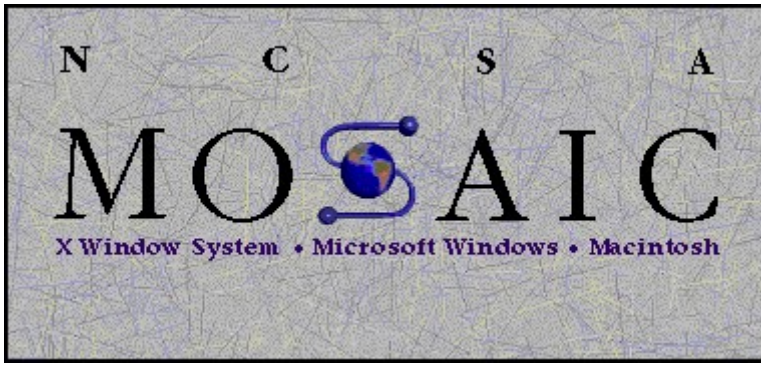
WORLD WIDE WEB

html **url** **http**

История www

1989 Тим Бернерс-Ли (CERN)



1993 Mosaic

1994 Netscape

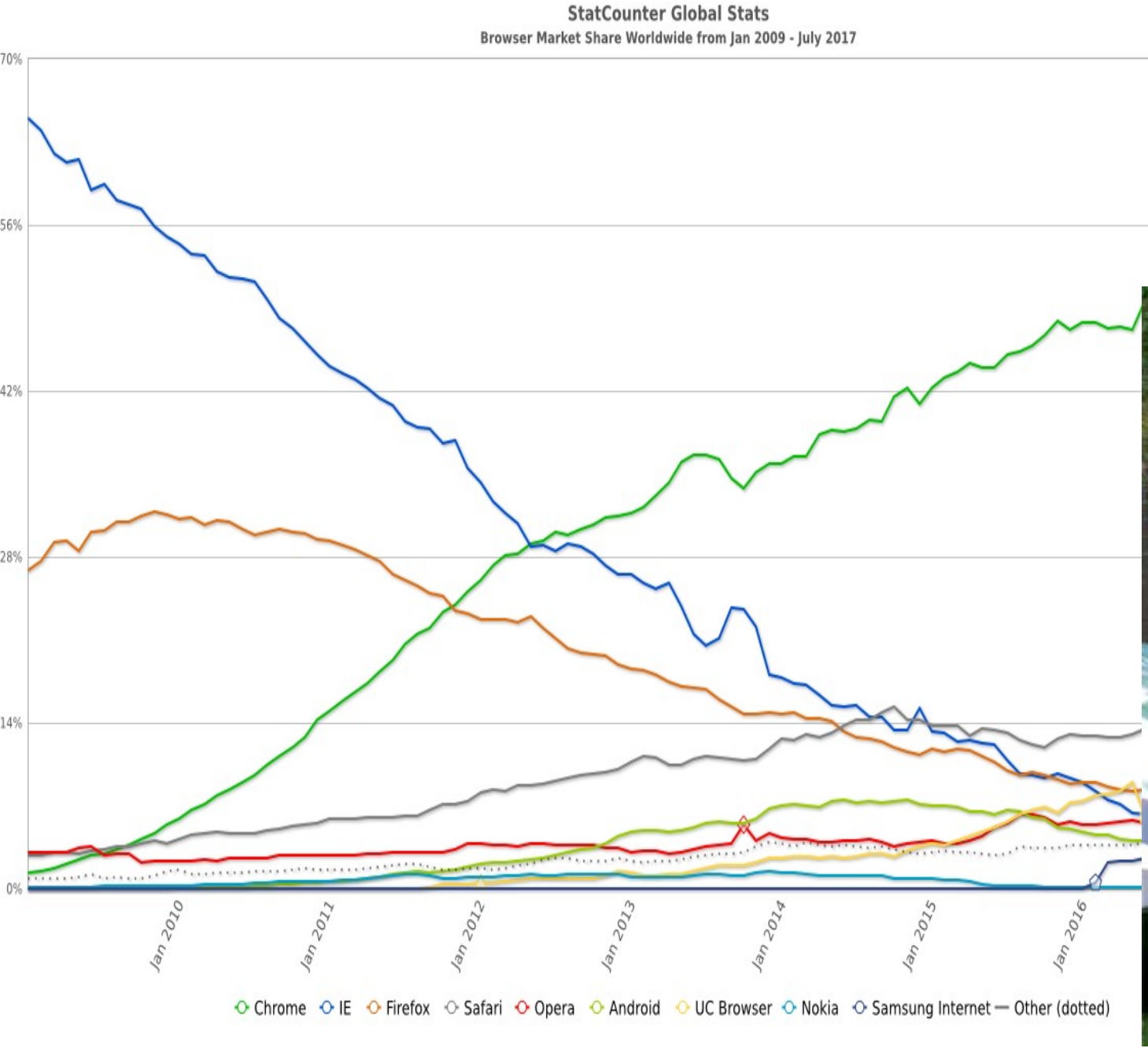
1995 The World Wide Web Consortium

1996 IE3

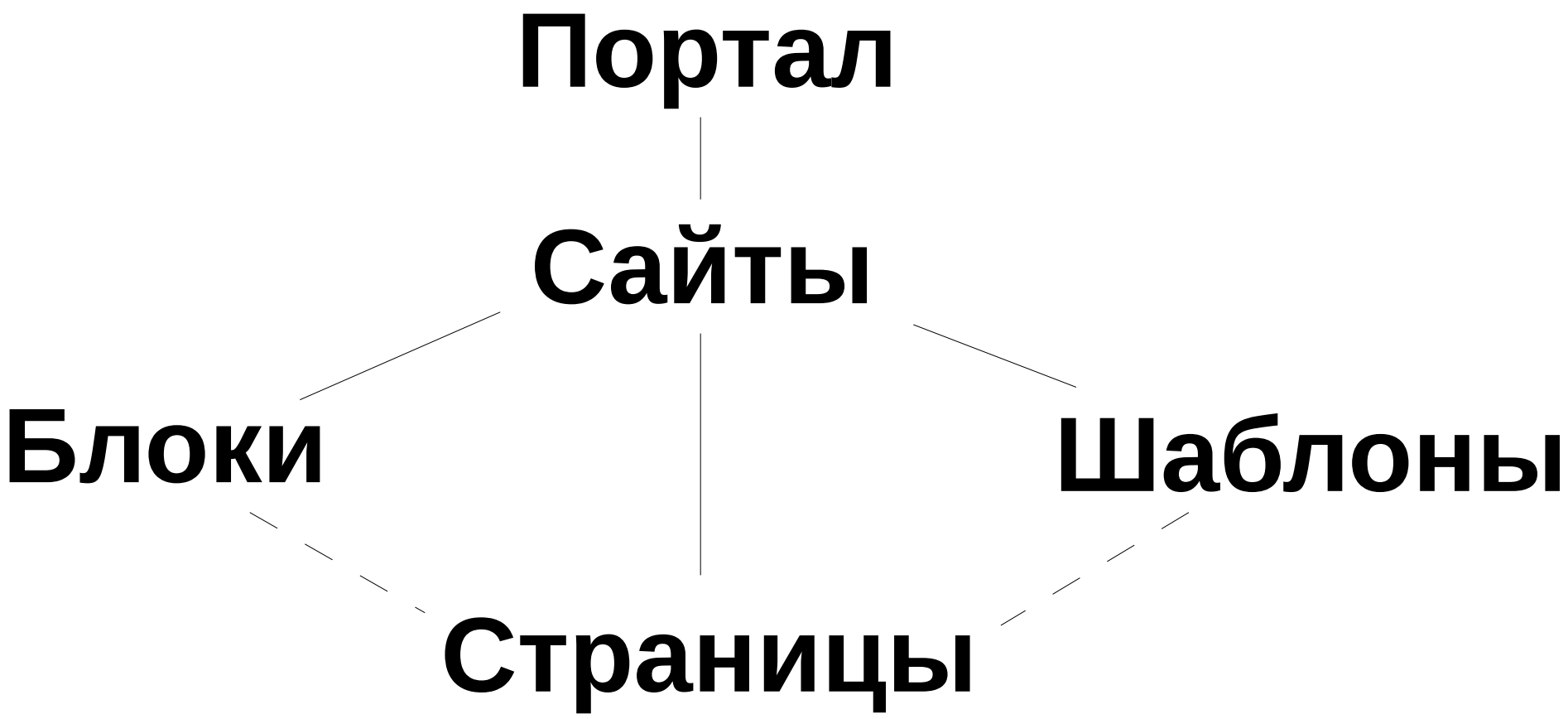
1998 XML

2012 HTML5

Война браузеров

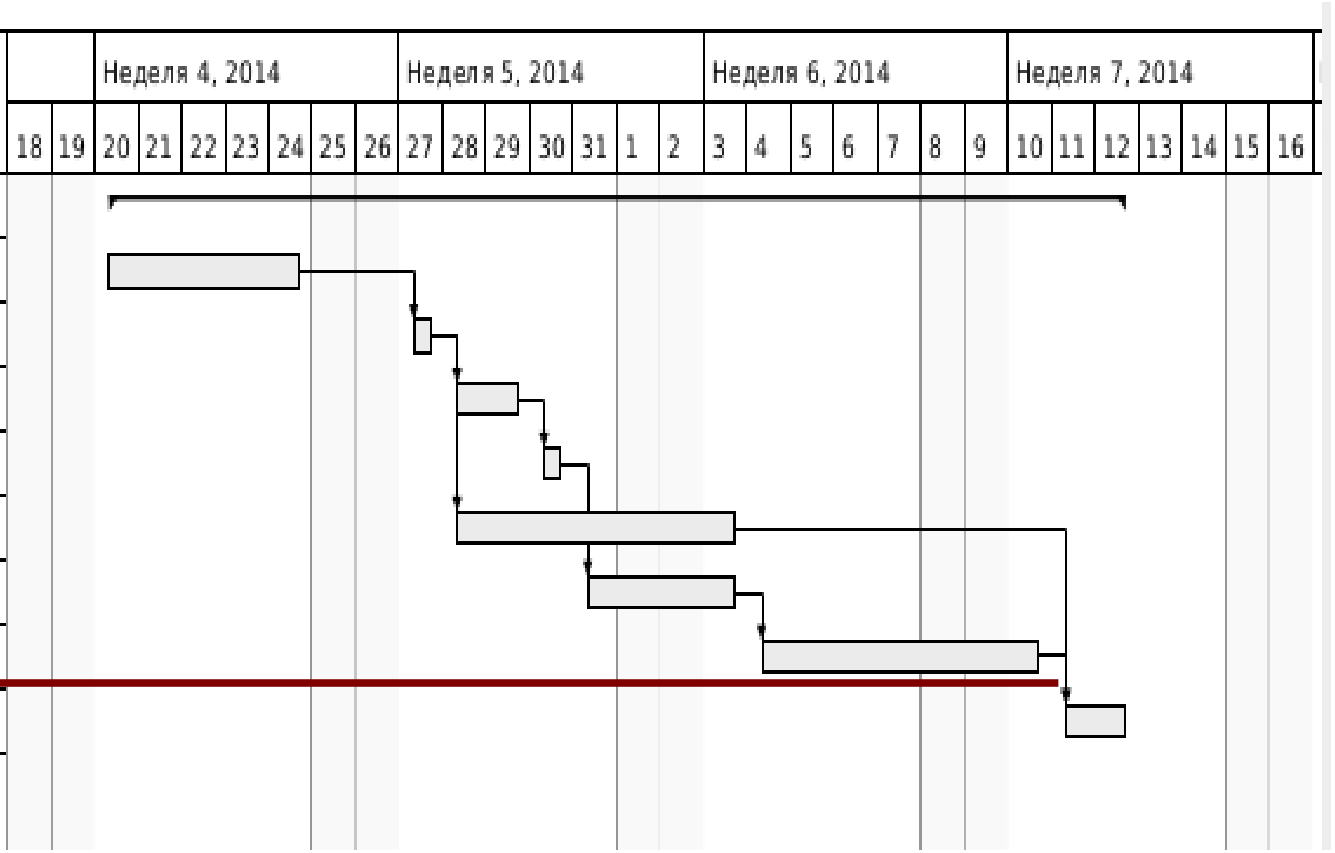


Логическая структура сайта



Разработка сайта

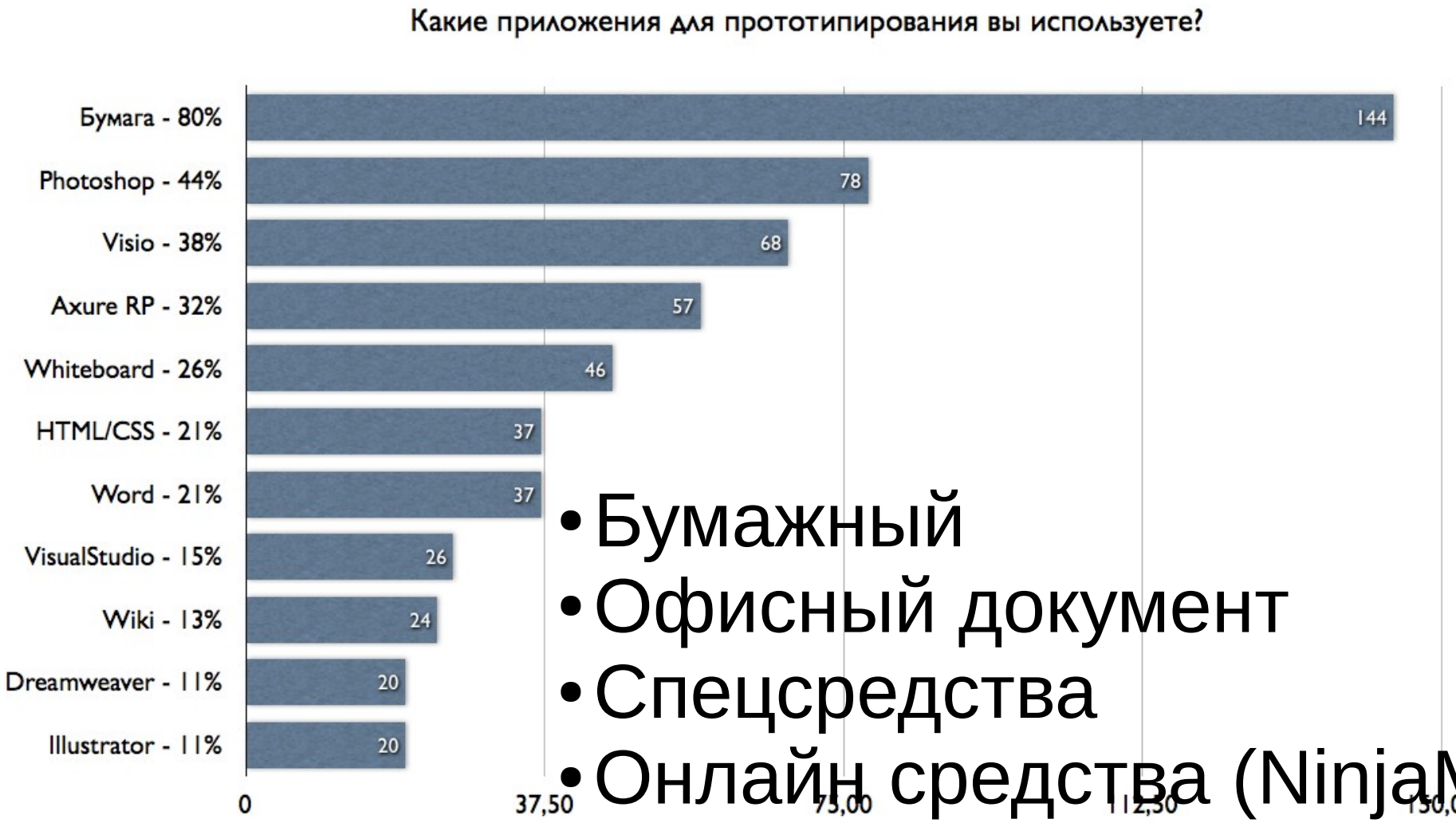
Название	Работа
Разработка сайта	**23д**
Проектирование	5д
Согласование	1д
Дизайн первой страницы	2д
Согласование дизайна	1д
Программирование адм. интерфейса	5д
Технический дизайн	2д
Верстка html, css, javascript	5д
Программирование польз. интерфейса	2д



Последовательность проектирования

- Прототип
- Техническое задание
- Смета
- Календарный план
- Договор

Прототипы



- Бумажный
- Офисный документ
- Спецсредства
- Онлайн средства (NinjaMock)
- «Недосайт»

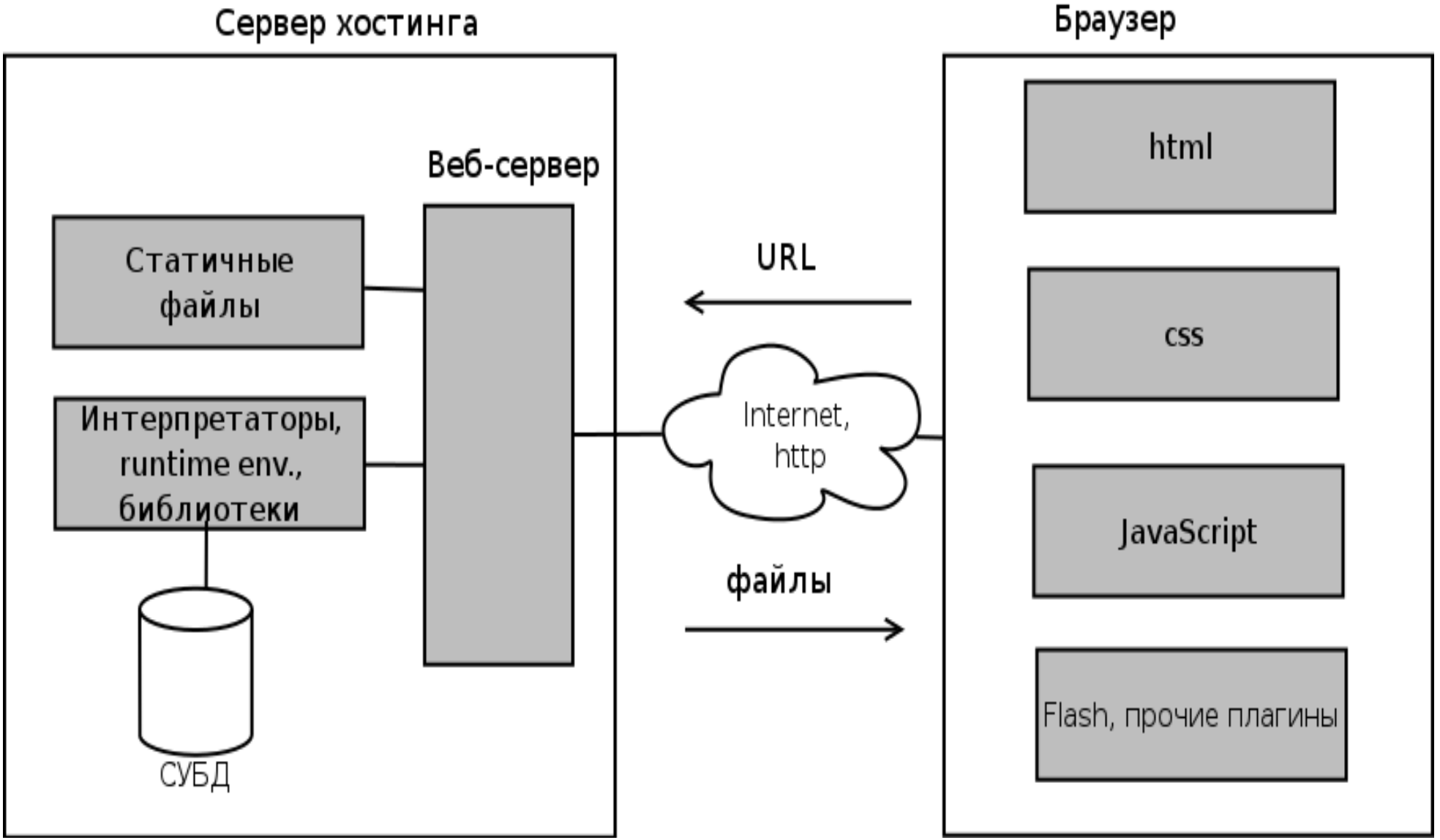
Смета

N этапа	Наименование	Стоимость, р.
1	Разработка прототипа системы	x 000
2	Разработка технического задания	x 000
3	Разработка программного интерфейса ЛК с АИС ДаДаДа	x 000
4	Верстка веб-страниц	x 000
5	Разработка программного модуля аутентификации	x 000
6	Разработка программного модуля отображения статуса клиента	x 000
7	Разработка программного модуля отображения статистики компенсаций	x 000
8	Разработка программного модуля отображения новостей	x 000
9	Разработка программного модуля учета прочтения новостей	x 000
	Итого	**xx 000**

ТЕХНИЧЕСКОЕ ЗАДАНИЕ

- ГОСТ34
- ГОСТ19
- IEEE STD 830

Программная структура сайта

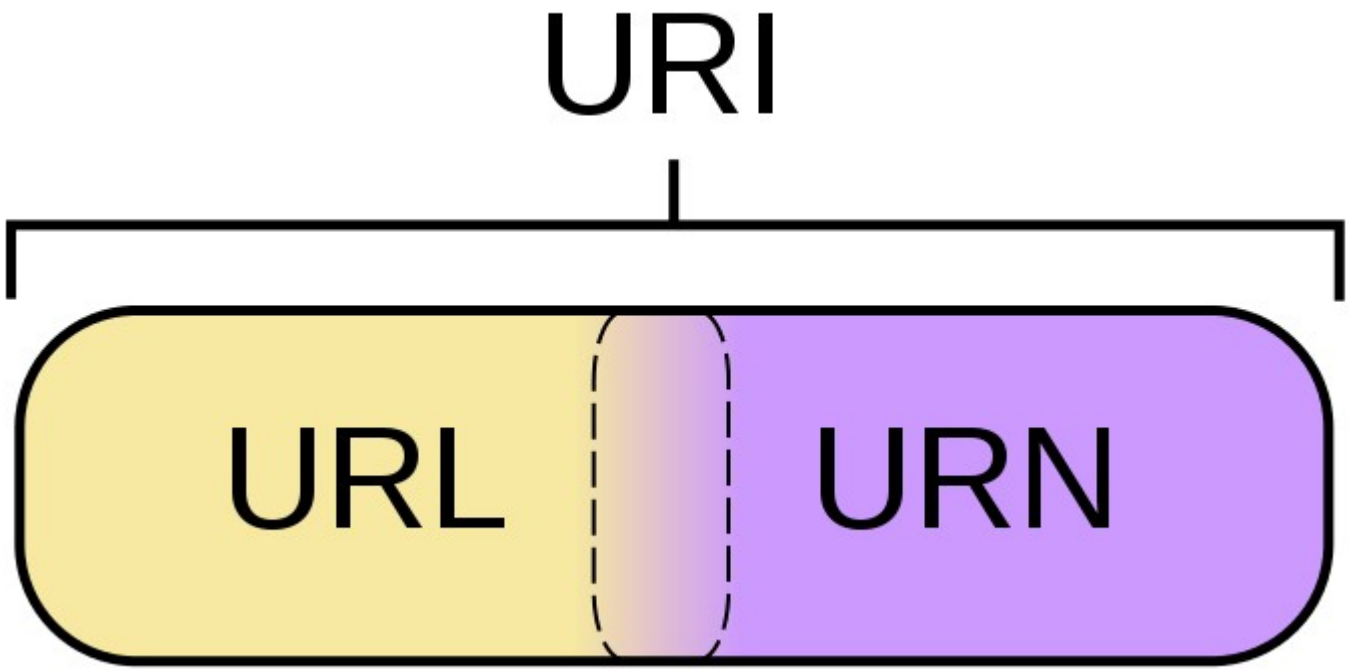


Hypertext Transfer Protocol

```
$ telnet www.w3.org 80
Trying 128.30.52.37...
Connected to www.w3.org.
Escape character is '^]'.
GET /
HTTP/1.1 200 OK
Date: Sat, 18 Jan 2014 16:55:18 GMT
Server: Apache/2
Content-Location: Home.html
Vary: negotiate,accept
TCN: choice
Last-Modified: Sat, 18 Jan 2014 10:16:06 GMT
ETag: "8a5d-4f03bf2516580;89-3f26bd17a2f00"
Accept-Ranges: bytes
Content-Length: 35421
Cache-Control: max-age=600
Expires: Sat, 18 Jan 2014 17:05:18 GMT
P3P: policyref="http://www.w3.org/2001/05/P3P/p3p.xml"
Connection: close
Content-Type: text/html; charset=utf-8

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/x
html1/DTD/xhtml1-strict.dtd">
<html>
<head>
<title>World Wide Web Consortium (W3C)</title>
```

Uniform resource locator

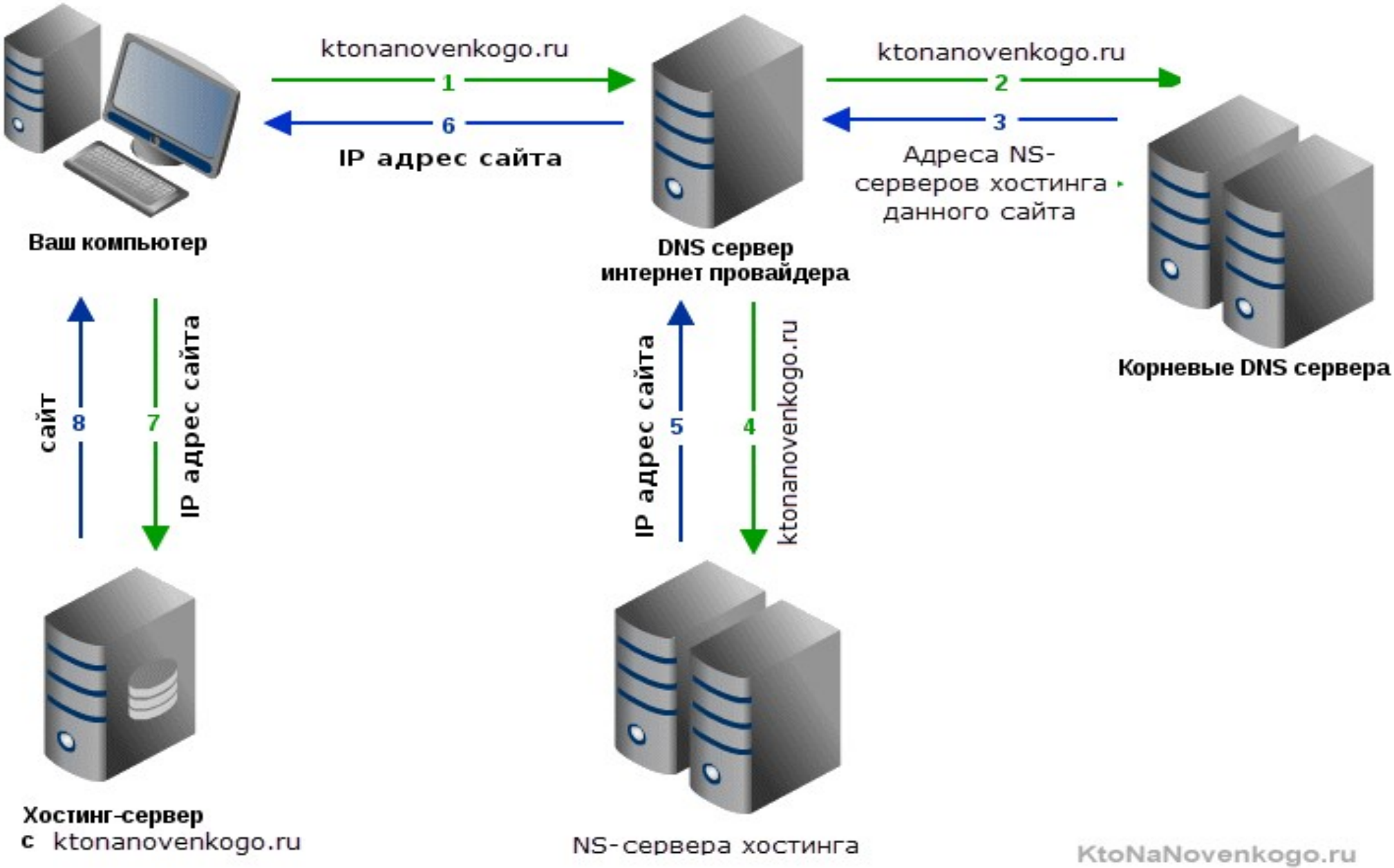


протокол://доменноеИмя/путь

http://www.lostfilm.tv/Static/icons/cat_arrow.jpeg
http://google.com/calendar

Настройка рабочего места

Сервера имен



Хостинг

https://stud.kdenisb.org:1500/ispmgr

Пользователь u??
Пароль U??hse

Языки разметки текста

Разметка — добавление важности документу. Является дополнительным текстом, включенным в документ, некоторым образом отделяемым от содержимого документа.

TeX, html, xml, xhtml, sgml

```
The quadratic formula is $-b \pm \sqrt{b^2 -
4ac} \over 2a$
\bye
```

The quadratic formula is $\frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$

```
<ТЭГ АТРИБУТ='значение'>текст</ТЭГ>

<ТЭГ />
```

sgml

Standard Generalized Markup Language — метаязык для определения языков разметки документов.

- Лексика (SGML Declaration)
- Синтаксис (Document Type Definition)
- Размеченный документ

Документ sgml

<memo>
<to>Дедушке</to>
<from>Ваня</from>
<date>5 февраля 2013 г.</date>
<subject>Подарок</subject>
<text>
<para>
Пришли пожалуйста подарок!
</para>
<para>
А то я приеду к тебе в гости. И ты узнаешь!
</para>
</text>
</memo>

DTD sgml

```
<memo>
<to>Дедушке</to>
<from>Ваня</from>
<date>5 февраля 2013 г.</date>
<subject>Подарок</subject>
<text>
<para>
Пришли пожалуйста подарок!
</para>
<para>
```

```
<!DOCTYPE memo [
<!ELEMENT memo O O ((to & from & date &
subject?), text) >
<!ELEMENT text - O (para+) >
<!ELEMENT para O O (#PCDATA) >
<!ELEMENT (to, from, date, subject) - O (#PCDATA) >
]>
```

HyperText Markup Language

Гипертекст — это форма организации текстового материала, при которой его единицы представлены не в линейной последовательности, а как система явно указанных возможных переходов, связей между ними.

<p>
HTML — гипертекстовый язык разметки документов, интерпретируемый и отображаемый браузерами в виде документа в удобной для человека форме.

Структура документа HTML

```
<!ELEMENT HTML 0 0 (HEAD, BODY)>
```

```
<!DOCTYPE HTML PUBLIC
"-//W3C//DTD HTML 4.01//EN"
"http://www.w3.org/TR/html4/strict.dtd">
<html>
<head>
.....
</head>
<body>
.....
</body>
</html>
```

Заголовок HTML

```
<head>
<meta http-equiv="Content-Type"
content="text/html; charset=utf-8" />
<title>Портал муниципальных образований
Пермского края</title>
<meta name="description" content="Портал
муниципальных районов Пермского края" />
<meta name="keywords" content="Пермь, районы
пермского края, Пермский край,
администрация" />
<link href="/css/style.css" rel="stylesheet"
type="text/css" />
<script type="text/javascript" src="/js/jQuery.js" />
</head>
```

Разметка текста в HTML

текст <TT>пример использования 'TT' тэга</TT>
текст <I>пример использования 'I' тэга</I>
текст <B>пример использования 'B' тэга</B>
текст <BIG>пример использования 'BIG' тэга</BIG>
текст <SMALL>пример использования 'SMALL' тэга</SMALL>
текст <EM>пример использования 'EM' тэга</EM>
текст <STRONG>пример использования 'STRONG' тэга</STRONG>
текст <DFN>пример использования 'DFN' тэга</DFN>
текст <CODE>пример использования 'CODE' тэга</CODE>
текст <SAMP>пример использования 'SAMP' тэга</SAMP>
текст <KBD>пример использования 'KBD' тэга</KBD>
текст <VAR>пример использования 'VAR' тэга</VAR>
текст <CITE>пример использования 'CITE' тэга</CITE>
текст <ABBR>пример использования 'ABBR' тэга</ABBR>
текст <ACRONYM>пример использования 'ACRONYM' </ACRONYM>
текст _{пример использования 'SUB' тэга}
текст ^{пример использования 'SUP' тэга}

Результат разметки текста

текст пример использования 'TT' тэга
текст *пример использования 'I' тэга*
текст **пример использования 'B' тэга**
текст **пример использования 'BIG' тэга**
текст пример использования 'SMALL' тэга
текст *пример использования 'EM' тэга*
текст **пример использования 'STRONG' тэга**
текст *пример использования 'DFN' тэга*
текст пример использования 'CODE' тэга
текст пример использования 'SAMP' тэга
текст пример использования 'KBD' тэга
текст *пример использования 'VAR' тэга*
текст *пример использования 'CITE' тэга*
текст пример использования 'ABBR' тэга
текст пример использования 'ACRONYM' тэга
текст пример использования 'SUB' тэга
текст пример использования 'SUP' тэга

Маркированный список HTML

```
<UL>
<LI>Первый элемент списка</LI>
<LI>Второй элемент списка</LI>
<LI>и т.д.</LI>
</UL>
```

Нумерованный список HTML

```
<OL>
<LI>Первый элемент списка</LI>
<LI>Второй элемент списка</LI>
<LI>и т.д.</LI>
</OL>
```

Таблицы HTML

```
<TABLE BORDER='1'>
<TR><TH>ЗАГ1</TH><TH>ЗАГ2</TH></TR>
<TR><TD>Пол1.1</TD><TD>Пол1.2</TD></TR>
<TR><TD COLSPAN='2'>Пол2.1-2</TD></TR>
<TR><TD ROWSPAN='2'>Пол3-4.1</TD>
<TD>Пол3.2</TD></TR>
<TR><TD>Пол4.2</TD></TR>
</TABLE>
```

Таблицы HTML отображение

```
<TABLE BORDER='1'>
<TR><TH>ЗАГ1</TH><TH>ЗАГ2</TH></TR>
<TR><TD>Пол1.1</TD><TD>Пол1.2</TD></TR>
<TR><TD COLSPAN='2'>Пол2.1-2</TD></TR>
<TR><TD ROWSPAN='2'>Пол3-4.1</TD>
<TD>Пол3.2</TD></TR>
<TR><TD>Пол4.2</TD></TR
</TABLE>
```

ЗАГ1	ЗАГ2
Пол1.1	Пол1.2
Пол2.1-2	
Пол3-4.1	Пол3.2
	Пол4.2

Формы HTML

<**form** method="post" action="url обработчика">
Кому:

<**input** type="**checkbox**" value="mama@gmail.com" name="to[0]">маме

<input type="checkbox" value="wife@yandex.ru" name="to[1]">жене

От:<**select** name="from">
<option>kdb@perm.ru</option>
<option>kdenisb@mail.ru</option>
</select>
Тема: <input type="**text**" name="subject">

Тип:

<input type="**radio**" value="html" name="tip">html

<input type="radio" value="txt" name="tip">текст

<**textarea** name="mesbody">Привет, !</textarea>

<input type="**submit**" value="Послать">

Формы HTML

Кому:
- маме
- жене

От: kdb@perm.ru
- kdb@perm.ru
- kdenisb@mail.ru

Тем
Тип
- html
- текст

Привет, !

Послать

```
<br><input type="checkbox"
value="mama@gmail.com" name="to[0]">маме
<br><input type="checkbox"
value="wife@yandex.ru" name="to[1]">жене
<br>От:<select name="from">
<option>kdb@perm.ru</option>
<option>kdenisb@mail.ru</option>
</select><br>Тема: <input type="text"
name="subject">
<br>Тип:
<br><input type="radio" value="html"
name="tip">html
<br><input type="radio" value="txt"
name="tip">текст
<br><textarea name="mesbody">Привет, !
</textarea>
<br><input type="submit" value="Послать">
```

div и span

Используются для дальнейшего задания стилей (с помощью css) включенным в них элементам.

тескт
текст внутри div
текст текст текст внутри span текст текст

```
тескт <div>текст внутри
div</div> текст текст
<span>текст внутри
span</span> текст текст
```

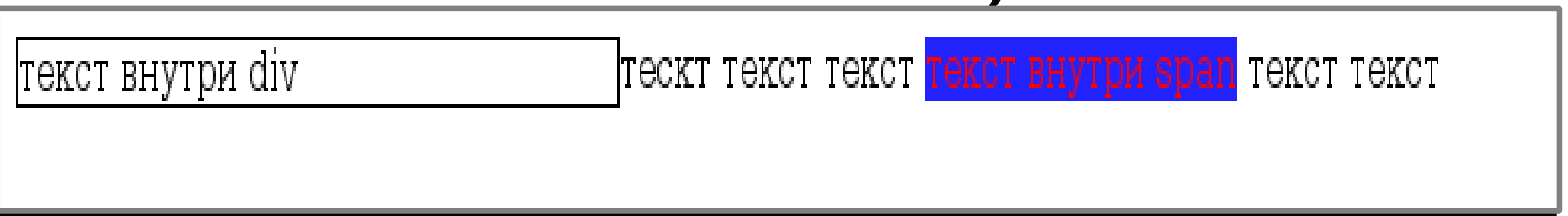
CSS

Стили (CSS, Cascading Style Sheets, каскадные таблицы стилей) представляют собой набор параметров, управляющих видом и положением элементов веб-страницы.

Стандарты:
CSS1 — 1996 г.
CSS2 — 1998 г.
CSS2.1 — 2011 г. (IE8, Gecko, Webkit, Presto)
CSS3 — ?
CSS4 — ?

http://htmlbook.ru/books http://www.w3.org/TR/REC-CSS1/

Пример css (размещение в заголовке)




```
<head><style>
span { color: red;
 background-color: #22f; }
div { border: 1px solid;
    width: 300px;
    float: left; }
</style><body>
тескт <div>текст внутри div</div> текст текст
<span>текст внутри span</span> текст текст
<div>текст внутри div</div> текст текст
<span>текст внутри span</span> текст текст
```

Размещение CSS в тэге

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.

```
<p style="text-align: justify;text-indent: 3em;">
Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum.
```

Размещение css в файле

```
<head>
<meta http-equiv="Content-Type" content="text/html;
charset=utf-8" />
<link rel="stylesheet" href="style1.css">
<body>
тескт <div>текст внутри div</div> текст текст
<span>текст внутри span</span> текст
текст
```

style1.css

```
div {
color: blue;
}
```

Приоритет css

1.!important
2.точность селектора: идентификатор, класс, тэг
3.место размещения стиля: тэг, заголовок, файл

```
<STYLE TYPE="text/css">
  #x97z { color: blue }
</STYLE>
<P ID=x97z STYLE="color: red">
```

Типы селекторов CSS

Для стилей вне тэгов (заголовок, файл)

```
селектор {
Описание стилей
}
```

- Имя тэга
- Название класса
- Идентификатор
- Комбинации

Примеры селекторов

```
span {
color: blue;
}
span.yy {
font-weight: bold;
}
#xx {
font-size: 30px;
}
span span {
font-size: 8px;
}
```

```
тескт текст текст <span id="xx"
class="yy">текст внутри span1</span>
текст
<span class="yy">текст<span>текст
внутри span2.1</span> внутри span2
</span>
```

Примеры селекторов

```
span {
color: blue;
}
span.yy {
font-weight: bold;
}
#xx {
font-size: 30px;
}
span span {
font-size: 8px;
}
```

```
тескт текст текст <span id="xx"
class="yy">текст внутри span1</span>
текст
<span class="yy">текст<span>текст
внутри span2.1</span> внутри span2
</span>
```

тескт текст текст **текст внутри span1** текст текст текст внутри span2.1 внутри span2

Стили

- Текст
- Цвет
- Свойства box-модели
- Специфичные свойства тэгов

Стили текста CSS

font-family — имя фонта
font-style — с наклоном
font-variant — small-caps
font-weight — жирный
font-size — размер
font — все вместе
word-spacing, letter-spacing, text-decoration
vertical-align, text-transform, text-align, text-indent
line-height

```
P { font: bold italic 150% serif }
```

Единицы измерения css

em Размер шрифта текущего элемента
ex Высота символа x
px Пиксел
% Процент

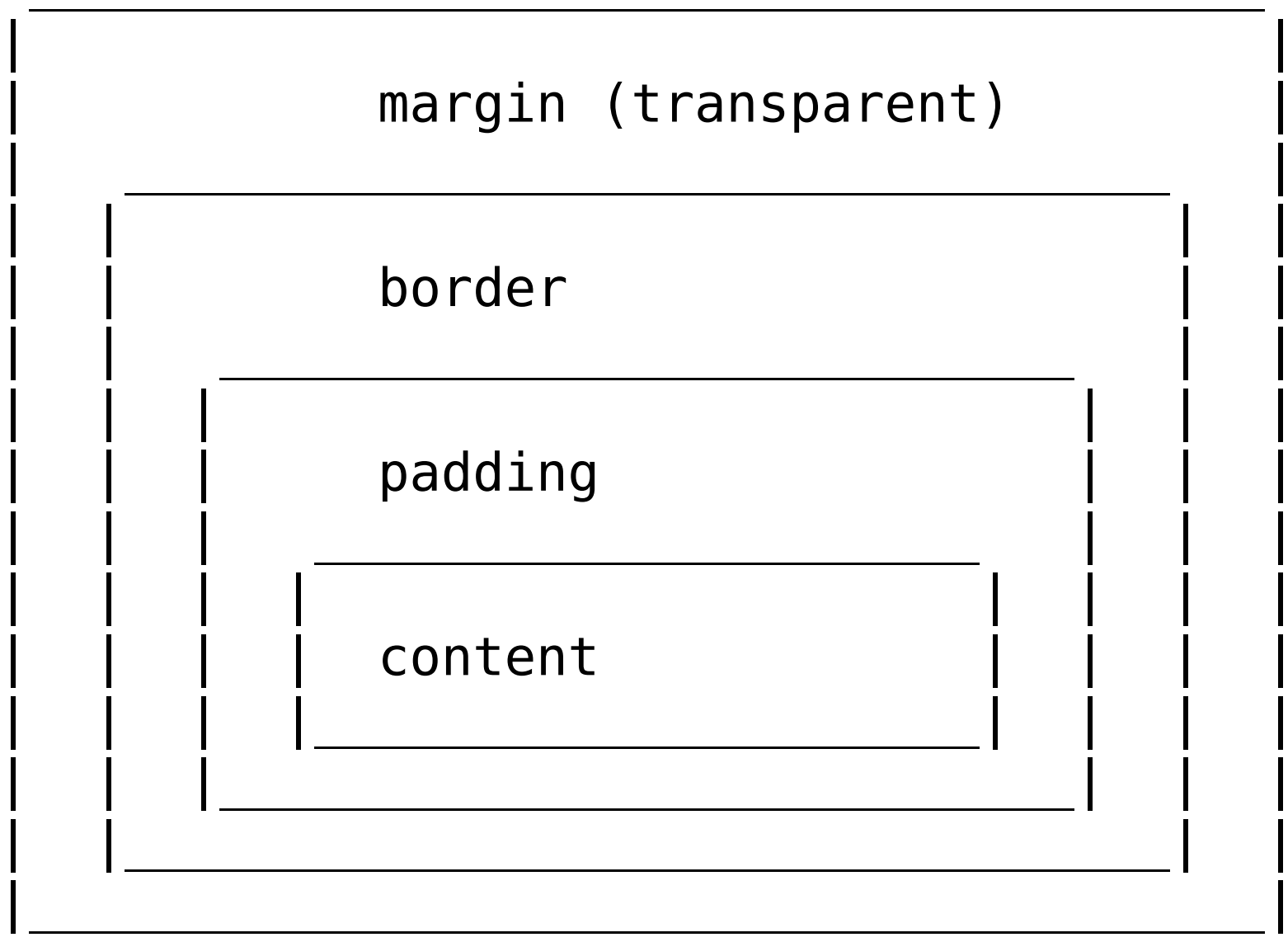
in Дюйм (1 дюйм равен 2,54 см)
cm Сантиметр
mm Миллиметр
pt Пункт (1 пункт равен 1/72 дюйма)
pc Пика (1 пика равна 12 пунктам)

Цвета CSS

color
background-color

Название — red, blue...
Номером hex — #FF4458
Номером dec — rgb(255, 0, 0)
Процентами — rgb(100%, 20%, 0%)

Box model



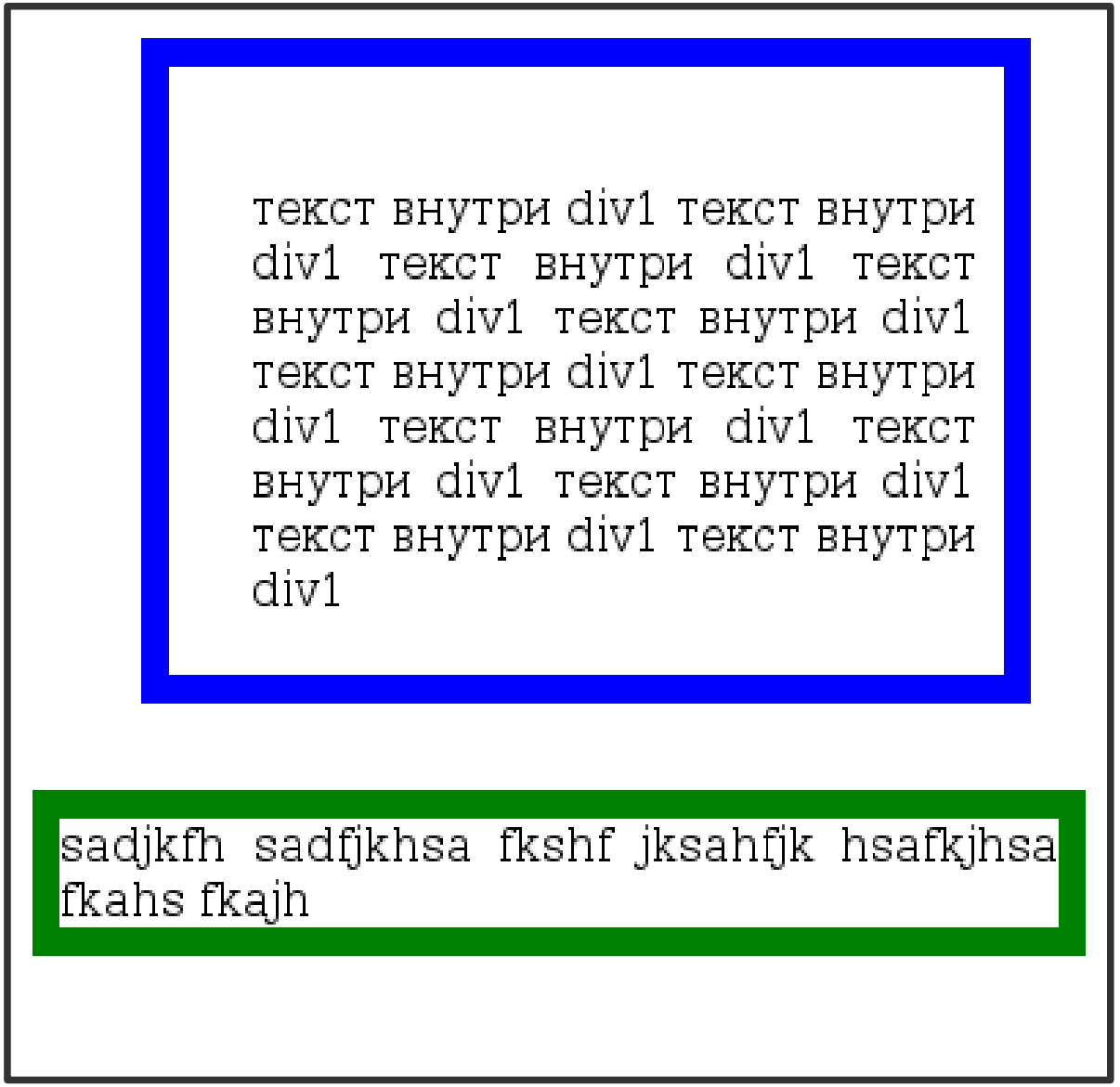
Пример box-модели

```
div {
border: solid 10px;
border-color: green;
margin: 0px 0px
0px 0px;
text-align: justify;}
div.xx {
margin: 10px 20px
30px 40px;
border-color: blue;
padding: 40px 10px
20px 30px;}
```

```
<body style="padding:0px;">
<div class="xx">текст внутри
div1 текст внутри div1 текст
внутри div1 текст внутри div1
текст внутри div1
текст внутри div1 текст внутри
div1 текст внутри div1 текст
внутри div1
текст внутри div1 текст внутри
div1 текст внутри div1 </div>
<div>sadjkfh sadfjkhsa fkshf
jksahfjk hsafkjhsa fkahs
fkajh</div>
```

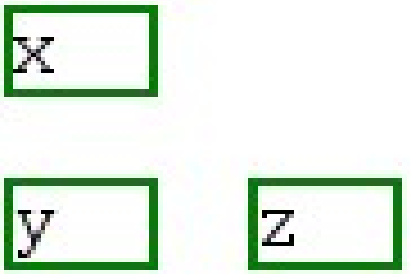
Пример box-модели

```
div {
border: solid 10px;
border-color: green;
margin: 0px 0px
0px 0px;
text-align: justify;}
div.xx {
margin: 10px 20px
30px 40px;
border-color: blue;
padding: 40px 10px
20px 30px;}
```



Позиционирование

```
<style>
div {
border: solid 2px;
border-color: green;
margin: 10px 10px 10px 10px;
width:30px;
}
#x {
/* float: left;
*/
}
#y {
float: left;
}
#z {
float: left;
}
</style>
<body><div id="x">x</div><div id="y">y</div>
<div id="z">z</div>
```



Javascript

Краткое введение в *Javascript*

Javascript это:

1. Интерпретируемый язык. Его интерпретатор обычно встроен в браузер.
2. Основное назначение – определять «динамическое» поведение страниц при загрузке (формирование страницы перед ее открытием) и при работе пользователя со страницей (UI элементы).
3. Текст на *Javascript* может быть вложен в HTML-страницу непосредственно или находиться в отдельном файле (как CSS).
4. Похож на языки *Java* и *C#* синтаксически, но сильно отличается от них по внутреннему содержанию.

Характеристика *Javascript*

Некоторые важнейшие характеристики *Javascript* :

1. Язык объектно-ориентированного программирования. Объекты в языке имеют «тип», «атрибуты» и «методы»

```
"John,Jane,Paul,Michael".split(",").length
```

1. Переменные не имеют заранее заданного типа, то есть в разные моменты времени могут содержать значения разных типов

```
var number = 25;  number = (number < 0);  number = "25";
```

1. Типы объектов могут быть: `number, string, function, object, undefined`. Оператор `typeof` позволяет «вычислить» тип объекта.

```
typeof 25 == "number"   typeof null == "object"
```

Характеристика *Javascript*

Некоторые важнейшие характеристики *Javascript* :

1. Язык объектно-ориентированного программирования. Объекты в языке имеют «тип», «атрибуты» и «методы»

```
"John,Jane,Paul,Michael".split(",").length
```

1. Переменные не имеют заранее заданного типа, то есть в разные моменты времени могут содержать значения разных типов

```
var number = 25;  number = (number < 0);
```

- `number = "25";`

1. Типы объектов могут быть: `number, string, function, object, undefined`. Оператор `typeof` позволяет «вычислить» тип объекта.

```
typeof 25 == "number"   typeof null == "object"
```

Основные встроенные типы

Есть набор встроенных «классов», порождающих «объекты», различающиеся набором атрибутов и методов. Программисты могут динамически изменять поведение этих «классов» и создавать свои собственные. Каждый «класс» является объектом, у которого есть «прототип», определяющий набор атрибутов и методов у всех вновь создаваемых объектов этого класса.

Типы, встроенные в язык, это:

- `Number` : 64-х-разрядные числа с плавающей точкой.
- `String` : строки с символами в формате Unicode.
- `Array` : массивы с переменными границами.
- `Function` : Функции. Каждая функция, кроме того, может служить конструктором объекта.
- `Boolean, Date, Math...` : логические значения, даты,…

Некоторые сведения о синтаксисе

Описание переменных:

```
var count = 25,
    msg = 'Сообщение об ошибке';
var nullVar;  // получает начальное значение null
```

Операции такие же, как в Java и C#, но более широко используется преобразование типов

```
+   -   *   /   %   ++  --   =   +=   -=   *=
/=  %=  ==  !=  >   <   >=   <=  &&   ||   !
```

```
  2 + '3' == '23', но     2 + 3 == 5
```

Многие операторы очень похожи на соответствующие операторы Java и C#, но могут иметь некоторые отличия в семантике.

```
for (var i = 0; i < 100; ++i) { ... }

if (x * y < 100) { ... } else { ... }

try { ... } catch (e) { ... } finally { ... }
```

Объекты, встроенные в браузеры

При программировании можно использовать ряд встроенных объектов. Основные из них это:

- `window` : представляет «глобальный контекст» и позволяет работать с атрибутами и методами окна.
- `document` : загруженная страница со своей структурой элементов.
- `navigator` : объект, представляющий браузер и его свойства.
- `location` : характеристики текущего URL (порт, хост и т.п.).
- объекты, представляющие элементы различных типов в HTML-странице, такие как `<body>`, `<link>`, `<img>... и их стили`
- события (events), возникающие от действий пользователя, например, нажатие кнопки мыши (`click`), загрузка новой страницы (`load`) и т.д.

Включение Javascript в HTML-страницу

Фрагменты кода можно включать в заголовок или тело HTML-документа. Кроме того, можно разместить код в отдельном файле, а в HTML-странице разместить ссылку на этот файл.

```
<html>
 <head>
  <script type="text/javascript"> ... </script>
  <script type="text/javascript" src="scripts/myscript1.js"/>
 <head>
 <body>
  <script type="text/javascript"> ... </script>
  <script type="text/javascript" src="scripts/myscript2.js"/>
 </body>
</html>
```

Два простых примера

Метод `document.write` используется для непосредственного включения HTML-текста в содержимое страницы, например, можно сгенерировать длинный текст в параграфе:

```
<body>
  <p>
     <script type="text/javascript">
       for (var i = 0; i < 100; ++i) {
         document.write("Hello, world! ");
       }
     </script>
  </p>
</body>
```

Выявите разницу между исходным кодом страницы и анализом элементов

Два простых примера (продолжение)

Во втором примере датчик случайных чисел используется для генерации случайной ссылки (из заданного набора):

```
<script type="text/javascript">
   var rand = Math.random();// в диапазоне: [0, 1)
    var numb = Math.floor(rand * 10);
    var image = "images/image" + numb + ".jpg";
    var insert = "<img class=\"floatRight\"
     src=\"" + image + "\" alt=\"Фотографии\"/>";
    document.write(insert);
</script>
```

Тип String

Строки заключаются либо в апострофы, либо в двойные кавычки

```
var slogan = "Don't be evil!";
var image ='<img src="myphoto.jpg"/>';
```

Экранирование и последовательности: `\\` `\'` `\"` `\t` `\n`

Операции над строками: `+` `<` `>` `==` `!=`

```
"2" + "3"   "23"          "a" == "A"  false
"10" < "5"  true           5 == "5"   true
10 < "5"    false          5 === "5"  false
5 + "5"     "55"
```

Атрибут строки: `length` – длина строки.

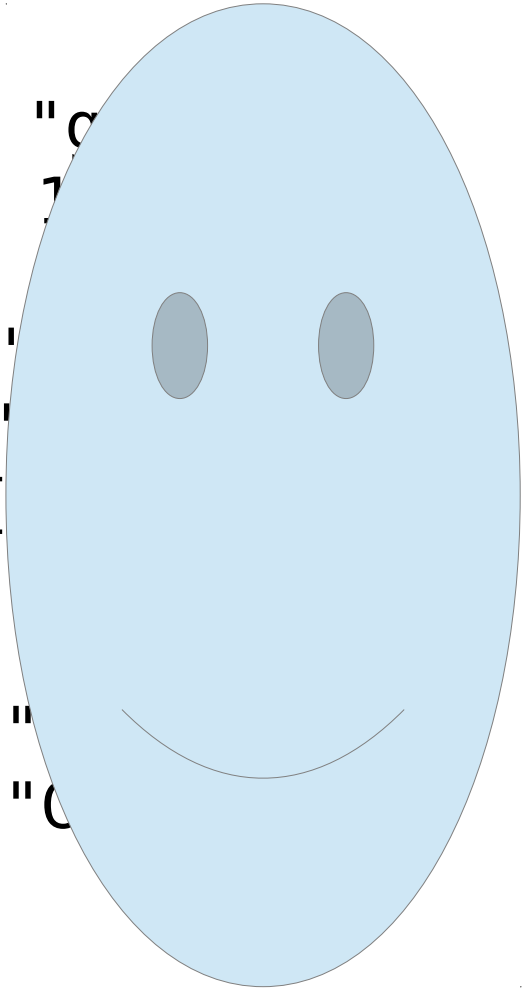
```
"abc".length == 3
```

Преобразования типов: `String(n)` `Number(s)`

```
String(10) < "5" == true
                    Number('3.' + '14') == 3.14
```

Стандартные методы объектов типа `String`

```
"Google".charAt(3)                "g
"Google".indexOf("o")              1
"Google".lastIndexOf("o")
"Google".replace("o", "oo")       "
"Google".replace(/o/g, "oo")      "
"Google".split("o")               [
"Google".substr(1,3)
"Google".substring(1,3)
"Google".toLowerCase()            "
"Google".toUpperCase()            "G
```



Тип Number

Числа – это 64-х-разрядные двоичные числа с плавающей точкой.

```
Number.MIN_VALUE              5e-324
Number.MAX_VALUE              1.7976931348623157e+308
Number.NaN                    NaN
Number.POSITIVE_INFINITY      Infinity
Number.NEGATIVE_INFINITY      -Infinity
```

Операции над числами: `+ - * / % < > == !=`

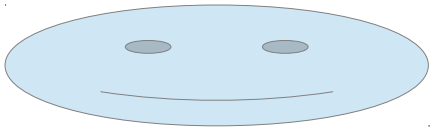
```
3.14 % 2                    1.14
```

Функции преобразования: `parseInt, parseFloat, Number, toString`

```
parseInt("3.14")     3
parseFloat("*3.14")  NaN
Number("3.xaxa")     NaN
3.14.toString()      "3.14"
isNaN(3.14 / 0)      false
isNaN(0 / 0)         true
```

Тип Boolean

Стандартные логические значения – `true` и `false`. Однако в качестве условий можно использовать любое значение.



```
if (2 < 5)                          if ("")
if (25)                             if (0)
if ('Google могуч и ужасен')        if (null)
```

Логические условия используются в условных операторах и операторах циклов.

```
if (x < y) { z = x; } else { z = y; }

while (x < 100) { x = x * 2; n++; }

do { x = Math.floor(x / 2); n++; } while (x > 0);

for (var y = 0, x = 0; x < 100; ++x) { y += x; }
```

Тип Date

Объекты типа Date содержат дату в виде числа миллисекунд, прошедших с 1 января 1970 г. Диапазон от -10^8 до 10^8 дней от 1 января 1970 г.

Конструкторы:

```
var now = new Date();                    // сейчас
var january1st1970 = new Date(0);  // в миллисек
var gagarin = new Date(1961, 3, 12);
var newYear = new Date("January 1, 2015");
```

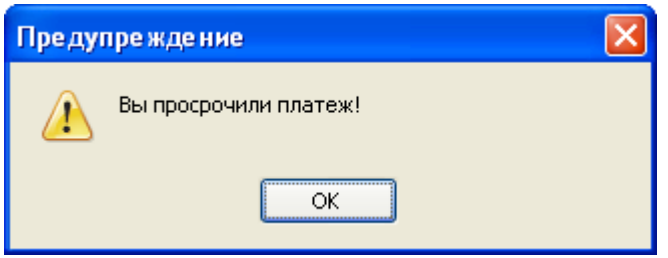
Методы, применимые для работы с датами: getDate, getMonth, getFullYear, getTime, getTimezoneOffset, setDate, setFullYear,...

```
function DaysToDate(day, month) {
  var now = new Date(), year = now.getFullYear();
  var bd = new Date(year, month-1, day);
  var fullDay = 24 * 60 * 60 * 1000;
  var diff = Math.ceil((bd - now) / fullDay);
  return diff < 0 ? diff + 365 : diff;
}
```

Сообщения, выдаваемые в popup-окнах

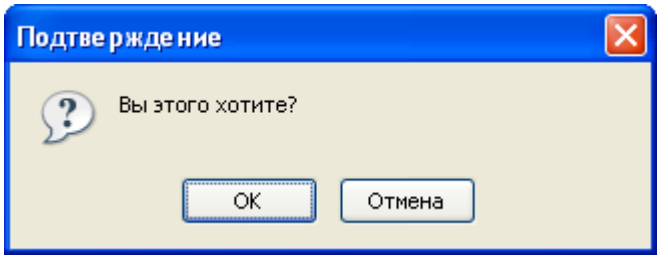
Три стандартные функции используются для генерации сообщений в popup-окнах: `alert`, `confirm`, `prompt`.

```
alert('Вы просрочили платеж!');
```



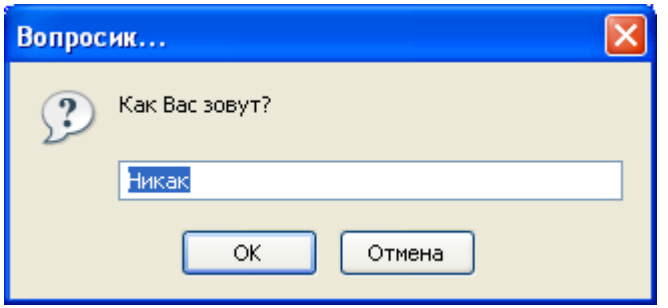

```
confirm('Вы этого хотите?');
```

Выдает **true** или **false**




```
var name = prompt('Как Вас зовут?',
    'Никак', 'Вопросик...');
```

Выдает **введенную строку** или **null**



События и реакции на них

Имеется большое количество событий, которые можно разделить на следующие классы:

- события от мыши (`click`, `dblclick`, `mousedown`,...);
- события от клавиатуры (`keypress`, `keydown`,...);
- события от элементов ввода (`focus`, `submit`, `select`,...);
- события страницы (`load`, `unload`, `error`,...);

```
<p>День независимости России
   <span onclick=
          "alert('Осталось ' +
          DaysToDate(12, 6) + 'дней');">
   12 июня</span>.
</p>
```

Тип Array

Существует несколько способов создания массива:

```
var holidays = ["1 января", "7 января", "23 февраля"];
var holidays = new Array("1 января", "7 января", "23 февраля");
var holidays = new Array(3);
holidays[0] = "1 января";
holidays[1] = "7 января";
holidays[2] = "23 февраля";
```

Атрибут массива: `length` – длина массива.

```
var myArray = new Array();
myArray[2] = new Date(2008,2,23);
myArray[5] = new Date(2008,5,9);
myArray.length ==
```

Тип Array (продолжение)

Методы, определенные для работы с массивом:
`concat, join, pop, push, shift , unshift, slice`

```
var names = ["Петя", "Вася"];
 names = names.concat(["Сережа", "Наташа"],
                      ["Оля", "Люба"]);
```

- `names == ["Петя", "Вася", "Сережа", "Наташа", "Оля", "Люба"]`

```
var s = names.join(';');
  s == "Петя;Вася;Сережа;Наташа;Оля;Люба"
var e = names.pop();  e == "Люба"
names == ["Петя", "Вася", "Сережа", "Наташа", "Оля"]
var l = names.push("Саша");  l == 6
```

`names == ["Петя", "Вася", "Сережа", "Наташа", "Оля", "Саша"]`

shift и unshift – точно так же, как pop и push, но с началом массива.

```
names = names.slice(1, 4);
names == ["Вася", "Сережа", "Наташа", "Оля"]
```

Тип Array (продолжение)

```
var names = ["Вася", "Сережа", "Наташа", "Оля"];
names.reverse();
names == ["Оля", "Наташа", "Сережа", "Вася"]
names.sort();
names == ["Вася", "Наташа", "Оля", "Сережа"]
var a = [5, 3, 40, 1, 10, 100].sort();
a == [1, 10, 100, 3, 40, 5]
var a = [5, 3, 40, 1, 10, 100].sort(function(a,b)
{return a-b;});
```

toString – точно так же, как join(',').

```
names.toString() == "Вася,Саша,Таня,Нина,Сережа"
```

Работа с таймером

Можно создать таймер и определить реакцию на событие от таймера.

```
var timer = setTimeout(func, timeinterval);
```

`func` – это функция или строка с кодом; `timeinterval` – время в миллисекундах. Таймер срабатывает один раз и запускает функцию.

```
function launchTimer() {
  setTimeout("alert('Амкар – чемпион!');", 2000);
}
```

Теперь можно запустить этот таймер, например, по событию `click`:

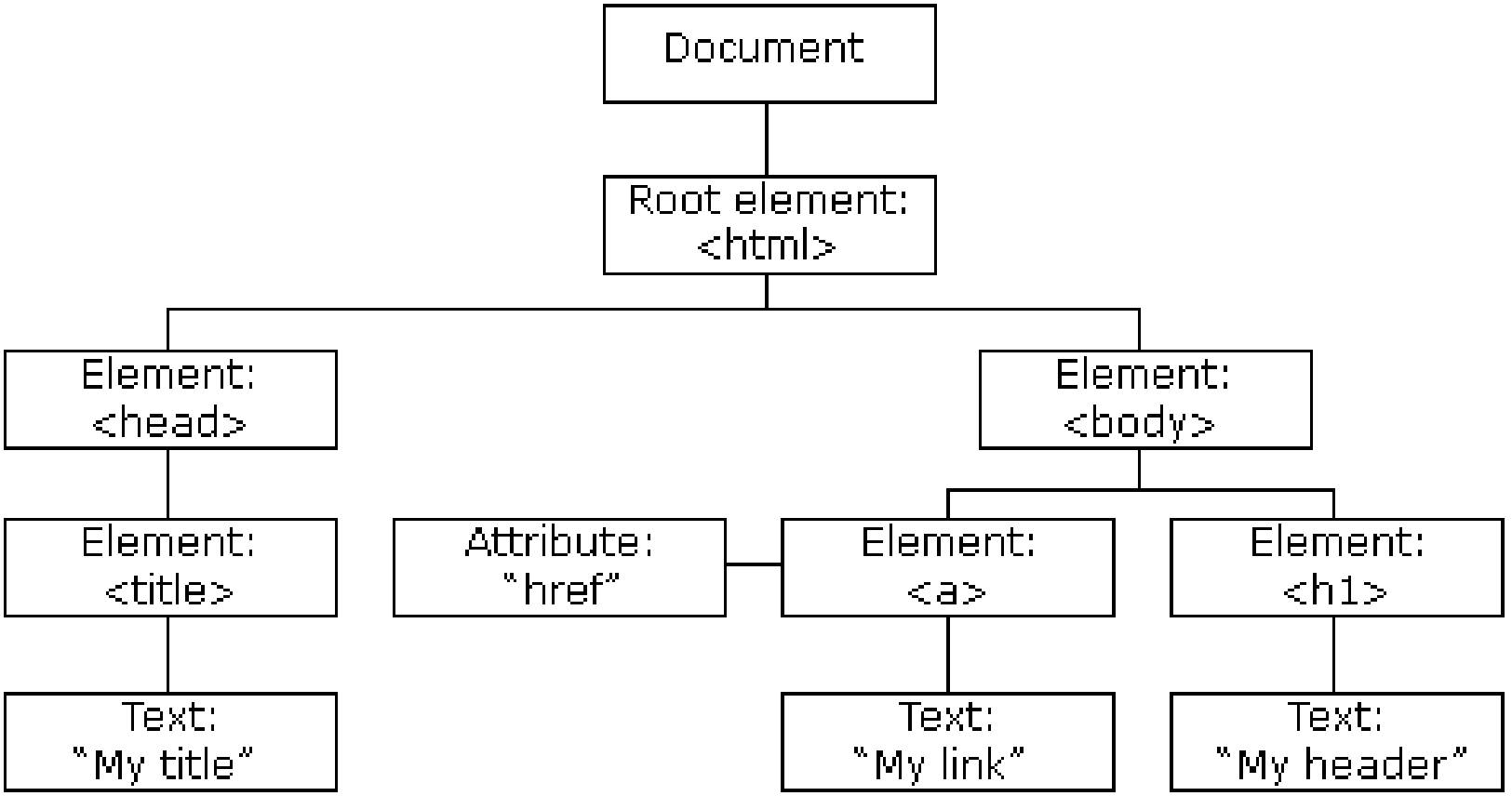
```
<body>
  <p>Нажми <span onclick="launchTimer();">сюда!</span></p>
</body>
```

Пока событие еще не случилось, таймер можно остановить:

```
var timer = setTimeout(func, timeinterval);
clearTimeout(timer);
```

DOM

When a web page is loaded, the browser creates a **D**ocument **O**bject **M**odel of the page.



Обращение к элементам DOM

document.getElementById
document.getElementsByClassName
document.getElementsByTagName

document.getElementById('xxx').innerHTML=
'текст внутри тэга с идентификатором xxx'

Создание элементов DOM

document.createElement
document.createAttribute
document.createTextNode
element.appendChild
element.setAttribute

```
var btn = document.createElement("i");
var t = document.createTextNode("bams");
btn.appendChild(t);
```

```
document.getElementsByTagName("INPUT")[0].setAttribute("type","button");
```

jQuery

JavaScript-фреймворки

Сферы использования
• обработки данных на стороне клиента
• создания визуальных эффектов
• «обогащения» интерфейса
• создания клиентской части web-приложений

Популярные JavaScript-фреймворки

- prototype
- jQuery
- dojo

Возможности jQuery

- переход по дереву DOM, включая поддержку XPath как плагина,
- события,
- визуальные эффекты,
- AJAX-дополнения,
- JavaScript-плагины

Подключение jQuery

- jQuery включается в веб-страницу как один внешний JavaScript-файл:

```
<script type="text/javascript"
   src="путь/к/jQuery.js"></script>
```

- Есть постоянно действующая и обновляемая ссылка `http://ajax.googleapis.com/ajax/libs/jquery/1.11.1/jquery.min.js`

Вызовы jQuery

$("div.test").add("p.quote").addClass("blue").slideDown("slow");

Выбирает все элементы с классом test, а также все элементы <p> с классом quote, и затем добавляет им всем класс blue и визуально плавно спускает вниз.

jq0_blue_js.html jq0_blue.html

Селекторы

- Вызов $(selector) или jQuery(selector) возвращает специальный объект, содержащий массив элементов DOM.
- Селекторы в jQuery базируются на CSS селекторах, а так же поддерживают XPath.

Примеры селекторов

$('#sidebar') – выбор элемента с id = sidebar

$('.post') – выбор элементов с class = post

$('div#sidebar') – выбор элемента div с id = sidebar

$('div.post') – выбор элементов div с class = post

$('div span') – выбор всех span элементов в элементах div

Примеры селекторов

$('div < span') – выбор всех span элементов в элементах div, где span является прямым потомком div

$('div, span') – выбор всех div и span элементов

$('span + img') – выбор всех img элементов перед которыми идут span элементы

$('span ~ img') – выбор всех img элементов после первого элемента span

Примеры селекторов

$('#banner').prev() – выбор предыдущего элемента от найденного

$('#banner').next() – выбор следующего элемента от найденного

$('*') – выбор всех элементов

$('p < *') – выбор всех потомков элементов p

Примеры селекторов

$('#banner').prev() – выбор предыдущего элемента от найденного

$('#banner').next() – выбор следующего элемента от найденного

$('*') – выбор всех элементов

$('p < *') – выбор всех потомков элементов p

Селекторы с фильтрами

$('div:first') – выбираем первый div в DOMe

$('div:not(.red)') – выбираем div'ы у которых нет класса red

$('div:eq(N)') – выбираем div, идущий под номером N в DOMe

$(':header') – выбор заголовков h1, h2, h3 и т.д.

$('div:hidden') – выбираем скрытые div

Селекторы с фильтрами

$('div.red').filter('.bold') – выбираем div'ы которые содержат класс red и класс bold

$("div[id]") – выбор всех div с атрибутом id

$("div[title='my']") – выбор всех div с атрибутом title=my

$("div[title*='my']") – выбор всех div с атрибутом title содержащим my

Селекторы для форм

$(":text") – выбор всех input элементов с типом =text

$("input:enabled") – выбор всех включенных элементов input

$("input:checked") – выбор всех отмеченных чекбоксов

$("div[name=city]:visible:has(p)") – выбор видимого div'а с именем city, который содержит тег p

Работа с DOM

- Создание элементов jq1.html

  $(" Hello! ");

- Вставка в DOM

$(" Привет!
").insertAfter("#followMe");

  - append, appendTo, prepend, prependTo
  - after, before, insertAfter, insertBefore
  - wrapAll, wrapInner

Создание элементов

```
$("<div/>",
  { id: "foo",
    css: { height: "50px", width: "50px",
       color: "blue", backgroundColor: "#ccc"
    },
    click: function() {
       $(this).css("backgroundColor", "red");
    }
}).appendTo("body");
```

Манипуляции с наборами

- Манипуляции:

add()

filter()

not()

eq(N)

first(), last()

…

- Пример:

$(‘p’).add(‘ Привет! ’)

События

Назначение обработчиков:

- click(fn)
- hover(fnIn, fnOut)
- change(fn)
- focus(fn), blur(fn)
- …

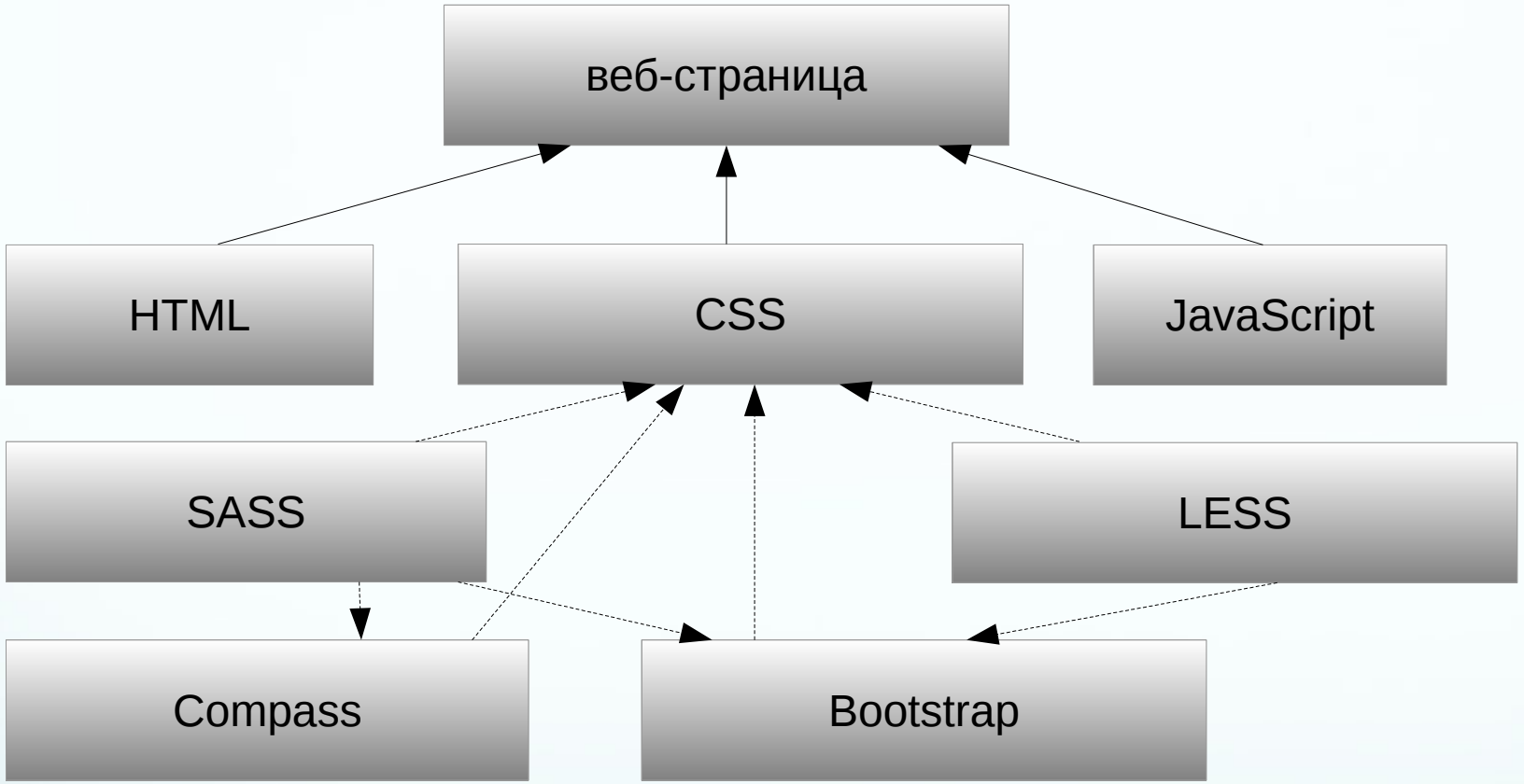
Генерация событий:

- click()

Примеры

- выдвижная панель jq2.html
- связанная анимация jq3.html

“Адаптивная” верстка



Препроцессор LESS

Twitter Bootstrap

- Свободнораспространяемый фремворк для веб-сайтов и веб-приложений
- Основывается на HTML5, CSS (LESS) и JavaScript (jQuery)

Авторы

Mark Otto

@mdo

Designer at Twitter, creator of Bootstrap (@TwBootstrap).

San Francisco Bay Area · http://t.co/HqYRIAl

jacob thornton

@fat

I work for twitter... I write comics sometimes (http://t.co/GucZD6o) and javascript (http://t.co/oFcXhQl) other times.

San Francisco, CA

Куча фич

Designed for everyone, everywhere.

Need reasons to love Bootstrap? Look no further.

Built for and by nerds

Like you, we love building awesome products on the web. We love it so much, we decided to help people just like us do it easier, better, and faster. Bootstrap is built for you.

For all skill levels

Bootstrap is designed to help people of all skill levels—designer or developer, huge nerd or early beginner. Use it as a complete kit or use to start something more complex.

Cross-everything

Originally built with only modern browsers in mind, Bootstrap has evolved to include support for all major browsers (even IE7!) and, with Bootstrap 2, tablets and smartphones, too.

12-column grid

Grid systems aren't everything, but having a durable and flexible one at the core of your work can make development much simpler. Use our built-in grid classes or roll your own.

Responsive design

With Bootstrap 2, we've gone fully responsive. Our components are scaled according to a range of resolutions and devices to provide a consistent experience, no matter what.

Styleguide docs

Unlike other front-end toolkits, Bootstrap was designed first and foremost as a styleguide to document not only our features, but best practices and living, coded examples.

Growing library

Despite being only 10kb (gzipped), Bootstrap is one of the most complete front-end toolkits out there with dozens of fully functional components ready to be put to use.

Custom jQuery plugins

What good is an awesome design component without easy-to-use, proper, and extensible interactions? With Bootstrap, you get custom-built jQuery plugins to bring your projects to life.

Built on LESS

Where vanilla CSS falters, LESS excels. Variables, nesting, operations, and mixins in LESS makes coding CSS faster and more efficient with minimal overhead.

HTML5

Built to support new HTML5 elements and syntax.

CSS3

Progressively enhanced components for ultimate style.

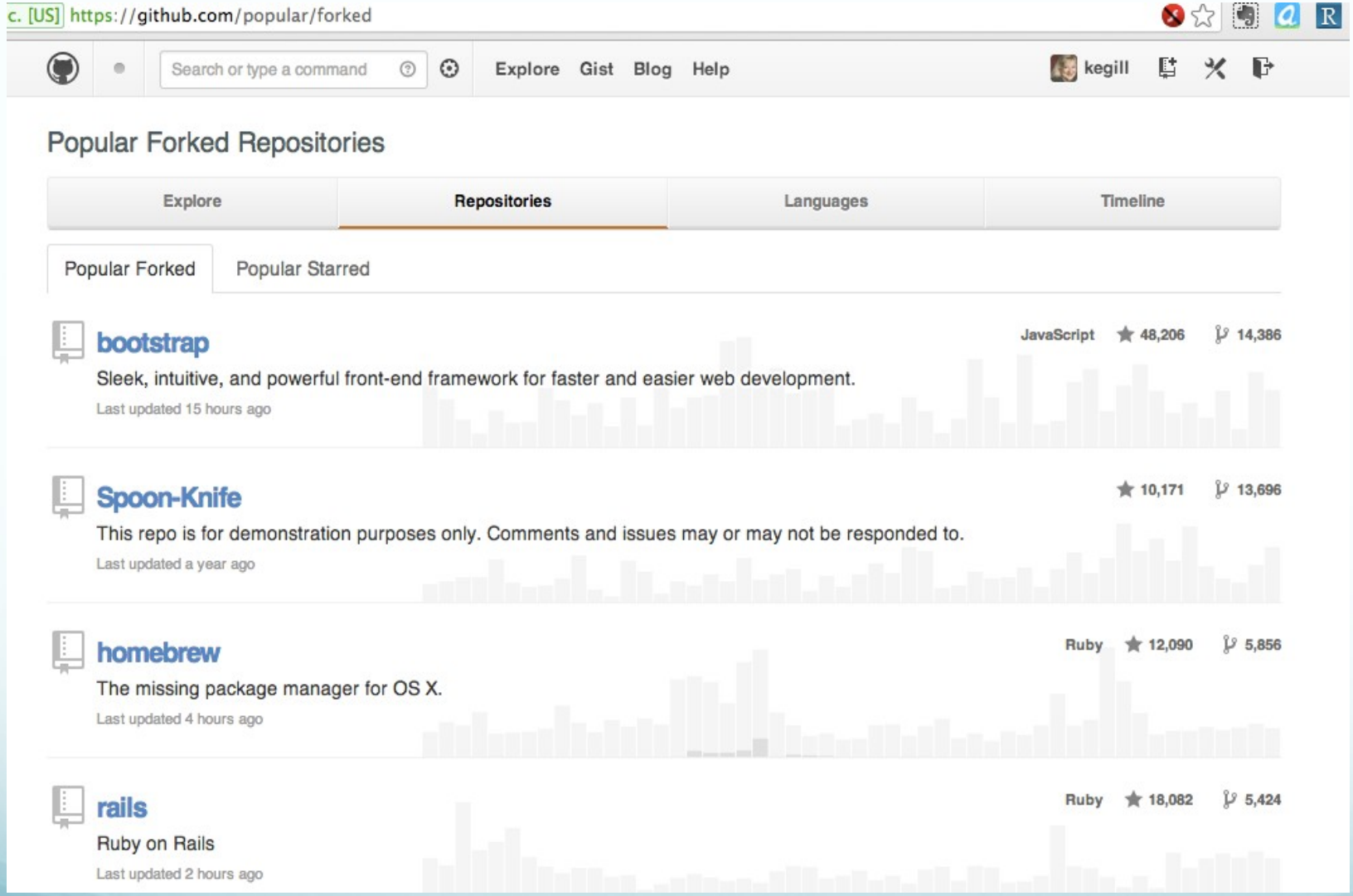
Open-source

Built for and maintained by the community via GitHub.

Made at Twitter

Brought to you by an experienced engineer and designer.

Популярность

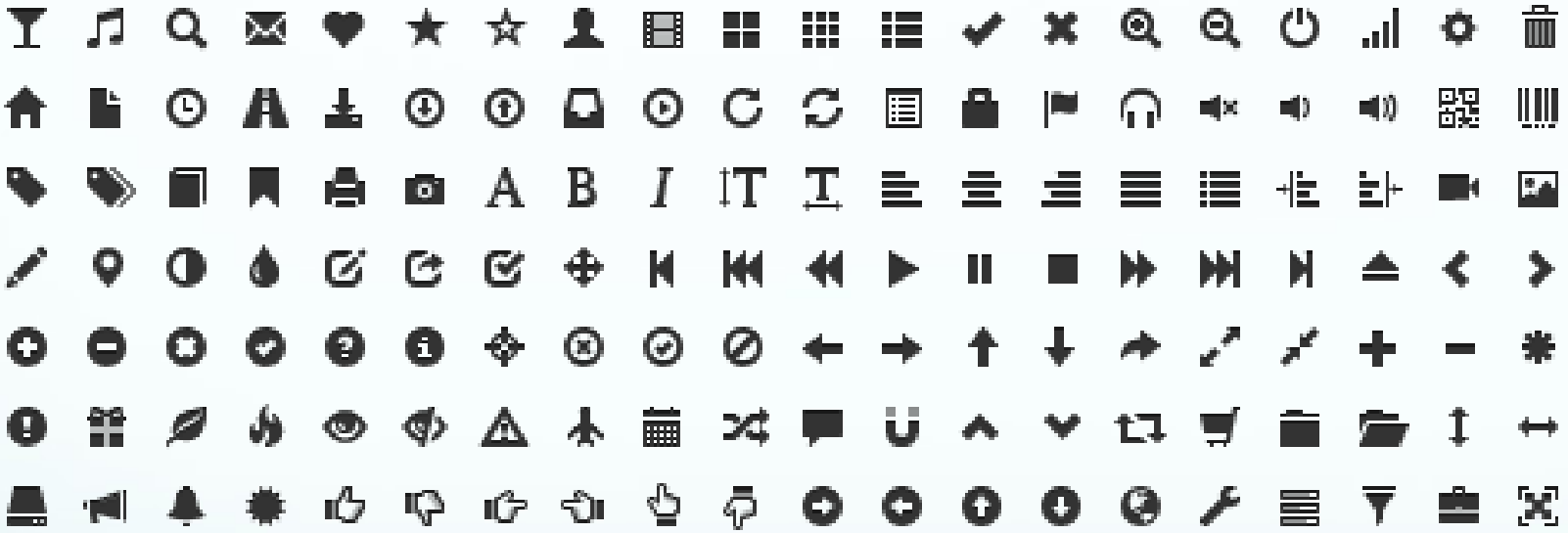


Поддержка брауз.

MSIE, Opera



Набор иконок



Сетка

Extra small devices ~ Phones (< 768px)
Small devices ~ Tablets (>= 768px)
Medium devices ~ Desktops (>= 992px)
Large devices ~ Desktops (>= 1200px)

col-xs- ~ Extra small devices
col-sm- ~ Small devices
col-md- ~ Medium devices
col-lg- ~ Large devices

Компоненты

- Кнопки:

- Вкладки:

- Пути:

- Пагинатор:

- Оповещения:

- Progress bar: