Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана

JavaScript. Уровень 1. Основы веб - программирования - NEW

Алексей Тарасов

http://jdrupal.ru

www.specialist.ru

Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана®

Модуль 1. Основы программирования

Алексей Тарасов

http://jdrupal.ru

www.specialist.ru

Модуль 1. Основы программирования

- Введение в JavaScript
- Обзор базовых типов
- Операторы
- Выражения и инструкции
- Переменные
- Приведение типов
- Тривиальные типы
- **Практикум: Использование переменных, типов данных и операторов**

Введение в JavaScript

- Интерпретируемый язык программирования
- Не зависит от платформы
- Регистрозависимый
- Верблюжья нотация
- ECMAScript
- Unicode

Инструментарий

- Текстовый редактор (Notepad++, Eclipse, WebStorm и т.д.)
- Доступ к консоли браузера
  - Google Chrome, MS Internet Explorer (F12)
  - Mozilla FireFox(Ctrl + Shift + I)
- Вывод данных
  - console.log() //в консоль
  - document.write() // в окно браузера
  - alert() //в диалоговое окно браузера

Базовые тип Number: числа

- 0
- 23
- 8570000
- 0xff

- 2.17
- .34
- 4e45

Операторы

- 4 + 5, 10 % 3
- +, -, *, /, %
- Приоритет ()

Базовые тип String: строки

- Одинарные и двойные кавычки
  - "Привет, JavaScript!"
  - 'Привет, JavaScript!'
- Специальные символы
  - 'Привет, д\'Артаньян!'
  - 'Привет, \n\t JavaScript!'
  - 'Привет, \u2010\x20 JavaScript!'
- Склеивание (конкатенация) строк
  - "Привет " + 'JavaScript!'

Базовые тип Boolean: логический (булев)

- true
- false

- Операторы сравнения:
- <, >, <=, >=, !=, ==, !==, ===

- Сравнение строк происходит посимвольно

Логические операторы

- Логическое И (&&)
- Логическое ИЛИ (||)
- Логическое НЕ (!)

- Число 0 и пустая строка – «» превращаются в false
  - !5
  - true && “текст”
  - 0 || “Все!”

Практическая работа

- Вычислите результат выражения
- (5 && (“специалист” || !0)) && !(0 && (56 || true))

Выражения и инструкции

- Выражение – фраза JavaScript, которая может быть вычислена интерпретатором
  - "JavaScript – мощный скриптовый язык программирования"
  - true
  - 45.23
  - I
  - 3 + 5
- Инструкции JavaScript
  - 3 + 5;
  - !true;

Переменные

- Объявление и инициализация, долговременные переменные
  - var i ;
  - var day = 5 ;
  - var name = “Фаддей Фаддеевич” ;
  - var flag = true, num = 7 ;
  - firstname = “Фаддей”; //недолговременная
- Использование переменных
  - var name = “Фаддей Фаддеевич”;
  - name = name + “Беллинсгаузен”;
  - name += “Беллинсгаузен”;

Приведение типов

- В число:
  - n = Number(false);
  - n = “100” * 1;
  - n = +“100”;
  - n = parseInt(“3.14”)
  - n = parseFloat(“3.14”)
- В строку:
  - x = 100 + “” ;
  - x = String(false);
- В логический тип:
  - n = !!5;
  - n = Boolean(false);

Тривиальные типы

- undefined
- null

- Оператор typeof

Размещение JavaScript в браузере

```
<script></script>
<script type="text/javascript"></script>
<script type="text/javascript" src="myscript.js"></script>

<script>
//однострочный комментарий
var level = 1;
/*
 многострочный комментарий
*/
console.log("JavaScript " + level);
</script>
```

Практическая работа

- Проверить работу скрипта, убедиться в выводе результата в консоль

Итоги

- Введение в JavaScript
- Обзор базовых типов
- Операторы
- Выражения и инструкции
- Переменные
- Приведение типов
- Тривиальные типы

Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана®

Модуль 2. Управляющие конструкции

Алексей Тарасов

http://jdrupal.ru

Модуль 2. Управляющие конструкции

- Циклы
- Цикл while
- Операторы инкремента и декремента
- Цикл for
- Цикл do while
- Управляющие конструкции if – else if – else
- Метки
- Прерывание и продолжение цикла
- Управляющая конструкция switch
- **Практикумы: Практическое применение управляющих конструкций в JavaScript**

Управляющие конструкции if, if – else

```
if (выражение)
        инструкция;

if (выражение)
        инструкция1;
else
        инструкция2;
```

```
if (day == 1){
        //выполнение блока кода 1;
}
else if (day == 2){
        //выполнение блока кода 2;
}
else if (day == 3){
        //выполнение блока кода 3;
}
else{
        /*если не выполнены
предыдущие условия, выполняем
блок кода 4*/;
}
```

Тернарные оператор

```
if (выражение)
        если true ;
else
        если false;
```

- (выражение) ? если true : если false;

Практическая работа

- У Вас есть интернет-магазин. В корзине отображается количество товаров, добавленных покупателем. Используя оператор **if**, напишите программу, выводящую правильно слово «товаров» для любого числа в корзине.
- Товар, товара, товаров

Инструкция switch

```
switch (выражение){
 инструкции
}
```

Инструкция switch

```
switch (day){
  case 1:  //выполнение при day == 1;
   //выполнение блока кода 1;
  break; //останов
 case 2:  //выполнение при day == 2;
   //выполнение блока кода 2;
  break; //останов
 case 3:  //выполнение при day == 3;
   //выполнение блока кода 3;
  break; //останов
 default:  //ничего из предыдущего не подходит;
   //выполнение блока кода 1;
  break; //останов
}
```

Практическая работа

- Используя предыдущую практическую работу, выполните задание с использованием операторов **if** и **switch**

Операторы инкремента ++ и декремента --

- ++, --
- Постинкремент (сначала используем, потом увеличиваем)
  - var i = 5;
  - console.log(i++); //выведено 5, хранит 6
- Преинкремент (сначала увеличиваем, потом используем)
  - var i = 5;
  - console.log(++i); //выведено 6, хранит 6

- Предекремент, постдекремент

Циклы

- Инициализация "счетчика"
- Проверка условия
- Выполнение инструкций
- Изменение "счетчика "

Инструкция while

```
while (выражение){
 инструкция1;
 инструкция2;
…
}
```

Практическая работа

- Используя инструкцию **while**, выведите все числа от 1 до 20
- Выведите все нечетные числа от 1 до 20
- Выведите квадраты всех чисел от 1 до 20

Практическая работа

- Используя инструкцию **while**, выведите таблицу умножения для числа 8
- Например,
  - 8 * 1 = 8
  - 8 * 2 = 16
  - и т.д.

Инструкция for

```
for (инициализация; условие; инкремент)
        инструкция;


for (инициализация; условие; инкремент){
        инструкция1;
        инструкция2;
        …
}
```

Практическая работа

- Используя инструкцию **for**, выведите все числа от 1 до 20
- Используя инструкцию **for**, создайте выпадающий список с числами от 1 до 20

Практическая работа (если будет время)

- Построить таблицу умножения

Инструкция do while

```
do{
 инструкция1;
 инструкция2;
…
} while (выражение);
```

Метки

```
outer: for (инициализация; условие; инкремент){
   inner: while (условие){
            while-инструкция;
             if (условие 1) break inner;
            while-инструкция; }
   for-инструкция;
   }
```

Прерывание и продолжение цикла

```
for (инициализация; условие; инкремент)
{
  инструкция;
  if (условие 1) break;
  инструкция;
  if (условие 2) continue;
  инструкция;
}
```

Практикум

- Практическое применение управляющих конструкций в JavaScript

Итоги

- Управляющие конструкции if – else if – else
- Управляющая конструкция switch
- Циклы
- Цикл while
- Операторы инкремента и декремента
- Цикл for
- Цикл do while
- Метки
- Прерывание и продолжение цикла

Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана®

Модуль 3. Функции

Алексей Тарасов

http://jdrupal.ru

Модуль 3. Функции

- Понятие функций
- Возврат значений
- Области видимости
- Анонимная функция
- Замыкания
- Рекурсия
- **Практикумы: Использование функций при процедурном подходе в программировании**

Понятие функций

- Функция – блок программного кода, который может многократно вызываться

```
function hi(){
 console.log(“Привет, JavaScript!”)
}
hi();


function hi(name){
 console.log(“Привет, “ + name +”!”)
}
hi(“Михаил”);
```

Практическая работа

- Написать функцию, которая будет являться оберткой для вывода в консоль console.log()

Практическая работа

- Написать функцию ipoteka, для подсчета аннуитетных ипотечных платежей
- Функция принимает следующие параметры
  - S – величина кредита
  - p – процентная ставка за год
  - n – количество лет, на которые берется кредит
- Обратите внимание на **p** и **n**!

Возврат значений

```
function sum(x, y) {
 return x + y;
}
result = sum(10, 20);
console.log(result);

console.log(sum(2, 3));
```

Практическая работа

- Исправьте код предыдущей практической так, чтобы функция возвращала результат

Области видимости

```
var x = "x-global";
var y = "y-global";
function outerFunction() {
        var x = "x-local";
        console.log(x); //x-local
        function innerFunction(z) {
                console.log(x); //x-local
                console.log(y); //y-global
                console.log(z); //z-local
                y = "y-local";
         }
        innerFunction('z-local');
 }
 outerFunction();
console.log(x); //x-global
console.log(y); //y-local
```

Функциональный литерал и анонимная функция

```
//инструкция function
function hi(name){ console.log(“Привет, “ + name +”!”)}

//функциональный литерал
var hi = function (){ console.log(“Привет, “ + name +”!”)}

hi(‘AJAX’);
```

Анонимная функция

Замыкания

```
var x = "глобальная", my;
function f() {
        var x = "локальная";
        function g() {
                console.log(x);
        }
        return g;
}
my = f();//или f()()
my();
```

Практическая работа

- Напишите функцию show(), которая принимает название любого HTML-элемента, состоящего из двух тегов, и возвращает анонимную функцию.
- Анонимная функция должна принимать строковое значение txt и выводить результат в виде текста, заключенного в соответствующие теги

Рекурсия

```
function factorial(n)  {
if (n != 1)
        return n*factorial(n - 1);
else
        return 1;
}
factorial(5);
```

Практикум

- Выведите числа от 1 до 20 при помощи рекурсии

Работа с аргументами функции

```
function hi(name){
 console.log(“Привет, “ + name +”!”)
}
hi(“Node!”)
```

```
function hi(){
 console.log(“Привет, “ + ? +”!”)
}
hi(“Node!”)
```

- arguments
- arguments.length //1
- arguments[0] // ‘Node!’
- arguments.callee // hi

Практикум

- Написать функцию **sum**, принимающую неограниченное число аргументов и возвращающую их сумму

Итоги

- Понятие функций
- Возврат значений
- Области видимости
- Анонимная функция
- Замыкания
- Рекурсия

Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана

Модуль 4. Объекты и массивы

Алексей Тарасов

http://jdrupal.ru

Модуль 4. Объекты и массивы

- Объектный тип: Объект (Object)
- Свойства объекта
- Методы объекта
- Методы функций
- Объектный тип: Массив (Array)
- Свойство и методы массива
- Встроенный объект Math
- **Практикумы: Использование объектных типов в JavaScript**

Объектный тип: Объект (Object)

```
var book1 = { };

book1.title = ‘JavaScript Подробное руководство';
book1.pubyear = 2012;
book1.price = 1000;

console.log(book1.title);
```

Свойства объекта

```
var book1 = {
    title: ‘JavaScript Подробное руководство‘,
    pubyear: 2012,
    price: 1000
};

book1.pubyear = 2013;
```

Свойства объекта

```
var book1 = {
    ‘title book‘: ‘JavaScript Подробное руководство‘,
    pubyear: 2012,
    ‘price’: 1000
};

book1[‘title book‘] = ‘Изучаем JavaScript ’;
console.log(book1[‘price’]);
```

Проверка наличия свойств

- var book = {price: 2500};
- console.log(‘price' in book);
- delete book.price

Проход по свойствам объекта

```
var o = {0:'Guest', 1:0, 2:false};
for (var cur = 0; cur in o; cur++){
  console.log(cur + ": " + o[cur]);
}

var o = {name:'Guest', age:0, 'var':false};
for (var cur in o){
  console.log(cur + ": " + o[cur]);
}
```

Практическая работа

- Создайте два объекта **car1** и **car2** со свойствами **скорость** и **масса**

Сравнение и передача объектов

- Объекты копируются по ссылке!

Методы объекта

```
var book1 = {
    title: ‘JavaScript Подробное руководство‘,
    pubyear: 2012,
    price: 1000,

    show0: function (){console.log(book1.title)},
    show: function (){console.log(this.title)}
};
book1.show0();
book1.show();
```

this

```
var book1 = {
   title: ‘JavaScript Подробное руководство‘,
   …
   show: function (){console.log(this.title)}
   …
};
```

Практическая работа

- Для объектов **car1** и **car2** напишите методы, выводящие их свойства

Методы функций

```
function sum(x, y){
  return this.num + x + y;
}
var obj = {num: 10};

obj.m = sum;
obj.m(20, 30);
 delete obj.m;

sum.apply(obj, [20, 30]);
sum.call(obj, 10, 20);
```

Объектный тип Array: Массив

- var a = [];
- var x = 'moon';
- var y = function(){ console.log('Hello'); };

- var a = [10, "sun", x, y, true];
- a[1];
- a[3]();
- a[9] = 100;

Длина массива

- var a = [1, 5];
- console.log(a.length); // 2

- a[23] = 11;
- console.log(a.length); // 24

- var a = [1, 5];
- a.length = 3; // [1, 5, undefined]

Проход по элементам массива

```
var a = [5, 'abc', 73]; a[99] = 1;

for (var i=0, x=0; i<a.length; i++) {
  x++;
  }

var x = 0;
for (var i in a) {
  x++;
}
```

Практическая работа

- Числа Фибоначчи
- Задайте переменную N с произвольным целочисленным значением – сколько чисел считаем
- Создайте пустой массив fibs
- Присвойте первым двум элементам массива значение 1
- В цикле заполните массив до N элементов следуя формуле:
  - Каждое последующее число равно сумме двух предыдущих чисел

Удаление элементов массива

- var x = 0;
- var a = [4, 12, 7];
- a[1] = undefined;

- for (var i in a) { x++; }
- var x = 0; var a = [4, 12, 7];
- delete a[1];
- for (var i in a) { x++; }

Получение из массива строки

- var a = [1, 5, 7];
- var s = a.toString(); // 1,5,7
- var s = a.join(); // 1,5,7
- var s = a.join('---'); // 1---5---7

Практическая работа

- Из произвольного массива создать ненумерованный список наиболее быстрым способом

Склеивание массивов

- var a = [1, 5];
- var b = [11, 8];
- var arr = a.concat(3, b); // [1, 5, 3, 11, 8]

Получение части массива

- var a = [1, 5, 7, 12, 9];
- var arr = a.slice(2); // [7, 12, 9]
- var arr = a.slice(1, 3); // [5, 7]
- var arr = a.slice(-3, -1); // [7, 12]
- var arr = a.slice(2, 1); // []

Сортировка массива

- var a = [14, 51, 7, 2];
- a.reverse();

- var a = [14, 51, 7, 2];
- a.sort(); // [14, 2, 51, 7] !?

- function mySort(a, b) {return a–b;}
- a.sort(mySort); // [2, 7, 14, 51]

Практическая работа (если будет время)

- написать функцию сортировки массива на четные и нечетные числа

Практическая работа* (если будет время)

- Задачу формулирует преподаватель

Работа с концом массива

- Исходный массив
- var a = [5, 'abc', 73];

- Извлечение элемента
- var v = a.pop(); // [5, 'abc'], v = 73

- Добавление элементов
- var v = a.push(12, 3); //[5, 'abc', 12, 3], v = 4

Работа с началом массива

- Исходный массив
- var a = [5, 'abc', 73];

- Извлечение элемента
- var v = a.shift(); // ['abc', 73], v = 5

- Добавление элементов
- var v = a.unshift(12, 3); //[12, 3, 'abc', 73], v = 4

Вставка и удаление элементов из любого места массива

- Исходный массив
- var a = [5, 'abc', 73, 12, 8];

- var arr = a.splice(1, 2); // [5, 12, 8]
- // arr = ['abc', 73]

- var arr = a.splice(1, 0, 3); // [5, 3, 'abc', 73, 12, 8]
- // arr = []

Встроенный объект Math

- Свойства
  - E, LN10, LN2, LOG10E, LOG2E, PI, SQRT1_2, SQRT2
- Методы
  - sin(), asin(), cos(), acos(), tan(), atan(), atan2()
  - ceil(), floor(), round(), abs(), max(), min(), random()
  - exp(), log(), pow(), sqrt()

- console.log(Math.PI);
- console.log(Math.pow(4, 2));

Практическая работа

- Исходный код:

```
var a = [5, 12];
var b = [];
a[99] = 7;
```

- Записать в массив b квадраты значений массива a:
  - с помощью цикла for
  - с помощью цикла for/in

Использование массивов

- Полезные функции
- Декомпозиция

Итоги

- Объектный тип: Объект (Object)
- Свойства объекта
- Методы объекта
- Методы функций
- Объектный тип: Массив (Array)
- Свойство и методы массива
- Встроенный объект Math

Центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана

Модуль 5. Использование объектов JavaScript

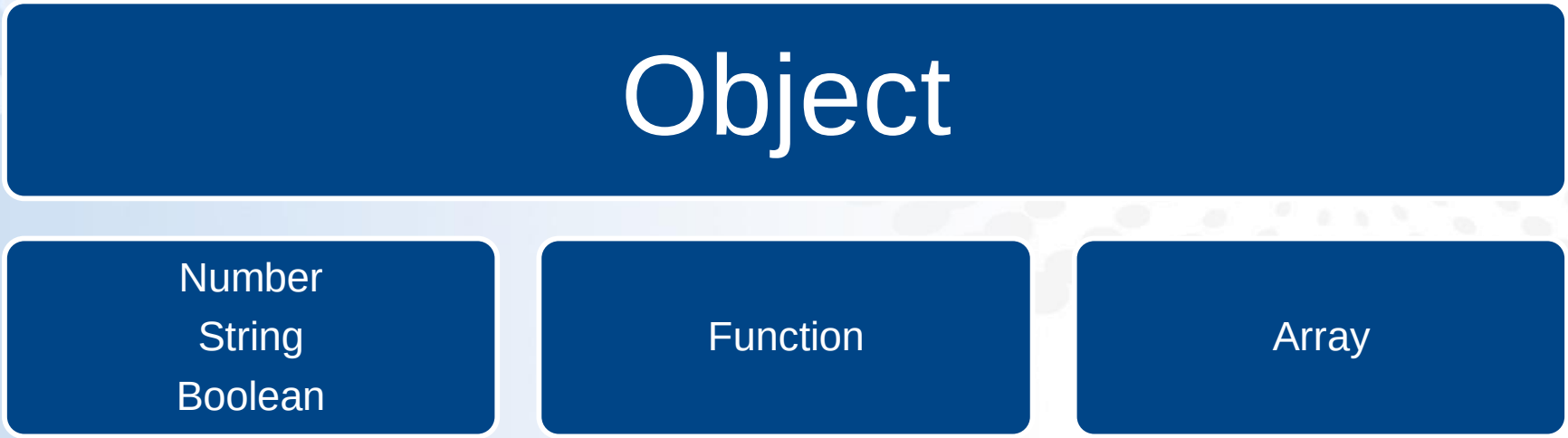
Алексей Тарасов

http://jdrupal.ru

Модуль 5. Использование объектов JavaScript

- Свойства и методы объекта Number
- Свойства и методы глобального объекта
- Свойство и методы объекта String
- Использование регулярных выражений
- **Практикум: Использование базовых типов как объектов с регулярными выражениями**

Что мы имеем?



Объект Number

- Статические свойства
  - Number.NEGATIVE_INFINITY
  - Number.POSITIVE_INFINITY
  - Number.NaN
  - Number.MIN_VALUE
  - Number.MAX_VALUE

Преобразование числа в строку

- toString()
- var n = 12345.6789;
- n.toFixed(2); //'12345.68'
- n.toExponential(2); //'1.23e+4'
- n.toPrecision(6); // '12345.7'

- n.toPrecision(4); //1.235e+4

Свойства и методы глобального объекта

- Infinity
- –Infinity
- NaN

- NaN != NaN

- isNaN(3 * 'a'); // true
- isFinite(3 * 'a'); // false
- isFinite(3 / 0); // false

Преобразование строки в число

- var s = '37.5 км';
- var n = parseFloat(s); // 37.5
- var n = parseInt(s); // 37
- var s = '$99.9';
- var n = parseFloat(s); // NaN
- var n = parseInt(s); // NaN

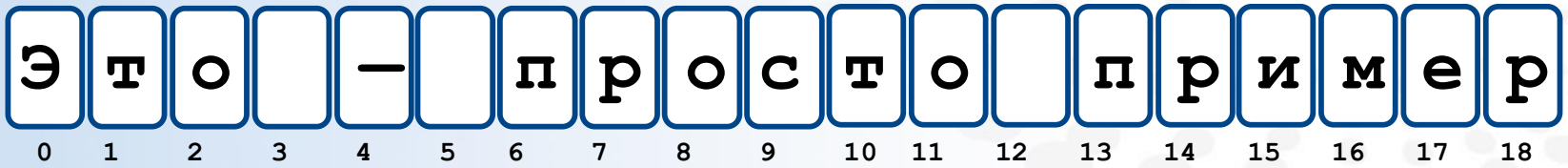
Использование систем счисления

- var n = 255;
- var s = n.toString(16); // ff

- var s = 'ff';
- var n = parseInt(s, 16); // 25

Длина строки

- var str = 'Это \u2014\u0020просто\x20пример';



- console.log(str.length);

Склеивание строк

- var s1 = 'Это';
- var s2 = ' \u2014\x20';
- var s3 = 'просто пример';

- var s = s1 + s2 + s3;
- var s = s1.concat(s2, s3);

Регистр символов

- var x = s.toLowerCase();
- var x = s.toUpperCase()

Получение символа из строки

- var s = 'просто пример';
- var x = s.charAt(4); // 'т'
- var x = s.charAt(40); // ''

- var x = s.charCodeAt(4);//1090 (U+0442)
- var x = s.charCodeAt(40); // NaN

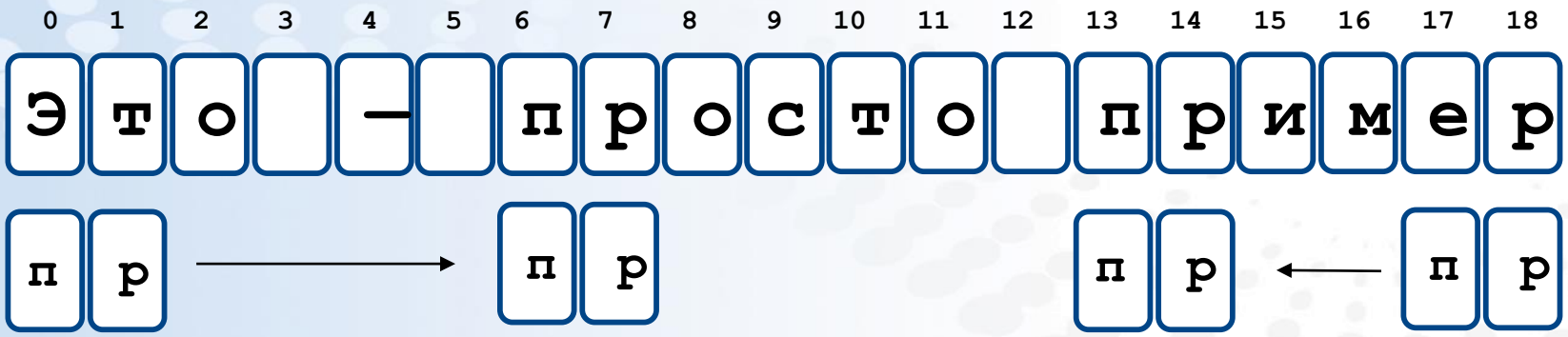
- Создание строки из кодов символов
- var x = String.fromCharCode(1051, 1091, 1085, 1072); // Луна

Получение части строки

- var s = 'просто пример';
- var x = s.slice(3, 6); // сто
- var x = s.substring(3, 6); // сто
- var x = s.slice(6); // пример
- var x = s.substring(6); // пример

- var x = s.slice(); // просто пример
- var x = s.substring(); // просто пример
- var x = s.slice(6, 3); // ""
- var x = s.substring(6, 3); // сто
- var x = s.slice(-6, -3); // при

Поиск по строке



- var x = s.indexOf('пр'); // 6
- var x = s.indexOf('пр', 7); // 13
- var x = s.lastIndexOf('пр'); // 13

Практическая работа

- Исходные данные:
- var s = 'Мы знаем, что монохромный цвет – это градации серого';
- var txt = 'хром';
- var word;
- В значении переменной s найдите слово, содержащее подстроку, которая является значением переменной txt и присвойте его переменной word

Замена в строке

- var s = 'Это просто пример';
- var x = s.replace('просто', 'сложный');
- // 'Это сложный пример'

Разбиение строки

- var s = 'Это просто пример';
- var x = s.split(' ');
- // [Это, просто, пример]
- var x = s.split(' ', 2);
- // [Это, просто]

Практическая работа

- Опишите функцию substrCount(), которая возвращает количество вхождений подстроки
- Функция принимает следующие аргументы:
- input – строка, в которой ведется поиск
- needle– искомая подстрока
- offset – смещение начала отсчета
- length –максимальная длина строки в которой будет производится поиск подстроки после указанного смещения.
- substrCount('Good Golly Miss Molly', 'll', 7, 10)

Практическая работа

- Опишите функцию strPad(), которая дополняет строку другой строкой до заданной длины
- Функция принимает следующие аргументы:
  - input – входная строка
  - fullLen–длина конечной строки
  - fillStr – дополняющая строка
  - fillType– 'FILL_RIGHT', 'FILL_LEFT' или 'FILL_BOTH'.

- strPad('star', 10, '-*-') //star-*--*-
- strPad('star', 8, '-*-', 'FILL_LEFT') //-*--star
- strPad('star', 8, '*', 'FILL_BOTH') //**star**

Регулярные выражения

- var re = /regexp/флаги;
- Методы объекта String
  - replace(regexp, string);
  - split(regexp)
  - search(regexp)
  - match(regexp)
- Методы объекта RegExp
  - test(string);
  - exec(string)

Базовое использование

- var pattern = /@/;
- var str = "vasya@gmail.com";
- str.search(pattern); // 5 indexOf()

- var re = /gmail|hotmail/;
- str.search(pattern);
- "folder/file".search(/\//); // 6
- /\//.test("folder/file"); // true

Специальные последовательности

- \0 Символ NULL (\u0000)
- \t Горизонтальная табуляция (\u0009)
- \n Перевод строки (\u000A)
- \r Возврат каретки (\u000D)
- \\ Обратная косая черта (\u005C)
- \xXX Символ Latin-1, заданный двумя шестнадцатеричными цифрами
- \uXXXX Символ Unicode, заданный четырьмя шестнадцатеричными цифрами

- ^ $. * + ? = ! : | \ / () [] {

Классы символов

- var re = /[abc]/;
- var re = /[^abc]/;
- var re = /[a-f]/;
- . Любой символ, кроме перевода строки
- \w Любой символ ASCII [a-zA-Z0-9_]
- \W Противоположное \w [^a-zA-Z0-9_]
- \d Любая цифра ASCII [0-9]
- \D Противоположное \d [^0-9]
- \s Любой символ-разделитель Unicode
- \S Противоположное \s

Повторения

- {n, m} Шаблон повторяется не менее n, но и не более m раз
- {n,} {,n} Шаблон повторяется не менее или не более n раз
- {n} Шаблон повторяется точно n раз
- ? Эквивалентно {0, 1}
- + Эквивалентно {1,}
- * Эквивалентно {0,}

Практическая работа

- Составить регулярное выражение, которое соответствует следующим форматам даты:
- 25-02-2012
- 25-2-2012
- 01-12-2011
- 2-12-1978
- Для проверки использовать метод test()

Позиция соответствия

- ^ Поиск с начала строки
- $ Поиск до конца строки
- \b Позиция между символом ASCII и не символом ASCII (граница слова)
- \B Позиция между двумя символами ASCII (не граница слова)
- var re = /^abc/;
- var re = /abc$/;
- var re = /\bjava\b/;
- var re = /\bjava\B/;

Флаги

- ignoreCase
  - var r = /a/i; //ищутся символы 'a' и 'A'
- global
  - var r = /a/g; //ищутся все символы 'a'
- multiline
  - var r = /a/gim

Группировка и ссылки

- var r = /ab(cd)?/;
- Ищем содержимое в одинаковых кавычках
  - var r = /['"][^'"]*['"]/; //проблема
  - var r = /(['"])[^'"]*\1/; //ссылка \1
- Внешние ссылки
  - '1A'.replace/(\d+)([a-z]+)/i, '$2-$1')

Практическая работа

- В переменной names содержится строка:
  - "Lennon, John\nMcCartney, Paul\nHarrison, George\nStar, Ringo"
- Задача: получить строку вида:
  - John Lennon
  - Paul McCartney
  - George Harrison
  - Ringo Star

Методы match() и exec()

- var re = /(ab)(cd)(xyz)/;
- var res = 'abcdxyz'.match(re);
- Результат (переменная res)
  - ['abcdxyz', // индекс 0
  - 'ab',
  - 'cd',
  - 'xyz']
- /(ab)(cd)(xyz)/.exec('abcdxyz');
- ['abcdxyz', 'ab', 'cd', 'xyz‘]

Жадные квантификаторы

- + и *
  - var re = /a+/;
- Таблетка от жадности
  - var re = /a+?/

Итоги

- Свойства и методы объекта Number
- Свойства и методы глобального объекта
- Свойство и методы объекта String
- Использование регулярных выражений
- **Практикум: Использование базовых типов как объектов с регулярными выражениями**

Учебный центр компьютерного обучения «СПЕЦИАЛИСТ» при МГТУ им. Н.Э.Баумана

Модуль 6. Объектно-ориентированное программирование

Алексей Тарасов

http://jdrupal.ru

Модуль 6. Объектно-ориентированное программирование

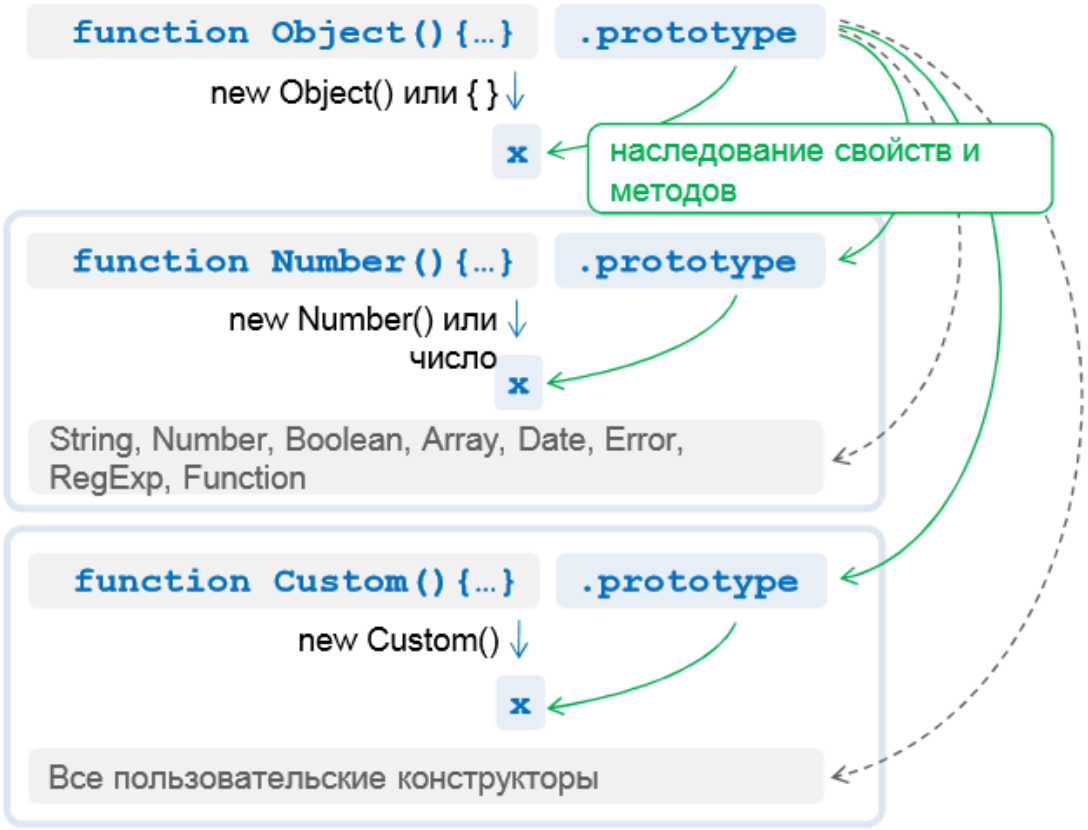
- Конструкторы объектов
- Прототипы
- Методы объекта
- Объект Date
- Объект Error
- **Практикум: Применение объектно-ориентированного подхода в программировании**

Конструкторы

```
function Point(x, y) {
  this.x = x;
  this.y = y;
  this.getCoords = function(){ return [this.x, this.y]; };
}
```

- var start = new Point(10, 20);
- var coords = start.getCoords();
- start.constructor;

Как всё происходит?



Новый взгляд на привычное

- var n = new Number(5);
- var s = new String('John');
- var b = new Boolean(true);

- var a = new Array(5);
- var o = new Object();

- var re = new RegExp('[a-z+]','i');

Проверка конструктора

- var myObject = [];
- myObject instanceof Array;
- var x = new Number(5); x instanceof Object; // true
- x++; x instanceof Object; // false

Прототипы

- Point.prototype.add = function(obj) {
- return new Point(this.x + obj.x, this.y + obj.y);
- };
- var start = new Point(10, 10);
- var finish = new Point(20, 20);
- var line = start.add(finish);
- var line = new Point(10, 10).add(new Point(20, 20));
- line.constructor.prototype.draw = ...

Методы объекты

- var obj = {x: 10};
- obj.valueOf();
- obj.hasOwnProperty('x');
- obj.propertyIsEnumerable('x');
- Object.prototype.isPrototypeOf(obj)

Практическая работа

- Создайте конструктор Book со свойствами:
  - title
  - pubYear
  - price
- Создайте объект типа Book передав в конструктор произвольные значения
- Добавьте в прототип конструктора Book метод show, который выводит значения свойств title и price
- Вызовите метод show созданного объект

Практическая работа

- Опишите конструктор Dictionary, который принимает в качестве аргумента объект вида:
  - {'ключ':'значение', 'ключ':'значение', …}
- Опишите метод get(name), который возвращает значение по ключу
- Опишите метод set(name, value), который сохраняет новую пару "ключ: значение"
- Использование:
  - var users = new Dictionary({'john':'admin'});
  - var johnRole = users.get('john');
  - users.set('mike', 'manager')

Объект Date

- Время хранится в миллисекундах
- Точка отсчета –1 января 1970 года (00:00:00)
- GreenwichMeanTime (GMT)
  - Дата и время указывается в соответствии с местным часовым поясом
  - Указывается смещение относительно Гринвичского меридиана
  - Смещение зависит от переходов на летнее и зимнее время
- UniversalTimeCoordinated(UTC)
  - Дата и время в любой точке планеты одинаково
  - Точка отсчета совпадает с точностью до долей секунды с точкой отсчета GMT
  - Никаких переходов на летнее и зимнее время в UTC нет

Создание объекта Date

- var d = new Date();
- var d = new Date('Jan 01 2010 01:00:00');
- var d = new Date(1234567890000);
- var d = new Date(2011, 5);год, номер месяца, дата, часы, минуты, секунды, миллисекунды
- var d = new Date(2011, 5, 21);
- var d = new Date(2011, 12, 40);
- var d = new Date(2011, -1)

Методы объекта Date

- getFullYear()
- getUTCFullYear()
- getMonth() 0-11
- getDate() 1-31
- getHours() 0-23
- getMinutes() 0-59
- getSeconds() 0-59
- getMilliseconds() 0-999
- getDay() 0-6
- 0 - Воскресенье

Дополнительные методы

- getTimezoneOffset() смещение GMT в минутах
- Количество миллисекунд с 01.01.1970
  - getTime(), valueOf()
  - var n = Date.parse('Feb 09 2012 03:45:15'); var d = new Date(n);
  - var n = Date.UTC(2012, 5)

Строчное представление даты

- toString() (GMT)
- toLocaleString() (GMT)
- toUTCString()
- toTimeString()
- toLocaleTimeString()
- toDateString()
- toLocaleDateString()

Запись информации

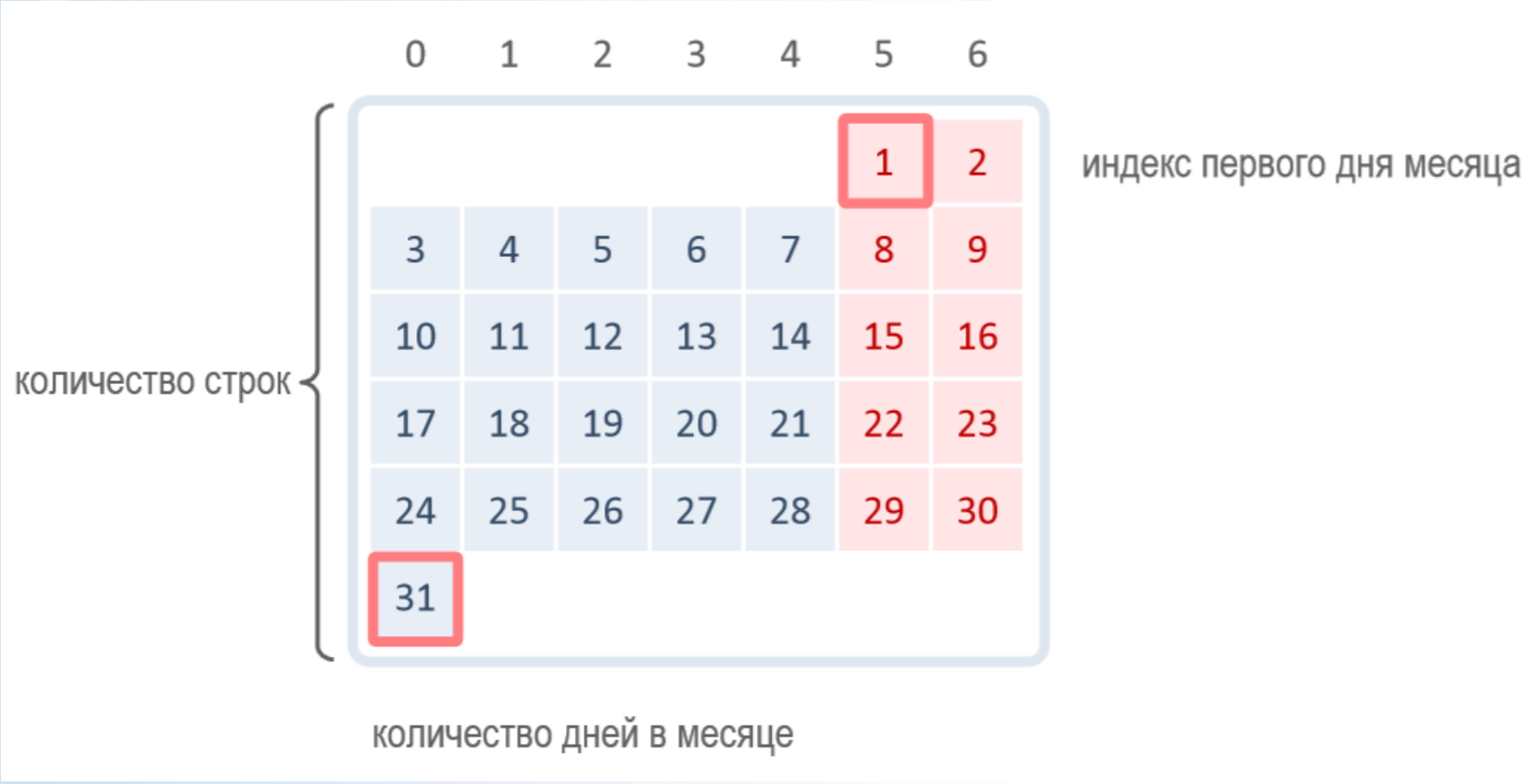
- setFullYear()
- setUTCFullYear()
- setMonth()
- setDate()
- setHours()
- setMinutes()
- setSeconds()
- setMilliseconds()
- setTime() не имеет UTC

Практическая работа

- Опишите функцию , которая возвращает количество секунд до любой даты следующего года

Практическая работа (если будет время)

- Постройте календарь



Объект Error

```
try{
  console.log(x);
}catch(e){
   console.log(e.name);
   console.log(e.message);
}
```

- console.log('test');

Конструктор объекта Error

```
try{
        var a = 1, b = 0, x;
        if(b == 0) throw new Error('Ошибка!');
        x = a / b;
}catch(e){
        console.log(e.message);
}
```

Практическая работа

- Сгенерировать ошибку при получении ошибки в функции

Итоги

- Конструкторы объектов
- Прототипы
- Методы объекта
- Объект Date
- Объект Error
- **Практикум: Применение объектно-ориентированного подхода в программировании**

Вопросы?

- Можно задавать
  - на бесплатных семинарах
  - на форуме сайта http://specialist.ru
  - в группе поддержки в ВК: http://vk.com/jsspec