Документация по Site3D Configurator API

Оглавление

Передача начальных параметров виджета 2
Загрузка виджетов 3
Доступ к виджету 4
Получение выбор пользователя 5
Получение доступа к нужному элементу конфигуратора и его настройкам 7
Изменение цвета 7
Загрузка собственной текстуры из внешнего источника 7
Создание скриншота 7
Запуск примерки в дополненной реальности (AR) 8
Не нашли ответ на свой вопрос? 8

Передача начальных параметров виджета

Через атрибут data-options контейнера виджета можно передавать начальные параметры:

- caption – Заголовок виджета
- theme – Тема оформления виджета. Возможные строковые значения: default, lite
- fontFamily – Шрифт виджета. Возможные строковые значения: opensans, oswald, Helvetica
- borderRadius – Радиус скругления углов виджета в виде числа в пикселях. По умолчанию: 10
- backgroundColor – Цвет фона сцены в виде строки в HEX-формате. По умолчанию: #ffffff00
- preloadBackgroundColor – Цвет фона перед загрузкой виджета в виде строки в HEX-формате. По умолчанию: #ffffff00
- preloadBackgroundImage – Ссылка на фоновое изображение перед загрузкой виджета
- preloadPoster – Ссылка на изображение над кнопкой загрузки виджета
- autoCenter – Если истина, то модель будет размещена в центре сцены
- boundCenter – Если истина, то у модели будут исправлены центры её частей
- posCorrection – Коррекция позиции модели в % от своего размера. По умолчанию: {x:0, y: 0, z: 0}
- sizeCorrection – Коррекция размера модели в %. По умолчанию: 0
- rotateCorrection – Коррекция поворота модели в % от своего размера. По умолчанию: [0, 0, 0]
- startRotate – Настройки вращения модели при запуске виджета
    - isEnabled – Истина, если нужно включить однократное вращение модели при запуске виджета
- startHelp – Настройки показа анимационных иконок, поясняющих принципы управления сценой
    - isEnabled – Истина, если нужно включить показ анимационных иконок при запуске виджета
- autoLoad – Если истина, то виджет будет загружен автоматически
- navControl – Настройки управления навигацией
    - autoHide – Истина, если нужно скрывать навигацию при вращении модели
- navHelp – Настройки всплывающего окна помощи
    - text – Текст помощи
    - autoStart – Истина, если окно помощи нужно показать при загрузке виджета
- navStart – Настройки показа навигационных элементов при загрузке виджета
    - showButtons – Истина, если нужно показать основные кнопки навигации
    - showHotAr – Истина, если нужно показать отдельную кнопку вызова режима AR
    - showPanels – Истина, если нужно показать дополнительные панели с инструментами
- scaleControl – Настройки управления масштабированием (расстоянием камеры до модели)
    - isEnabled – Истина, если нужно включить изменение масштаба
    - minFromModel – Минимальное расстояние до модели в процентах от своего размера в виде строки. По умолчанию 0%
    - maxFromModel – Максимальное расстояние до модели в процентах от своего размера в виде строки. По умолчанию 300%
- rotateControl – Настройки управления вращением камеры
    - isEnabled – Истина, если нужно включить вращение камеры
    - min – минимальный угол вращения перпендикулярно вертикали. По умолчанию: 0
    - max – максимальный угол вращения перпендикулярно вертикали. По умолчанию: 90
    - minY – минимальный угол вращения вокруг вертикали. По умолчанию: -180
    - maxY – максимальный угол вращения вокруг вертикали. По умолчанию: 180
- autoRotate – Настройки автоматического вращения модели
    - autoStart – Истина, если нужно включить вращение модели при загрузке виджета
    - direction – Направление вращения в виде строки. Возможные значения: clockwise (по часовой), counterclockwise (против часовой)
    - speed – Скорость вращения в виде целого числа. По умолчанию: 1
- sizeControl – Настройки управления показом размеров модели
    - autoStart – Истина, если нужно включить показ размеров модели при загрузке виджета

        - unit – Единицы измерения в виде строки. Возможные значения: mm (мм), cm (см), m (м)
        - recalc – Истина, если нужно вывести реальные размеры модели в соответствии с выбранной единицей измерения
    - camera – Настройки камеры
        - posFromModel – Начальное положение камеры в виде массива из трех значений: расстояние до модели в процентах от её размера (возможно отрицательное значение для просмотра модели изнутри) и 2 угла вращения вокруг и перпендикулярно вертикали. По умолчанию: ['150%', 30, 30]
    - floor – Настройки пола под модель
        - isEnabled – Истина, если нужно показать пол
        - view – Вид пола в виде строки. Возможные значения: rectangle (прямоугольник), circle (круг)
        - scale – Размер пола в зависимости от размера модели в виде целого числа. По умолчанию: 2
        - color – Цвет пола в виде строки в HEX-формате. По умолчанию: #ffffffff
    - ar – Настройки дополненной реальности
        - preloadCaption – Название кнопки вызова режима AR до загрузки виджета

Ниже представлен пример вставки контейнера виджета в React:

```
<div data-site3d="1,height=440px" data-options={JSON.stringify({autoLoad: false, ar: {preloadCaption: 'Примерить в комнате'}})}/>
```

Загрузка виджетов

Если по каким-то причинам виджеты не загружаются автоматически (например, скрипт загрузился раньше контейнеров), то можно вызвать процесс загрузки в другое время:

```
const configuratorStart = window['site3dLoader']?.configuratorStart;

if (configuratorStart !== undefined)
{
 configuratorStart();
}
```

Для обработки процесса загрузки всех виджетов на странице необходимо определить функцию site3dConfiguratorLoadStatus глобального объекта window, где info – объект с результатом текущего этапа загрузки виджетов (loaded – истина, если процесс загрузки завершен, message – описание этапа загрузки, percent – процент завершения текущего этапа (может отсутствовать)).

Ниже представлен пример загрузчика в React:

```
useEffect(() =>
{
 window.site3dConfiguratorLoadStatus = info =>
 {
  if (info.loaded)
  {
   dispatch(loader(false)); // Скрыть загрузчик
   return;
  }

  dispatch(loader(true, info.message, info.percent)); // Показать загрузчик
```

```
  };
}, []);
```

Для ручной загрузки достаточно вызвать метод load объекта 3D-виджета:

```
await widget.load();
```

Доступ к виджету

Для начала работы с API Site3D Configurator необходимо получить доступ к объекту 3D-виджета (Site3dWidget) нужного конфигуратора. Для этого удобнее всего воспользоваться событием site3dWidgetLoad:

```
window.addEventListener('site3dWidgetLoad', event => {
  const configuratorId = event.configuratorId;
  const widget = event.widget;

  widget.event('widgetLoadCompleted', () => {
    // Для виджета без автозагрузки
  });
});
```

В данном событии мы получаем идентификатору проекта в нашем сервисе и доступ к виджету. Если наш виджет загружается не автоматически, а по клику на кнопку, то мы можем дождаться события widgetLoadCompleted, чтобы узнать, когда виджет будет загружен окончательно.

Также доступ к виджету можно получить и другими способами:

- Через свойство объекта window[`site3dWidget_${id}`], где id – идентификатор виджета (его можно увидеть на последнем шаге создания проекта)
- Через массив всех виджетов, доступный в свойстве объекта window['site3dWidgets']

Для проверки, загружен ли виджет полностью в настоящий момент, служит свойство isLoad.

Далее приведен пример реализации универсального метода получения доступа к виджету, где мы можем передать в виде объекта или идентификатор проекта (configuratorId) в виде строки или порядковый индекс виджета (index) в виде целого числа, начиная с нуля. Также можно передать параметр ожидания загрузки модели виджета (waitLoad) в виде истины или лжи (по умолчанию false):

```
async function getSite3dWidget(options = undefined)
{
  const configuratorId = options?.configuratorId ?? '';
  const index = options?.index ? parseInt(options.index) : 0;
  const waitLoad = options?.waitLoad === true;

  const widgets = window['site3dWidgets'];

  let widget = null;
```

```
  if (widgets)
  {
    if (configuratorId !== '')
    {
      widget = widgets.find(item => item.options.configuratorService?.id === configuratorId);
    }
    else
    {
      widget = widgets[index];
    }
  }

  if (widget)
  {
    return widget;
  }

  return new Promise(resolve => {
    window.addEventListener('site3dWidgetLoad', event => {
      const indexCur = event.index;
      const configuratorIdCur = event.configuratorId;
      const widgetCur = event.widget;

      if ((configuratorId !== '' && configuratorId === configuratorIdCur) || index === indexCur)
      {
        if (!waitLoad)
        {
          resolve(widgetCur);
          return;
        }

        widgetCur.event('widgetLoadCompleted', () => {
          resolve(widgetCur);
        });
      }
    });
  });
}
```

Для дальнейшего изучения возможностей класса Site3dWidget можно обратиться к документации библиотеки Site3D по адресу https://doc.site3d.site.

Получение выбор пользователя

Для того, чтобы определить, какие настройки применил пользователь достаточно обратиться к методу getConfiguratorInfo объекта 3D-виджета. Он возвращает объект с перечнем элементов конфигуратора и информацией об их настройках.

Ниже приведен пример вызова и возвращаемого данным методом объекта:

```
const widget = window['site3dWidgets'][0];
console.log(widget.getConfiguratorInfo());
```

```
{
  "items": [
    {
      "caption": "Загородный дом",
      "settings": [
        {
          "caption": "Цвет декоративных элементов",
          "data": {
            "value": "#e7e7e7",
            "price": 600
          }
        },
        {
          "caption": "Цвет черепицы",
          "data": {
            "value": "RAL 9010",
            "price": 900
          }
        },
        {
          "caption": "Вид черепицы",
          "data": {
            "caption": "Композитная",
            "color": "RAL 9010",
            "price": 3000
          }
        },
        {
          "caption": "Вид балкона",
          "data": {
            "caption": "Стандарт",
            "price": 5000
          }
        }
        {
          "caption": "Показать крышу",
          "data": {
            "value": "Да",
            "price": 0
          }
        },
      ]
    }
  ]
}
```

Получение доступа к нужному элементу конфигуратора и его настройкам

Ниже приведен пример получения доступа к элементу конфигуратора с именем **main** (такое имя у элемента, которому соответствует модель, загружаемая на первом шаге создания или редактирования проекта), а далее, к его настройкам в виде объекта Map:

```
const item = widget.configurator.items.get('main');
const settingsItems = item.settings.items;
```

Изменение цвета

Для редактирования значения настройки цвета, можно вызвать у неё метод setValue, передав ему цвет в формате HEX.

Ниже представлен пример изменения цвета настройки, привязанной к части модели с именем **core**:

```
const colorSetting = settingsItems.get('color_core');
colorSetting.setValue('#ff0000');
```

Если же нам нужно сменить цвет у настройки материала, то можно вызвать у неё метод setColor, например:

```
const materialSetting = settingsItems.get('material_image');
materialSetting.setColor('#ff0000');
```

Загрузка собственной текстуры из внешнего источника

Чтобы загрузить свою текстуру для настройки материала, необходимо вызвать у неё метод setTextureFromBlob, передав ему изображение в виде Blob-объекта.

Ниже продемонстрирован пример загрузки текстуры **blob** для настройки материала, которая привязана к части модели с именем **image**:

```
const materialSetting = settingsItems.get('material_image');
materialSetting.setTextureFromBlob(blob);
```

Создание скриншота

Для того, чтобы получить файл изображения сцены в формате png или jpeg, можно вызвать метод exportImage объекта 3D-виджета, передав ему нужные параметры в виде объекта (смотрите в примере ниже). Данный метод возвращает **Blob-объект** скриншота или **null** в случае невозможности создания изображения. Параметр exceptions – массив имен частей модели для исключения из скриншота.

```
await widget.exportImage({
  fileName: 'image',
  format: 'png',
  width: 1920,
  height: 1080,
  isWatermark: false,
  isDownload: true,
  exceptions: []
});
```

Запуск примерки в дополненной реальности (AR)

Достаточно вызвать метод arOn объекта 3D-виджета. Метод возвращает объект с информацией о результате вызова. Если метод вызывается на десктопе, то он вернет свойство **qr** – QR-код. Это свойство можно использовать для отображения QR-кода в нужном месте сайта.

```
const result = await widget.arOn();
```

Не нашли ответ на свой вопрос?

Обратитесь, пожалуйста, в нашу поддержку, поможем с интеграцией сервиса с вашим сайтом.