



Информатика

UI, event-driven, клиенты

Ранее...

- ✓ Client, server
- ✓ Handler
- ✓ Request, response
- ✓ HTTP
- ✓ Cookies, session

План лекции

- Эволюция пользовательского интерфейса
- GUI
- События в GUI
- Веб-страницы – GUI
- ЯП для фронта
- AJAX

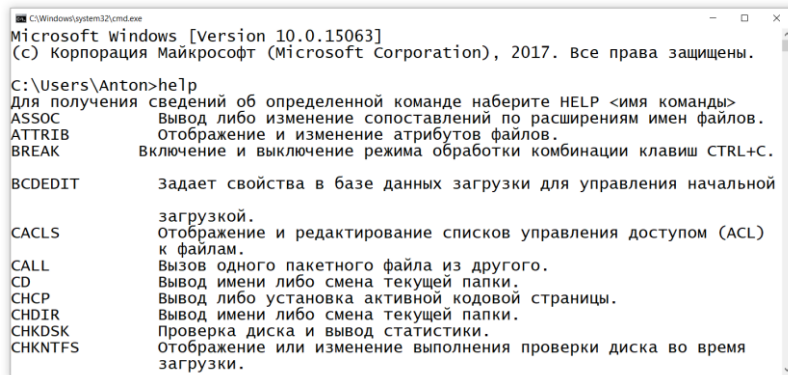
Пользовательский интерфейс

- **User Interface (UI)**

*обеспечивает взаимодействие
пользователя-человека с программно-
аппаратными компонентами
компьютерной системы*

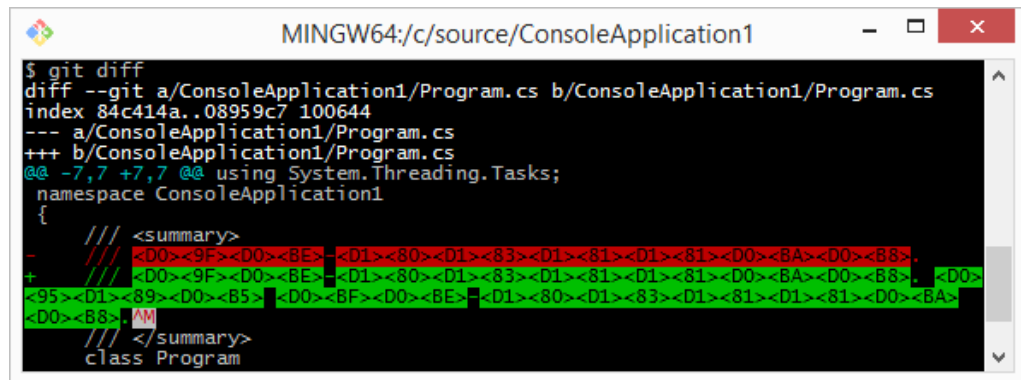
Командная строка

- **Command prompt**
- Консоль, терминал (Google!)



```
Microsoft Windows [Version 10.0.15063]
(C) Корпорация Майкрософт (Microsoft Corporation), 2017. Все права защищены.

C:\Users\Anton>help
Для получения сведений об определенной команде наберите HELP <имя команды>
ASSOC      Вывод либо изменение сопоставлений по расширениям имен файлов.
ATTRIB     Отображение и изменение атрибутов файлов.
BREAK      Включение и выключение режима обработки комбинации клавиш CTRL+C.
BCDEDIT     Задаёт свойства в базе данных загрузки для управления начальной
            загрузкой.
CACLS      Отображение и редактирование списков управления доступом (ACL)
            к файлам.
CALL       Вызов одного пакетного файла из другого.
CD         Вывод имени либо смена текущей папки.
CHCP       Вывод либо установка активной кодовой страницы.
CHDIR      Вывод имени либо смена текущей папки.
CHKDSK     Проверка диска и вывод статистики.
CHKNTFS    Отображение или изменение выполнения проверки диска во время
            загрузки.
```

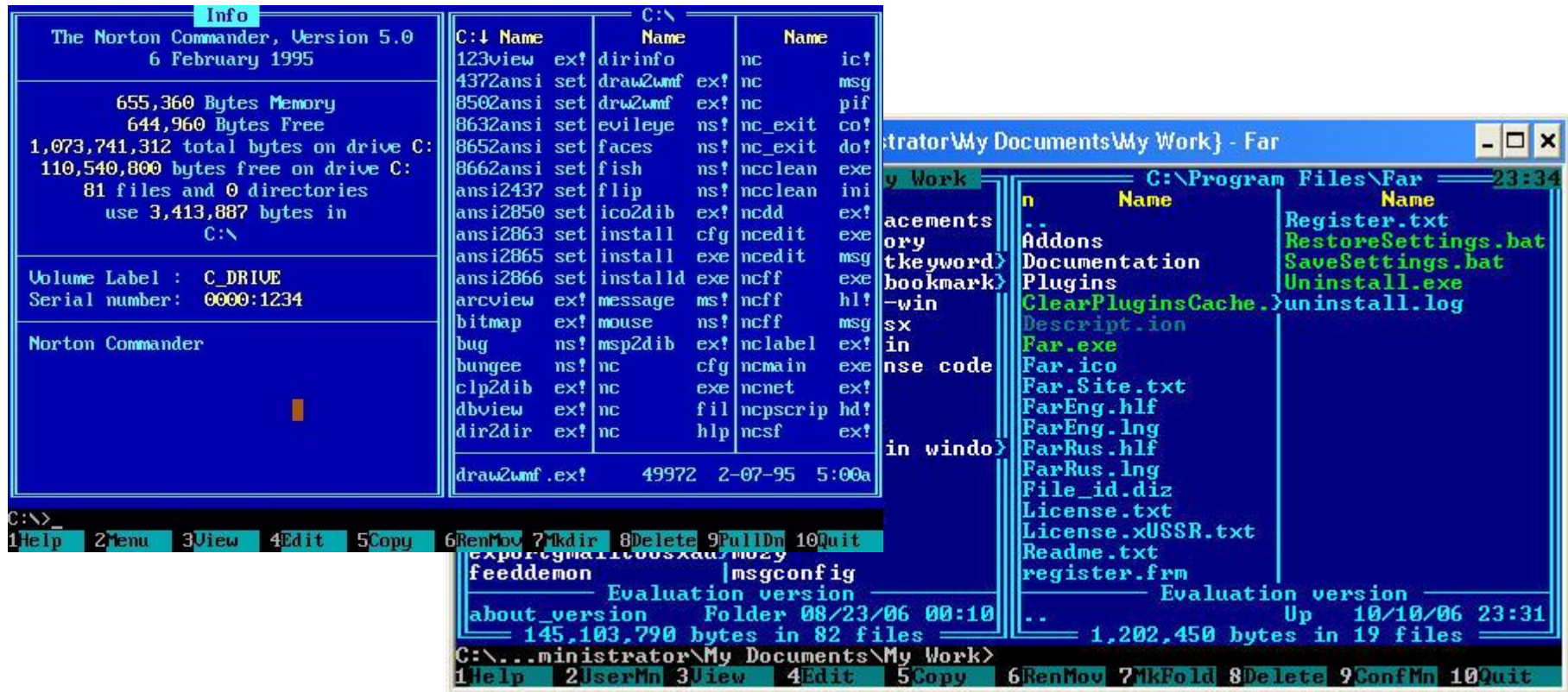


```
MINGW64:/c/source/ConsoleApplication1
$ git diff
diff --git a/ConsoleApplication1/Program.cs b/ConsoleApplication1/Program.cs
index 84c414a..08959c7 100644
--- a/ConsoleApplication1/Program.cs
+++ b/ConsoleApplication1/Program.cs
@@ -7,7 +7,7 @@ using System.Threading.Tasks;
 namespace ConsoleApplication1
 {
     /// <summary>
-    /// <D0><9F><D0><BE> <D1><80><D1><83><D1><81><D1><81><D0><BA><D0><B8>.
+    /// <D0><9F><D0><BE> <D1><80><D1><83><D1><81><D1><81><D0><BA><D0><B8>. <D0>
-95><D1><89><D0><B5> <D0><BF><D0><BE> <D1><80><D1><83><D1><81><D1><81><D0><BA>
<D0><B8>.AM
     /// </summary>
     class Program
```

Консольные приложения с графическими элементами

- Графические меню:
 - BIOS, GRUB,
- Консольные файловые менеджеры:
 - Norton Commander, FAR Manager,
- Консольные игры:

Примеры



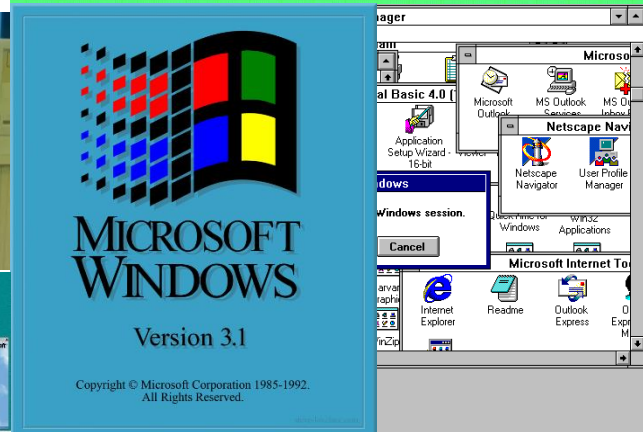
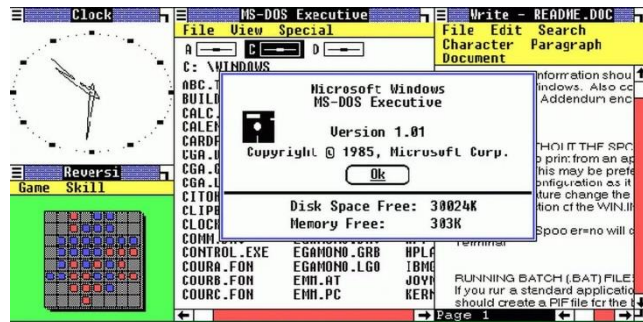
Graphical User Interface (GUI)

- 1960-ые, Дуглас Энгельбарт
- 1970-ые, Xerox PARC WIMP
- 1984, Apple Macintosh
- 1985, ...



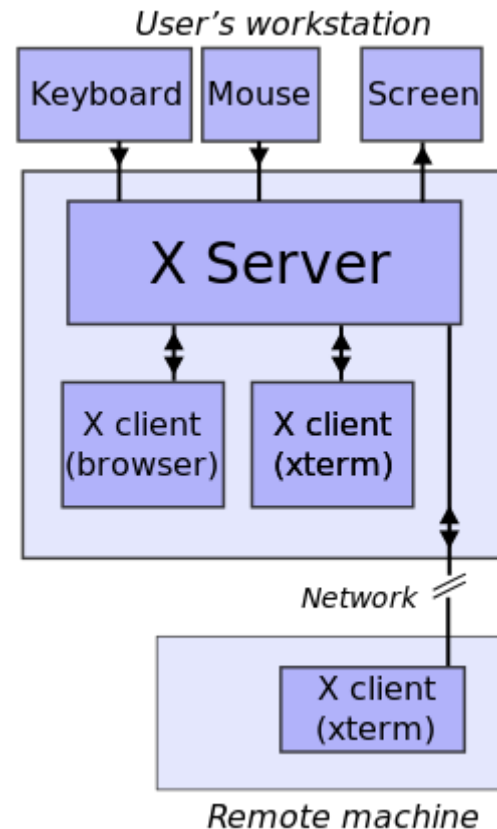
Windows

- 1985, Windows 1.0
 - надстройка над MS-DOS
- 1992, Windows 3.1
 - OLE, WYSIWYG
- 1995, Windows 95



Unix (Linux, Free BSD)

- «Иксы» (X Window System)
 - Клиент-серверная модель
 - Ядро не связано с графической оболочкой
- Много различных оболочек
 - Gnome, KDE, Unity, ...



GUI. Users viewpoint

- Приложение – набор окон
- В каждом окне – стандартные графические элементы
 - кнопки, меню, списки, поля,...
- Клавиатурой и мышью управляем приложением (кликаем, перетаскиваем, печатаем...)

GUI. Developers viewpoint

- Приложение – набор окон (window, frame, form)
- Окно – набор графических элементов (компонентов, виджетов), имеющих определенное расположение (layout)
- Каждый графический элемент имеет свойства (координаты, текст, ширина, ...)

GUI. Developers viewpoint

Графический элемент – объект

А где объект и свойства, там и методы

- С каждым графическим элементом связан набор *событий* (events), которые пользователь может *инициировать* с помощью устройств ввода

GUI. Developers viewpoint

- С каждым событием связан обработчик (handler) – действие (метод), который выполняется при возникновении события
- Действие меняет данные приложения и графических элементов
- **GUI app dev = layout + handlers + logic**

Вопросы

- Что из этого пишем сами, а что используем?
- Как зависит приложение от среды исполнения?

Библиотеки разработки приложений с GUI

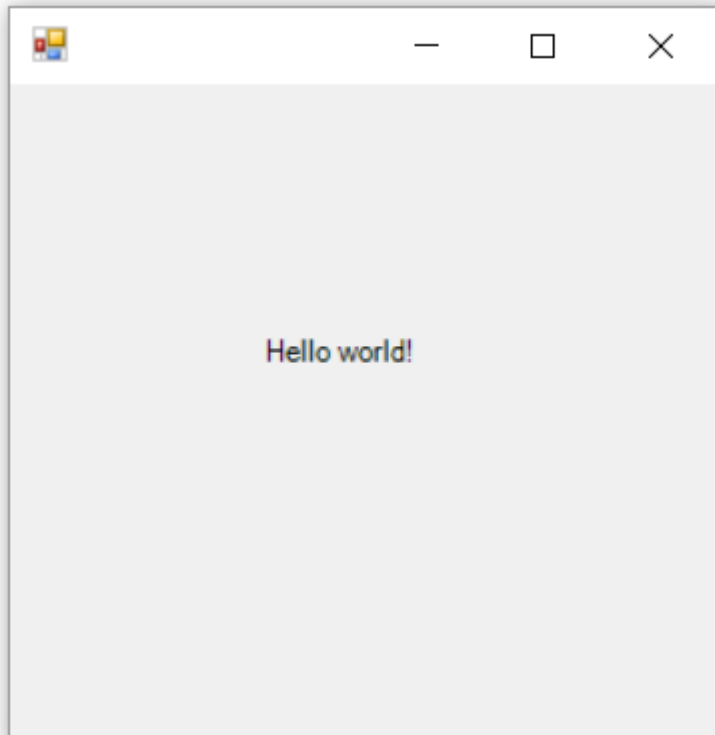
- **Низкоуровневые:**
 - WinAPI, Xlib, Carbon (до 2012)
- **Высокоуровневые:**
 - QT, GTK+, WinForms, WPF, Swing, Cocoa, ...

Пример с Windows Forms

- **System.Windows.Forms**
- Form – окно
- На нём размещаем элементы
- Label, Button, TextBox и др.

Hello world

```
using System.Windows.Forms;
public class Program
{
    public static void Main()
    {
        var form = new Form();
        form.Controls.Add(new Label()
        {
            Text = "Hello world!",
            Left = 100,
            Top = 100
        });
        Application.Run(form);
    }
}
```

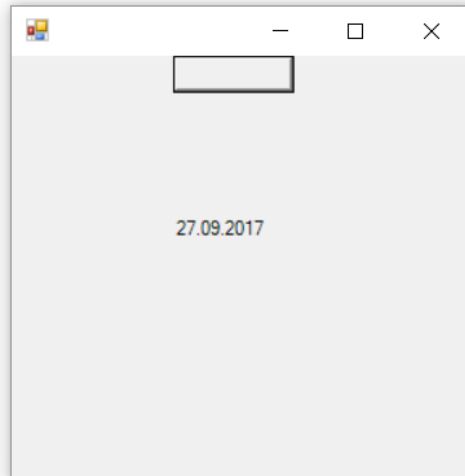


Обработчик

- К событию привязывается метод
- В С# - подписка на мультикаст делегат
 - Google! «отличия события от делегата»

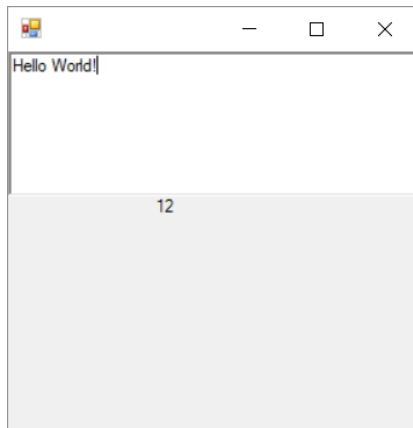
Подстановка текущей даты

```
using System;
using System.Windows.Forms;
public class Program
{
    public static void Main()
    {
        var form = new Form();
        form.Controls.Add(new Label
            { Name = "label1", Left = 100, Top = 100 });
        var button = new Button { Left = 100 };
        button.Click += (sender, e) =>
            { ((Label)form.Controls.Find("label1", false)[0]).Text =
                DateTime.Now.ToShortDateString(); };
        form.Controls.Add(button);
        Application.Run(form);
    }
}
```



Подсчет количества символов

```
using System.Windows.Forms;
public class Program
{
    public static void Main()
    {
        var form = new Form();
        form.Controls.Add(new Label{Name="label",Left=100,Top=100});
        var textBox=new TextBox
            { Height=100, Multiline = true,
              Name = "textBox", Width = form.Width };
        textBox.TextChanged += (sender, e) =>
            { ((Label)form.Controls["label"]).Text =
              textBox.Text.Length.ToString(); };
        form.Controls.Add(textBox);
        Application.Run(form);
    }
}
```



Эти примеры – ненормальны!

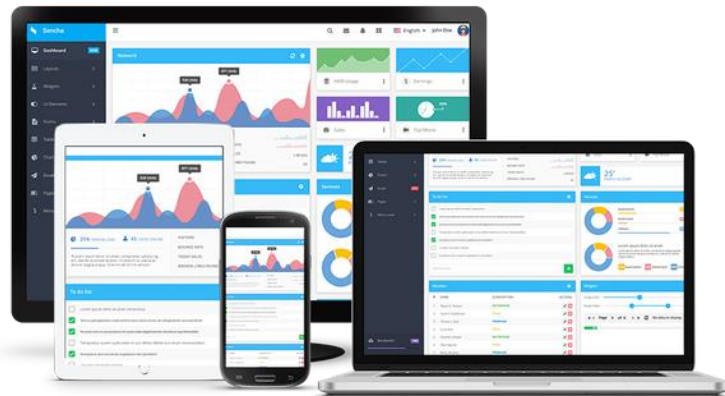
- Рассмотрим Windows Forms и WPF подробнее в дальнейших лекциях
- Лепить всё программно – не очень
- Писать логику в «онкликах» – тоже
- И вообще, нужно MVP и MVVM!

Вернёмся в браузер

- Браузер – тоже оконное графическое приложение
 - есть меню, поле ввода (адресная строка, поиск) и др.
- **С помощью браузера работаем с web-приложениями!**
 - По аналогии ОС и десктоп приложений

Преимущества web-приложений

- **Не нужна установка**
 - только браузер и интернет
- **Легко поддерживать**
 - Обновил на сервере и готово
- **Кросс-платформенность**
 - Нужна кроссбраузерность, но её обеспечить проще
- **Еще?**



Web-приложение

- GUI – HTML-страница
- Поддерживает стандартные элементы GUI
- Поддерживает события
 - С помощью форм
- Но формы только для доставки данных на сервер
 - Обработывают события для общения с сервером

Есть и другие события

- Не инициирующие запрос к серверу
 - Работа модальных окон, подсказок
 - Слайдер/карусель
 - Наведение курсора
 - Заккрытие окошка чата ВК
 - ...
- Значит, есть код, исполняемый в браузере, обеспечивающий такую динамику.
And his name is...



???

~~Javascript~~ (не торопитесь) ECMAScript

- 1995, Брендан Эйх
+ Марк Андерсен (Netscape)
+ Билл Джой (Sun Microsystems)
Mocha -> LiveScript -> JavaScript
- Ядро браузерных языков
 - JavaScript, ActionScript, Jscript - реализации
- **Javascript \approx ECMAScript + DOM + BOM**



DOM и BOM

- **DOM – Document Object Model**

- Позволяет получать доступ к структуре HTML и менять его содержимое
- 1998 DOM 1, 2000 DOM 2, 2004 DOM 3
- HTML-элементы – объекты со свойствами. DOM даёт JS возможность изменять эти свойства

- **BOM – Browser Object Model**

- Надмножество DOM
- Плохо стандартизировано

Встраиваем в HTML

```
<html>
<head>
</head>
<body>
    <script type="application/javascript">
        ... Вот тут JS в чистом виде ...
    </script>
</body>
</html>
```

Можно выделять в отдельные файлы (*.js)

Встраиваем в HTML

```
<script type="text/javascript" >
```

```
    for (var i = 0; i < 10; i++) {  
        //печать в страницу  
        document.writeln(i);  
        //печать в консоль  
        console.log(i);  
    }
```

```
</script>
```

Синтаксис JS

- Родственный C, Java (кто создатели?)
- Есть:
 - Динамическая, утиная типизация
 - Объектно-ориентированный подход
 - Функции-объекты первого рода
 - Анонимные функции
 - ...

Утиная типизация

- Тест на утку

If it walks like a duck and quacks like a duck, it must be a duck
(Если что-то ходит, как утка, и крякает, как утка, то это утка)

- Объект реализует интерфейс, если реализует все его методы, независимо от связей в иерархии

- Как с ней в C#?

Объектно-ориентированный

- Вот именно объектно, а не класс-ориентированный
- В Javascript нет классов
- Но один объект может быть прототипом другого:

```
var human = { "name": "John" };  
var student = { "university": "KFU" };  
student.__proto__ = human;  
alert(student.university);  
alert(student.name);
```

Конструктор

- Создавать прототипы можно через конструктор

```
var human = { "name": "John" };  
function Student(university) {  
    this.university = university;  
    //Не работает MS IE10-  
    this.__proto__ = human;  
}  
//Для MS IE10-  
Student.prototype = human;  
var student = new Student("university");  
alert(student.university);  
alert(student.name);
```

Функции – объекты

```
var f = function () {  
    alert("I am F!");  
}  
var g = f; //вызова функции еще не было  
g(); //ВОТ ВЫЗОВ
```

Благодаря этому связка «событие-обработчик» в JS реализовано *естественным* образом

Пример с датой на JS

```
<script type="text/javascript" >  
showDate = function () {  
var textField =  
document.getElementById("dateText");  
textField.value = new Date();  
}  
</script>
```

```
<p><input type="text" id="dateText" /></p>  
<button onclick="showDate()">Show the Date</button>
```

Show the Date

Wed Sep 27 2017 22:18

Show the Date

Подсчёт количества символов

count.js

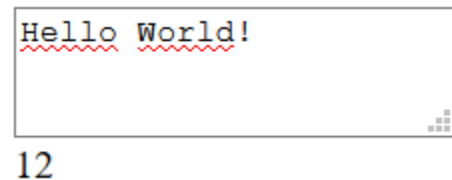
```
function countSymbols() {  
    var textArea = document.getElementById("textA");  
    document.getElementById('counter').innerText =  
        textArea.value.length;  
}
```



0

count.html

```
<script type="application/javascript" src="count.js"></script>  
<textarea id="textA"></textarea>  
<div id="counter">0</div>  
<script type="application/javascript">  
    var obj = document.getElementById('textA');  
    obj.oninput = countSymbols;  
</script>
```



12

Подсчёт количества символов

count.html

```
<textarea id="textA"></textarea> <div id="counter">0</div>
<script type="application/javascript">
    var obj = document.getElementById('textA');
    obj.oninput = function() {
        var textArea = document.getElementById("textA");
        document.getElementById('counter').innerText =
            textArea.value.length;
    };
</script>
```

Итак

- Javascript позволяет изменять содержимое HTML
- А что, если при этом данные нужно брать с сервера?
 - Читаем новости VK, добавилась еще одна
 - Переписываемся, пришло сообщение

AJAX

- Asynchronous Javascript and XML
- Фоновый (без перезагрузки страницы) обмен информацией с сервером + динамическое изменение содержимого страницы
- Преимущества такого подхода?

Преимущества AJAX

- Экономия трафика
- Интерфейс остается отзывчивым во время выполнения длительной операции
- Возможность реализации динамического графического интерфейса
- !

Примеры AJAX

1. Подгрузка
2. Проверка
3. Текстовые трансляции
4. Чаты

```
<script type="text/javascript">
    function loadXMLDoc() {
        var xmlhttp; // Объект для совершения запроса
        if (window.XMLHttpRequest) {
            //for IE7+, Firefox, Chrome, Opera, Safari
            xmlhttp = new XMLHttpRequest();
        }
        else {
            // for IE6, IE5
            xmlhttp = new ActiveXObject("Microsoft.XMLHTTP");
        }
        // Отложенный вызов (callback) функции, когда меняется статус запроса
        xmlhttp.onreadystatechange = function () {
            /* readystate - статус запроса 0 - Uninitialized, 1 - Loading,
            2 - Loaded, 3 - Interactive, 4 - Complete */
            if (xmlhttp.readyState == 4 && xmlhttp.status == 200) {
                document.getElementById("myDiv").innerHTML = xmlhttp.responseText;
            }
        }
        xmlhttp.open("GET", "http://localhost:8080/ajaxtest", true); // открывается соединение
        xmlhttp.send(""); // посылает запрос
    }
</script>

<button type="button" onclick="loadXMLDoc()">
    Get secret info from Server
</button> <p>Secret info is:<div id="myDiv"></div></p>
```

Ajaxtest HTTPListener

```
using System.Net;
public class Program
{
    public static void Main()
    {
        var listener = new HttpListener();
        listener.Prefixes.Add("http://localhost:8080/ajaxtest/");
        listener.Start();

        var context = listener.GetContext();

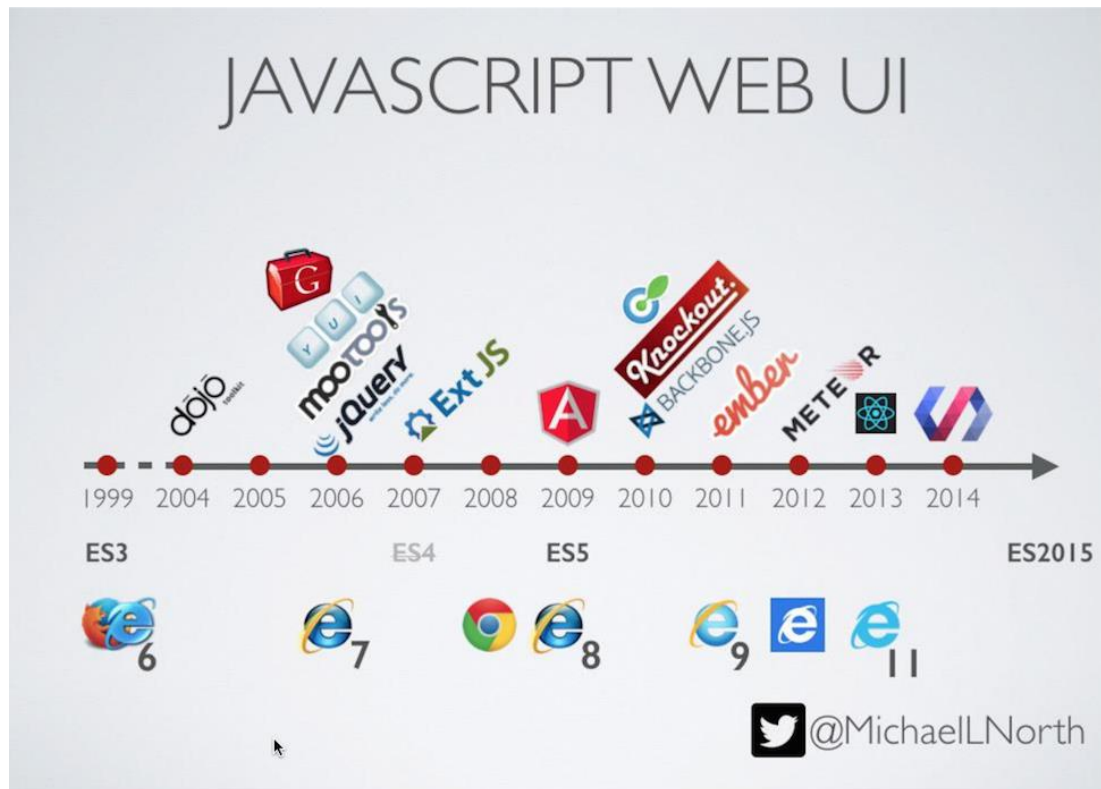
        context.Response.AppendHeader("Access-Control-Allow-Origin", "*");

        context.Response.ContentType = "text/xml";
        var secret = @"<date>42</date>";
        var bytes = System.Text.Encoding.UTF8.GetBytes(secret);
        context.Response.StatusCode = (int)HttpStatusCode.OK;
        context.Response.ContentLength64 = bytes.Length;
        context.Response.OutputStream.Write(bytes, 0, bytes.Length);

        listener.Stop();
    }
}
```

JS. Библиотеки/фреймворки

- jQuery
- AngularJS
- React
- Vue.js
- Node.js
- ...



Почитать

- [Терминология GUI-проектирования](#)
- <http://javascript.ru/ajax>
- <https://learn.javascript.ru>
- [Выразительный Javascript](#)



Вопросы?

e-mail: marchenko@it.kfu.ru