В этой статье разберем основы того что вам пригодится при разработке виджетов в AmoCrm. А так же в статье есть задачи выполнив которые вы лучше поймете материал и прокачаете свой скилл. Рекомендуется все ваши задачи хранить в вашем репозитории и использовать для них **git** соответственно. Это сохранит ваши разработки и возможно пригодится в будущем.

### **JQuery - верните мой 2007**

JQuery - JavaScript библиотека. на данный момент уже считается устаревшей, но все еще используется во многих проектах. Любую информацию для работы с ней можете найти в [документации](https://api.jquery.com/). И [гугл](https://www.google.com/search?newwindow=1&ei=c45RXOqmIujgrgSn-qmAAQ&q=jquery+help+me) никто не отменял, почти.

***jQuery*** *— библиотека JavaScript, фокусирующаяся на взаимодействии JavaScript и HTML. Библиотека jQuery помогает легко получать доступ к любому элементу DOM, обращаться к атрибутам и содержимому элементов DOM, манипулировать ими. Также библиотека jQuery предоставляет удобный API для работы с AJAX. Разработка jQuery ведётся командой добровольцев на пожертвования.*

Для чего она нужна в работе с AmoCRM? Клиентская часть AmoCRM использует JQuery, что позволяет использовать эту библиотеку и в виджетах, для упрощения разработки.

Начнем с самых простых вещей и позже, в следующих постах, перейдем к использованию полученных знаний на кейсах, которые были в нашей работе.

Задание:

Есть некая форма

<form class="awesome\_form">

<div class="text\_input">

<input name="CVF[1677]">

</div>

<div class="text\_input">

<input name="CVF[1234]">

</div>

<div class="text\_input text\_input--textarea">

<textarea name="CVF[1677]"></textarea>

</div>

<button type="submit">submit</button>

</form>

Необходимо с использованием JQuery скрыть все **input** и **textarea** и подставить на их места новые **input** с классом **awesome\_input**. При вводе значений в новые поля нужно подставлять значения в соответствующие, скрытые, поля. Форму оформить с помощью препроцессора SCSS(SASS) и сделать валидацию формы (проверьте чтобы не было пустых полей) с помощью JQuery. Все собрать с помощью Webpack.

По выполнению задания отправьте [мне](http://t.me/bessamu) решение и я дам вам новое задание.

### **Ajax**

Ajax - методология которая позволяет обращаться к серверу без перезагрузки страницы, для получения или отправки различных данных бэку. С использованием этой методологии веб-приложение выглядит как десктопное.

***AJAX*** *(Asynchronous Javascript and XML) — подход к построению интерактивных пользовательских интерфейсов веб-приложений, заключающийся в «фоновом» обмене данными браузера с веб-сервером. В результате, при обновлении данных веб-страница не перезагружается полностью, и веб-приложения становятся быстрее и удобнее. По-русски иногда произносится по существующей аналогии у имени Ajax — Аякс, но у аббревиатуры AJAX нет устоявшегося варианта на кириллице.*

Это основная часть которая понадобиться вам при разработке на клиентской части. Рекомендуем внимательно изучить документацию.

* [Документация JQuery](https://api.jquery.com/)
* [Документация SCSS/SASS](https://sass-lang.com/)

### **Виджеты в AmoCRM**

Виджет - набор кода дополняющий стандартный функционал, в данном случае функционал AmoCRM. Как говорилось в этой [статье](https://telegra.ph/AmoCRM-nachalo-raboty-02-01) к каждому бизнесу нужен свой подход. И виджеты это часть индивидуального подхода к бизнесу.

Для того чтобы понимать как устроены виджеты в Amo нужно внимательно изучить [документацию](https://www.amocrm.ru/developers/content/integrations/script_js).

Также изучить [принцип загрузки](https://www.amocrm.ru/developers/content/integrations/upload) виджета в Amo.

### **Twig**

И в дополнение, при создании виджетов вам понадобиться использовать шаблонизатор twig.

***Twig*** *— компилирующий обработчик шаблонов с открытым исходным кодом, написанный на языке программирования PHP.*

По сути, вы просто верстаете HTML разметку и можете подставлять различные переменные, как если бы использовали PHP. В случае разработки виджетов вам нужно будет передавать переменные из js файла.

**someTwigFile.twig**

<body>

{% if isBool %}

<div>

{{ variable }}

</div>

{% endif %}

</body>

Для понимания, что и как обязательно изучить [документацию](https://twig.symfony.com/doc/2.x/) twig.

Задание:

Создать виджет который должен, в случае если название поля длинное и не помещается целиком, например, полное название поля: "Какое-то очень длинное название поля", а отображается только: "Какое-то очень длинное н", нужно переносить слова на новую строку, чтобы было видно название целиком.

Выполненную работу отправлять в [Telegram](http://t.me/bessamu), а именно архив с виджетом. В ответ получите новое задание, заготовлено три варианта, чем больше выполните тем лучше. После выполнения каждого варианта присылайте ваше решение.

Материалы:

* [Документация по виджетам AmoCRM](https://www.amocrm.ru/developers/content/integrations/script_js)
* [Документация по Twig](https://www.amocrm.ru/developers/content/integrations/script_js)