100% native and cool looking animated JavaScript/CoffeeS

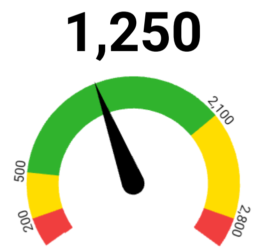
gauge

coffee

gauge.coffee   gauge.min.js

Example

Variant selection



Options:

Current Val:

Anim speed:

Angle:

Line width:

Radius:

Ptr length:

Ptr color: 000000

Ptr stroke:

Font size:

Color start: 6FADCF

Color stop: 8FC0DA

Background: E0E0E0

Ticks: □

□ **Share it!** If checked, the option values will be stored in the URL so that you can easily share your settings.

Features

- No images, no external CSS - pure canvas
- No dependencies (jQuery is supported, but not required)
- Highly configurable
- Resolution independent
- Animated gauge value changes (!)
- Works in all major browsers
- MIT License

Usage

```
var opts = {
  angle: -0.2, // The span of the gauge arc
  lineWidth: 0.2, // The line thickness
  radiusScale: 1, // Relative radius
  pointer: {
    length: 0.6, // // Relative to gauge radius
    strokeWidth: 0.056, // The thickness
    color: '#000000' // Fill color
  },
  limitMax: false,     // If false, max value increases automatically if value > maxValue
  limitMin: false,     // If true, the min value of the gauge will be fixed
  colorStart: '#6FADCF',   // Colors
  colorStop: '#8FC0DA',    // just experiment with them
  strokeColor: '#E0E0E0',  // to see which ones work best for you
  generateGradient: true,
  highDpiSupport: true,     // High resolution support

};
var target = document.getElementById('foo'); // your canvas element
var gauge = new Gauge(target).setOptions(opts); // create sexy gauge!
gauge.maxValue = 3000; // set max gauge value
gauge.setMinValue(0);  // Prefer setter over gauge.minValue = 0
gauge.animationSpeed = 32; // set animation speed (32 is default value)
gauge.set(1250); // set actual value
```

The `Gauge` class handles drawing on canvas and starts the animation.

Advanced options

- **Percentage color**
  If you want to control how Gauge behaves in relation to the displayed value you can use the Guage option called **percentColors**. To use it add following entry to the options:

  ```
  percentColors = [[0.0, "#a9d70b" ], [0.50, "#f9c802"], [1.0, "#ff0000"]];
  ```

  see working example at JSFiddle
- **Value labels**
  For displaying value labels, enable the staticLabels option. A label will be printed at the given value just outside the display arc.

  ```
  staticLabels: {
    font: "10px sans-serif",  // Specifies font
    labels: [100, 130, 150, 220.1, 260, 300],  // Print labels at these values
    color: "#000000",  // Optional: Label text color
    fractionDigits: 0  // Optional: Numerical precision. 0=round off.
  },
  ```

- **Static zones**
  When separating the background sectors or zones to have static colors, you must supply the staticZones property in the Gauge object's options.

  ```
  staticZones: [
     {strokeStyle: "#F03E3E", min: 100, max: 130}, // Red from 100 to 130
     {strokeStyle: "#FFDD00", min: 130, max: 150}, // Yellow
     {strokeStyle: "#30B32D", min: 150, max: 220}, // Green
     {strokeStyle: "#FFDD00", min: 220, max: 260}, // Yellow
     {strokeStyle: "#F03E3E", min: 260, max: 300}  // Red
  ],
  ```

  staticZones, percentColors and gradient are mutually exclusive. If staticZones is defined, it will take precedence.

  Note: Zones should always be defined within the gauge objects .minValue and .maxValue limits.
- Additionally, a **height** parameter may be passed in to increase the size of the zone (see example 4 gauge above).

  ```
    staticZones: [
      {strokeStyle: "rgb(255,0,0)", min: 0, max: 500, height: 1.4},
      {strokeStyle: "rgb(200,100,0)", min: 500, max: 1000, height: 1.2},
      {strokeStyle: "rgb(150,150,0)", min: 1000, max: 1500, height: 1},
      {strokeStyle: "rgb(100,200,0)", min: 1500, max: 2000, height: 0.8},
      {strokeStyle: "rgb(0,255,0)", min: 2000, max: 3100, height: 0.6}
    ],
  ```

  Note:

```
{strokeStyle: "rgb(80,80,80)", min: 2470, max: 2530, height: 1.3}
```

You can use this as an additional indicator (like in example 4) by making its color stand out, having a tall height and narrow range.

- **Tick marks**
  Now you may also add Ticks on two levels, major and minor (or divisions and sub divisions).

  renderTicks options:
  - **divisions** This is the number of major divisions around your arc.
  - **divWidth** This is to set the width of the indicator.
  - **divLength** This is a fractional percentage of the height of your arc line (0.5 = 50%)
  - **divColor** This sets the color of the division markers
  - **subDivisions** This sets the minor tick marks count between major ticks.
  - **subLength** This is a fractional percentage of the height of your arc line (0.5 = 50%)
  - **subWidth** This is to set the width of the indicator.
  - **subColor** This sets the color of the subdivision markers

  Example:

```
        renderTicks: {
          divisions: 5,
          divWidth: 1.1,
          divLength: 0.7,
          divColor: #333333,
          subDivisions: 3,
          subLength: 0.5,
          subWidth: 0.6,
          subColor: #666666
        }
```

- **Gauge pointer tip icon**
  From pull request 133: You can add an icon (image) to the tip of the gauge pointer with the iconPath and iconScale options. The icon also rotates with the angle of the pointer.

```
pointer: {
  // Extra optional pointer options:
  iconPath: 'myicon.png',  // Icon image source
  iconScale: 1,    // Size scaling factor
  iconAngle: 90.0  // Rotation offset angle, degrees
},
```

jQuery plugin

Gauge.js does not require jQuery. Anyway, if you use jQuery you may use the following plugin:

```
$.fn.gauge = function(opts) {
  this.each(function() {
    var $this = $(this),
        data = $this.data();

    if (data.gauge) {
      data.gauge.stop();
      delete data.gauge;
    }
    if (opts !== false) {
      data.gauge = new Gauge(this).setOptions(opts);
    }
  });
  return this;
};
```

Supported browsers

Gauge.js has been (not yet!) successfully tested in the following browsers:

- Chrome
- Safari 3.2+

- Firefox 3.5+
- IE 9
- Opera 10.6+
- Mobile Safari (iOS 3.2+)
- Android 2.3+

Changes

Version 1.3.8 (18.02.2024)

- Merged #229 scope polution.
- Fixed #208 so radial gradients take gauge/donut radius and linewidth into account.
- Added build script and instructions to the readme.

Version 1.3.7 (15.06.2019)

- AnimationUpdater now removes references finished rendering to prevent memory leaks.

Version 1.3.6 (28.11.2017)

- Added support for scalable staticzone sections
- Added optional Ticks(Major/Minor)
- Fixed issue #146: Prevent requestAnimationFrame() callbacks from piling up
- Fixed issue #147: Correct use of options.generateGradient for Donut

Version 1.3.5 (08.07.2017)

- Fixed issue #139: Donut support for limitMin and -Max.

Version 1.3.4 (13.05.2017)

- New feature: Add icon to tip of gauge pointer PR #133
- Fixed issue #17 for Donut.

Version 1.3.3 (09.04.2017)

Improved protection for non-numerical inputs to .set(), which could cause problems like #124.

Version 1.3.2 (11.02.2017)

Bug-fixes (#116 and #117), performance improvements.

Version 1.3.1 (05.02.2017)

Highlights:

- Added option 'minLimit' and improved max/min-hit value a lot (issue #84).
- Fixed multiple pointers color problem, issue #26 and #72.
- Added ability to scale the gauge radius to deal with issue #112.

A couple of other bugs and issues sorted out as well.

Version 1.3 (07.01.2017)

This version is a mix of new functionality and various smaller fixes and improvements. Some of the inner transformations and options definition have been slightly altered/improved and made more consistent.

Highlights:

- New feature: Value lables above the dial.
- New feature: Static color setting of the dial issue #81. Based on rsreimer's work.
- Gauge dial can be more than 180 degress. Negative options.angle allowed. (New feature)
- Better scaling in parent canvas. (Improvement)
- Set numerical precision for value fiels (Improvement)

Version 1.2.1 (9.03.2014)

- Proper handling of color params issue #47.
- Moved percentage color to example/docs + JSFiddle

Version 1.2 (16.08.2012)

- Prototype chain fix. See issue #7.
- Refactored code a bit to make it more flexible. Default class that has some extra features like gradient shadows is called Donut while more flexible one (for devs) is called BaseDonut - use it if you would don't need extra automatic stuff.
- Ability to scale gauges (requested via email) - example at JSFiddle

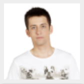
Version 1.1 (15.08.2012)

- Fixed color picker bug in FF & Opera
- Added a shadow option. See issue #5.
- Added multiple pointer option (requested via email). This needed some code refactoring. No demo for it yet. Use array of values to check it ex. gauge.set([44, 554]);
- Added wrapper for formatting text output issue #4.

Version 1.0 (27.6.2012)

- Initial release

Contact



If you encounter any problems, please use the GitHub issue tracker.
If you like gauge.js and use it in the wild, let me know.
If you want to contact me, drop me a message via email