Лабораторная работа №3
Функция и обработка события

Теория

Основным элементом языка JavaScript является функция. Описание функции имеет вид
function F (V) {S},
где F - идентификатор функции, задающий имя, по которому можно обращаться к функции; V - список параметров функции, разделяемых запятыми; S - тело функции, в нем задаются действия, которые нужно выполнить, чтобы получить результат. Необязательный оператор return определяет возвращаемое функцией значение. Обычно все определения и функции задаются в разделе <head> документа. Это обеспечивает интерпретацию и сохранение в памяти всех функций при загрузке документа в браузер.

Пример 1. Нахождение площади треугольника.

В предыдущих примерах пользователю не предоставлялась возможность вводить значения, и в зависимости от них получать результат. Интерактивные документы можно создавать, используя формы. Предположим, что мы хотим создать форму, в которой поля Основание и Высота служат для ввода соответствующих значений. Кроме того, в форме создадим кнопку Вычислить. При щелчке мышью по этой кнопке мы хотим получить значение площади треугольника. Действие пользователя (например, щелчок кнопкой мыши) вызывает событие. События в основном связаны с действиями, производимыми пользователем с элементами форм HTML. Обычно перехват и обработка события задается в параметрах элементов форм. Имя параметра обработки события начинается с приставки on, за которой следует имя самого события. Например, параметр обработки события click будет выглядеть как onclick.

Листинг 1. Реакция на событие Click.

```
<HTML>
<HEAD>
<title>Обработка значений из формы</title>
<script language="JavaScript">
<!--//
function care (a, h)
{
var s=(a*h)/2;
```

```
document.write ("Площадь прямоугольного треугольника равна ",s);
return s
}
//-->
</script>
</HEAD>
<BODY>
<P>Пример сценария со значениями из формы</P>
<FORM name="form1">
Основание: <input type="text" size=5 name="st1"><hr>
Высота: <input type="text" size=5 name="st2"><hr>
<input type="button" value=Вычислить
onClick="care(document.form1.st1.value, document.form1.st2.value)"> /*По клику мыши на кнопке в функцию care передаются два параметра - содержимое полей ввода*/
</FORM>
</BODY>
</HTML>
```

При интерпретации HTML-страницы браузером создаются объекты JavaScript. Взаимосвязь объектов между собой представляет иерархическую структуру. На самом верхнем уровне иерархии находится объект windows, представляющий окно браузера. Объект windows является предком или родителем" всех остальных объектов. Каждая страница кроме объекта windows имеет объект document. Свойства объекта document определяются содержимым самого документа: цвет фона, цвет шрифта и т. д. Для получения значения основания треугольника, введенного в первом поле формы, должна быть выполнена конструкция

document.forml.stl.value

т.е., говоря русским языком (при этом читаем с конца), используем данные value из поля ввода с именем st1 находящегося на форме form1 объекта document.

Пример 2. Вычисление площади квадрата.

Напишем сценарий, определяющий площадь квадрата по заданной стороне. Площадь должна вычисляться в тот момент, когда изменилось значение его стороны. Пусть форма содержит два текстовых поля: одно для длины стороны квадрата, другое для вычисленной площади. Кнопка Обновить очищает поля формы. Площадь квадрата вычисляется при возникновении события change, которое происходит в тот момент, когда значение элемента формы с именем num1 изменилось, и элемент потерял фокус. HTML-код представлен в примере 2.

Листинг 2. Реакция на событие Change

```
<HTML> <HEAD>
<title>Обработка события Change - изменение значения элемента</title>
<script>
function srec(obj)
{obj.res.value=obj.num1.value* obj.num1.value}
</script>
            </HEAD>
<BODY>
<P>Вычисление площади квадрата</P>
<FORM name="form1">
Сторона: <input type="text" size=7 name="num1"
onChange="srec(form1)">
<hr>
Площадь: <input type="text" size=7 name="res"><hr>
<input type="reset" value=Обновить>
</FORM>
</BODY>
</HTML>
```

Событие Focus возникает в момент, когда пользователь переходит к элементу формы с помощью клавиши <Tab> или щелчка мыши.

Событие "потеря фокуса" **(Blur**) происходит в тот момент, когда элемент формы теряетфокус.

Рассмотрим пример для проверки введённых данных.

В примере ниже:

- Обработчик blur проверяет, введён ли email, и если нет – показывает ошибку.
- Обработчик focus скрывает это сообщение об ошибке (в момент потери фокуса проверка повторится):

```
<style>
  .invalid { border-color: red; }
  #error { color: red }
</style>

Ваш email: <input type="email" id="input">

<div id="error"></div>

<script>
input.onblur = function() {
  if (!input.value.includes('@')) { // не email
    input.classList.add('invalid');
```

```
    error.innerHTML = 'Пожалуйста, введите правильный email.'
  }
};

input.onfocus = function() {
  if (this.classList.contains('invalid')) {
    // удаляем индикатор ошибки, т.к. пользователь хочет ввести данные заново
    this.classList.remove('invalid');
    error.innerHTML = "";
  }
};
</script>
```

Современный HTML позволяет делать валидацию с помощью атрибутов required, pattern и т.д.

JavaScript можно использовать, когда мы хотим больше гибкости. А ещё мы могли бы отправлять изменённое значение на сервер, если оно правильное.

Методы focus/blur

Методы elem.focus() и elem.blur() устанавливают/снимают фокус.

Например, запретим посетителю переключаться с поля ввода, если введённое значение не прошло валидацию:

```
<style>
  .error {
    background: red;
  }
</style>

Ваш email: <input type="email" id="input">
<input type="text" style="width:280px" placeholder="введите неверный email и кликните сюда">

<script>
  input.onblur = function() {
    if (!this.value.includes('@')) { // не email
      // показать ошибку
      this.classList.add("error");
      // ...и вернуть фокус обратно
      input.focus();
    } else {
      this.classList.remove("error");
    }
  };
</script>
```

Это сработает во всех браузерах, кроме Firefox (bug).

Если мы что-нибудь введём и нажмём Tab или кликнем в другое место, тогда onblur вернёт фокус обратно.

Отметим, что мы не можем «отменить потерю фокуса», вызвав event.preventDefault() в обработчике onblur потому, что onblur срабатывает *после* потери фокуса элементом.

Однако на практике следует хорошо подумать, прежде чем внедрять что-то подобное, потому что мы обычно *должны показывать ошибки* пользователю, но они *не должны мешать* пользователю при заполнении нашей формы. Ведь, вполне возможно, что он захочет сначала заполнить другие поля.

Потеря фокуса, вызванная JavaScript
Потеря фокуса может произойти по множеству причин.

Одна из них – когда посетитель кликает куда-то ещё. Но и JavaScript может быть причиной, например:

- alert переводит фокус на себя – элемент теряет фокус (событие blur), а когда alert закрывается – элемент получает фокус обратно (событие focus).
- Если элемент удалить из DOM, фокус также будет потерян. Если элемент добавить обратно, то фокус не вернётся.

Из-за этих особенностей обработчики focus/blur могут сработать тогда, когда это не требуется.

Используя эти события, нужно быть осторожным. Если мы хотим отследить потерю фокуса, которую инициировал пользователь, тогда нам следует избегать её самим.

Событие select вызывается выбором части или всего текста в текстовом поле. Например, щелкнув дважды мышью по полю, мы выделим поле, наступит событие select, обработка которого приведет к вычислению требуемого значения. В табл.3 представлены события и элементы документов HTML, в которых эти события могут происходить. В языке JavaScript определены некоторые стандартные объекты и функции,пользоваться которыми можно без предварительного описания. Одним из стандартных объектов является объект Math. В свойствах упомянутого объекта хранятся основные математические константы, а его методы можно использовать для вызова основных математических функций. В табл.4 приведены некоторые методы объекта Math. Выражение y=log x запишется y=Math.log(x).

Задание

1. Проверить примеры из лабораторной работы.
2. На плоскости заданы координаты трех точек. Напишите сценарий, который вычисляет площадь треугольника (использовать событие Focus).
3. Напишите сценарий, который для точки, заданной координатами на плоскости, определяет расстояние до начала координат (использовать событие Select).
4. Напишите сценарий, который обменивает местами значения двух введенных переменных (использовать событие Blur).