Timelines

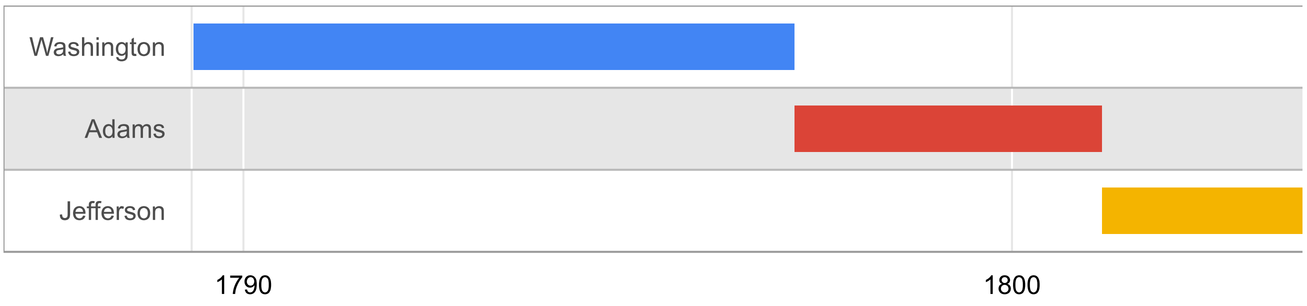
Overview

A *timeline* is a chart that depicts how a set of resources are used over time. If you're managing a software project and want to illustrate who is doing what and when, or if you're organizing a conference and need to schedule meeting rooms, a timeline is often a reasonable visualization choice. One popular type of timeline is the *Gantt chart*.

Note: In JavaScript Date objects, months are indexed starting at zero and go up through eleven, with January being month 0 and December being month 11. If your timeline seems off by a month, this is most likely why. For more information, see the Dates and Times (https://google-developers.appspot.com/chart/interactive/docs/datesandtimes) page.

A simple example

Let's say you want to plot when American presidents served their terms. Here, the "resources" are the presidents, and we can plot each president's term as a bar:



Hovering over a bar brings up a tooltip with more detailed information.

CODE IT YOURSELF ON JSFIDDLE

After loading the `timeline` package and defining a callback to draw the chart when the page is rendered, the `drawChart()` method instantiates a `google.visualization.Timeline()` and then fills a `dataTable` with one row for each president.

Inside the `dataTable`, the first column is the president's name, and the second and third columns are the start and end times. These have the JavaScript `Date` type, but they could also be plain numbers.

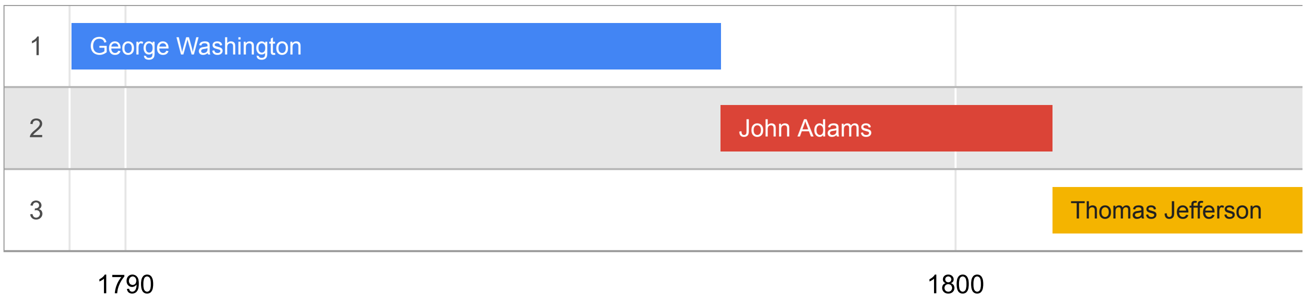
Finally, we invoke the `draw()` method of the chart, which displays it inside a `div` with the same identifier (`timeline`) used when `container` was declared in the first line of `drawChart()`.

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['timeline']});
      google.charts.setOnLoadCallback(drawChart);
      function drawChart() {
        var container = document.getElementById('timeline');
        var chart = new google.visualization.Timeline(container);
        var dataTable = new google.visualization.DataTable();

        dataTable.addColumn({ type: 'string', id: 'President' });
        dataTable.addColumn({ type: 'date', id: 'Start' });
        dataTable.addColumn({ type: 'date', id: 'End' });
        dataTable.addRows([
          [ 'Washington', new Date(1789, 3, 30), new Date(1797, 2, 4) ],
          [ 'Adams',      new Date(1797, 2, 4),  new Date(1801, 2, 4) ],
          [ 'Jefferson',  new Date(1801, 2, 4),  new Date(1809, 2, 4) ]]);

        chart.draw(dataTable);
      }
    </script>
  </head>
  <body>
    <div id="timeline" style="height: 180px;"></div>
  </body>
</html>
```

Google Charts timelines are customizable, and in the following examples we'll show you some common ways to tailor the appearance of your timelines.

Labeling the bars

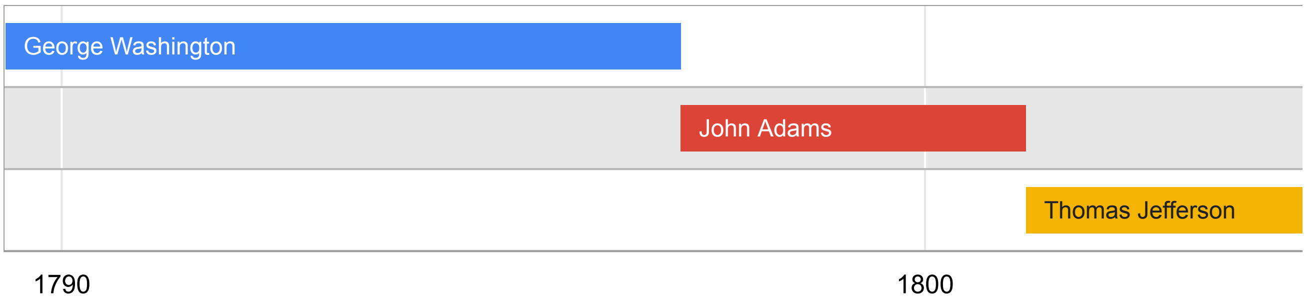
In addition to the row labels ("Washington", "Adams", "Jefferson" above) you can label individual bars. Here, we change the row labels to simple numbers and place each president's name on his bar.



In this code, we've inserted a new column into our data to hold the bar labels: the full name of each president. When there are four columns in a timeline `dataTable`, the first is interpreted as the row label, the second as the bar label, and the third and fourth as start and end.

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example2.1');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();

    dataTable.addColumn({ type: 'string', id: 'Term' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });

    dataTable.addRows([
      [ '1', 'George Washington', new Date(1789, 3, 30), new Date(1797, 2, 4)
      [ '2', 'John Adams',        new Date(1797, 2, 4),  new Date(1801, 2, 4)
      [ '3', 'Thomas Jefferson',  new Date(1801, 2, 4),  new Date(1809, 2, 4)

    chart.draw(dataTable);
  }
</script>

<div id="example2.1" style="height: 200px;"></div>
```

Our new row labels above aren't very informative, so let's remove them with the timeline `showRowLabels` option.



By default, `showRowLabels` is `true`. Setting it to `false` removes the row labels:

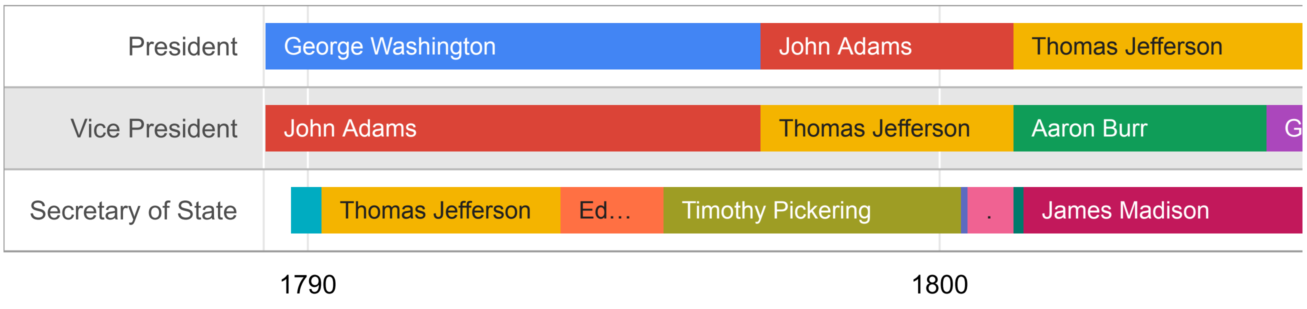
```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example2.2');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Term' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ '1', 'George Washington', new Date(1789, 3, 30), new Date(1797, 2, 4)
      [ '2', 'John Adams',        new Date(1797, 2, 4),  new Date(1801, 2, 4)
      [ '3', 'Thomas Jefferson',  new Date(1801, 2, 4),  new Date(1809, 2, 4)

    var options = {
      timeline: { showRowLabels: false }
    };

    chart.draw(dataTable, options);
  }
</script>

<div id="example2.2" style="height: 180px;"></div>
```

An advanced example

To make our timeline more complex, let's add vice presidents and secretaries of state to our chart. John Adams was vice president before he became president, and Thomas Jefferson was secretary of state, then vice president, and finally president.

In timelines, a resource will have the same color even when it appears on multiple rows, so in the following chart each person is represented with a consistent color.

Some secretaries of state served for very short times, so this chart is a good test of labeling. When a label is too big for the bar, it's abbreviated or eliminated, depending on the bar size. Users can always hover over the bar to get tooltip information.



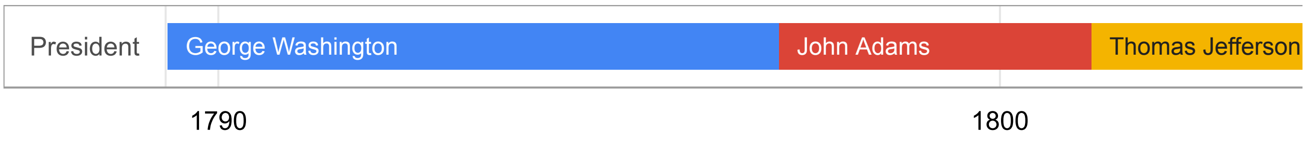
The timeline will lay out the rows in order—president on top of vice president on top of secretary of state—because that's the order that they appear in the code below. However, the layout of the bars is determined solely by the start and end times, so swapping two secretaries of state or two presidents in the `dataTable` has no effect on the chart.

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js":

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {

    var container = document.getElementById('example3.1');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Position' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'President', 'George Washington', new Date(1789, 3, 30), new Date(179
      [ 'President', 'John Adams', new Date(1797, 2, 4), new Date(1801, 2, 4)
      [ 'President', 'Thomas Jefferson', new Date(1801, 2, 4), new Date(1809,
      [ 'Vice President', 'John Adams', new Date(1789, 3, 21), new Date(1797,
      [ 'Vice President', 'Thomas Jefferson', new Date(1797, 2, 4), new Date(
      [ 'Vice President', 'Aaron Burr', new Date(1801, 2, 4), new Date(1805,
      [ 'Vice President', 'George Clinton', new Date(1805, 2, 4), new Date(18
      [ 'Secretary of State', 'John Jay', new Date(1789, 8, 25), new Date(179
      [ 'Secretary of State', 'Thomas Jefferson', new Date(1790, 2, 22), new
      [ 'Secretary of State', 'Edmund Randolph', new Date(1794, 0, 2), new Da
      [ 'Secretary of State', 'Timothy Pickering', new Date(1795, 7, 20), new
      [ 'Secretary of State', 'Charles Lee', new Date(1800, 4, 13), new Date(
      [ 'Secretary of State', 'John Marshall', new Date(1800, 5, 13), new Dat
      [ 'Secretary of State', 'Levi Lincoln', new Date(1801, 2, 5), new Date(
      [ 'Secretary of State', 'James Madison', new Date(1801, 4, 2), new Date
    ]);

    chart.draw(dataTable);
  }
</script>

<div id="example3.1" style="height: 200px;"></div>
```

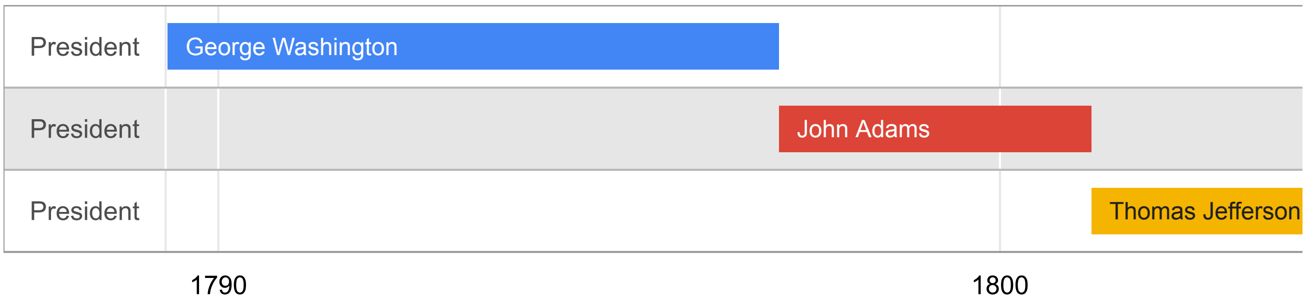
Putting bars on one row

Unlike popes (http://en.wikipedia.org/wiki/Western_Schism), there can only be one U.S. president at a time, so if we label all of our rows as "President", the timeline will combine the three rows of our first chart into one row for a cleaner presentation. You can control this behavior with the `groupByRowLabel` option.

Here's the default behavior:



Now let's set `groupByRowLabel` to `false` and split the one row into three:



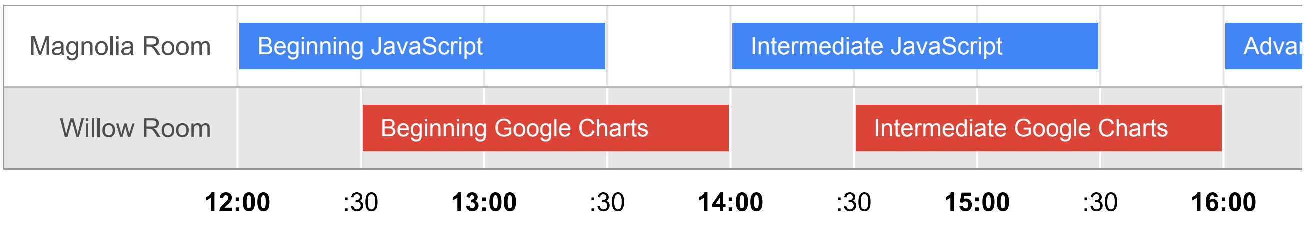
The code to turn off grouping:

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example4.2');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();

    dataTable.addColumn({ type: 'string', id: 'Role' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'President', 'George Washington', new Date(1789, 3, 30), new Date(179
      [ 'President', 'John Adams', new Date(1797, 2, 4), new Date(1801, 2, 4)
      [ 'President', 'Thomas Jefferson', new Date(1801, 2, 4), new Date(1809,

    var options = {
      timeline: { groupByRowLabel: false }
    };

    chart.draw(dataTable, options);
  }
</script>

<div id="example4.2" style="height: 200px;"></div>
```

Controlling the colors

By default, Google Charts chooses colors optimized for aesthetics and readability (including users with visual disabilities). You can tailor the default behavior with the `colorByRowLabel`, `singleColor`, `backgroundColor` and `colors` options.

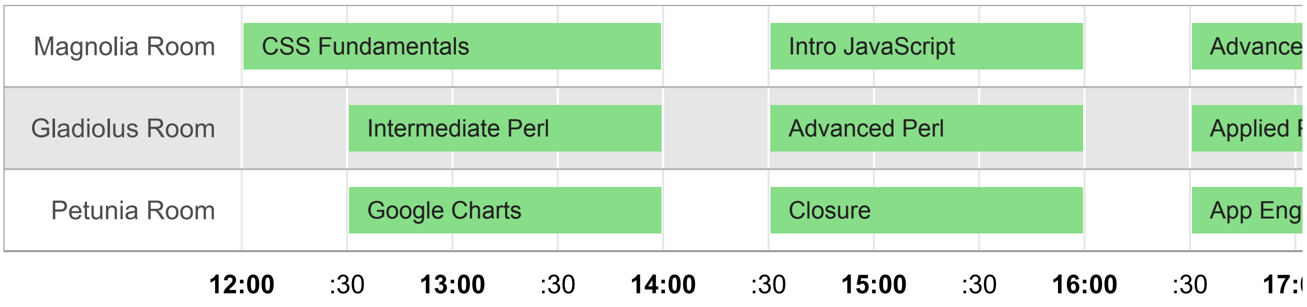
The `colorByRowLabel` option colors all bars on the same row the same. This can be a good choice when there are gaps between your bars.



`colorByRowLabel` defaults to `false`, so here we override that and set it to `true`.

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js":

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {

    var container = document.getElementById('example5.1');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Room' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'Magnolia Room', 'Beginning JavaScript',       new Date(0,0,0,12,0,0)
      [ 'Magnolia Room', 'Intermediate JavaScript',    new Date(0,0,0,14,0,0)
      [ 'Magnolia Room', 'Advanced JavaScript',        new Date(0,0,0,16,0,0)
      [ 'Willow Room',   'Beginning Google Charts',    new Date(0,0,0,12,30,0
      [ 'Willow Room',   'Intermediate Google Charts', new Date(0,0,0,14,30,0
      [ 'Willow Room',   'Advanced Google Charts',     new Date(0,0,0,16,30,0

    var options = {
      timeline: { colorByRowLabel: true }
    };

    chart.draw(dataTable, options);
  }

</script>

<div id="example5.1" style="height: 100px;"></div>
```

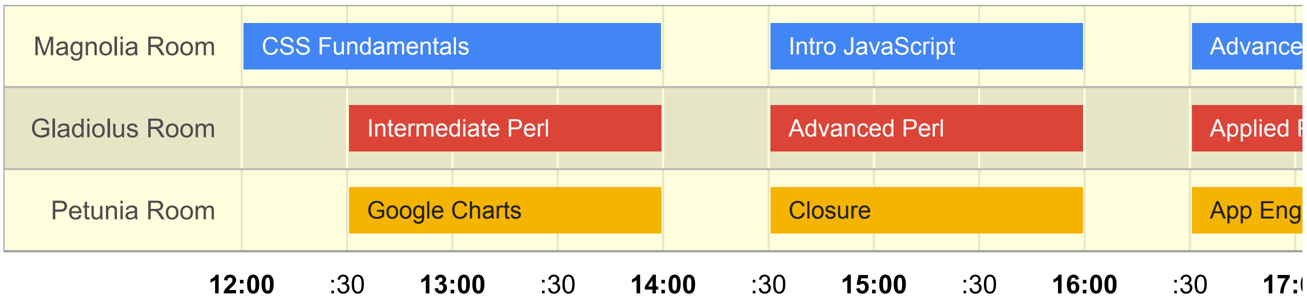
If you want all bars the same color regardless of what row they're on, use the `singleColor` option:



In the code below, **`singleColor`** is set to a hex value to color all the bars light green:

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js">

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {

    var container = document.getElementById('example5.2');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();

    dataTable.addColumn({ type: 'string', id: 'Room' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'Magnolia Room',  'CSS Fundamentals',    new Date(0,0,0,12,0,0),  new
      [ 'Magnolia Room',  'Intro JavaScript',    new Date(0,0,0,14,30,0), new
      [ 'Magnolia Room',  'Advanced JavaScript', new Date(0,0,0,16,30,0), new
      [ 'Gladiolus Room', 'Intermediate Perl',   new Date(0,0,0,12,30,0), new
      [ 'Gladiolus Room', 'Advanced Perl',       new Date(0,0,0,14,30,0), new
      [ 'Gladiolus Room', 'Applied Perl',        new Date(0,0,0,16,30,0), new
      [ 'Petunia Room',   'Google Charts',       new Date(0,0,0,12,30,0), new
      [ 'Petunia Room',   'Closure',             new Date(0,0,0,14,30,0), new
      [ 'Petunia Room',   'App Engine',          new Date(0,0,0,16,30,0), new

    var options = {
      timeline: { singleColor: '#8d8' },
    };

    chart.draw(dataTable, options);
  }
</script>

<div id="example5.2" style="height: 150px;"></div>
```

You can control the background color of the rows with the `backgroundColor` option:

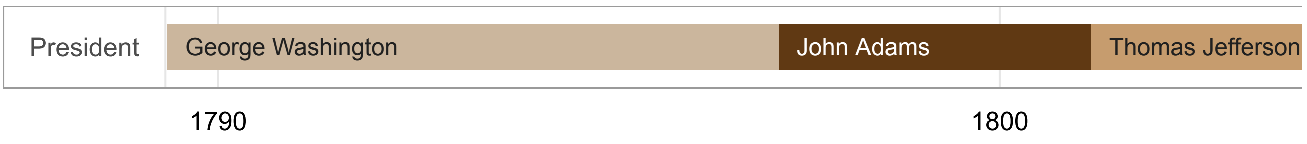


Background colors are darkened on alternating rows for legibility.

The `backgroundColor` is specified as a hex value. Here, we pair it with `colorByRowLabel` to show tracks in a conference:

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js":

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example5.3');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Room' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'Magnolia Room',  'CSS Fundamentals',    new Date(0,0,0,12,0,0),  new
      [ 'Magnolia Room',  'Intro JavaScript',    new Date(0,0,0,14,30,0), new
      [ 'Magnolia Room',  'Advanced JavaScript', new Date(0,0,0,16,30,0), new
      [ 'Gladiolus Room', 'Intermediate Perl',   new Date(0,0,0,12,30,0), new
      [ 'Gladiolus Room', 'Advanced Perl',       new Date(0,0,0,14,30,0), new
      [ 'Gladiolus Room', 'Applied Perl',        new Date(0,0,0,16,30,0), new
      [ 'Petunia Room',   'Google Charts',       new Date(0,0,0,12,30,0), new
      [ 'Petunia Room',   'Closure',             new Date(0,0,0,14,30,0), new
      [ 'Petunia Room',   'App Engine',          new Date(0,0,0,16,30,0), new

    var options = {
      timeline: { colorByRowLabel: true },
      backgroundColor: '#ffd'
    };
    chart.draw(dataTable, options);
  }
</script>

<div id="example5.3" style="height: 150px;"></div>
```

If you want to control the colors of individual bars, use the `colors` option:



`colors` takes an array of hex values, which are applied to the bars in order:

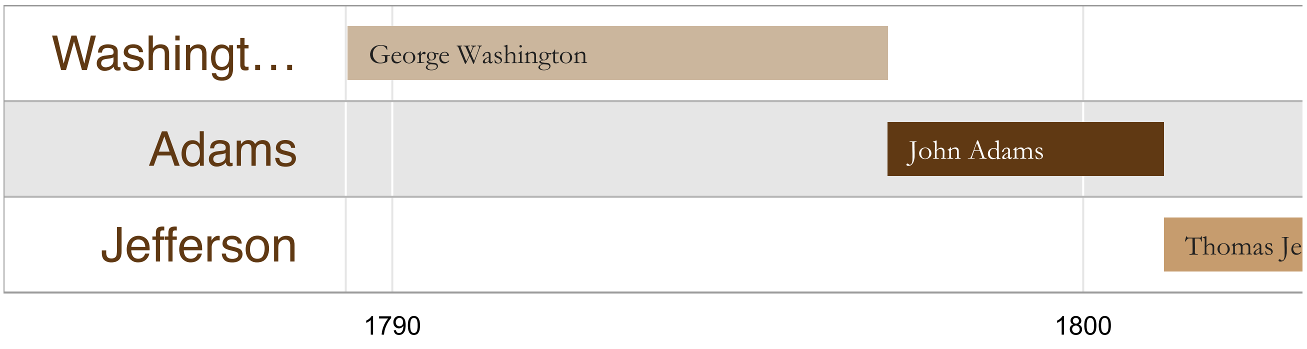
```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example5.4');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Role' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'President', 'George Washington', new Date(1789, 3, 30), new Date(179
      [ 'President', 'John Adams', new Date(1797, 2, 4), new Date(1801, 2, 4)
      [ 'President', 'Thomas Jefferson', new Date(1801, 2, 4), new Date(1809,

    var options = {
      colors: ['#cbb69d', '#603913', '#c69c6e'],
    };

    chart.draw(dataTable, options);
  }

</script>
```

```
<div id="example5.4" style="height: 150px;"></div>
```

If your chart requires more colors than listed, the chart will behave as though `singleColor` is set to the first color in the list. (This is true for all Google Charts, not just timelines.)

Changing the fonts

You can choose the typeface and font for the labels of each row with `rowLabelStyle`, and for the labels on each bar with `barLabelStyle`. Both are demonstrated below.



Note: Be sure to choose typefaces that your users' browsers will be able to render.

```
<script type="text/javascript" src="https://www.gstatic.com/charts/loader.js":

<script type="text/javascript">
  google.charts.load("current", {packages:["timeline"]});
  google.charts.setOnLoadCallback(drawChart);
  function drawChart() {
    var container = document.getElementById('example6.1');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Role' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'Washington', 'George Washington', new Date(1789, 3, 30), new Date(17
      [ 'Adams', 'John Adams', new Date(1797, 2, 4), new Date(1801, 2, 4) ],
      [ 'Jefferson', 'Thomas Jefferson', new Date(1801, 2, 4), new Date(1809,

    var options = {
      colors: ['#cbb69d', '#603913', '#c69c6e'],
      timeline: { rowLabelStyle: {fontName: 'Helvetica', fontSize: 24, color:
                     barLabelStyle: { fontName: 'Garamond', fontSize: 14 } }
    };

    chart.draw(dataTable, options);
  }
</script>

<div id="example6.1" style="height: 200px;"></div>
```

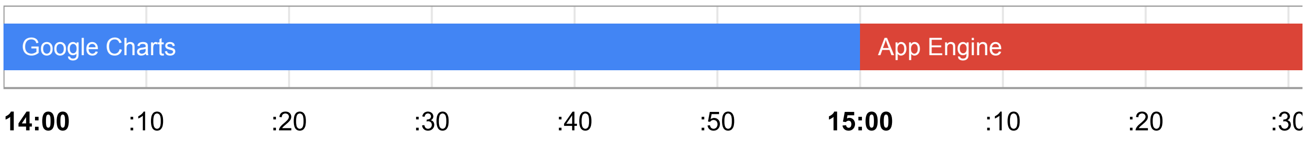
You can't set the color of `barLabel` text.

Overlapping gridlines

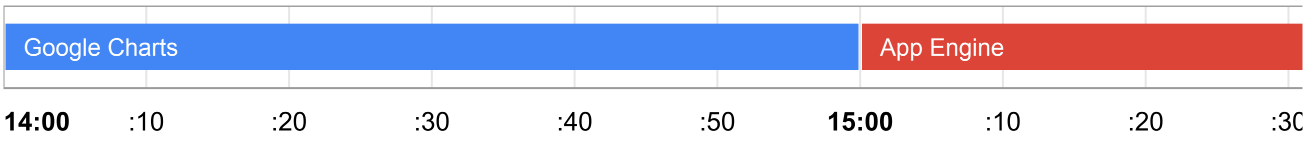
Google Charts makes tiny adjustments to bar endpoints to avoid obscuring timeline gridlines. To prevent this behavior, set the `avoidOverlappingGridLines` option to `false`.

To illustrate the effect, here are two examples, the first with an overlapped gridline and the second without.

First, with overlapping gridlines:



Now, without overlapping gridlines (the default):



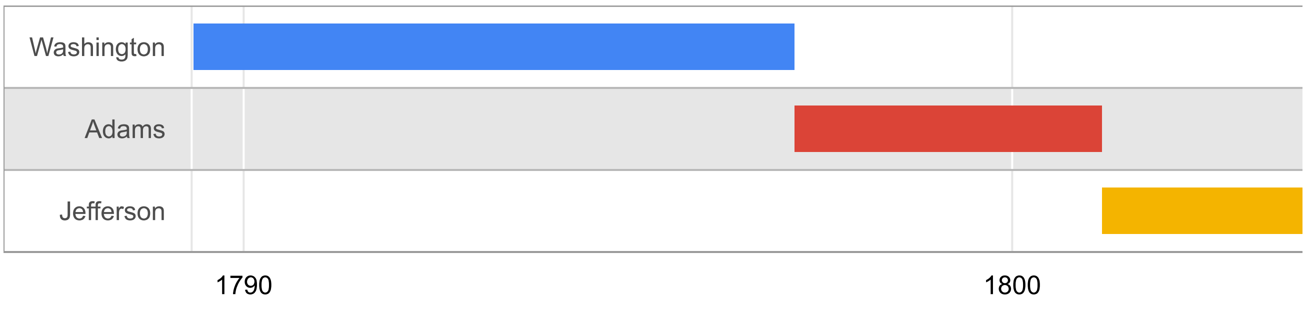
Here's code that overlaps gridlines:

```
  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.j

  <script type="text/javascript">
    google.charts.load("current", {packages:["timeline"]});
    google.charts.setOnLoadCallback(drawChart);
    function drawChart() {
```

```
    var container = document.getElementById('example7.1');
    var chart = new google.visualization.Timeline(container);
    var dataTable = new google.visualization.DataTable();
    dataTable.addColumn({ type: 'string', id: 'Room' });
    dataTable.addColumn({ type: 'string', id: 'Name' });
    dataTable.addColumn({ type: 'date', id: 'Start' });
    dataTable.addColumn({ type: 'date', id: 'End' });
    dataTable.addRows([
      [ 'Magnolia Room', 'Google Charts', new Date(0,0,0,14,0,0), new Date(
      [ 'Magnolia Room', 'App Engine',    new Date(0,0,0,15,0,0), new Date(

    var options = {
      timeline: { showRowLabels: false },
      avoidOverlappingGridLines: false
    };

    chart.draw(dataTable, options);
  }

</script>

<div id="example7.1" style="height: 200px;"></div>
```

Customizing tooltips

You can customize what users see when they hover over the bars of a timeline by adding a tooltip column into a five-column datatable. To provide non-default tooltips, every row of your datatable must have all five columns (row label, bar label, tooltip, start, and end):



Hovering over a bar brings up a tooltip with the text defined in the third column. In this chart, we have to set the second column to dummy values (here, `null`) so that our tooltips can exist in the third column.

CODE IT YOURSELF ON JSFIDDLE

```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {'packages':['timeline']});
      google.charts.setOnLoadCallback(drawChart);

      function drawChart() {
        var container = document.getElementById('timeline-tooltip');
        var chart = new google.visualization.Timeline(container);
        var dataTable = new google.visualization.DataTable();

        dataTable.addColumn({ type: 'string', id: 'President' });
        dataTable.addColumn({ type: 'string', id: 'dummy bar label' });
        dataTable.addColumn({ type: 'string', role: 'tooltip' });
        dataTable.addColumn({ type: 'date', id: 'Start' });
        dataTable.addColumn({ type: 'date', id: 'End' });
        dataTable.addRows([
          [ 'Washington', null, 'George', new Date(1789, 3, 29), new Date(179
          [ 'Adams', null, 'John', new Date(1797, 2, 3),  new Date(1801, 2, 3
          [ 'Jefferson', null, 'Thomas', new Date(1801, 2, 3),  new Date(1809

        chart.draw(dataTable);
      }
    </script>
  </head>
  <body>
    <div id="timeline-tooltip" style="height: 180px;"></div>
  </body>
</html>
```

Loading

The `google.charts.load` package name is `timeline`:

```
google.charts.load("current", {packages: ["timeline"]});
```

The visualization's class name is `google.visualization.Timeline`:

```
var visualization = new google.visualization.Timeline(container);
```

Data format

Rows: Each row in the table represents a timeline bar.

Columns:

	Column 0	Column 1	Column 2	Column 3
Purpose:	Row label	Bar label (optional)	Tooltip (optional)	Start
Data Type:	string	string	string	number or date
Role:	data	data	tooltip (https://google-developers.appspot.com/chart/interactive/docs/roles#tooltiprole)	data

Configuration options

Name	
avoidOverlappingGridLines	Whether display elements (e.g., the bars in a timeline) should obscure lines. If false, grid lines may be covered completely by display element true, display elements may be altered to keep grid lines visible. **Type:** boolean **Default:** true
backgroundColor	The background color for the main area of the chart. Can be either a si HTML color string, for example: `'red'` or `'#00cc00'`, or an object wi following properties. **Type:** string or object **Default:** 'white'
colors	The colors to use for the chart elements. An array of strings, where eac element is an HTML color string, for example: `colors: ['red','#004411']`. **Type:** Array of strings

	Default: default colors
enableInteractivity	Whether the chart throws user-based events or reacts to user interacti false, the chart will not throw 'select' or other interaction-based events throw ready or error events), and will not display hovertext or otherwise change depending on user input. **Type:** boolean **Default:** true
fontName	The default font face for all text in the chart. You can override this usin properties for specific chart elements. **Type:** string **Default:** 'Arial'
fontSize	The default font size, in pixels, of all text in the chart. You can override using properties for specific chart elements. **Type:** number **Default:** automatic
forceIFrame	Draws the chart inside an inline frame. (Note that on IE8, this option is ignored; all IE8 charts are drawn in i-frames.) **Type:** boolean **Default:** false
height	Height of the chart, in pixels. **Type:** number **Default:** height of the containing element
timeline.barLabelStyle	An object that specifies the bar label text style. It has this format: `{fontName: <string>, fontSize: <string>}` Also see `fontName` and `fontSize` in this table. **Type:** object **Default:** null
timeline.colorByRowLabel	If set to true, colors every bar on the row the same. The default is to us color per bar label. **Type:** boolean **Default:** false
timeline.groupByRowLabel	If set to false, creates one row for every `dataTable` entry. The default collect bars with the same row label into one row.

	Type: boolean **Default:** true
timeline.rowLabelStyle	An object that specifies the row label text style. It has this format: `{color: <string>, fontName: <string>, fontSize: <str` The **`color`** can be any HTML color string, for example **`'red'`** or **`'#00`** Also see **`fontName`** and **`fontSize`** in this table. **Type:** object **Default:** null
timeline.showBarLabels	If set to false, omits bar labels. The default is to show them. **Type:** boolean **Default:** true
timeline.showRowLabels	If set to false, omits row labels. The default is to show them. **Type:** boolean **Default:** true
timeline.singleColor	Colors all bars the same. Specified as a hex value (e.g., '#8d8'). **Type:** string **Default:** null
tooltip.isHtml	Set to **`false`** to use SVG-rendered (rather than HTML-rendered) tooltip Customizing Tooltip Content (https://google-developers.appspot.com/chart/interactive/docs/customizing_tooltip_ for more details. **Note:** customization of the HTML tooltip content via the tooltip column role (https://google-developers.appspot.com/chart/interactive/docs/roles#tooltiprole) is **not** supported by the Bubble Chart (https://google-developers.appspot.com/chart/interactive/docs/gallery/bubblechart) visualization. **Type:** boolean **Default:** true
tooltip.trigger	The user interaction that causes the tooltip to be displayed:

	• 'focus' - The tooltip will be displayed when the user hovers over the element. • 'none' - The tooltip will not be displayed. **Type:** string **Default:** 'focus'
width	Width of the chart, in pixels. **Type:** number **Default:** width of the containing element

Methods

Method	
`draw(data, options)`	Draws the chart. The chart accepts further method calls only after the `ready` (#Events)event is fired. `Extended description` (https://google-developers.appspot.com/chart/interactive/docs/reference#visdraw) . **Return Type:** none
`clearChart()`	Clears the chart, and releases all of its allocated resources. **Return Type:** none
`getSelection()`	Returns an array of the selected chart entities. Selectable entities are b legend entries and categories. For this chart, only one entity can be sel at any given moment. `Extended description` (https://google-developers.appspot.com/chart/interactive/docs/reference#visgetsele . **Return Type:** Array of selection elements

Events

Name	
`error`	Fired when an error occurs when attempting to render the chart.

	Properties: id, message
`onmouseover`	Fired when the user mouses over a visual entity. Passes back the row and column indices of the corresponding data table element. A bar correlates to a cell in the data table, a legend entry to a column (row index is null), and a category to a row (column index is null). **Properties:** row, column
`onmouseout`	Fired when the user mouses away from a visual entity. Passes back the row and column indices of the corresponding data table element. A bar correlates to a cell in the data table, a legend entry to a column (row index is null), and a category to a row (column index is null). **Properties:** row, column
`ready`	The chart is ready for external method calls. If you want to interact with the chart, and call methods after you draw it, you should set up a listener for this event *before* you call the `draw` method, and call them only after the event was fired. **Properties:** none
`select`	Fired when the user clicks a visual entity. To learn what has been selected, call `getSelection()` (#Methods). **Properties:** none

Data policy

All code and data are processed and rendered in the browser. No data is sent to any server.