

# Настройка для любой витрины

Настройка Smart Contract на любую витрину происходит через JavaScript код, и может быть выполнена практически любым специалистом.

В данной статье рассмотрим настройку на сайт <https://avilon.ru/catalog/cars/new/>

## Настройки детальной страницы

Детальная страница: <https://avilon.ru/catalog/cars/new/4407759/>

Мы будем интегрироваться в правый блок детальной страницы

СТОИМОСТЬ ПО ПРАЙСУ СКИДКА 11%

~~1 123 900 ₺~~ - 130 000 ₺

✓ **ОНЛАЙН-РЕЗЕРВ** - 10 000 ₺

^ **КРЕДИТ** - 100 000 ₺

- 100 000 ₺ ?


^ **КАСКО** - 20 000 ₺

- 20 000 ₺ ?

СТОИМОСТЬ СО СКИДКОЙ

**993 900 ₺**

**Получить персональное предложение**

 [Скачать брошюру](#)

ПРЕДОПЛАТА ПРИ РЕЗЕРВЕ: 25 000 ₺

[Зарезервировать онлайн](#)

- Предоплата фиксирует цену на авто
- Возврат предоплаты без комиссии
- Персональные условия при покупке

Нам необходимо проинспектировать код этого блока, для получения цен и выгод

```
▼ <div class="row" itemprop="offers" itemscope itemtype="https://schema.org/AggregateOffer"> (flex) == $0 .r
  <meta itemprop="highPrice" content="1123900">
  <meta itemprop="lowPrice" content="993900">
  <meta itemprop="priceCurrency" content="RUB">
  <meta itemprop="offerCount" content="1">
  ▼ <div class="col-12"> }
    ▼ <div class="car-detail__aside-price"> *
      ▶ <div class="car-price">...</div>
      ▶ <div class="car-detail__aside-accordion car-detail__aside-accordion_fixed">...</div> }
      ▼ <div class="car-detail__aside-accordion js-accordion is-open car-detail__checkbox-block"> *,
        ▶ <div class="car-detail__aside-accordion-title js-accordion-title">... *;
        </div> (flex) }
        ▼ <div class="car-detail__aside-accordion-content js-accordion-content" style="display: block;"> di
          ▼ <label class="label-cbx car-detail__aside-accordion-label"> ln
            <input id="credit-cbx4" type="checkbox" checked data-id="4890282" data-amount="100000" data-credit="1" data-tradein="0" data-kasko="0" value="Y" name="credit"> .r
            ▶ <div class="checkbox">...</div> }
            ▶ <span class="checkbox-text">...</span> ln
          </label> ht
        </div>
        </div>
        ▶ <div class="car-detail__aside-accordion js-accordion is-open car-detail__checkbox-block">...</div>
        ▶ <div class="car-price car-price_total">...</div>
        ▶ <div class="car-detail__aside-cntrl p-0">...</div>
        ▶ <div class="car-detail__aside-link">...</div>
        ▶ <div class="car-price car-price_row">...</div>
        <a href="/order/4407759/?option[]=4890282&option[]=4890283" id="reserved" class="button_style button_style4 car-detail__aside-button"> }
        Зарезервировать онлайн</a> bc
        ▶ <ul class="car-detail__aside-list">...</ul>
      </div>
```

На данной витрине не передаётся VIN номер автомобиля, поэтому мы будем использовать URL для определения идентификатора автомобиля.

## WidgetId

Получаем его со вкладки виджетов. Код будет выглядеть так

```
InitSmartContract({
  widget_id: "widgetId"
})
```

## Идентификатор автомобиля

Так как у нас нет информации о VIN'е, то будет привязываться к ссылке на автомобиль, для этого определим функцию для получения идентификатора автомобиля.

```
detail: {
```

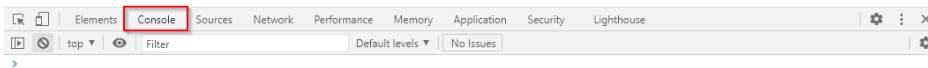
```
identity: function() {  
  return window.location.href.split('#')[0];  
}  
}
```

Мы используем переменную `window.location.href`, которая вернёт нам ссылку на страницу, например `https://avilon.ru/catalog/cars/new/4407759/` и обрезаем всё, что находится после символа `#` в строке.

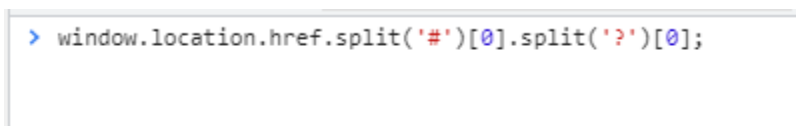
Если у нас используется ЧПУ (Как в примере выше), то правильным будет так же вырезать всё, что находится после символа `?`, т.к. параметры автомобиля зашиты в ссылку, тогда код будет выглядеть таким образом

```
detail: {  
  identity: function() {  
    return window.location.href.split('#')[0].split('?')[0];  
  }  
}
```

Проверить, что мы получили правильную ссылку очень просто. Так можно проверить любой код. Откроем консоль браузера (F12 для Google Chrome) и перейдём на вкладку Console



Вставим наш код в окно ввода



И нажмём клавишу `Enter` для выполнения, чтобы увидеть результат.



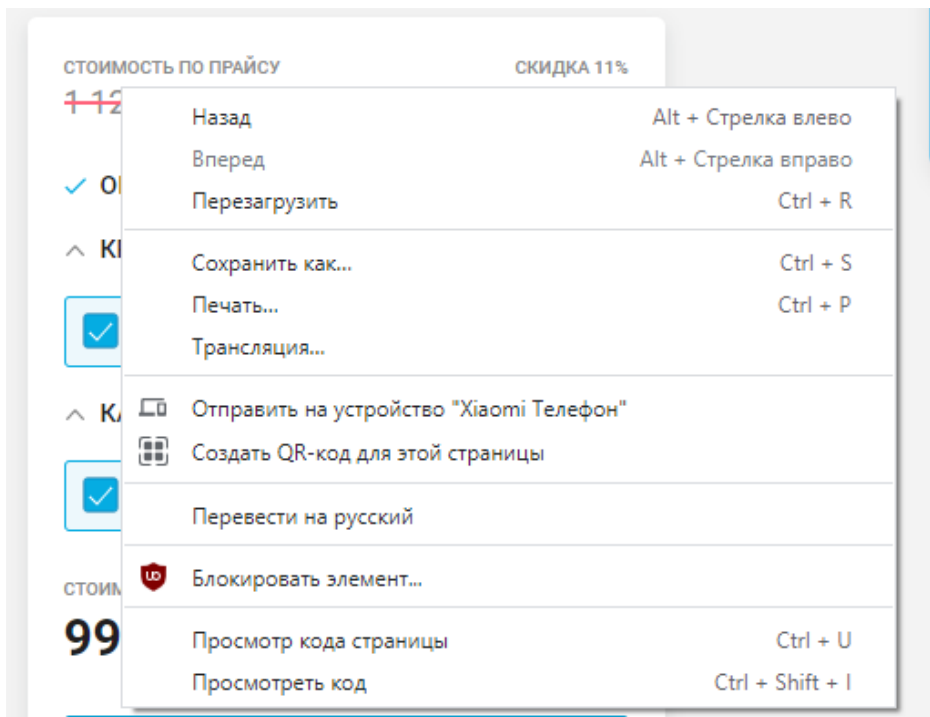
Мы получили строку со ссылкой на автомобиль. Можем попробовать выполнить указанный код, на странице с `utm` меткой, для этого перейдём на страницу `https://avilon.ru/catalog/cars/new/4407759/?utm_source=test` и выполним код в консоли

```
> window.location.href.split('#')[0].split('?')[0];  
< "https://avilon.ru/catalog/cars/new/4407759/"  
> |
```

Мы увидим, что параметры, не влияющие на идентификатор автомобиля, успешно удалились.

## Цена автомобиля

Нам необходимо получить максимальную цену за автомобиль. Для этого найдём максимальную цену на странице и откроем её исходный код



Мы увидим примерно следующий код

```
<meta itemprop="highPrice" content="1123900">  
<meta itemprop="lowPrice" content="993900">  
<meta itemprop="priceCurrency" content="RUB">  
<meta itemprop="offerCount" content="1">  
▼<div class="col-12">  
  ▼<div class="car-detail__aside-price">  
    ▼<div class="car-price">  
      ▼<div class="car-price__row"> flex  
        ▼<div class="car-price__col">  
          <div class="car-price__title">Стоимость по прайсу </div>  
          ▼<div id="base-price" class="car-price__value car-price__value_old">  
            " 1 123 900 P" == $0  
          ...</div>
```

Естественно, он может и будет разным в зависимости от платформы. Попробуем получить данную цену, для этого используем функцию `document.querySelector()`, которая позволит получить нам DOM элемент.

Используйте `#` для ID и `.` для классов. Т.е. в приведённом выше примере, если мы решим получать элемент по ID = `base-price`, нам нужно составить такой запрос

```
document.querySelector('#base-price')
```

А если по классам, то

```
document.querySelector('.car-price__value')
```

Допустим, мы будем обращаться по ID, выполним код и проверим что мы получили

```
> document.querySelector('#base-price')
< <div id="base-price" class="car-price__value car-price__value_old">
  " 1 123 900 P"
  ::after
</div>
```

Теперь нам нужно извлечь стоимость, для этого используем свойство `.innerText` у данного DOM элемента.

```
> document.querySelector('#base-price').innerText
< "1 123 900 P"
```

Теперь нам осталось вырезать из этого текста всё кроме цифр, используем регулярное выражение для этого `.replace(/\D/g, '')`, который заменит все не цифры на пустые строки.

```
> document.querySelector('#base-price').innerText.replace(/\D/g, '')
< "1123900"
```

И приведём данное значение к целому числу с помощью функции `parseInt`

```
> parseInt(document.querySelector('#base-price').innerText.replace(/\D/g, ''))
< 1123900
```

Теперь, когда у нас есть цена мы можем использовать её при передаче параметров в функцию настройки, например так

```
InitSmartContract({
  detail: {
    price: function() {
      return parseInt(document.querySelector('#base-price').innerText.replace(/\D/g, '')) || 0;
    }
  }
});
```

```
    }  
  }  
})
```

Мы так же используем конструкцию `|| 0` в конце, чтобы передать значение 0 в цену, если нам не удалось расшифровать её.

Как правило, коды детальных страниц совпадают, и кода выше будет достаточно, но периодически случается так, что на некоторых витринах автомобили со скидками и без скидок имеют разный код. Например, на сайте выше есть 2 представления.

## Что делать если цена указана по-разному?

Например, цена на нашем сайте для автомобиля <https://avilon.ru/catalog/cars/new/4630404/> указана совершенно по-другому

```
<meta itemprop="highPrice" content="1127575.1">  
<meta itemprop="lowPrice" content="1127580">  
<meta itemprop="priceCurrency" content="RUB">  
<meta itemprop="offerCount" content="1">  
▼<div class="col-12">  
  ▼<div class="car-detail__aside-price">  
    ▼<div class="car-price car-price_total">  
      ▼<div class="car-price__row"> flex  
        ▼<div class="car-price__col">  
          <div id="final-price-title" class="car-price__title">Стоимость</div>  
          ▼<div class="car-detail__aside-price-full car-price__value car-price__value_total" data-dis="1127575.1" data-baseprice="1127575.1" data-price="1127580">  
            <span>1 127 575</span> == $0  
            " p "  
          </div>
```

Она обёрнута в дополнительный `span`, и элемента с `ID = base-price` нет на странице вообще.

В таком случае, необходимо найти что-то общее, или писать более сложный код. Например, я вижу, что в коде на той и другой странице есть элемент

```
<meta itemprop="highPrice" content="1127575.1"> ==
```

Который общий для всех страниц. Сделаем его обработку `document.querySelector('[itemprop="highPrice"]').content`

```
> document.querySelector('[itemprop="highPrice"]').content  
< "1123900"
```

Как видно выше, данный код может содержать точку, обрежем его

```
> document.querySelector('[itemprop="highPrice"]').content.split('.')[0]
< "1127575"
> |
```

И приведём к целому числу

```
> parseInt(document.querySelector('[itemprop="highPrice"]').content.split('.')[0]) || 0
< 1123900
```

Тогда, итоговый код, который будет работать для всех страниц, будет выглядеть вот так:

```
InitSmartContract({
  detail: {
    price: function() {
      return
      parseInt(document.querySelector('[itemprop="highPrice"]').content.split('.')[0]) || 0;
    }
  }
})
```

Если же потребуется обработать несколько различных элементов цен, то обычно потребуется сделать код, примерно такого содержания.

```
const prices = []; // Инициализируем массив цен

const element1 = document.querySelector('selector-1'); // Ищем элемент 1
const element2 = document.querySelector('selector-2'); // Ищем элемент 2

const parsePrice = price => parseInt(price.split('.')[0].replace(/\D/g, ''))
|| 0; // Определяем функцию, которая нам по значению извлечёт нормально цену

if( element1 ) prices.push(parsePrice(element1.innerHTML)); // Добавляем цену
из 1 элемента
if( element2 ) prices.push(parsePrice(element2.innerHTML)); // Добавляем цену
из 2 элемента

if( prices.length === 0 ) return 0; // Если цен не найдено, возвращаем 0

return Math.max(...prices); // Если цены найдены, возвращаем максимальную
```

## Генератор кнопки

Последний штрих, для минимальной настройки виджета — это указание кода, который будет отвечать за отображение кнопки.

Мы возьмём данную кнопку за основу

Получить персональное предложение

Посмотрим её код

```
▼ <div class="car-detail__aside-cntrl p-0">  
  <a data-toggle="modal" data-href="/ajax/modal/personal-offer.php" data-  
    products="0" data-sold="324" data-sold-used="325" href="#modal" data-  
    data="{\"name\": \"Volkswagen Polo 1.6 MPI MT (90 л.с.) Respect + Зимний + К  
    омфорт Белый\", \"id\": \"4407759\", \"markid\": \"2314704\", \"modelid\": \"2316374\", \"typ  
    е\": \"Получить персональное предложение\", \"ID\": \"4407759\", \"USED\": false, \"PROD  
    UCT_TYPE\": \"CARS\", \"ITEM\": \"Volkswagen Polo\", \"BADGE\": [{\"NAME\": \"Лучшая цен  
    а\", \"CODE\": \"SPECIAL_OFFER\", \"CLASS\": \"lib__badge--cars_special_offer\", \"BOTT  
    OM_CLASS\": \"lib__bottom_badge--cars_special_offer\", \"HINT\": \"Действуют особ  
    ые условия на покупку данного автомобиля.\", \"COLOR_BACK\": \"#ff9800\", \"COLOR  
    _TEXT\": \"#fff\", \"BADGE_TYPE\": \"SPEC\"}, {\"NAME\": \"В резерве\", \"CODE\": \"RESERVE  
    D\", \"CLASS\": \"lib__badge--cars_reserved\", \"BOTTOM_CLASS\": \"lib__bottom_badge  
    --cars_reserved\", \"HINT\": \"Данный заказ находится в резерве. Отправьте зая  
    ву, и мы уведомим вас, если он снова поступит в продажу, либо подберем  
    вариант максимально близкий по характеристикам и по привлекательной цен  
    е.\", \"COLOR_BACK\": \"#666666\", \"COLOR_TEXT\": \"#fff\", \"BADGE_TYPE\": \"LOGISTICS\"}  
    ], {\"NAME\": \"Выгодно в кредит\", \"CODE\": \"ADVANTAGE_CREDIT\", \"CLASS\": \"lib__badg  
    e--cars_advantage_credit\", \"BOTTOM_CLASS\": \"lib__bottom_badge--cars_advant  
    age_credit\", \"HINT\": \"\", \"COLOR_BACK\": \"#339704\", \"COLOR_TEXT\": \"#FFF\"}], \"DESC  
    RIPTION\": \"Лифтбек, VI, 2021 года выпуска, двигатель 1598 см3, 90 л.с., б  
    ензиновый, Механика, передний привод, цвет белый (Белый \\\"Pure white\\\"),  
    темный салон (Черный), тканевый (Черный Titanium / Черный / Черный / Сер  
    ый)\", \"sumDiscount\": 0, \"discount\": \"4890282;4890283\", \"sum\": \"993900\", \"old_p  
    rice\": \"1123900\"}\" class="button_style button_style1 h-auto js-open-perso  
    nalOffer car-detail__aside-button">Получить персональное предложение</a>  
</div>
```

И опишем функцию, которая генерирует новую кнопку

```
InitSmartContract({  
  detail: {  
    generator: function(text) {  
      // В параметре text будет находиться текст кнопки  
  
      const original = document.querySelector('.car-detail__aside-cntrl'); //  
      Получаем текущую кнопку (Мы получили её wrapper, что бы всё скопировалось  
      красиво)  
      const clone = original.cloneNode(true); // Клонировем эту кнопку, что бы  
      заменить в неё всё лишнее  
  
      const cloneLink = clone.querySelector('a'); // Находим элемент А,  
      который отвечает за кнопку и удаляем лишнее  
  
      cloneLink.classList.remove('js-open-personalOffer'); // Удаляем класс,  
      отвечающий за открытие модального окна  
  
      const dataset = cloneLink.dataset; // Получаем data атрибуты  
      for (let key in dataset) { // Идём по всем data атрибутам
```



```

        cloneLink.removeAttribute("data-" + key.split(/(?=[A-Z])/).join("-")
        .toLowerCase()); // Удаляем все data атрибуты, что бы у нас не открывалось
        модельное окно
    }

    cloneLink.innerText = text; // Устанавливаем текст кнопки на полученный
    из Smart Contract

    original.before(clone); // Вставляем новую кнопку до старой
    original.remove(); // Удаляем оригинальную кнопку (Этого можно не
    делать)

    return [ cloneLink ]; // Возвращаем элемент кнопки, что бы при нажатии
    открылся Smart Contract
    }
    }
    })

```

В целом, после того как мы указали 3 параметра выше, код УЖЕ можно установить на витрину. Общий код будет иметь следующий формат:

```

InitSmartContract({
  widget_id: "widgetId",
  detail: {
    identity: function() {
      return window.location.href.split('#')[0].split('?')[0];
    },
    price: function() {
      return
      parseInt(document.querySelector('[itemprop="highPrice"]').content.split('.')[
      0]) || 0;
    },
    generator: function(text) {
      // В параметре text будет находиться текст кнопки

      const original = document.querySelector('.car-detail__aside-cntrl'); //
      Получаем текущую кнопку (Мы получили её wrapper, что бы всё скопировалось
      красиво)
      const clone = original.cloneNode(true); // Клонировем эту кнопку, что бы
      заменить в неё всё лишнее

      const cloneLink = clone.querySelector('a'); // Находим элемент А,
      который отвечает за кнопку и удаляем лишнее

      cloneLink.classList.remove('js-open-personalOffer'); // Удаляем класс,
      отвечающий за открытие модального окна

      const dataset = cloneLink.dataset; // Получаем data атрибуты
      for (let key in dataset) { // Идём по всем data атрибутам
        cloneLink.removeAttribute("data-" + key.split(/(?=[A-Z])/).join("-")
        .toLowerCase()); // Удаляем все data атрибуты, что бы у нас не открывалось
        модельное окно
      }

      cloneLink.innerText = text; // Устанавливаем текст кнопки на полученный
      из Smart Contract
    }
  }
});

```

```

        original.before(clone); // Вставляем новую кнопку до старой
        original.remove(); // Удаляем оригинальную кнопку (Этого можно не
        делать)

        return [ cloneLink ]; // Возвращаем элемент кнопки, что бы при нажатии
        открылся Smart Contract
    }
}
})

```

## Не забываем обернуть указанный выше код в загрузку скрипта Smart Contract

```

var s = document.createElement('script');
s.src = "https://script.smart-contract.digital/bundle.js?" +
Math.random(10000);
s.onload = function(){
    InitSmartContract({
        widget_id: "widgetId",
        detail: {
            identity: function() {
                return window.location.href.split('#')[0].split('?')[0];
            },
            price: function() {
                return
                parseInt(document.querySelector('[itemprop="highPrice"]').content.split('.')[
0]) || 0;
            },
            generator: function(text) {
                // В параметре text будет находиться текст кнопки

                const original = document.querySelector('.car-detail__aside-cntrl');
                // Получаем текущую кнопку (Мы получили её вapper, что бы всё скопировалось
                красиво)
                const clone = original.cloneNode(true); // Клонировем эту кнопку, что
                бы заменить в неё всё лишнее

                const cloneLink = clone.querySelector('a'); // Находим элемент А,
                который отвечает за кнопку и удаляем лишнее

                cloneLink.classList.remove('js-open-personalOffer'); // Удаляем
                класс, отвечающий за открытие модального окна

                const dataset = cloneLink.dataset; // Получаем data атрибуты
                for (let key in dataset) { // Идём по всем data атрибутам
                    cloneLink.removeAttribute("data-" + key.split(/(?=[A-Z])/).join("-"
                    ).toLowerCase()); // Удаляем все data атрибуты, что бы у нас не открывалось
                    модельное окно
                }

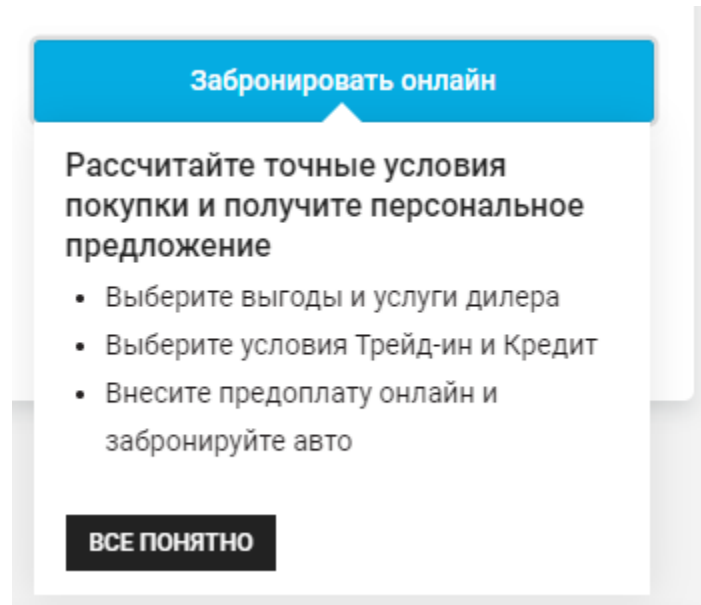
                cloneLink.innerText = text; // Устанавливаем текст кнопки на
                полученный из Smart Contract

                original.before(clone); // Вставляем новую кнопку до старой
                original.remove(); // Удаляем оригинальную кнопку (Этого можно не
                делать)
            }
        }
    });
}

```

```
        return [ cloneLink ]; // Возвращаем элемент кнопки, что бы при
нажатии открылся Smart Contract
    }
}
})
}
document.head.append(s);
```

И на детальной странице всё успешно работает.



The image shows a modal window with a blue header containing the text "Забронировать онлайн". Below the header, the main text reads "Рассчитайте точные условия покупки и получите персональное предложение". Underneath this, there is a bulleted list of three items: "Выберите выгоды и услуги дилера", "Выберите условия Трейд-ин и Кредит", and "Внесите предоплату онлайн и забронируйте авто". At the bottom left of the modal, there is a black button with the white text "ВСЕ ПОНЯТНО".

### УСЛОВИЯ СДЕЛКИ

Стоимость автомобиля

Цена автомобиля  
**1 123 900 ₹**

#### 1 Трейд-ин

Трейд-ин с преимуществом **-500 ₹**  
– Оцените свой автомобиль онлайн

Без Трейд-ин

#### 2 Финансовые услуги

Покупка в кредит **от 26 848 ₹/мес.**  
– Выберите кредитную программу

Покупка за наличные

#### 3 Дополнительные предложения

Футболка с надписью BMW **бесплатно**  
Размеры XL и XXL

Цена авто:	<b>1 123 900 ₹</b>
<b>ИТОГО</b> к оплате:	<b>1 123 900 ₹</b>

SMART CONTRACT | Сделано в Kodix Automotive

Дальнейшая настройка необходима для того, чтобы Smart Contract был более полным и сохранял в себя больше информации, по которой он сможет произвести более точные расчёты.

## Получаем фотографию автомобиля

Вспользуемся известным нам методом `document.querySelector` и с его помощью получим первую фотографию на автомобиль

Находим необходимые элементы на странице

```
▼<div class="swiper-slide swiper-slide-active" style="width: 730px;">
  ▼<div class="car-detail__swiper-top-img">
    ▼<a class="swiper-lazy swiper-lazy-loaded" title="Volkswagen Polo 1.6 MPI M
      T (90 л.с.) Respect + Зимний + Комфорт Белый" href="/upload/iblock/9e9/9e93
      8e0cdb5a802ee11c563dc511096b.jpg" data-fancybox="gallery-car-detail" data-
      options="{\"thumb\" : \"/upload/resize_cache/iblock/2e9/110_250_0/2e93e308ae14
      eda911edf2d647ff50c5.jpg}\"" data-mb-id="4767142" data-mb-pattern="images_ex
      terior" data-mb-rule="4" data-mb-file-use="LIVE">flex
       == $0
    ><div class="car-detail__fancy-ico">...</div>
  </a>
</div>
</div>
```

И формируем запрос для их получения `document.querySelector('.car-detail__swiper-top-img a.swiper-lazy')`, получаем из него значение ссылки

```
> document.querySelector('.car-detail__swiper-top-img a.swiper-lazy')
< a class="swiper-lazy swiper-lazy-loaded" title="Volkswagen Polo 1.6 MPI MT (90 л.с.) Respect + Зимний + Комфорт Белый" href="/upload/iblock/9e9/9e93
8e0cdb5a802ee11c563dc511096b.jpg" data-fancybox="gallery-car-detail" data-options="{\"thumb\" : \"/upload/resize_cache/iblock/2e9/110_250_0/2e93e308ae14
eda911edf2d647ff50c5.jpg}\"" data-mb-id="4767142" data-mb-pattern="images_exterior" data-mb-rule="4" data-mb-file-use="LIVE">...</a> (flex)
```

`document.querySelector('.car-detail__swiper-top-img a.swiper-lazy').href`

```
> document.querySelector('.car-detail__swiper-top-img a.swiper-lazy').href
< "https://avilon.ru/upload/iblock/9e9/9e938e0...jpg"
>
```

И вставляем в инициализацию виджета

```
InitSmartContract({
  detail: {
    picture: function() {
      return document.querySelector('.car-detail__swiper-top-img a.swiper-lazy').href;
    }
  }
})
```

Проверяем, что в наш контракт загрузилось изображение

УСЛОВИЯ СДЕЛКИ



Стоимость автомобиля

Цена автомобиля  
**1 123 900 ₺**