 [Chart.js](http://docs.google.com/docs/3.9.1/)

[Home](http://docs.google.com/docs/3.9.1/)

[API](http://docs.google.com/docs/3.9.1/api/)

[Samples](http://docs.google.com/docs/3.9.1/samples/)

Ecosystem Ecosystem

* [Awesome (opens new window)](https://github.com/chartjs/awesome)
* [Slack (opens new window)](https://chartjs-slack.herokuapp.com/)
* [Stack Overflow (opens new window)](https://stackoverflow.com/questions/tagged/chart.js)

[GitHub (opens new window)](https://github.com/chartjs/Chart.js)

[Home](http://docs.google.com/docs/3.9.1/)

[API](http://docs.google.com/docs/3.9.1/api/)

[Samples](http://docs.google.com/docs/3.9.1/samples/)

Ecosystem Ecosystem

* [Awesome (opens new window)](https://github.com/chartjs/awesome)
* [Slack (opens new window)](https://chartjs-slack.herokuapp.com/)
* [Stack Overflow (opens new window)](https://stackoverflow.com/questions/tagged/chart.js)

[GitHub (opens new window)](https://github.com/chartjs/Chart.js)

* [Chart.js](http://docs.google.com/docs/3.9.1/)
* Getting Started
* General
* Configuration
  + [Configuration](http://docs.google.com/docs/3.9.1/configuration/)
  + [Animations](http://docs.google.com/docs/3.9.1/configuration/animations.html)
  + [Canvas background](http://docs.google.com/docs/3.9.1/configuration/canvas-background.html)
  + [Data Decimation](http://docs.google.com/docs/3.9.1/configuration/decimation.html)
  + [Device Pixel Ratio](http://docs.google.com/docs/3.9.1/configuration/device-pixel-ratio.html)
  + [Elements](http://docs.google.com/docs/3.9.1/configuration/elements.html)
  + [Interactions](http://docs.google.com/docs/3.9.1/configuration/interactions.html)
  + [Layout](http://docs.google.com/docs/3.9.1/configuration/layout.html)
  + [Legend](http://docs.google.com/docs/3.9.1/configuration/legend.html)
  + [Locale](http://docs.google.com/docs/3.9.1/configuration/locale.html)
  + [Responsive Charts](http://docs.google.com/docs/3.9.1/configuration/responsive.html)
  + [Subtitle](http://docs.google.com/docs/3.9.1/configuration/subtitle.html)
  + [Title](http://docs.google.com/docs/3.9.1/configuration/title.html)
  + [Tooltip](http://docs.google.com/docs/3.9.1/configuration/tooltip.html)
* Chart Types
* Axes
* Developers

[**#**](#gjdgxs) Canvas background

In some use cases you would want a background image or color over the whole canvas. There is no built-in support for this, the way you can achieve this is by writing a custom plugin.

In the two example plugins underneath here you can see how you can draw a color or image to the canvas as background. This way of giving the chart a background is only necessary if you want to export the chart with that specific background. For normal use you can set the background more easily with [CSS (opens new window)](https://www.w3schools.com/cssref/css3_pr_background.asp).

config setup plugin

const config = { type: 'doughnut', data: data, plugins: [plugin], };

const config = {  
 type: 'doughnut',  
 data: data,  
 plugins: [plugin],  
};

const data = { labels: [ 'Red', 'Blue', 'Yellow' ], datasets: [{ label: 'My First Dataset', data: [300, 50, 100], backgroundColor: [ 'rgb(255, 99, 132)', 'rgb(54, 162, 235)', 'rgb(255, 205, 86)' ], hoverOffset: 4 }] };

const data = {  
 labels: [  
 'Red',  
 'Blue',  
 'Yellow'  
 ],  
 datasets: [{  
 label: 'My First Dataset',  
 data: [300, 50, 100],  
 backgroundColor: [  
 'rgb(255, 99, 132)',  
 'rgb(54, 162, 235)',  
 'rgb(255, 205, 86)'  
 ],  
 hoverOffset: 4  
 }]  
};

// Note: changes to the plugin code is not reflected to the chart, because the plugin is loaded at chart construction time and editor changes only trigger an chart.update(). const plugin = { id: 'custom\_canvas\_background\_color', beforeDraw: (chart) => { const {ctx} = chart; ctx.save(); ctx.globalCompositeOperation = 'destination-over'; ctx.fillStyle = 'lightGreen'; ctx.fillRect(0, 0, chart.width, chart.height); ctx.restore(); } };

// Note: changes to the plugin code is not reflected to the chart, because the plugin is loaded at chart construction time and editor changes only trigger an chart.update().  
const plugin = {  
 id: 'custom\_canvas\_background\_color',  
 beforeDraw: (chart) => {  
 const {ctx} = chart;  
 ctx.save();  
 ctx.globalCompositeOperation = 'destination-over';  
 ctx.fillStyle = 'lightGreen';  
 ctx.fillRect(0, 0, chart.width, chart.height);  
 ctx.restore();  
 }  
};

config setup plugin

const config = { type: 'doughnut', data: data, plugins: [plugin], };

const config = {  
 type: 'doughnut',  
 data: data,  
 plugins: [plugin],  
};

const data = { labels: [ 'Red', 'Blue', 'Yellow' ], datasets: [{ label: 'My First Dataset', data: [300, 50, 100], backgroundColor: [ 'rgb(255, 99, 132)', 'rgb(54, 162, 235)', 'rgb(255, 205, 86)' ], hoverOffset: 4 }] };

const data = {  
 labels: [  
 'Red',  
 'Blue',  
 'Yellow'  
 ],  
 datasets: [{  
 label: 'My First Dataset',  
 data: [300, 50, 100],  
 backgroundColor: [  
 'rgb(255, 99, 132)',  
 'rgb(54, 162, 235)',  
 'rgb(255, 205, 86)'  
 ],  
 hoverOffset: 4  
 }]  
};

// Note: changes to the plugin code is not reflected to the chart, because the plugin is loaded at chart construction time and editor changes only trigger an chart.update(). const image = new Image(); image.src = 'https://www.chartjs.org/img/chartjs-logo.svg'; const plugin = { id: 'custom\_canvas\_background\_image', beforeDraw: (chart) => { if (image.complete) { const ctx = chart.ctx; const {top, left, width, height} = chart.chartArea; const x = left + width / 2 - image.width / 2; const y = top + height / 2 - image.height / 2; ctx.drawImage(image, x, y); } else { image.onload = () => chart.draw(); } } };

// Note: changes to the plugin code is not reflected to the chart, because the plugin is loaded at chart construction time and editor changes only trigger an chart.update().  
const image = new Image();  
image.src = 'https://www.chartjs.org/img/chartjs-logo.svg';  
const plugin = {  
 id: 'custom\_canvas\_background\_image',  
 beforeDraw: (chart) => {  
 if (image.complete) {  
 const ctx = chart.ctx;  
 const {top, left, width, height} = chart.chartArea;  
 const x = left + width / 2 - image.width / 2;  
 const y = top + height / 2 - image.height / 2;  
 ctx.drawImage(image, x, y);  
 } else {  
 image.onload = () => chart.draw();  
 }  
 }  
};

Last Updated: 8/3/2022, 12:46:38 PM

←  [Animations](http://docs.google.com/docs/3.9.1/configuration/animations.html)   [Data Decimation](http://docs.google.com/docs/3.9.1/configuration/decimation.html)  →