**Система иконок с SVG-спрайтами**

* [Сайтостроение](https://www.internet-technologies.ru/)
* [Статьи](https://www.internet-technologies.ru/articles/)
* [Веб-дизайн](https://www.internet-technologies.ru/articles/webdesign/)

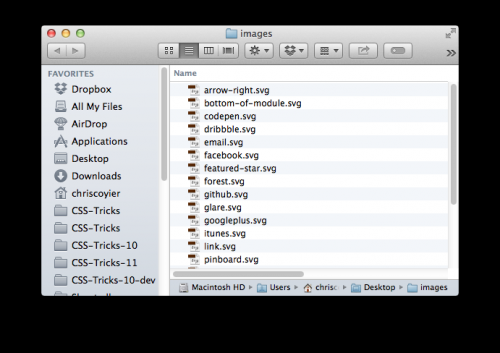
31.03.148.2K

Я всегда был приверженцем шрифтов иконок. Многим сайтам действительно необходима система упорядочивания иконок, и шрифты иконок предлагают для этого отличную возможность.

И даже если в IE версии 9 и выше нормально заработают встроенные **SVG** и элементы <use>для ссылок, использование иконок остается более прогрессивной практикой.

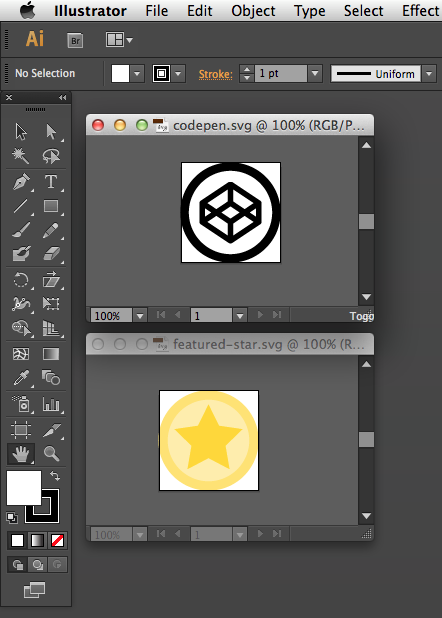
Прежде всего, давайте рассмотрим, как это работает.

Для того чтобы работать с иконками, можно создать папку с **.svg файлами**:



*Это одна из самых крутых вещей в работе с SVG — они сами являются исходными файлами.*

**SVG**-файлы могут быть цветными, черно-белыми, разных форм, размеров, какими угодно:



Вы можете через **Illustrator** (или другую программу) сохранить их как угодно со всеми сопутствующими элементами:

<?xml version="1.0" encoding="utf-8"?>

<!-- Генератор: Adobe Illustrator 16.0.4, плагин экспорта SVG. 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">

<svg version="1.1" id="Layer\_1" xmlns="http://www.w3.org/2000/svg" xmlns:xlink="http://www.w3.org/1999/xlink" x="0px" y="0px"

width="100px" height="100px" viewBox="0 0 100 100" enable-background="new 0 0 100 100" xml:space="preserve">

<g>

<path d="M50.049,0.3c14.18,0.332,25.969,5.307,35.366,14.923S99.675,36.9,100,51.409c-0.195,11.445-3.415,21.494-9.658,30.146 - yadda yadda yadda"/>

</g>

</svg>

**Содержание**

* [**Создание .svg-файла**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-1)
* [**Подключение SVG в верхней части документа**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-2)
* [**Использование иконок в любом месте**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-3)
* [**Ура: вы можете задавать их стили (и стили их частей) с помощью CSS**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-4)
* [**Другой способ: IcoMoon**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-5)
* [**Поддержка браузерами**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-6)
* [**Делать следующим образом будет намного правильнее**](https://www.internet-technologies.ru/articles/sistema-ikonok-s-svg-spraytami.html#header-7)

**Создание .svg-файла**

Если захотите, вы можете сделать это вручную. Я так и сделал. Вам даже не нужно просматривать конечный файл. Просто назовите его **svg-defs.svg** или как-то в этом роде.

В этом файле должен быть только тег <svg> с тегом <defs> (который просто означает, что вы определяете элементы для последующего использования), а затем связка тэгов <g> (группа).

Каждый тег <g> будет иметь уникальный идентификатор, и включать полный маршрут к файлу, а также содержать еще много дополнительных данных для каждой иконки:

<svg>

<defs>

<g id="shape-icon-1">

<!-- все пути и формы, а также другие данные для этой иконки -->

<g>

<g id="shape-icon-2">

<!-- все пути и формы, а также другие данные для этой иконки -->

<g>

<!-- и так далее -->

</defs>

</svg>

Опять же вы можете сделать это вручную, но это, конечно, немного трудоемкий процесс. Фабрис Вайнберг создал плагин [grunt-svgstore](https://github.com/FWeinb/grunt-svgstore), который автоматизирует данную процедуру.

Если вы никогда не использовали **Grunt**, можете попробовать его. [Вот ссылка на скринкаст](http://css-tricks.com/video-screencasts/130-first-moments-grunt/), который поможет вам разобраться, как это сделать.

С помощью данной команды вы можете установить **Grunt**:

npm install grunt-svgstore --save-dev

Проверьте, доступны ли в нем все задачи:

grunt.loadNpmTasks('grunt-svgstore');

А затем задайте конфигурацию:

svgstore: {

options: {

prefix : 'shape-', // Это префикс для каждого идентификатора тега <g>

},

files: {

'processed/svg-defs.svg': ['source/\*.svg']

}

},

В исходном файле **svg-defs.svg** каждая иконка (независимо от пути и прочих атрибутов источника. **SVG** файла) заключается в тег с уникальным идентификатором, включающим префикс и имя файла (без расширения **.svg**).

Например:

<g id="shape-codepen">

**Подключение SVG в верхней части документа**

Подключается **.SVG файл** следующим образом:

<!DOCTYPE html>

<html lang="en">

<head>

...

</head>

<body>

<?php include\_once("processed/svg-defs.svg"); ?>

Или любым другим способом по вашему усмотрению.

Код должен размещаться в верхней части. К сожалению, [в Chrome есть один баг](https://code.google.com/p/chromium/issues/detail?id=349175), из-за которого это не будет работать. С этим мы разберемся позже.

**Использование иконок в любом месте**

Теперь вы можете использовать иконки везде(!), где вам нравится:

<svg viewBox="0 0 100 100" class="icon shape-codepen">

<use xlink:href="#shape-codepen"></use>

</svg>

Убедитесь, что вы используете соответствующие имена классов **SVG** для задания размеров:

/\* Вы можете делать здесь любые доступные действия.

Просто знайте, что svg будут

выводиться в огромном размере на 100%,

если вы не зададите размер отдельно. \*/

.icon {

display: inline-block;

width: 25px;

height: 25px;

}

**Ура: вы можете задавать их стили (и стили их частей) с помощью CSS**

Одна из причин, по которым мне очень нравятся шрифты иконок, это то, что вы можете оформлять их с помощью CSS.

Эта техника одного окна, которая позволяет нам делать, все что нужно, и даже больше этого:

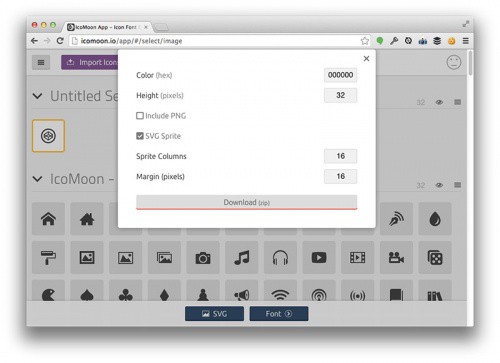
1. Мы можем задавать стили для отдельных частей;
2. SVG имеет намного больше элементов, которыми вы можете управлять, таких как специальные фильтры и кисти.

**SVG** представляет собой своего рода **DOM**, так же как **JavaScript**. Вот некоторые возможности для оформления стилей и их демонстрация на практике: [Посмотреть пример на Codepen](http://codepen.io/chriscoyier/pen/EBHlD).

**Другой способ: IcoMoon**

[IcoMoon](http://icomoon.io/app/), который известен, как инструмент для создания шрифтов иконок, на самом деле также предоставляет фантастические возможности по созданию **SVG-спрайтов**.

После выбора необходимых шрифтов просто нажмите кнопку **SVG** внизу окна приложения, и на выходе вы получите набор элементов, в том числе демо-страницу, созданную с помощью встроенного **SVG-метода**:



**Поддержка браузерами**

Что касается поддержки браузерами, то здесь зоной риска являются IE версии 8 и ниже, Safari 5 и ниже, IOS 4.3 и ниже и Android 2.3 и ниже.

Но если вы ориентируетесь на пользователей «*двух последних основных версий*»- то, как правило, в подавляющем большинстве браузеров **SVG** поддерживаются.

Следует помнить, что иконки можно использовать в качестве вспомогательного элемента. В этом случае поддержка браузерами не имеет критического значения.

Если это будет отдельный основной элемент, без которого содержимое сайта не может отображаться, это может стать проблемой.

Я мог бы еще много рассказать о поддержке браузерами шрифтов иконок. Однако просто убедитесь, что вы применяете их правильно.

**Делать следующим образом будет намного правильнее**

В идеале у нас должно быть что-то наподобие следующего:

<svg viewBox="0 0 100 100" class="icon shape-codepen">

<use xlink:href="http://cdn.css-tricks.com/images/svg-defs.svg#shape-codepen"></use>

</svg>

Это действительно работает в некоторых браузерах, то есть вы могли бы и не включать их в верхней части документа.

Это означает, что будет отправляться дополнительный HTTP-запрос, но тогда вы можете более эффективно использовать кэширование (не раздувать кэш документа).

При тестировании Джонатан Нил обнаружил, что для нормальной работы в тег <svg>необходимо включать **xmlns-атрибут**:

<svg xmlns="http://www.w3.org/2000/svg">

Но даже в этом случае иконки в любой версии IE не поддерживаются. Если вы не хотите поменять все элементы <svg><use>, которые выполняют работу, в <object>.

Джонатан Нил [установил и этот факт](http://sandbox.thewikies.com/svg/):

/MSIE|Trident/.test(navigator.userAgent) && document.addEventListener('DOMContentLoaded', function () {

[].forEach.call(document.querySelectorAll('svg'), function (svg) {

var use = svg.querySelector('use');

if (use) {

var object = document.createElement('object');

object.data = use.getAttribute('xlink:href');

object.className = svg.getAttribute('class');

svg.parentNode.replaceChild(object, svg);

}

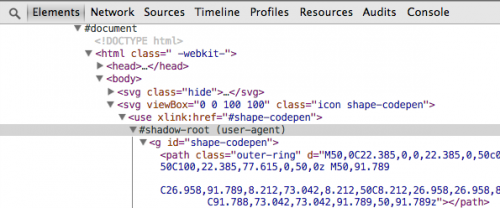
});

});

Его [демо-версия](http://sandbox.thewikies.com/svg/) теперь содержит также метод, который формирует Ajax-запрос к содержимому и включает этот блок, который позволяет обрабатывать заливки и в IE 9. Не так эффективно, скорее как полизаливки.

Я полагаю, что когда-нибудь этот вопрос можно будет решить непосредственно связав <svg><use> с **.SVG**. Или даже через <img>, работающими с идентификаторами фрагментов URL-адресов в **SVG**.

Браузеры воспринимают <use> как своего рода **DOM**:



Теперь, мы можем определять, скажем, конкретный <path> с помощью CSS:

.targetting-a-path {

fill: red;

}

Но это будет влиять на все объекты этого пути. Вы могли бы придумать что-то на манер:

svg.shape-version-2 .targetting-a-path {

fill: red;

}

Но это не работает. Это выходит за пределы **DOM**. В идеале нужно использовать селектор «*hat*«:

svg.shape-version-2 ^ .targetting-a-path {

fill: red;

}

Но в одних браузерах он не поддерживается, в других не понятно, как будет работать, и будет ли работать вообще.

**«Минусы» шрифтов иконок**

**Векторная основа**: ничья

**Стили через CSS**: Стили через CSS имеют небольшое преимущество перед **SVG-спрайтами**(ориентированные части, специальные элементы стиля **SVG**, такие как кисти).

**Сбои в работе**: На первый взгляд **SVG** кажется простым в работе (если поддерживается браузером).

Однако иногда с ними происходят странные вещи. Например, если вы используете карту символов, как с обычными буквами, то при загрузке шрифтов может произойти сбой, и вы получите при выводе большое количество случайных символов.

Или применяете карту «*Для области частного использования*», и некоторые браузеры могут повторно привязать их к [действительно специфическим символам, таким как розы](http://cdn.css-tricks.com/wp-content/uploads/2014/03/icon-font-fail.png), однако при повторной привязке могут возникнуть сложности с воспроизведением оригинальной структуры.

Или вы захотите разместить файлы **@font-face** на **CDN**, но они имеют кросс основу, а [Firefox](https://www.mozilla.org/ru/firefox/new/?utm_source=firefox-com&amp;utm_medium=referral)ненавидит это. Таким образом, вам потребуется собственный сервер для корректного обслуживания кросс-заголовков.

Но установки **Nginx** могут подхватить их неправильно. К сожалению.

**Семантика**: Не имеет большого значения, но я все же думаю, что использование для изображений <svg> имеет больше смысла, чем <span>.

**Применимость**: Может быть, кто-нибудь подскажет мне? Можем ли мы / могли бы мы придать элементу <svg> атрибут заголовка или что-то в этом роде?

Или элемент <text>, который был бы визуально спрятан?

**Обновление**: элемент <title> мы [могли бы придать](https://twitter.com/dirkschulze/status/443769999153303552). Или, возможно, элемент <desc>, использование которого описано в этой [спецификации доступа SVG](http://www.w3.org/TR/SVG-access/#Fig-3.5).

**Простота в использовании**: Инструменты типа [Fontello](http://fontello.com/) и [IcoMoon](http://icomoon.io/) хороши для работы со шрифтами иконок, но использование специальной папки **SVG**, я думаю, еще проще.