GitHub This repository Search Explore Features Enterprise Blog Sign up Sign in

lauren / **pick-a-color**

Watch 20 Star 194 Fork 67

an easy-to-use jQuery color picker for Twitter Bootstrap http://lauren.github.com/pick-a-color/

198 commits **3** branches **14** releases **6** contributors

branch: **master** **pick-a-color** / +

readme update

lauren authored on 25 Apr 2014 `latest commit ec5e90483a`

build	add option for inline dropdown on desktop. fixes #45.	a year ago
docs	add option for inline dropdown on desktop. fixes #45.	a year ago
src	add option for inline dropdown on desktop. fixes #45.	a year ago
.gitignore	clean the build mess	a year ago
Gruntfile.js	add bootstrap-src to the Gruntfile.js watch task	2 years ago
LICENSE	add package.json	2 years ago
README.md	readme update	a year ago
bower.json	add description to bower manifest	a year ago
package.json	add option for inline dropdown on desktop. fixes #45.	a year ago
pick-a-color.jquery.json	add option for inline dropdown on desktop. fixes #45.	a year ago
sample.html	add option for inline dropdown on desktop. fixes #45.	a year ago

Code

Issues 33

Pull requests 10

Pulse

Graphs

HTTPS clone URL

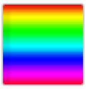
`https://github.com/l`

You can clone with HTTPS or Subversion.

Clone in Desktop

Download ZIP

README.md



Pick-a-Color: a jQuery color picker for Twitter Bootstrap

For a documentation-reading experience that includes rainbow gradients and live examples, check out the official docs at http://lauren.github.io/pick-a-color.

There are some great color picker plugins out there, but most cater to the needs of techies and designers, providing complicated controls to access every color imaginable.

Pick-a-Color is designed to be easy for anyone to use. The interface is based on Twitter Bootstrap styles so it looks lovely with the styles of almost any site.

Features

For your site's users

Flexible text entry: Accepts HEX, RGB, HSL, HSV, HSVA, and names, thanks to Brian Grinstead's amazing Tiny Color library.

Saved colors: Saves up to 16 recently used colors. Colors are stored in localStorage or cookies.

Advanced: Advanced tab lets users modify hue, saturation, and lightness to make any color their hearts desire.

Basic color palette: Easy-to-use preset colors that can be lightened and darkened.

Chunky mobile styles: Dragging is easy, even on a touch device.

For you

Tested: Tested in Chrome (Mac/PC/iOS), Safari (Mac/iOS), IE 8+, Firefox (Mac/PC), and Opera (Mac/PC).

No conflicts: Anonymous JavasScript function and namespaced CSS won't mess up your code.

Simple initialization: As little as one line of HTML and one line of JavaScript.

Done: You didn't have to write your own color picker. 'Nuff said.

What? Why? Who?

I'm Lauren. I originally wrote Pick-a-Color for my friends at Broadstreet Ads because they needed a color picker that was easy for people at online publishing companies to use.

If you have any bugs to report in Pick-a-Color, let me know by making a ticket here: https://github.com/lauren/pick-a-color/issues/new

Pick-a-Color is available under the MIT License: https://github.com/lauren/pick-a-color/blob/master/LICENSE

OK! Let's do this color picking thing!

How to Use

Bower

```
bower install pick-a-color
```

Compiled CSS For Use With Default Bootstrap Settings

1) Download the required files: https://github.com/lauren/pick-a-color/archive/master.zip. Add the CSS and JS from the latest release in /build to your CSS and JS folders and include them in your document as follows:

For Bootstrap 3, use Pick-a-Color 1.2.3:

In the `<head>` :

```
<link rel="stylesheet" href="css/bootstrap-3.0.0.min.css">
<link rel="stylesheet" href="css/pick-a-color-1.2.3.min.css">
```

Before the ending `</body>` :

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<script src="js/tinycolor-0.9.14.min.js"></script>
<script src="js/pick-a-color-1.2.3.min.js"></script>
```

For Bootstrap 2, use Pick-a-Color 1.1.8:

In the `<head>` :

```
<link rel="stylesheet" href="css/bootstrap-2.2.2.min.css">
<link rel="stylesheet" href="css/pick-a-color-1.1.8.min.css">
```

Before the ending `</body>` :

```
<script src="//ajax.googleapis.com/ajax/libs/jquery/1.9.0/jquery.min.js"></script>
<script src="js/tinycolor-0.9.14.min.js"></script>
<script src="js/pick-a-color-1.1.8.min.js"></script>
```

2) Add this to your HTML wherever you want a Pick-A-Color. Replace `YOUR-NAME` with your unique identifier for the color picker (e.g. "border-color" or "background-color") and `YOUR-DEFAULT` with the default color you'd like to show in the color picker:

```
<input type="text" value="YOUR-DEFAULT" name="YOUR-NAME" class="pick-a-color form-control
```

For instance, yours might look like this:

```
<input type="text" value="222" name="border-color" class="pick-a-color form-control">
```

Notes:

i) If you don't provide a `name` attribute, one will be added in the pattern "pick-a-color-INT," where INT is (unoriginally enough) the index of your Pick-a-Color on the page, starting from 0.

ii) You can change the class of your `input` , but make sure to match it in your JavaScript in the next step and be aware that the class "pick-a-color" will be added regardless...

3) Add this to your JavaScript somewhere after the DOM is ready. Make sure the class selector matches the class of your div:

```
$(".pick-a-color").pickAColor();
```

4) To optimize IE and mobile support, I recommend adding these tags to your `<head>` :

```
<meta http-equiv="x-ua-compatible" content="IE=10">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Ta-da! You have a color picker! You might even have several!

Source LESS for Use with Customized Bootstrap

1) Download the source: https://github.com/lauren/pick-a-color/archive/master.zip.

2) Add `src/less/pick-a-color.less` to your LESS folder.

3) Update the import statements in lines 7 and 8 of `pick-a-color.less` from:

```
@import "bootstrap-src/variables.less";
@import "bootstrap-src/mixins.less";
```

To:

```
@import "PATH/TO/YOUR/variables.less";
@import "PATH/TO/YOUR/mixins.less";
```

4) Compile `pick-a-color.less` using your customized variables.

5) Return to Step 1 of instructions for use of the compiled CSS.

Sample HTML

Here's an example of how a simple HTML page using Pick-a-Color might look:

```
<!doctype html>
<html>
    <head>
        <meta name="viewport" content="width=device-width, initial-scale=1.0">
        <meta http-equiv="x-ua-compatible" content="IE=10">
        <link rel="stylesheet" href="css/bootstrap-2.2.2.min.css">
        <link rel="stylesheet" href="css/pick-a-color-1.1.5.min.css">
    </head>

    <body>

        <input type="text" value="222" name="border-color" class="pick-a-color form-contr
        <input type="text" value="aaa" name="font-color" class="pick-a-color form-control
        <input type="text" value="a1beef" name="backgound-color" class="pick-a-color form
        <input type="text" value="551033" name="highlight-color" class="pick-a-color form
        <input type="text" value="eee" name="contrast-color" class="pick-a-color form-con
        <input type="text" class="pick-a-color form-control">
        <input type="text" class="pick-a-color form-control">

        <script src="js/jquery-1.9.1.min.js"></script>
        <script src="js/tinycolor-0.9.14.min.js"></script>
        <script src="js/pick-a-color-1.1.5.min.js"></script>

        <script type="text/javascript">

            $(document).ready(function () {

            $(".pick-a-color").pickAColor();

            });

        </script>

    </body>
</html>
```

Events

Each time a user chooses a new color (or enters one manually), there will be a `change` event on the input field.

Here's sample code for accessing the new color from a Pick-a-Color initialized with the name `border-color` :

```
$("#border-color input").on("change", function () {
    console.log($(this).val());
});
```

Options

If you'd like to change any of my default options, you can specify your preferred settings like this:

```
$(".pick-a-color").pickAColor({
        showSpectrum          : true,
        showSavedColors       : true,
        saveColorsPerElement  : false,
        fadeMenuToggle        : true,
        showAdvanced          : true,
        showBasicColors       : true,
        showHexInput          : true,
        allowBlank            : true
 });
```

showSpectrum

Specifies whether or not there is a spectrum next to each basic color allowing users to lighten and darken it.

showSavedColors

Specifies whether or not there is a tab called "Saved Colors" that keeps track of the last 16 colors a user customized.

showAdvanced

Specifies whether or not there is a tab called "Advanced" that allows users to modify hue, lightness, and saturation to make any color their hearts desire.

saveColorsPerElement (for showSavedColors only)

If set to `false` : Every Pick-a-Color on a page will show the same set of saved colors, which will be updated continuously as users customize colors.

If set to `true` : Each Pick-a-Color will get its own set of saved colors, which will be updated as users customize colors. The colors are saved across pageviews using the data-attribute as the key. If this is set to `true` but you don't have a data-attribute in your initializing HTML, your Pick-a-Colors will behave as if the setting was false.

I recommend setting this to `false` . Imagine you're a user filling out a big form to configure a custom page: You find the perfect color for your background. Five fields later, you want to use that same color for your link hover state. It'd be pretty nice if it was hanging out in your Saved Colors tab.

fadeMenuToggle

Specifies whether or not the dropdown menu should fade in and out when it's opened and closed. This setting is overridden for mobile devices, in which Pick-a-Color never ever ever uses a fade because WOW they look terrible in mobile browsers.

showBasicColors

Specifies whether or not the dropdown should show a list of basic colors that the user can select from. Thanks to Ryan Johnson for adding this feature!

showHexInput

Specifies whether or not to show the hex text input. If false the input has an input type of 'hidden'. Thanks to Ryan Johnson for adding this feature!

allowBlank

Specifies whether or not the field can be left blank. Use this if the color input is not a required field. Thanks to San for adding this feature!

inlineDropdown

Specifies whether the dropdown should be shown below the input field (instead of the default, to the right) on desktop browsers.

Tested Browsers

I've tested Pick-a-Color in these browsers:

- Google Chrome 24.0.1312.57 - 32.0.1700.107 (Mac OSX, Windows 7, Windows XP, iOS 6.0.2)
- Safari 6.0.1 - 6.1.0 (Mac OSX and iOS 6.0.2)
- Internet Explorer 10 (Windows 7)
- Internet Explorer 9 (Windows 7)
- Internet Explorer 8 (Windows XP)

- Firefox 18.0.1 - 26.0 (Mac OSX and Windows 7)
- Opera 12.13 - 12.14 (Mac OSX and Windows 7)

Minor issues in these browsers are documented here: https://github.com/lauren/pick-a-color/issues

The only major platform I haven't been able to test yet is Android. I'm working on it.

Notes on IE Support

I highly recommend using the X-UA-COMPATIBLE tag in your html `<head>` to ensure that Internet Explorer 8 and higher use their own "Browser Mode" instead of switching to the Browser Mode of a previous version. It works like this:

```
<meta http-equiv="x-ua-compatible" content="IE=10">
```

Notes on Mobile Support

You must use a viewport tag in your html `<head>` for content to be displayed at the correct size in a mobile browser. It works like this:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

Contributing

I ♥ pull requests. Here's how to contribute:

1. Fork and pull repo.
2. If you don't have node, install it.
3. From the repo directory, `npm install --save-dev`
4. Update version number in package.json and pick-a-color.jquery.json. Otherwise, grunt will overwrite the current version with your changes.
5. `grunt watch` : This will automatically JSHint, concatenate, and minify the JavaScript and automatically compile and minify the LESS every time you save. Watch for JSHint errors and correct them.
6. Code!
7. Use sample.html to check your changes.
8. Copy /build/{previous-ver}/sample.html to /build/{your-ver}/sample.html and update the version number in your JS and CSS links.