

Как изменить цвет SVG-изображения с помощью CSS (замена SVG-изображения jQuery)?

Спрошено 11 лет, 4 месяца назад Изменено 4 месяца назад Просмотрено 770 тысяч раз



На этот вопрос есть ответы на [Stack Overflow на русском](#)

469



Это самостоятельный вопрос-ответ к удобному фрагменту кода, который я придумал.

В настоящее время не существует простого способа встроить SVG-изображение и затем получить доступ к SVG-элементам через CSS. Существуют различные методы использования JS SVG-фреймворков, но они чрезмерно сложны, если все, что вы делаете, это создаете простую иконку с состоянием ролловера.

Итак, вот что я придумал, и я думаю, что это, безусловно, самый простой способ использовать SVG-файлы на веб-сайте. Концепция заимствована из ранних методов замены текста на изображение, но, насколько мне известно, никогда не делалась для SVGs.

Вот в чем вопрос:

Как мне встроить SVG-файл и изменить его цвет в CSS без использования JS-SVG фреймворка?

[jquery](#) [css](#) [svg](#)

Поделиться Подписаться

отредактировано 14 сентября 2015 г. в 1:26

спрошено 16 августа 2012 г. в 0:05



Pang

9692

146

82

122



Дрю Бейкер

14.2k

15

59

97

- К сожалению, тег `img` не работает с `svg`-файлами в IE, поэтому имейте это в виду. IE распознает теги для встраивания. В любом случае, отличная работа! —пользователь1973469 26 апреля 2013 в 17:50
- Для `svg` следует использовать `css`-свойство `"fill"`. Для изображений уместно использовать `"filter"`. "Фильтр" на самом деле работает для обоих, но нет необходимости выполнять всю эту работу для векторной графики. —Сам Бенчериф 15 сентября 2016 в 3:58

19 Ответов

Отсортировано по: Highest score (default)



Во-первых, используйте тег `IMG` в вашем HTML для встраивания SVG-графики. Для создания графики я использовал Adobe Illustrator.

560



```

```



Это похоже на то, как вы вставляете обычное изображение. Обратите внимание, что вам нужно установить для IMG класс `svg`. Класс `'social-link'` приведен только для примера. Идентификатор не требуется, но полезен.

Затем используйте этот код jQuery (в отдельном файле или встроенный в HEAD).

```
/**
 * Replace all SVG images with inline SVG
 */
jQuery('img.svg').each(function(){
    var $img = jQuery(this);
    var imgID = $img.attr('id');
    var imgClass = $img.attr('class');
    var imgURL = $img.attr('src');

    jQuery.get(imgURL, function(data) {
        // Get the SVG tag, ignore the rest
        var $svg = jQuery(data).find('svg');

        // Add replaced image's ID to the new SVG
        if(typeof imgID !== 'undefined') {
            $svg = $svg.attr('id', imgID);
        }
        // Add replaced image's classes to the new SVG
        if(typeof imgClass !== 'undefined') {
            $svg = $svg.attr('class', imgClass+' replaced-svg');
        }

        // Remove any invalid XML tags as per http://validator.w3.org
        $svg = $svg.removeAttr('xmlns:a');

        // Replace image with new SVG
        $img.replaceWith($svg);

    }, 'xml');
});
```

Приведенный выше код выполняет поиск всех изображений IMG с классом `'svg'` и заменяет его встроенным SVG из связанного файла. Огромное преимущество заключается в том, что это позволяет вам использовать CSS для изменения цвета SVG прямо сейчас, вот так:

```
svg:hover path {
    fill: red;
}
```

Код jQuery, который я написал, также переносит идентификатор исходного изображения и классы. Так что этот CSS тоже работает:

```
#facebook-logo:hover path {
    fill: red;
}
```

Или:

```
.social-link:hover path {
    fill: red;
}
```

Вы можете увидеть пример его работы здесь: <http://labs.funkhausdesign.com/examples/img-svg/img-to-svg.html>

У нас есть более сложная версия, которая включает кэширование здесь:

<https://github.com/funkhaus/style-guide/blob/master/template/js/site.js#L32-L90>

Поделиться Подписаться

отредактировано 18 сентября 2019 г. в 7:16

ответил 16 августа 2012 г. в 0:05



Jonas Sourlier

14k 17 80 149



Дрю Бейкер

14.2k 15 59 97

Это потрясающе! Я искал способ сохранить разметку чистой, но при этом разрешить доступ к "внутренностям" svg для использования в CSS. Это должно сработать, но я получаю сообщение об ошибке с JS: "XMLHttpRequest не может загрузить файл: ///H:/svg /test/test.svg. Исходное значение null не разрешено Access-Control-Allow-Origin." Есть идеи? —[cmeagown](#) 13 февраля 2013 в 19:13

Я предполагаю, потому что вы запускаете его локально и получаете междоменную проблему. Разместите его на сервере, и он должен работать. — [Дрю Бейкер](#) 14 февраля 2013 в 20:50

Если вы поместите это в обработчик onDocumentReady, вы можете получить эффект FOUC (flash of unstyled content), при котором он отображает исходные теги перед заменой их тегами <svg>. Если бы был какой-то способ запускать это непосредственно перед отображением каждого тега , это было бы идеально. Я полагаю, вы также могли бы использовать CSS для скрытия элементов 'img.svg', чтобы они отображались только после замены. —[Micah](#) 26 мая 2013 в 21:27 ✎

7 Иногда это может не работать в Safari (например), чтобы убедиться, что возвращаемые данные читаемы, замените }); из \$.get на }, 'xml'); —[Джоан](#) 24 июля 2013 в 10:28

31 Вероятно, вы могли бы даже заменить селектор на img[src\$=".svg"] и устранить необходимость в svg классе. —[Кейси Чу](#) 4 августа 2013 в 23:40

Стиль

59

```
svg path {  
  fill: #000;  
}
```

Скрипт

```
$(document).ready(function() {  
  $('img[src$=".svg"]').each(function() {  
    var $img = jQuery(this);  
    var imgURL = $img.attr('src');  
    var attributes = $img.prop("attributes");  
  
    $.get(imgURL, function(data) {  
      // Get the SVG tag, ignore the rest  
      var $svg = jQuery(data).find('svg');  
  
      // Remove any invalid XML tags  
      $svg = $svg.removeAttr('xmlns:a');  
  
      // Loop through IMG attributes and apply on SVG  
      $.each(attributes, function() {  
        $svg.attr(this.name, this.value);  
      });  
  
      // Replace IMG with SVG  
      $img.replaceWith($svg);  
    }, 'xml');  
  });  
});
```

```
});  
});
```

Поделиться Подписаться

ответил 1 февраля 2016 г. в 9:21



Henrik Albrechtsson

1,731 2 17 17

- 2 Если у вас нет значения ширины, оно просто создает значение с неправильным номером. `width="170.667"` в моем случае —[stallingOne](#) 29 мая 2017 в 10:10
- 3 Это не идеально, так как при этом теряются предыдущие размеры `img`. —[RichieHH](#) 17 июня 2017 в 11:35
- 1 Здравствуйте, предположим, у меня разные `svg`-файлы с разным цветом для каждого. Используя этот метод, все мои `svg`-цвета становятся такими же, как у первого `svg`-файла, который за циклируется. Есть идеи, как я могу обойти это, чтобы каждый цвет оставался таким же, как раньше? —[tnkh](#) 24 ноября 2017 в 10:16
- 2 обратите внимание, что если ваш `svg` также сделан из `path` фигур (например, `rect`), вам также нужно будет обрабатывать их в `css` —[Mugen](#) 26 июня 2019 в 9:44



43



Теперь вы можете использовать [свойство filter CSS](#) в [большинстве современных браузеров](#) (включая Edge, но не IE11). Это работает как с SVG-изображениями, так и с другими элементами. Вы можете использовать `hue-rotate` или `invert` для изменения цветов, хотя они не позволяют изменять разные цвета независимо. Я использую следующий класс CSS для отображения "отключенной" версии значка (где оригиналом является SVG-изображение насыщенного цвета):

```
.disabled {  
  opacity: 0.4;  
  filter: grayscale(100%);  
  -webkit-filter: grayscale(100%);  
}
```

Это делает его светло-серым в большинстве браузеров. В IE (и, вероятно, Opera Mini, которую я не тестировал) он заметно выцветает из-за свойства `opacity`, которое по-прежнему выглядит довольно хорошо, хотя и не серое.

Вот пример с четырьмя различными классами CSS для значка колокольчика [Twemoji](#): original (желтый), вышеупомянутый "отключенный" класс, `hue-rotate` (зеленый) и `invert` (синий).

[Показать фрагмент кода](#)

Поделиться Подписаться

отредактировано 2 декабря 2018 г. в 10:47

ответил 28 июня 2016 г. в 19:10



Брайан Бернс

21k 9 85 78



aldel

6621 1 27 32

- 1 Только что заметил, что инвертировать - хорошее решение, если вы не хотите создавать шрифты для значков. Я использовал этот код jQuery, чтобы изменить значок в заголовке моего веб-сайта в соответствии со свойством `css color` (обратите внимание, что я использую белые значки `png`): `if ($('#.w3-top img').css("color") == "rgb(0, 0, 0)") { $('#.w3-top img').css("filter", "invert(100%)"); $('#.w3-top img').css("-webkit-filter", "invert(100%)"); }` —[пользователь5147563](#) 15 апреля 2017 в 11:23
- 1 Отличный подход. Отредактировал мой `SVG+xml`, чтобы добавить целевой цвет значка, затем использовал класс `.icon-disabled`, чтобы сделать его серым. —[Сушигуй](#) 16 июня 2018 в 0:20

26

В качестве альтернативы вы могли бы использовать CSS `mask`, учитывая, что [поддержка браузера](#) не очень хороша, но вы могли бы использовать запасной вариант

```
.frame {  
  background: blue;  
  -webkit-mask: url(image.svg) center / contain no-repeat;  
}
```

Поделиться Подписаться

отредактировано 18 ноября 2014 в 19:21

ответил 14 августа 2014 г. в 10:22



paulalexandru

9,286 7 66 95



seanjacob

2,248 1 22 26

16 [MDN указывает, что](#) это `-webkit-mask` не должно использоваться ни на одном производственном веб-сайте. –vaindil 7 марта 2016 в 22:20

3 не окрашивает svg –вишал 15 марта 2017 в 13:03

1 Возможно, уместно сказать, что сейчас, четыре года спустя, это решение работает во всех основных браузерах. Только что протестировано здесь, и оно на 100% в порядке. –Дэниел Лемес 12 декабря 2018 в 18:56

24

Если вы можете включить файлы (включить PHP или включить через выбранную вами CMS) на своей странице, вы можете добавить SVG-код и включить его на свою страницу. Это работает так же, как вставка SVG-источника на страницу, но делает разметку страницы более чистой.

Преимущество заключается в том, что вы можете настраивать таргетинг на части вашего SVG-файла с помощью CSS для наведения курсора - javascript не требуется.

<http://codepen.io/chriscoyier/pen/evcBu>

Вам просто нужно использовать правило CSS, подобное этому:

```
#pathidorclass:hover { fill: #303 !important; }
```

Обратите внимание, что бит `!important` необходим для переопределения цвета заливки.

Поделиться Подписаться

ответил 1 октября 2013 г. в 16:01



требор1979

1,123 9 6

4 Для тех, кто использует AngularJS: `<div ng-include="'svg.svg'"></div>` –Микель 24 февраля 2016 в 16:22

1 Не очень элегантное решение для хранения svg-данных в базе данных. Не так уж и плохо, но выкачивание данных xml / html / svg DOM из API или CMS вместо использования шаблонов или других ресурсов просто кажется неправильным. –ChristoKiwi 17 августа 2017 г. в 5:00

1 Кроме того, если ваш SVG-файл имеет прозрачные области, они не будут считаться зависшими, и у вас может возникнуть "резкое наведение курсора". Чтобы это исправить, просто добавьте элемент-оболочку (`<a>`, если это удобно), а затем добавьте это в правило CSS. `#pathidorclass:hover`, `.wrapperclass:hover`



24



TL / DR: ПЕРЕЙДИТЕ сюда-> <https://codepen.io/sosuke/pen/Pjoqqp>

Объяснение:

Я предполагаю, что у вас есть html что-то вроде этого:

```

```


Обязательно используйте фильтр, т.е.. ваш svg, скорее всего, черный или белый. Вы можете применить фильтр, чтобы придать ему любой желаемый цвет, например, у меня есть черный svg, который я хочу мятно-зеленый. Сначала я инвертирую его, чтобы он был белым (что технически соответствует всем цветам RGB в полном объеме), затем играю с насыщенностью оттенка и т.д. Чтобы все получилось правильно:

```
filter: invert(86%) sepia(21%) saturate(761%) hue-rotate(92deg) brightness(99%)
contrast(107%);
```

Еще лучше то, что вы могли бы просто использовать инструмент для преобразования нужного вам шестнадцатеричного значения в фильтр для вас: <https://codepen.io/sosuke/pen/Pjoqqp>

Поделиться Подписаться

ответил 19 февраля 2020 г. в 19:27

 Ав AhrenFullStop
польз: 462 4 11



19



@Drew Baker предложил отличное решение для решения проблемы. Код работает правильно. Однако те, кто использует AngularJS, могут обнаружить большую зависимость от jQuery. Следовательно, я подумал, что было бы неплохо вставить для пользователей AngularJS код, следующий за решением @Drew Baker.

AngularJS - способ того же кода

1. *Html*: используйте приведенный ниже тег в вашем HTML-файле:

```
<svg-image src="/icons/my.svg" class="any-class-you-wish"></svg-image>
```

2. *Директива*: это будет директива, которая вам понадобится для распознавания тега:

```
'use strict';
angular.module('myApp')
  .directive('svgImage', ['$http', function($http) {
    return {
      restrict: 'E',
      link: function(scope, element) {
        var imgURL = element.attr('src');
        // if you want to use ng-include, then
        // instead of the above line write the bellow:
        // var imgURL = element.attr('ng-include');
```

```

    var request = $http.get(
      imgURL,
      {'Content-Type': 'application/xml'}
    );

    scope.manipulateImgNode = function(data, elem){
      var $svg = angular.element(data)[4];
      var imgClass = elem.attr('class');
      if(typeof(imgClass) !== 'undefined') {
        var classes = imgClass.split(' ');
        for(var i = 0; i < classes.length; ++i){
          $svg.classList.add(classes[i]);
        }
      }
      $svg.removeAttribute('xmlns:a');
      return $svg;
    };

    request.success(function(data){
      element.replaceWith(scope.manipulateImgNode(data, element));
    });
  }
};
}]);

```

3. CSS:

```

.any-class-you-wish{
  border: 1px solid red;
  height: 300px;
  width: 120px
}

```

4. Модульный тест с karma-jasmine:

```

'use strict';

describe('Directive: svgImage', function() {

  var $rootScope, $compile, element, scope, $httpBackend, apiUrl, data;

  beforeEach(function() {
    module('myApp');

    inject(function($injector) {
      $rootScope = $injector.get('$rootScope');
      $compile = $injector.get('$compile');
      $httpBackend = $injector.get('$httpBackend');
      apiUrl = $injector.get('apiUrl');
    });

    scope = $rootScope.$new();
    element = angular.element('<svg-image src="/icons/icon-man.svg" class="svg"></svg-image>');
    element = $compile(element)(scope);

    spyOn(scope, 'manipulateImgNode').andCallThrough();
    $httpBackend.whenGET(apiUrl + 'me').respond(200, {});

    data = '<?xml version="1.0" encoding="utf-8"?>' +
      '<!-- Generator: Adobe Illustrator 17.0.0, SVG Export Plug-In . SVG Version: 6.00 Build 0) -->' +
      '<!DOCTYPE svg PUBLIC "-//W3C//DTD SVG 1.1//EN" ' +
      '"http://www.w3.org/Graphics/SVG/1.1/DTD/svg11.dtd">' +
      '<!-- Obj -->' +
      '<!-- Obj -->' +
      '<svg version="1.1" id="Capa_1" xmlns="http://www.w3.org/2000/svg"

```

```

xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px" ' +
'width="64px" height="64px" viewBox="0 0 64 64" enable-background="new 0 0 64 64"
xml:space="preserve">' +
'<g>' +
'<path fill="#F4A902" d=""/>' +
'<path fill="#F4A902" d=""/>' +
'</g>' +
'</svg>';
$httpBackend.expectGET('/icons/icon-man.svg').respond(200, data);
});

afterEach(function() {
$httpBackend.verifyNoOutstandingExpectation();
$httpBackend.verifyNoOutstandingRequest();
});

it('should call manipulateImgNode at least once', function () {
$httpBackend.flush();
expect(scope.manipulateImgNode.callCount).toBe(1);
});

it('should return correct result', function () {
$httpBackend.flush();
var result = scope.manipulateImgNode(data, element);
expect(result).toBeDefined();
});

it('should define classes', function () {
$httpBackend.flush();
var result = scope.manipulateImgNode(data, element);
var classList = ["svg"];
expect(result.classList[0]).toBe(classList[0]);
});
});

```

Поделиться Подписаться

отредактировано 23 февраля
2015 г. в 8:18

ответил 27 января 2015 г. в 9:34



Макс

2,435 6 36 54

- 2 ваше решение не работает, может быть `<div ng-include="/icons/my.svg" class="any-class-you-wish"></div>` —Гийом Винсент 22 февраля 2015 в 11:18 ✎
- 1 @guillaumevincent если вы хотите использовать это с `ng-include`, то просто измените эту строку `var imgURL = element.attr('src');` на `var imgURL = element.attr('ng-include');` —Макс 23 февраля 2015 в 8:15
- 1 Это очень удобное решение, но будьте осторожны при его чрезмерном использовании, поскольку это может довольно сильно ударить по производительности - например, набор из 5 значков общего доступа, повторяющихся в списке статей или что-то в этом роде. —skxc 11 марта 2015 в 15:24
- 2 Проблема с вашим кодом в IE. Вы можете использовать просто `if (typeof(imgClass) !== 'undefined') { $svg.setAttribute("class", imgClass); }` вместо `split` и `for loop`. —Роберт Бокори 18 июня 2015 в 21:02
- 3 Потрясающая работа! Но для определенного изображения вам нужно взять первый элемент `svg` (`angular.element(data)[0];`) и заставить его работать с использованием IE `if ($svg.getAttribute('class')) { $svg.setAttribute('class', $svg.getAttribute('class') + ' ' + imgClass); } else { $svg.setAttribute('class', imgClass); }`. Также вы можете захотеть добавить `cache: true` в опции `$http.get`, иначе ваша страница может стать очень медленной. —leo 7 марта 2016 в 13:32



<path style="fill:#010002;" d="M394.854,205.444c9.218-15.461,19.102-30.181,14.258-49.527

...
C412.843,226.163,402.511,211.451,394.854,205.444z"/>

Это может сэкономить массу уродливого скрипта. Извините, если это нестандартно, но иногда простые решения могут быть упущены из виду.

... даже замена нескольких svg-изображений может быть меньше по размеру, чем некоторые фрагменты кода для этого вопроса.

Поделиться Подписаться

отредактировано 31 января 2018
г. в 21:03

ответил 3 января 2018 г. в 22:23



DShultz

4,431 3 32 49

Я написал директиву для решения этой проблемы с помощью AngularJS. Она доступна [здесь - ngReusableSvg](#).

8

Он заменяет SVG-элемент после того, как он был отрисован, и помещает его внутри div элемента, делая его CSS легко изменяемым. Это помогает использовать один и тот же SVG-файл в разных местах с использованием разных размеров / цветов.

Использование простое:

```
<object oa-reusable-svg
  data="my_icon.svg"
  type="image/svg+xml"
  class="svg-class"
  height="30" // given to prevent UI glitches at switch time
  width="30">
</object>
```

После этого вы можете легко получить:

```
.svg-class svg {
  fill: red; // whichever color you want
}
```

Поделиться Подписаться

ответил 16 сентября 2015 г. в 20:21



Омри Аарон

17k 5 40 58

- Привет, спасибо за предоставление этого решения. Я пробовал, и это дает результат: <div ng-click="EventHandler()" ng-class="classEventHandler()" style="height:30px; width:30px;float:left;" class="svg-class" id="my-svg" height="30" width="30">[[объект SVGSVGElement]]</div> В затем html просто помещает [[объект SVGSVGElement]]. Вы знаете, в чем проблема? Другой вопрос, это сильно влияет на производительность или я могу использовать это для многих svg-файлов на странице? И, наконец, это все еще работает в angular 1.3 (the bower). —берслинг 5 июня 2016 в 11:18 ✎
- Какую версию angular вы используете? Не сталкивались с вашей проблемой.. Может быть, это что-то с SVG? С точки зрения производительности переключатель относительно тяжелый, я сам использовал его примерно на 10, и все было в порядке.. Я думаю, это зависит от количества / размера, поэтому попробуйте и



6



Вот версия для `knockout.js`, основанная на принятом ответе:

Важно: на самом деле для замены тоже требуется jQuery, но я подумал, что некоторым это может быть полезно.

```
ko.bindingHandlers.svgConvert =
{
  'init': function ()
  {
    return { 'controlsDescendantBindings': true };
  },

  'update': function (element, valueAccessor, allBindings, viewModel,
bindingContext)
  {
    var $img = $(element);
    var imgID = $img.attr('id');
    var imgClass = $img.attr('class');
    var imgURL = $img.attr('src');

    $.get(imgURL, function (data)
    {
      // Get the SVG tag, ignore the rest
      var $svg = $(data).find('svg');

      // Add replaced image's ID to the new SVG
      if (typeof imgID !== 'undefined')
      {
        $svg = $svg.attr('id', imgID);
      }
      // Add replaced image's classes to the new SVG
      if (typeof imgClass !== 'undefined')
      {
        $svg = $svg.attr('class', imgClass + ' replaced-svg');
      }

      // Remove any invalid XML tags as per http://validator.w3.org
      $svg = $svg.removeAttr('xmlns:a');

      // Replace image with new SVG
      $img.replaceWith($svg);

    }, 'xml');
  }
};
```

Затем просто примените `data-bind="svgConvert: true"` к вашему тегу `img`.

Это решение полностью заменяет `img` тег SVG, и любые дополнительные привязки не будут соблюдаться.

Поделиться Подписаться

ответил 1 октября 2016 г. в 2:53



[Simon_Weaver](#)

142k 84 655 695

3 Это здорово! Если вы хотите перейти на следующий уровень, у нас есть обновленная версия, которая включает кэширование, поэтому один и тот же SVG-файл не запрашивается дважды.

- 1 Я немного беспокоился по этому поводу, но у меня не было времени разобраться в этом самому. Просто нужно было что-то быстрое –[Simon_Weaver](#) 1 октября 2016 в 23:28
- 2 @DrewBaker на самом деле меня больше беспокоило, что `ter img` запросит файл, а затем `get` запросит его снова. Я рассматривал возможность изменения `src` на `data-src` атрибут в `img` теге, но пришел к выводу, что современные браузеры, вероятно, достаточно умны, чтобы кэшировать файл в любом случае –[Simon_Weaver](#) 1 октября 2016 в 23:30



6



Существует библиотека с открытым исходным кодом под названием SVGInject, которая использует `onload` атрибут для запуска внедрения. Вы можете найти проект на GitHub по адресу <https://github.com/iconfu/svg-inject>

Вот минимальный пример использования SVGInject:

```
<html>
  <head>
    <script src="svg-inject.min.js"></script>
  </head>
  <body>
    
  </body>
</html>
```

После загрузки изображения `onload="SVGInject(this)` запустится внедрение, и `` элемент будет заменен содержимым SVG-файла, указанного в `src` атрибуте.

Это решает несколько проблем с внедрением SVG:

1. SVG-файлы могут быть скрыты до завершения внедрения. Это важно, если стиль уже применен во время загрузки, что в противном случае вызвало бы кратковременную "вспышку содержимого без стиля".
2. `` Элементы внедряются автоматически. Если вы добавляете SVG-файлы динамически, вам не нужно беспокоиться о повторном вызове функции внедрения.
3. К каждому идентификатору в SVG добавляется случайная строка, чтобы избежать многократного использования одного и того же идентификатора в документе, если SVG вводится более одного раза.

SVGInject - это простой Javascript и работает со всеми браузерами, поддерживающими SVG.

Отказ от ответственности: я являюсь соавтором SVGInject

Поделиться Подписаться

ответил 4 сентября 2018 г. в 16:04



Варуяма

3,333 1 32 44



4

Здесь нет фреймворкового кода, только чистый js :

```
document.querySelectorAll('img.svg').forEach(function(element) {
  var imgID = element.getAttribute('id')
  var imgClass = element.getAttribute('class')
```

```

var imgURL = element.getAttribute('src')

xhr = new XMLHttpRequest()
xhr.onreadystatechange = function() {
    if(xhr.readyState == 4 && xhr.status == 200) {
        var svg = xhr.responseXML.getElementsByTagName('svg')[0];

        if(imgID != null) {
            svg.setAttribute('id', imgID);
        }

        if(imgClass != null) {
            svg.setAttribute('class', imgClass + ' replaced-svg');
        }

        svg.removeAttribute('xmlns:a')

        if(!svg.hasAttribute('viewBox') && svg.hasAttribute('height') &&
svg.hasAttribute('width')) {
            svg.setAttribute('viewBox', '0 0 ' + svg.getAttribute('height')
+ ' ' + svg.getAttribute('width'))
        }
        element.parentElement.replaceChild(svg, element)
    }
}
xhr.open('GET', imgURL, true)
xhr.send(null)
})

```

Поделиться Подписаться

отредактировано 17 ноября 2017
г. в 10: 22

ответил 7 ноября 2017 г. в 20: 41



пользователь3144480
353 1 2 10

Если у нас больше таких svg-изображений, мы также можем воспользоваться помощью файлов шрифтов.

3

Сайты типа <https://glyphter.com/> могут предоставить нам файл шрифта из наших svg-файлов.

Например.

```

@font-face {
    font-family: 'iconFont';
    src: url('iconFont.eot');
}
#target{
    color: white;
    font-size:96px;
    font-family:iconFont;
}

```

Поделиться Подписаться

ответил 7 июля 2015 г. в 19:25



Абхишек Борар
413 3 7

7 Лично я ненавижу технику "изображения как шрифт". Это затрудняет добавление / редактирование изображений, добавляет много бессмысленной разметки. Шрифты должны быть шрифтами, изображения должны быть изображениями и т.д. – [Дрю Бейкер](#) 7 июля 2015 в 21:47

Согласен. Вам также необходимо запомнить / выполнить поиск изображений, назначенных символам. но для конкретного случая, когда изображения используются как значки / кнопки / маркеры, действуют скорее как

даже github больше не использует шрифт для значка github.com/blog/2112-delivering-octicons-with-svg
–gagarine 25 июня 2017 в 18:25



3



```
<?php
$content    = file_get_contents($pathToSVG);
?>
```



Затем я напечатал содержимое "как есть" внутри контейнера div

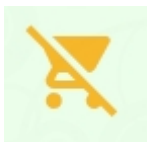
```
<div class="fill-class"><?php echo $content;?></div>
```

Чтобы окончательно установить правило для дочерних элементов SVG контейнера в CSS

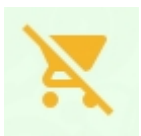
```
.fill-class > svg {
  fill: orange;
}
```

Я получил этот результат с помощью значка материала SVG:

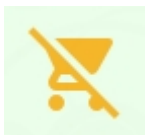
1. Mozilla Firefox 59.0.2 (64-разрядная версия) Linux



2. Google Chrome 66.0.3359.181 (Официальная сборка) (64 бита) Linux



3. Opera 53.0.2907.37 Linux



Поделиться Подписаться

ответил 6 июня 2018 г. в 20:17



Бенджамин

558 7 15



2

Для этого вы можете использовать data-image. используя data-image (data-URI), вы можете получить доступ к SVG как к inline.

Вот эффект опрокидывания с использованием чистого CSS и SVG.

Я знаю, что это **грязно**, но вы можете сделать это таким образом.

```
.action-btn {
  background-size: 20px 20px;
  background-position: center center;
  background-repeat: no-repeat;
  border-width: 1px;
  border-style: solid;
  border-radius: 30px;
  height: 40px;
  width: 60px;
  display: inline-block;
}

.delete {
  background-image: url("data:image/svg+xml;charset=UTF-8,%3csvg version='1.1'
id='Capa_1' fill='#FB404B' xmlns='http://www.w3.org/2000/svg'
xmlns:xlink='http://www.w3.org/1999/xlink' x='0px' y='0px' width='482.428px'
height='482.429px' viewBox='0 0 482.428 482.429' style='enable-background:new 0 0
482.428 482.429;' xml:space='preserve'%3e%3cg%3e%3cg%3e%3cpath d='M381.163,57.799h-
75.094C302.323,25.316,274.686,0,241.214,0c-33.471,0-61.104,25.315-64.85,57.799h-75.098
c-30.39,0-55.111,24.728-
55.111,55.117v2.828c0,23.223,14.46,43.1,34.83,51.199v260.369c0,30.39,24.724,55.117,55.112
h210.236c30.389,0,55.111-24.729,55.111-55.117V166.944c20.369-8.1,34.83-27.977,34.83-
51.199v-2.828 C436.274,82.527,411.551,57.799,381.163,57.799z
M241.214,26.139c19.037,0,34.927,13.645,38.443,31.66h-76.879
C206.293,39.783,222.184,26.139,241.214,26.139z M375.305,427.312c0,15.978-13,28.979-
28.973,28.979H136.096 c-15.973,0-28.973-13.002-28.973-28.979V170.861h268.182V427.312z
M410.135,115.744c0,15.978-13,28.979-28.973,28.979H101.266 c-15.973,0-28.973-13.001-
28.973-28.979v-2.828c0-15.978,13-28.979,28.973-
28.979h279.897c15.973,0,28.973,13.001,28.973,28.979 V115.744z'/%3e%3cpath
d='M171.144,422.863c7.218,0,13.069-5.853,13.069-13.068V262.641c0-7.216-5.852-13.07-
13.069-13.07 c-7.217,0-13.069,5.854-
13.069,13.07v147.154C158.074,417.012,163.926,422.863,171.144,422.863z'/%3e%3cpath
d='M241.214,422.863c7.218,0,13.07-5.853,13.07-13.068V262.641c0-7.216-5.854-13.07-13.07-
13.07 c-7.217,0-13.069,5.854-
13.069,13.07v147.154C228.145,417.012,233.996,422.863,241.214,422.863z'/%3e%3cpath
d='M311.284,422.863c7.217,0,13.068-5.853,13.068-13.068V262.641c0-7.216-5.852-13.07-
13.068-13.07 c-7.219,0-13.07,5.854-
13.07,13.07v147.154C298.213,417.012,304.067,422.863,311.284,422.863z'/%3e%3c/g%3e%3c
");
  border-color: #FB404B;
}

.delete:hover {
  background-image: url("data:image/svg+xml;charset=UTF-8,%3csvg version='1.1'
id='Capa_1' fill='#fff' xmlns='http://www.w3.org/2000/svg'
xmlns:xlink='http://www.w3.org/1999/xlink' x='0px' y='0px' width='482.428px'
height='482.429px' viewBox='0 0 482.428 482.429' style='enable-background:new 0 0
482.428 482.429;' xml:space='preserve'%3e%3cg%3e%3cg%3e%3cpath d='M381.163,57.799h-
75.094C302.323,25.316,274.686,0,241.214,0c-33.471,0-61.104,25.315-64.85,57.799h-75.098
c-30.39,0-55.111,24.728-
55.111,55.117v2.828c0,23.223,14.46,43.1,34.83,51.199v260.369c0,30.39,24.724,55.117,55.112
h210.236c30.389,0,55.111-24.729,55.111-55.117V166.944c20.369-8.1,34.83-27.977,34.83-
51.199v-2.828 C436.274,82.527,411.551,57.799,381.163,57.799z
M241.214,26.139c19.037,0,34.927,13.645,38.443,31.66h-76.879
C206.293,39.783,222.184,26.139,241.214,26.139z M375.305,427.312c0,15.978-13,28.979-
28.973,28.979H136.096 c-15.973,0-28.973-13.002-28.973-28.979V170.861h268.182V427.312z
M410.135,115.744c0,15.978-13,28.979-28.973,28.979H101.266 c-15.973,0-28.973-13.001-
28.973-28.979v-2.828c0-15.978,13-28.979,28.973-
28.979h279.897c15.973,0,28.973,13.001,28.973,28.979 V115.744z'/%3e%3cpath
d='M171.144,422.863c7.218,0,13.069-5.853,13.069-13.068V262.641c0-7.216-5.852-13.07-
13.069-13.07 c-7.217,0-13.069,5.854-
13.069,13.07v147.154C158.074,417.012,163.926,422.863,171.144,422.863z'/%3e%3cpath
d='M241.214,422.863c7.218,0,13.07-5.853,13.07-13.068V262.641c0-7.216-5.854-13.07-13.07-
13.07 c-7.217,0-13.069,5.854-
13.069,13.07v147.154C228.145,417.012,233.996,422.863,241.214,422.863z'/%3e%3cpath
d='M311.284,422.863c7.217,0,13.068-5.853,13.068-13.068V262.641c0-7.216-5.852-13.07-
13.068-13.07 c-7.219,0-13.07,5.854-
13.07,13.07v147.154C298.213,417.012,304.067,422.863,311.284,422.863z'/%3e%3c/g%3e%3c
");
}
```



```
// Get the svg file and replace the <svg> element.
$.ajax({
  url: src,
  cache: false
}).done(function (html) {
  let svg = $(html);
  svg.attr("width", width);
  svg.attr("height", height);
  svg.attr("class", cls);
  var newHtml = $('<a></a>').append(svg.clone()).html();
  ctx.replaceWith(newHtml);
});

return this;
};

}(jQuery));
```

В вашем html укажите svg-элемент следующим образом:

```
<svg src="images/someSvgFile.svg" height="45" width="45" class="mySVGClass"/>
```

И примените плагин:

```
$(".mySVGClass").svgLoader();
```

Поделиться Подписаться

отредактировано 8 июня 2017 г.
в 6:33

ответил 8 июня 2017 г. в 6:15



Johann

28.1k

39

172

287

Да, безусловно, есть более эффективные способы использования кода, который я дал. Вот как мы на самом деле используем его на производственных сайтах. Он кэширует SVG! github.com/funkhaus/style-guide/blob/master/template/js/... – Дрю Бейкер 12 июня 2017 в 21:38

для анимации событий наведения курсора мы можем оставить стили внутри svg-файла, например

```
<svg xmlns="http://www.w3.org/2000/svg">
<defs>
  <style>
    rect {
      fill:rgb(165,225,75);
      stroke:none;
      transition: 550ms ease-in-out;
      transform-origin:125px 125px;
    }
    rect:hover {
      fill:rgb(75,165,225);
      transform:rotate(360deg);
    }
  </style>
</defs>
<rect x='50' y='50' width='150' height='150' />
</svg>
```

[проверьте это на svgshare](#)

Поделиться Подписаться

ответил 20 июня 2018 г. в 12:50



```
.carousel-control-prev-icon {  
  background-image: url("data:image/svg+xml,%3csvg xmlns='http://www.w3.org/2000/svg'  
fill='rgb(3,122,247)' width='8' height='8' viewBox='0 0 8 8'%3e%3cpath d='M5.25 0l-4 4  
4 4 1.5-1.5L4.25 4l2.5-2.5L5.25 0z'/%3e%3c/svg%3e");  
}
```

chnage color : fill='rgb (3,122,247)'

Поделиться Подписаться

отредактировано 26 декабря
2022 г. в 7:40

ответил 26 апреля 2021 г. в 11:12



Фахиме Ахмади

839 8 13

С 2015 года существует простой способ сделать это на обычном JavaScript, используя [JavaScript Fetch API](#)

"Было время, когда **XMLHttpRequest** использовался для выполнения запросов API. Используя **jQuery**, вы могли бы использовать более чистый синтаксис `jQuery.ajax()`. **Теперь в JavaScript есть собственный встроенный способ** отправлять запросы API." ~ [Digital Ocean](#)

1. HTML: Используйте `` теги и присваивайте класс " `svg` " всем тем `` тегам, источником которых является SVG-файл.

```

```

2. JavaScript: Поместите этот код в отдельный файл или встроенный, в конце `<body>` или в `<head>`, не имеет значения.

```
window.onload = () => {  
  // Find all <img> elements with a class of "svg"  
  const imgSVGs = document.querySelectorAll("img.svg");  
  
  // Loop over each found element  
  imgSVGs.forEach((img) => {  
  
    // Fetch the SVG file  
    fetch(img.getAttribute("src"))  
      .then(response => response.text())  
      .then(svgContent => {  
        // Parse the XML data from the SVG file into a DOM object for easier  
evaluation  
        const parser = new DOMParser();  
        const svgXML = parser.parseFromString(svgContent, 'text/xml');  
  
        // Create a new <svg> element  
        const svg = document.createElement("http://www.w3.org/2000/svg",  
"svg");  
  
        // Copy attributes from <img> to <svg>: class, alt, id
```

```

        // Remove the helper class "svg"
        svg.setAttribute("class", img.getAttribute("class"));
        svg.classList.remove("svg");
        if (img.getAttribute("id"))
            svg.setAttribute("id", img.getAttribute("id"));
        if (img.getAttribute("alt"))
            svg.setAttribute("alt", img.getAttribute("alt"));

        // Copy data from SVG file to <svg>: viewBox attribute, SVG path
        svg.setAttribute("viewBox", svgXML.getElementsByTagName("svg")
[0].getAttribute("viewBox"));
        const path = svgXML.getElementsByTagName("svg")
[0].querySelector("path");
        svg.appendChild(path);

        // Replace <img> with <svg>
        img.replaceWith(svg);
    })
    .catch(error => { console.log("Error fetching SVG: ", error); });

});
};

```

Этот короткий скрипт преобразует все `` элементы с классом `svg` в настоящие `<svg>` элементы, которые затем можно стилизовать с помощью CSS, вот так:

```

svg:hover {
    fill: red;
}

```

Он копирует атрибуты `id`, `alt` и `class` из `` в `<svg>` (если они присутствуют) и берет `<path>` элемент и `viewBox` атрибут из исходного SVG-файла.

✓ То же поведение, что и в принятом коде jQuery, но только с JavaScript, без фреймворков.

Поделиться Подписаться

отредактировано 23 августа в 19:37

ответил 23 августа в 18:47



Ichristmann
141 1 10



Очень активный вопрос. Заработайте 10 репутации (не считая [бонуса за ассоциацию](#)), чтобы ответить на этот вопрос. Требование репутации помогает защитить этот вопрос от спама и действий, не позволяющих ответить.