

Mini border radius

kounelios13

Published
with GitBook



Table of Contents

1. [Introduction](#) 0
2. [What is mini border radius](#) 1
3. [How do I install it](#) 2
4. [Dependencies](#) 3
5. [So now what?](#) 4
6. [Wait.Nothing happened.](#) 5
7. [Generator methods](#) 6

Introduction

Mini border radius generator

In this manual you will find anything you need to know about this javascript plugin

What is mini border radius

What is mini border radius?

Mini border radius is a small javascript plugin written in JQuery that allows you to add one or more border radius generator in a web page or in a web app and it is free

.

How do I install it

How do I install it

Installation is not that hard. All you have to do is to include the following tags in your html file:

1. `<script src="path/to/miniBorderRadius.js"></script>`
2. `<link rel="stylesheet" href="path/to/miniBorderRadius.css">`

Dependencies

Dependencies

Since this plugin is written in jQuery you need to have jQuery in your web page. Also this plugin uses some bootstrap css components you need to also have bootstrap in your webpage

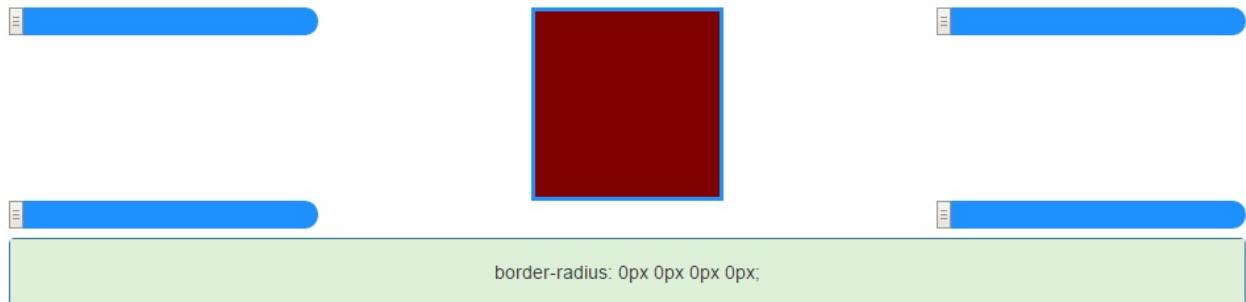
So now what?

So now what?

So you installed mini border radius. Right? Well there are a few things left to do. Each generator needs its own div to live on. So to create a new generator you have to do it like this:

```
var generator;  
  
$(document).ready(function(){generator = new Generator("#sample");  
});
```

This is how a default generator looks like:



Beware that you have to create the generator inside the .ready() function();

Instead of using the id of the host-parent div you can pass a list of 5 elements in the constructor like this:

```
generator=new  
Generator([host_div_id,height,width,max_value,bg_color]);
```

Now the first parameter of the list is the id of the host-parent div. The next 2 parameters determine the size of the generator in the page. The fourth value determines the max value for the sliders inside the generator and the last one determines the background color of the generator. There is also a third constructor (3rd way to create a generator) which takes 2 arguments. The first one can be the id of the host-parent div or it can be a list like the previous list. The second one parameter is the id of an existing object in your page. What is this object? This object is going to replace the default object that the generator uses to apply the border-radius property.

```
Example: generator=new  
Generator([host_div_id,height,width,max_value,bg_color],"#image");
```

Let me tell you what the above code does. It looks for an html element with the id "#image". It removes it from where it is and it replaces the default object used by the generator with this element.

Wait.Nothing happened.

Wait.Nothing happened.

So you have followed all the instructions up to here only to see that the generator is visible in your page but it is not working.This is absolutely normal.Creating a generator does not mean that is active.To activate generator you have to write the following code:

```
generator.activateGenerator();
```

After this you are ready to go

Generator methods

Generator methods

Each generator has some methods:

.activateGenerator()

This is the main function that activates the generator.Chainable:Yes.

.deactivateGenerator()

This function resets and deactivates the generator so before executing that method be sure you have saved somewhere the code produced by the generator.Chainable:Yes.

.reset()

Resets any slider inside a generator and set the border radius property of the object used by the generator to 0.Chainable:Yes

.replaceObject(object,restrict_size)

Replace the default object where border radius is applied with another one.If you want the new object to have the same size as the previous one pass true as the second parameter in the function.
.Chainable:Yes

.setMax(value)

Set the max value for all the sliders inside a generator. Chainable:Yes

.setSize(height,width)

Sets the size of a generator.Chainable:Yes

.setBackground(color)

Sets the background color of a generator.Chainable:Yes

.init(options)

This function accepts a list of 4 default options:height,width,max_value,default_background.Chainable:Yes

.getCode()

Gets the code generated from a miniRadiusGenerator.Chainable:No

.getId()

Returns the id of the host div.Chainable:No

.addToFavourites()

Adds the current code produced by the generator into a list.Chainable:Yes

.getFavourites()

Returns favourites list.Chainable:No

.removeFavourites()

Removes favourites.Chainable:Yes

.toggleFavourites()

Toggle the list of favourites created by the current generator.Chainable:Yes <<<<<<< HEAD If you want to know whether a method is chainable you can call the `isChainable()` function and pass the name of the method as an argument and you will get true or false.

For example:

`isChainable("activateGenerator")` will return true

If you want to know whether a function is chainable or not you can call the `isChainable` function and pass the name of the function you want to know about. You will get a true or false.

For example: `isChainable("getId")` will return false.

origin/manual_book