**FormData**

Объекты [**FormData**](https://xhr.spec.whatwg.org/#interface-formdata) используются, чтобы взять данные из HTML-формы и отправить их с помощью **fetch** или другого метода для работы с сетью. Методы для работы с сетью, например fetch, позволяют указать объект FormData в свойстве тела запроса **body**. Он будет соответствующим образом закодирован и отправлен с заголовком Content-Type: form/multipart.

Конструктор:

let formData = new FormData([form]);

Если передать в конструктор элемент HTML-формы form, то создаваемый объект автоматически прочитает из неё поля.

<form id="**formElem**">

<input type="text" name="name" value="John">

<input type="text" name="surname" value="Smith">

<input type="submit">

</form>

<script>

formElem.**onsubmit** = async (e) => {

e.preventDefault();

let response = await fetch('/article/formdata/post/user', {

method: 'POST',

body: **new FormData**(formElem)

});

let result = await response.json();

alert(result.message);

};

</script>

Мы можем также создать объект вообще без формы и затем добавить к нему поля с помощью **методов**:

* formData.append(name, value) – добавляет к объекту поле с именем name и значением value
* formData.append(name, blob, fileName) – добавляет поле, как будто в форме имеется элемент <input type="file">, третий аргумент fileName устанавливает имя файла (не имя поля формы), как будто это имя из файловой системы пользователя,
* formData.set(name, value) - добавляет к объекту поле с именем name и значением value, как и append, только сначала удаляет предыдущие поля с таким же именем, а append – нет. В этом их единственное отличие.
* formData.set(name, blob, fileName) - аналогично

Еще **методы**:

* formData.delete(name) – удаляет поле с заданным именем name,
* formData.get(name) – получает значение поля с именем name,
* formData.has(name) – если существует поле с именем name, то возвращает true, иначе false

### Отправка формы с файлом

Объекты FormData всегда отсылаются с заголовком Content-Type**: form/multipart**, этот способ кодировки позволяет отсылать файлы. Таким образом, поля <input type="file"> тоже отправляются, как это и происходит в случае обычной формы. Пример такой формы:

<form id="formElem">

<input type="text" name="firstName" value="John">

Картинка: <input type="file" name="picture" accept="image/\*">

<input type="submit">

</form>

<script>

formElem.onsubmit = async (e) => {

e.preventDefault();

let response = await fetch('/article/formdata/post/user-avatar', {

method: 'POST',

body: new FormData(formElem)

});

let result = await response.json();

alert(result.message);

};

</script>