API конструктора продукции

Данный документ объясняет способы управления конструктором и обработки его событий. Конструктор можно запускать с разными параметрами на разных страницах сайта, можно давать ссылку на полноэкранный конструктор с разными параметрами для разных клиентов, можно управлять конструктором скриптами вашего сайта, и можно обрабатывать события конструктора и делать на их основе различные действия на вашем сайте, такие как добавление заказа в корзину сайта (это нужно программировать).

В данном документе описана работа только с модулем конструктора. У сервера конструктора нет API. Вы не можете например получить от сервера хранящийся там заказ.

Запуск конструктора с определенными параметрами

Конструктор можно запускать с разными параметрами, другими словами – с разными уже выбранными предметами, моделями, цветами и т.д. Например, в разделе красных футболок лучше показать конструктор уже с выбранной красной футболкой, а не с любым предметом.

Указывать параметры можно в хеше url-адреса **конструктора** или его **фрейма**. Просто дописываете после адреса конструктора хеш # с нужными параметрами, например:

https://cosuv.ru/app/40#**base=1;color=1**

Конструктор поддерживает следующие параметры:

base – номер предмета;

model – номер модели;

color – номер цвета предмета;

side – номер стороны;

sex – номер пола (0 – мужской, 1 – женский);

age – номер возраста;

size – номер строки размера в списке размеров;

method – номер метода печати;

lang – код языка (rus – Русский, eng – English, …) или если вы добавили несколько одинаковых языков, то номер языка (0, 1, 2, …);

compos – номер заготовки композиции;

btn-base, btn-model, btn-color, btn-side, btn-sex, btn-age, btn-size, btn-method, btn-image, btn-figure, btn-text – служат для скрытия соответствующих кнопок в меню: 0 – скрыть, 1 – показать обратно.

При написании этих параметров важен порядок их следования в url-адресе:

Предметы → Модели → Цвета → Стороны

Если вы сначала пишете выбор второго по счету предмета, то это значит, что модели вы будете выбирать уже у этого предмета. Поэтому если вы напишете наоборот, сначала выбор модели и уже потом предмета, то выбор может быть непредсказуемым (фактически будет выбран второй предмет с первой его моделью).

Несколько параметров отделяются точкой с запятой. Например:

https://cosuv.ru/app/40#**base=1;model=5;btn-base=0;btn-model=0**

Эта строка означает, что вы хотите запустить конструктор со вторым предметом **(счет начинается с нуля)**, с шестой моделью этого второго предмета, и хотите скрыть кнопки выбора предметов и моделей, чтобы клиент не выбирал другие или чтобы ему было проще ориентироваться в конструкторе.

Для встроенного в страницу конструктора код будет выглядеть соответственно:

<iframe src="https://cosuv.ru/app/40#**base=1;model=5;btn-base=0;btn-model=0**" style="width:100%;height:600px;" frameborder="0" allowfullscreen></iframe>

То есть дописывается только адрес в кавычках.

Если вы будете дописывать что-то к адресу вашей веб-страницы, то конструктор на это реагировать не будет. Конструктор никак не связан со страницей когда размещён в ней.

Для старта конструктора с разными заготовленными композициями, используйте параметр compos. Прежде чем выбрать композицию, выберите нужные предметы и модели, на которых она появится, то есть параметр compos пишите после параметров выбора предмета, модели и прочего.

Конструктор реагирует на смену своего адреса и меняет на ходу параметры соответственно, возможно вам пригодится это. Однако лучше управлять конструктором через сообщения.

Управление конструктором через сообщения

(Данный подраздел будет понятен JavaScript-программистам и верстальщикам)

Конструктором можно управлять на ходу, во время того, как пользователь взаимодействует со страницей вашего сайта. Например, пользователь выбирает на странице сайта фильтр «только женские», и конструктор может отфильтровать только женские модели, без перезагрузки страницы сайта или конструктора.

Чтобы задавать конструктору параметры во время его работы, нужно посылать ему сообщения функцией postMessage через JavaScript вашего сайта.

Пример такого сообщения:

frame.contentWindow.postMessage('cosuv-api;base=0;model=2','*');

Переменная frame – это найденный на странице фрейм конструктора (HTMLElement).

Начинать сообщение нужно с префикса **cosuv-api;** который явно отсеивает сообщения для данных целей от других сообщений.

Вместо **base=0;model=2** вы можете писать любые параметры. Правила их написания те же, что описаны выше в разделе «Запуск конструктора с определенными параметрами».

Несмотря на то, что конструктор будет реагировать и на одиночные параметры, во многих случаях надежнее посылать ему серии параметров. Например если вы хотите выбрать вторую модель первого предмета, то посылайте два параметра – предмет и затем модель, а не просто модель.

Обработка событий конструктора

(Данный подраздел будет понятен JavaScript-программистам и верстальщикам)

Сейчас в конструкторе реализовано только одно событие – событие отправки заказа. Оно позволяет добавлять заказ в корзину вашего сайта. Вы можете использовать его и в других целях. Обработка вами этого события не отменяет добавление заказа в кабинет управления. Вы в любом случае не потеряете заказ и никак его не измените, это событие служит просто для удобства клиентов и для оплаты заказа через интерфейс вашего сайта.

Чтобы обработать это событие, добавьте в ваш JavaScript слушатель события «message». Пример:

window.addEventListener('message',function(e){ console.log(e) });

Данный код при заказе в конструкторе выведет в консоль объект события. В этом объекте есть ключ **data**, в котором записаны все существенные параметры заказа в виде строки JSON. Вам нужно будет распарсить этот JSON в объект параметров (ассоциативный массив). В ключе **vars** в этих параметрах указаны номера выбранных предмета, модели и так далее. В ключе **readable** описаны их названия, в ключе **readableUserLang** – названия на языке клиента. В ключе **fields** содержится массив заполненных клиентом полей формы заказа. В ключе **thumb** – ссылка на уменьшенное изображение заказа. В ключе **sum** – сумма заказа.

Событие message может быть отправлено не только конструктором. Проверяйте в своем скрипте то, что данные из этого события являются JSON-строкой и после парсинга этой строки проверяйте, что ключ type имеет значение cosuvOrder. В будущем могут появиться другие события конструктора с другим значением type.

В следующем примере скрипта вы можете наглядно посмотреть, как обрабатывается событие заказа, и как отправлять данные заказа на сервер: https://cosuv.ru/js/order-to-server.js

---

Если вы заметили ошибки в данной инструкции или в работе API, напишите пожалуйста на support@cosuv.ru