Лабораторная работа № 4. Пользовательские объекты JS. Специальные операторы

Цель: изучить способы создания пользовательских объектов, познакомиться с правилами применения специальных операторов: delete, in, instanceof, typeof.

Теория

Простейший способ – использовать литерал объекта.

‘Классический’ литерал объекта – это список пар ‘имя свойства: значение’ в фигурных скобках.

Имя свойства – идентификатор или строковый литерал, значение свойства – любое выражение. Пример литерала обьекта:

```
let empty = {};  // это объект без свойств
let point = {x: 0, y: 0};
let point3D = {x: point.x, y: point.y + 1, z: 0, };
let book = {
  ‘main title’: ‘JavaScript’,
  ‘sub-title’: ‘The Definitive Guide’,
  for: ‘all audiences’
};
```

‘Короткие’ свойства в литерах

Позволяют сократить время написания кода. Например, этот литерал:

```
var name = ‘Alex’;
var age = 20;
var user = {
  name: name,
  age: age
};
```

Можно представить вот в таком виде:

```
let name = ‘Alex’;
let age = 20;
let user = {
  name,
  age
};
```

Для создания и инициализации объекта можно использовать оператор new.

Функция после new, называется конструктором и служит для инициализации вновь созданного объекта.

Можно использовать один из встроенных конструкторов или определить

собственный конструктор. Пример:

```
// создать новый пустой объект: то же, что и {}
var o = new Object();
// создать пустой массив: то же, что и []
var a = new Array();
// создать объект RegExp
var r = new RegExp('js');
```

Обращаться к свойствам объектов можно двумя способами:

- используя точку после имени объекта, например **gr1.kurs=2**;
- заключая название свойства в квадратные скобки после имени объекта, например **gr1.['kurs']=2**.

С помощью свойства **prototype** можно добавлять новые свойства и методы к конструкторам объектов. Добавленные к конструктору свойства и методы будут также добавлены ко всем объектам, которые были созданы данным конструктором. Например, `Gruppa.prototype.kurs=this.kurs`.

Удаление свойств объекта (delete).

С помощью оператора **delete** можно удалить свойство объекта, а также элемент массива. При удалении элемента массива удаляется и его индекс, но оставшиеся элементы сохраняют свои прежние индексы, а длина массива не изменяется.

Пример, delete mas[2] - удалить 3-й элемент массива.

Проверка наличия свойств (in).

Оператор **in** позволяет проверить, имеется ли некоторое свойство или метод у того или иного объекта. Если свойство или метод содержится в объекте, то возвращается true, иначе - false. Отсюда следует, что оператор in можно применять в условных выражениях (в операторах if, switch, for, while, do-while).

Примеры:

```
document.write('spec' in grl) // проверить, есть ли свойство spec у объекта gr1;
document.write(1 in mas) //проверить, есть ли элемент с номером 1 в массиве mas.
```

Проверка принадлежности объекта модели (instanceof)

Оператор **instanceof** позволяет проверить, принадлежит ли некоторый объект объектной модели JavaScript. Если они совпадают, метод возвращает true, если нет false.

Выражение с оператором instanceof может использоваться в условных выражениях (в операторах if, switch, for, while, do-while).

Пример, document.write(mas instanceof Array) - проверяет, является ли mas массивом.

Определение типа (typeof)

Оператор **typeof** позволяет проверить, относится ли значение к одному из следующих типов: string, number, boolean, object, function или undefined. Значение, возвращаемое оператором typeof, является строковым и содержит одно из перечисленных названий типа.

Пример,`document.write(typeof gr1.kurs).`

'Настоящие' методы объекта

До ES2015 'метод объекта' – это альтернативное название для свойства-функции.

В ES2015 добавлены именно 'методы объекта', которые, по сути, являются свойствами-функциями, привязанными к объекту.

Особенности 'настоящих' методов объекта:

Короткий синтаксис объявления: вместо prop:function(){…} пишем просто prop(){ … }.

В методе есть спецсвойство [[HomeObject]], ссылающееся на объект, которому метод принадлежит.

Методами объекта автоматически станут объявления геттеров get prop() и сеттеров set prop() (позже).

Пример:

```
let name = 'Alex';
let user = {
  name,  // вместо 'sayHi: function() {...}'
sayHi() { alert(this.name); }
};
user.sayHi();  // Alex
```

Задания к лабораторной работе № 4

Задание 1. Создать пользовательский объект Gruppa (использовать пример, представленный выше). Добавить метод sub_stud (исключить из группы k студентов). Создать несколько экземпляров объекта Gruppa (gr2, gr3, gr4). Применить методы add_stud и sub_stud к каждому экземпляру. Вывести на страницу количество студентов в каждой группе.

Задание 2. Создать пользовательский объект Студент. Свойства: имя, фамилия, физика (оценка), математика, информатика. Методы: вывести свое имя и фамилию в окно, рассчитать средний балл и вывести на страницу. Создать 3 экземпляра объекта Студент. Вывести информацию о всех

студентах.

Добавить какое-нибудь свойство объекту Студент. Задать значение этого свойства для всех экземпляров. Вывести информацию о всех студентах на страницу.

Вместо объекта Студент можно придумать свой пример. Это приветствуется.

Задание 3. Использовать объекты Math, Array и String из лабораторной работы № 3.

Delete. Удалить из массива элемент с номером 3, вывести исходный массив и полученный.

In. Проверить, имеется ли у массива 2-й элемент. Проверить наличие любого свойства у созданных ранее пользовательских объектов Gruppa и Студент.

Instanceof. Проверить, являются ли созданные объекты объектами Array, String, Object.

Typeof. Создать функцию, которая просто возвращает значение 5. Узнать тип всех созданных ранее объектов, включая функцию. Узнать тип всех свойств пользовательских объектов Gruppa и Студент.