Тема 11. **Введение в JavaScript**

СОДЕРЖАНИЕ

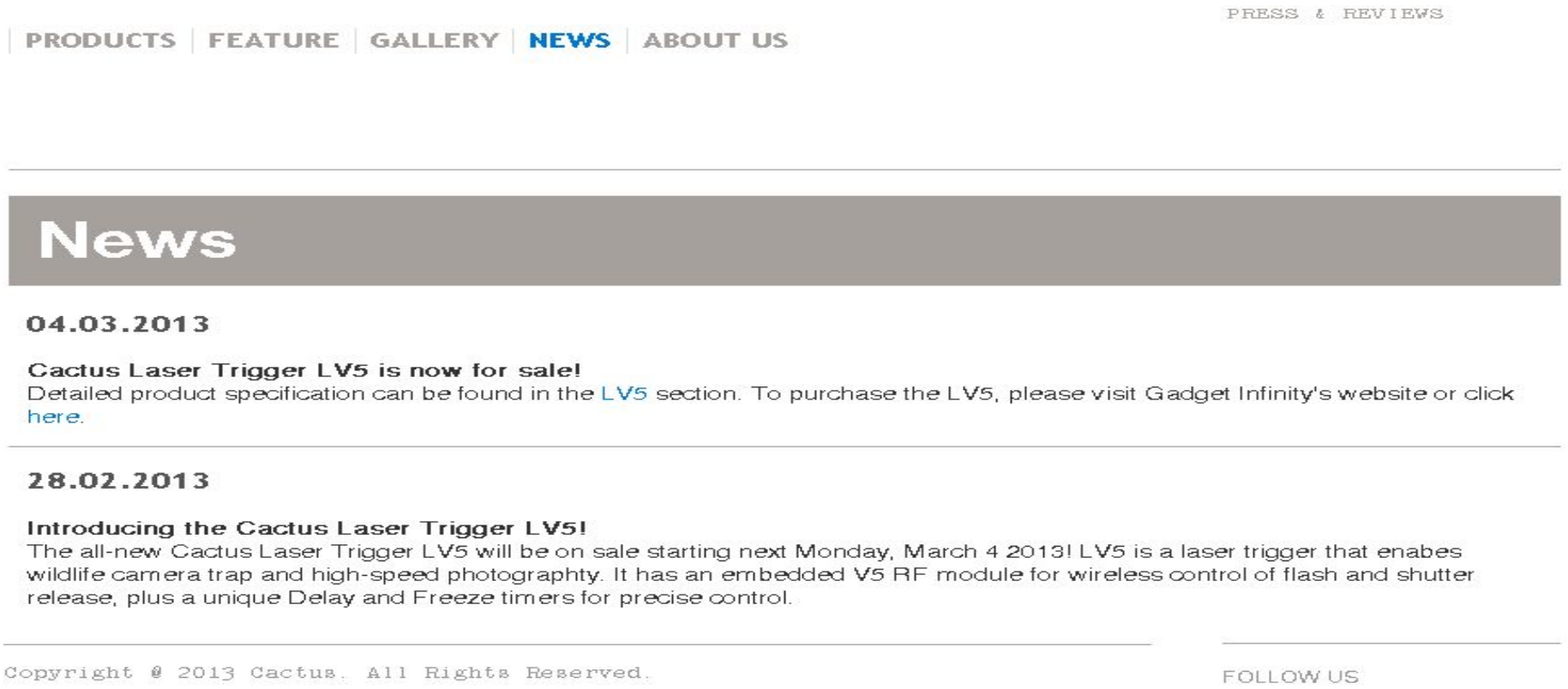
1. Введение в Web-программирование
2. Введение в JavaScript
3. Общее описание JS
4. Синтаксис
5. Примеры

ПОНЯТИЕ „WEB-ПРОГРАММИРОВАНИЕ”

- В общем предполагает **программирование сайтов** для **Интернета** (глобальная сеть) или **Интранета** (локальная сеть)
- Некоторые специалисты часто используют понятие **«веб-разработка»** (“web development”), которая ссылается на:
  - кодирование и маркировке,
  - анализ области для которой разрабатывается сайт
  - проектирование решения
  - менеджмент (планирование и управление) содержимого сайта
  - написание клиентских скриптов и скриптов на стороне сервера
  - настройка безопасности сети и веб-серверов
  - развитие электронного бизнеса

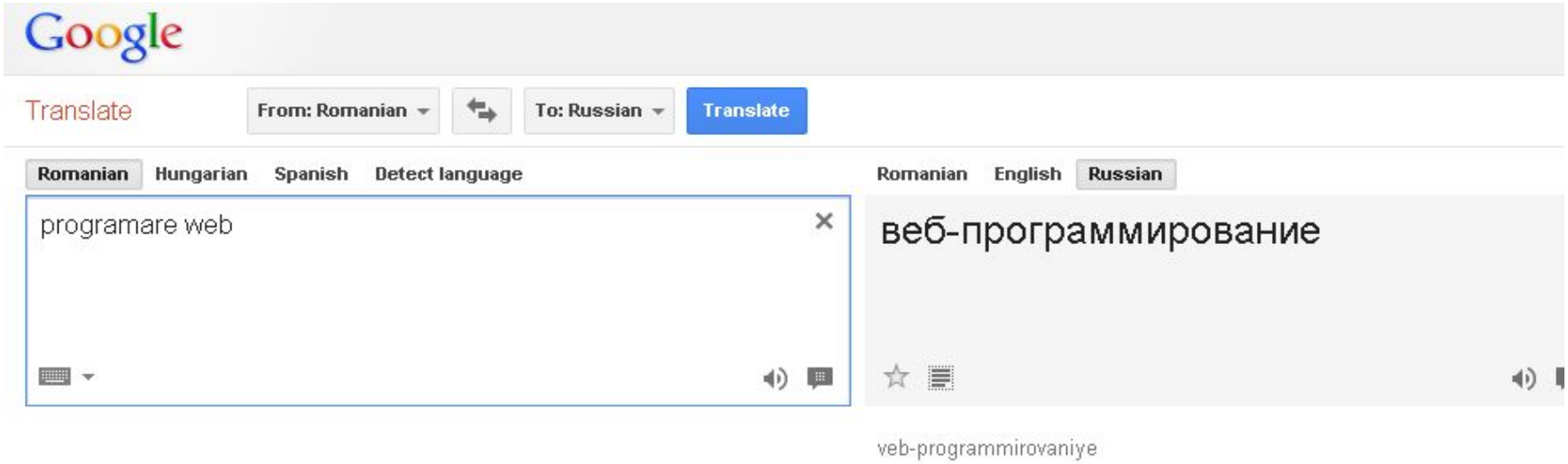
ОБЛАСТИ ПРИМЕНЕНИЯ ВЕБ-ПРОГРАММИРОВАНИЯ

Веб-программирование относится к разработке **статических сайтов** (HTML+CSS), которые, обычно имеют расширение .html



ОБЛАСТИ ПРИМЕНЕНИЯ ВЕБ-ПРОГРАММИРОВАНИЯ

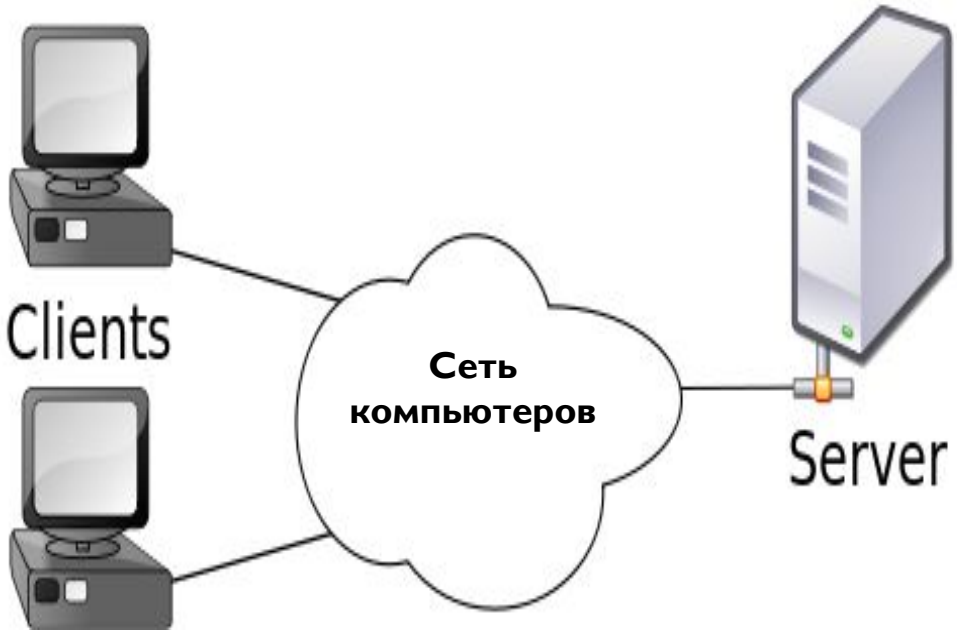
- Веб-программирование относится и к разработке **динамических сайтов** – сложные web-приложения или системы, электронный бизнес, социальные сети (Facebook) etc.

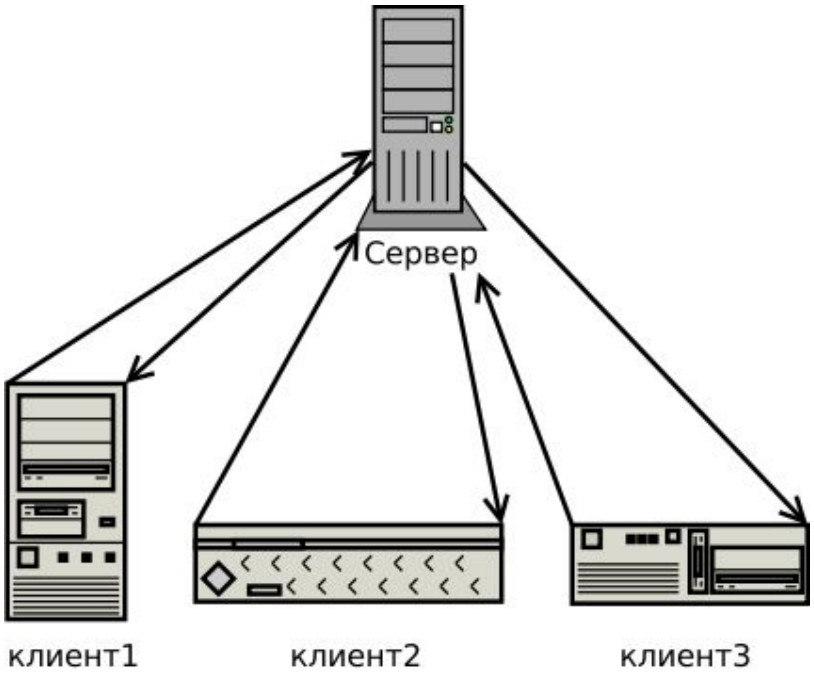


КЛИЕНТ-СЕРВЕРНАЯ МОДЕЛЬ

- **Клиент-сервер** это вычислительная или сетевая архитектура, в которой задания или сетевая нагрузка распределены между поставщиками услуг, называемыми **серверами**, и заказчиками услуг, называемыми **клиентами**
- Физически клиент и сервер — это программное обеспечение
- Обычно клиент и сервер взаимодействуют через компьютерную сеть посредством сетевых протоколов и находятся на разных вычислительных машинах, но могут выполняться также и на одной машине
- Программы — сервера, ожидают от клиентских программ запросы и предоставляют им свои ресурсы в виде данных (например, загрузка файлов посредством HTTP, FTP или работа с базами данных) или сервисных функций (например, работа с электронной почтой, просмотр web-страниц во всемирной паутине)
- Клиент не делится ни с кем, никакими ресурсами

КЛИЕНТ-СЕРВЕРНАЯ МОДЕЛЬ





Примеры:
- Интернет-банкинг
- Проверка эл. почты
- google-translate

WEB-СКРИПТЫ

- Web-скрипт это программные коды для компьютера созданные с целью динамизации Web-страницы
- Пример (источник: Microsoft):
  - Web-скрипт можно использовать для включения счетчика, считающий посетителей – его значение растет при каждом новым посещении сайта
  - Web-скрипт можно использовать для включения счетчика, считающий в убывающем порядке для какого-то специального события: „осталось только *x* дней”, где *x* уменьшается на 1 каждый день
  - Гостевые книги, доски объявлений, голосования и т.д

WEB-СКРИПТЫ

- Обычно web-скрипты выполняются веб-браузером, когда веб-страница открывается для отображения информации сгенерированные сценариями скрипта или выполняются на сервере
- Создание веб-скриптов требует знаний в области программирования
- Скрипты в веб-программирование могут быть:
  - Client-side (*Client-side scripts*) – на стороне клиента
  - Server-side (*Server-side scripts*) – на стороне сервера

„CLIENT-SIDE” ПРОГРАММИРОВАНИЕ

- Клиентские скрипты выполняются на стороне клиента, веб-браузером пользователя
- Клиентские скрипты часто включаются/ внедряются
  - в HTML документе (и называются "встроенными сценариями")
  - или в отдельном файле, к которому документ (или документы) его использующий, делает ссылку (и поэтому в данном случае называется «внешним сценарием
- Браузер пользователя выполняет сценарий, затем отображает документ
- Клиентские скрипты могут также содержать инструкции для пользовательского браузера, с целью проверки действий пользователя (например, нажатие кнопки)
- Часто, эти инструкции можно использовать без дальнейшего общения с сервером
- Клиентские скрипты не требуют дополнительного программного обеспечения на сервере
- Им необходимо чтобы веб-браузер пользователя, понимал язык сценариев, на которых они были написаны
- Scipting языки на стороне клиента: JavaScript, ActionScript, VBScript, и т.д. Ajax является важным дополнением к языку JavaScript

DHTML

- Программирование на стороне клиента является важной частью концепта Dynamic-HTML (DHTML)
- DHTML не является языком программирования
- В DHTML - используются несколько технологий вместе, для создания интерактивных и анимированных веб-сайтов:
  - Язык маркировки (как HTML),
  - Язык для client-side скриптинга (как например JavaScript),
  - Язык описывающий стили страниц сайта (как CSS),
  - DOM (Document Object Model) для HTML (http://www.w3schools.com/js/js_htmldom.asp)

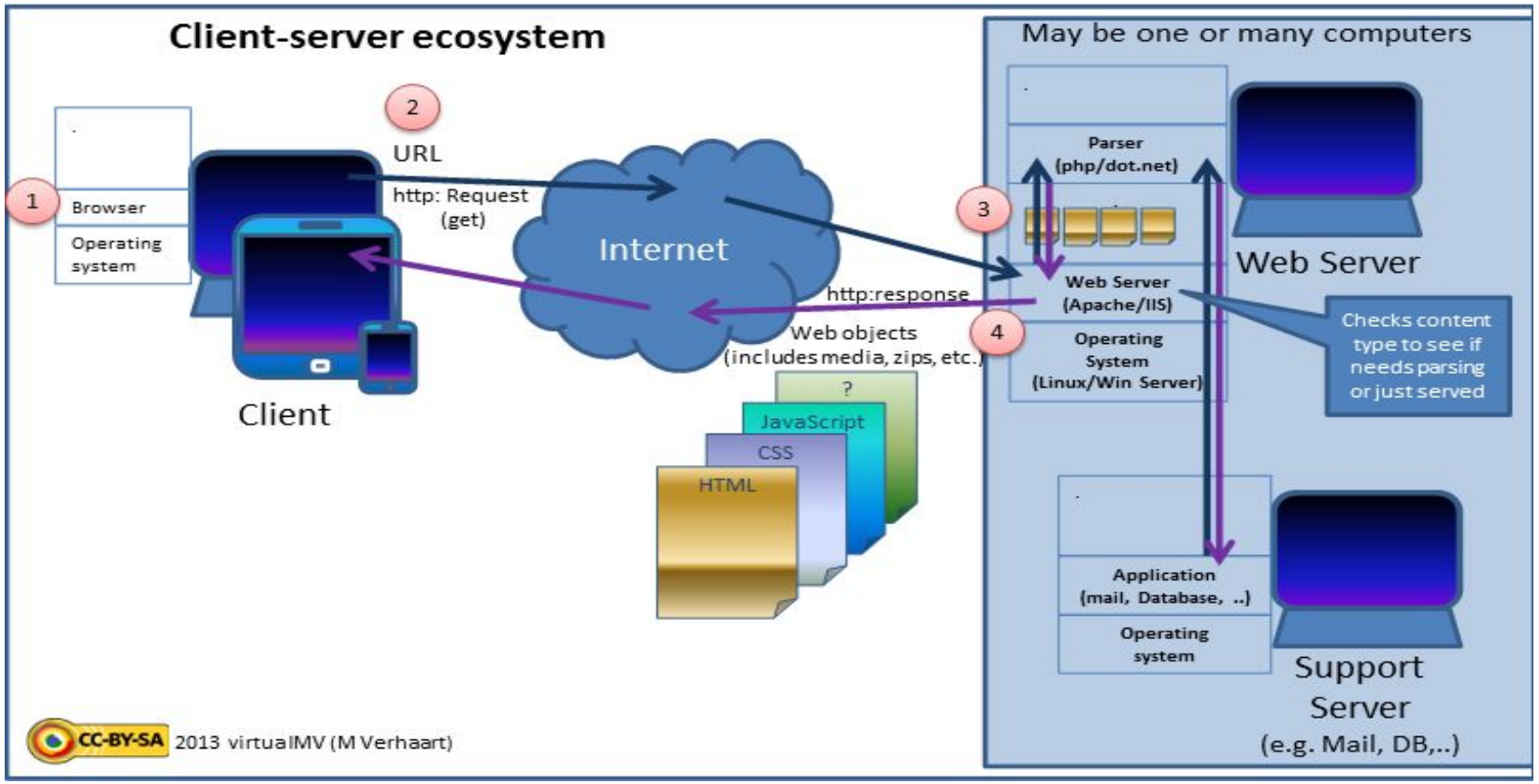
Dynamic HTML это возможность/метод представления страниц сайта

Контент страницы не меняется!!!

Примечание: не путайте понятие DHTML с понятием динамический сайт!

ДИНАМИЧЕСКИЕ WEB-СТРАНИЦЫ

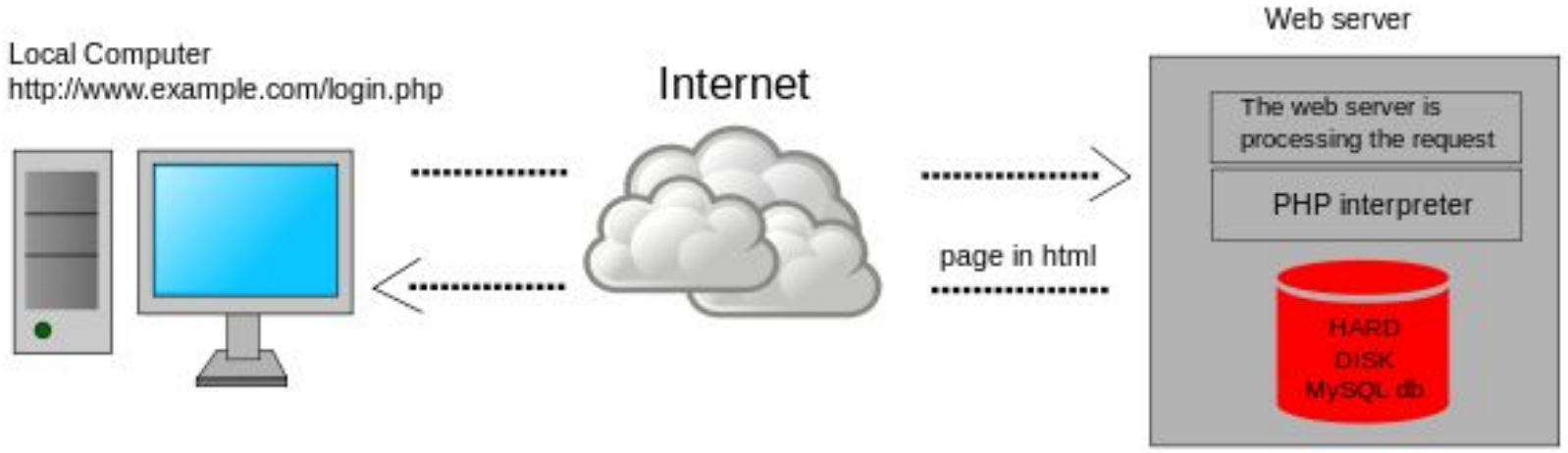
- **Динамическая страница** это веб-страница, сгенерированная программно, в отличие от статичной страницы, которая является просто файлом, лежащим на сервере
- Сервер генерирует HTML-код динамической страницы для обработки браузером
- Динамические страницы обычно обрабатывают и выводят информацию из базы данных
- Пример: Google.translate, Internet-Banking etc.


Client-server ecosystem
May be one or many computers
1
Browser
Operating system
2
URL
http: Request (get)
Internet
Client
3
Parser (php/dot.net)
Web Server
Web Server (Apache/IIS)
Operating System (Linux/Win Server)
Checks content type to see if needs parsing or just served
http:response
Web objects (includes media, zips, etc.)
4
?
JavaScript
CSS
HTML
Application (mail, Database, ..)
Operating system
Support Server
(e.g. Mail, DB,..)
CC-BY-SA 2013 virtualMV (M Verhaart)

SERVER-SIDE ПРОГРАММИРОВАНИЕ

- Сценарии, предназначенные для выполнения на стороне сервера, располагаются на сервере
- Принимая запрос пользователя, сервер запускает программу на выполнение (которая была указана при активировании события)
- В результате выполнения программы, выходные данные передаются web-серверу, а затем клиенту (в виде HTML-страниц)
- Документы сгенерированные на сервере могут содержать скрипты client-side
- Для написания сценариев, работающих на стороне сервера, обычно используются такие технологии, как Perl, ASP, ASP NET, PHP, server-side JavaScript etc.

SERVER-SIDE ПРОГРАММИРОВАНИЕ



- Скрипты server-side выполняются на сервере, и генерируют одни и те же выходы, в не зависимости от браузера клиента, его операционной системы и др.

Дополнительно: http://www.w3.org/wiki/How_does_the_Internet_work#Static_vs._Dynamic_Web_Sites

https://www.ischool.utexas.edu/~hristova/ia/

http://codingcraft.ru/web-programming.php

JAVA SCRIPT - ОБЩЕЕ

- JavaScript является языком программирования (один из самых популярных), используемый для программирования в сетях компьютеров, серверов, смарт-фон-ов и т.д.
- Почти все современные HTML страницы используют JavaScript
- Изучение языка JavaScript необходимо потому, что:
  - HTML необходимо для определения содержания веб-страниц
  - CSS – для определения стиля веб-страницы
  - JavaScript – для программирования поведения сайта

ИСПОЛЬЗОВАНИЕ JAVA SCRIPT

- DOM HTML (**D**ocument **O**bject **M**odel) это официальный стандарт консорциума W3C, используемый для доступа к HTML-элементу какого-то HTML-документа
  - *"The W3C Document Object Model (DOM) is a platform and language-neutral interface that allows programs and scripts to dynamically access and update the content, structure, and style of a document."*
- JavaScript может изменить контент HTML-документа манипулируя с DOM HTML
- JavaScript может быть использован для: изменения HTML-элементов, удаления HTML-элементов, создание HTML-элементов, копирование HTML-элементов и т.д.
- Пример **Изменение контента HTML-элемента**:

Метод **document.getElementById()** один из многочисленных методов используемых с HTML

ПРИМЕР ИЗМЕНЕНИЯ КОНТЕНТА HTML-ЭЛЕМЕНТА

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
</head>
<html>
<body>
    <h1>Пример изменения контента</h1>
    <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
    <button type="button" onclick="functie()">Нажми на кнопку!</button>
    <p id="et">Посмотри результат!</p>
    <script>
        function functie() {document.getElementById("et").innerHTML =
"Измененный контент!!!";}
    </script>
</body>
</html>
```

РЕЗУЛЬТАТ ПРИМЕРА

- До нажатия кнопки:

- После нажатия кнопки:

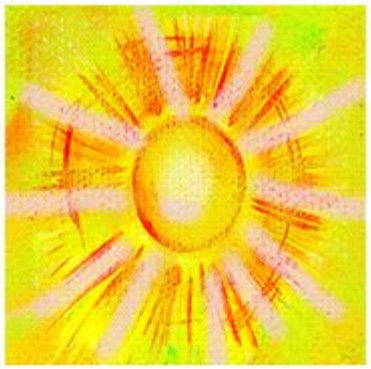
ИСПОЛЬЗОВАНИЕ JAVA SCRIPT

- **Пример изменения HTML-атрибутов**

```
<!DOCTYPE html>
<html>
<body>
<script>
<img id="img" onclick="changeImage()" border="0" src="galben.gif" width="180" height="180">
<p>Da click pe imagine...</p>
function changeImage(){
element=document.getElementById("img")
if (element.src.match("rosu"))
  {  element.src="galben.gif";  }
else
  {  element.src="rosu.gif";  }
}
</script>
</body>
</html>
```

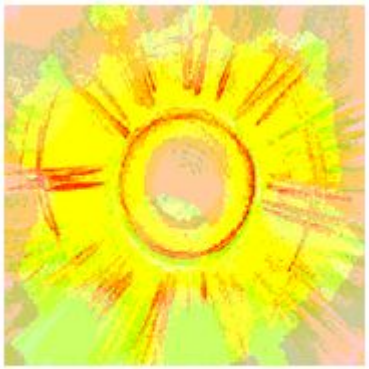
РЕЗУЛЬТАТ ПРИМЕРА

До нажатия:



Da click pe imagine...

После...:



Da click pe imagine...

ИСПОЛЬЗОВАНИЕ JAVA SCRIPT

- **Изменение стилей CSS в HTML-документах**

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
</head>
<html>
<body>
   <h1>Пример изменения контента</h1>
   <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
   <button type="button" onclick="functie()">Нажми на кнопку!</button>
   <p id="et">Посмотри результат!</p>
   <script>
     function functie()   {document.getElementById("et").style.fontSize =
   "25px";}
   </script>
</body>
</html>
```

РЕЗУЛЬТАТ ПРИМЕРА

- До нажатия кнопки:

- После:

ДРУГИЕ ИСПОЛЬЗОВАНИЯ JAVA SCRIPT

- Проверка ввода данных
- Вставка даты, дня недели, времени и т.д.
- Управление событиями: нажатие какой-то кнопки, передвижение мыши, координаты мыши и т.д.
- Вывод предупреждений
- Вызов и выполнение каких-то функций
- И т.д., и т.п.

Тег „SCRIPT”

1. Используется для вставки JavaScript-ов
   - ***<script>****Содержание (коды JavaScript)****</script>***

или

2. Используется для определения ссылки к внешнему файлу со скриптами
   - ***<script src=„fisierScript.js"></script>***

ВОЗМОЖНОСТИ ОПРЕДЕЛЕНИЯ И ВСТАВКИ СКРИПТОВ

1. JavaScript-ы можно поместить за пределами (вне) HTML-документа
   - Представляют отдельные файлы содержащие скрипты, которые можно использовать в одной или нескольких веб-страницах
   - Файлы со сценариями JavaScript имеют расширение **.js**
   - В HTML-документе делается ссылка на файл содержащий скрипты, используя в теге **<script>** атрибут **“src”**. Пример: ***<script src=„fisierScript.js"></script>***
   - Ссылку на JS файл можно сделать в теге **<head>** или **<body>**
2. JavaScript-ы можно поместить в HTML-документе
   - Внутри тега <head>
   - Внутри тега <body>

ПРИМЕР ОПРЕДЕЛЕНИЯ ВНЕШНЕГО СЦЕНАРИЯ

- **Файлы со сценариями (.js) не содержат тег <script>!!!**
- Файл .html, тег **<script>** в теге **<body>**:

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
</head>
<html>
<body>
    <h1>Пример изменения контента</h1>
    <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
    <button type="button" onclick="functie()">Нажми на кнопку!</button>
    <p id="et">Посмотри результат!</p>
    <script src="script_prim.js"></script>
</body>
</html>
```

Файл .js:

```
function functie() {document.getElementById("et").style.fontSize = "25px";}
```

ПРИМЕР ОПРЕДЕЛЕНИЯ ВНЕШНЕГО СЦЕНАРИЯ

- Файл .html, тег **<script>** в теге **<head>**:

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
<script src="script_prim.js"></script>
</head>
<html>
<body>
    <h1>Пример изменения контента</h1>
    <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
    <button type="button" onclick="functie()">Нажми на кнопку!</button>
    <p id="et">Посмотри результат!</p>
</body>
</html>
```

Fişierul .js:

```
function functie() {document.getElementById("et").style.fontSize = "25px";}
```

ВНУТРЕННИЕ JAVASCRIPT-Ы

- Определяются при помощи тега **<script>**, внутри тега **<head>** или **<body>** или и в **<head>** и в **<body>**
- Если сценарий определяются в тег **<body>**, лучше всего его определить в конце HTML-документа – это уменьшит время загрузки и вывода веб-ресурса

ПРИМЕР ОПРЕДЕЛЕНИЯ ВНУТРЕННЕГО СЦЕНАРИЯ

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
<script>
    function functie() {document.getElementById("et").style.fontSize = "25px";}
</script>
</head>
<html>
<body>
    <h1>Пример изменения контента</h1>
    <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
    <button type="button" onclick="functie()">Нажми на кнопку!</button>
    <p id="et">Посмотри результат!</p>
</body>
</html>
```

ПРИМЕР ОПРЕДЕЛЕНИЯ ВНУТРЕННЕГО СЦЕНАРИЯ

```
<!DOCTYPE html>
<head><title>Пример использования JS</title>
<link rel="stylesheet" type="text/css" href="stil1.css">
</head>
<html>
<body>
    <h1>Пример изменения контента</h1>
    <p>JavaScript может изменить контент какого-то HTML-элемента:</p>
    <button type="button" onclick="functie()">Нажми на кнопку!</button>
    <p id="et">Посмотри результат!</p>

    <script>
        function functie() {document.getElementById("et").style.fontSize = "25px";}
    </script>
</body>
</html>
```

ВЫХОДЫ В JAVASCRIPT

- JavaScript не имеет специальных функций для печати или представления каких-то выходов
- В HTML, JavaScript, можно использовать только для **манипулирования** элементами HTML
- Чтобы получить доступ к какому-то HTML- элементу, при помощи JavaScript-ов рекомендуется использовать метод **document.getElementById(*id*)**
- Для идентификации элемента которым необходимо манипулировать необходимо использовать глобальный атрибут „**id**”
- А для того чтобы была возможность ссылаться на содержание какого-то HTML-элемента используется свойство **innerHTML** (устанавливает или возвращает содержимое HTML-элемента)
- Основная форма: ***HTMLElementObject*.innerHTML=*text*** (устан)

ПРИМЕР МАНИПУЛИРОВАНИЯ ВЫХОДОМ

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Нажми на кнопку чтобы увидеть результат манипулирования содержанием элемента - удаление контента (innerHTML)</p>
<button onclick="functie1()">Click</button>
<p id="et2">Нажми на кнопку чтобы увидеть результат манипулирования содержанием элемента - изменение контента (innerHTML)</p>
<button onclick="functie2()">Click</button>
<script>function functie1() {    document.getElementById("et1").innerHTML = ""; }
</script>
<script>
 function functie2() { document.getElementById("et2").innerHTML = "Надеюсь все поняли!!!???";}
</script>
</body>
</html>
```

РЕЗУЛЬТАТ ПРИМЕРА

Нажми на кнопку чтобы увидеть результат манипулирования содержанием элемента - удаление контента (innerHTML)

Click

Нажми на кнопку чтобы увидеть результат манипулирования содержанием элемента - изменение контента (innerHTML)

Click

После нажатия кнопок:

Click

Надеюсь все поняли!!!???

Click

ВЫВОД ДАННЫХ НА ЭКРАН

- Метод ***write()*** выводит HTML выражение или JavaScript код в HTML-документ
- Если данный метод вызывается уже после загруженного документа, все выведенное будет удалено и выведено то что написано в методе ***write()***
- В общем этот метод используется для тестирования
- Синтаксис: **document.write(exp1,exp2,exp3,...)**

Пример:

```
<body>
<script>
    document.write("<h1>Уважаемый пользователь!</h1><p>Просьба быть внимательным!</p>");
</script>
```

ПРИМЕР ВЫВОДА ДАННЫХ НА ЭКРАН

*… Текущая дата: *

*<script>**document.write(Date())**; </script>…*

Rezultat:

Текущая дата:

Tue Nov 04 2014 21:13:20 GMT+0200 (GTB Standard Time)

Или:

*…
*

***<button onclick="functie3()">Нажми чтобы узнать текущую дату</button>**
*

<script>

function functie3() {

document.write(Date());}

</script> …

Tue Nov 04 2014 21:16:57 GMT+0200 (GTB Standard Time)

ВВОД КОМЕНТАРИЕВ

- JavaScript является языком для написания сценариев
- Операторы в JavaScript это «строки команд", выполненные в веб-браузере
- JS-оператор заканчивается "**;**"
- Коды JavaScript могут быть комментированы
  - Комментарии размещенные на одной строке, размещаются после «//»
  - Пример: *document.getElementById("et1").innerHTML = "";* //заменяется контент
  - Комментарии размещенные на несколько строк, размещаются между /* и */

ПЕРЕМЕННЫЕ В JAVASCRIPT

- В JavaScript-е переменные представляют „контейнеры” для различных значений
- В JS переменные могут иметь короткие названия (a, b, x, z), но рекомендуется использовать названия со смыслом: имя, возраст, стоимость etc.
- Имена переменных могут содержать буквы, цифры, знак подчеркивания, и знак доллара ($)
- Имена переменных должны начаться на букву
- Имена переменных могут, также, начаться на знак $ или _ (но это не очень практично)
- Имена переменных регистрозависимы
- Зарезервированные слова в JavaScript не могут быть использованы в качестве имен переменных

ПЕРЕМЕННЫЕ В JAVASCRIPT. II

- Переменным можно присвоить значения или выражения
  - Пример: ***cost=200;*** или ***cost = cost+30;***
  - Присвоение производится при помощи символа "="
- Переменным можно присвоить несколько типов данных
  - Данные типа текст или типа «string»
    - Пример: ***numePrenume = "Cutarescu Ana";***
    - Значения переменных типа «текст» заключаются между кавычками или апострофом
  - Числовой тип данных
    - Прмер: ***varsta=50;***
    - Значения не заключаются между кавычками (только в том случае когда цифры должны рассматриваться как текстовые символы)
  - В JS переменные объявляются с использованием зарезервированного слова „***var***"

ПРИМЕР ОПРЕДЕЛЕНИЯ ПЕРЕМЕННЫХ В JS

```
<!DOCTYPE html>
<html>
<body>
<p>Определение переменных</p>
<p id="et1"></p><p id="et2"></p><p id="et3"></p>
<script>
var nume = "Иванов";
var prenume = "Иван";
var varsta = 35;
document.getElementById("et1").innerHTML = nume;
document.getElementById("et2").innerHTML = prenume;
document.getElementById("et3").innerHTML = varsta;
</script>
</body>
</html>
```

Результат:

Определение переменных

Иванов

Иван

35

ОПЕРАЦИИ В JS

- Над числовыми переменными можно применить операцию сложения
  - Пример:

var adaos = 5;

var cost = 500 + adaos;

- Над переменными строчного типа можно применить операцию конкатенации
  - Пример:

var nume = "Ivanov";

var datePersonale = nume + " Ivan";

  - Другой пример:

var cod = "+373";

var nrTelefon = cod + 334455;

+373334455

ДЛИНА ТЕКСТОВОЙ СТРОКИ

- В JS для определения длинны текстовой строки используется свойство ***length***

```
<!DOCTYPE html>
<html>
<body>
<p>Определение переменных</p>
<p id="et3"></p><p id="et4"></p>
<script>
    var cod = "+373";
    var nrTelefon = cod + 334455;
    document.getElementById("et3").innerHTML = nrTelefon;
    document.getElementById("et4").innerHTML = nrTelefon.length;
</script>
</body>
</html>
```

Определение переменных

+373334455

10

ДРУГИЕ ТИПЫ ДАННЫХ. Массивы в JS

- Значения массива в JS включаются между **[** и **]**
- Элементы, разделены запятой
- Пример:

var персДанные = [«Иванов», «Иван», «1990»];

- Индексирование элементов, для доступа к элементам массива, начинается с „0”
- Пример: для доступа к году рождения, из ***персДанных*** пишется ***персДанные[2]***

```
…<p id="et1">
<script>
   var persDanniie = [«Ivanov», «Ivan», «1990»];
   document.getElementById("et1").innerHTML = persDanniie[2];
</script>…
```

ФУНКЦИИ В JS

- В JS функция это блок кодов написанный с целью решения какой-то конкретной задачи
- Функция выполняется вызывая её
- Синтаксис:
  - Функция определяется используя ключевое слово „***function***"
  - Следует название функции и круглые скобки: ***имяФункции()***
  - Имя функции может содержать буквы, цифры, знак подчеркивания, знак $ (те же правила как для переменных)
  - Между скобками можно включить параметры, разделенные запятой
  - Программный код, который исполнится, включается между фигурными скобками
- Итак, основная форма:

function имяФункции(параметр1, параметр2, …)

{ обьявление1; объявление2;…

}

ПРИМЕР ОПРЕДЕЛЕНИЯ ФУНКЦИИ

- Параметры или аргументы функции используются в качестве местных переменных, внутри функции
- Пример:

```
…<p id="et1">
<script>
function functieCalcul(salariu, adaos) {
    return salariu+adaos;}
document.getElementById("et1").innerHTML =
   functieCalcul(3000, 500);
</script>…
```

Результат: 3500

ВЫЗОВ ФУНКЦИИ

- Программный код, определенный внутри функции, выполнится при вызове функции
- Функцию можно вызвать несколькими способами:
  - При активации какого-то события (пользователь нажимает какую-то кнопку, браузер загрузил страницу и др.)
  - Функция вызывается кодами JavaScript или автоматически
- После выполнения функции, как результат её вызова, JS продолжит выполнение следующих операторов/ объявлений

Объекты в JS

- Язык программирования JS это ОО зык программирования
- Любой объект характеризуется **свойствами** (человек: фамилия, имя, возраст, кол. детей etc.) и определяется его поведение, при помощи операций/ методов (человек: ходит, ест, разговаривает, прыгает etc.)
  - Значения свойств отличаются для каждого объекта
  - Поведение, определяется для всех объектов какой-то группы
  - Методы определяются используя функции
- Доступ к свойству объекта осуществляется так:
- ***названиеОбьекта.названиеСвойства*** или ***названиеОбьекта [названиеСвойства]***

СОБЫТИЯ В JS

- HTML-события представляют „что-то” что происходит с HTML-элементами страницы – что-то что может сделать браузер или что-то что может сделать пользователь
  - Страница была полностью загружена
    - *<body onload="afisare()">*
  - Было изменено значение какого-то поля или входящей переменной
  - Была нажата какая-то кнопка пользователем
- JavaScript-ы используемые в какой-то web-странице могут реагировать на эти события
- HTML позволяет контролировать события используя соответствующие атрибуты
- Основная форма:

<HTMLэлемент названиеСобытия =“JavaScript”>

ПРИМЕР КОНТРОЛЯ СОБЫТИЯ

```
<!DOCTYPE html>
<html>
<body>
<button onclick="getElementById('et').innerHTML=Date();">Нажми чтобы узнать время</button>
<p id="et"></p>
</body>
</html>
```

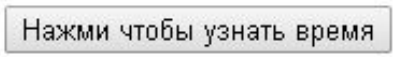
Mon Nov 10 2014 12:20:50 GMT+0200 (GTB Standard Time)

Внимание в использование кавычек!!!

ПРИМЕР КОНТРОЛЯ СОБЫТИЯ. II

- В предыдущем примере JS изменил контент HTML-элемента с указанном ID
- Можно использовать метод „**this**” для того чтобы изменить содержание текущего HTML элемента

```
<!DOCTYPE html>
<html>
<body>
<button onclick="this.innerHTML=Date();">Нажми чтобы узнать время</button> </body>
</html>
```

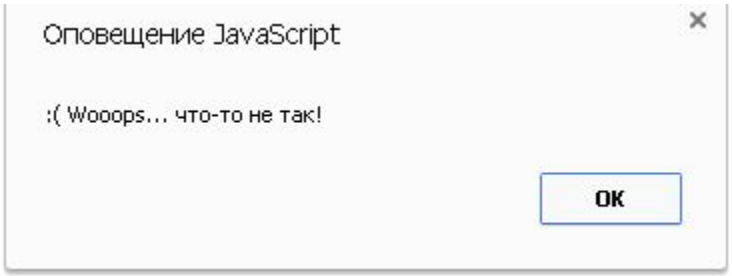


После нажатия:

ПРИМЕР КОНТРОЛЯ СОБЫТИЯ. III

- События можно контролировать используя функции

```
<!DOCTYPE html>
<html>
<body>
<button onclick="mesaj()">Отправить данные</button>
<script>
    function mesaj() {
       alert(":( Wooops... что-то не так!"); }
</script>
</body>
</html>
```



ПРИМЕР КОНТРОЛЯ СОБЫТИЯ. IV

… <button onclick="afiseazaData()">Нажми чтобы узнать время </button>

* *

<script>

function afiseazaData() { document.getElementById("et").innerHTML = Date();}

</script> …

Другие события которые можно контролировать: **onmouseover** – пользователь ставит мышь поверх какого-то HTML-элемента, **onload** – браузер заканчивает загрузку страницы, **onchange** – какой-то элемент был изменен etc.

Больше информаций: http://www.w3schools.com/jsref/dom_obj_event.asp

ПРИМЕР С 2-мя СОБЫТИЯМИ

```
<!DOCTYPE html>
<html
<head>
<script>
function afiseazaData() {
document.getElementById("data").innerHTML = "Astazi este: " + Date();}
function daNume(valoare) {
   document.getElementById("nume").innerHTML = "Bine ati venit: " + valoare;}
</script>
</head>
<body onload="afiseazaData();">
<p id="data">Aici va fi afisata data</p>
<p id="nume">Aici va fi afisat un mesaj de salutare</p>
Introdu numele in caseta de mai jos pentru a vedea efectul metodei 'onchange'...<input
type="text" name="txt" onchange="daNume(this.value)">
</body></html>
```

РЕЗУЛЬТАТ

Astazi este: Tue Apr 21 2015 22:18:53 GMT+0300 (GTB Daylight Time)

Aici va fi afisat un mesaj de salutare

Introdu numele in caseta de mai jos pentru a vedea efectul metodei 'onchange'...

Astazi este: Tue Apr 21 2015 22:18:53 GMT+0300 (GTB Daylight Time)

Bine ati venit: Ivan

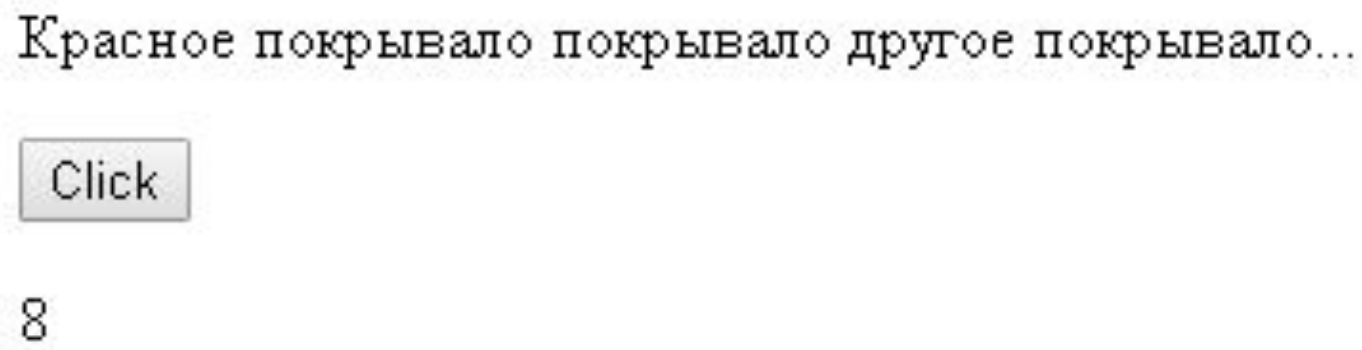
Introdu numele in caseta de mai jos pentru a vedea efectul metodei 'onchange'... Ivan

МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Метод **indexOf()** возвращает индекс (позицию), первого появления какого-то текста в строке
- Пример:

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.indexOf("покрывало");}
</script>
</body>
</html>
```

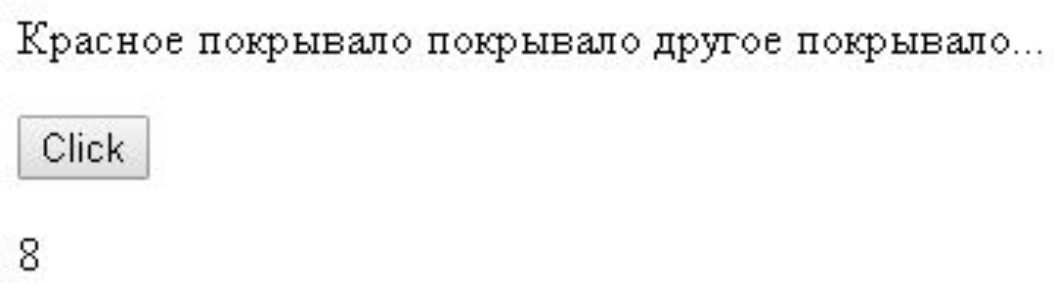
РЕЗУЛЬТАТ ПРИМЕРА



МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Метод **search()** ищет подстроку в символьной строке, возвращая позицию начала

```
…<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.search("покрывало");}
</script>…
```

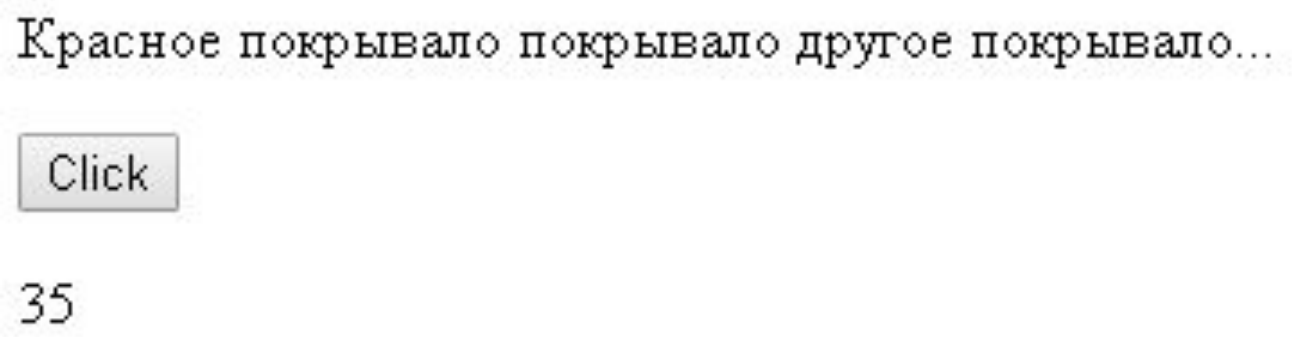


МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Метод **lastIndexOf()** возвращает индекс (позицию), последнего появления какого-то текста в символьной строке
- Пример:

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.lastIndexOf("покрывало");}
</script>
</body>
</html>
```

РЕЗУЛЬТАТ ПРИМЕРА



МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

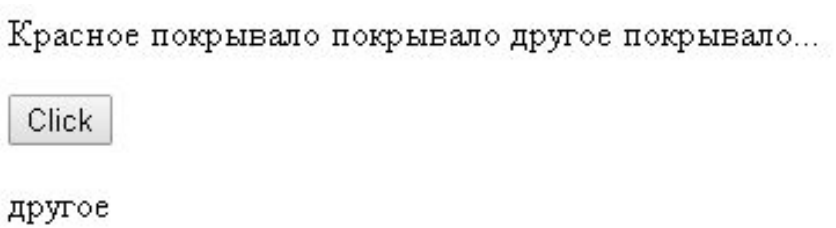
- Чтобы извлечь подстроку из строки можно использовать следующие методы:
  - slice(start, end)
  - substring(start, end)
  - substr(start, length)

- *Примечание: Метод **slice** и **substring** имеют тот же эффект*

ПРИМЕР С МЕТОДОМ „SLICE”

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>

<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.slice(28,34);}
</script>
</body>
</html>
```

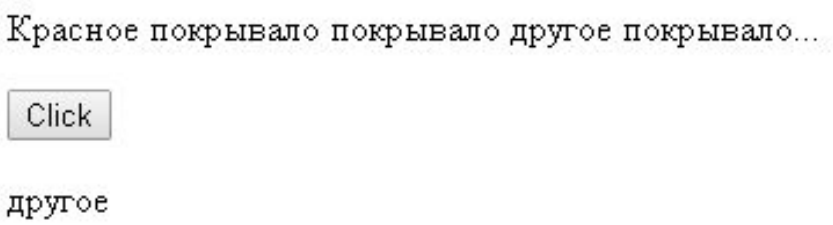


Попробуйте **document.getElementById("et2").innerHTML = sir.slice(8);**

ПРИМЕР С МЕТОДОМ „SUBSTR”

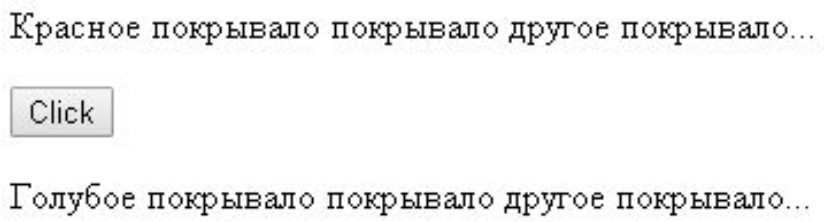
```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>

<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.substr(28,6);}
</script>
</body>
</html>
```



МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Метод **Replace()** заменяет одно значение на другое
- Пример:



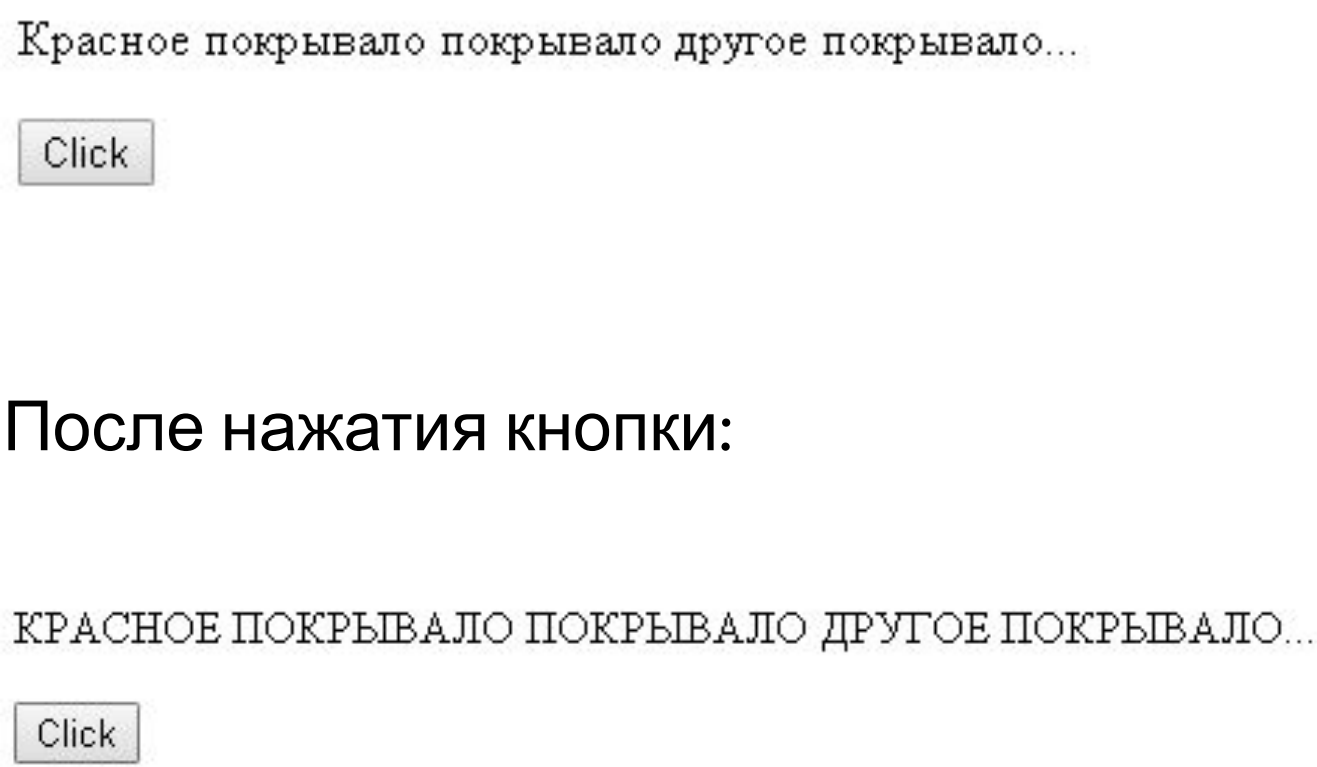

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.replace("Красное","Голубое");}
</script>
</body>
</html>
```

МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Методы для преобразования строчных букв в прописные и наоборот
  - **toUpperCase()**
  - **toLowerCase()**
- Пример:

```
…<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et1").innerHTML = sir.toUpperCase();}
</script>…
```

РЕЗУЛЬТАТ ПРИМЕРА



После нажатия кнопки:

МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Метод **concat()** – объединение / конкатенация сток
- Пример:

```
…<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<script>
function functie() {
    var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et1").innerHTML = sir.concat(" ", "вот так :)");}
</script>…
```

РЕЗУЛЬТАТ ПРИМЕРА

Красное покрывало покрывало другое покрывало...

Click

После нажатия кнопки:

Красное покрывало покрывало другое покрывало... вот так :)

Click

МЕТОДЫ JS ДЛЯ РАБОТЫ С СИМВОЛЬНЫМИ СТРОКАМИ

- Чтобы извлечь символ из текстовой строки могут быть использованы следующие методы
  - **charAt(position)**
  - **charCodeAt(position)**
- Пример:

```
…<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
        var sir = document.getElementById("et1").innerHTML;
    document.getElementById("et2").innerHTML = sir.charAt(0);}
</script>…
```

РЕЗУЛЬТАТ ПРИМЕРА

Красное покрывало покрывало другое покрывало...

Click

К

Применив метод **charCodeAt(0),** для этого примера

Красное покрывало покрывало другое покрывало...

Click

1050

ПРЕОБРАЗОВАНИЕ ТЕКСТОВОЙ СТРОКИ В МАССИВ

- Преобразование производится используя метод **split()**
- Пример:

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
        var sir = document.getElementById("et1").innerHTML;
    var vector = sir.split(" ");
    document.getElementById("et1").innerHTML = vector[0];}
</script>
</body>
</html>
```

РЕЗУЛЬТАТ ПРИМЕРА

Красное покрывало покрывало другое покрывало...

Click

Результат: Красное

Click

*…document.getElementById("et1").innerHTML = **vector[4]**;}…*

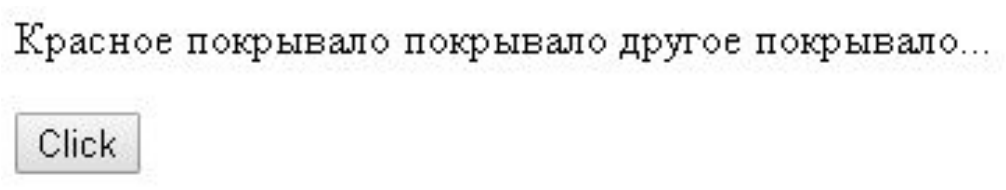
покрывало...

Click

ОПРЕДЕЛЕНИЕ ТИПА ПЕРЕМЕННОЙ

- Для определения типа переменной используется метод **typeof()**
- Пример:

```
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
        var sir = document.getElementById("et1").innerHTML;
      var vector = sir.split(" ");
      document.getElementById("et1").innerHTML = typeof(sir);}
</script>…
```



ОПРЕДЕЛЕНИЕ ТИПА ПЕРЕМЕННОЙ

```
…var vector = sir.split(" ");
    var varsta = 56;
    document.getElementById("et1").innerHTML = typeof(varsta);…
…
…document.getElementById("et1").innerHTML = typeof(vector);…
```

JS ФУНКЦИЯ isNaN()

- В JS есть резервированное слово **NaN**, которое указывает если какое-то значение не является числом (Not a Number)
- **isNaN()** - глобальная функция которая может быть использована чтобы определить, является ли некоторое значение **не**-числовым

ПРИМЕР NaN

```
<!DOCTYPE html>
<html>
<body>
<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
        var sir = document.getElementById("et1").innerHTML;
        var vector = sir.split(" ");
        var varsta = 56;
        document.getElementById("et1").innerHTML = varsta*sir;}
</script></body>
</html>
```

А для:

…document.getElementById("et1").innerHTML = varsta*2;…

ПРИМЕР isNaN()

```
…<p id="et1">Красное покрывало покрывало другое покрывало...</p>
<button onclick="functie();">Click</button>
<p id="et2"></p>
<script>
function functie() {
        var sir = document.getElementById("et1").innerHTML;
        var vector = sir.split(" ");
        var varsta = 56;
        document.getElementById("et1").innerHTML = isNaN(varsta*2);}
</script>…
```

false

Click

true

Click

document.getElementById("et1").innerHTML = **isNaN(vector);**

JS МЕТОДЫ ДЛЯ РАБОТЫ С ЧИСЛАМИ

- Все методы используемые для работы с числами могут быть использованы для любых типов чисел, констант, переменных или выражений
- Метод **toString()** преобразовывает число в строку
- Пример:

```
…<p id="et1"></p><p id="et2">
<script>
document.getElementById("et1").innerHTML = functieCalcul(3000, 500).toString();
var tip = functieCalcul(3000, 500).toString();
document.getElementById("et2").innerHTML = typeof(tip);
function functieCalcul(salariu, adaos) {    return salariu+adaos;}
</script>…
```

3500

string

JS МЕТОДЫ ДЛЯ РАБОТЫ С ЧИСЛАМИ

- Метод **toFixed()** возвращает строку с определенным, указанном количеством знаков после запятой
- Пример:

```
…<p id="et1"></p><p id="et2">
<script>
document.getElementById("et1").innerHTML = functieCalcul(3000, 500.6).toFixed(2);
var tip = functieCalcul(3000, 500.6).toFixed(2);
document.getElementById("et2").innerHTML = typeof(tip);
function functieCalcul(salariu, adaos) {    return salariu+adaos;}
</script>…
```

3500.60

string

Прим: Смотрите и другие методы применяемые для работы со строками и числами

!!!

- **3** понятия выученные сегодня
- **2** вопроса которые возникли сегодня
- **1** предложение для следующей пары