# Валидация с JavaScript

# novalidate - сообщает браузеру о том, что стандартные сообщения об ошибке не нужны.

<form class="form" novalidate>

<input class="form\_\_input" type="url" placeholder="Адрес" required>

<button class="form\_\_submit">Войти</button>

</form>

# ValidityState - встроенный объект для сверки данных. Он присутствует в каждом поле ввода и тесно связан с его html-атрибутами. Само свойство называется validity.

# Свойство validity — это объект из 11 свойств с булевыми значениями.

# 

* valueMissing — принимает true, когда обязательное поле пустое;
* typeMismatch — принимает true, когда ввели неправильные значения данных для атрибута type. Это круто работает в связке с type="email" и type="url";
* tooLong — принимает false, когда количество символов не превышает значение атрибута maxlength. А true не существует в современных браузерах. Невозможно ввести больше символов, чем указано в maxlength;
* tooShort — принимает true, когда количество символов не превышает значение атрибута minlength.
* validity — итоговое решение проверки данных. Если во всех других 10 свойствах значения корректны, поле ввода валидно свойство valid приобретает значение true

# validationMessage - есть у всех полей ввода. В нём записан текст сообщения об ошибке.

# Чтобы добавить это сообщение в свой код, нужно передать функции showInputError свойство validationMessage

<span id="email-input-error" class="form\_\_input-error"></span>

const formElement = document.querySelector('.form');

const formInput = formElement.querySelector('.form\_\_input');

const formError = formElement.querySelector(`#${formInput.id}-error`);

*// Передадим текст ошибки вторым параметром*

const showInputError = (element, errorMessage) => {

element.classList.add('form\_\_input\_type\_error');

*// Заменим содержимое span с ошибкой на переданный параметр*

formError.textContent = errorMessage;

formError.classList.add('form\_\_input-error\_active');

};

const hideInputError = (element) => {

element.classList.remove('form\_\_input\_type\_error');

formError.classList.remove('form\_\_input-error\_active');

*// Очистим ошибку*

formError.textContent = '';

};

const isValid = () => {

if (!formInput.validity.valid) {

*// Передадим сообщение об ошибке вторым аргументом*

showInputError(formInput, formInput.validationMessage);

} else {

hideInputError(formInput);

}

};

Можно не придумывать свой текст ошибки, а использовать стандартный.

# Встроенная браузерная валидация

# type - популярные примеры — значения url и email:

* url — значение поля ввода начинается с http:// или https:// и содержит минимум один символ после;
* email — значение поля содержит минимум один символ до @ и один после.

# minlength и maxlength - задают минимальное и максимальное количество символов в поле ввода

<form class="form">

<input class="form\_\_input" type="url" placeholder="Адрес">

<input class="form\_\_input" type="email" placeholder="Email">

<input class="form\_\_input" type="text" placeholder="Имя" minlength="2">

<button>Отправить</button>

</form>

# псевдоклассы в HTML - подсвечивают поле в момент появления ошибки

* :valid — указывает, что введены корректные данные;
* :invalid — указывает, что данные некорректны;
* :checked — применяет стили к отмеченным чекбоксам type="checkbox" или радиокнопкам type="radio";
* :not — показывает элементы, которые не отмечены как :checked.

.form\_\_input:invalid {

border: 1px solid red; */\* красная рамка для некорректных данных \*/*

}

.form\_\_input:valid {

border: 1px solid green; */\* зелёная - для корректных \*/*

}

# required - делает данное поле обязательным для заполнения

<form class="form">

<input class="form\_\_input" type="url" placeholder="Адрес" required>

<input class="form\_\_input" type="email" placeholder="Email" required>

<button>Отправить</button>

</form>

При попытке отправить форму с пустыми полями появляется сообщение об ошибке.