Методы и средства решения стандартных задач профессиональной деятельности

ЛАБОРАТОРНАЯ РАБОТА №3.
Взаимодействие с пользователями через веб формы.

Интерактивность — ключевой компонент любого современного сайта. И одним из наиболее часто используемых событий для создания интерактивности является событие onclick.
Рассмотрим пример: при нажатии на ссылку появляется всплывающее окно с сообщением:

```
<a href="#" onclick="alert('onclick сработал по ссылке');
return false;">Ссылка</a>
```

При работе с формами javascript-код разделен от html тегов. В теге будем прописывать название функции, а саму функцию вынесем в отдельный блок.

Формы представляют собой интерактивные элементы HTML, позволяющие разработчикам страниц взаимодействовать с посетителями. С их помощью пользователь может возвращать комментарии по поводу посещения определенного узла, пересылать запросы или регистрироваться.

Разработчик задает вопросы, создавая форму, а пользователь отвечает на них, заполняя её.

Форма создается при помощи различных тэгов и атрибутов, заключенных в пару <FORM> текст </FORM>.

С помощью форм можно получить доступ к любому элементу формы, изменять его и применять по своему усмотрению.

Поля в форме именуются тегом <input> имеют один важный атрибут – type, который определяет тип поля, который будет использован. Type может принимать значения:

Тип	Описание	Вид
button	Кнопка.	Кнопка
checkbox	Флажки. Позволяют выбрать более одного варианта из предложенных.	☐ Пиво ☐ Чай ☐ Кофе
file	Поле для ввода имени файла, который пересылается на сервер.	Выберите файл Файл не выбран
hidden	Скрытое поле. Оно никак не отображается на веб-странице.	
image	Поле с изображением. При нажатии на рисунок данные формы отправляются на сервер.	Отправить
password	Обычное текстовое поле, но отличается от него тем, что все символы показываются звездочками. Предназначено для того, чтобы никто не подглядел вводимый пароль.	
radio	Переключатели. Используются, когда следует выбрать один вариант из нескольких предложенных.	○ Пиво ○ Чай ○ Кофе
reset	Кнопка для возвращения данных формы в первоначальное значение.	Сбросить
submit	Кнопка для отправки данных формы на сервер.	Отправить
text	Текстовое поле. Предназначено для ввода символов с помощью клавиатуры.	

Рисунок 1 - Работа с формами

Для работы с данными используется объектная модель документа – DOM.
Чтобы обратиться к **DOM** используется **объект document** - это глобальный объект веб-страницы.

Одним из методов объекта document является метод getElementById (получить элемент по id) Любому элементу (тегу) веб-документа может быть назначен свой id-идентификатор, благодаря которому элемент становиться уникальным и к нему можно получить доступ для работы именно с ним.

Рассмотрим пример:

```
<!DOCTYPE html>
<html>
<body>
<p id="demo">Нажмите кнопку, чтобы изменить текст в этом параграфе.</p>
<button onclick="myFunction()">Попробовать</button>
<script>
function myFunction() {
  document.getElementById("demo").innerHTML = "Привет мир";
}
</script>
</body>
</html>
```

В данном примере для кнопки button задаем onclick вызывая функцию myFunction. В свою очередь в функции myFunction мы обращаемся к объекту документа с id равным значению demo и с помощью свойства innerHTML установливаем новое значение данного элемента HTML, в примере заменяем текст.

Предоставляя пользователю возможность заполнения полей на сайте, следует проверять введенные данные на валидность. Это позволит предупредить пользователя о случайных ошибках, а так же даст дополнительную защиту от спама.

Валидность адреса электронной почты

Рассмотрим адрес электронной почты (test@mail.ru). Вот его обязательные части:

Название (test) — один или много символов;

Знак собаки (@);

Доменное имя почтового сервера (mail) — один или много символов;

Точка (.);

Доменное имя первого уровня (ru) от двух до пяти букв.

Составим регулярное выражение для наших требований:

/^[\w-\.]+@[\w-]+\.[a-z]{2,4}$/i

Разберём правило по частям:

Регулярное выражение должно открываться и закрываться символами «/». После закрывающегося символа можно указать директиву. В нашем случае такой директивной является «i», которая отключает проверку вводимых букв на регистр. То есть, становится не важно, ввели «test@mail.ru» или «Test@Mail.RU».

Знаки «^» и «$» обозначают начало и конец проверяемой строки. Если их убрать, то правило вернет положительный результат даже если в начале или конце электронного адреса поставить запрещенные знаки. То есть, при вводе «%:&test@mail.ru#6&» функция проверки вернет положительный результат, так как в строке имеется последовательность символов, удовлетворяющая нашему правилу. Для исключения такой возможности указываем, что правило должно применяться ко всей строке, а не к её части.

Блок «[\w-\.]+» отвечает за проверку названия ящика. В квадратных скобках указываем разрешенные символы: «\w» — все латинские буквы, цифры и знак подчеркивания. Так же рекомендую добавлять знак тире и точку «-\.». «+» после квадратных скобок указывает на возможность повторения символов — один или много раз.

Далее идет знак собаки и доменное имя почтового сервера — «@[\w-]+». Здесь практически тоже самое что и в предыдущем блоке. Исключаем только из набора символов точку.

Осталось прописать правило для проверки наличия точки и корректности доменного имени верхнего уровня (ru,com,info). «\.[a-z]{2,4}». Для обозначения знака точки мы указываем её с обратным слешем «\.» Без него она будет восприниматься зарезервированным символом регулярки, который обозначает возможность наличия на её месте любого символа. За точкой должно следовать доменное имя верхнего уровня. Это минимум 2 латинские буквы — «[a-z]{2,4}».

Разобранный пример немного упрощен для лучшего восприятия. У него есть недостаток — первым знаком в email не может быть тире или точка, но приведенное регулярное выражение этого не учитывает. Чтобы это исправить следует его исправить:

/^[\w]{1}[\w-\.]*@[\w-]+\.[a-z]{2,4}$/i

Пример формы:

```
<html>

<head>

<meta http-equiv="Content-Type" content="text/html;
charset=UTF-8">



<script type="text/javascript">



function ValidMail() {

    let re = /^[\w-\.]+@[\w-]+\.[a-z]{2,4}$/i;

    let myMail = document.getElementById('email').value;
```

```
    let valid = re.test(myMail);

    if (valid) output = 'Адрес электронной почты введен
правильно!';

    else output = 'Адрес электронной почты введен
неправильно!';

    document.getElementById('message').innerHTML = output;

    return valid;

}

</script>

</head>

<body>

<p id="message" >Пожалуйста, заполните все поля формы!</p>

E-mail: <input id="email" name="email" type="text"
size="20" /><br />

<input name="button" type="submit" value="Проверить"
onClick="ValidMail();" />

</body>

</html>
```

ЗАДАНИЕ:

Создайте страницу обратной связи от клиента для интернет-магазина. Веб страница должна включать дизайн описанный в CSS файле. Форма обратной связи должна включать поля: имя, адрес электронной почты, поле для ввода отзыва. При нажатии на кнопку «Отправить» форма должна проверять правильность ввода данных, все поля должны быть заполнены, при выполнении всех условий появляется всплывающее сообщение «Ваш отзыв отправлен».

Пример: 
Оставьте отзыв
Имя
Ваше имя
Электронная почта
primer@donstu.ru
Заголовок
Заголовок
Текст отзыва
Отправить