

Internet- технологии

ЛЕКЦИЯ №7
ЯЗЫК НАПИСАНИЯ СЦЕНАРИЕВ JAVASCRIPT

Содержание

- Объект Window. Методы и свойства объекта window.
- Объектная модель документа
- Поиск и манипуляция элементами страницы.
- Библиотека jQuery.



Объектная модель документа



Структура HTML-документа отображается в иерархической структуре связанных объектов, соответствующих HTML тегам.



Объект Window

Window – это самый старший класс в иерархии объектов JavaScript. К нему относятся объект Window и объект Frame. Объект Window ассоциируется с окном программы-браузера, а объект **Frame** – с окнами внутри окна браузера, которые порождаются последним при использовании автором HTML-страниц контейнеров FRAMESET и FRAME.

Чаще всего используют следующие свойства и методы объектов типа Window:

Свойства:

status, location, history, navigator

Методы:

open()

close()

focus()

Свойства	Методы	События
• status	• open()	Событий нет
• location	• close()	
• history	• focus()	
• navigator		



Объект Window. Управление окнами

Окно можно открыть (создать), закрыть (удалить), поднять его поверх всех других открытых окон (передать фокус). Кроме того, можно управлять свойствами окна и свойствами подчиненных ему объектов.

Наиболее популярные методы управления окнами:

`alert();`

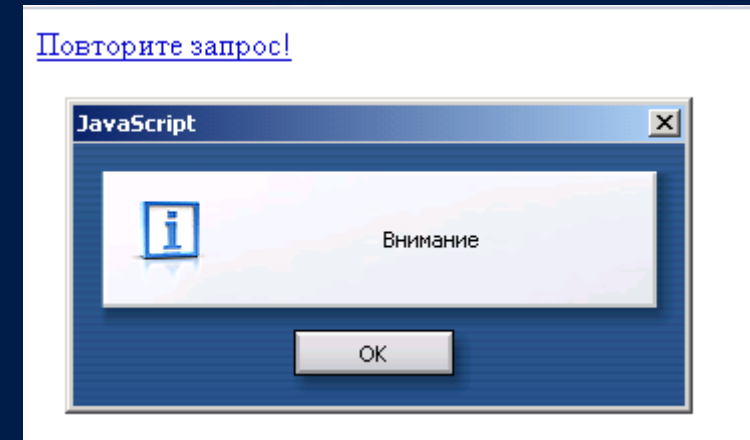
`confirm();`

`prompt();`

`open();`

`close();`

`focus();`



Управление окнами. *confirm()*

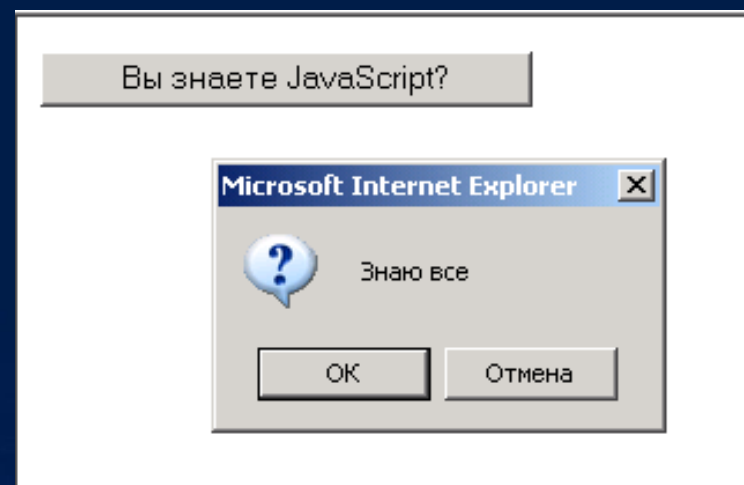
Метод `confirm()` позволяет задать пользователю вопрос, на который тот может ответить либо положительно, либо отрицательно:

```
<FORM>
```

```
<INPUT TYPE=button VALUE="Вы знаете JavaScript?"  
onClick = "if(window.confirm('Знаю все') == true) {  
document.forms[o].elements[o].value='Да'; } else {  
document.forms[o].elements[o].value='Нет'; };">
```

```
<BR>
```

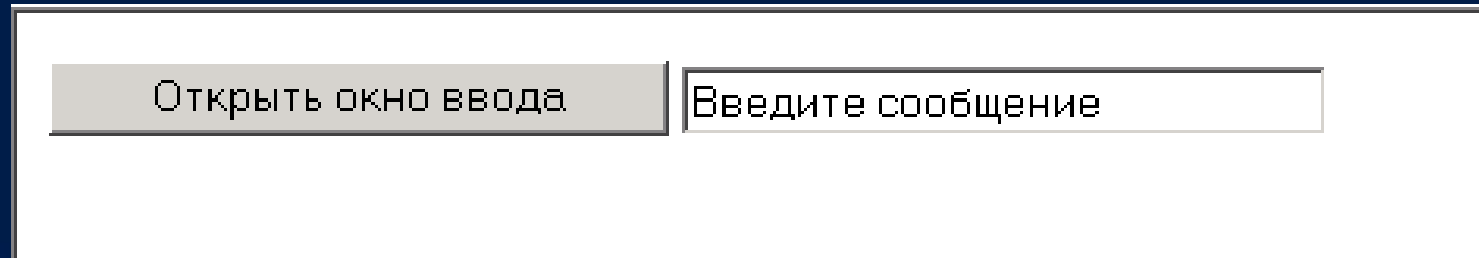
```
</FORM>
```



Управление окнами. *prompt()*

Метод `prompt()` позволяет принять от пользователя короткую строку текста, которая набирается в поле ввода информационного окна:

```
<FORM>  
<INPUT TYPE=button VALUE="Открыть окно ввода"  
onClick="document.forms[o].elements[1].value=window  
.prompt('Введите сообщение');">  
<INPUT SIZE=30>  
</FORM>
```



The screenshot shows a web form with a button labeled "Открыть окно ввода" and a text input field with the placeholder text "Введите сообщение".

Управление окнами. open()

Метод open() предназначен для создания новых окон. В общем случае его синтаксис выглядит следующим образом:

```
open("URL","window_name","param,param,...", replace);
```

где: URL – страница, которая будет загружена в новое окно, window_name – имя окна, которое можно использовать в атрибуте TARGET в контейнерах A и FORM.

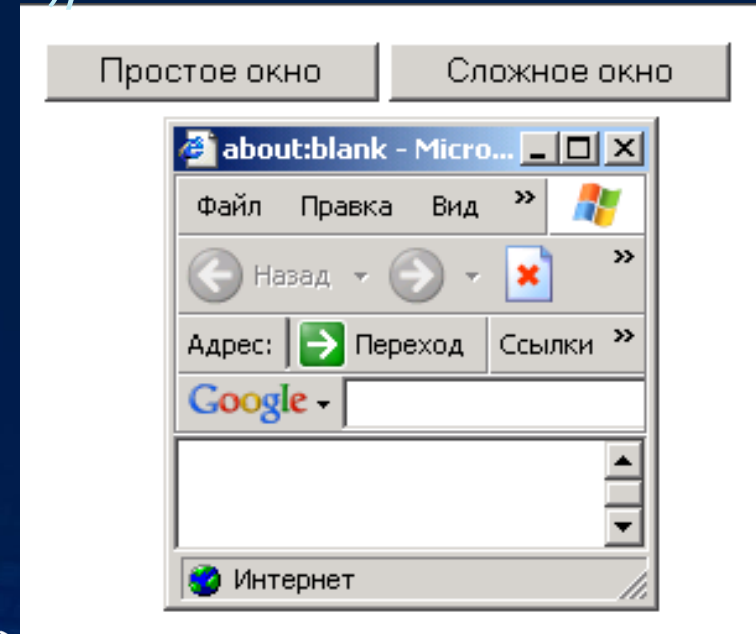
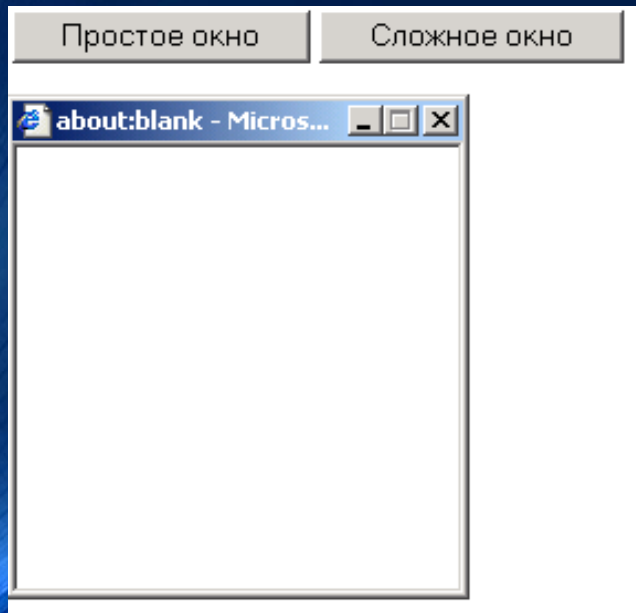
Параметры	Назначение
replace	Позволяет при открытии окна управлять записью в массив History
Width, Height	Ширина и Высота окна в пикселах
Toolbar	Создает <i>окно</i> с системными кнопками браузера
Location	Создает <i>окно</i> с полем location
Status	Создает <i>окно</i> с полем статуса status
Menubar	Создает <i>окно</i> с меню
Scrollbar	Создает <i>окно</i> с полосами прокрутки
Resizable	Создает <i>окно</i> , размер которого можно будет изменять



Управление окнами. Пример

```
<INPUT TYPE=button VALUE="Простое окно" onClick = "window.open('about:blank', 'test1','directories=no, height=200, location=no, menubar=no, resizable=no, scrollbars=no, status=no, toolbar=no, width=200');">
```

```
<INPUT TYPE=button VALUE="Сложное окно" onClick = "window.open('about:blank', 'test2', 'directories=yes, height=200, location=yes, menubar=yes, resizable=yes, scrollbars=yes, status=yes, toolbar=yes, width=200');">
```



Управление окнами. close()

Метод close() позволяет закрыть окно.

Если необходимо закрыть текущее, то:

```
window.close();
```

```
self.close();
```

Если необходимо закрыть родительское окно, т.е. окно, из которого было открыто текущее, то:

```
window.opener.close();
```

Если необходимо закрыть произвольное окно, то тогда сначала нужно получить его идентификатор:

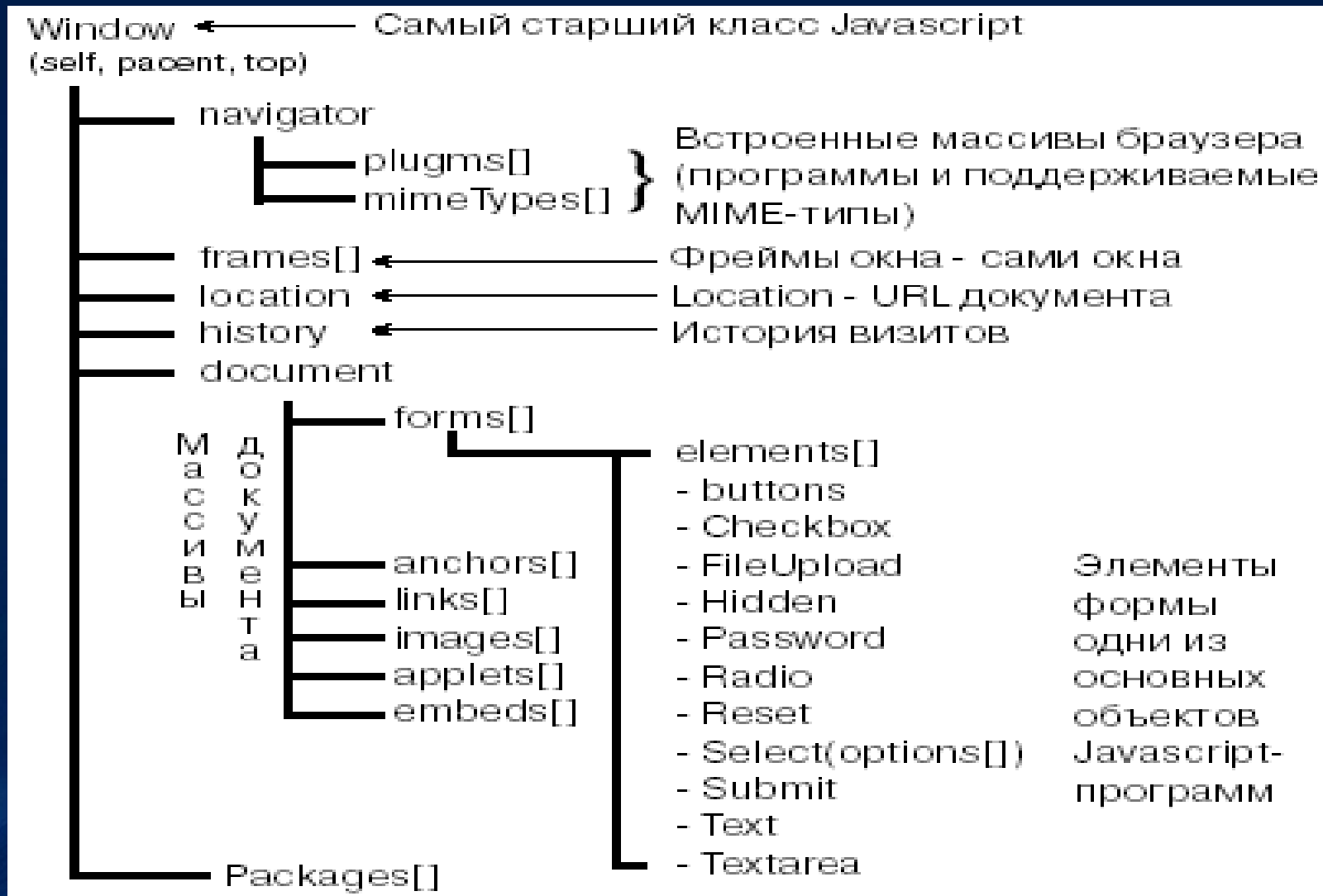
```
id=window.open(...);
```

```
...
```

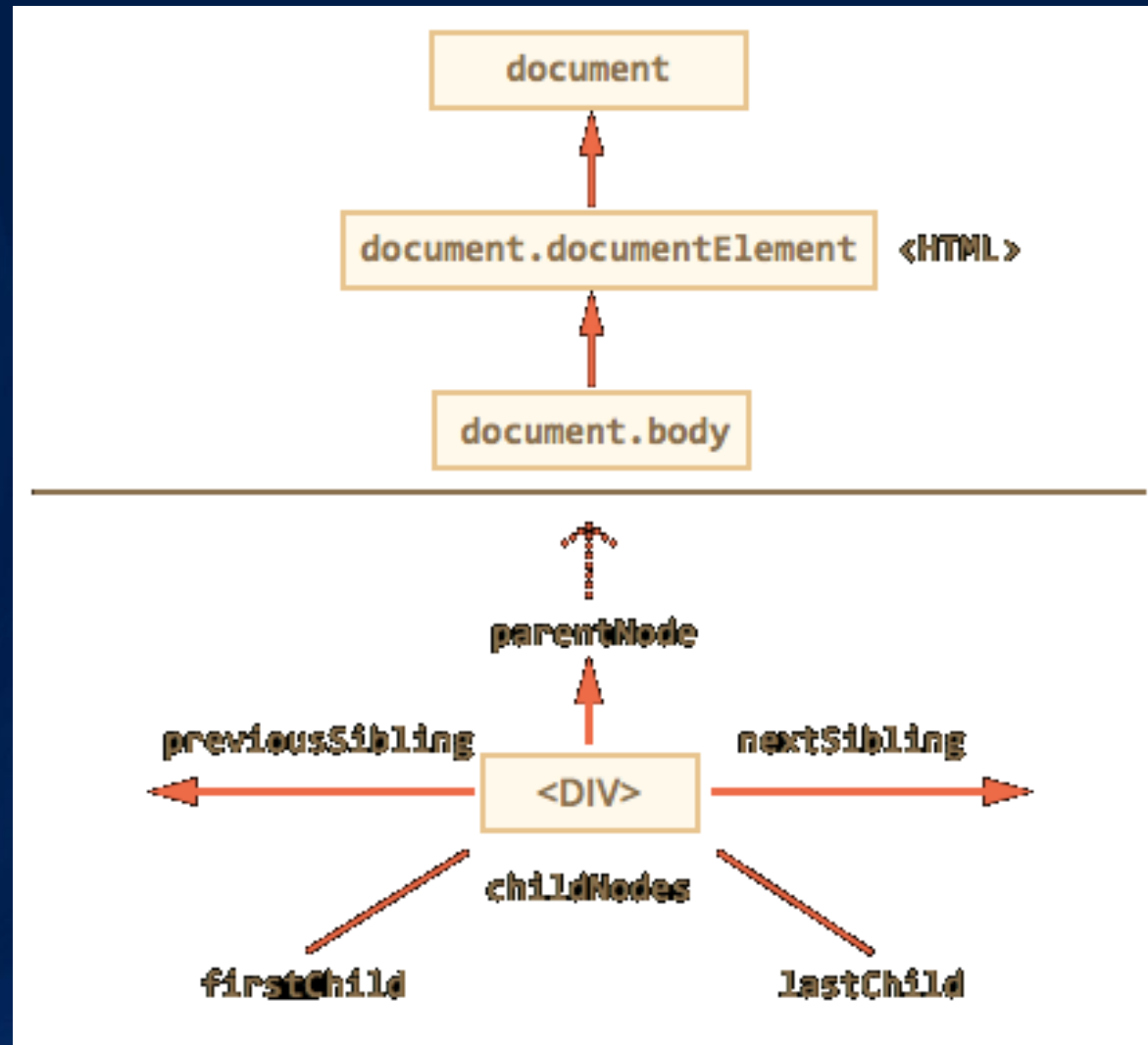
```
id.close();
```



Объектная модель документа



Дерево элементов страницы



Манипуляция элементами страницы

Для того, чтобы изменить узел DOM, например, изменить его свойства, добавить содержимое, нужно сначала его получить. Доступ к элементам DOM начинается с объекта **document**. Оттуда можно добраться до любых других узлов.

```
<body>
  <div>Пользователи:</div>
  <ul> <li>Маша</li>   <li>Вовочка</li> </ul>

  <script>
    var childNode = document.body.childNodes[1];
    for(var i=0; i<childNodes.length; i++) {
      document.write(childNode.childNodes[i].childNodes[0].nodeValue);
    }
  </script>
</body>
```

Пользователи:

- Маша
- Вовочка

Маша Вовочка



Поиск элементов страницы

Свойство	Описание
anchors	Возвращает массив содержащий все закладки имеющиеся на странице.
forms	Возвращает массив содержащий все формы имеющиеся на странице.
images	Возвращает массив содержащий все картинки имеющиеся на странице.
links	Возвращает массив содержащий все ссылки имеющиеся на странице.
referrer	Возвращает URL страницы с которой был совершен переход на данную.
document.readyState	Позволяет узнать статус загрузки документа.
title	Устанавливает или возвращает заголовок документа.



Поиск элементов страницы. Таблицы

У таблиц есть дополнительные свойства для более удобной навигации по ним:

table.rows – список строк TR таблицы.

tbody.rows – список строк TR секции.

tr.cells – список ячеек TD/TH.

tr.sectionRowIndex – номер строки в текущей секции THEAD/TBODY.

tr.rowIndex – номер строки в таблице.

td.cellIndex – номер ячейки в строке.

```
<table>
```

```
<tr><td> один</td><td> два</td></tr> <tr><td> три</td><td> четыре</td></tr>
```

```
</table>
```

```
<script>
```

```
var table = document.body.children[0];
```

```
alert(table.rows[0].cells[0].innerHTML); // "один"
```

```
</script>
```



Поиск элементов страницы

Самый удобный способ найти элемент в DOM – это получить его по id. Для этого используется вызов **document.getElementById(id)**

Следующий способ – получить все элементы с определенным тегом с помощью метода **document.getElementsByTagName(tag)**. Возвращает массив из элементов, имеющих заданный тег.

Метод **document.getElementsByName(name)** возвращает все элементы, у которых имя (атрибут name) равно данному.

Все методы **Elements** возвращают список узлов.

Метод **querySelectorAll** позволяет выбирать элементы с помощью CSS3-селектора.



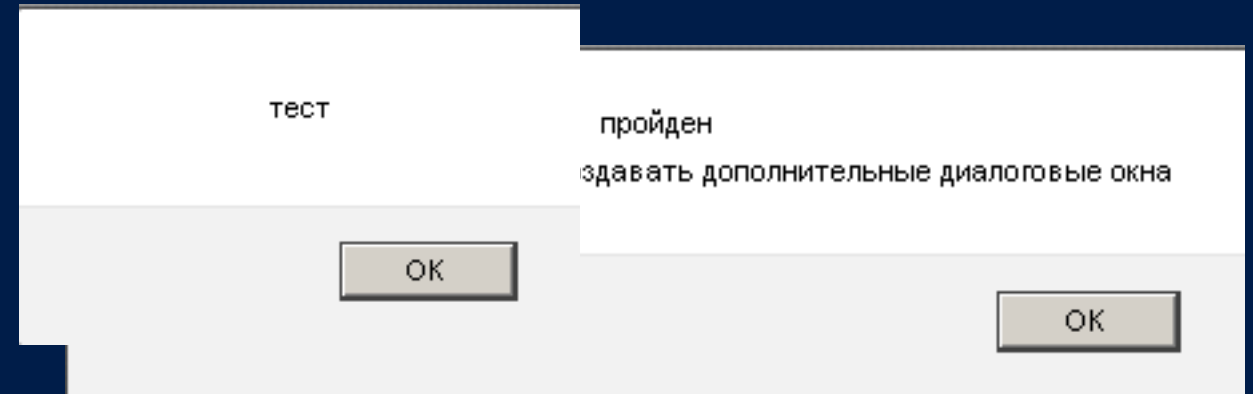
Поиск элементов страницы

Метод **querySelectorAll** позволяет выбирать элементы с помощью CSS-селектора.

```
<ul>  
  <li>Этот</li> <li>тест</li>  
</ul>
```

```
<ul>  
  <li>полностью</li>  
  <li>пройден</li>  
</ul>
```

```
<script>  
  var elements = document.querySelectorAll('ul > li:last-child');  
  for(var i=0; i<elements.length; i++) {  
    alert(elements[i].innerHTML ); // "тест", "пройден"  
  }  
</script>
```



jQuery

jQuery – библиотека JavaScript, позволяющая легко получать доступ к элементу DOM, обращаться к их атрибутам и содержимому, манипулировать ими, создавать анимацию, устанавливать обработчики событий. Также библиотека jQuery предоставляет и удобный API для работы с AJAX.

Библиотека представлена в виде обычного текстового файла, имеющего расширение .js. Сейчас существует две основные ветки версий jQuery 1.x и 2.x. Их отличия заключаются лишь в том, что в версиях 2.x перестали поддерживаться браузеры IE версий 8, 7 и 6.

jQuery доступна в **сжатом** и **несжатом** варианте.

Подключение jQuery:

```
<script type="text/javascript" src="js/jquery-1.6.1.min.js"></script>
```



jQuery. Подключение

Подключение jQuery с **CDN** (Content Delivery Network):

```
<script type="text/javascript" src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.min.js"></script>
```

Сначала попытайтесь загрузить jQuery со стороннего хранилища и в случае его отказа можете запросить jQuery со своего сервера:

```
<script type="text/javascript" src="http://ajax.microsoft.com/ajax/jquery/jquery-1.4.2.min.js"></script>  
<script type="text/javascript">  
  if (typeof jQuery == 'undefined') {  
    document.write(unescape("%3Cscript src='/js/jquery-1.6.1.min.js'  
type='text/javascript'%3E%3C/script%3E"));  
  }  
</script>
```



jQuery. Синтаксис

Стандартный синтаксис jQuery команд:

`$(селектор).метод();`

Знак **\$** сообщает, что символы идущие после него являются jQuery кодом.
Можно переопределить.

Селектор позволяет выбрать элемент на странице;

Метод задает действие, которое необходимо совершить над выбранным элементом. Методы в jQuery разделяются на следующие группы:

- Методы для **манипулирования** DOM;
- Методы для **оформления** элементов;
- Методы для привязки обработчиков **событий**;
- Методы для создания **AJAX** запросов;
- Методы для создания **эффектов**.



jQuery() или \$() . Поиск элементов

Функция **jQuery()** или **\$()** возвращает объект jQuery, который содержит выбранные элементы и имеет методы для работы с этими элементами. С ее помощью можно как находить существующие элементы на странице, так и добавлять новые.

<code>\$("div")</code>	вернет все div-элементы на странице.
<code>\$(".someBlock")</code>	вернет все элементы с классом someBlock.
<code>\$("#content")</code>	вернет элемент с идентификатором content.
<code>\$("#content2 div.someBlock")</code>	вернет div-элементы с классом someBlock, которые находятся внутри элемента с идентификатором content2.
<code>\$("div:odd")</code>	вернет div-элементы, находящиеся на странице под нечетными номерами.
<code>\$("[value = 5]")</code>	вернет все элементы с атрибутом value, равным 5.



jQuery. Методы для изменения элементов

<code>\$("#biglt").css("height")</code>	возвратит значение высоты у элемента с идентификатором biglt.
<code>\$("#div").css("width", "20px")</code>	установит новое значение ширины всем div-элементу на странице.
<code>\$("#biglt").attr("class")</code>	возвратит значение класса элемента с id = biglt.
<code>\$("#biglt").attr("class", "box")</code>	установит новое значение атрибута class у элемента с id = biglt.
<code>\$("#biglt").html(<p>New!</p>)</code>	изменит все html-содержимое элемента с id = biglt, на заданное в методе html.
<code>\$("#biglt").text()</code>	возвратит текст, находящийся внутри элемента с id = biglt.
<code>\$(".someBox").empty()</code>	очистить от содержимого элементы с классом someBox.



jQuery. Преждевременное выполнение

Выполнение кода до полной загрузки документа часто приводит к ошибкам. Возможно четыре способа предотвращения этой проблемы:

```
<script type='text/javascript'>
$(document).ready(function() {
    alert("1");
});
$.ready(function() {
    alert ("2");
});
$(function(){
    alert ("3");
});
</script>
```

```
<body>
<div>Содержимое тела
документа</div>

<script type='text/javascript'>
    alert ("4");
</script>
</body>
```



jQuery. Пример изменения свойств элементов

```
$(document).ready(function() {  
    $("p").css("fontSize", "20px");  
    $("#el2").css("color", "green");  
    $(".el3").css("color", "red");  
    $("#el2, .el3").css("fontWeight", "bold");  
    $("input").css("color", "blue");  
    $("[href]").css("fontSize", "20px");  
    $("[href='http://www.htmlbook.ru']").css("color", "green");  
});
```

```
<p> Первый элемент.</p>  
<p id="el2"> Второй элемент.</p>  
<p class="el3">Третий элемент.</p>  
<input type="button" value="Кнопка" />  
<br><a href="http://www.w3schools.com">  
http://www.w3schools.com</a>  
<br><a href="http://www.htmlbook.ru">  
HTML учебник</a>
```

Первый элемент.

Второй элемент.

Третий элемент.

Кнопка

<http://www.w3schools.com>

[HTML учебник](http://www.htmlbook.ru)



jQuery. Пример

```
<body>
  <ul id="list">
    <li class="item">Меркурий</li >
    <li class="item">Венера</li >
    <li class="item">Земля</li >
    .....
    <li class="item">Плутон</li >
  </ul>
  <script>
    $("#list .item").css("background-color", function(i,val){
      if($(this).text() == "Земля")
        return "#cceecc";
      else
        return val;
    });
  </script></body>
```

Меркурий

Венера

Земля

Марс

Юпитер

Сатурн

Уран

Нептун

Плутон



jQuery. Работа с атрибутами и свойствами

<code>.attr()</code>	возвращает/изменяет значение атрибута у элементов на странице
<code>.removeAttr()</code>	удаляет атрибут у элементов на странице
<code>.addClass()</code>	добавляет класс элементам на странице
<code>.removeClass()</code>	удаляет класс(ы) у элементов на странице
<code>.val()</code>	возвращает/изменяет (в зависимости от числа параметров) значение атрибута value у элементов на странице



jQuery. Добавление содержимого

<code>.html()</code>	Возвращает/изменяет (в зависимости от числа параметров) html-содержимое элементов на странице
<code>.text()</code>	Возвращает/изменяет (в зависимости от числа параметров) текст, находящийся в элементах на странице
<code>.append()</code> <code>.appendTo()</code>	Добавляет заданное содержимое в конец элементов на странице
<code>.after()</code> <code>.insertAfter()</code>	Добавляет заданное содержимое после элементов на странице
<code>.before()</code> <code>.insertBefore()</code>	Добавляет заданное содержимое перед элементами на странице
<code>.wrap()</code> <code>.wrapAll()</code>	Окружает элементы на странице заданными html-элементами
<code>.wrapInner()</code>	Окружает содержимое элементов на странице заданными html-элементами



jQuery. Работа с html-содержимым элемента

Функция `.html()` возвращает или изменяет html-содержимое выбранных элементов страницы. Функция имеет три варианта использования:

`.html()` – возвращает html-содержимое выбранного элемента. Если таких элементов несколько, то значение будет взято у первого.

`.html(newHTML)` – заменяет содержимое всех выбранных элементов на *newHTML*.

`.html(function(index, value))` – заменяет содержимое выбранных элементов на возвращенное пользовательской функцией значение. Функция вызывается отдельно, для каждого из выбранных элементов. При вызове ей передаются следующие параметры: *index* – позиция элемента в наборе, *value* – текущее html-содержимое.

<code>\$(".topBlock").html()</code>	вернет html-содержимое первого элемента с классом topBlock.
<code>\$(".topBlock").html("<p>New</p>")</code>	изменит содержимое всех элементов с классом topBlock на параграф с текстом "New".



jQuery. Работа с текстовым содержимым элемента

Функция **.text()** возвращает или изменяет текстовое содержимое выбранных элементов страницы. Функция имеет три варианта использования:

.text() – возвращает текст содержащийся в выбранном элементе. Если таких элементов несколько, метод возвратит строку, в которой будет содержимое всех элементов, расположенное через пробел.

.text(newText) – заменяет все содержимое у выбранных элементов, на текст **newText**.

.text(function(index, value)) – заменяет все содержимое у выбранных элементов на возвращенный пользовательской функцией текст. Функция вызывается отдельно, для каждого из выбранных элементов.

<code>\$(".topBlock").text()</code>	вернет текстовое содержимое всех элементов с классом topBlock (одной строкой).
<code>\$(".topBlock").text("<p>New</p>")</code>	заменит содержимое всех элементов с классом topBlock на текст "New".



jQuery. Добавление содержимого после элементов

Функции `.after()` `.insertAfter()` вставляют заданное содержимое сразу после определенных элементов страницы. Имеется два варианта использования :

`elements.after(content), content.insertAfter(elements)`

сразу после элементов *elements* будет добавлено содержимое *content*, который может быть задан html-текстом, объектом jQuery или DOM объектом. Различия функций заключается только в порядке следования содержимого и элементов, после которых это содержимое должно быть вставлено.

`.after(function(index))`

после выбранных элементов будет добавлен html-текст, который будет возвращен пользовательской функцией. Функция вызывается отдельно, для каждого из выбранных элементов. При вызове ей передается один параметр: *index* – позиция элемента в наборе.



jQuery. Добавление содержимого после элементов

```
<ul class="list l1">
<li class="item it1"> Высоко </li>
<li class="item it2"> Быстро </li>
<li class="item it3"> Сильно </li>
</ul>
<ul class="list l2">
<li class="item it1"> Выше </li>
<li class="item it2"> Быстрее </li>
<li class="item it3"> Сильнее </li>
</ul>
```

```
$(".it1").after("<li class='item'>Тест</li>");
или
$("<li class='item'>Тест</li>").insertAfter($(".it1"));
```

```
<ul class="list l1">
<li class="item it1"> Высоко </li>
<li class="item"> Тест </li>
<li class="item it2"> Быстро </li>
<li class="item it3"> Сильно </li>
</ul>
<ul class="list l2">
<li class="item it1"> Выше </li>
<li class="item"> Тест </li>
<li class="item it2"> Быстрее </li>
<li class="item it3"> Сильнее </li>
</ul>
```



Вопросы

- Методы объекта window для взаимодействия с пользователем.
- Чем отличается setTimeout от setInterval?
- Способы получения элементов страницы. Получение дочерних элементов таблиц.
- Как избежать преждевременного выполнения кода JavaScript?
- Библиотека jQuery, способы подключения и различия версий.
- Как получить элемент DOM при помощи jQuery?
- Какой метод jQuery позволяет добавить содержимое в конце заданного элемента?

