FREE eBook

LEARNING

# fabricjs

Free unaffiliated eBook created from
Stack Overflow contributors.

#fabricjs

Table of Contents

About 1

Chapter 1: Getting started with fabricjs 2

Remarks 2

Examples 2

Installation or Setup 2

Hello World 2

Chapter 2: Fabricjs canvas events 4

Syntax 4

Parameters 4

Remarks 4

Examples 4

Fabric js canvas events Demo 4

Chapter 3: FabricJS canvas Serialization 6

Introduction 6

Syntax 6

Remarks 6

Examples 6

Canvas Serilization 6

Credits 8

About

You can share this PDF with anyone you feel could benefit from it, downloaded the latest version from: fabricjs

It is an unofficial and free fabricjs ebook created for educational purposes. All the content is extracted from Stack Overflow Documentation, which is written by many hardworking individuals at Stack Overflow. It is neither affiliated with Stack Overflow nor official fabricjs.

The content is released under Creative Commons BY-SA, and the list of contributors to each chapter are provided in the credits section at the end of this book. Images may be copyright of their respective owners unless otherwise specified. All trademarks and registered trademarks are the property of their respective company owners.

Use the content presented in this book at your own risk; it is not guaranteed to be correct nor accurate, please send your feedback and corrections to info@zzzprojects.com

Chapter 1: Getting started with fabricjs

Remarks

fabric.js is a powerful and quite simple javascript library for HTML5 canvas. It provide a interactive platform to work with the HTML5 canvas. Using fabric you can create object/shapes on canvas from simple geometrical shapes to complex shapes. You can even work with images using fabric.js. fabric.js also supports animations. It provides controls to work with images like dragging them, scaling them rotating them. It allows grouping of shapes/objects that can be manipulated together. it supports events not only limited to the canvas element but with objects as well. It even provide with functionlaity to serialize the canvas to SVG or JSON and reuse it as and when needed. It even supports running fabric under Node.js, with help of jsdom and node-canvas libraries.

Examples

Installation or Setup

Fabric.js is just like any other JS library just specific to canvas. Easy to setup and get started. All you need to do is download the fabric.js from HERE and include it in your project just like any other JS library for example the way you do it for jQuery. Then create the html file suppose index.html like:

```
<head>
    <script src="fabric.js"></script>
</head>

<body>
    <canvas id="canvas" width="400" height="400" style="border:2px solid #000000</canvas>
    <script>
        var canvas = new fabric.Canvas('canvas');
        canvas.add(new fabric.Circle({ radius: 30, fill: '#f55', top: 100, left: 100 }));
    </script>
</body>
```

the src attribute in the script is referring to a file fabric.js kept in the same folder you can keep it in some other folder and give a relative path. And you are good to go. For more about Fabric visit Offcial Page. Here is a very basic Demo

Hello World

```
<canvas id="c" width="400" height="400"></canvas>
var canvas = new fabric.Canvas("c");
var text = new fabric.Textbox('Hello world From Fabric JS', {
            width:250,
            cursorColor :"blue",
            top:10,
            left:10
```

```
        });
canvas.add(text)
```

The example creates a text `'Hello world From Fabric JS'` using fabricjs. Demo

Read Getting started with fabricjs online: https://riptutorial.com/fabricjs/topic/7518/getting-started-with-fabricjs

Chapter 2: Fabricjs canvas events

Syntax

1. **on(eventName, handler)** - Attaches an event listener with a callback to the object.
2. **off(eventName, handler)** - Remove the event listener from the object. Calling this function witout any arguments will remove all event listeners on the object.
3. **trigger(eventName, optionsopt)** - Fires the event and optional options object.

Parameters

Parameter	Description
`eventName`	The name of the event you want to subscribe such as 'object:moving'
`eventHandler`	The function you want to execute when that particluar event is triggered
`optionsopt`	Options object

Remarks

Fabric supports a number of events to allow for interactivity and extensibility. In order to subscribe to events for a canvas use the `on` method the way its used in jQuery. And you wish to manually trigger any event use the `trigger` method. All the events are within the scope of a particular canvas instance. Visit Link for more information on events

Examples

Fabric js canvas events Demo

```
<canvas id="c" width="400" height="400"></canvas>
```

```
var canvas = new fabric.Canvas("c");

canvas.on('mouse:up', function () {
  console.log('Event mouse:up Triggered');
});

canvas.on('mouse:down', function () {
  console.log('Event mouse:down Triggered');
});

canvas.on('after:render', function () {
  console.log('Event after:render Triggered');
});

canvas.on('object:moving', function () {
  console.log('Event object:moving Triggered');
});

canvas.on('object:modified', function () {
  console.log('Event object:modified Triggered');
});

var text = new fabric.Textbox('Hello world', {
  width:250,
  cursorColor :"blue"
});
canvas.add(text);
```

The code above displays how the event API in Fabric.js works. By calling `on` on the canvas instance, or even on the Fabric.js other objects, such as `Rect` instance, you can listen to their events and when the listeners are triggered, the callback you passed to them will be triggered as well.

Read Fabricjs canvas events online: https://riptutorial.com/fabricjs/topic/7532/fabricjs-canvas-events

Chapter 3: FabricJS canvas Serialization

Introduction

Once you have started building an application with FabricJS it won't be late when you realize that you will need to save results of the canvas/its contents to server. Or may exporting them as files to the client. For this very purpose you will need **canvas serialization**. Now some of you might ask Why Serialization we can save it as an image. So the answer would Saving image to a server needs bandwidth and text proves much better in this case.

Syntax

1. **JSON.stringify(canvas)** - implicitly calls toJSON method on passed object. Gives String representation
2. **canvas.toObject()** - returns the same representation as toJSON, only in a form of actual object
3. **canvas.toSVG()** - returns an SVG representation of the canvas

Remarks

For more about FabricJS canvas serialization Refer the link Canvas Serialization

Examples

Canvas Serilization

```
<canvas id = "canvas" height='400' width='500'></canvas>

var canvas = new fabric.Canvas(document.getElementById('canvas'));
console.log(JSON.stringify(canvas)); // '{"objects":[],"background":""}'

canvas.add(new fabric.Rect({
  left: 10,
  top: 10,
  height: 50,
  width: 50,
  fill: 'green',
     stroke:'black'
}));
canvas.renderAll();

console.log(JSON.stringify(canvas));//logs the string representation
console.log(canvas.toObject());//logs canvas as an object
console.log(canvas.toSVG());//logs the SVG representation of canvas
```

Fiddle

Read FabricJS canvas Serialization online: https://riptutorial.com/fabricjs/topic/8274/fabricjs-canvas-serialization

Credits

S. No	Chapters	Contributors
1	Getting started with fabricjs	Community, InferOn, Manish, RamenChef
2	Fabricjs canvas events	Ben, InferOn, MahdiPOnline, Manish
3	FabricJS canvas Serialization	Manish