ам на аснове іх ідэнтыфікацыйных дадзеных.

Аўтарызацыя

ASP.NET сумесна з вэб-серверам забяспечвае некалькі магчымых тыпаў аўтэнтыфікацыі:

- Windows (па змаўчанні - default),
- Forms
- Passport
- None

Выбар тыпу вызначае механізм захоўвання маркераў аўтэнтыфікаванага user’а. У выпадку тыпу Windows маркер змяшчаецца ў кантэкст патоку працоўнага працэсу ASP.NET. Калі выкарыстоўваецца тып Forms, маркер перадаецца ад кліента да сервера і зваротна ў cookie-файле..

Задаць патрабаванні аўтэнтыфікацыі кліентаў вэб-прыкладанні можна шляхам дадання адпаведных элементаў у канфігурацыйны файл прыкладання **web.config**.

Тып аўтэнтыфікацыі кліентаў задаецца з дапамогай элемента <authentication>, атрыбут mode якога можа прымаць адно з чатырох значэнняў: Windows, Forms, Passport і None. Сканфігураваўшы тып аўтэнтыфікацыі, трэба прадумаць мэты аўтэнтыфікацыі. Аўтэнтыфікацыя выконваецца для абмежавання доступу кліентаў да ўсяго з дадаткам або да яго частках з дапамогай элемента <authorization>.

У гэты элемент дадаюцца падэлементы <allow> і <deny>, якія задаюць прадастаўленне або адмову ў доступе асобным user'ам і ролям.

Метасімвал * выкарыстоўваецца для прадстаўлення ўсіх user'аў, а ? – для прадстаўлення ананімных user'аў.

Наступны прыклад канфігурацыйнага файла адмаўляе ананімным user'ам у праве доступу да сайта:

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

Элементы <allow> і <deny> падтрымліваюць тры атрыбуты: users, roles і verbs. Значэннямі гэтых атрыбутаў могуць быць падзеленыя коскамі спісы user'аў, роляў і каманд.

У корані WEB-config павінен выглядаць так:

```
<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>
```

Файл **LoginPage.aspx** павінен выглядаць так:

```
<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: Andy

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

<= Invalid login

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

<= Invalid password

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

Password:

```
<script language="C#" runat="server">

 void OnLogIn (Object sender, EventArgs e)

 {   if (FormsAuthentication.Authenticate(UserName.Text, Password.Text))

        FormsAuthentication.RedirectFromLoginPage(UserName.Text,  false);

     else

        {  l_Login.Text = "<= Invalid login";

           l_or.Text="or";

           l_Password.Text="<= Invalid password";

        }
 }

</script>
```

Больш падрабязна аб канфігурацыі і бяспецы гл: http://asp.net.by/Config/

Ярошевіч Андрэй Алегавіч

Беларускі Дзяржаўнага Універсітэта

e-mail: ao@asp.net.by

yaroshevich.com asp.net.by