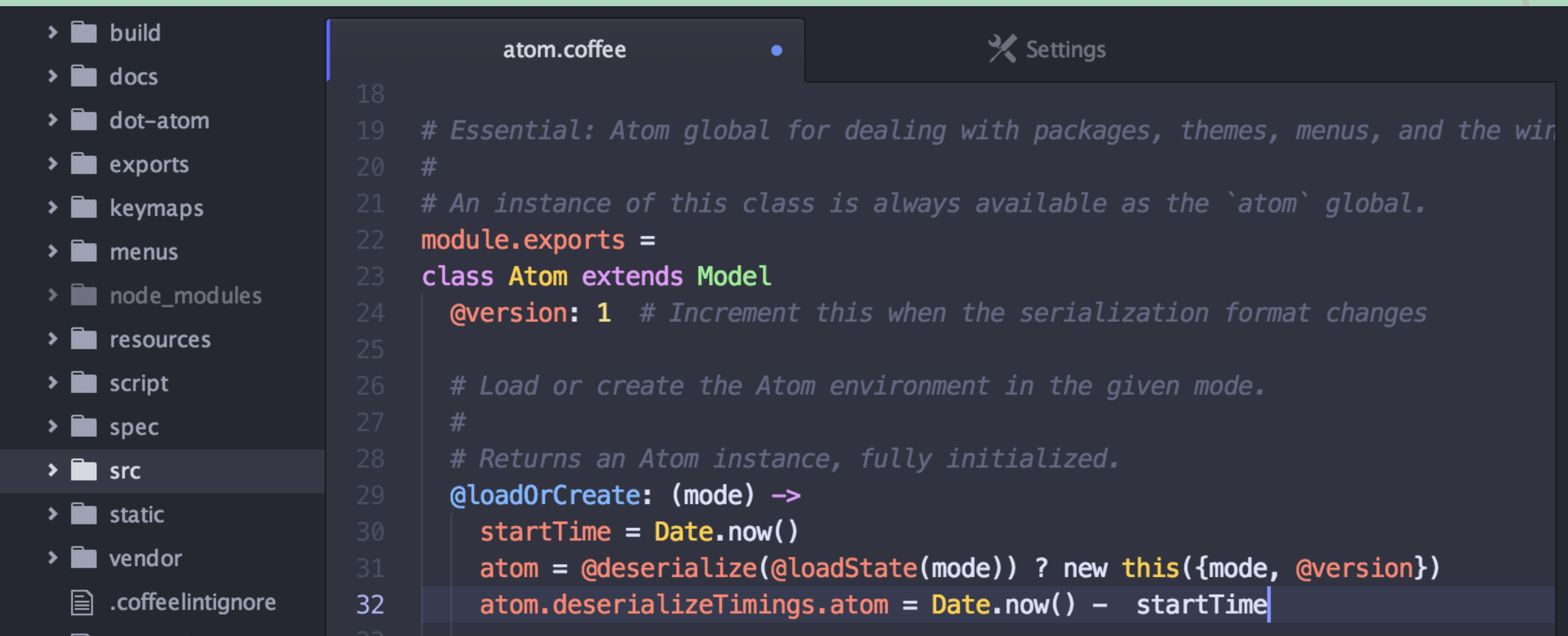
ATOM

A hackable text editor

for the 21st Century



关于我

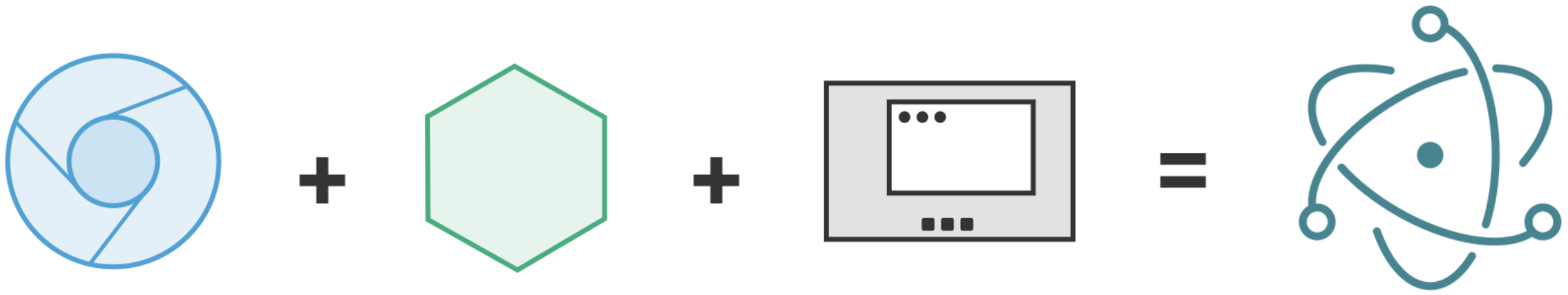
- 王子亭
- 20 岁
- Node.js 开发者
- LeanCloud
- https://jysperm.me
- GitHub: jysperm

History

Why Atom

- Out of the box (Sublime Text)
- Customizable (Emacs)
- JavaScript and Web Technology
- Open Source and Community

Web Technology



Chromium
for making
web pages

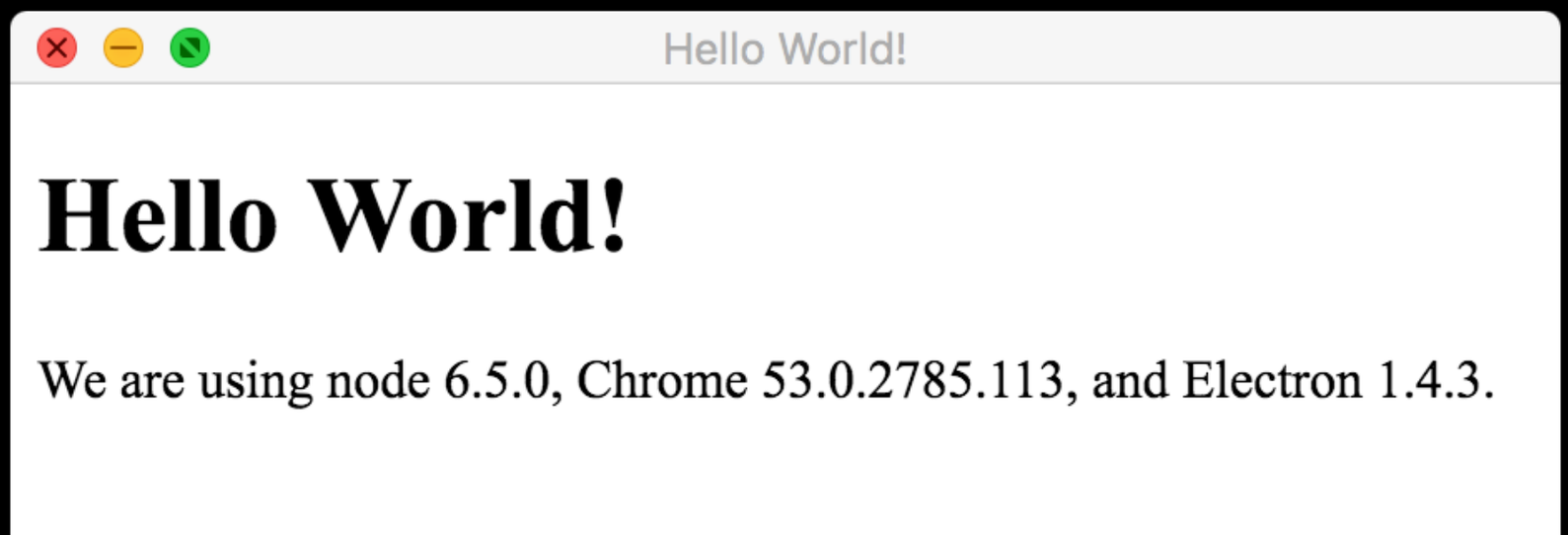
Node.js
for filesystems
and networks

Native APIs
for three
systems

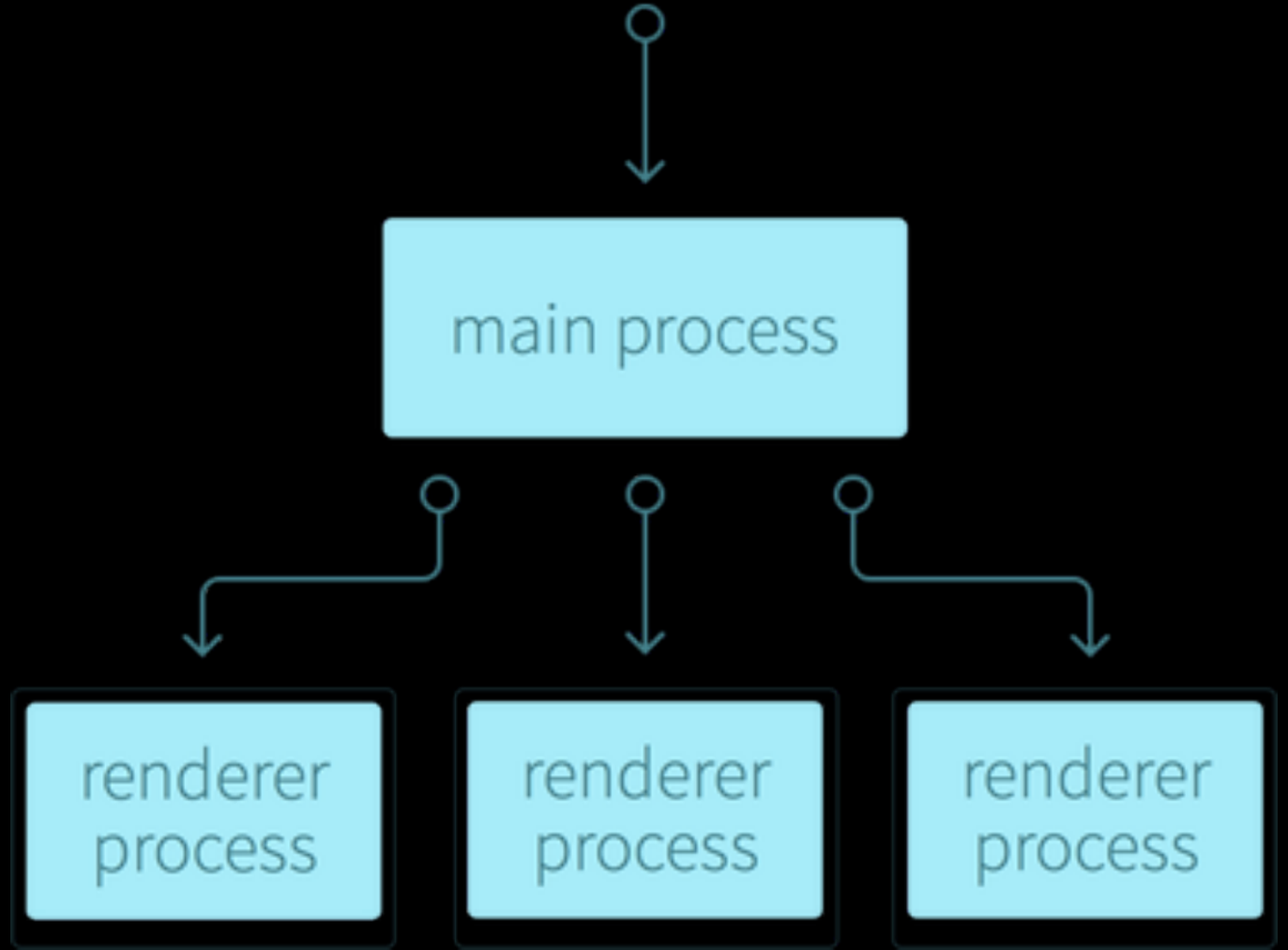
ELECTRON

```
const {app, BrowserWindow} = require('electron')

let mainWindow

app.on('ready', function() {
  mainWindow = new BrowserWindow({width: 800, height: 600})
  mainWindow.loadURL(`file://${__dirname}/index.html`)
})
```

```
<body>
  <h1>Hello World!</h1>
  We are using node <script>document.write(process.versions.node)</script>,
  Chrome <script>document.write(process.versions.chrome)</script>,
  and Electron <script>document.write(process.versions.electron)</script>.
</body>
```



Multi Process



Electron Apps

VS Code

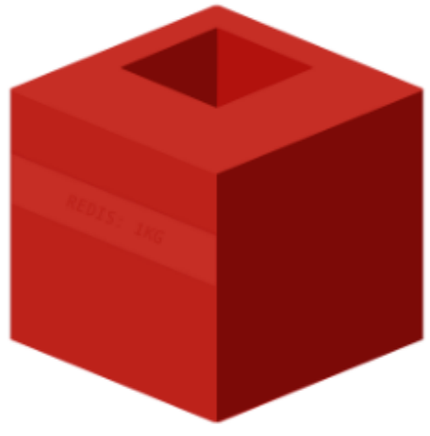
Slack

Postman

Hyper

Nylas N1

GitKraken



Medis

Mongotron

file-system-blob-store.js — /Users/jysperm/GitHub/atom

file-system-blob-store.js

```
1   'use strict'
2
3   const fs = require('fs-plus')
4   const path = require('path')
5
6   module.exports =
7   class FileSystemBlobStore {
8     static load (directory) {
9       let instance = new FileSystemBlobStore(dire
10      instance.load()
11      return instance
12    }
13
14    constructor (directory) {
15      this.blobFilename = path.join(directory, 'B
16      this.blobMapFilename = path.join(directory,
17      this.invalidationKeysFilename = path.join(d
18      this.lockFilename = path.join(directory, 'L
19      this.reset()
20    }
21
22    reset () {
23      this.inMemoryBlobs = new Map()
24      this.invalidationKeys = {}
```

div 791px × 126px

No Issues src/file-s LF UTF-8 JavaScript master 9 up

Elements Console Sources Network » 1 2

```
▼<div class="lines" style="height: 508.125px;
background-color: rgb(0, 0, 0);">
  ▼<div style="isolation: isolate; z-index: 0;">
    ▶<div style="position: absolute; display: block;
    z-index: 5; height: 126px; width: 791px;
    transform: translate3d(0px, 0px, 0px); background-
    color: rgb(0, 0, 0);">…</div>
    ▼<div style="position: absolute; display: block;
    z-index: 4; height: 126px; width: 791px;
    transform: translate3d(0px, 126px, 0px);
    background-color: rgb(0, 0, 0);">
      <div class="highlights"></div>
      ▼<div class="line" data-screen-row="6">
        ▼<span class="source js">
```

… div atom-text-editor #shadow-root div div div div div div div.line

Styles Event Listeners DOM Breakpoints Properties

Filter + .cls

```
element.style {
  position: absolute;
  display: block;
  z-index: 4;
  height: 126px;
  width: 791px;
  transform: translate3d(0px, 126px, 0px);
  background-color: rgb(0, 0, 0);
}
div {                          user agent stylesheet
  display: block;
}
```

Inherited from div.editor-contents--private

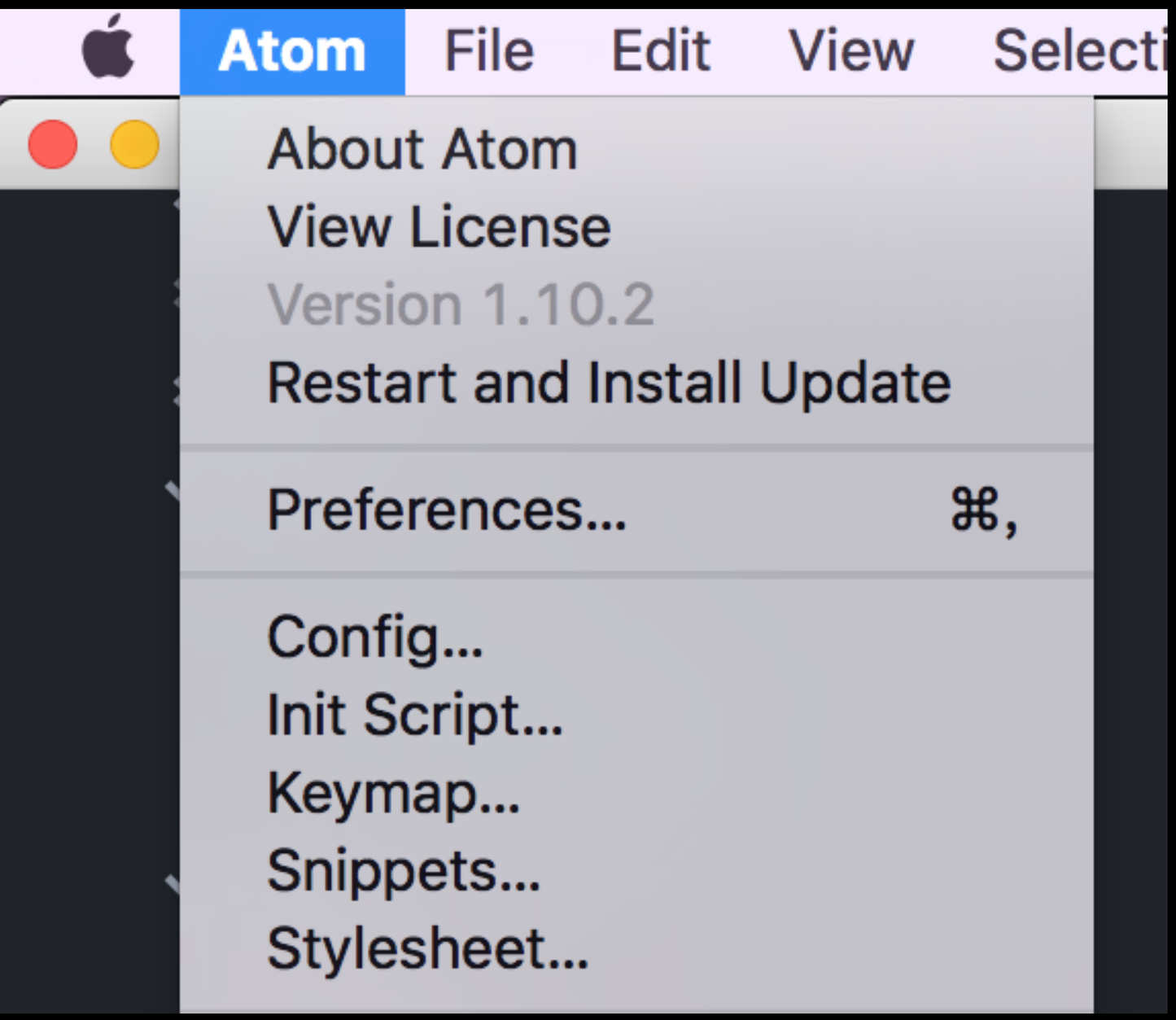
```
.platform-darwin atom-text- <style>…</style>
editor:not([mini])::shadow .editor-contents-
-private {
  cursor: -webkit-image-
    set(url('…
```

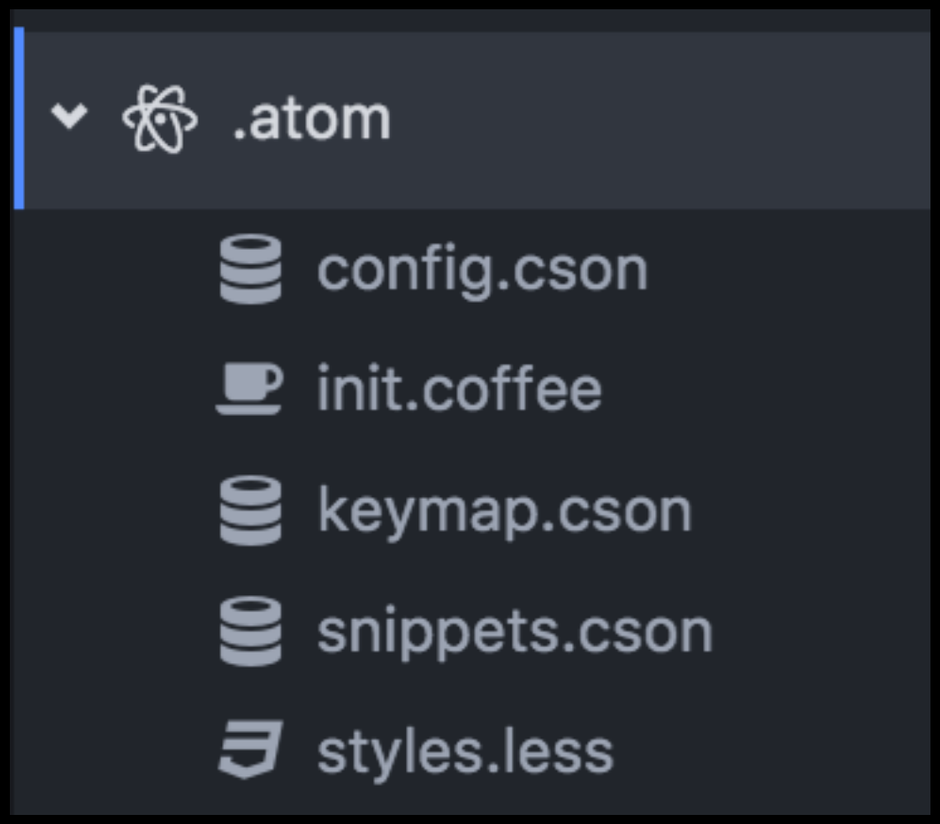
position -
margin -
border -
padding -
791 × 126

Filter Show all

background-color
rgb(0, 0, 0)
color
rgb(171, 178, 191)

Hack Atom


Atom
File
Edit
View
Selecti
About Atom
View License
Version 1.10.2
Restart and Install Update
Preferences... ⌘,
Config...
Init Script...
Keymap...
Snippets...
Stylesheet...


.atom
config.cson
init.coffee
keymap.cson
snippets.cson
styles.less

Stylesheet

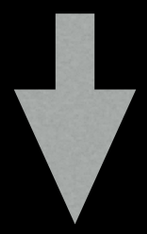
```
// To style other content in the text editor's shadow DOM,
// use the ::shadow expression
atom-text-editor::shadow .cursor {
  border-color: red;
}
```

```
### macOS

Download the latest [Atom re

Atom will automatically upda
```

Init Script

```
atom.commands.add('atom-text-editor', 'markdown:paste-as-link', () => {
  let selection = atom.workspace.getActiveTextEditor().getLastSelection()
  let clipboardText = atom.clipboard.read()

  selection.insertText(`[${selection.getText()}](${clipboardText})`)
})
```

Welcome to My Blog



Welcome to [My Blog](https://jysperm.me)

Keymap

```
'atom-workspace':
  'ctrl-l': 'markdown:paste-as-link'
  'ctrl-m ctrl-l': 'markdown:paste-as-link'
```

README.md — /Users/jysperm/GitHub/atom

ADME.md

md link

Markdown: Paste As **Link** ^L

```
docu

## In

### Prerequisites
- [Git](https://git-scm.com/)

### macOS

Download the latest [Atom
```

Packages

package.json

```
67    "packageDependencies": {
68      "atom-dark-syntax": "0.27.0",
69      "atom-dark-ui": "0.52.0",
70      // Themes ...
71      "about": "1.7.0",
72      "archive-view": "0.62.0",
73      "autocomplete-atom-api": "0.10.0",
74      "autocomplete-css": "0.13.1",
75      "autocomplete-html": "0.7.2",
76      "autocomplete-plus": "2.31.4",
77      "autocomplete-snippets": "1.11.0",
78      "autoflow": "0.27.0",
79      "autosave": "0.23.1",
80      "background-tips": "0.26.1",
```

text-editor.coffee — /Users/jysperm/GitHub/atom

tree-view

tabs

one-dark-ui

one-dark-syntax

language-coffee-script

git-diff

```
368     subscribeToBuffer: ->
369       @buffer.retain()
370
371       # disposables is a CompositeDis
372
373       @disposables.add @buffer.onDidChangePath =>
374         @emitter.emit 'did-change-title', @getTitle()
375         @emitter.emit 'did-change-path', @getPath()
376       @disposables.add @buffer.onDidChangeEncoding =>
377         @emitter.emit 'did-change-encoding', @getEncoding()
378       @disposables.add @buffer.onDidDestroy => @destroy()
379       @disposables.add @buffer.onDidChangeModified =>
380         @terminatePendingState() if not @hasTerminatedPendingSta
381
382       @preserveCursorPositionOnBufferReload()
383
```

```
scrollbar-corner-compo
special-token-symbols.
state-store.js
storage-folder.coffee
style-manager.coffee
styles-element.coffee
task-bootstrap.coff
task.coffee
text-editor-component.
text-editor-element.cof
text-editor-presenter.c
text-editor-registry.js
text-editor.coffee
text-utils.coffee
theme-manager.coffee
theme-package.coffee
tiled-component.coffee
title-bar.coffee
token-iterator.coffee
token.coffee
tokenized-buffer-iterat
```

572 results found for 'buffer'

find-and-replace

Finding with Options: Case Insensitive

buffer 572 found | Find | .* | Aa

Replace in current buffer | Replace | Replace All

No Issues src/text-editor.coffee 381:1 LF UTF-8 CoffeeScript master +3 10

status-bar

grammar-selector

diff

command-palette

Git Plus: **Diff**

Git Plus: **Diff** All

Remote Sync: **Diff** File

Remote Sync: **Diff** Folder

Git Plus: **Diff**tool

Git Diff: Toggle **Diff** List

Git Diff: Move To Next **Diff**

Git Diff: Move To Previous

text

fuzzy-finder

text-utils.coffee

src/**text**-utils.coffee

text-edit

Settings

settings-view

src/**text**-

text-utils

spec/**tex**

text-edit

src/**text**-

text-edit

static/**tex**

Core

Editor

Keybindings

Packages

Themes

Updates

Install

Open Config Folder

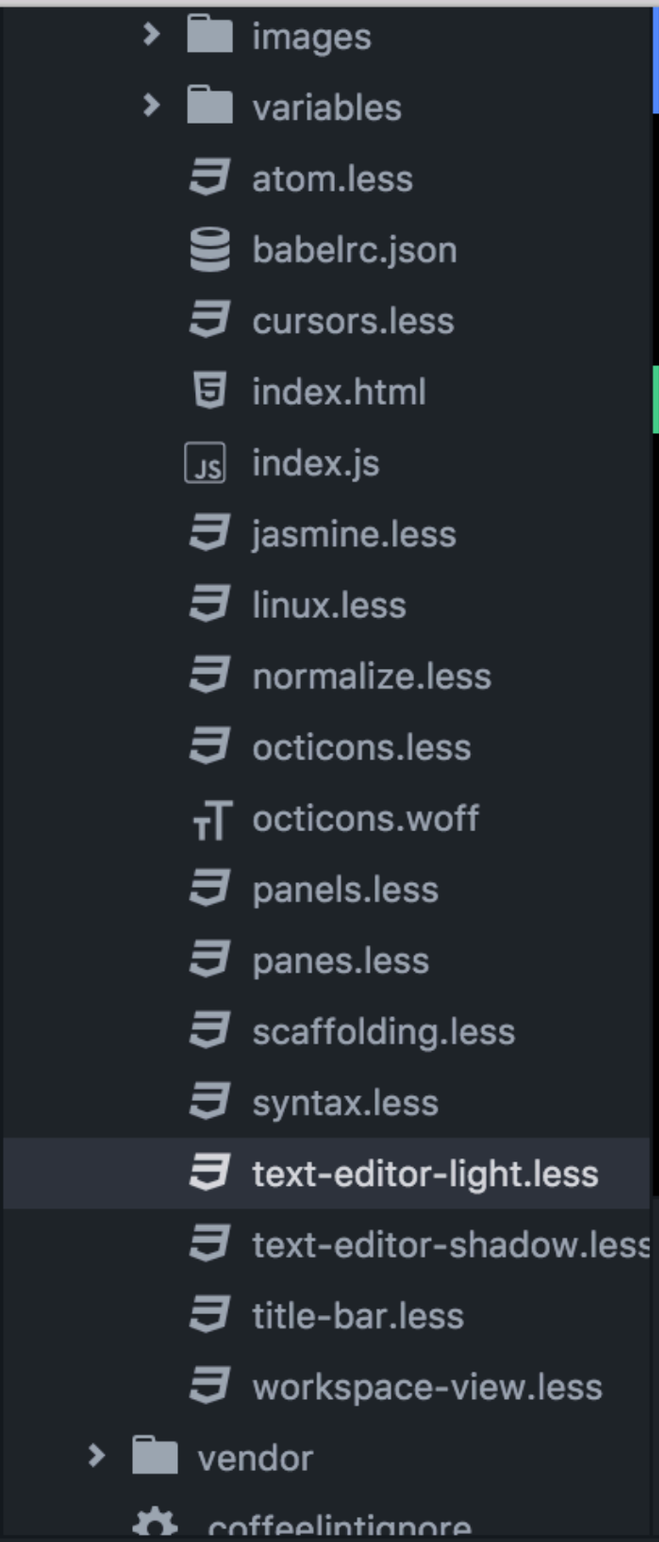
Keybindings

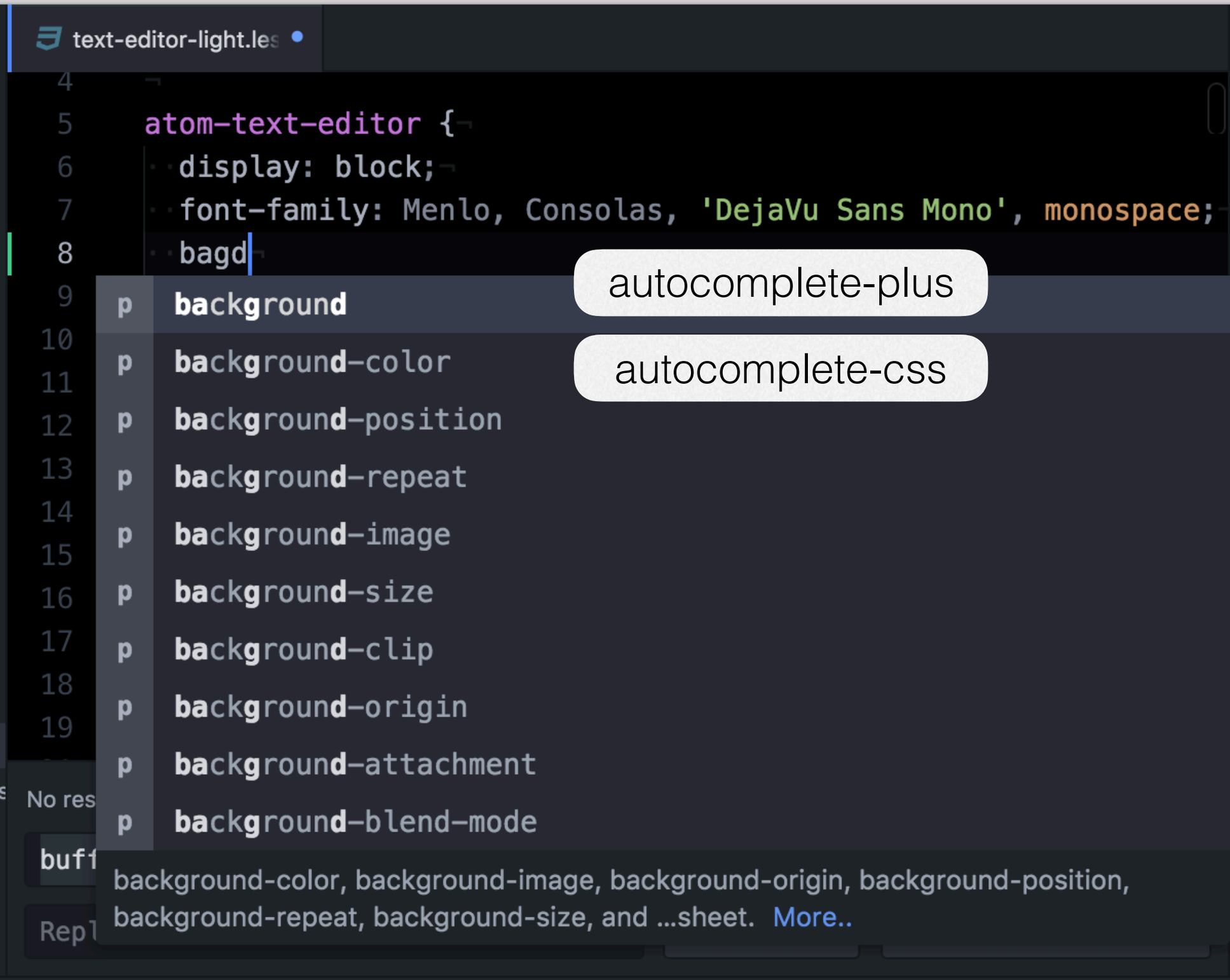
You can override these keybindings by copying and pasting them into your keymap file

Search keybindings

Keystroke	Command
a	tree-view:add-file
alt->	linter:next-error
alt-a	project-find:show-in-current-direct
alt-b	editor:move-to-beginning-of-word
alt-backspace	editor:delete-to-beginning-of-word

text-editor-light.less — /Users/jysperm/GitHub/atom





autocomplete-plus

autocomplete-css

Services API

```
// package.json of autocomplate-plus
"consumedServices": {
  "autocomplete.provider": {
    "versions": {
      "2.0.0": "consumeProvider_2_0",
      "3.0.0": "consumeProvider_3_0"
    }
  }
}

// package.json of autocomplate-css
"providedServices": {
  "autocomplete.provider": {
    "versions": {
      "2.0.0": "getProvider"
    }
  }
}
```

Connected together

```
"providedServices": {
  "status-bar": {
    "description": "A container for indicators
    at the bottom of the workspace",
    "versions": {
      "1.1.0": "provideStatusBar",
      "0.58.0": "legacyProvideStatusBar"
    }
  }
}
```

wakatime

settings-view

linter

go-to-line

line-ending-selector

about

highlight-selected

encoding-selector

grammar-selector

Provider API

Steel Brain edited this page on 28 Jan · 33 revisions

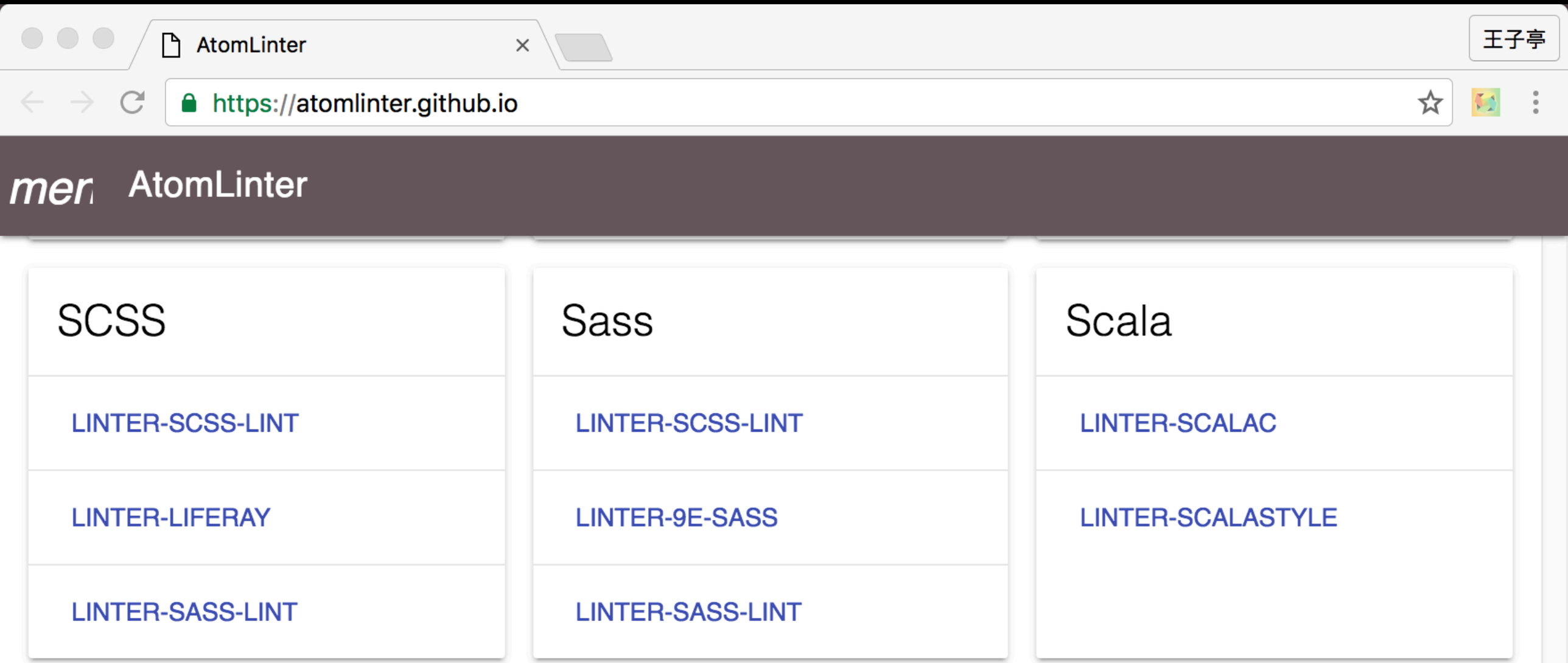
The Provider API allows you to make autocomplete+ awesome. The Atom community will ultimately judge the quality of Atom's autocomplete experience by the breadth and depth of the provider ecosystem. We're so excited that you're here reading about how to make Atom awesome.

> The following examples are in CoffeeScript. If you would like to add JavaScript examples, please feel free to edit this page!

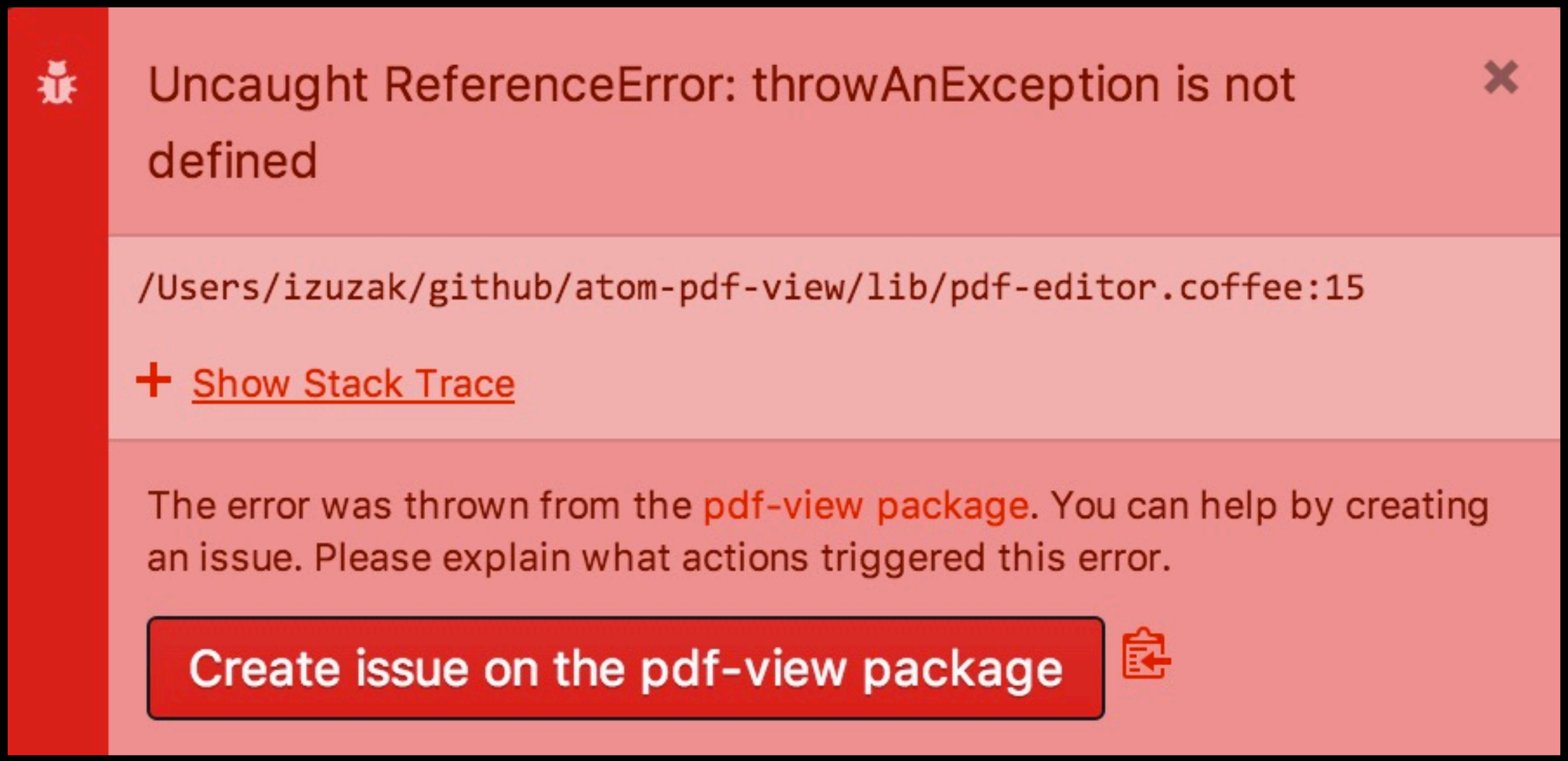
Defining A Provider

```
provider =
  # This will work on JavaScript and CoffeeScript files, but not in js comments.
  selector: '.source.js, .source.coffee'
  disableForSelector: '.source.js .comment'
```

Packages of Packages



Oops


Uncaught ReferenceError: throwAnException is not defined
/Users/izuzak/github/atom-pdf-view/lib/pdf-editor.coffee:15
Show Stack Trace
The error was thrown from the pdf-view package. You can help by creating an issue. Please explain what actions triggered this error.
Create issue on the pdf-view package

TextEditor ESSENTIAL

This class represents all essential editing state for a single `TextBuffer`, including cursor and selection positions, folds, and soft wraps. If you're manipulating the state of an editor, use this class. If you're interested in the visual appearance of editors, use `TextEditorElement` instead.

A single `TextBuffer` can belong to multiple editors. For example, if the same file is open in two different panes, Atom creates a separate editor for each pane. If the buffer is manipulated the changes are reflected in both editors, but each maintains its own cursor position, folded lines, etc.

Accessing TextEditor Instances

The easiest way to get hold of `TextEditor` objects is by registering a callback with `::observeTextEditors` on the `atom.workspace` global. Your callback will then be called with all current editor instances and also when any editor is created in the future.

```
atom.workspace.observeTextEditors (editor) ->
  editor.insertText('Hello World')
```

Registry

atom.commands | atom.grammars | atom.views

atom.keymaps | atom.packages | atom.deserializers

```
// register a command
atom.commands.add('atom-text-editor', 'markdown:paste-as-link', someAction)

// invoke a command
let target = atom.views.getView(atom.workspace.getActiveTextEditor())
atom.commands.dispatch(target, 'markdown:paste-as-link')
```

Bubbles upward

```
140
141
142   app
143
144
145
146
147
148
```

- app
- applications

Key Binding Resolver: tab

✔ autocomplete-plus:confirm	atom-text-editor.autocomplete-active	atom-text-editor.autocomplete-active
✔ snippets:next-tab-stop	atom-text-editor:not([mini])	/Applications/Atom Beta.app/Contents/Res
✔ snippets:expand	atom-text-editor:not([mini])	/Applications/Atom Beta.app/Contents/Res
✔ editor:indent	atom-text-editor:not([mini])	/Applications/Atom Beta.app/Contents/Res
✖ core:focus-next	body .native-key-bindings	/Applications/Atom Beta.app/Contents/Res
✖ native!	.browser-plus webview	/Users/jysperm/.atom/packages/browser-p
✖ find-and-replace:focus-next	.find-and-replace, .project-find, .project-find .results-view	/Applications/Atom Beta.app/Contents/Res
✖ core:confirm	.corrections atom-text-editor[mini]	/Applications/Atom Beta.app/Contents/Res

Workspace

Settings — /Users/jysperm/.atom

.atom
.apm
.node-gyp
blob-store
compile-cache
local-history
packages
recovery
storage
.apmrc
.gitignore
config.cson
init.coffee
keymap.cson
snippets.cson
styles.less
environments
electron-demo
.DS_Store
index.html

Left Panel

init.coffee

```
4   # after packages are loaded/
5   # has been restored.
6   #
7   # An example hack to log to
8   #
9   # atom.workspace.observeText
10  #   editor.onDidSave ->
11  #     console.log "Saved! #{
12
13  atom.commands.add 'atom-text
14    selection = atom.workspace
15    clipboardText = atom.clipb
16
17    selection.insertText("[#{s
18
```

Pane

(TextEditor)

Settings

Core
Editor
Keybindings
Packages
Themes
Updates
Install
Open Config Folder

Core Setti

These are Atom's c
behavior unrelated to te
packages may have the
found within their packa
list.

Allow Pending F
items to be p
them to a pane per
single clicking files

Audio Beep
Trigger the system
certain actions can
are no results.

Pane

(SettingsView)

Find in Current Buffer Close this panel with the esc key Finding with Options: Case Insensitive

Find in current buffer no results Find .* Aa

Replace in current buffer Replace Replace All

Bottom Panel

Bottom Panel

Settings Highlighted: 1 10 updates

Workspace ESSENTIAL

Event Subscription

::observeTextEditors(callback)

::observePaneItems(callback)

::onDidChangeActivePaneItem(callback)

::onDidStopChangingActivePaneItem(callback)

::observeActivePaneItem(callback)

::onDidOpen(callback)

...

Panels

::getBottomPanels()

::addBottomPanel(options)

::getLeftPanels()

::addLeftPanel(options)

::getRightPanels()

TextEditor ESSENTIAL

Event Subscription

::onDidChangeTitle(callback)

::onDidChangePath(callback)

::onDidChange(callback)

::onDidStopChanging(callback)

::onDidSave(callback)

::onDidDestroy(callback)

...

Selections

::getSelectedText()

::getSelectedBufferRange()

::getSelectedBufferRanges()

::setSelectedBufferRange(bufferRange, [options])

::setSelectedBufferRanges(bufferRanges, [options])

Emitter & Disposable

```
class SomePackage {
  activate() {
    this.disposable = workspace.observeTextEditors( () => {
      console.log('found a TextEditor')
    })
  }

  deactivate() {
    this.disposable.dispose()
  }
}
```

And more API

atom.config	Color	Point
atom.clipboard	Cursor	Selection
atom.project	File	Task
atom.notifications	Directory	GitRepository

Why not VS Code

Improve startup time

- Lazy load packages

```
{
  "name": "markdown-link",
  "activationCommands": {
    "atom-text-editor": "markdown:paste-as-link"
  }
}
```

- Timecop: find the slowest package

Startup Time

Window load time **430ms**
Shell load time **595ms**
Workspace load time **177ms**
Project load time **141ms**
Window state load time **318ms**

Compile Cache

CoffeeScript files compiled **0**
Babel files compiled **0**
Typescript files compiled **0**
CSON files compiled **4**
Less files compiled **0**

Package Loading

Loaded 94 packages in 106ms. 3 packages took longer than 5ms to load.

atom-typescript **13ms**
browser-plus **13ms**
file-icons **11ms**

Package Activation

Activated 85 packages in 1377ms. 17 packages took longer than 5ms to activate.

atom-typescript **575ms**
tree-view **214ms**
wakatime **189ms**
remote-sync **60ms**
linter **58ms**
linter-eslint **47ms**

Theme Loading

Loaded 13 themes in 0ms. 0 themes took longer than 5ms to load.

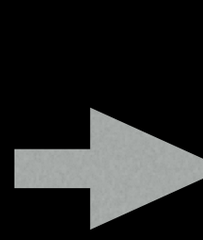
Theme Activation

Activated 2 themes in 9ms. 1 theme took longer than 5ms to activate.

one-dark-ui **6ms**

- Packaging dependencies to single file

app (include node_modules)
12068 files



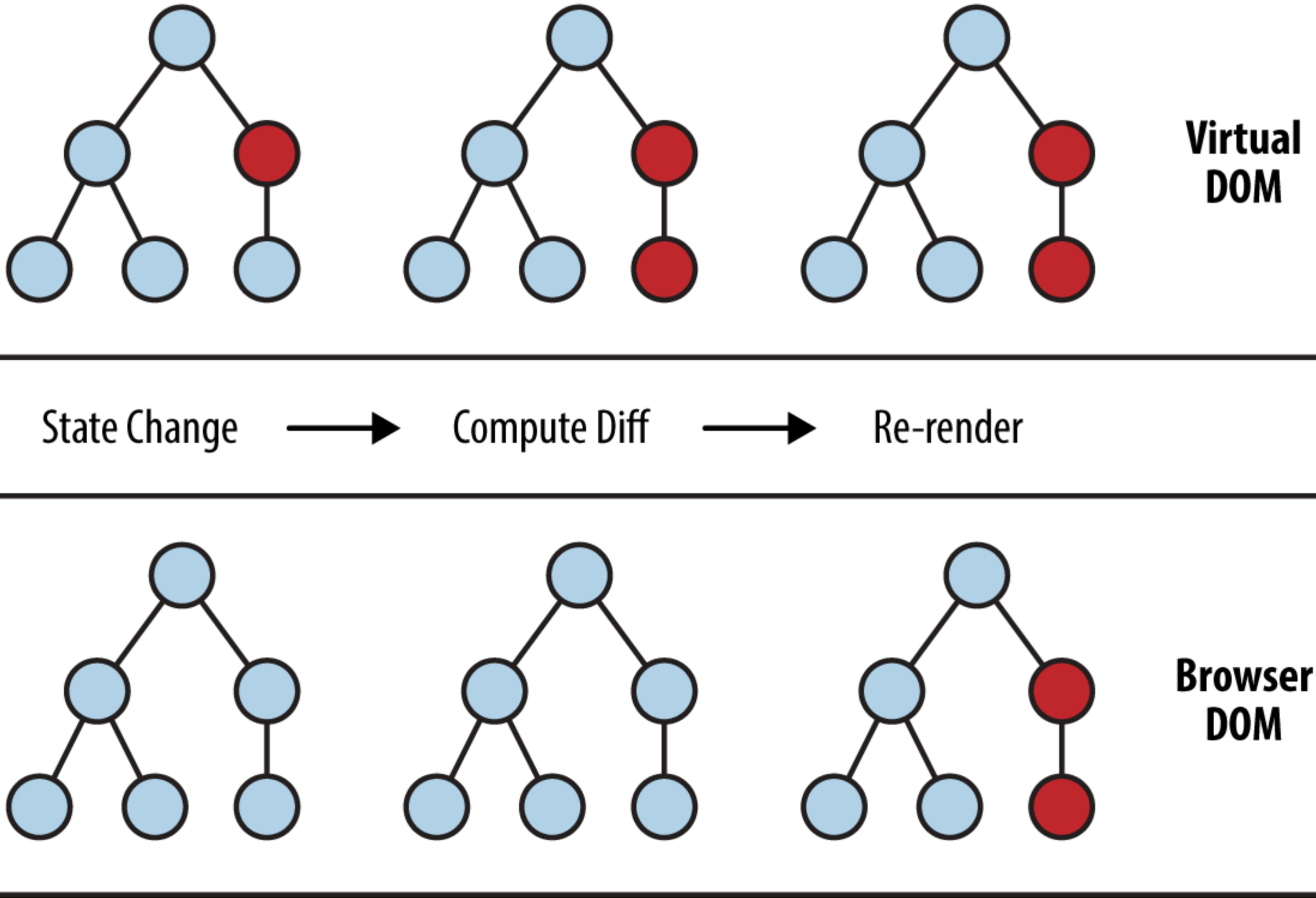
app.asar
single file

Rendering performance

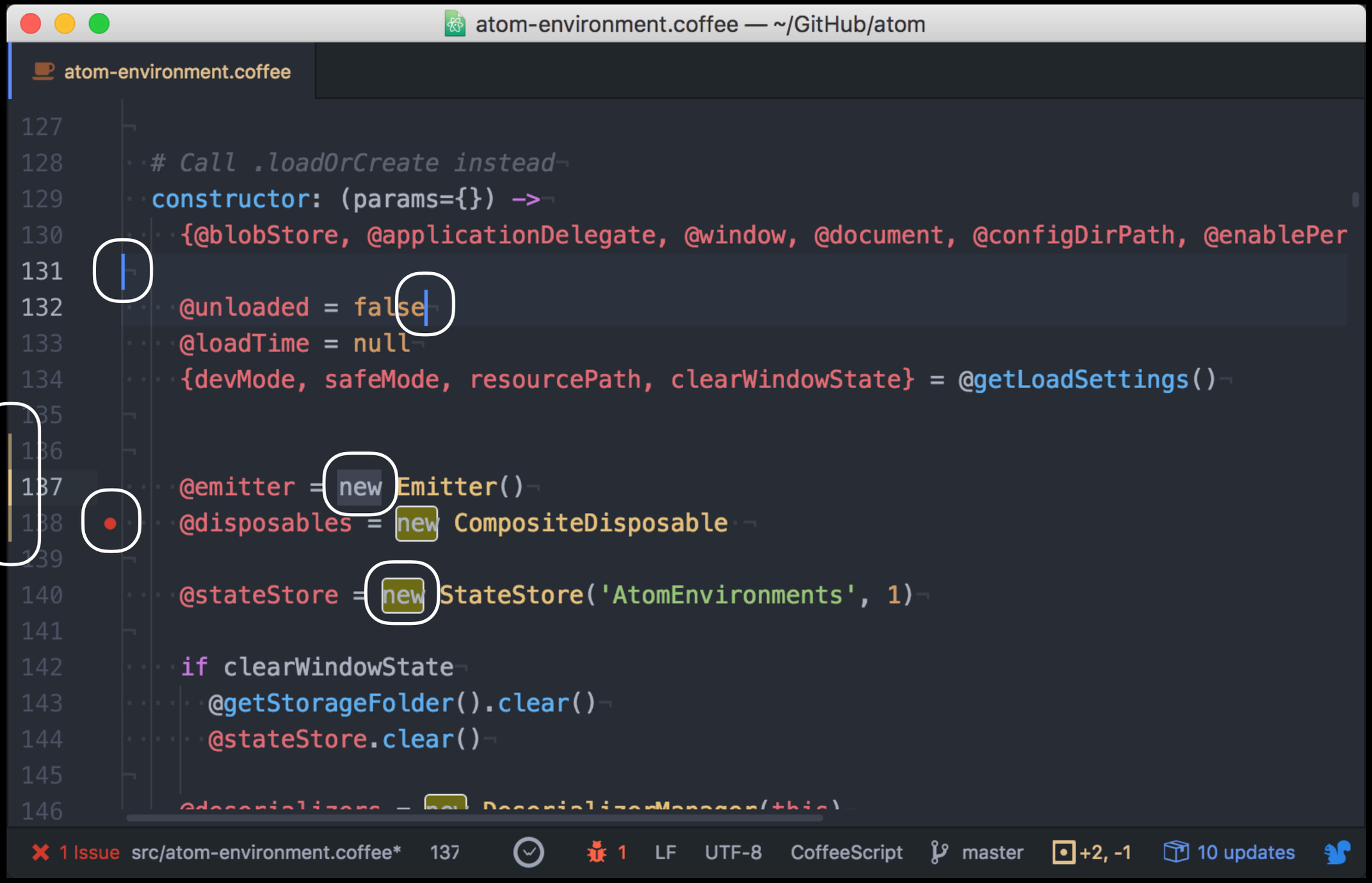
- Avoiding manipulate DOM directly
- Keep the DOM as small as possible

Performance Killer

Reflow & Repaint


Virtual
DOM
State Change
Compute Diff
Re-render
Browser
DOM

Marker


atom-environment.coffee — ~/GitHub/atom
atom-environment.coffee
127
128 # Call .loadOrCreate instead
129 constructor: (params={}) ->
130 {@blobStore, @applicationDelegate, @window, @document, @configDirPath, @enablePer
131
132 @unloaded = false
133 @loadTime = null
134 {devMode, safeMode, resourcePath, clearWindowState} = @getLoadSettings()
135
136
137 @emitter = new Emitter()
138 @disposables = new CompositeDisposable
139
140 @stateStore = new StateStore('AtomEnvironments', 1)
141
142 if clearWindowState
143 @getStorageFolder().clear()
144 @stateStore.clear()
145
146
1 Issue src/atom-environment.coffee* 137 1 LF UTF-8 CoffeeScript master +2, -1 10 updates

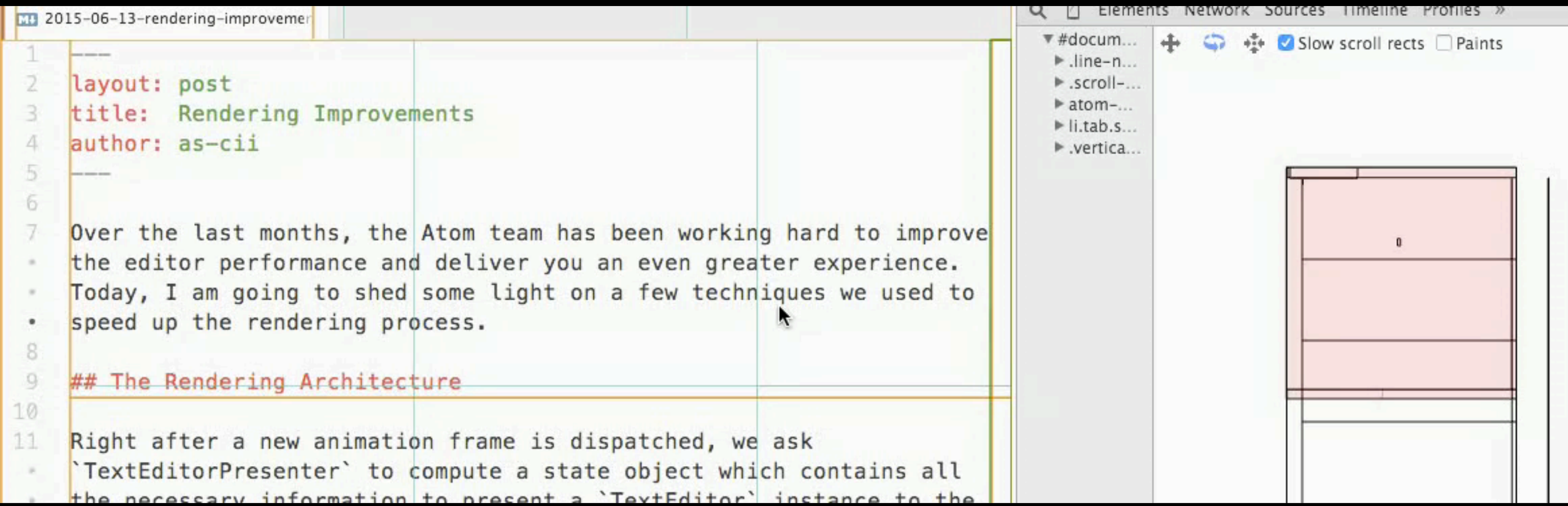
Decoration

```
let range = editor.getSelectedBufferRange()
// never, surround, overlap, inside, touch
let marker = editor.markBufferRange(range, {invalidate: 'overlap'})
// line, line-number, highlight, overlay, gutter, block
editor.decorateMarker(marker, {type: 'highlight'}, {class: 'highlight-selected'})
```

```
.highlights {
  .highlight-selected .region {
    border-radius: 3px;
    box-sizing: border-box;
    background-color: transparent;
    border-width: 1px;
    border-style: solid;
  }

  .highlight-selected .region {
    border-color: #ddd;
  }

  .highlight-selected.background .region {
```

rendering only the visible lines

- transform3d



Background Task

- Runs in separate process (like child_process)

```coffee
# main process

scan = (regex, options={}, iterator) ->
  deferred = Q.defer()

  task = Task.once require.resolve('./scan-handler'), regex, options, ->
    deferred.resolve()

  task.on 'scan:result-found', (result) =>
    iterator(result)

  deferred.promise
```

Some packages

- git-plus
- file-icons
- local-history
- highlight-selected
- linter (linter-eslint)
- atom-ternjs / atom-typescript / react

More

- https://github.com/atom/atom
- https://atom.io/docs
- http://electron.atom.io/docs/
- https://discuss.atom.io/
- http://blog.atom.io/
- https://atom-china.org/
- https://github.com/atom-china

Q & A

- Node.js
- Docker
- LeanCloud
- Bitcoin
- Atom