* [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/configuration/)
  + [Animations](http://docs.google.com/configuration/animations.html)
  + [Layout](http://docs.google.com/configuration/layout.html)
  + [Legend](http://docs.google.com/configuration/legend.html)
  + [Title](http://docs.google.com/configuration/title.html)
  + [Tooltip](http://docs.google.com/configuration/tooltip.html)
  + [Elements](http://docs.google.com/configuration/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/)
  + [Cartesian](http://docs.google.com/)
    - [Category](http://docs.google.com/category.html)
    - [Linear](http://docs.google.com/linear.html)
    - [Logarithmic](http://docs.google.com/logarithmic.html)
    - [Time](http://docs.google.com/time.html)
  + [Radial](http://docs.google.com/radial/)
    - [Linear](http://docs.google.com/radial/linear.html)
  + [Labelling](http://docs.google.com/labelling.html)
  + [Styling](http://docs.google.com/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)

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

Time Cartesian Axis

The time scale is used to display times and dates. When building its ticks, it will automatically calculate the most comfortable unit base on the size of the scale.

## Data Sets

### Input Data

The x-axis data points may additionally be specified via the t attribute when using the time scale.

data: [{  
 x: new Date(),  
 y: 1  
}, {  
 t: new Date(),  
 y: 10  
}]

### Date Formats

When providing data for the time scale, Chart.js supports all of the formats that Moment.js accepts. See [Moment.js docs](http://momentjs.com/docs/#/parsing/) for details.

## Configuration Options

The following options are provided by the time scale. You may also set options provided by the [common tick configuration](http://docs.google.com/#tick-configuration).

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Type | Default | Description |
| distribution | String | linear | How data is plotted. [more...](#gjdgxs) |
| bounds | String | data | Determines the scale bounds. [more...](#30j0zll) |
| ticks.source | String | auto | How ticks are generated. [more...](#1fob9te) |
| time.displayFormats | Object |  | Sets how different time units are displayed. [more...](#3znysh7) |
| time.isoWeekday | Boolean | false | If true and the unit is set to 'week', then the first day of the week will be Monday. Otherwise, it will be Sunday. |
| time.max | [Time](#2et92p0) |  | If defined, this will override the data maximum |
| time.min | [Time](#2et92p0) |  | If defined, this will override the data minimum |
| time.parser | String/Function |  | Custom parser for dates. [more...](#tyjcwt) |
| time.round | String | false | If defined, dates will be rounded to the start of this unit. See [Time Units](#3dy6vkm) below for the allowed units. |
| time.tooltipFormat | String |  | The moment js format string to use for the tooltip. |
| time.unit | String | false | If defined, will force the unit to be a certain type. See [Time Units](#3dy6vkm) section below for details. |
| time.stepSize | Number | 1 | The number of units between grid lines. |
| time.minUnit | String | 'millisecond' | The minimum display format to be used for a time unit. |

### Time Units

The following time measurements are supported. The names can be passed as strings to the time.unit config option to force a certain unit.

* millisecond
* second
* minute
* hour
* day
* week
* month
* quarter
* year

For example, to create a chart with a time scale that always displayed units per month, the following config could be used.

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 scales: {  
 xAxes: [{  
 time: {  
 unit: 'month'  
 }  
 }]  
 }  
 }  
})

### Display Formats

The following display formats are used to configure how different time units are formed into strings for the axis tick marks. See [moment.js](http://momentjs.com/docs/#/displaying/format/) for the allowable format strings.

|  |  |  |
| --- | --- | --- |
| Name | Default | Example |
| millisecond | 'h:mm:ss.SSS a' | 11:20:01.123 AM |
| second | 'h:mm:ss a' | 11:20:01 AM |
| minute | 'h:mm a' | 11:20 AM |
| hour | 'hA' | 11AM |
| day | 'MMM D' | Sep 4 |
| week | 'll' | Sep 4 2015 |
| month | 'MMM YYYY' | Sep 2015 |
| quarter | '[Q]Q - YYYY' | Q3 - 2015 |
| year | 'YYYY' | 2015 |

For example, to set the display format for the 'quarter' unit to show the month and year, the following config would be passed to the chart constructor.

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 scales: {  
 xAxes: [{  
 type: 'time',  
 time: {  
 displayFormats: {  
 quarter: 'MMM YYYY'  
 }  
 }  
 }]  
 }  
 }  
})

### Scale Distribution

The distribution property controls the data distribution along the scale:

* 'linear': data are spread according to their time (distances can vary)
* 'series': data are spread at the same distance from each other

var chart = new Chart(ctx, {  
 type: 'line',  
 data: data,  
 options: {  
 scales: {  
 xAxes: [{  
 type: 'time',  
 distribution: 'series'  
 }]  
 }  
 }  
})

### Scale Bounds

The bounds property controls the scale boundary strategy (bypassed by min/max time options)

* 'data': make sure data are fully visible, labels outside are removed
* 'ticks': make sure ticks are fully visible, data outside are truncated

### Ticks Source

The ticks.source property controls the ticks generation

* 'auto': generates "optimal" ticks based on scale size and time options.
* 'data': generates ticks from data (including labels from data {t|x|y} objects)
* 'labels': generates ticks from user given data.labels values ONLY

### Parser

If this property is defined as a string, it is interpreted as a custom format to be used by moment to parse the date.

If this is a function, it must return a moment.js object given the appropriate data value.

results matching ""

No results matching ""