* [Chart.js](http://docs.google.com/)
* [Getting Started](http://docs.google.com/getting-started/)
  + [Installation](http://docs.google.com/getting-started/installation.html)
  + [Integration](http://docs.google.com/getting-started/integration.html)
  + [Usage](http://docs.google.com/getting-started/usage.html)
* [General](http://docs.google.com/general/)
  + [Responsive](http://docs.google.com/general/responsive.html)
  + [Pixel Ratio](http://docs.google.com/general/device-pixel-ratio.html)
  + [Interactions](http://docs.google.com/general/interactions/)
    - [Events](http://docs.google.com/general/interactions/events.html)
    - [Modes](http://docs.google.com/general/interactions/modes.html)
  + [Options](http://docs.google.com/general/options.html)
  + [Colors](http://docs.google.com/general/colors.html)
  + [Fonts](http://docs.google.com/general/fonts.html)
* [Configuration](http://docs.google.com/)
  + [Animations](http://docs.google.com/animations.html)
  + [Layout](http://docs.google.com/layout.html)
  + [Legend](http://docs.google.com/legend.html)
  + [Title](http://docs.google.com/title.html)
  + [Tooltip](http://docs.google.com/tooltip.html)
  + [Elements](http://docs.google.com/elements.html)
* [Charts](http://docs.google.com/charts/)
  + [Line](http://docs.google.com/charts/line.html)
  + [Bar](http://docs.google.com/charts/bar.html)
  + [Radar](http://docs.google.com/charts/radar.html)
  + [Doughnut & Pie](http://docs.google.com/charts/doughnut.html)
  + [Polar Area](http://docs.google.com/charts/polar.html)
  + [Bubble](http://docs.google.com/charts/bubble.html)
  + [Scatter](http://docs.google.com/charts/scatter.html)
  + [Area](http://docs.google.com/charts/area.html)
  + [Mixed](http://docs.google.com/charts/mixed.html)
* [Axes](http://docs.google.com/axes/)
  + [Cartesian](http://docs.google.com/axes/cartesian/)
    - [Category](http://docs.google.com/axes/cartesian/category.html)
    - [Linear](http://docs.google.com/axes/cartesian/linear.html)
    - [Logarithmic](http://docs.google.com/axes/cartesian/logarithmic.html)
    - [Time](http://docs.google.com/axes/cartesian/time.html)
  + [Radial](http://docs.google.com/axes/radial/)
    - [Linear](http://docs.google.com/axes/radial/linear.html)
  + [Labelling](http://docs.google.com/axes/labelling.html)
  + [Styling](http://docs.google.com/axes/styling.html)
* [Developers](http://docs.google.com/developers/)
  + [Chart.js API](http://docs.google.com/developers/api.html)
  + [Updating Charts](http://docs.google.com/developers/updates.html)
  + [Plugins](http://docs.google.com/developers/plugins.html)
  + [New Charts](http://docs.google.com/developers/charts.html)
  + [New Axes](http://docs.google.com/developers/axes.html)
  + [Contributing](http://docs.google.com/developers/contributing.html)
* [Additional Notes](http://docs.google.com/notes/)
  + [Comparison Table](http://docs.google.com/notes/comparison.html)
  + [Popular Extensions](http://docs.google.com/notes/extensions.html)
  + [License](http://docs.google.com/notes/license.html)
* [Published with GitBook](https://www.gitbook.com)

[**Legend**](http://docs.google.com/)

Legend Configuration

The chart legend displays data about the datasets that area appearing on the chart.

## Configuration options

The legend configuration is passed into the options.legend namespace. The global options for the chart legend is defined in Chart.defaults.global.legend.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Default | Description |
| display | Boolean | true | is the legend shown |
| position | String | 'top' | Position of the legend. [more...](#gjdgxs) |
| fullWidth | Boolean | true | Marks that this box should take the full width of the canvas (pushing down other boxes). This is unlikely to need to be changed in day-to-day use. |
| onClick | Function |  | A callback that is called when a click event is registered on a label item |
| onHover | Function |  | A callback that is called when a 'mousemove' event is registered on top of a label item |
| reverse | Boolean | false | Legend will show datasets in reverse order. |
| labels | Object |  | See the [Legend Label Configuration](#30j0zll) section below. |

## Position

Position of the legend. Options are:

* 'top'
* 'left'
* 'bottom'
* 'right'

## Legend Label Configuration

The legend label configuration is nested below the legend configuration using the labels key.

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Default | Description |
| boxWidth | Number | 40 | width of coloured box |
| fontSize | Number | 12 | font size of text |
| fontStyle | String | 'normal' | font style of text |
| fontColor | Color | '#666' | Color of text |
| fontFamily | String | "'Helvetica Neue', 'Helvetica', 'Arial', sans-serif" | Font family of legend text. |
| padding | Number | 10 | Padding between rows of colored boxes. |
| generateLabels | Function |  | Generates legend items for each thing in the legend. Default implementation returns the text + styling for the color box. See [Legend Item](#1fob9te) for details. |
| filter | Function | null | Filters legend items out of the legend. Receives 2 parameters, a [Legend Item](#1fob9te) and the chart data. |
| usePointStyle | Boolean | false | Label style will match corresponding point style (size is based on fontSize, boxWidth is not used in this case). |

## Legend Item Interface

Items passed to the legend onClick function are the ones returned from labels.generateLabels. These items must implement the following interface.

{  
 // Label that will be displayed  
 text: String,  
  
 // Fill style of the legend box  
 fillStyle: Color,  
  
 // If true, this item represents a hidden dataset. Label will be rendered with a strike-through effect  
 hidden: Boolean,  
  
 // For box border. See https://developer.mozilla.org/en/docs/Web/API/CanvasRenderingContext2D/lineCap  
 lineCap: String,  
  
 // For box border. See https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/setLineDash  
 lineDash: Array[Number],  
  
 // For box border. See https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/lineDashOffset  
 lineDashOffset: Number,  
  
 // For box border. See https://developer.mozilla.org/en-US/docs/Web/API/CanvasRenderingContext2D/lineJoin  
 lineJoin: String,  
  
 // Width of box border  
 lineWidth: Number,  
  
 // Stroke style of the legend box  
 strokeStyle: Color  
  
 // Point style of the legend box (only used if usePointStyle is true)  
 pointStyle: String  
}

## Example

The following example will create a chart with the legend enabled and turn all of the text red in color.

var chart = new Chart(ctx, {  
 type: 'bar',  
 data: data,  
 options: {  
 legend: {  
 display: true,  
 labels: {  
 fontColor: 'rgb(255, 99, 132)'  
 }  
 }  
}  
});

## Custom On Click Actions

It can be common to want to trigger different behaviour when clicking an item in the legend. This can be easily achieved using a callback in the config object.

The default legend click handler is:

function(e, legendItem) {  
 var index = legendItem.datasetIndex;  
 var ci = this.chart;  
 var meta = ci.getDatasetMeta(index);  
  
 // See controller.isDatasetVisible comment  
 meta.hidden = meta.hidden === null? !ci.data.datasets[index].hidden : null;  
  
 // We hid a dataset ... rerender the chart  
 ci.update();  
}

Lets say we wanted instead to link the display of the first two datasets. We could change the click handler accordingly.

var defaultLegendClickHandler = Chart.defaults.global.legend.onClick;  
var newLegendClickHandler = function (e, legendItem) {  
 var index = legendItem.datasetIndex;  
  
 if (index > 1) {  
 // Do the original logic  
 defaultLegendClickHandler(e, legendItem);  
 } else {  
 let ci = this.chart;  
 [ci.getDatasetMeta(0),  
 ci.getDatasetMeta(1)].forEach(function(meta) {  
 meta.hidden = meta.hidden === null? !ci.data.datasets[index].hidden : null;  
 });  
 ci.update();  
 }  
};  
  
var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 legend: {  
  
 }  
 }  
});

Now when you click the legend in this chart, the visibility of the first two datasets will be linked together.

## HTML Legends

Sometimes you need a very complex legend. In these cases, it makes sense to generate an HTML legend. Charts provide a generateLegend() method on their prototype that returns an HTML string for the legend.

To configure how this legend is generated, you can change the legendCallback config property.

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 legendCallback: function(chart) {  
 // Return the HTML string here.  
 }  
 }  
});

Note that legendCallback is not called automatically and you must call generateLegend() yourself in code when creating a legend using this method.

results matching ""

No results matching ""