Вторая часть основ языка сценариев JavaScript

В этом **очень кратеньком** пособии мы рассмотрим применение навыков из прошлой части на примерах.

Данное пособие **не является исчерпывающим** по языку, и не претендует на досканальное изучение всех возможностей языка JavaScript, скорее это краткое изложение основных возможностей на примерах.

Данное пособие написано специально для сайта **snakrproject.ru**. На сайте Вы можете в разделе контактов написать мне отзыв в письме.

Приятного чтения!

Примеры будут включать в себя изначально три файла:

Index.php – наша страничка:

```
<html>
<head>
<script src='script.js'></script>
<link rel='stylesheet' type='text/css' href='style.css'>
</head>
<body>
</body>
</html>
```

Script.js – файл с кодом JavaScript:

Изначально пусть будет пустой.

Style.css – файл стилей:

```
#bDiv{
        color: red;
}

#tDiv{
```

```
    background: yellow;
    color: red;
    height: 30px;
    width: 30px;
}
```

1. **Alert(), Confirm(), Prompt() и некоторые события браузера**

Рассмотрим несколько часто применяемых возможностей взаимодействия с пользователем:

Напишем код в script.js:

```
per = 1;
function func(){
    document.write(2+window.per + '<br />');
    alert('Alert!');
    var per = confirm('Alert?');
    document.write(per + '<br />');
    var per = prompt('Insert num: ', 0)
    document.write(per + '<br />');
}
```

И немного переделаем index.php:

```
<body onload="console.log('onload is works');" onresize="console.log('onresize is works');">
<a href="javascript:func()" onclick="document.write('onclick is works!' + '<br />');">Href</a>
<hr />
```

Собственно загрузив в браузере нашу страничку **index.php** мы увидим ссылку, по которой при клике мышкой будет срабатывать событие «**onclick**» и спомощью **«document.write»** выведется строка **«'onclick is works!»**

Далее отработает **alert** (выскачет предупреждение в виде окошка)

Далее отработает **confirm** (окошко с возможностью нажать ОК или Отмена)

Далее отработает **prompt** (Тут нужно будет ввести что-либо, значение по умолчанию было определено в функции)

Результаты выполнения будут появляться в теле страницы, так можно узнать, какие возвращаются значения и в дальнейшем использовать их.

Теперь например в **GoogleChrome** нажмите **f12** и перейдите во вкладку **console**, там можно будет увидеть **'onload is works'**, говорящее об отработанном коде после события **onload** в теге **<body>,** и если попробовать поменять размер окна браузера, то отработает событие **onresize**. Их можно использовать для своих каких-то нужд, например изменения размера картинки в процентном соотношении от размера браузера.

2. **Как поменять например цвет шрифта в блоке?**

Дополняем index.php:

```
<a href="#" onclick="funcElId()">funcElId</a>

<div id ='bDiv'>Div</div>

<hr />
```

И дополним script.js

```
function funcElId(){

    document.getElementById("bDiv").style.color = "green";

}
```

Логика очень простая:

По клику ссылки срабатывает событие **onclick**, и вызывается функция **funcElId()**

Функция находит блок ** ** по **id='bDiv'** и изменяет атрибут в **CSS – color**

3. **Склеивание введенных значений плюс проверка заполнения формы**

Есть два поля, при введении в них значений, они должны склеиться в одну строку и вывестись в третьем поле.

Дополним index.php:

```
<form id='bForm' action='index.php' method='POST'>

    <p>Any num:<input name='one' id='one' onchange="funcTotal();"></input></p>

    <p>Any num:<input name='two' id='two' onchange="funcTotal();"></input></p>

    <p>Total:<input name='total' id='total'></input></p>

    <input type='submit' onclick="funcTotalSub();">

<hr />
```

Дополним script.js:

```
function funcTotal(){
	var a1=document.getElementById("one").value;
	var a2=document.getElementById("two").value;
	var a3 = a1 + a2;
	document.getElementById("total").value = a3;
}

function funcTotalSub(){
	var a1=document.getElementById("one").value;
	var a2=document.getElementById("two").value;

	if(a1=="") alert("1 field is empty");
	else if(a2=="") alert("2 field is empty");
	else
	{
		alert("All good!");
		form.submit();
	}
}
```

Тут тоже все просто.

Мы создаем форму и присваиваем полям **id**.

После изменения поля срабатывает событие **onchange**, которое вызывает функцию **funcTotal(),** которая собственно и присваивает третьему полю введенное значение.

При клике по кнопке срабатывает уже знакомое событие **onclick**, и вызывается функция **funcTotalSub(),** которая делает проверку двух полей, находя их по **id**, и в случае их нулевой длины вызывает **alert**, если оба поля были заполнены, форма отправляется **form.submit();**

4. **Изменение размеров с таймаутом**

Дополним index.php:

```
<a href="#" onclick="funcElTime()">funcElTime</a>
```

```
<div id ='tDiv'>Div</div>

<hr />
```

Дополним script.js:

```
function funcElTime(){
        setTimeout('document.getElementById("tDiv").style.color = "green";', 3000);
        setTimeout('document.getElementById("tDiv").style.height = "45px";', 3000);
        setTimeout('document.getElementById("tDiv").style.width = "45px";', 3000);
}
```

Тут по клику ссылки отработает функция **funcElTime**, которая выполнит последовательно три изменения параметров блока **div c id = 'tDiv'**, при этом будет задержка каждого действия на 3 секунды из-за отработки **setTimeout** с параметром **3000**.

5. **Установка «Куки»**

Дополним index.php:

```
<a href="#" onclick="funcCookie()">funcCookie</a>

<hr />
```

Дополним script.js:

```
function funcCookie(){
        var coo = document.cookie;
        if(coo){
                console.log(coo);
        }
        else{
                var name = prompt("Enter your name:");
                var date = new Date();
                date.setSeconds(59);
                alert(date);
                document.cookie = "foo"+"="+name+"; expires"+"="+date.toGMTString()+";";
        }
}
```

По клику ссылки произойдет установка «куки», с названием **foo**, внутри будет введенное Вами значение, кука будет действительно до конца текущей минуты, т.к. мы поставили свойство **date.setSeconds(59);**

6. **Предупреждения о незаполненности поля формы**

Давайте предупредим пользователя, о том, что поле обязательно к заполнению.

Дополним index.php:

```
<form id='wForm' action='index.php' method='GET'>
	<p>Any num:<input name='one' id='one' onblur="warnForm('one', this);"></input></p>
	<p>Any num:<input name='two' id='two' onblur="warnForm('two', this);"></input></p>
	<p>Result:<input name='result' id='result'></input></p>
</form>
<hr />
```

Дополним script.js:

```
function warnForm(name, arg){
	if(arg.value.length == 0){
		document.getElementById("result").value = name + " is empty!";
	}
}
```

Тут событие **onblur** каждого из полей будет отрабатывать при потере фокуса, т.е. человек кликнул в поле, и оставил его пустым, при этом в итоговом поле появится надпись, что поле – пусто.

Для этого в функцию **warnForm** передаются два аргумента с названием поля и само значение, которое проверяется на ненулевую длину.

7. **Еще несколько примеров обращения к HTML тегам**

Дополним index.php:

```
<form name='iForm' id='iForm' action='index.php' method='GET'>
	<p>Any num:<input name='one' id='one' onblur="iFormF(this);"></input></p>
	<p>Result:<b id='idB'>test</b></p>
</form>
```

```
<br />
<div onclick="funcBB();">CLic me and edit bgcolor</div>

<br />
<div onclick="funcBC();">CLic me and edit bgcolor</div>

<br />
<div onclick="funcBD();">CLic me and edit value</div>
<hr />
```

Дополним script.js:

```
function iFormF(arg){
	document.getElementById("idB").innerHTML = arg.value;
}

function funcBB(){
	document.getElementsByTagName("div")[3].style.background = 'green';
}

function funcBC(){
	document.body.style.background = 'yellow';
	document.body.style.fontSize = '18';
}

function funcBD(){
	document.iForm.elements.one.value = 'Test';
}
```

В этих примерах первым будет отработка метода **innerHTML**, который будет вписывать в найденный **HTML** тег значение из поля, после потерей его фокусом.

Второй пример показывает возможность обращения к четвертому по счету с верху блока ** ** и изменения его **CSS** атрибута

Третий пример показывает возможность обращения по имени **HTML** тега

Четвертый пример показывает возможность обращения к элементу формы, и изменении значения.

На этом наше краткое руководство подошло к концу.

В нем я постарался разобрать несколько часто встречаемых задачек по обращению к тегам и изменения свойств CSS.

Надеюсь оно было для Вас полезным.