# JavaScript Performance


Order Matters

```
<!DOCTYPE html>
<html class="no-js  page-1 app-F_109" lang="en" >
<head>
  <meta http-equiv="x-ua-compatible" content="IE=edge" />
  <meta charset="utf-8">
  <title>Home</title>
  <link rel="stylesheet" href="/i/app_ui/css/Core.min.css?v=5.1.1.00.08" type="text/css" />
<link rel="stylesheet" href="/i/app_ui/css/Theme-Standard.min.css?v=5.1.1.00.08" type="text/css"
<link rel="stylesheet" href="/i/libraries/jquery-ui/1.10.4/themes/base/jquery-ui.min.css?v=5.1.1.

  <link rel="stylesheet" href="/i/libraries/font-apex/1.0/css/font-apex.min.css?v=5.1.1.00.08" ty
<link rel="stylesheet" href="/i/themes/theme_42/1.1/css/Core.min.css?v=5.1.1.00.08" type="text/cs


  <link rel="stylesheet" href="/i/themes/theme_42/1.1/css/Vita.min.css?v=5.1.1.00.08" type="text/



  <link rel="shortcut icon" href="/i/favicon.ico">
<link rel="icon" sizes="16x16" href="/i/favicon-16x16.png">
<link rel="icon" sizes="32x32" href="/i/favicon-32x32.png">
<link rel="apple-touch-icon" sizes="180x180" href="/i/favicon-180x180.png">

<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<meta http-equiv="Pragma" content="no-cache" /><meta http-equiv="Expires" content="-1" /><meta ht
  <meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=1.0, user-s
</head>
<body class="t-PageBody t-PageBody--hideLeft t-PageBody--hideActions no-anim   apex-side-nav apex
<form action="wwv_flow.accept" method="post" name="wwv_flow" id="wwvFlowForm" novalidate  autocom
<input type="hidden" name="p_flow_id" value="109" id="pFlowId" /><input type="hidden" name="p_flo
<header class="t-Header" id="t_Header">

  <div class="t-Header-branding">
    <div class="t-Header-controls">
      <button class="t-Button t-Button--icon t-Button--header t-Button--headerTree" title="Expand
    </div>
```

# Generalization Warning

- CSS
- ...
- ...
- ...
- ...
- JavaScript

# Generalization Warning

```
<link rel="stylesheet" href="/i/app_ui/css/Core.min.css?v=5.1.1.00.08" type="text/css" />
...
...
...
...
...

<script type="text/javascript" src="/i/libraries/apex/minified/widget.apexTabs.min.js?v=5.1.1.00.
```

# But why?

Short Answer:

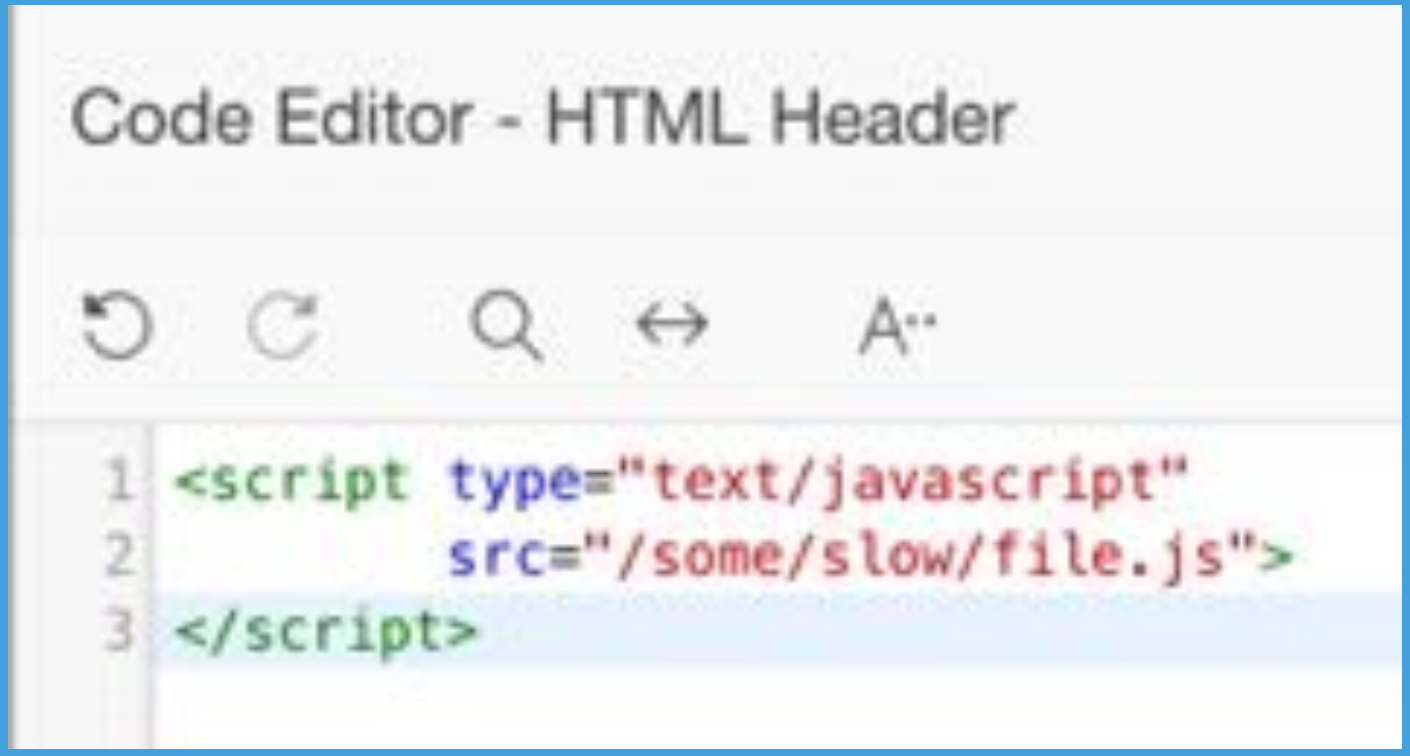
Limited HTTP requests

Long Answer:

gives priority to visual presentation displayed to user, as opposed to logic or processing

# But why?

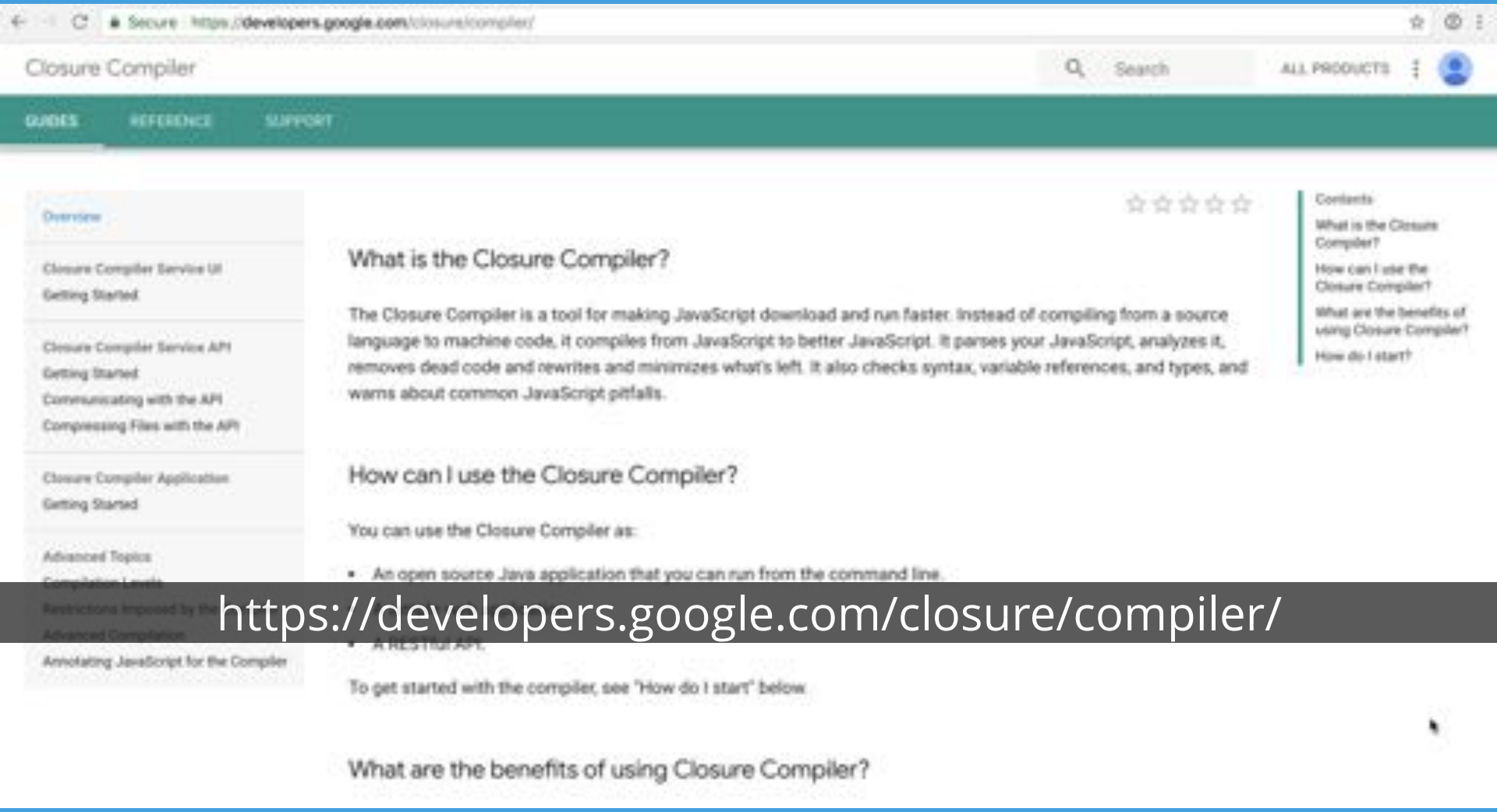
So be wary of...



# Combine and Minify

```
<script src="#APP_IMAGES#js/file1.js"></script>
<script src="#APP_IMAGES#js/file2.js"></script>
<script src="#APP_IMAGES#js/file3.js"></script>
```

```
<script src="#APP_IMAGES#js/big-file.js"></script>
```




https://developers.google.com/closure/compiler/


Closure Compiler
REST API | Help
Add a URL:
Add
Example: http://www.example.com/foo/bar.js
Optimization: Whitespace only Simple Advanced
Which optimization is right for my code?
Formatting: Pretty print Print input delimiter
Compile Reset
// ==ClosureCompiler==
// @compilation_level ADVANCED_OPTIMIZATIONS
// @output_file_name default.js
// ==/ClosureCompiler==
// ADD YOUR CODE HERE
function hello(name) {
alert('Hello, ' + name);
}
hello('New user');
Compilation was a success!
Original Size: 102 bytes gzipped (92 bytes uncompressed)
Compiled Size: 45 bytes gzipped (25 bytes uncompressed)
Saved 55.88% off the gzipped size (72.83% without gzip)
The code may also be accessed at default.js
Compiled Code
Warnings
Errors
POST data
alert("Hello, New user");
©2009 Google - Terms of Service - Privacy Policy - Google Home

```
function hello(name) {
  alert('Hello, ' + name);
}
hello('New user');
```

```
alert("Hello, New user");
```

# Legacy Javascript

Legacy htmldb_get +

Legacy Browser +

Legacy APEX

```
function getData() {
  var ajaxRequest = new htmldb_Get(null, & APP_ID., 'APPLICATION_PROCESS=getData', 0);
  ajaxRequest.add('P1_SEARCH', $v('P1_SEARCH'));
  ajaxResult = ajaxRequest.get();
  if (ajaxResult) {
    // do something
  }
}
```

=

```
function getData() {
  var ajaxRequest = new htmldb_Get(null, & APP_ID., 'APPLICATION_PROCESS=getData', 0);
  ajaxRequest.add('P1_SEARCH', $v('P1_SEARCH'));
  ajaxResult = ajaxRequest.get();
  if (ajaxResult) {
    // do something
  }
}
```

Can Block

# Who do I need to care?

- If you using an old(er) version of APEX
- Old browser
- Blocking behaviour can cause 'slowness' perception for users.

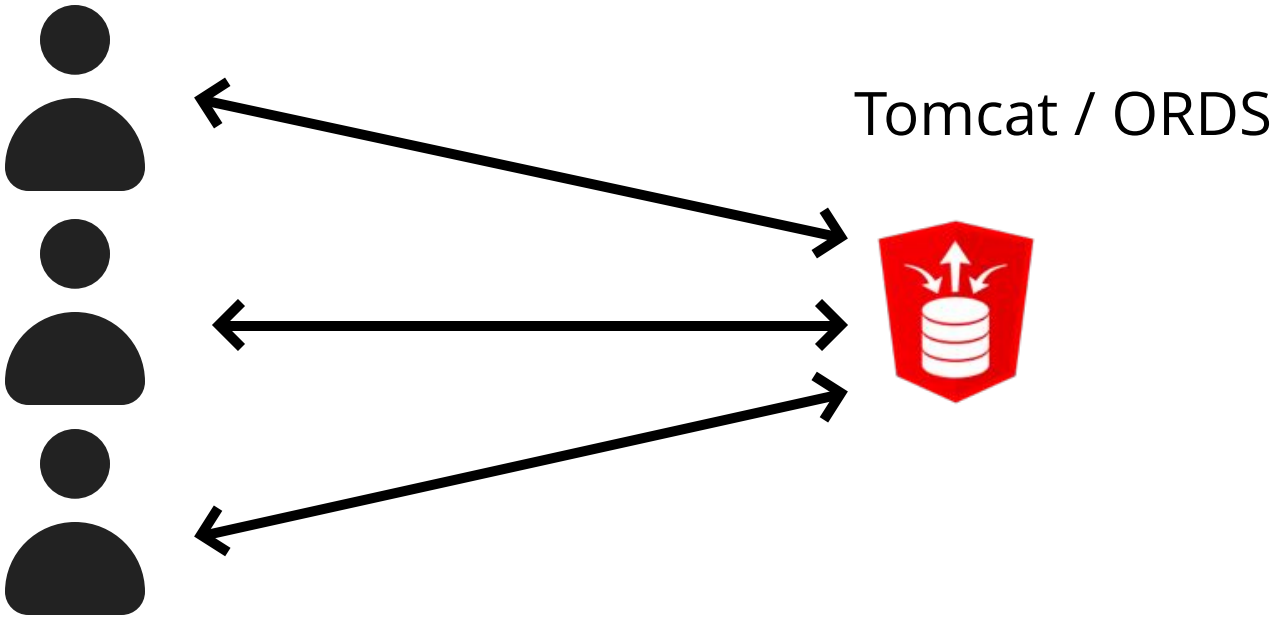
# New Method
## apex.server.process

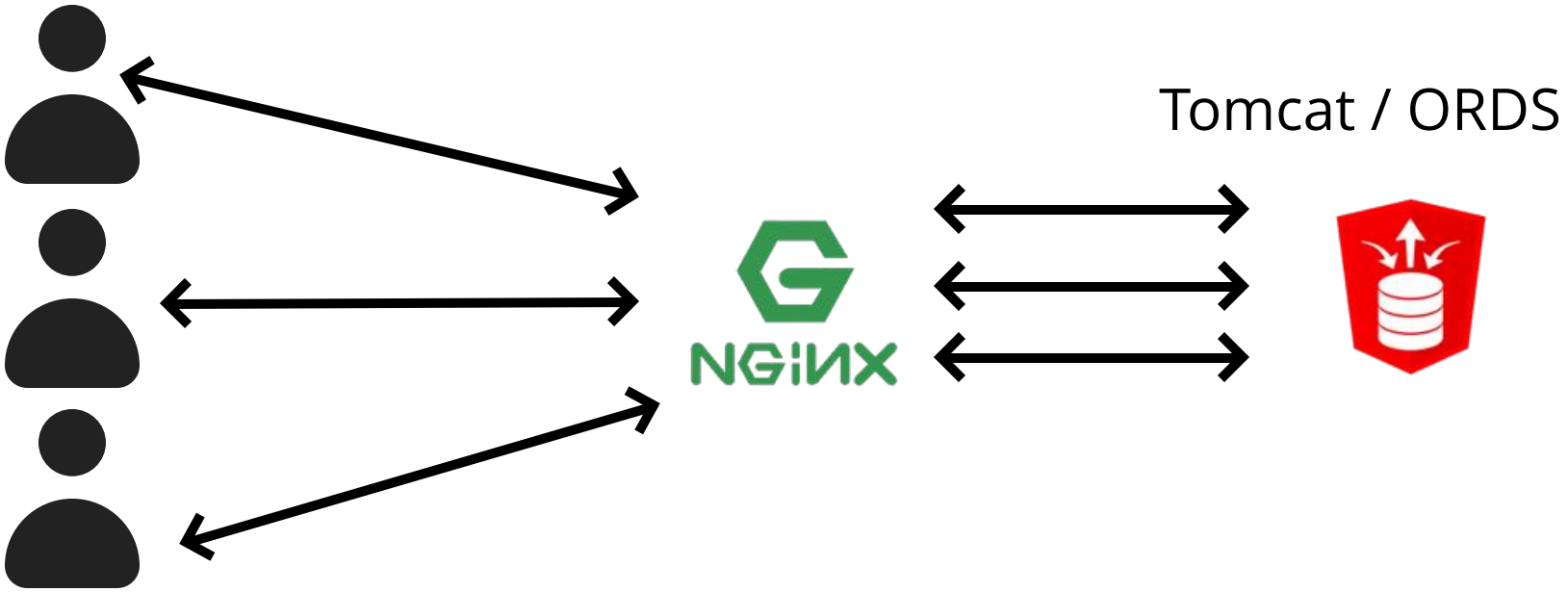
```
apex.server.process ( "GET_DATA", {
        x01: "test",
  },{
      success: function( pData ) {
          // do Something
    }
  } );

// now I don't need to wait
run_some_other_function();
```

# Upgrade!

# Webserver Considerations


Tomcat / ORDS


Tomcat / ORDS
NGINX

# GZIP Compression

Rationale

- compress the content before sending it back to the browser
- less data to send = quicker response
- NGINX processes free up faster to serve other requests

# GZIP Compression

```
location /i {
            gzip on;
            gzip_vary on;
            gzip_min_length 1024;
            gzip_proxied expired no-cache no-store private auth;
            gzip_types text/plain
                       text/css
                       text/xml
                       text/javascript
                       application/javascript
                       application/xml
            gzip_disable "MSIE [1-6]\.";
            root /apex;
            include /etc/nginx/mime.types;
        }
```

# Expiry Headers

Rationale

- Tell the browser that this content can be cached for 'X' duration
- Browser can use local cache and avoid a request to the webserver
- Fewer requests to webserver means able to deal with more 'valid' requests

# Expiry Headers

```
location /i {
            ...
            expires 1h;
            ...
            root /apex;
            include /etc/nginx/mime.types;
        }
```

| Configuration | Requests/s | Relative Factor |
|---|---|---|
| Tomcat + ORDS | 16.78 | 1.00 |
| Tuned ORDS | 19.84 | 1.18 |
| NGINX Proxy (serving /i) | 23.39 | 1.39 |
| NGINX Proxy (GZIP) | 25.69 | 1.53 |
| NGINX Proxy (Expiry) | 27.33 | 1.63 |
| NGINX Proxy (GZIP + Expiry) | 33.96 | 2.02 |

# HTTP/2

# HTTP/2

- **Multiplexing and concurrency**: Several requests can be sent in rapid succession on the same TCP connection, and responses can be received out of order - eliminating the need for multiple connections between the client and the server

- **Stream dependencies**: the client can indicate to the server which of the resources are more important than the others
- **Header compression**: HTTP header size is drastically reduced

- **Server push**: The server can send resources the client has not yet requested

# "It is / can be *really* FAST"

- Modern Browsers support it
- Requires SSL  (Lets Encrypt / Self Signed etc)
- No need for "domain sharding"
- Optimises the TCP layer
  - switch from multiple connections to 1 long-lived one

```
server {
        listen                  443 ssl http2;
        server_name             localhost;

        ssl_protocols           TLSv1 TLSv1.1 TLSv1.2;
        ssl_certificate         nginx.crt;
        ssl_certificate_key     nginx.key;

        ...
}
```

# Debouncing & Throttling

- Debounce
  - Group multiple sequential calls into one call
- Throttle
  - Disallow more than one execution every X

Key difference - debouncing guarantees execution regularly, at least once every X

# Lets look at an Example


S1 (cascades) -> S2 (cascades) -> S3
Charts refreshed multiple times

```
function refreshCharts() {
  apex.region('chart1').refresh();
  apex.region('chart2').refresh();
  console.log('refreshCharts called via Debounce');
}



apex.util.debounce(refreshCharts, 500))
```

# Lodash

A modern JavaScript utility library delivering modularity, performance & extras.

Documentation FP Guide

```
_.defaults({ 'a': 1 }, { 'a': 3, 'b': 2 });
// → { 'a': 1, 'b': 2 }
_.partition([1, 2, 3, 4], n => n % 2);
// → [[1, 3], [2, 4]]
```

Star 24,301 Fork 2,462 Follow @lodashjs Tweet

## Download

Core build (~4kB gzipped)

Full build (~24kB gzipped)

CDN copies

Lodash is released under the MIT license & supports modern environments.
Review the build differences & pick one that's right for you.

## Installation

In a browser:

```
<script src="lodash.js"></script>
```

# UNDERSCORE.JS

Underscore is a JavaScript library that provides a whole mess of useful functional programming helpers without extending any built-in objects. It's the answer to the question: "If I sit down in front of a blank HTML page, and want to start being productive immediately, what do I need?" ... and the tie to go along with jQuery's tux and Backbone's suspenders.

Underscore provides over 100 functions that support both your favorite workaday functional helpers: **map**, **filter**, **invoke** — as well as more specialized goodies: function binding, javascript templating, creating quick indexes, deep equality testing, and so on.

A complete Test Suite is included for your perusal.

You may also read through the annotated source code.

Enjoying Underscore, and want to *turn it up to 11?* Try Underscore-contrib.

The project is hosted on GitHub. You can report bugs and discuss features on the issues page, on Freenode in the `#documentcloud` channel, or in our Gitter channel.

*Underscore is an open-source component of DocumentCloud.*

## Downloads *(Right-click, and use "Save As")*

| | |
|---|---|
| Development Version (1.8.3) | *52kb, Uncompressed with Plentiful Comments* |
| Production Version (1.8.3) | *5.7kb, Minified and Gzipped* (Source Map) |
| Edge Version | *Unreleased, current `master`, use at your own risk* |

```
function refreshCharts() {
  apex.region('chart1').refresh();
  apex.region('chart2').refresh();
  console.log('refreshCharts called via Debounce');
}

var _refreshCharts = _.debounce(refreshCharts,
                                500,
                                {'maxwait': 500 }
                               );
```

# Thank You!