Министерство образования Республики Беларусь
Учреждение образования
«Белорусский государственный университет информатики и радиоэлектроники»
Филиал
«Минский радиотехнический колледж»

Учебная дисциплина «Программные средства создания Internet-приложений»

Инструкция

по выполнению лабораторной работы
«Получение данных в формате JSON и их обработка на стороне клиента»

Минск
2021

Лабораторная работа № 30

Тема работы: Получение данных в формате JSON и их обработка на стороне клиента

1. Цель работы

Формирование умений обработки данных в формате JSON в сценариях на языке JavaScript.

2. Задание

Выполнить задания в соответствии с порядком выполнения лабораторной работы.

3. Оснащение работы

ПК, редактор исходного кода, браузер.

4. Основные теоретические сведения

JSON (JavaScript Object Notation) – это текстовый формат обмена данными, основанный на мультипарадигменном (основан на подмножестве языка программирования JavaScript) языке программирования. Его основное назначение состоит в хранении и передаче структурированного потока информации.

JSON используется для хранения данных, подобных XML. Он основан на синтаксисе нотации объектов JavaScript. В JSON четыре основных синтаксических правила, зная которые легко с нуля написать объекты JSON:

– данные хранятся в парах ключ/значение.

“city”:”Chicago”

Это значение может быть числом, строкой, булевской переменной (true/false), массивом, объектом или псевдозначением null;

– для разделения данных используются запятые;

– объекты заключаются в фигурные скобки:

{“city”:”Chicago”, “state”:”Illinois”}

– массивы заключаются в квадратные скобки:

```
Var CitiesVisited = [{“city”:”Chicago”, “state”:”Illinois”, “year”:”2003”},
                     {“city”:”Portland”, “state”:”Oregon”, “year”:”2007”},
                    {“city”:”New York”, “state”:”New York”, “year”:”2013”}]
```

Можно создавать простые базы данных, размещая несколько объектов в одном массиве. Это позволит в программе использовать любую часть этого массива.

Пример объекта "CitiesVisited", содержащего массив городов (выше):

```
document.write (‘City visited: ’CitiesVisited[1].city);
                                        // Output: City visited: Portland
```

Объекты в формате JSON похожи на обычные JavaScript-объекты, но отличаются от них более строгими требованиями к строкам – они должны быть именно в двойных кавычках.

Кроме того, в формате JSON не поддерживаются комментарии. Он предназначен только для передачи данных.

Основные методы для работы с JSON в JavaScript – это:

JSON.parse – читает объекты из строки в формате JSON;

JSON.stringify – превращает объекты в строку в формате JSON, используется, когда нужно из JavaScript передать данные по сети.

Вызов JSON.parse(str) превратит строку с данными в формате JSON в JavaScript-объект/массив/значение.

```
var numbers = "[0, 1, 2, 3]";
numbers = JSON.parse(numbers);
alert( numbers[1] ); // 1

var user = '{ "name": "Вася", "age": 35, "isAdmin": false, "friends": [0,1,2,3] }';
user = JSON.parse(user);
alert( user.friends[1] ); // 1
```

Данные могут быть сколь угодно сложными, объекты и массивы могут включать в себя другие объекты и массивы. Главное, чтобы они соответствовали формату.

Метод JSON.parse поддерживает и более сложные алгоритмы разбора.

Например, объект с данными события event

```
// title: название события, date: дата события
var str = '{"title":"Конференция","date":"2014-11-30T12:00:00.000Z"}';

var str = '{"title":"Конференция","date":"2014-11-30T12:00:00.000Z"}';
var event = JSON.parse(str);
alert( event.date.getDate() ); // ошибка!
```

Дело в том, что значением event.date является строка, а отнюдь не объект Date. Метод JSON.parse не знает, что нужно превратить строку именно в дату.

Для интеллектуального восстановления из строки у JSON.parse(str, reviver) есть второй параметр reviver, который является функцией function(key, value).

Если она указана, то в процессе чтения объекта из строки JSON.parse передаёт ей по очереди все создаваемые пары ключ-значение и может возвратить либо преобразованное значение, либо undefined, если его нужно пропустить.

В данном случае мы можем создать правило, что ключ date всегда означает дату:

```
// дата в строке - в формате UTC
var str = '{"title":"Конференция","date":"2014-11-30T12:00:00.000Z"}';
```

```
var event = JSON.parse(str, function(key, value) {
  if (key == 'date') return new Date(value);
  return value;
});

alert( event.date.getDate() ); // теперь сработает!
```

эта возможность работает и для вложенных объектов тоже:

```
var schedule = '{ \
  "events": [ \
    {"title":"Конференция","date":"2014-11-30T12:00:00.000Z"}, \
    {"title":"День рождения","date":"2015-04-18T12:00:00.000Z"} \
  ]\
}';

schedule = JSON.parse(schedule, function(key, value) {
  if (key == 'date') return new Date(value);
  return value;
});

alert( schedule.events[1].date.getDate() ); // сработает!
```

Функцию JSON.parse() можно использовать в контексте файла HTML.
Пример:

```
<!DOCTYPE html>
<html>
<body>
<p id="user"></p>
<script>
var s = '{"first_name" : "John", "last_name" : "Smith", "location" : "London"}';
var obj = JSON.parse(s);
document.getElementById("user").innerHTML =
"Name: " + obj.first_name + " " + obj.last_name + "<br>" +
"Location: " + obj.location;
</script>
</body>
</html>
```

Name: John Smith
Location: London

Метод JSON.stringify(value, replacer, space) преобразует («сериализует») значение в JSON-строку.

```
var event = {
  title: "Конференция",
```

```
  date: "сегодня"
};

var str = JSON.stringify(event);
alert( str ); // {"title":"Конференция","date":"сегодня"}

// Обратное преобразование.
event = JSON.parse(str);
```

При сериализации объекта вызывается его метод toJSON. Если такого метода нет – перечисляются его свойства, кроме функций.

```
var room = {
  number: 23,
  occupy: function() {
    alert( this.number );
  }
};

var event = {
  title: "Конференция",
  date: new Date(Date.UTC(2014, 0, 1)),
  room: room
};

alert( JSON.stringify(event) );
/*
  {
    "title":"Конференция",
    "date":"2014-01-01T00:00:00.000Z",  // (1)
    "room": {"number":23}               // (2)
  }
*/
```

Обратим внимание на два момента:

1. Дата превратилась в строку. Это не случайно: у всех дат есть встроенный метод toJSON. Его результат в данном случае – строка в таймзоне UTC.
2. У объекта room нет метода toJSON. Поэтому он сериализуется перечислением свойств.

Если добавить такой метод, тогда в итог попадет его результат:

```
var room = {
  number: 23,
  toJSON: function() {
     return this.number;
    }
```

```
};

alert( JSON.stringify(room) ); // 23
```

Попытаемся преобразовать в JSON объект, содержащий ссылку на DOM.

```
var user = {
  name: "Вася",
  age: 25,
  window: window
};

alert( JSON.stringify(user) ); // ошибка!
// TypeError: Converting circular structure to JSON (текст из Chrome)
```

Глобальный объект window – сложная структура с кучей встроенных свойств и круговыми ссылками, поэтому его преобразовать невозможно.

Во втором параметре JSON.stringify(value, replacer) можно указать массив свойств, которые подлежат сериализации.

```
var user = {
  name: "Вася",
  age: 25,
  window: window
};

alert( JSON.stringify(user, ["name", "age"]) );
// {"name":"Вася","age":25}
```

Для более сложных ситуаций вторым параметром можно передать функцию function(key, value), которая возвращает сериализованное value либо undefined, если его не нужно включать в результат:

```
var user = {
  name: "Вася",
  age: 25,
  window: window
};

var str = JSON.stringify(user, function(key, value) {
  if (key == 'window') return undefined;
  return value;
});

alert( str ); // {"name":"Вася","age":25}
```

В примере выше функция пропустит свойство с названием window. Для остальных она просто возвращает значение, передавая его стандартному алгоритму.

Функция replacer работает рекурсивно. То есть, если объект содержит вложенные объекты, массивы и т.п., то все они пройдут через replacer.

В методе JSON.stringify(value, replacer, space) есть ещё третий параметр space.

Если он является числом – то уровни вложенности в JSON оформляются указанным количеством пробелов, если строкой – вставляется эта строка.

Например:

```
var user = {
  name: "Вася",
  age: 25,
  roles: {
    isAdmin: false,
    isEditor: true
  }
};

var str = JSON.stringify(user, "", 4);

alert( str );
/* Результат -- красиво сериализованный объект:
{
    "name": "Вася",
    "age": 25,
    "roles": {
        "isAdmin": false,
        "isEditor": true
    }
}
*/
```

5. Порядок выполнения работы

1. Создайте объект-конструктор (класс), который содержит информацию, в соответствии с вариантом, таблица 30.1.

№ варианта	Задание
1	Информация о сотрудниках фирмы (ФИО, возраст, название отдела, стаж)
2	Информация о клиентах фирмы (ФИО, адрес, статус, размер скидки)
3	Информация о товаре (код товара, наименование, товарная группа, количество)
4	Информация о пациентах больницы (ФИО, дата рождения, диагноз, ФИО лечащего врача)
5	Информация о стране (Название, столица, население, национальный язык)
6	Информация об учащихся (ФИО, группа, отделение, пол)
7	Информацию об автошколах (название, адрес, номер телефона, количество автомобилей, стоимость)
8	Информация о кинотеатрах (название, адрес, телефон, количество залов)

9	Информация о парковках (адрес, количество мест, телефон, стоимость)
10	Информация о книгах (название, автор, год издания, количество страниц)

2. Создайте экземпляр объекта, выполнив ввод данных с помощью диалогового окна prompt.

3. По нажатия на кнопку, преобразуйте созданный объект JavaScript в JSON-строку.

4. Выведите полученное строковое значение, выполнив встраивание текстового содержимого в HTML-элемент.

5. Преобразуйте созданную JSON-строку назад в объект JavaScript таким образом, чтобы полученный объект содержал только те свойства, для которых значение свойств имеет тип string.

6. Создайте строковую переменную, которая содержит данные в виде JSON-строки.

6. Форма отчета о работе

Лабораторная работа № ___

Номер учебной группы _____

Фамилия, инициалы учащегося _____

Дата выполнения работы _____

Тема работы: _____

Цель работы: _____

Оснащение работы: _____

Результат выполнения работы: _____

_____

_____

7. Контрольные вопросы и задания

1. Что представляет собой JSON?

2. Опишите методы для работы с JSON в JavaScript.

3. Какой еще формат представления данных может быть использован для обмена данными между браузером и сервером?

8. Рекомендуемая литература

1. **JAVASCRIPT.RU** [Электронный ресурс] / Современный учебник JavaScript – 2007—2020 Илья Кантор. – Режим доступа: https://learn.javascript.ru. – Дата доступа: 04.03.2020.

2. **Никсон, Р.** Создаем динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5 / Р. Никсон. 4-е изд. – СПб.: Питер, 2018.

3. **Симпсон, К.** ES6 и не только / К. Симпсон. – СПб.: Питер, 2017.

4. **Хавербеке, М.** Выразительный JavaScript. Современное веб-программирование / М. Хавербеке – СПб.: Питер, 2019.