2011 © melifrya@gmail.com

Тест на JavaScript с использованием AJAX.

Цель данного задания заключается в том чтобы понять принцип и научиться использовать автоматическую генерацию страницы или отдельных ее элементов с помощью скриптов, при этом еще и использовать элементы ajax для организации асинхронной загрузки этих самых элементов.
Мы будем тренироваться на создании теста, т.е.первоначальный вид html страницы будет содержать только контейнер куда будет генерироваться текст, к примеру при нажатии на заголовок "Тест".

```
<html>
  <head>
    <title>Тест</title>
  </head>
  <body>
    <h1>Тест</h1><div></div>
  </body>
</html>
```

Теперь нам нужен текстовый файл с массивом вопросов и ответов, по желанию можно записать вопросы и ответы в один массив.
текстовый файл q.txt :

```
var q = [
["Сколько будет 2*2: ", "6", "2", "4"],
["Сколько будет 2х6: ", "11", "14", "12"],
["Столица Германии: "]
];
var qq = [4, 4, "Берлин"];    // массив ответов
```

Теперь алгоритм написания скрипта, его можно отдельным файлом подключить или же писать прям в html странице между тегов script. Мне больше нравится отдельным файлом, звать его будет script.js
подключим его к страничке: <script src="script.js" type="text/javascript"></script>

В самой скриптовом файле у нас должна быть функция вызываемая при нажатии на "Тест"(назовем её **doIt**) и загрузка текстового файла q.txt

Алгоритм написания скрипта примерно такой:

- объявляем необходимые переменные

```
var cod; // сюда положим текст из текстового файла
var ql; //длинна массива с вопросами
var qql; // длина массива с ответами
var ans = new Array();  // массив для ответов данных пользователем
// массивы q с вопросами и qq с ответами будут определены за счет того в текстовом файле мы дописали им var
```

- подгружаем текстовый файл (http://dist-learn.spb.ru/students/ajax в прицепи почти полностью копируем то что написано в ссылке)

```
var xhr = new XMLHttpRequest() || new ActiveXObject('Microsoft.XMLHTTP');
xhr.open("GET","q.txt",true);
xhr.onreadystatechange = function() {
        if (xhr.readyState != 4) return;  // четверка означает что ошибок
        cod =  xhr.responseText;   // в переменную cod кладется текст который был в q.txt
}
xhr.send(null);
```

это обеспечит нам подгрузку текстового файла без перезагрузки страницы, т.е. ajax

- теперь наконец то описываем нашу функцию doIt

В html файле она будет вызываться в виде: <h1 onclick="doIt(this.nextSibling)">Тест</h1>

this.nextSibling у нас позволяет обратиться к элементу следующему сразу за элементов где вызывается функция, любой пробел и enter считается следующим элементом, потому между h1 и div мы следим чтоб не было ничего лишнего, прям впритык пишем

Таким образом мы функции doIt передаем одно значение и это у нас div, назовем его obj

```
function doIt(obj) {
    ...
}
```

Эта функция должна прежде всего превратить полученный текст из файла в код, это делает метод eval(имя переменной), т.е. у нас это будет **eval(cod);**

Выясняем длинную массива с вопросами: ql=q.length; и массива с ответами qql=qq.length;

Теперь заходим в цикл по длине массива с вопросами:

```
  for(i=0; i<ql; i++){
        ...
  }
```

В этом цикле мы идем уже по массивам в массиве вопросов.
Нам надо достать первый элемент с текстом вопроса и присыновить его к странице методом (кому присыновляем).appendChild(кого присыновляем);

```
   var tx = document.createTextNode(q[i][0]);  // достаем сам вопрос из массива
   obj.appendChild(tx); // присыновили к div
```

метод **createTextNode**() создает просто текст
метод **createElement**() создает необходимый элемент
метод **push** позволяет записывать в массив желаемый элемент. (имя массива).push(элемент)

Теперь надо понять как нам определить что создавать input или select
Все просто, если у нас длинна массива с вопросом единица то воздается input, так как ответ пользователь сам вписывать должен, если же длинна больше единицы создаем select

```
if ((q[i].length) == 1){
  //создаем input для ответа
  var inp = document.createElement("input");
  obj.appendChild(inp);
  ans.push(inp);   //
}

else{
//создаем select и option с пустым вариантом
  var sel = document.createElement("select");
  var opt = document.createElement("option");
  opt.setAttribute("value", 0);
  var tx = document.createTextNode(" ");
  opt.appendChild(tx);
  sel.appendChild(opt);

// достаем варианты ответов и пихаем в option
  for (j=1; j < (q[i].length); j++){
        var opt = document.createElement("option");
        opt.setAttribute("value", j+1);
        var tx = document.createTextNode(q[i][j]);
```

```
        opt.appendChild(tx);
        sel.appendChild(opt);
    }

  obj.appendChild(sel);
  ans.push(sel);
  }
```

и в завершении цикла по массивам вопросов надо разделить все тегами
 чтобы не было каши на странице

```
// разделяем тегом <br>
  var br = document.createElement("br");
  obj.appendChild(br);
```

С вопросами закончили, теперь надо кнопку для проверки!

```
                var btn= document.createElement("button");
                var tx = document.createTextNode("Проверить");
                btn.appendChild(tx);

                btn.onclick = checkTest();
                obj.appendChild(btn); // прицепили кнопку к div
```

- создаем функцию проверки теста
как видно из строчки выше назвали мы ее checkTest();

```
function checkTest() {
        eval(cod);  // тут я еще раз вызывала эту функцию потому что может возникнуть ошибку, потому как в предыдущей функции мы вызывали eval внутри функции doIt? а сейчас мы уже за её пределами
        var score = 0;  // начальное количество баллов
        obj = document.getElementsByTagName("div")[0];  // тут мы удалим все что содержится в div и вместо этого запишем результат теста

        for (i=0; i < qql; i++){
                if (ans[i].value == qq[i]) score++;  //сравнение введенных ответов с ответами
                //alert(ans[i].value);     // вывод ответа введеного пользователем, для проверки работы
                }
```

```
        obj.innerHTML = "Ваш результат: " + score + " из " + qql;   // вывод результата
}
```

И вот! Ура! :)
Наш прекрасный тест работает без перезагрузки страницы с автоматической генерацией всех элементов!