Word Trees

Overview

A *word tree* depicts multiple parallel sequences of words. It could be used to show which words most often follow or precede a target word (e.g., "Cats are...") or to show a hierarchy of terms (e.g., a decision tree).

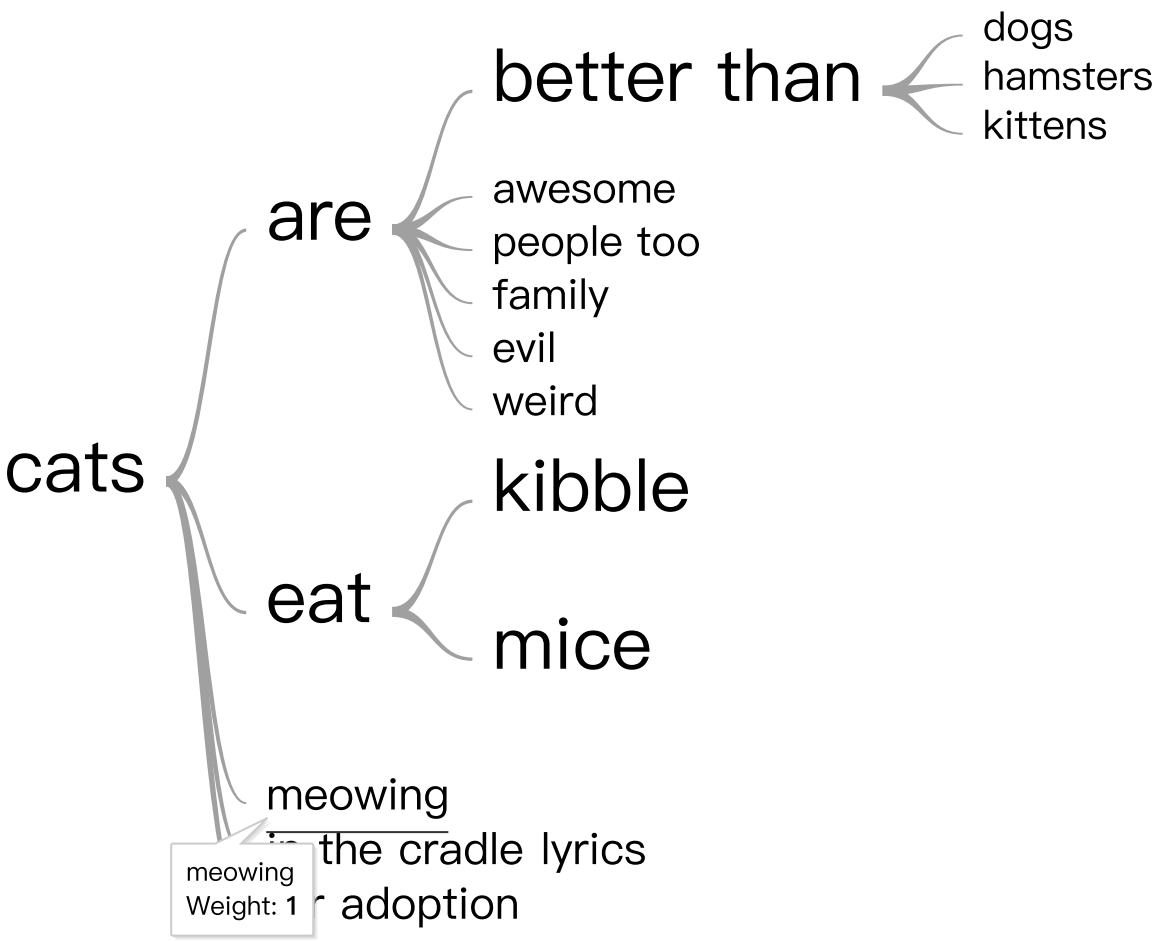
Google word trees are able to process large amounts of text quickly. Modern systems should be able to handle novel-sized amounts of text without significant delay.

Note: The word tree is in beta and may be undergoing substantial revisions in future Google Charts releases.

Word trees are rendered in the browser using SVG (http://www.w3.org/Graphics/SVG/), which means it will work in all modern browsers (e.g., Chrome, Firefox, Opera, and Internet Explorer 9+). Like all Google charts, word trees display tooltips when the user hovers over the data.

A simple example

Suppose you've collected a set of phrases about cats (e.g., "cats eat mice", "cats are better than kittens") and you want to highlight the most important attributes from the set.



CODE IT YOURSELF ON JSFIDDLE

This word tree depicts a tree of phrases, with the size of the words proportional to their usage. In this set of phrases, "cats eat mice" occurs four times, and "cats eat" occurs six times (four times with "mice", and twice with "kibble").

Try hovering over the words to see information about frequency.

Here's the web page that generates the above word tree:

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

      function drawChart() {
        var data = google.visualization.arrayToDataTable(
          [ ['Phrases'],
            ['cats are better than dogs'],
```

```
            ['cats eat kibble'],
            ['cats are better than hamsters'],
            ['cats are awesome'],
            ['cats are people too'],
            ['cats eat mice'],
            ['cats meowing'],
            ['cats in the cradle'],
            ['cats eat mice'],
            ['cats in the cradle lyrics'],
            ['cats eat kibble'],
            ['cats for adoption'],
            ['cats are family'],
            ['cats eat mice'],
            ['cats are better than kittens'],
            ['cats are evil'],
            ['cats are weird'],
            ['cats eat mice'],
          ]
        );

        var options = {
          wordtree: {
            format: 'implicit',
            word: 'cats'
          }
        };

        var chart = new google.visualization.WordTree(document.getElementById
        chart.draw(data, options);
      }
    </script>
  </head>
  <body>
    <div id="wordtree_basic" style="width: 900px; height: 500px;"></div>
  </body>
</html>
```

Word trees are *case-sensitive*. If you want "Cats" and "cats" to be treated the same, use JavaScript's `toLowerCase()` or `toUpperCase()` methods on your text before providing it to the word tree.

Implicit and explicit Word Trees

There are two ways to create word trees: *implicitly* and *explicitly*. The choice is specified with the `wordtree.format` option.

`format: 'implicit'`

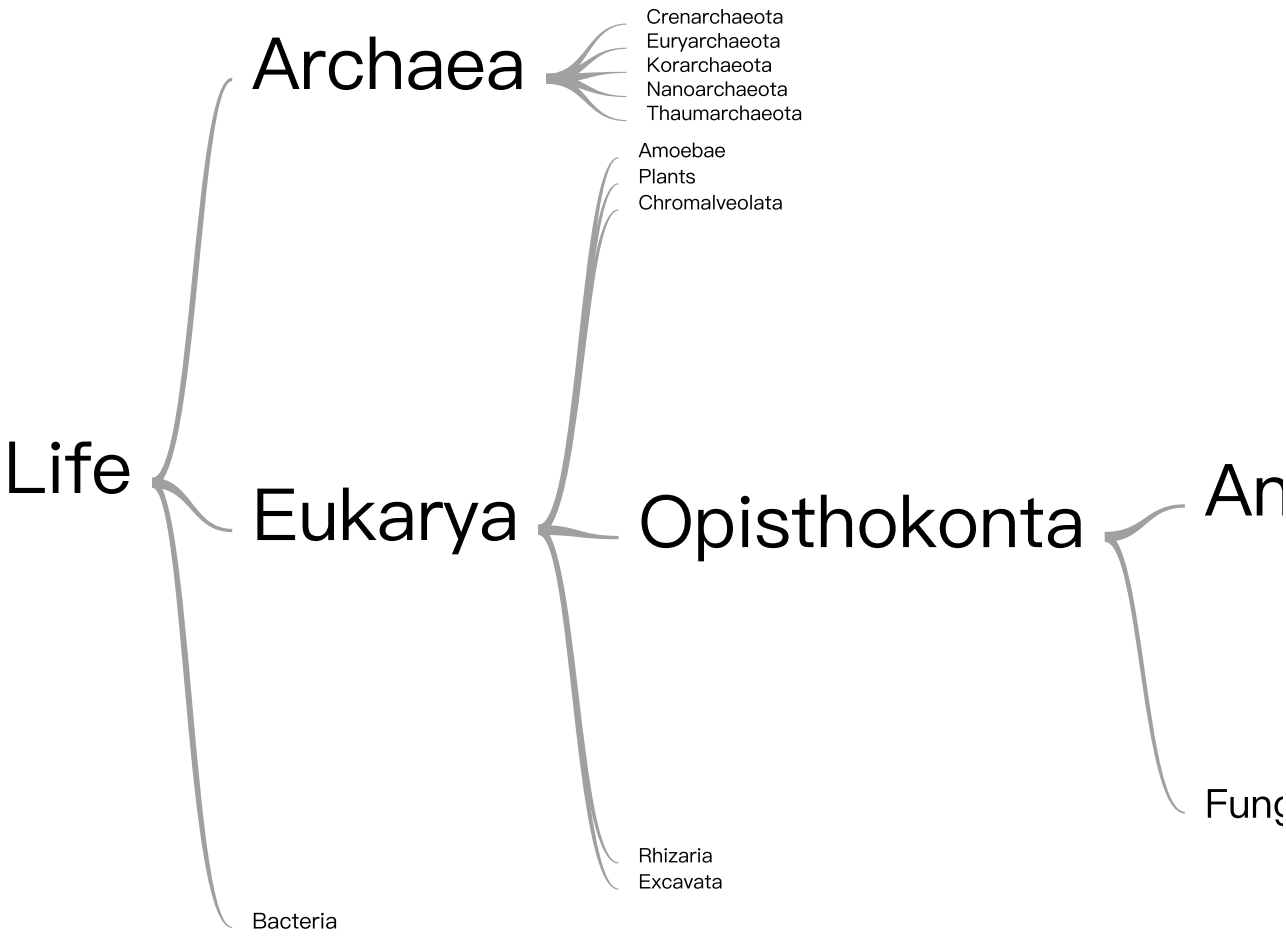
> The word tree will take a set of phrases, in any order, and construct the tree according to the frequency of the words and subphrases.

`format: 'explicit'`

> You tell the word tree what connects to what, how big to make each subphrase, and what colors to use.

The word tree in the previous section was an *implicit* Word Tree: we just specified an array of phrases, and the word tree figured out how big to make each word.

In an *explicit* word tree, the chart creator directly provides information about which words link to which, their color, and size.



There are several differences between the two word trees we've seen so far. The layout of the first word tree was calculated implicitly from a set of phrases, but in this word tree we've specified which words appear, where they appear, and how big they are.

This word tree is so wide that it's unlikely to fit onscreen. When that's the case, the word tree fades out at the edge. You can navigate the tree by clicking on any word.

IMPORTANT PARTS FULL WEB PAGE

```
function drawSimpleNodeChart() {
  var nodeListData = new google.visualization.arrayToDataTable([
    ['id', 'childLabel', 'parent', 'size', { role: 'style' }],
    [0, 'Life', -1, 1, 'black'],
    [1, 'Archaea', 0, 1, 'black'],
    [2, 'Eukarya', 0, 5, 'black'],
    [3, 'Bacteria', 0, 1, 'black'],

    [4, 'Crenarchaeota', 1, 1, 'black'],
    [5, 'Euryarchaeota', 1, 1, 'black'],
    [6, 'Korarchaeota', 1, 1, 'black'],
    [7, 'Nanoarchaeota', 1, 1, 'black'],
    [8, 'Thaumarchaeota', 1, 1, 'black'],

    [9, 'Amoebae', 2, 1, 'black'],
    [10, 'Plants', 2, 1, 'black'],
    [11, 'Chromalveolata', 2, 1, 'black'],
    [12, 'Opisthokonta', 2, 5, 'black'],
    [13, 'Rhizaria', 2, 1, 'black'],
    [14, 'Excavata', 2, 1, 'black'],

    [15, 'Animalia', 12, 5, 'black'],
    [16, 'Fungi', 12, 2, 'black'],

    [17, 'Parazoa', 15, 2, 'black'],
    [18, 'Eumetazoa', 15, 5, 'black'],

    [19, 'Radiata', 18, 2, 'black'],
    [20, 'Bilateria', 18, 5, 'black'],

    [21, 'Orthonectida', 20, 2, 'black'],
    [22, 'Rhombozoa', 20, 2, 'black'],
    [23, 'Acoelomorpha', 20, 1, 'black'],
    [24, 'Deuterostomia', 20, 5, 'black'],
    [25, 'Chaetognatha', 20, 2, 'black'],
    [26, 'Protostomia', 20, 2, 'black'],

    [27, 'Chordata', 24, 5, 'black'],
    [28, 'Hemichordata', 24, 1, 'black'],
    [29, 'Echinodermata', 24, 1, 'black'],
    [30, 'Xenoturbellida', 24, 1, 'black'],
    [31, 'Vetulicolia', 24, 1, 'black']]);
```

```
        var options = {
          colors: ['black', 'black', 'black'],
          wordtree: {
            format: 'explicit',
            type: 'suffix'
          }
        };
```

In the above code, you can see that we construct our DataTable manually. We first declare our five columns:

1. The index number (used to identify the parent of a word).
2. The text to appear in the tree. (It doesn't have to be a word.)
3. The parent of the word, with -1 meaning "no parent".
4. The size of the word.
5. The color of the word.

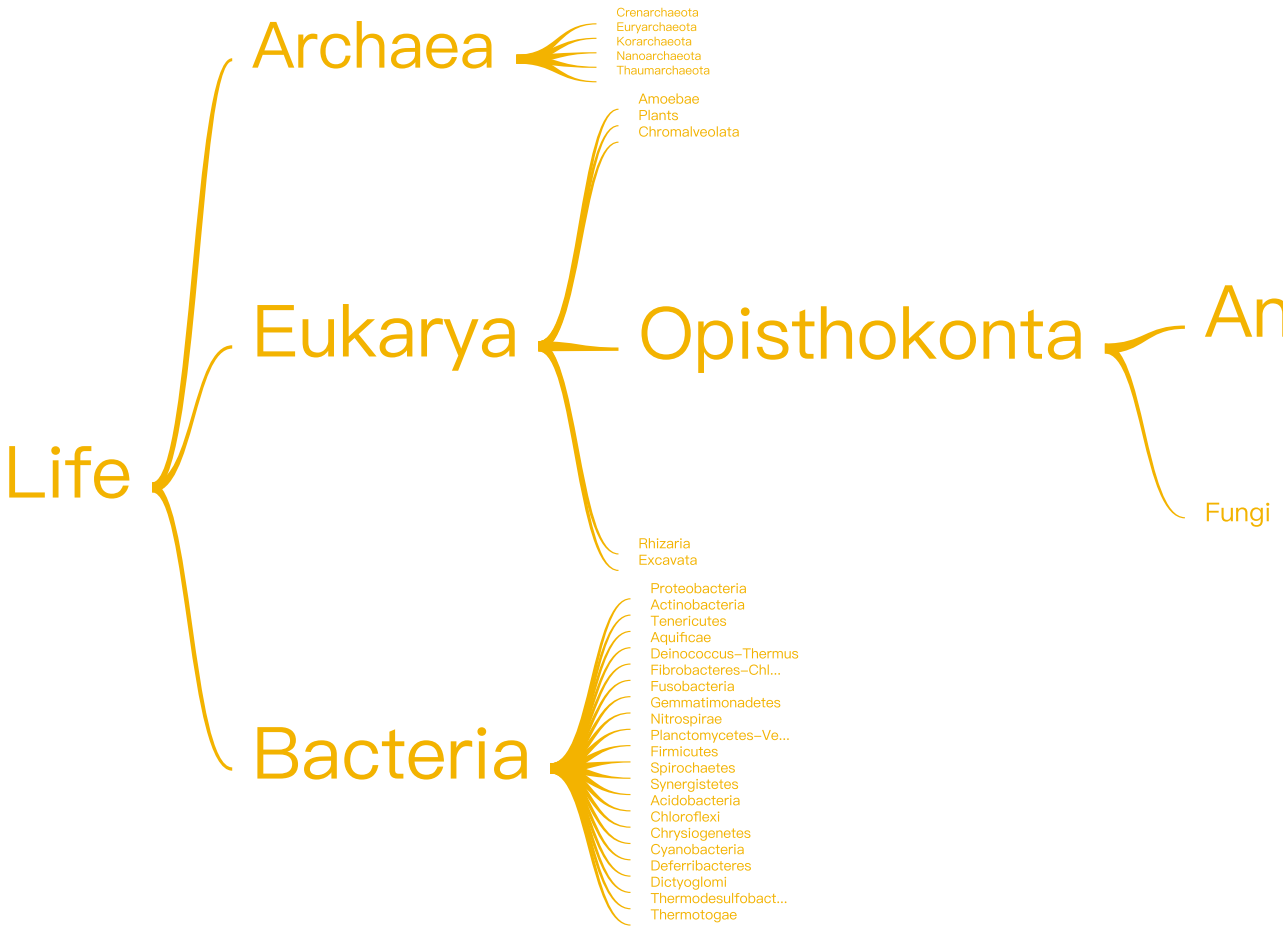
Then we add a row for each word. Here's an example:

```
nodeListData.addRow([9, 'Amoebae', 2, 1, 'black']);
```

This is row #9, adding the word `Amoebae` to the word tree. The parent is row 2 (`Eukarya`), the size is 1 (in no particular unit), and the color is 0. All of the colors in this word tree are black, but the sizes are different.

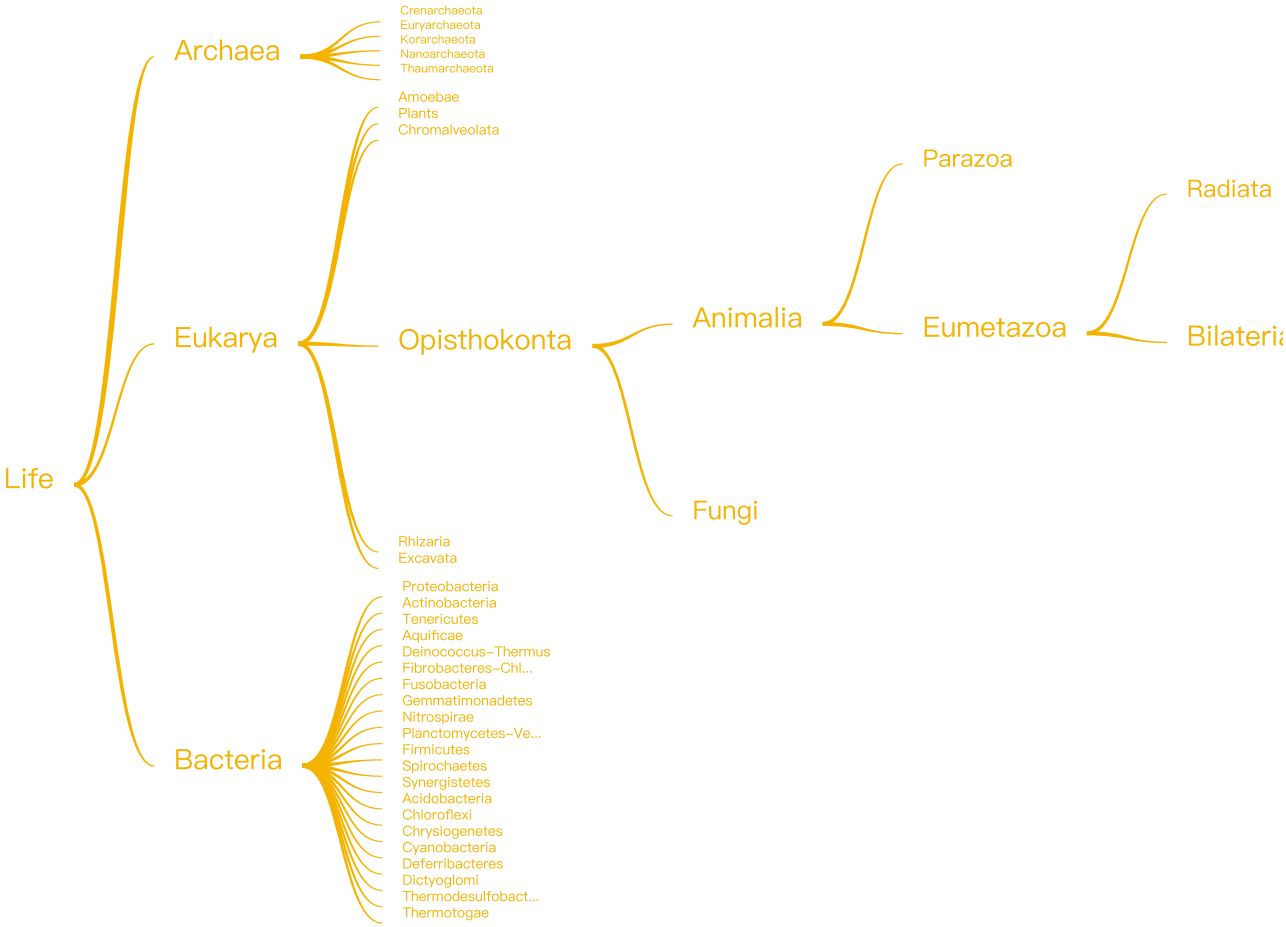
Text size

In implicit word trees, the actual display size of each word is affected by two things: the size specified for the word, and the size specified for all the words below it (that is, to the right) in the tree. In the above word tree, `Life` has three children: `Archaea` (size 1), `Eukarya` (size 5), and `Bacteria` (size 1). Archaea is displayed at a larger size than its siblings because it has five children, all of size 1. If we add the 21 phyla of bacteria, the relative sizes will change:



Because we haven't provided much vertical space for this word tree, the 21 phyla of bacteria are likely to overflow the available space. The word tree collapses them, so you see a tendril labelled "21 more". If you click on `Bacteria`, the word tree will recenter and you'll be able to see the 21 phyla. Clicking on `Bacteria` again will recenter "up" the tree.

If this automatic calculation of text size makes some words too big, you can set an upper bound with the `maxFontSize` option:



IMPORTANT PARTS FULL WEB PAGE

```
var options = {
  maxFontSize: 14,
  wordtree: {
    format: 'explicit',
    type: 'suffix'
  }
};
```

Prefix, suffix, and double Word Trees

The word trees we've seen so far are all *suffix* word trees: the root word is on the left, and words immediately following the root are on the right. In a *prefix* word tree, the root is on the right, and in a *double* word tree, it's in the center. Here's Lincoln's Gettysburg address as a prefix word tree culminating in the word 'nation':

ven years ago our fathers brought forth on this continent

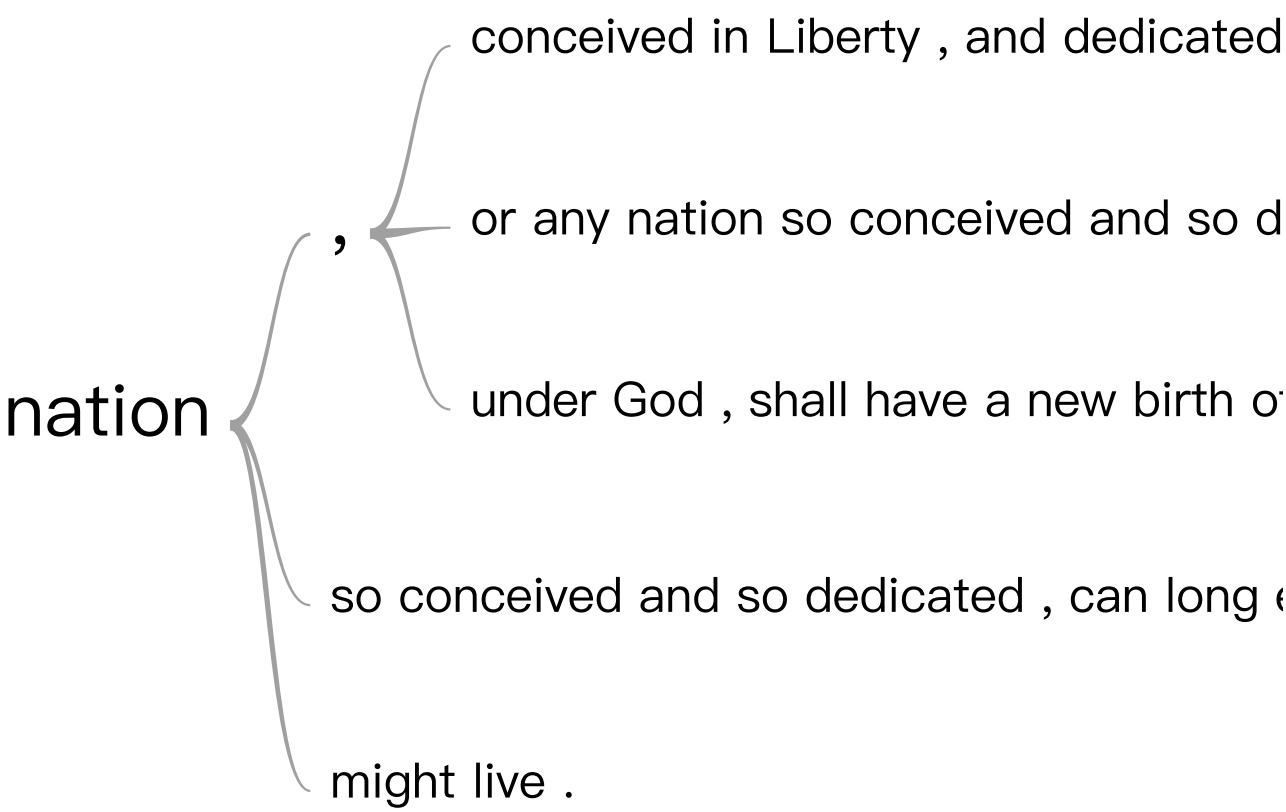
engaged in a great civil war , testing whether that nation ,

we are engaged in a great civil war , testing whether

resting place for those who here gave their lives that

ly resolve that these dead shall not have died in vain -- th

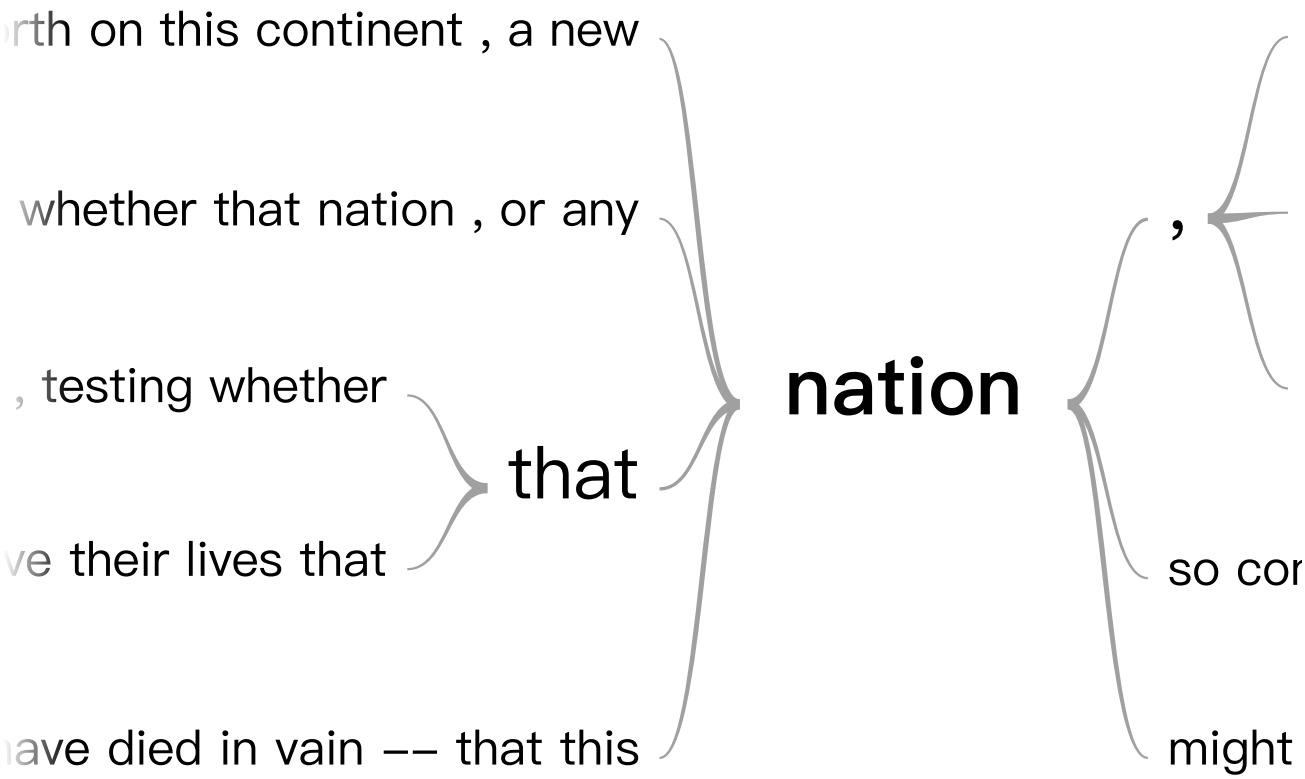
Here's the same speech as a suffix word tree, also rooted at 'nation':



You specify a suffix tree by providing `type: 'suffix'` in the chart options:

```
var options = {
  wordtree: {
    format: 'implicit',
    type: 'suffix',
    word: 'nation'
  }
};
```

A double word tree marries the prefix and suffix word trees:



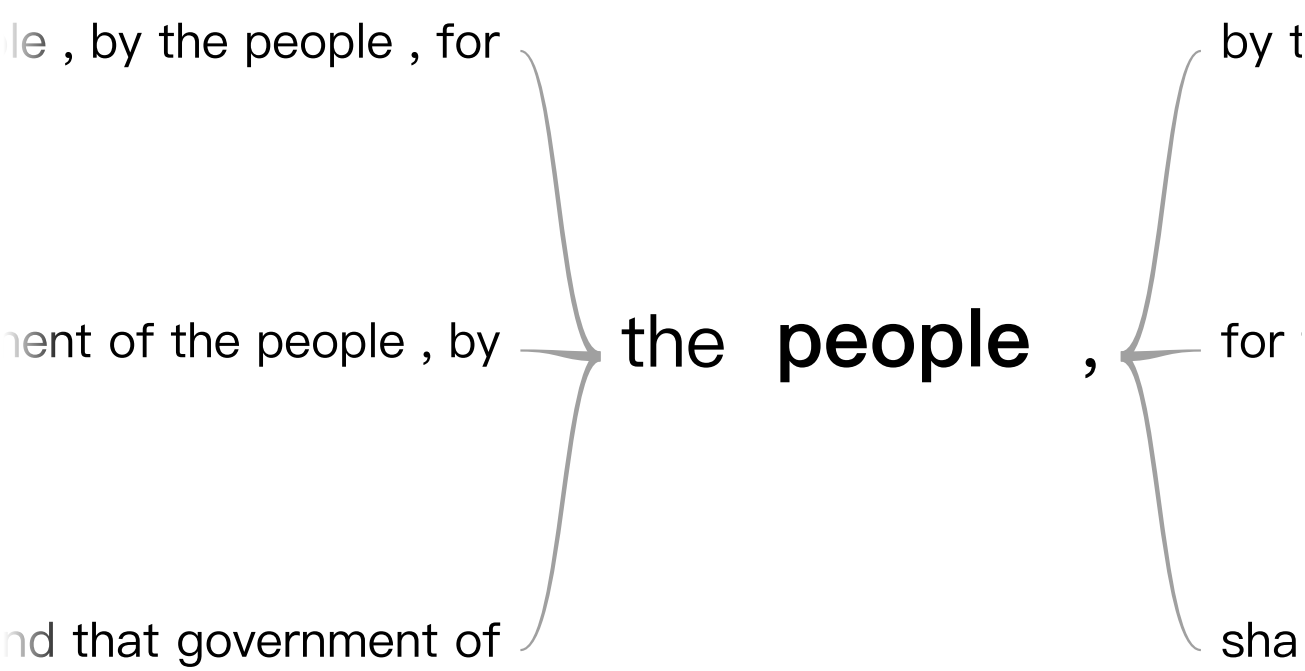
You specify a double word tree by providing `type: 'double'` in the chart options. Note that double word trees must always specify a root word, and should always be `format: 'implicit'`.

```
var options = {
  wordtree: {
    format: 'implicit',
    type: 'double',
    word: 'nation'
  }
};
```

The root of the tree is specified in the `word` option, so with a little HTML we can give users the ability to select the root from their web page:

Root word:

people GO *(try: "dead", "nation", "people", "we")*



The full web page for this word tree:

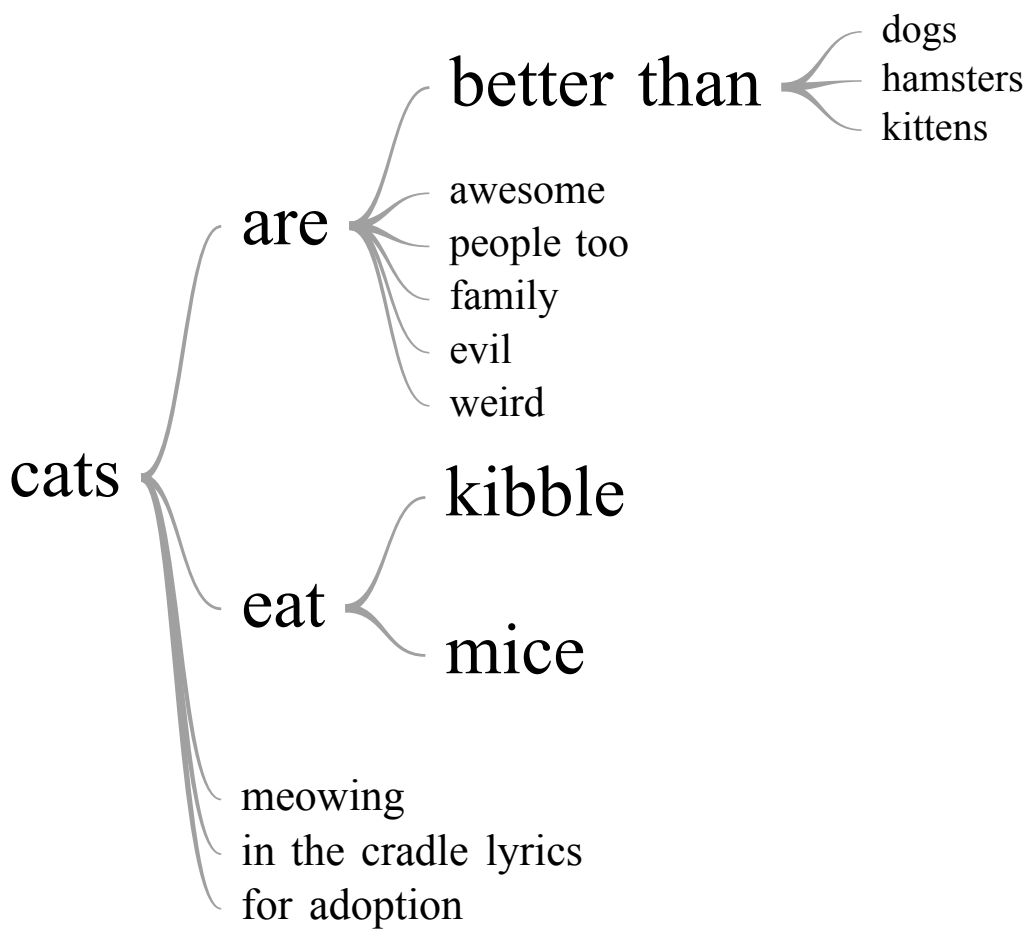
```
<html>
  <head>
    <script type="text/javascript" src="https://www.gstatic.com/charts/loader
    <script type="text/javascript">
      google.charts.load('current', {packages:['wordtree']});
      google.charts.setOnLoadCallback(drawSimpleNodeChart);
      function drawSimpleNodeChart() {
        var data = google.visualization.arrayToDataTable(
          [ ['Phrases'],
['Four score and seven years ago our fathers brought forth on this continent,
          ]
        );

        var options = {
          wordtree: {
            format: 'implicit',
            type: 'double',
            word: 'nation'
          }
        };
```

```
        var wordtree = new google.visualization.WordTree(document.getElementBy
        wordtree.draw(data, options);
      }
    </script>
  </head>
  <body>
    <div id="wordtree_double" style="width: 900px; height: 500px;"></div>
  </body>
</html>
```

Styling Word Trees

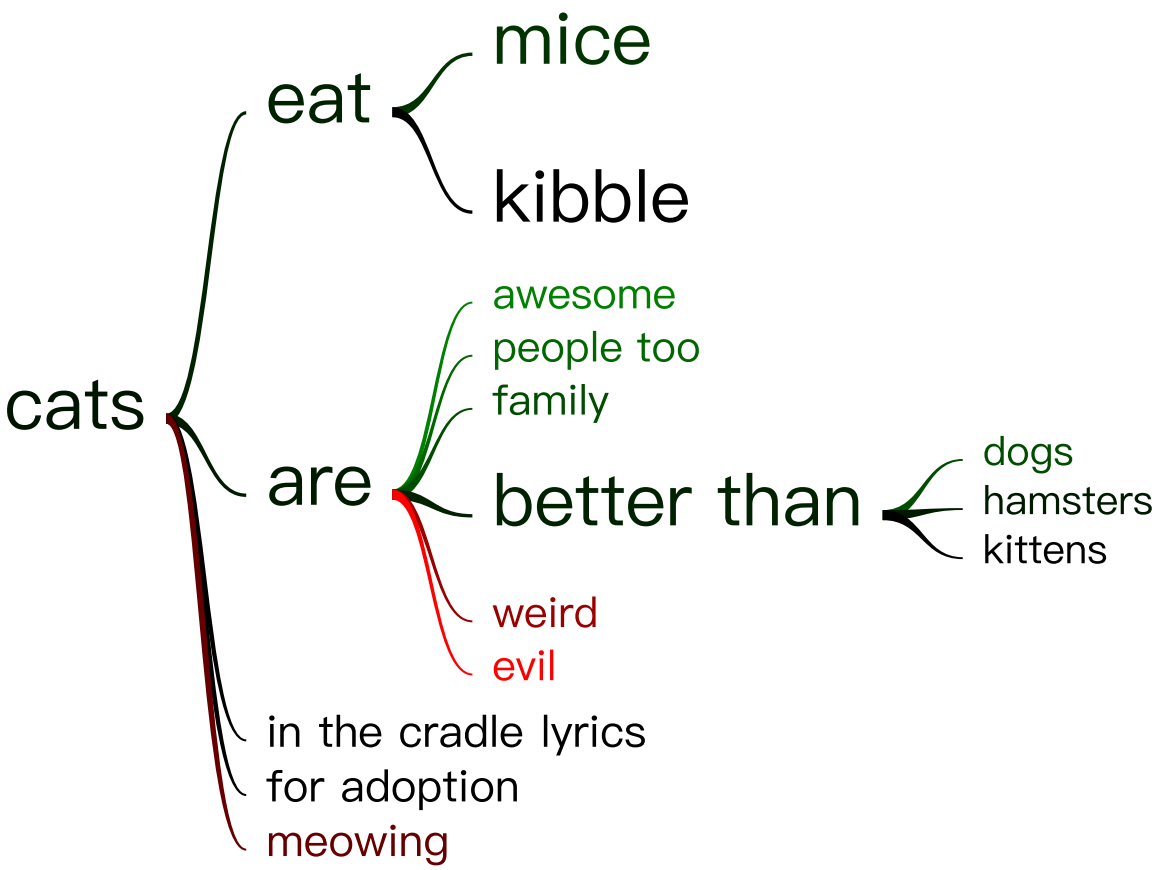
You can control the typeface and colors of your word trees. Typefaces are set with the `fontName` option:



Here's the options stanza for the above chart:

```
var options = {
  format: 'implicit',
  word: 'cats',
  fontName: 'Times-Roman'
}
```

Color is more subtle. Like size, it can optionally be used to indicate some attribute of the words in the tree. If we wanted to color the words in our "cats" word tree to indicate sentiment, we can supply that in our DataTable.



In the above word tree, we construct our DataTable as follows:

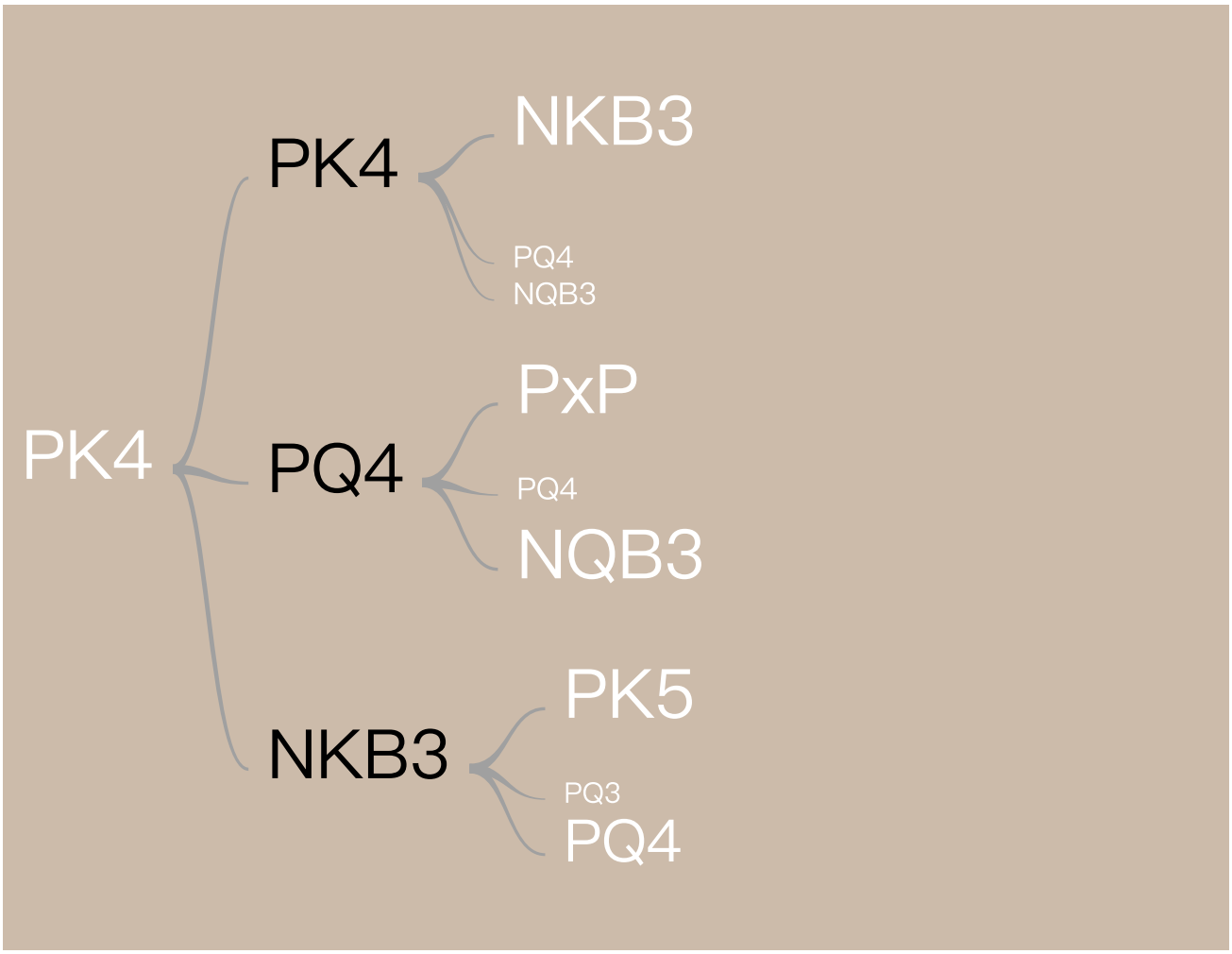
```
var data = new google.visualization.arrayToDataTable([
  ['phrase', 'size', 'value'],
  ['cats are better than dogs', 1, 8],
  ['cats eat kibble', 1, 5],
  ...
```

We set the sizes of all our words to 1, but let color (labeled arbitrarily as `'value'` in the above snippet) range from 0 ("cats are evil") to 10 ("cats are awesome") to indicate

sentiment. Then in our options, we specify three colors: lowest, neutral, and highest:

```
var options = {
  format: 'implicit',
  word: 'cats',
  colors: ['red', 'black', 'green']
};
```

Colors can also be specified explicitly. Here's a word tree that shows potential moves for a chess game. In addition setting the colors of the words to `'white'` and `'black'`, the background color is set to hex value `'#cba'` with the `backgroundColor` option:



The DataTable for this word tree is declared as follows:

```
function drawChart() {
  var data = new google.visualization.arrayToDataTable([
    ['id', 'childLabel', 'parent', 'weight', { role: 'style' }],
    [0, 'PK4', -1, 1, 'white'],
    [1, 'PK4', 0, 1, 'black'],
    ...
```

Note that the column containing the colors is identified not by a text label like `'parent'` or `'weight'`, but the style role: `{ role: 'style' }`.

We set the background color in the options:

```
var options = {
  format: 'explicit',
  backgroundColor: '#cba'
};
```

Tokenizing sentences

Implicit word trees are broken into sentences and words according to simple rules, expressed as regular expressions. In rare cases you might want to override the default behavior, and in those cases you can use the `wordSeparator` and `sentenceSeparator` options.

If you're fluent in regular expressions, the defaults may make sense to you:

sentenceSeparator: `\s*(.+?(?:[?!]+|$|\.(?=\s+[A-Z]|$)))\s*`

wordSeparator: `([!?,;:.&"-]+|\S*[A-Z]\.|\S*(?:[^!?,;:.\s&-]))`

Note: Regex splitting is nonstandard in Internet Explorer 8 and may lead to unexpected results.

Loading

The `google.charts.load` package name is `"wordtree"`:

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

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

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

Data format

Rows: Each row in the DataTable represents text to be displayed. For implicit word trees, the text of all rows is combined and tokenized before being displayed.

Columns for Implicit Word Trees:

	Column 0	Column 1	Column 2
Purpose:	Text	Size (optional)	Style (optional)
Data Type:	string	number	string
Role:	domain	data	data

Columns for Explicit Word Trees:

	Column 0	Column 1	Column 2	Column 3	Column 4
Purpose:	ID	Text	Parent	Size	Style
Data Type:	number	string	number	number	string
Role:	domain	data	data	data	data

Configuration options

Name	
colors	A list of three colors, specified either by English name or hex value. The colors for words will be taken from a spectrum that begins at the first color (the low value), moves through the middle color (neutral), and ends at the last color (high). **Type:** Array of strings **Default:** default colors
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

fontName	The word tree typeface. **Type:** string **Default:** default
height	Height of the chart, in pixels. **Type:** number **Default:** height of the containing element
maxFontSize	The upper limit for font size of displayed words. **Type:** number **Default:** null
width	Width of the chart, in pixels. **Type:** number **Default:** width of the containing element
wordtree.format	If `implicit`, the input text will be split into sentences and words, and displayed according to frequency. If `explicit`, words and their connections are specified directly. **Type:** string **Default:** `'implicit'`
wordtree.sentenceSeparator	For implicit word trees, the regular expression to use to break the text into sentences. The sentences are then broken into words using the `wordSeparator` option. **Type:** regex **Default:** `\s*(.+?(?:[?!]+|$\|\.(?=\s+[A-Z]\|$)))\s*`
wordtree.type	Whether the word tree is a prefix tree (root word on the right), a suffix tree (left), or double tree (middle). **Type:** string **Default:** 'suffix'
wordtree.word	For implicit word trees, which word to use as the root of the tree. (Note that word trees are case sensitive.) This option must be specified for double word trees. **Type:** string **Default:** null
wordtree.wordSeparator	For implicit word trees, the regular expression to use to break sentences into individual words to be displayed. **Type:** regex

	Default: `/([!?,;:.&"-]+|\S*[A-Z]\.|\S*(?:[^!?,;:.\s&-]))`

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://developers.google.com/chart/interactive/docs/reference#vis . **Return Type:** none
`clearChart()`	Clears the chart, and releases all of its allocated resources. **Return Type:** none

Events

Name	
`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 selects a word, either to "zoom" into or out of the tree. **Properties:** word, color, weight

Data policy

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

上次更新日期：二月 23, 2017