Pace

Automatic page load progress bar

version v1.2.4

Star 15,517

Home ▾

Pace

Include pace.js (https://github.com/CodeByZach/pace/releases/latest) and the theme (https://codebyzach.github.io/pace/) css of your choice on your page (as early as is possible), and you're done!

Pace will automatically monitor your ajax requests, event loop lag, document ready state, and elements on your page to decide the progress. On ajax navigation it will begin again!

If you use AMD (https://requirejs.org) or Browserify (http://browserify.org), require in pace.js (https://github.com/CodeByZach/pace/releases/latest) and call `pace.start()` as early in the loading process as is possible.

Example

```
<head>
    <script src="pace.min.js"></script>
    <link rel="stylesheet" href="pace-theme-default.min.css"
</head>
```

Configuration

Pace is fully automatic, no configuration is necessary to get started.

If you would like to make some tweaks, here's how:

You can set `window.paceOptions` before bringing in the file:

```
paceOptions = {
    // Disable the 'elements' source
    elements: false,

    // Only show the progress on regular and ajax-y page navigation, not every request
    restartOnRequestAfter: false
}
```

You can also put options on the script tag:

```
<script data-pace-options='{ "ajax": false }' src="pace.min.js"></script>
```

If you're using AMD or Browserify, you can pass your options to `start`:

```
define(['pace'], function(pace) {
        pace.start({
        document: false
    });
});
```

Themes

Pace includes a bunch of themes (https://codebyzach.github.io/pace/) to get you started. Just include the appropriate css file. Send us a Pull reques (https://github.com/CodeByZach/pace/pulls)t with any interesting themes you create.

Collectors

Collectors are the bits of code which gather progress information. Pace includes four default collectors:

- Ajax

  Monitors all ajax requests on the page

- Elements

  Checks for the existance of specific elements on the page

- Document

  Checks the document readyState

- Event Lag

  Checks for event loop lag signaling that javascript is being executed

They can each be configured or disabled through configuration options of the same name.

```
paceOptions = {
    ajax: false, // disabled
    document: false, // disabled
    eventLag: false, // disabled
    elements: {
        selectors: ['.my-page']
    }
};
```

Add your own classes to `paceOptions.extraSources` to add more sources. Each source should either have a `.progress` property, or a `.elements` property which is a list of objects with `.progress` properties. Pace will automatically handle all scaling to make the progress changes look smooth to the user.

Elements

Elements being rendered to the screen is one way for us to decide that the page has been rendered. If you would like to use that source of information (not required at all), specify one or more selectors. You can comma separate the selectors to propertly handle error states, where the progress bar should disappear, but the element we are looking for may never appear:

```
paceOptions = {
    elements: {
        selectors: ['.timeline,.timeline-error', '.user-profile,.profile-error']
    }
}
```

Pace will consider the elements test successful when each selector matches something. For this example, when either `.timeline` or `.timeline-error` exist, and either `.user-profile` or `.profile-error` exist.

Restart Rules

Most users want the progress bar to automatically restart when a pushState event occurs (generally means ajax navigation is occuring). You can disable this:

```
paceOptions = {
    restartOnPushState: false
}
```

You can also have pace restart on every ajax request which lasts longer than x ms. You'll want to disable this if you make ajax requests the user doesn't need to know about, like precaching:

```
paceOptions = {
    restartOnRequestAfter: false
}
```

You can always trigger a restart manually by calling `Pace.restart()`

See the source (https://github.com/CodeByZach/pace/blob/master/pace.js) for a full list of all options.

API

Pace exposes the following methods:

- `Pace.start` : Show the progress bar and start updating. Called automatically if you don't use AMD or CommonJS.
- `Pace.restart` : Show the progress bar if it's hidden and start reporting the progress from scratch. Called automatically whenever `pushState` or `replaceState` is called by default.
- `Pace.stop` : Hide the progress bar and stop updating it.
- `Pace.track` : Explicitly track one or more requests, see Tracking below
- `Pace.ignore` : Expliticly ignore one or more requests, see Tracking below

Events

Pace fires the following events:

- `start` : When pace is initially started, or as a part of a restart
- `stop` : When pace is manually stopped, or as a part of a restart
- `restart` : When pace is restarted (manually, or by a new AJAX request)
- `done` : When pace is finished
- `hide` : When the pace is hidden (can be later than `done` , based on `ghostTime` and `minTime`)

You can bind onto events using the `on` , `off` and `once` methods:

- `Pace.on(event, handler, [context])` : Call `handler` (optionally with context) when `event` is triggered
- `Pace.off(event, [handler])` : Unbind the provided `event` and `handler` combination.
- `Pace.once(event, handler, [context])` : Bind `handler` to the next (and only the next) incidence of `event`

Tracking

By default, Pace will show any ajax requests which begin as a part of a normal or ajax-y page load, or which last longer than 500ms.

You can disable all ajax tracking by setting `ajax` to false:

```
Pace.options = {
    ajax: false
}
```

You can disable ajax tracking except on page navigation by setting `restartOnRequestAfter` to false:

```
Pace.options = {
    restartOnRequestAfter: false
}
```

You can manually disable tracking for a specific request or requests by triggering them within a `Pace.ignore` callback:

```
Pace.ignore(function(){
    $.ajax(...)
});
```

You can force the progress bar to be shown for a specific request by triggering them within a `Pace.track` callback:

```
Pace.track(function(){
    $.ajax(...)
});
```

You can also ignore URLs based on a pattern:

```
Pace.options = {
    ajax: {
        ignoreURLs: ['some-substring', /some-regexp/]
    }
}
```

Dependencies

None!

Support

Pace is designed to support IE8+ (standards mode), FF 3.5+, Chrome, Safari 4+, Opera 10.5+, and all modern mobile browsers. If you run into a compatibility issue, or can make a case for supporting something else, please create an Issue (https://github.com/CodeByZach/pace/issues/new).

Size

pace.js is 4kb minified and gzipped. The themes vary between 0.5 and 4kb.

Issues

We have obviously not tested this on every website. If you run into an issue, or find a way the automatic detection could be better, please create an Issue (https://github.com/CodeByZach/pace/issues/new). If you can include a test case, that's even better.

Contributing

Pull requests (https://github.com/CodeByZach/pace/pulls) welcome!

Credits

HubSpot (https://dev.hubspot.com)

CodeByZach (https://github.com/CodeByZach)

Javascript by Zack Bloom (https://twitter.com/zackbloom) CSS by Adam Schwartz (https://twitter.com/adamfschwartz)

Themes inspired by Mary Lou (https://tympanus.net/codrops/2013/09/18/creative-loading-effects/)

Project inspired by nprogress (https://ricostacruz.com/nprogress/)