

Работа с формами

работа с формами. продолжение

change

Событие `change` срабатывает по окончании изменения элемента.

Для текстовых `<input>` это означает, что событие происходит при потере фокуса.

Пока мы печатаем в текстовом поле в примере ниже, событие не происходит. Но когда мы перемещаем фокус в другое место, например, нажимая на кнопку, то произойдёт событие `change`

input

- в JS не только тег, но и событие. Генерируется при изменении текстового поля.

Событие input срабатывает каждый раз при изменении значения.

```
let x = document.addEventListener('input' , showMessage);
```

cut, copy, paste

Эти события происходят при вырезании/копировании/вставке данных.

Они относятся к классу `ClipboardEvent` и обеспечивают доступ к копируемому/вставляемому данным.

Мы также можем использовать `event.preventDefault()` для предотвращения действия по умолчанию, и в итоге ничего не скопируется/не вставится.

required

Для вызова ошибки, если поле не заполнено

```
<input name="email" required/>
```

Автоматический переход без табуляции

Чтобы упростить и ускорить ввод данных в форме, можно осуществлять автоматический переход к следующему полю, если текущее заполнено.

Получить максимальное кол-во введенных символов можно при помощи свойства `maxlength`, примененного к элементу:

```
inputTest.maxLength
```

Регулярные выражения

- это спецификация синтаксиса простого языка. При помощи регулярных выражений осуществляются методы поиска, замены и получения информации из строк. К методам работы с регулярными выражениями относятся `regexp.exec`, `regexp.test`, `string.match`, `string.replace`, `string.search`, `string.split`.

Регулярные выражения достаточно часто используются в формах для проверки введенной информации на соответствие требованиям.

Шаблоны ввода

В HTML5 для текстовых полей представлен атрибут `pattern`, указывающий регулярное выражение, с которым должно быть сопоставлено соответствующее значение в форме.

```
<input type="text" pattern="\d+" name="count">
```

*текстовое поле принимает только числа

pattern

Прочитать шаблон поля можно при помощи свойства pattern:

```
var pattern = document.forms[0].elements["count"].pattern;
```

Проверка допустимости

С помощью метода `checkValidity()` можно проверить, допустимо ли значение конкретного поля формы. Он доступен для всех элементов и возвращает `true`, если значение поля допустимо, и `false` в противном случае. При проверке используются условия шаблона `pattern`. Если условие не соответствует значению, либо поле без значение, то возвращается `false`.

checkValidity()

Вызвав метод `checkValidity()` для самой формы, можно проверить все её поля. Если все они допустимы, метод возвратит `true`, а если хотя бы одно из полей недопустимо - `false`.

```
if(document.forms[0].checkValidity()) {  
  
    //форма допустима  
  
} else {  
  
    //поле недоступно  
  
}
```

свойства validity

В то время, как метод `checkValidity()` просто сообщает, допустимо ли значение поля, свойство `validity` указывает точную причину, почему оно допустимо или нет. У этого объекта есть следующие свойства логического типа:

- `patternMismatch` - true если значение не соответствует заданному атрибуту `pattern`.
- `rangeOverflow` - true, если значение больше чем `max`
- `rangeUnderflow` - true, если значение меньше чем `min`
- `tooLong` - true, если значение содержит больше чем допускает свойство `maxlength`
- `valueMissing` - true, если поле, отмеченное как обязательное, пусто

Задание 1

Создать калькулятор расхода и дохода. Пользователь вводит число в поле ввода и выбирает с помощью radio кнопки доход это или расход. На нажатие кнопки нужно рассчитать текущее значение счета, а также отрисовать столбики диаграммы дохода и расхода разными цветами. При нажатии кнопки поля ввода очищаются

Задание 2

Создать функцию, которая создает модальное окно с вопросом о имени и возрасте пользователя. Поля обязательные для заполнения. В окне 2 кнопки - отмена и добавить. При нажатии отмены, окно закрывается, при нажатии на добавить, проверяется, если возраст больше 18, то на странице появляется новый элемент списка с именем и возрастом пользователя, если нет выводится красное текстовое сообщение.

Функция вызывается на нажатие кнопки - Добавить пользователя

Задание 3

Создать собственную валидацию поля емейл:

При потере фокуса или отправке формы, проверять с помощью регулярного выражения значение поля email и показывать ошибку, если оно не соответствует

Задание 4

Создать функцию, которая создает геометрическую фигуру, данные о фигуре из формы - можно выбрать форму фигуры, ввести цвет, размеры, положение на странице. Если выбран круг, то появляется поле ввода радиуса, если квадрат, то сторона, если прямоугольник, то 2 поля размера.