Языки Интернет-программирования

Лекция 2. Javascript. DOM

- Язык Javascript
- Отладка Javascript-кода
- Модель W3C DOM
- Библиотеки jQuery, Dojo
- Прочие перспективные библиотеки

МГТУ им. Н.Э. Баумана, доц. каф. ИУ-6,
к.т.н. Самарев Роман Станиславович

samarev@acm.org

Javascript

10.09.2019

- Версия 1.0 была реализована в
  Netscape 2.0, март 1996 (разработчик Brendan Eich).

- Исторически является диалектом стандартизованного языка
  ECMA_262 (версия 3)
  European Computer Manufacturer’s Association (ECMA) –
  http://www.ecma-international.org/

- Торговая марка JavaScript принадлежит Oracle

- Текущая версия JavaScript:
  ECMAScript 7 / 8th edition (June 2017)
  http://www.ecma-international.org/publications/standards/Ecma-262.htm

Страницы с динамикой JavaScript

10.09.2019

```
<!DOCTYPE html>
<html>
<head>
    <title>Факториалы</title>
     <meta charset="utf-8"/>
</head>
<body>
    <h1>Таблица факториалов</h1>
    <script>
        for(fact = i = 1; i < 10; i++) {
            fact = fact * i;
            document.write(i + "! = " + fact + "<br />");
        }
    </script>
</body>
</html>
```

- URL: javascript: for(fact=i=1;i<10;i++){fact=fact*i; document.write(i+"! = "+fact+"
");};

Основные особенности JavaScript

- Язык с динамической типизацией и автоматическим управлением памяти
- Объектно-ориентированный
- Первоначальная область применения – web-приложения
- Используется как на серверах, так и на клиентах
- Используется для создания расширяемых интерфейсов (Mozilla и Chrome)
- Node.js может запускать js как обычные скрипты.

Подключение в HTML
Внедрение

10.09.2019

```
<!DOCTYPE html>
<html>
<head>
    <title>Факториалы</title>
    <meta charset="utf-8" />
</head>
<body>
    <h1>Таблица факториалов</h1>
    <script type="text/javascript">
        for(fact = i = 1; i < 10; i++) {
            fact = fact * i;
            document.write(i + "! = " + fact + "<br />");
        }
    </script>
</body>
</html>
```

Подключение в HTML
Подключение внешнего ресурса

10.09.2019

```
<!DOCTYPE html>
<html>
<head>
    <title>Факториалы</title>
    <meta charset= "utf-8" />
    <script type="text/javascript" src="fact.js"> </script>
</head>
<body>
    <h1>Таблица факториалов</h1>
    <script type="text/javascript">
        fact();
    </script>
</body>
</html>
```

Особенности синтаксиса

- ECMAScript v1 и v2
  7-ми битная ASCII-кодировка и UNICODE-символы в комментариях или строковых литералах.

- ECMAScript v3
  UNICODE в любой части программы.
  (рекомендуется использовать только латинские буквы в идентификаторах)

Особенности синтаксиса
Продолжение

10.09.2019

- Чувствителен к регистру
- Табуляция и перевод строк игнорируются
- Разделителем простых операторов является символ **;**
  не обязателен, если операторы разделены переводом строк.
  Рекомендован для использования -
  https://www.w3schools.com/js/js_conventions.asp

a=1; b=3;	return true	return true
Эквивалентно a=1 b=3	Не эквивалентно!!! **return true;**	Эквивалентно **return;** // возврат true; // код не выполняется

Комментарии

- // однострочный комментарий

- /* многострочный
  комментарий*/

Литералы

10.09.2019

- 12 // Число двенадцать
- 1.2 // Число одна целая две десятых
- "hello world" // Строка текста
- 'Hi' // Другая строка
- true // Логическое значение
- false // Другое логическое значение
- /javascript/gi // Регулярное выражение (поиск по шаблону)
- null // Отсутствие объекта

- { x:1, y:2 } // Инициализатор объекта
- [1,2,3,4,5] // Инициализатор массива

Числа

10.09.2019

- Внутреннее представление вещественное с плавающей точкой 64 бита (IEEE 754)
- Диапазоны:
  - Целые: –9007199254740992 (-2^{53}) до 9007199254740992 (2^{53})
  - Вещественные: от $\pm 1{,}7976931348623157 \times 10^{308}$ до $\pm 5 \times 10^{-324}$.
- Примеры:
  - 0, 234, 652345.
  - 0xF1, 0xfe340b
  - Вещ.: [цифры][.цифры][(E|e)[(+|_)]цифры]
  - ECMAScript 6:
    0b111110111 // двоичное число = 503
    0o767 // восьмеричное число = 503

Строки

- последовательность из нуля или более Unicode-символов, заключенная в ' или "
- Может содержать \n, \b, \0, \xXX, \xXXXX…
- Пример:
  - str = "Некоторый текст"

Некоторые строковые операции

- длина - str.length

  str = "123" + "456" // в итоге 123456

- // выделить подстроку 5 символов начиная с 2

  substr = str.substring(2,5);

- // получить индекс символа в строке

  i = str.indexOf('a');

Некоторые строковые функции

- charAt(number) – получить символ в указанной позиции
- lastIndexOf(substring) – найти последнюю позицию вхождения подстроки
- toLowerCase() – преобразовать в нижний регистр
- toUpperCase() –преобразовать в верхний регистр
- split(char) – разбить подстроку по указанному разделителю
- …

Строковые функции ECMAScript 6

10.09.2019

- String interpolation:
  ```
  let first = 'Jane';
  let last = 'Doe';
  console.log(`Hello ${first} ${last}!`);
  ```
- ```
  let multiLine = `
     This is a string
     with multiple
     lines`;
  ```
- ```
  'hello'.startsWith('hell') // true
  'hello'.endsWith('ello')  // true
  'hello'.includes('ell')     // true
  "hello".includes("ell", 1) // true
  "hello".includes("ell", 2) // false
  ```
- …

- http://www.2ality.com/2015/01/es6-strings.html

Преобразование чисел в строку


```
var n = 10
str = n + " получено в результате";
str = n.toString() + " получено в результате";

n = 123456.789;
n.toFixed(0);          // "123457"
n.toFixed(2);          // "123456.79"
n.toExponential(1); // "1.2e+5"
n.toExponential(3); // "1.235e+5"
n.toPrecision(4);    // "1.235e+5"
n.toPrecision(7);    // "123456.8"
```

Преобразование строки в число


```
var n = "12" * "2" // результат 24

n = "12" + "2" // результат "122", поскольку + является конкатенацией

Number("12")
parseInt("3 объекта");          // результат 3
parseFloat("3.14 метров");      // результат 3.14
parseInt("12.34");              // результат 12
parseInt("0xFF");               // результат 255
parseInt("11", 2);              // результат 3 (1*2 + 1)
parseInt("ff", 16);             // результат 255 (15*16 + 15)
parseInt("zz", 36);             // результат 1295 (35*36 + 35)
parseInt("077", 8);             // результат 63 (7*8 + 7)
parseInt("077", 10);            // результат 77 (7*10 + 7)
```

Функции


```
// декларируем функцию sum с двумя аргументами
function sum (a,b) {
   return a+b;
}

// функциональный литерал:
var sum = function(a,b) { return a+b; };
```

Объекты

Объект – это коллекция **именованных** значений (свойств)

```
var point = new Object(); // создаём пустой объект !!!
point.x = 2.3; // присоединяем свойство x
point.y = 1.2; // присоединяем свойство y
```

Обращение к свойствам:

- point.**x**
- point [**"x"**]

Литералы объекты:

- var point = { x:2.3, y:1.2 }; // Простой
- var rectangle = { // Вложенный
  upperLeft: { x: 2, y: 2 },
  lowerRight: { x: 4, y: 4 }
  };

Массивы

10.09.2019

Массив – это коллекция **нумерованных** значений (любых!)

```
// Последовательное заполнение
var a = new Array();
a[0] = 1.2;
a[1] = "JavaScript";
a[2] = true;
a[3] = { x:1, y:3 };

var a = new Array( 1.2, "JavaScript",
                   true, { x:1, y:3 } ); // инициализация в конструкторе

var a = new Array(10);  // массив ровно на 10 элементов

// Литералы
var a = [1.2, "JavaScript", true, { x:1, y:3 }];  // произвольные объекты
var matrix = [[1,2,3], [4,5,6], [7,8,9]];      // матрица

var base = 1024;
var table = [base, base+1, base+2, base+3]; // вычисленные выражения
```

Переменные
Области видимости

10.09.2019

```
function test(o) {
  var i = 0; // i определена во всей функции
  if (typeof o == "object") {
    var j = 0; // j определена везде, а не только в блоке
    for (var k = 0; k < 10; k++) {
     // k определена везде, а не только в цикле
      document.write(k);
    }
    // k все еще определена: печатается 10
    document.write(k);
  }
  // j определена, но может быть не инициализирована
  document.write(j);
}
```

Область видимости переменных

10.09.2019

- function closure

```
var x = function(){
        var i;
        mul = function(b) { return i*b; }
        for(fact = i = 1; i < 10; i++) {
                fact = mul(fact);   // i подставляется здесь
                document.write(i + "! = " + fact + "<br />");
        }
}
```

- ES-6. Block-Scoped Variables

```
for (let i = 0; i < a.length; i++) {
        let x = a[i] …
}
for (let i = 0; i < b.length; i++) {
        let y = b[i] …
}
```

Глобальные переменные


```
var scope = "глобальная";
function f() {   // Определяем функцию
  alert(scope); // Показывает "глобальная".
  scope = "локальная"; // Переменная глобальная.
  alert(scope); // Показывает "локальная"
}
f();   // Вызываем функцию
alert(scope); // Показывает "локальная“
```

- alert() – вывод сообщения в диалоговом окне

Локальные переменные

10.09.2019

```
var scope = "глобальная";
function f() {
  alert(scope); // Показывает "undefined", а не "глобальная".
  var scope = "локальная"; // Переменная инициализируется здесь,
                           // но определена она везде
                           // внутри функции.
  alert(scope); // Показывает "локальная"
}

f();

alert(scope); // Показывает "глобальная".
```

Операторы сравнения

10.09.2019

Оп.	Описание	Пример (x=5)	Результат
==	Равенство	x == 8 x == 5	*false* *true*
===	Равенство с учетом типа	*x === "5"* *x === 5*	*false* *true*
!=	Неравенство	x != 8	*true*
!==	Неравенство с учетом типа	x !== "5" x !== 5	*true* *false*
>	Больше	x > 8	*false*
<	Меньше	x < 8	*true*
>=	Больше или равно	x >= 8	*false*
<=	Меньше или равно	x <= 8	*true*

Ветвление

- **if** (*condition*) {
  …
  } **else** {
  …
  }

- *variablename* = **(***condition***) ?** *value1* **:** *value2*

- **switch**(n)
  {
  case 1:
  execute code block 1
  break;
  case 2:
  execute code block 2
  break;
  default:
  code to be executed if n is different from case 1 and 2
  }

Циклы

10.09.2019

- ```
  for (i=0; i<5; i++) {
        x = x + "The number is " + i + "<br />";
  }
  ```
- ```
  while (i<5) {
        x = x + "The number is " + i + "<br />";
        i++;
  }
  ```
- ```
  do {
        x = x + "The number is " + i + "<br />";
        i++;
  } while (i<5);
  ```

- ```
  var person = {fname:"John", lname:"Doe", age:25};
  for (x in person) {                    // цикл по именам свойств
     console.log(x +":"+ person[x])          // fname:John, lname:Doe, age:25
  }
  ```
- ```
  var array = [1, 2, 3, 4, 5];
  for(var i of array) {                  // цикл по значениям
     console.log(i);                     // 1, 2, 3, 4, 5
  }
  ```
  ```

Средства отладки для Javascript

10.09.2019

- Вывод диалогового окна
  - alert(…)
- Отладочный вывод в консоль браузера
  - console.log(…), console.error(…),…
- Средства пошаговой отладки
  - Mozilla Firefox + отладчик (+ Firebug)
  - GoogleChrome + отладчик
  - ...

“Классы” до ECMAScript 6

10.09.2019

```
// Определяем конструктор.
// инициализация объекта с помощью "this"!
function Rectangle(w, h) {
  this.width = w;
  this.height = h;
}
// Вызываем конструкторы для создания двух объектов Rectangle.
// инициализируем оба новых объекта.
var rect1 = new Rectangle(2, 4);      // rect1 = { width:2, height:4 };
var rect2 = new Rectangle(8.5, 11); // rect2 = { width:8.5, height:11 };
```

Добавим метод:

```
rect1.area = function() { return this.width * this.height; }
```

….

“Классы”
Метод как свойство

10.09.2019

```
function Rectangle(w, h) {
  this.width = w;
  this.height = h;
  this.area = function( ) { return this.width * this.height; }
}
```

area – свойство. Любой объект может его подменить!

“Классы”
Прототипы

10.09.2019

```
// Конструктор
function Circle(radius) {
  // r – свойство экземпляра объекта, определяется
  // и инициализируется конструктором.
  this.r = radius;
}
// Circle.PI – свойство класса(общее для всех), свойство конструктора.
Circle.PI = 3.14159;

// Метод для экземпляра, который рассчитывает площадь круга.
Circle.prototype.area = function( ) { return Circle.PI * this.r * this.r; }

// Метод класса (аналог статических методов в C++)
Circle.max = function(a,b) {
  if (a.r > b.r) return a;
  else return b;
}
```

“Классы”
Прототипы. Использование.

10.09.2019

```
var c = new Circle(1.0); // Создание экземпляра класса Circle

c.r = 2.2; // Установка свойства экземпляра r

var a = c.area(); // Вызов метода экземпляра area()

var x = Math.exp(Circle.PI); // Обращение к свойству PI класса

var d = new Circle(1.2); // Создание другого экземпляра класса Circle

var bigger = Circle.max(c,d); // Вызов метода класса max()
```

ECMAScript 6 New Features

10.09.2019

- http://es6-features.org
  - `const PI = 3.141593`
  - `odds = evens.map(v => v + 1)`
    `// odds = evens.map(function (v) { return v + 1; });`
  - `function f (x, y = 7, z = 42) { return x + y + z }`
  - ```
    class Shape {
      constructor (id, x, y) {
                    this.id = id; this.move(x, y);
       }
       move (x, y) { this.x = x; this.y = y; }
    }
    class Rectangle extends Shape {
      constructor (id, x, y, width, height) {
        super(id, x, y)
        this.width = width; this.height = height
      }
    }
    ```
  - `get `http://example.com/foo?bar=${bar + baz}&quux=${quux}`;`
  - …
    ```

Модель DOM

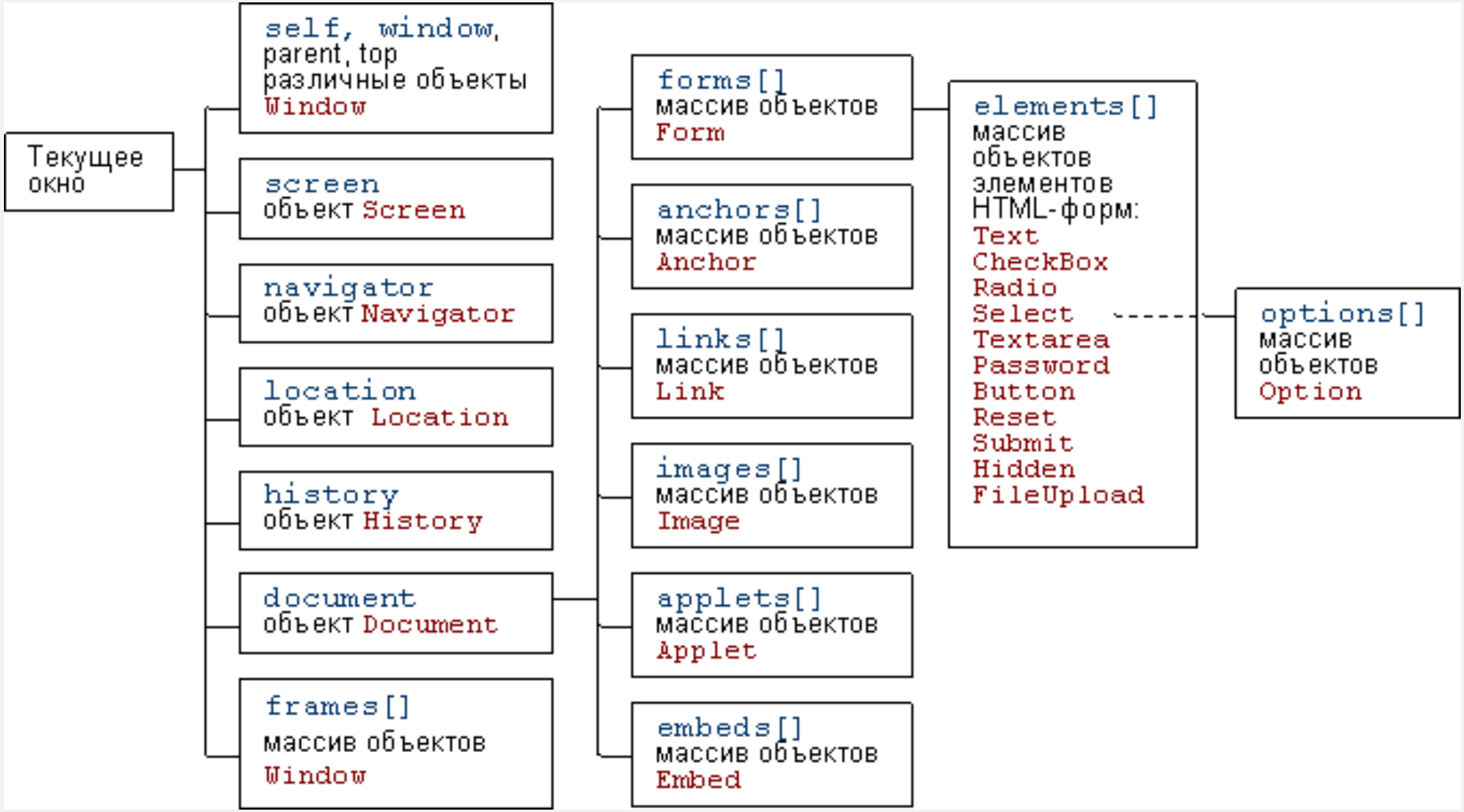
10.09.2019

W3C The Document Object Model Working Group:

- Document Object Model Level 1
- *Document Object Model Level 2 Core*
- *Document Object Model Level 2 Views*
- *Document Object Model Level 2 Events*
- *Document Object Model Level 2 Style*
- *Document Object Model Level 2 Traversal and Range*
- *Document Object Model Level 2 HTML*
- Document Object Model Level 3 Core
- Document Object Model Level 3 Load and Save
- Document Object Model Level 3 Validation

DOM 0

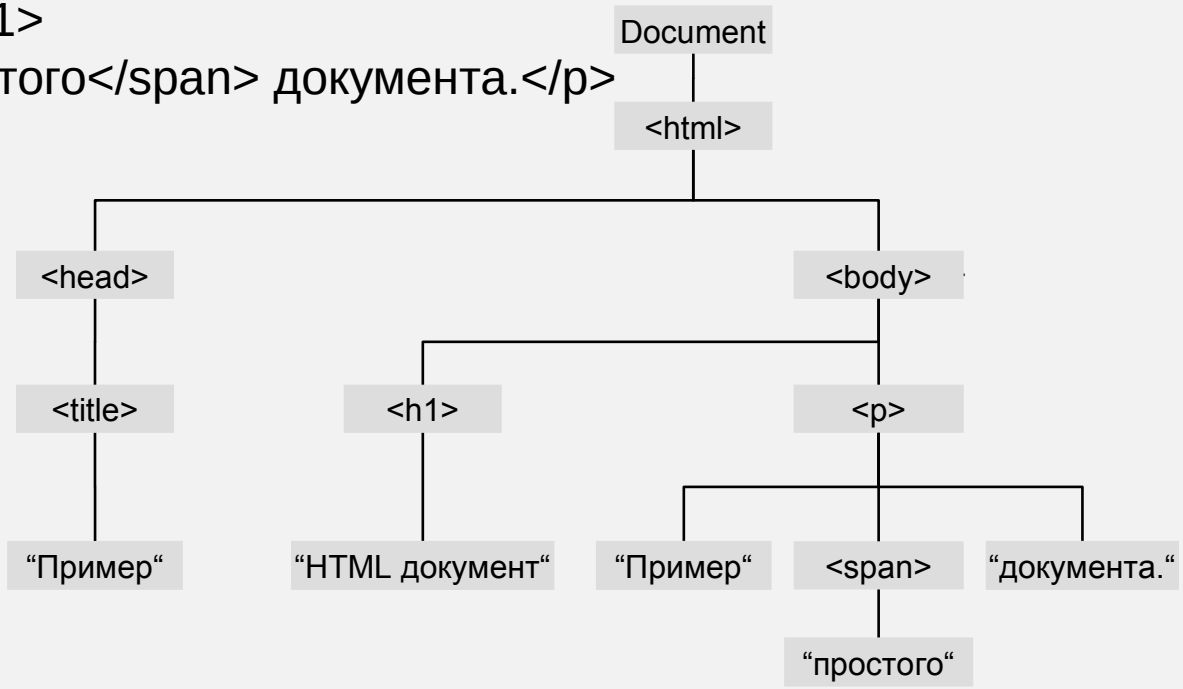
10.09.2019



Дерево элементов

10.09.2019

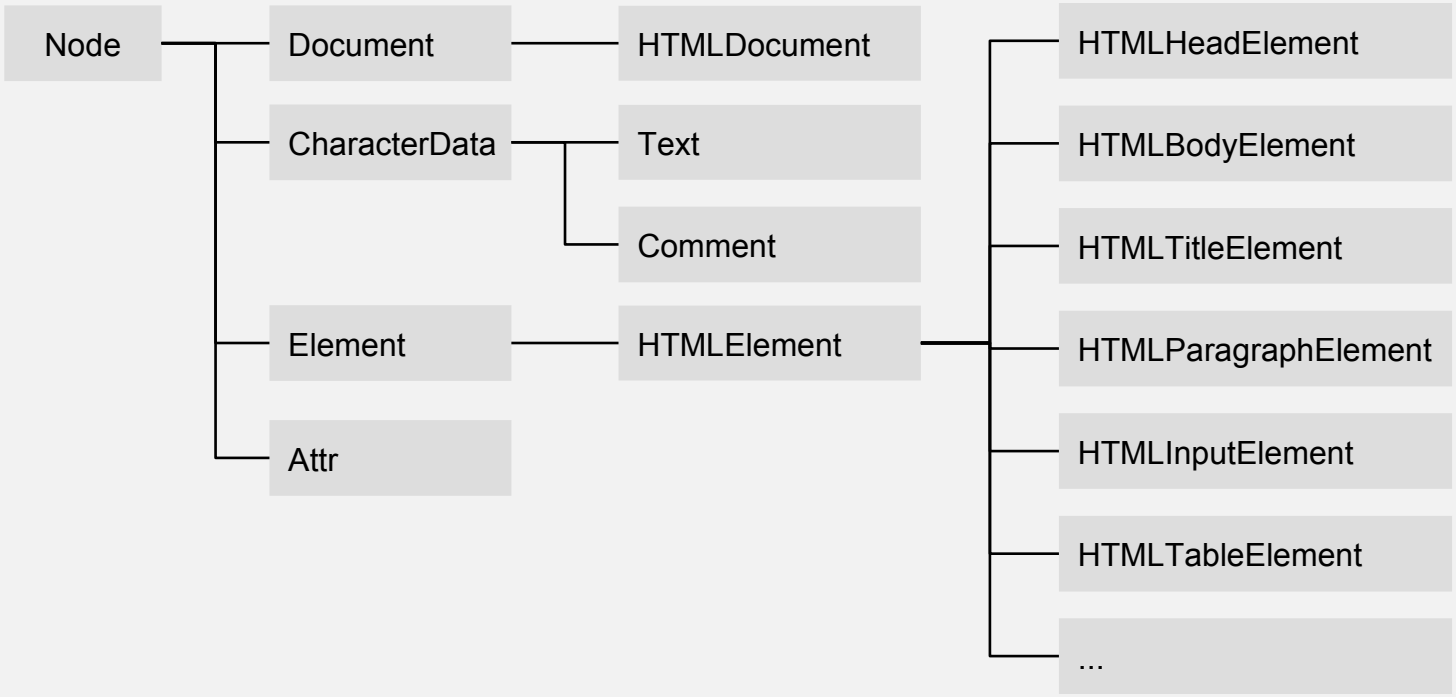
```
<html>
  <head>
    <title>Пример</title>
  </head>
  <body>
    <h1>HTML документ</h1>
    <p>Пример <span>простого</span> документа.</p>
  </body>
</html>
```



Типы узлов

10.09.2019

document.childNodes



Доступ к элементам и их атрибутам

10.09.2019

- Обращение через свойства и методы Node
  - childNodes (коллекция дочерних узлов, включая текст и комментарии),
  - children (коллекция дочерних узлов — только HTML)
  - firstChild, lastChild (первый и последний потомок)
  - nextSibling, previousSibling (элементы одного уровня)
  - parentNode (родительский элемент)
  - Добавить, удалить, заменить и вставить элемент перед указанным:
    appendChild(), removeChild()
    replaceChild(), insertBefore()

- Обращение по имени элемента document.**body.h1**

Доступ по имени элемента, классу или по идентификатору

10.09.2019

по имени тега - getElements**ByTagName**, getElements**ByTagNameNS**
по имени класса - getElements**ByClassName** // вернуть коллекцию элементов
по идентификатору - getElement**ById** // вернуть один элемент

Пример 1.

```
// найти все таблицы
var tables = document.getElementsByTagName("table");
alert("Количество таблиц в документе: " + tables.length);

// Взять первый элемент <table>
var table = tables[0];

// Установить толщину линии
table.border = 10;
```

Пример 2

```
<p id="specialParagraph">
…
var myParagraph = document.getElementById("specialParagraph");
myParagraph.style="font-family:monospace"
```

Добавление узлов


```
// Создать элемент <table>
var table = document.createElement("table");

table.border = 1; // Установить атрибут

// Добавить в таблицу заголовок Имя|Тип|Значение
table.innerHTML =
    "<tr><th>Имя</th><th>Тип</th><th>Значение</th></tr>";
```

Пример сортировки списка

10.09.2019

```
<!DOCTYPE html>
<html>
<head>
<title>Сортировка списка. Дэвид Флэнаган. JavaScript.</title>
<meta charset="UTF-8" />
<script> function sortkids(e) {…}</script>
</head>
<body>
    <ul id="list"> <!--Этот список будет отсортирован -->
    <!-- элементы не в алфавитном порядке -->
    <li>один<li>два<li>три<li>четыре<li>пять
    </ul>
    <!-- кнопка, щелчок на которой запускает сортировку списка -->
    <button onclick="sortkids('list')">Сортировать по
    алфавиту</button>
</body>
</html>
```

Пример сортировки списка Продолжение


```
function sortkids(e) {
    // Это элемент, потомки которого должны быть отсортированы
    if (typeof e == "string") e = document.getElementById(e);
    var kids = []; // Скопировать дочерние элементы (не текстовые узлы) в массив
    for(var x = e.firstChild; x != null; x = x.nextSibling)
    if (x.nodeType == 1 /* Node.ELEMENT_NODE */) kids.push(x);
    // Выполнить сортировку на основе текстового содержимого каждого
    подэлемента.
    // Предполагается, каждый потомок имеет единственный вложенный узел Text
    kids.sort(function(n, m) { // Функция сравнения для сортировки
    var s = n.firstChild.data;  // Текстовое содержимое узла n
    var t = m.firstChild.data;  // Текстовое содержимое узла m
    if (s < t) return -1;           // Узел n должен быть выше узла m
    else if (s > t) return 1;       // Узел n должен быть ниже узла m
    else return 0;    // Узлы n и m эквивалентны
    });
    // перенести узлы-потомки обратно в родительский элемент в новом порядке.
    // Уже существующие элементы при вставке удаляются из старой позиции
    for(var i = 0; i < kids.length; i++) e.appendChild(kids[i]);
}
```

Регистрация обработчиков

- Обработчик как атрибут элемента

<p **onclick**="print(this)">

- Обработчик как свойство объекта

```
var mydiv=document.getElementById('result');
mydiv.onclick = function(){…}
```

- Регистрация обработчика в DOM2

```
var mydiv = document.getElementById("mydiv");
mydiv.addEventListener("mousedown", handleMouseDown, true);
```

Обработка событий
Назначение обработчиков

10.09.2019

```
<body> <div>
     <h1>События HTML</h1>
     <p>Нажмите на текст любого параграфа</p>
  </div>
 <div class="text">
     <p onclick="print(this)">1. Параграф внутри div !</p>
     <p onclick="print(this)">2. Параграф внутри div !!</p>
  </div>
  <div class="text">
     <p onclick="print(this)"
    onMouseOut="mouseOut(this)" onMouseOver="mouseOver(this)">
            3. Параграф внутри интересующего нас div !!!</p>
     <p onclick="print(this)"
    onMouseOut="mouseOut(this)" onMouseOver="mouseOver(this)">
            4. Параграф внутри интересующего нас div !!!!</p>
  </div>
  <div id="result"><!-- Сюда поместим результат --></div>
</body>
```

Обработка событий
Реализация обработчиков

10.09.2019

```
function print(el) {
    // По идентификатору находим элемент,
    // в который надо поместить результат
    var result = document.getElementById('result');
    result.innerHTML = "<hr/><p>"+Date()+"</p>" +
            "&lt;" + el.nodeName + "&gt;" + el.textContent +
            "<p>Вывод завершен!</p><hr/>";
}
function mouseOver(el) {
    el.style.backgroundColor = "green";
}
function mouseOut(el) {
    el.style.backgroundColor = "yellow";
}
```

Активация кода после загрузки страницы

10.09.2019

- Событие onload
  - Регистрация в HTML-разметке
    <body onload="myFunction()">
  - Регистрация в коде JS
    window.onload = function() { myScript };
  - Регистрация в коде JS в цепочке обработчиков
    window.addEventListener("load", function() { myScript });
- https://www.w3schools.com/jsref/event_onload.asp

Модули, интерфейсы и типы событий...

10.09.2019

Имя модуля	Интерфейс Event	Типы событий
HTMLEvents	Event	abort, blur, change, error, focus, load, reset, resize, scroll, select, submit, unload
MouseEvents	MouseEvent	click, mousedown, mousemove, mouseout, mouseover, mouseup
UIEvents	UIEvent	DOMActivate, DOMFocusIn, DOMFocusOut
MutationEvents	MutationEvent	DOMAttrModified, DOMCharacterDataModified, DOMNodeInserted, DOMNodeInsertedIntoDocument, DOMNodeRemoved, DOMNodeRemovedFromDocument, DOMSubtreeModified

Сопутствующие технологии

10.09.2019

- https://nodejs.org – серверный «движок» JavaScript
- https://npmjs.com (npm) – используется для локальной разработки как средство управления
- https://www.typescriptlang.org/ – язык с явной типизацией. Транслируется в Javascript
- https://svelte.dev/ - декларативный генератор кода JS
- https://coffeescript.org/ – компактный язык для упрощения разработки. Транслируется в Javascript
- …

Исторические библиотеки для JS

- jQuery (http://www.jquery.com)
  - Синтаксис, основанный на селекторах CSS.
  - Предназначена для упрощения манипулирования элементами.
  - Имеются модули для создания интерактивных графических элементов
- Dojo (http://dojotoolkit.org/)
  - Универсальная библиотека для решения большинства задач клиентского Javascript программирования.
  - Концепция виджетов. Ориентирована на создание оконных интерфейсов.
  - Имеются средства для быстрой генерации форм.
- …

Основные принципы jQuery

10.09.2019

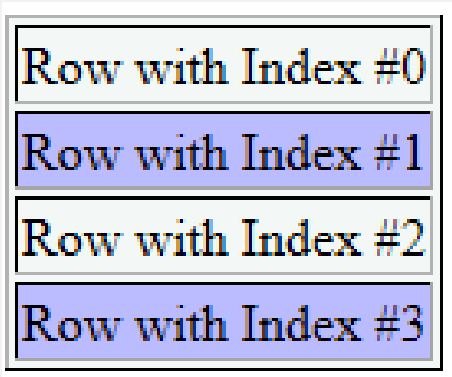
- Функции $(), jQuery()
- Синтаксис селекторов CSS
- События
- AJAX-дополнения
- Визуальные эффекты
- Подключение дополнительных модулей

- http://www.jquery.com , http://jquery-docs.ru/

jQuery
Пример использования

10.09.2019

```
<!DOCTYPE html>
<html>
<head>
  <style>
   table { background:#f3f7f5; }
  </style>
  <script src="http://code.jquery.com/jquery-latest.js"></script>
</head>
<body>
  <table border="1">
    <tr><td>Row with Index #0</td></tr>
    <tr><td>Row with Index #1</td></tr>
    <tr><td>Row with Index #2</td></tr>
    <tr><td>Row with Index #3</td></tr>
  </table>
  <script>$("tr:odd").css("background-color", "#bbbbff");</script>
</body>
</html>
```



jQuery. Обработка событий
Разметка HTML

10.09.2019

```
<body>
    <div>
        <h1>События HTML</h1>
        <p>Нажмите на текст любого параграфа</p>
    </div>
    <div class="text">
        <p>1. Параграф внутри интересующего нас div !</p>
        <p>2. Параграф внутри интересующего нас div !!</p>
    </div>
    <div class="text">
        <p>3. Параграф внутри интересующего нас div !!!</p>
        <p>4. Параграф внутри интересующего нас div !!!!</p>
    </div>

    <!-- Сюда поместим результат -->
    <div id="result"></div>
</body>
```

jQuery. Обработка событий
Код обработчиков

10.09.2019

```
$(document).ready(function(){
    $("div.text p").mouseover(function(){
        this.style.backgroundColor = "green";
    }).mouseout(function(){
        this.style.backgroundColor = "yellow";
    }).click(function () {
        // находим элемент, куда надо поместить результат
        $("#result").html("<hr/><p>"+Date()+"</p>" +
            "&lt;" + this.nodeName + "&gt;" + this.textContent +
            "<p>Вывод завершен!</p><hr/>");
    })
});
```

jQuery
Визуальные эффекты

10.09.2019

```
<!DOCTYPE html><html lang="en"><head>
    <script src="js/jquery.js"></script>
    <script src="js/ui/jquery-ui.js"></script>
    <link rel="stylesheet" href="js/demos.css">
    <style>
        #sortable { list-style-type: none; margin: 0; padding: 0; width: 60%; }
        #sortable li {
            margin: 0 5px 5px 5px; padding: 0.4em; padding-left: 1.5em;
            font-size: 1.4em; height: 18px; background-color:cyan }
        #sortable li span { position: absolute; margin-left: -1.3em; }
    </style>
    <script>$( function() {
            $( "#sortable" ).sortable();
            $( "#sortable" ).disableSelection(); });
    </script>
</head><body><div class="demo"><ul id="sortable">
    <li class="ui-state-default">Item 1</li>
    <li class="ui-state-default">Item 2</li>
    <li class="ui-state-default">Item 3</li>
</ul></div></body></html>
```

jQuery
Дополнительные компоненты

10.09.2019

- http://jqueryui.com/demos/
- http://pupunzi.com/
- http://pupunzi.open-lab.com/mb-jquery-components/mb-containerplus/
- http://www.jqueryplugins.com/

Dojo

- Модульная структура
- Dojo
- Dijit (Dojo widgets)
- Dojox(Dojo extensions)

- http://dojotoolkit.org/
- https://dojo.io/

Dojo
Пример кода обработчиков


```
require(["dojo/ready", "dojo/query"], function(ready, query){
  ready(function(){
     query("div.text p").forEach(function(node){
       node.onmouseover = function(){
            this.style.backgroundColor = "green";
       }
       node.onmouseout = function() {
            this.style.backgroundColor = "yellow";
       };
       node.onclick = function () {
            // находим элемент, куда надо поместить результат
            query("#result")[0].innerHTML = "<hr/><p>"+Date()+"</p>" +
                 "&lt;" + this.nodeName + "&gt;" + this.textContent +
                 "<p>Вывод завершен!</p><hr/>";    }
     })
  })
})
```

Основные принципы Dojo
На основе разметки

10.09.2019

```
<html><head>
<script type="text/javascript"
      src="dojoAjax/dojo.js"></script>
<script type="text/javascript">
   dojo.require("dojo.widget.HelloWorld");
   dojo.require("dojo.widget.Button");
</script>
</head><body>
   <div dojoType="HelloWorld">
      <div dojoType="Button"></div>
   </div>
</body></html>
```

```
dojo.provide("dojo.widget.HelloWorld");
dojo.require("dojo.widget.HtmlWidget");

dojo.widget.defineWidget("dojo.widget.HelloWorld", dojo.widget.HtmlWidget, {
   isContainer: true,
   templateString: '<div><div dojoAttachPoint="containerNode">'
      + '</div></div>'
});
```

Основные принципы Dojo
Динамическое создание


```
<html><head>
<script type="text/javascript" src="dojoAjax/dojo.js"></script>
<script type="text/javascript">
   dojo.require("dojo.widget.HelloWorld");
   dojo.require("dojo.widget.Button");
   dojo.addOnLoad(function(){
      var button = dojo.widget.createWidget("Button", {caption:
     "Submit"});
      dojo.widget.byId('someID').addChild(button);
   });
</script>
</head><body>
   <div id="someID" dojoType="HelloWorld">
   </div>
</body></html>
```

Dojo
Генерация интерфейса на основе данных


```
dojo.provide("dojobook.grid.grid_definitions.programmatic" );
(function() {
  dojobook.grid.grid_definitions.programmatic.structure = [
   {
    cells: [
                [
                  {name: 'Flavor' , field:"name" , width:'10em' },
                  {name: 'Base Flavor' , field:"baseFlavor" },
                  {name: 'Calories' , field:"calories" },
                  {name: 'Fat' , field:'fat' }
                ],
                [
                  {name: 'Add-Ins' , field:"mixins" ,
                  width:'15em' , colSpan:4}
                ]
            ]
    }
  ]
})();
```

Dojo Полезные материалы

10.09.2019

- http://anton.shevchuk.name/wp-demo/dojo-for-beginners/
- https://www.ibm.com/developerworks/web/tutorials/wa-dojotoolkit/index.html
- http://demos.dojotoolkit.org/demos/
- Интерактивный редактор интерфейсов Dojo: http://www.wavemaker.com/
  (ранее был доступен в виде бесплатного приложения)

Прочие
JS-средства разработки

10.09.2019

- https://angular.io/
- https://facebook.github.io/react/
- **https://vuejs.org/**
- https://www.emberjs.com/
- http://backbonejs.org/
- http://canjs.com/

- http://todomvc.com/ - см. обзор

Библиотеки, ориентированные на графический интерфейс

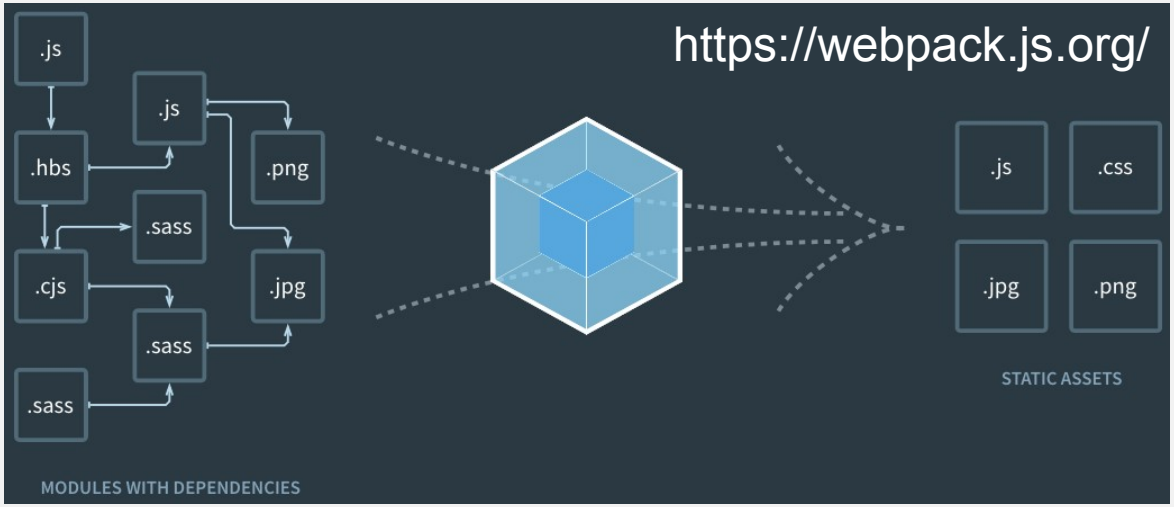
10.09.2019

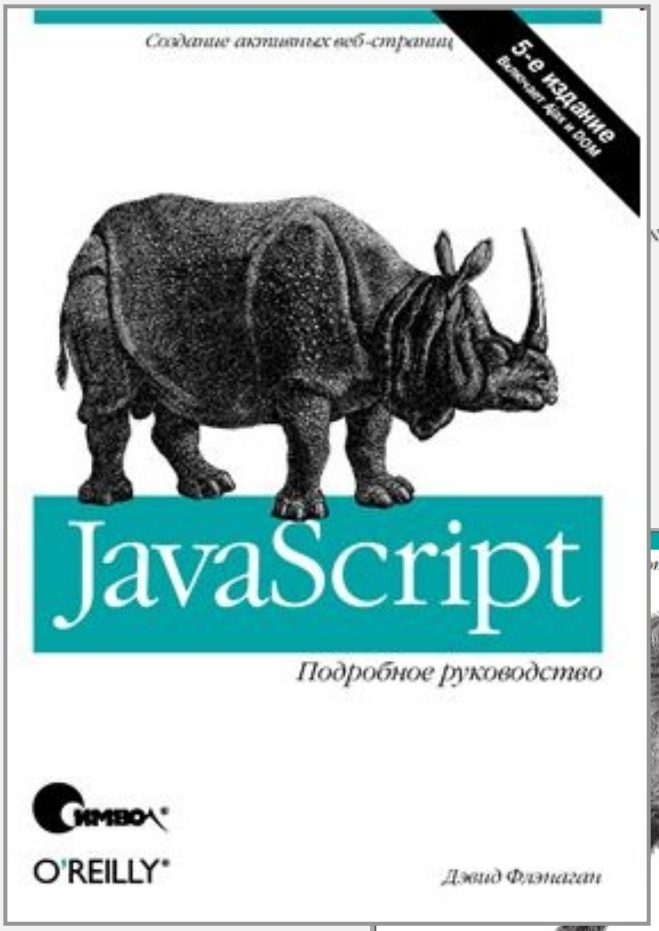
- Bootstrap - http://getbootstrap.com/
  Унификация интерфейса для различных устройств (настольные ПК, планшеты, смартфоны).
- http://foundation.zurb.com/
- http://firezenk.github.io/zimit/
- http://ink.sapo.pt/
- http://www.99lime.com/elements/
- http://getkickstart.com/
- http://purecss.io/
- http://getskeleton.com/
- ...

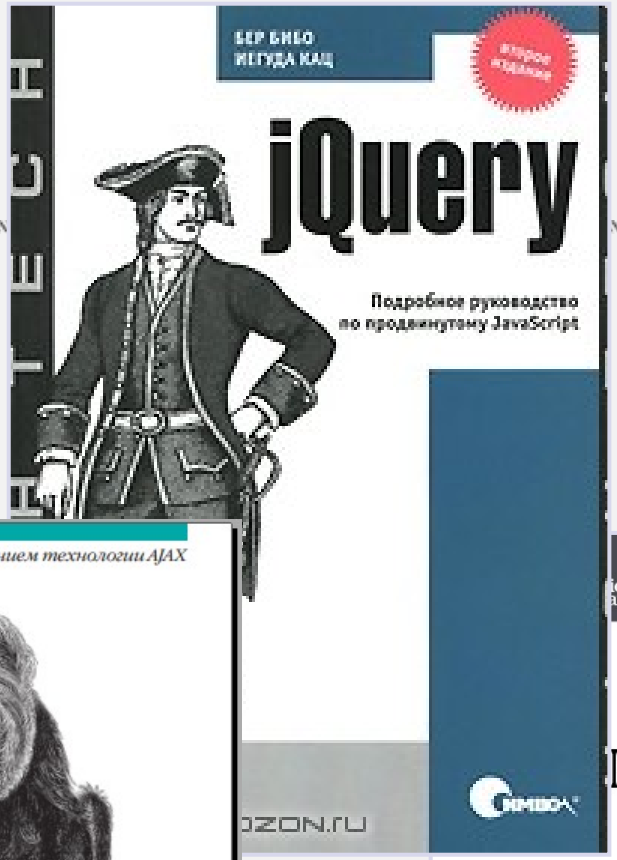
Управление пакетами и зависимостями

10.09.2019

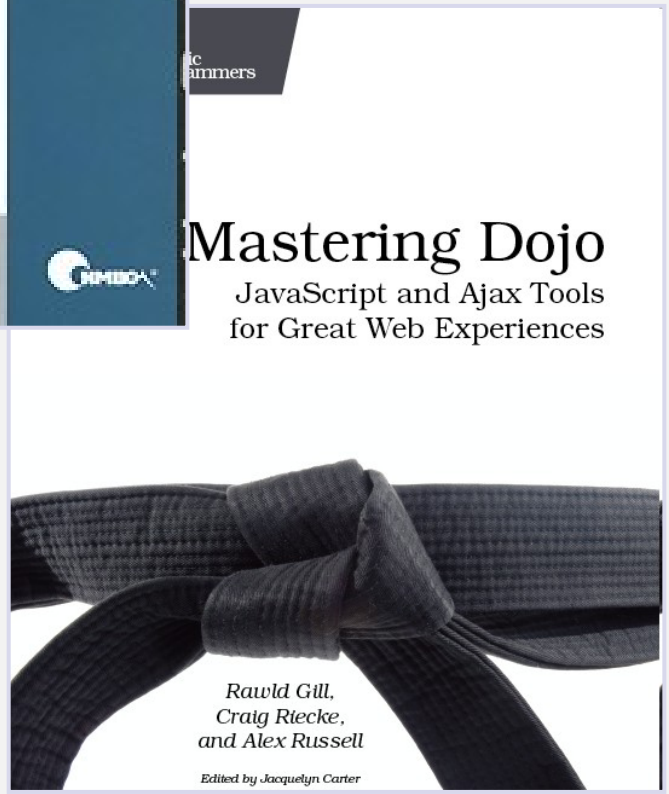
- https://bower.io/
- https://www.npmjs.com/
- https://webpack.js.org/
- https://yarnpkg.com/
- …




Создание активных веб-страниц
5-е издание
Включает Ajax и DOM
JavaScript
Подробное руководство
O'REILLY
Дэвид Флэнаган


БЕР БИБО
ИЕГУДА КАЦ
jQuery
Подробное руководство
по продвинутому JavaScript


Dojo
Подробное руководство
O'REILLY
Мэтью А. Расселл


Mastering Dojo
JavaScript and Ajax Tools
for Great Web Experiences
Rawld Gill,
Craig Riecke,
and Alex Russell
Edited by Jacquelyn Carter

10.09.2019

Литература

- *Флэнаган Д.* JavaScript. Подробное руководство. – Пер. с англ. – СПб: СимволПлюс, 2008. – 992 с., ил.
- Бер Бибо, Иегуда Кац. jQuery. Подробное руководство по продвинутому JavaScript.- Пер. с англ.– СПб: СимволПлюс, 2009. – 384 с., ил.
- Расселл М. Dojo. Подробное руководство – Пер. с англ. – СПб.: СимволПлюс, 2009. – 560 с., ил.
- Rawld Gill, Craig Riecke, Alex Russell. Mastering Dojo. JavaScript and Ajax Tools for Great Web Experiences. The Pragmatic Bookshelf, Raleigh, North Carolina Dallas, Texas, 2008.- 545 p.

- http://www.w3schools.com/js/
- https://developer.mozilla.org/ru/docs