**Работа с датами в JavaScript**

*В этом уроке мы с вами разберем работу с датами в JavaScript.*

**Объект Date**

Работа с **датами** в JavaScript ведется в основном через объект **Date**. Этот объект создается следующим образом:

var date **=** new Date();

Теперь переменная **date** - это **объект** с датой, который хранит в себе *текущий момент времени* (секунду, минуту, час и так далее). С помощью специальных функций мы можем получить нужные нам характеристики времени, например, текущий час, текущий день или текущий месяц.

К примеру, текущий час можно получить так: *date.getHours()*, а текущий месяц - вот так *date.getMonth()*. Смотрите все варианты:

var date **=** new Date();

alert(date.getMilliseconds()); //миллисекунды

alert(date.getSeconds()); //секунды

alert(date.getMinutes()); //минуты

alert(date.getHours()); //часы

alert(date.getDate()); //дни

alert(date.getMonth()); //месяцы

alert(date.getFullYear()); //год

Учтите, что **месяц**, который возвращает функция **getMonth**, **начинается с нуля** - январь нулевой, февраль первый и так далее.

**Работа с getDay**

С помощью объекта **Date** можно также получить номер текущего **дня недели**. Делается это с помощью метода **getDay** (он возвращает числа от 0 до 6-ти), причем неделя начинается с **воскресенья** и этот день имеет номер **ноль**. Понедельник - это день номер **1**, вторник - номер **2** и так далее.

Как работает этот метод, посмотрим на таком примере: пусть **сегодня вторник**, тогда следующий код выведет на экран число **2**:

var date **=** new Date();

alert(date.getDay());

Давайте теперь выведем текущий день недели не числом, а **словом** (для краткости воскресенье будет 'вс', понедельник 'пн' и так далее).

Для этого я составлю массив дней недели **days** и начну его с воскресенья (так как это нулевой день):

var days **=** ['вс', 'пн', 'вт', 'ср', 'чт', 'пт', 'сб'];

Как с помощью этого массива вывести на экран, к примеру, 'вт'? Нужно **передать** в квадратных скобках номер этого дня (вторник имеет номер 2):

var days **=** ['вс', 'пн', 'вт', 'ср', 'чт', 'пт', 'сб'];

alert(days[2]); //выведет 'вт'

Однако, мы не хотим передавать день недели вручную, а хотим, чтобы JavaScript **сам определил** текущий день и вывел его название. Для этого нужно **совместить** то, что возвращает метод **getDay**, и наш массив **days**:

var date **=** new Date();

var day **=** date.getDay();

var days **=** ['вс', 'пн', 'вт', 'ср', 'чт', 'пт', 'сб'];

alert(days[day]);

**Задаем определенный момент времени**

Когда мы с вами создавали объект **Date()** (таким образом: **new Date()**) мы оставляли круглые скобки **пустыми**. В этом случае мы получали **текущий** момент времени.

Однако, можно поступить и по-другому - передать ему параметры в формате *new Date(год, месяц, день, часы, минуты, секунды, миллисекунды)* и в этом случае в переменную **date** запишется не текущий момент времени, а тот, который мы указали в параметрах

Особенности такого формата: отсчет месяцев начинается с нуля, отсутствующие параметры, начиная с часов считаются равными нулю, а для года, месяцев и дней – единице.

Давайте узнаем день недели (словом) за определенную дату:

var date **=** new Date(2025, 10, 5, 12, 59, 59);

var day **=** date.getDay();

var days **=** ['вс', 'пн', 'вт', 'ср', 'чт', 'пт', 'сб'];

alert(days[day]);

Часы, минуты и секунды можно и не указывать - это не на что не повлияет:

var date **=** new Date(2025, 10, 5);

var day **=** date.getDay();

var days **=** ['вс', 'пн', 'вт', 'ср', 'чт', 'пт', 'сб'];

alert(days[day]);

**Работа с getTime**

При работе с датой существует специальный формат **timestamp**, который в JavaScript показывает количество миллисекунд, прошедшее с 1-го января 1970 года по текущий (или заданный) момент времени.

Существует специальный метод **getTime**, с помощью которого можно получить время в формате **timestamp**.

Получим текущий момент времени в этом формате:

var date **=** new Date();

alert(date.getTime());

Получим заданный момент времени в этом формате:

var date **=** new Date(2015, 12, 4, 23, 59, 59);

alert(date.getTime());

Получим разницу в миллисекундах между текущим и заданным моментом времени:

var now **=** new Date();

var date **=** new Date(2015, 12, 4, 23, 59, 59);

var diff **=** now.getTime() **-** date.getTime();

alert(diff);

**Работа с Date.parse**

Функция **Date.parse** принимает дату в формате *'год-месяц-деньTчасы:минуты:секунды'* и возвращает соответствующей ей timestamp (количество миллисекунд, прошедших с полуночи 1 января 1970 года до заданного момента времени).

Давайте выведем количество миллисекунд, которые прошли с 1 января 1970 года до 25 мая 2015, 12:59:59:

document.write(Date.parse('2015-05-25T12:59:59'));

Результат выполнения кода:

1432547999000

**Разница между датами**

Даты, представленные в виде объекта Date, можно вычитать друг из друга, результат их вычитания – разница в миллисекундах.

Давайте выведем количество миллисекунд, которые прошли с 25 мая 2015, 12:59:59 по настоящий момент времени:

var now **=** new Date();

var date **=** new Date(2015, 23, 4, 12, 59, 59);

var diff **=** now **-** date;

document.write(diff);

**Новые дополнительные видеоуроки**

Ниже расположены уроки, которые раскрывают дополнительные нюансы. Смотреть обязательно!

Продвинутая работа с new Date <https://youtu.be/17vIekE6pfs>

Сравнение и разность дат <https://youtu.be/WlR2g8DLyzI>

Нахождение последнего дня месяца и определение високосного года на JavaScript<https://youtu.be/s2LFyE3skhg>

Проверка корректности даты на JavaScript <https://youtu.be/_ymWJ9pTMcM>

Определение знака зодиака по дате <https://youtu.be/nZ2Pj_FGpnw>

Обратный отсчет до даты на JavaScript <https://youtu.be/ROehV_LVlzQ>