* [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)

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

Tooltips

## Tooltip Configuration

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

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Default | Description |
| enabled | Boolean | true | Are on-canvas tooltips enabled |
| custom | Function | null | See [custom tooltip](#gjdgxs) section. |
| mode | String | 'nearest' | Sets which elements appear in the tooltip. [more...](http://docs.google.com/general/interactions/modes.html#interaction-modes). |
| intersect | Boolean | true | if true, the tooltip mode applies only when the mouse position intersects with an element. If false, the mode will be applied at all times. |
| position | String | 'average' | The mode for positioning the tooltip. [more...](#30j0zll) |
| callbacks | Object |  | See the [callbacks section](#1fob9te) |
| itemSort | Function |  | Sort tooltip items. [more...](#3znysh7) |
| filter | Function |  | Filter tooltip items. [more...](#2et92p0) |
| backgroundColor | Color | 'rgba(0,0,0,0.8)' | Background color of the tooltip. |
| titleFontFamily | String | "'Helvetica Neue', 'Helvetica', 'Arial', sans-serif" | title font |
| titleFontSize | Number | 12 | Title font size |
| titleFontStyle | String | 'bold' | Title font style |
| titleFontColor | Color | '#fff' | Title font color |
| titleSpacing | Number | 2 | Spacing to add to top and bottom of each title line. |
| titleMarginBottom | Number | 6 | Margin to add on bottom of title section. |
| bodyFontFamily | String | "'Helvetica Neue', 'Helvetica', 'Arial', sans-serif" | body line font |
| bodyFontSize | Number | 12 | Body font size |
| bodyFontStyle | String | 'normal' | Body font style |
| bodyFontColor | Color | '#fff' | Body font color |
| bodySpacing | Number | 2 | Spacing to add to top and bottom of each tooltip item. |
| footerFontFamily | String | "'Helvetica Neue', 'Helvetica', 'Arial', sans-serif" | footer font |
| footerFontSize | Number | 12 | Footer font size |
| footerFontStyle | String | 'bold' | Footer font style |
| footerFontColor | Color | '#fff' | Footer font color |
| footerSpacing | Number | 2 | Spacing to add to top and bottom of each footer line. |
| footerMarginTop | Number | 6 | Margin to add before drawing the footer. |
| xPadding | Number | 6 | Padding to add on left and right of tooltip. |
| yPadding | Number | 6 | Padding to add on top and bottom of tooltip. |
| caretPadding | Number | 2 | Extra distance to move the end of the tooltip arrow away from the tooltip point. |
| caretSize | Number | 5 | Size, in px, of the tooltip arrow. |
| cornerRadius | Number | 6 | Radius of tooltip corner curves. |
| multiKeyBackground | Color | '#fff' | Color to draw behind the colored boxes when multiple items are in the tooltip |
| displayColors | Boolean | true | if true, color boxes are shown in the tooltip |
| borderColor | Color | 'rgba(0,0,0,0)' | Color of the border |
| borderWidth | Number | 0 | Size of the border |

### Position Modes

Possible modes are:

* 'average'
* 'nearest'

'average' mode will place the tooltip at the average position of the items displayed in the tooltip. 'nearest' will place the tooltip at the position of the element closest to the event position.

New modes can be defined by adding functions to the Chart.Tooltip.positioners map.

Example:

/\*\*  
 \* Custom positioner  
 \* @function Chart.Tooltip.positioners.custom  
 \* @param elements {Chart.Element[]} the tooltip elements  
 \* @param eventPosition {Point} the position of the event in canvas coordinates  
 \* @returns {Point} the tooltip position  
 \*/  
Chart.Tooltip.positioners.custom = function(elements, eventPosition) {  
 /\*\* @type {Chart.Tooltip} \*/  
 var tooltip = this;  
  
 /\* ... \*/  
  
 return {  
 x: 0,  
 y: 0  
 };  
}

### Sort Callback

Allows sorting of [tooltip items](#tyjcwt). Must implement at minimum a function that can be passed to [Array.prototype.sort](https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/Array/sort). This function can also accept a third parameter that is the data object passed to the chart.

### Filter Callback

Allows filtering of [tooltip items](#tyjcwt). Must implement at minimum a function that can be passed to [Array.prototype.filter](https://developer.mozilla.org/en/docs/Web/JavaScript/Reference/Global_Objects/Array/filter). This function can also accept a second parameter that is the data object passed to the chart.

## Tooltip Callbacks

The tooltip label configuration is nested below the tooltip configuration using the callbacks key. The tooltip has the following callbacks for providing text. For all functions, 'this' will be the tooltip object created from the Chart.Tooltip constructor.

All functions are called with the same arguments: a [tooltip item](#tyjcwt) and the data object passed to the chart. All functions must return either a string or an array of strings. Arrays of strings are treated as multiple lines of text.

|  |  |  |
| --- | --- | --- |
| Name | Arguments | Description |
| beforeTitle | Array[tooltipItem], data | Returns the text to render before the title. |
| title | Array[tooltipItem], data | Returns text to render as the title of the tooltip. |
| afterTitle | Array[tooltipItem], data | Returns text to render after the title. |
| beforeBody | Array[tooltipItem], data | Returns text to render before the body section. |
| beforeLabel | tooltipItem, data | Returns text to render before an individual label. This will be called for each item in the tooltip. |
| label | tooltipItem, data | Returns text to render for an individual item in the tooltip. |
| labelColor | tooltipItem, chart | Returns the colors to render for the tooltip item. [more...](#3dy6vkm) |
| labelTextColor | tooltipItem, chart | Returns the colors for the text of the label for the tooltip item. |
| afterLabel | tooltipItem, data | Returns text to render after an individual label. |
| afterBody | Array[tooltipItem], data | Returns text to render after the body section |
| beforeFooter | Array[tooltipItem], data | Returns text to render before the footer section. |
| footer | Array[tooltipItem], data | Returns text to render as the footer of the tooltip. |
| afterFooter | Array[tooltipItem], data | Text to render after the footer section |

### Label Callback

The label callback can change the text that displays for a given data point. A common example to round data values; the following example rounds the data to two decimal places.

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 tooltips: {  
 callbacks: {  
 label: function(tooltipItem, data) {  
 var label = data.datasets[tooltipItem.datasetIndex].label || '';  
  
 if (label) {  
 label += ': ';  
 }  
 label += Math.round(tooltipItem.yLabel \* 100) / 100;  
 return label;  
 }  
 }  
 }  
 }  
});

### Label Color Callback

For example, to return a red box for each item in the tooltip you could do:

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 tooltips: {  
 callbacks: {  
 labelColor: function(tooltipItem, chart) {  
 return {  
 borderColor: 'rgb(255, 0, 0)',  
 backgroundColor: 'rgb(255, 0, 0)'  
 }  
 },  
 labelTextColor:function(tooltipItem, chart){  
 return '#543453';  
 }  
 }  
 }  
 }  
});

### Tooltip Item Interface

The tooltip items passed to the tooltip callbacks implement the following interface.

{  
 // X Value of the tooltip as a string  
 xLabel: String,  
  
 // Y value of the tooltip as a string  
 yLabel: String,  
  
 // Index of the dataset the item comes from  
 datasetIndex: Number,  
  
 // Index of this data item in the dataset  
 index: Number,  
  
 // X position of matching point  
 x: Number,  
  
 // Y position of matching point  
 y: Number,  
}

## External (Custom) Tooltips

Custom tooltips allow you to hook into the tooltip rendering process so that you can render the tooltip in your own custom way. Generally this is used to create an HTML tooltip instead of an oncanvas one. You can enable custom tooltips in the global or chart configuration like so:

var myPieChart = new Chart(ctx, {  
 type: 'pie',  
 data: data,  
 options: {  
 tooltips: {  
 // Disable the on-canvas tooltip  
 enabled: false,  
  
 custom: function(tooltipModel) {  
 // Tooltip Element  
 var tooltipEl = document.getElementById('chartjs-tooltip');  
  
 // Create element on first render  
 if (!tooltipEl) {  
 tooltipEl = document.createElement('div');  
 tooltipEl.id = 'chartjs-tooltip';  
 tooltipEl.innerHTML = "<table></table>";  
 document.body.appendChild(tooltipEl);  
 }  
  
 // Hide if no tooltip  
 if (tooltipModel.opacity === 0) {  
 tooltipEl.style.opacity = 0;  
 return;  
 }  
  
 // Set caret Position  
 tooltipEl.classList.remove('above', 'below', 'no-transform');  
 if (tooltipModel.yAlign) {  
 tooltipEl.classList.add(tooltipModel.yAlign);  
 } else {  
 tooltipEl.classList.add('no-transform');  
 }  
  
 function getBody(bodyItem) {  
 return bodyItem.lines;  
 }  
  
 // Set Text  
 if (tooltipModel.body) {  
 var titleLines = tooltipModel.title || [];  
 var bodyLines = tooltipModel.body.map(getBody);  
  
 var innerHtml = '<thead>';  
  
 titleLines.forEach(function(title) {  
 innerHtml += '<tr><th>' + title + '</th></tr>';  
 });  
 innerHtml += '</thead><tbody>';  
  
 bodyLines.forEach(function(body, i) {  
 var colors = tooltipModel.labelColors[i];  
 var style = 'background:' + colors.backgroundColor;  
 style += '; border-color:' + colors.borderColor;  
 style += '; border-width: 2px';  
 var span = '<span style="' + style + '"></span>';  
 innerHtml += '<tr><td>' + span + body + '</td></tr>';  
 });  
 innerHtml += '</tbody>';  
  
 var tableRoot = tooltipEl.querySelector('table');  
 tableRoot.innerHTML = innerHtml;  
 }  
  
 // `this` will be the overall tooltip  
 var position = this.\_chart.canvas.getBoundingClientRect();  
  
 // Display, position, and set styles for font  
 tooltipEl.style.opacity = 1;  
 tooltipEl.style.position = 'absolute';  
 tooltipEl.style.left = position.left + tooltipModel.caretX + 'px';  
 tooltipEl.style.top = position.top + tooltipModel.caretY + 'px';  
 tooltipEl.style.fontFamily = tooltipModel.\_bodyFontFamily;  
 tooltipEl.style.fontSize = tooltipModel.bodyFontSize + 'px';  
 tooltipEl.style.fontStyle = tooltipModel.\_bodyFontStyle;  
 tooltipEl.style.padding = tooltipModel.yPadding + 'px ' + tooltipModel.xPadding + 'px';  
 }  
 }  
 }  
});

See [samples](http://www.chartjs.org/samples/) for examples on how to get started with custom tooltips.

## Tooltip Model

The tooltip model contains parameters that can be used to render the tooltip.

{  
 // The items that we are rendering in the tooltip. See Tooltip Item Interface section  
 dataPoints: TooltipItem[],  
  
 // Positioning  
 xPadding: Number,  
 yPadding: Number,  
 xAlign: String,  
 yAlign: String,  
  
 // X and Y properties are the top left of the tooltip  
 x: Number,  
 y: Number,  
 width: Number,  
 height: Number,  
 // Where the tooltip points to  
 caretX: Number,  
 caretY: Number,  
  
 // Body  
 // The body lines that need to be rendered  
 // Each object contains 3 parameters  
 // before: String[] // lines of text before the line with the color square  
 // lines: String[], // lines of text to render as the main item with color square  
 // after: String[], // lines of text to render after the main lines  
 body: Object[],  
 // lines of text that appear after the title but before the body  
 beforeBody: String[],  
 // line of text that appear after the body and before the footer  
 afterBody: String[],  
 bodyFontColor: Color,  
 \_bodyFontFamily: String,  
 \_bodyFontStyle: String,  
 \_bodyAlign: String,  
 bodyFontSize: Number,  
 bodySpacing: Number,  
  
 // Title  
 // lines of text that form the title  
 title: String[],  
 titleFontColor: Color,  
 \_titleFontFamily: String,  
 \_titleFontStyle: String,  
 titleFontSize: Number,  
 \_titleAlign: String,  
 titleSpacing: Number,  
 titleMarginBottom: Number,  
  
 // Footer  
 // lines of text that form the footer  
 footer: String[],  
 footerFontColor: Color,  
 \_footerFontFamily: String,  
 \_footerFontStyle: String,  
 footerFontSize: Number,  
 \_footerAlign: String,  
 footerSpacing: Number,  
 footerMarginTop: Number,  
  
 // Appearance  
 caretSize: Number,  
 cornerRadius: Number,  
 backgroundColor: Color,  
  
 // colors to render for each item in body[]. This is the color of the squares in the tooltip  
 labelColors: Color[],  
  
 // 0 opacity is a hidden tooltip  
 opacity: Number,  
 legendColorBackground: Color,  
 displayColors: Boolean,  
}

results matching ""

No results matching ""