

Configuring the Text Editor Atom

Stefano Balietti

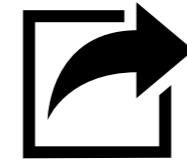
Center for European Social Science Research at Mannheim University (MZES)
Alfred-Weber Institute of Economics at Heidelberg University

@balietti | stefanobalietti.com | @nodegameorg | nodegame.org

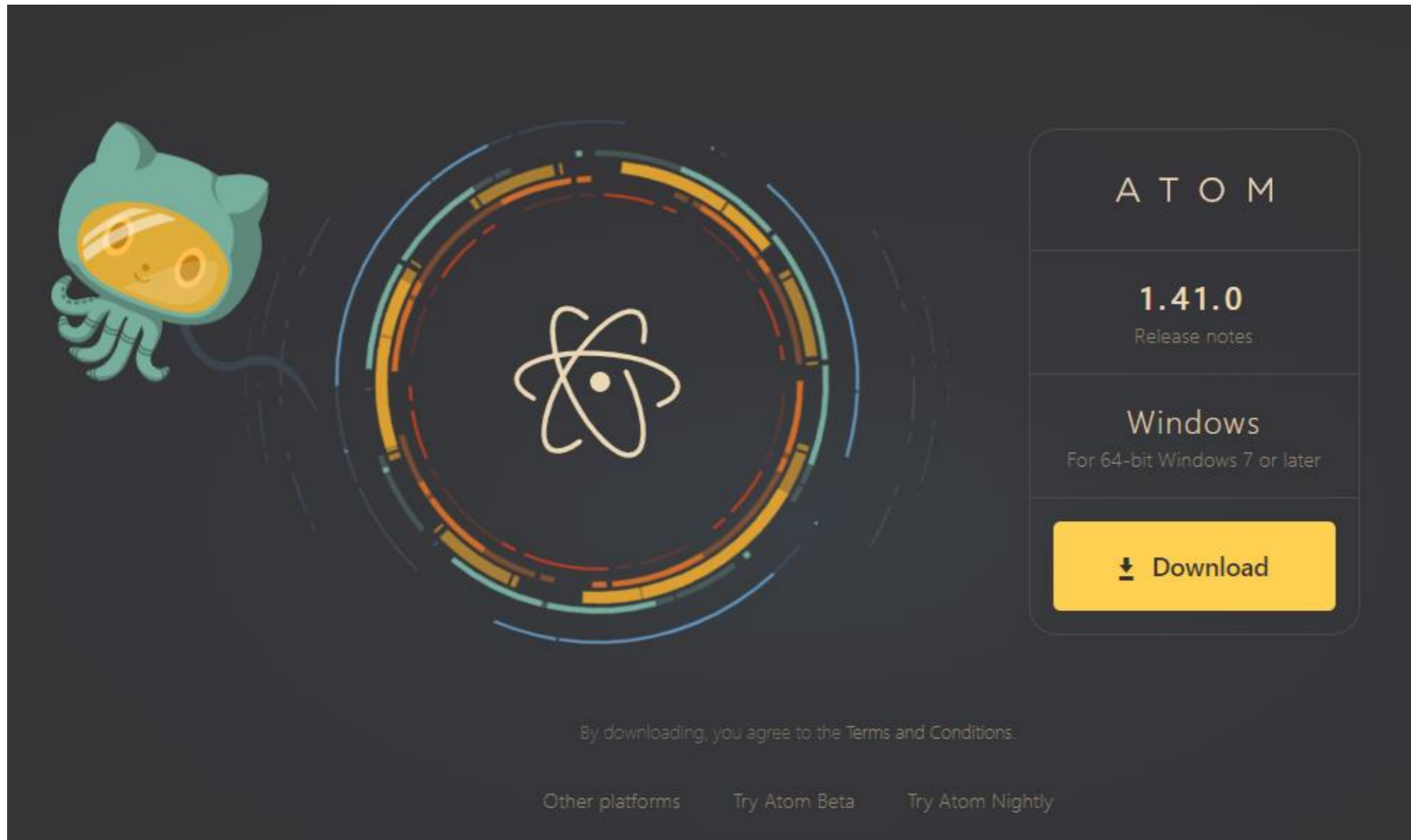


Building Digital Skills: 12-13 March 2020, University of Luzern

ATOM



[Atom.io](https://atom.io)




The image shows a dark-themed download page for the Atom text editor. On the left, there is a large graphic featuring the GitHub Octocat mascot on the left and a stylized atomic symbol in the center, surrounded by concentric circles of light. On the right, a white-bordered box contains the following information: the word "ATOM" in spaced-out letters, the version number "1.41.0" with a link to "Release notes", the operating system "Windows" with the note "For 64-bit Windows 7 or later", and a prominent yellow "Download" button with a download icon. At the bottom of the page, there is a line of text: "By downloading, you agree to the Terms and Conditions." and three links: "Other platforms", "Try Atom Beta", and "Try Atom Nightly".

ATOM

1.41.0
[Release notes](#)

Windows
For 64-bit Windows 7 or later

 Download

By downloading, you agree to the [Terms and Conditions](#).

[Other platforms](#) [Try Atom Beta](#) [Try Atom Nightly](#)

Indentation

Indentation is quality nr. 1 of quality coding

Some programming language enforce to be valid code (e.g., Python)

JavaScript requires the *discipline of the programmer and* proper use of the parentheses

Indentation

Indentation is quality nr. 1 of quality coding

Some programming language enforce to be valid code (e.g., Python)

JavaScript requires the *discipline of the programmer and* proper use of the parentheses

Not-Properly Indented

```
let a =          1234          ;
    a++;
                if ( a > 123 ) {
a--;
                }

else a++
    console.log(a);
```

Indentation

Indentation is key to write high-quality code

Some programming languages enforce it to be valid code (e.g., Python)

JavaScript requires the *discipline of the programmer and* proper use of the parentheses

Not-Properly Indented

```
let a=          1234          ;
    a++;
                if ( a > 123 ) {
a--;
                }

else a++
    console.log(a);
```

Properly Indented

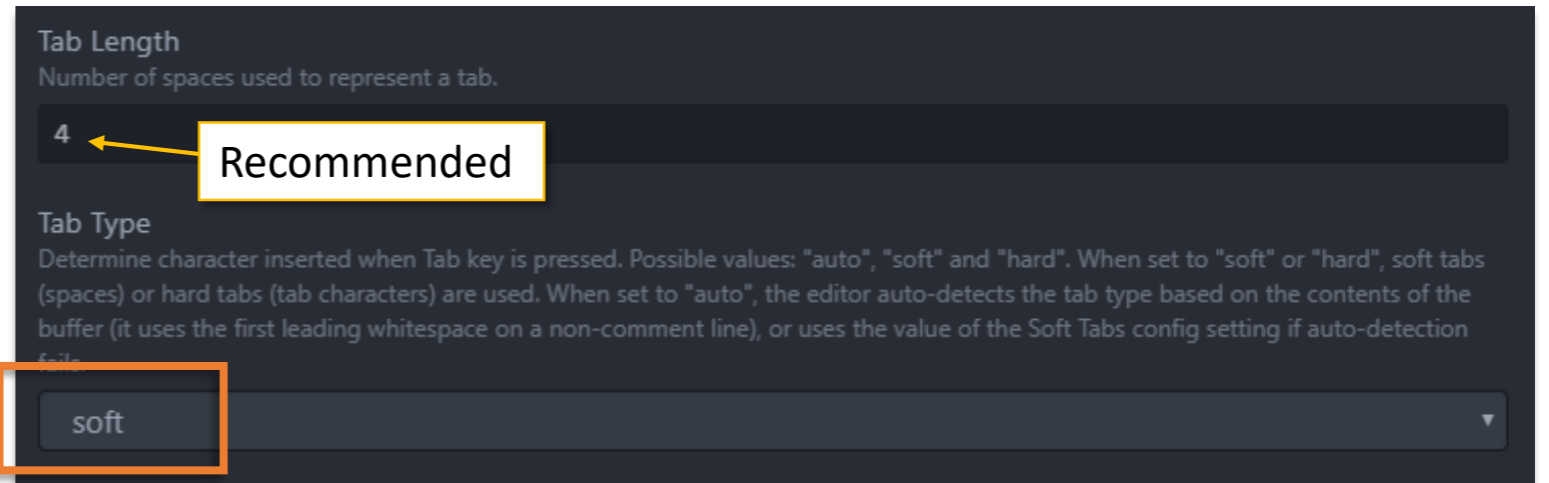
```
let a = 1234;
a++;
if ( a > 123 ) {
    a--;
}
else {
    a++;
}
console.log(a);
```

More Readable
Easier to catch errors

ATOM Editor Settings

Open the Settings Panel
from the menu **File/Settings**

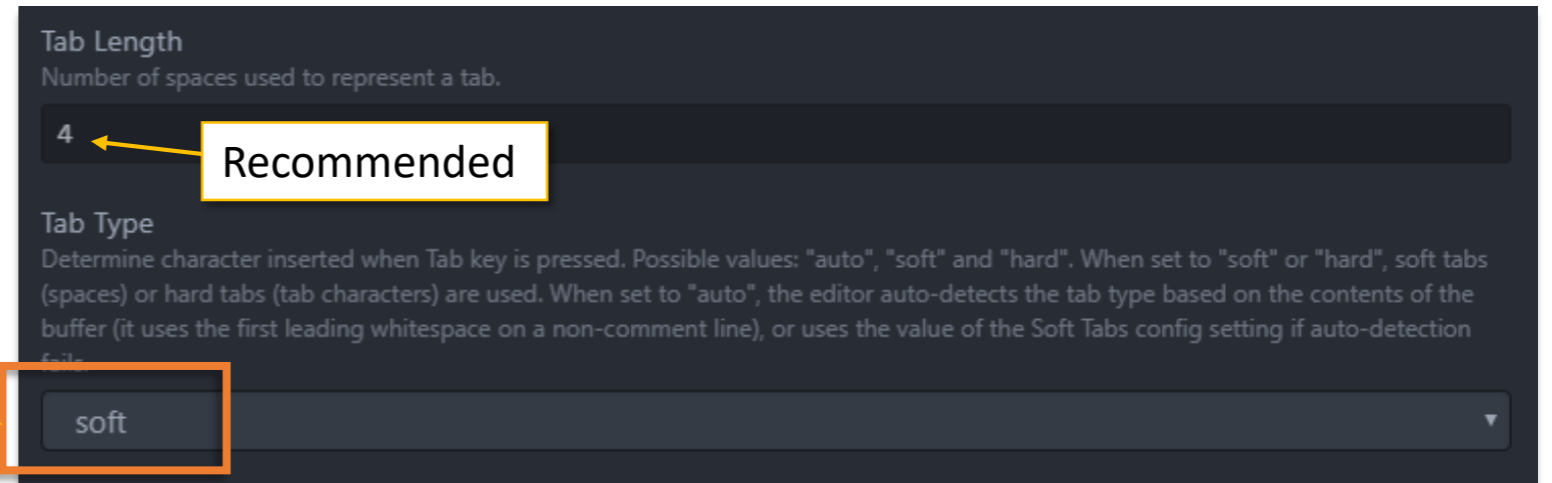
Make sure all your TABs are
automatically converted into spaces.



ATOM Editor Settings

Open the Settings Panel
from the menu **File/Settings**

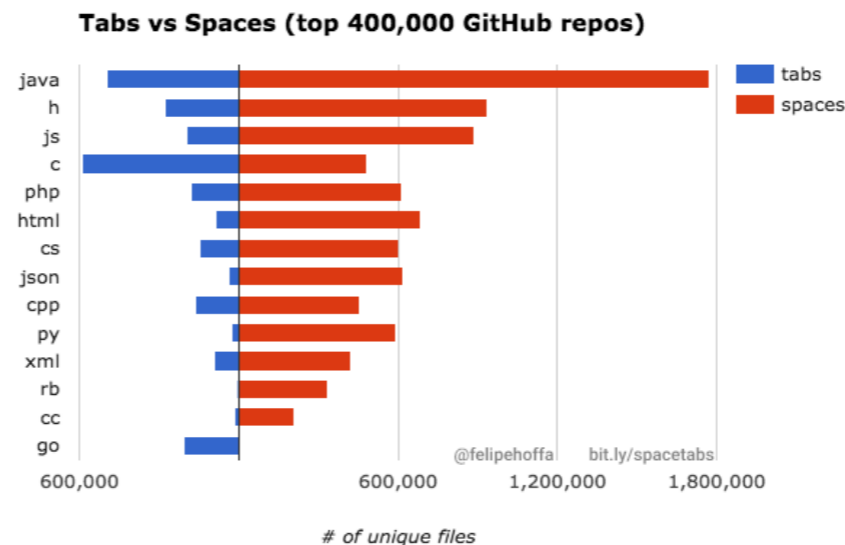
Make sure all your TABs are
automatically converted into spaces.



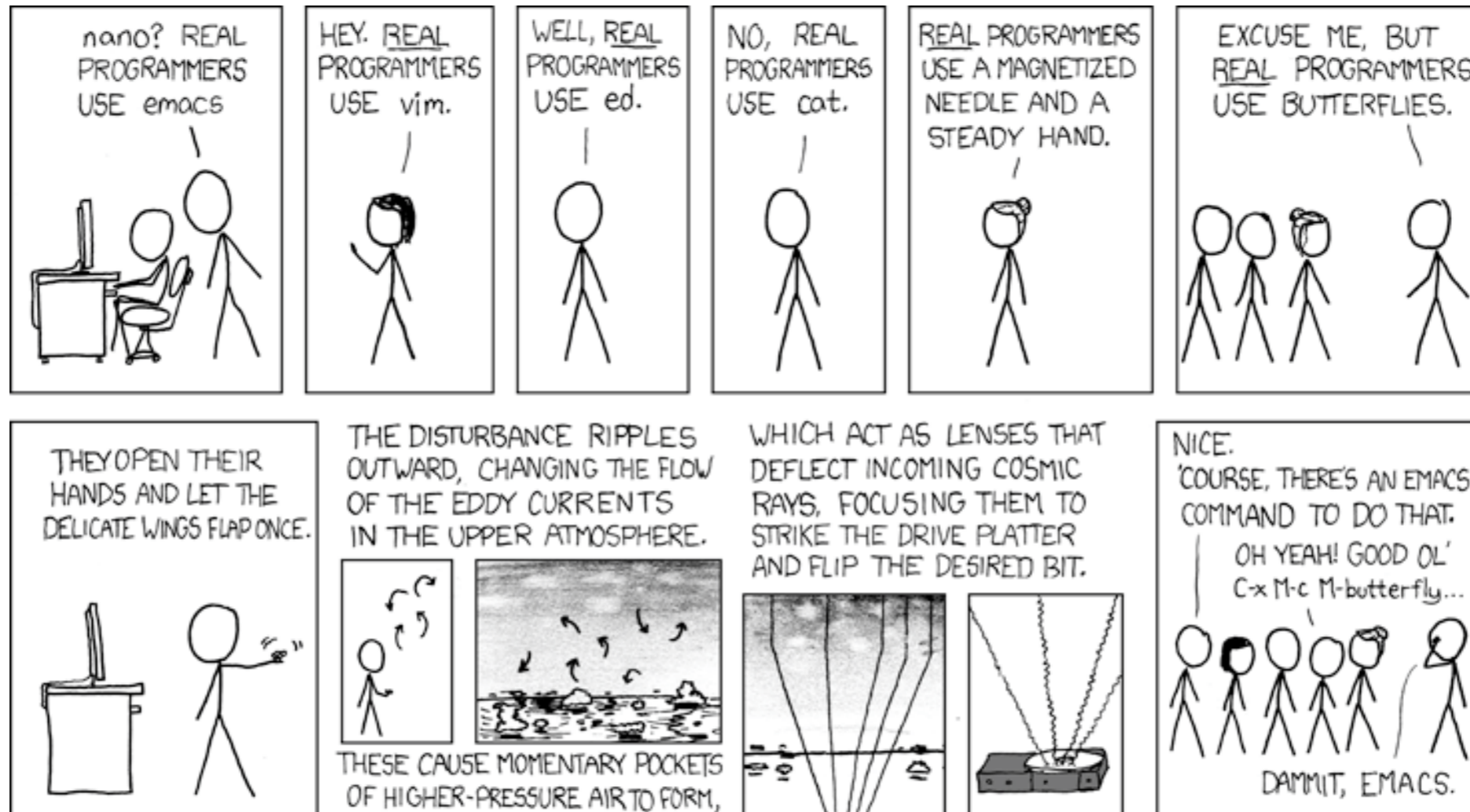
Why? Because TABs are evil!



Welcome to the
Spaces vs. Tabs Debate



The Text Editor War



ATOM Editor Settings

Open the Settings Panel
from the menu **File/Settings**

Make sure all your TABs are
automatically converted into spaces.

The best of the two worlds.

The screenshot shows the ATOM Editor Settings panel. The 'Tab Length' section is set to 4, with a yellow box labeled 'Recommended' pointing to it. The 'Tab Type' dropdown is set to 'soft', highlighted with an orange box and a yellow arrow pointing from the text 'Make sure all your TABs are automatically converted into spaces.' Below the 'Editor Settings' section, the 'Atomic Soft Tabs' checkbox is checked and highlighted with an orange box, with a yellow arrow pointing from the text 'The best of the two worlds.' Other checked settings include 'Auto Indent', 'Auto Indent On Paste', and 'Confirm Checkout HEAD Revision'.

Tab Length
Number of spaces used to represent a tab.

4 ← Recommended

Tab Type
Determine character inserted when Tab key is pressed. Possible values: "auto", "soft" and "hard". When set to "soft" or "hard", soft tabs (spaces) or hard tabs (tab characters) are used. When set to "auto", the editor auto-detects the tab type based on the contents of the buffer (it uses the first leading whitespace on a non-comment line), or uses the value of the Soft Tabs config setting if auto-detection fails.

soft

Editor Settings

These settings are related to text editing. Some of these can be overridden on a per-language basis. Check language settings by clicking its package card in the [Packages list](#).

- Atomic Soft Tabs
Skip over tab-length runs of leading whitespace when moving the cursor.
- Auto Indent
Automatically indent the cursor when inserting a newline.
- Auto Indent On Paste
Automatically indent pasted text based on the indentation of the previous line.
- Confirm Checkout HEAD Revision
Show confirmation dialog when checking out the HEAD revision and discarding changes to current file since last commit.

ATOM Editor Settings

All other options on this slide strongly recommended

Open the Settings Panel from the menu **File/Settings**

Make sure all your TABs are automatically converted into spaces.

The best of the two worlds.

- Show Indent Guide
Show indentation indicators in the editor.
- Show Invisibles
Render placeholders for invisible characters, such as tabs, spaces and newlines.
- Show Line Numbers
Show line numbers in the editor's gutter.

The screenshot shows the ATOM Editor Settings panel. The 'Tab Length' setting is set to 4, with a yellow box labeled 'Recommended' pointing to it. The 'Tab Type' dropdown is set to 'soft', with an orange box around it and a yellow arrow pointing from the text 'Make sure all your TABs are automatically converted into spaces.' Below the 'Editor Settings' section, the 'Atomic Soft Tabs' checkbox is checked, with an orange box around it and a yellow arrow pointing from the text 'The best of the two worlds.' Other checked settings include 'Show Indent Guide', 'Show Invisibles', 'Show Line Numbers', 'Auto Indent', 'Auto Indent On Paste', and 'Confirm Checkout HEAD Revision'.

Tab Length
Number of spaces used to represent a tab.

4 ← Recommended

Tab Type
Determine character inserted when Tab key is pressed. Possible values: "auto", "soft" and "hard". When set to "soft" or "hard", soft tabs (spaces) or hard tabs (tab characters) are used. When set to "auto", the editor auto-detects the tab type based on the contents of the buffer (it uses the first leading whitespace on a non-comment line), or uses the value of the Soft Tabs config setting if auto-detection fails.

soft

Editor Settings

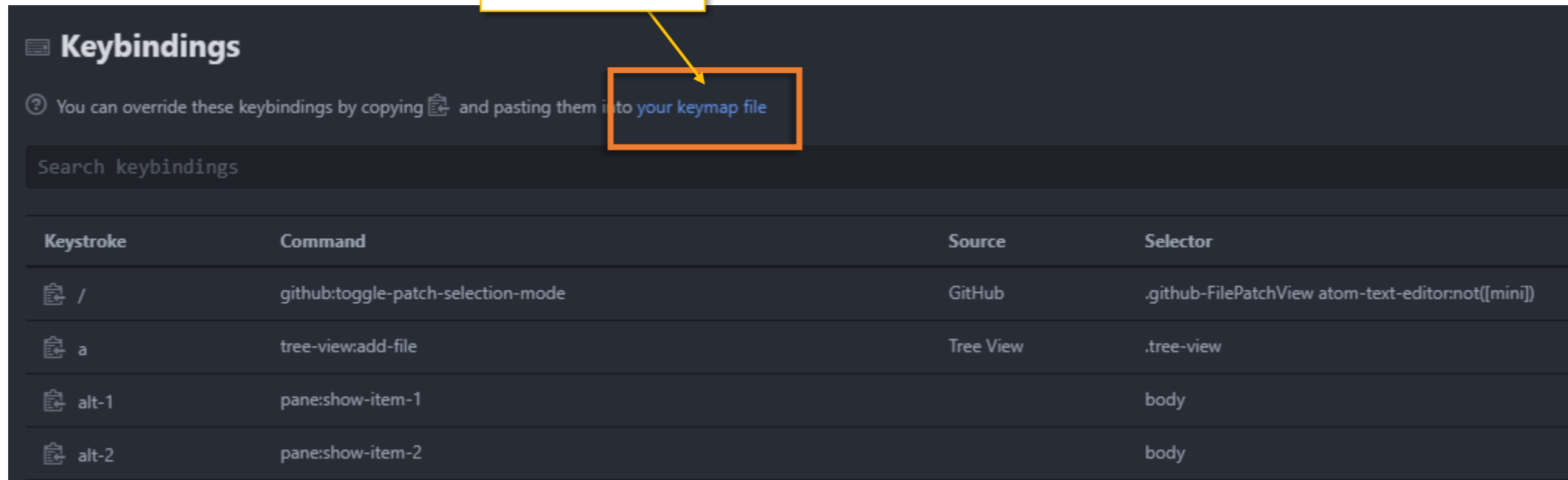
These settings are related to text editing. Some of these can be overridden on a per-language basis. Check language settings by clicking its package card in the [Packages list](#).

- Atomic Soft Tabs
Skip over tab-length runs of leading whitespace when moving the cursor.
- Auto Indent
Automatically indent the cursor when inserting a newline.
- Auto Indent On Paste
Automatically indent pasted text based on the indentation of the previous line.
- Confirm Checkout HEAD Revision
Show confirmation dialog when checking out the HEAD revision and discarding changes to current file since last commit.


ATOM Keybindings

Keybindings are shortcuts to commands used often. You can customize ATOM the way you like, but one shortcut is a **MUST**. Which one?





Click here



Keybindings

ⓘ You can override these keybindings by copying  and pasting them into [to your keymap file](#)

Search keybindings

Keystroke	Command	Source	Selector
 /	github:toggle-patch-selection-mode	GitHub	.github-FilePatchView atom-text-editor:not([mini])
 a	tree-view:add-file	Tree View	.tree-view
 alt-1	pane:show-item-1		body
 alt-2	pane:show-item-2		body

ATOM Keybindings

Set TAB to auto-indent.

Select the text you want to indent, press TAB, enjoy properly indented code.

Correct and consistent **indentation** is the key to write high-quality code.

'atom-text-editor:not([mini]):

'tab': 'editor:auto-indent'

```
1 # Your keymap
2 #
3 # Atom keymaps work similarly to style sheets. Just as style sheets use
4 # selectors to apply styles to elements, Atom keymaps use selectors to associate
5 # keystrokes with events in specific contexts. Unlike style sheets however,
6 # each selector can only be declared once.
7 #
8 # You can create a new keybinding in this file by typing "key" and then hitting
9 # tab.
10 #
11 # Here's an example taken from Atom's built-in keymap:
12 #
13 # 'atom-text-editor':
14 #   'enter': 'editor:newLine'
15 #
16 # 'atom-workspace':
17 #   'ctrl-shift-p': 'core:move-up'
18 #   'ctrl-p': 'core:move-down'
19 #
20 # You can find more information about keymaps in these guides:
21 # * http://flight-manual.atom.io/using-atom/sections/basic-customization/#customizing-keybindings
22 # * http://flight-manual.atom.io/behind-atom/sections/keymaps-in-depth/
23 #
24 # If you're having trouble with your keybindings not working, try the
25 # Keybinding Resolver: `Cmd+.` on macOS and `Ctrl+.` on other platforms. See the
26 # Debugging Guide for more information:
27 # * http://flight-manual.atom.io/hacking-atom/sections/debugging/#check-the-keybindings
28 #
29 # This file uses CoffeeScript Object Notation (CSON).
30 # If you are unfamiliar with CSON, you can read more about it in the
31 # Atom Flight Manual:
32 # http://flight-manual.atom.io/using-atom/sections/basic-customization/#configuring-with-cson
33 'atom-text-editor:not([mini]):
34   'tab': 'editor:auto-indent'
35
```

ATOM Packages

Packages extend ATOM's basic functionalities.

You can customize ATOM the way you like, but ~~one~~ 3 packages ~~is~~ are a **MUST**. Which ones?

ATOM Linters

+ **Install Packages**

🔍 Packages are published to atom.io and are installed to `C:\Users\balistef\atom\packages`

linter Packages Themes

linter 2.3.1 7,185,732

A Base Linter with Cow Powers

steelbrain

Settings Uninstall Disable

Let's face it. Coding isn't easy...
Wouldn't it be wonderful to
have someone telling you
spotting your mistakes for you? 🤔

You need a **Linter!** (and all 3 of
those packages)

📦 **Installed Packages** 3/90

linter

Community Packages 3/9

linter 2.3.1 7,185,732

A Base Linter with Cow Powers

steelbrain

Settings Uninstall Disable

linter-eslint 8.5.5 1,772,098

Lint JavaScript on the fly, using ESLint

AtomLinter

Settings Uninstall Disable

linter-ui-default 1.8.1 4,689,920

Default UI for the Linter package

steelbrain

Settings Uninstall Disable

ATOM Linters

Red dot next to the line containing an error or warning.

Explanation for the error in the tooltip and in the bottom panel.

But wait...only if you added a `.eslintrc.js` file in your project.

If needed, get one here:

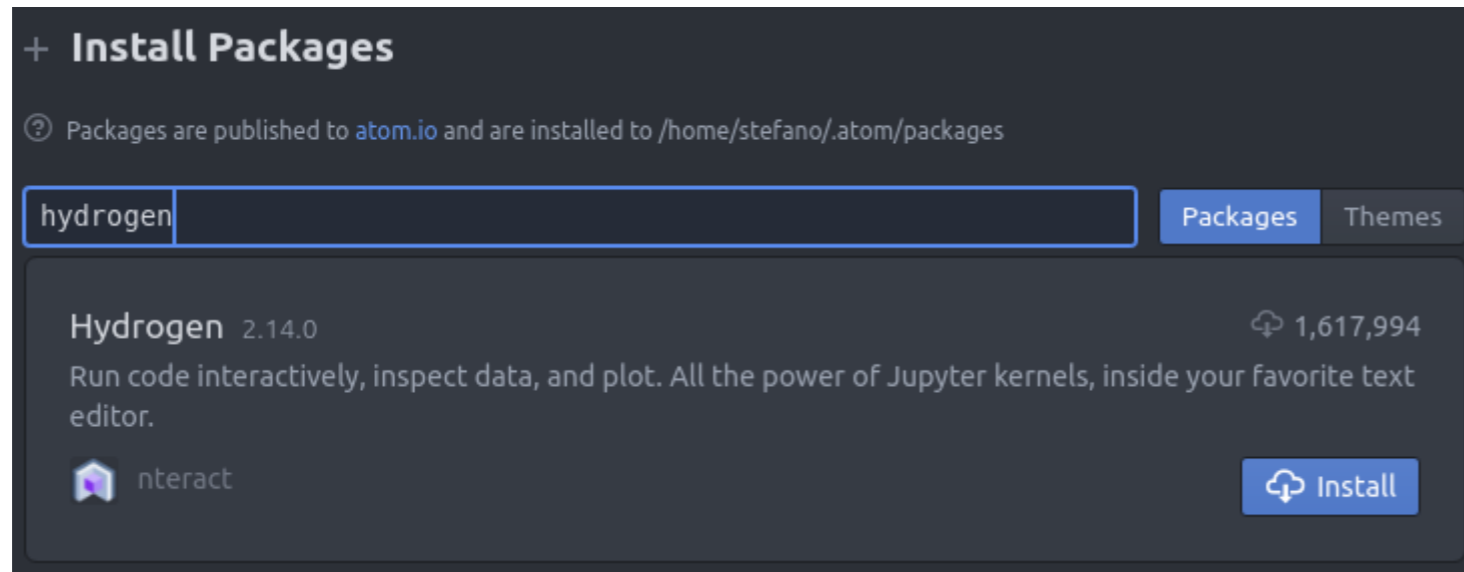
<https://github.com/nodeGame/eslintrc/blob/master/.eslintrc.js>

```
10 |
11 | • module.exports = function(treatmentName, settings, stager, setup, gameRoom) {
12 | |
13 | |   stager.setInit(function() {
14 | |     • var header, frame;
15 | |   });
16 | |   console.log('INIT PLAYER!');
17 | |
18 | |   node.game.oldContrib = null;
19 | |   node.game.oldPayoff = null;
20 | |   node.game.income = null;
21 | |
22 | |   // Setup page: header + frame.
23 | |   header = W.generateHeader();
24 | |   frame = W.generateFrame();
25 | |
26 | |   // Add widgets.
27 | |   this.visualRound = node.widgets.append('VisualRound', header);
28 | |   this.visualTimer = node.widgets.append('VisualTimer', header);
29 | }
```

Severity	Provider	Description	Line
Error	ESLint	'gameRoom' is defined but never used. (no-unused-vars)	11:67
Error	ESLint	'frame' is assigned a value but never used. (no-unused-vars)	14:21
Error	ESLint	Unexpected 'debugger' statement. (no-debugger)	183:13

Ok, One Last Package: Hydrogen

- The package Hydrogen let's you run JavaScript interactively directly in Atom using the **Jupyter Notebook**



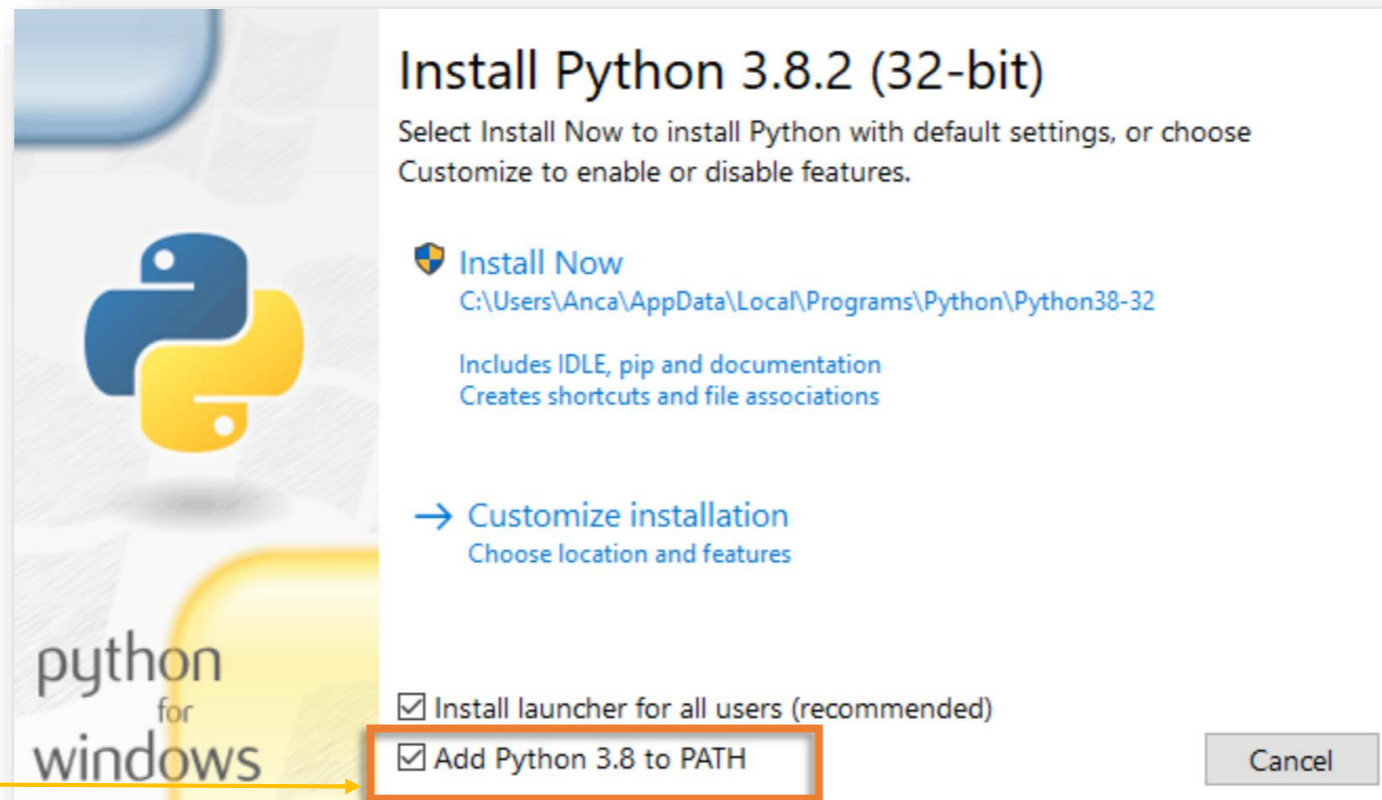
Hydrogen Installation Extra Steps

- After installing the package in Atom, installation is not over...
- You need to start or install the **Jupyter Notebook**
- **Follow the instructions reported in the next slides for your operating system**
- Important! Execute one line at the time
- Original installation instructions from:

<https://github.com/n-riesco/ijavascript>

Jupyter Notebook Requires Python3

If you don't have python, you also need to install it from: <https://www.python.org/>



If asked, make sure
this is selected

Ubuntu

UBUNTU 19.04

```
sudo apt-get install jupyter-notebook  
sudo npm install -g --unsafe-perm ijavascript  
sudo ijsinstall --install=global
```

UBUNTU 16.04

```
sudo apt-get install ipython ipython-notebook
```

Windows

Note! If you have **ANACONDA**, you already have Jupyter Notebooks, so go directly to Step 2.

Step 1. Install Jupyter Notebook with the PYTHON Official Distribution

```
pip3 install --upgrade pip  
pip3 install jupyter
```

Step 2 Install the JavaScript kernel for Jupyter Notebook

```
npm install -g ijavascript  
ijsinstall
```

Note! If `npm install -g` fails because of permission rights, install `ijavascript` without `-g` and invoke `ijsinstall` from the local dir.

MAC OSX

Note! If you have **ANACONDA**, you already have Jupyter Notebooks, so go directly to Step 2.

Step 1. Install Jupyter Notebook with the PYTHON Official Distribution

```
pip3 install --upgrade pip  
pip3 install jupyter
```

Step 2. Install HomeBrew and Necessary Dependency

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"  
brew install pkg-config zeromq
```

Step 3. Install the JavaScript kernel for Jupyter Notebook

```
sudo npm install -g ijavascript  
sudo ijsinstall
```

Note! If npm install -g fails because of permission rights, install ijavascript without -g and invoke ijsinstall from the local dir.

MAC OSX

Note! If you have **ANACONDA**, you already have Jupyter Notebooks, so go directly to Step 2.

Step 1. Install Jupyter Notebook with the PYTHON Official Distribution


```
pip3 install --upgrade pip  
pip3 install jupyter
```

Step 2. Install HomeBrew and Necessary Dependency

```
/bin/bash -c "$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install.sh)"  
brew install pkg-config zeromq
```

Step 3. Install the JavaScript kernel for Jupyter Notebook

```
sudo npm install -g ijavascript  
sudo ijsinstall
```

Password: 

sudo will prompt for your password. What you type may not be displayed, but it is recorded nonetheless. Just press enter once you have finished typing your password.

Note! If `npm install -g` fails because of permission rights, install `ijavascript` without `-g` and invoke `ijsinstall` from the local dir.

MAC OSX (Permission Issue)

If you see a permission error similar to the below:

```
prebuild-install WARN install EACCES: permission denied, access '/Users/balistef/.npm'  
fs.js:115  
  throw err;  
  ^
```

Try following the instructions here:

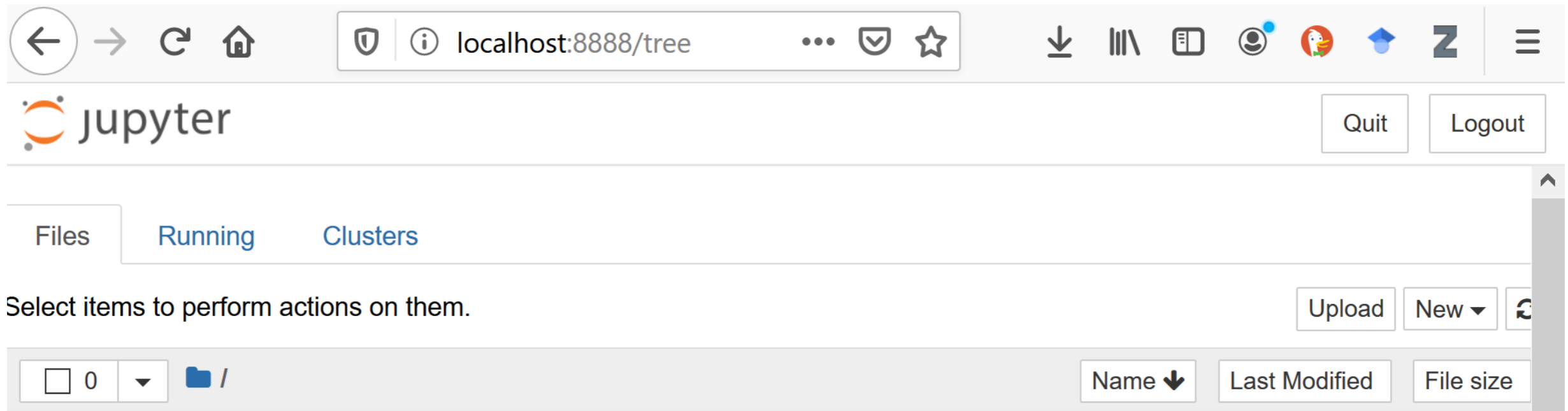
<https://www.kevinhooke.com/2018/02/09/fixing-npm-global-install-permissions-on-macos/>

Also try removing the sudo command, that is:

```
npm install -g ijavascript  
ijsinstall
```

Start the Notebook

- After you completed all the steps you can start the Jupyter notebook with `jupyter notebook`
- And you should see something like this in your browser:



Start the Notebook

- After you completed all the steps you can start the Jupyter notebook with `jupyter notebook`
- And you should see something like this in your browser:

You can stop the Jupyter notebook here when you do not need it.

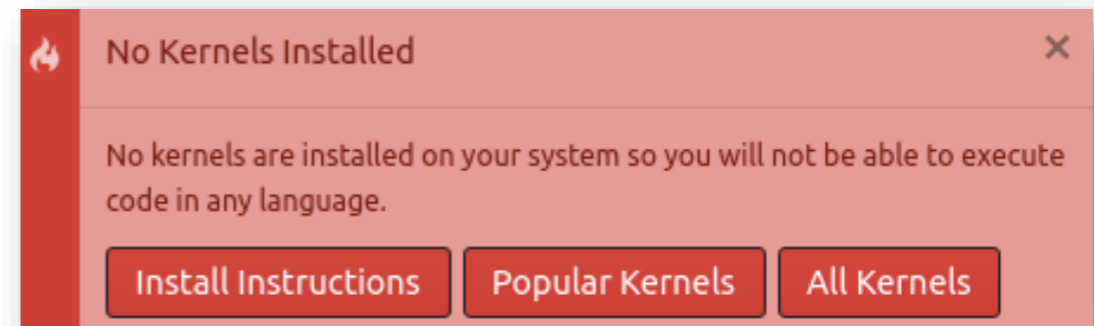
The screenshot shows the Jupyter Notebook web interface. The browser's address bar is highlighted with an orange box and contains the text `localhost:8888/tree`. Below the address bar, the Jupyter logo and name are visible. To the right, there are navigation icons and a 'Quit' button, which is also highlighted with an orange box. A yellow callout box points to the 'Quit' button with the text: "You can stop the Jupyter notebook here when you do not need it." Below the navigation bar, there are tabs for 'Files', 'Running', and 'Cluster'. A yellow callout box points to the address bar with the text: "You can always reach this page at this address (*localhost* means your local machine)." At the bottom, there is a toolbar with buttons for 'Upload', 'New', and a refresh icon. Below the toolbar, there is a table header with columns for 'Name', 'Last Modified', and 'File size'.

Did it work?

- Open a JavaScript file in Atom (if you create a new one, it must have the **.js extension**)
- Position the cursor on a line and press Ctrl+Enter to execute it
- If successful, you should see the result of the execution next to it
- If there is a problem an error will pop up

```
myVariable = 10; 10  
myVariable*=2; 20
```

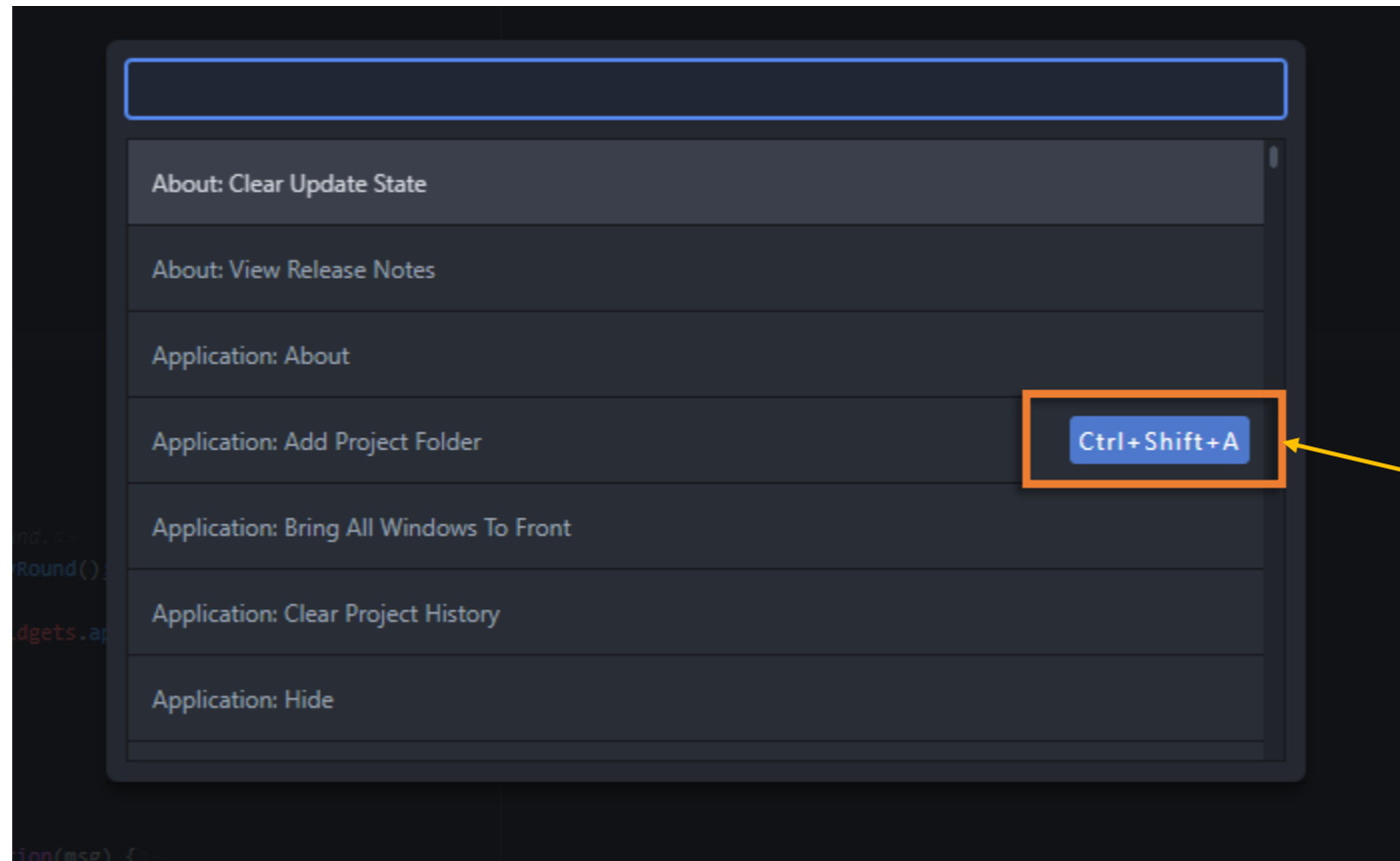
Well Done, You Did It!



It was a good try, but something did not work out.

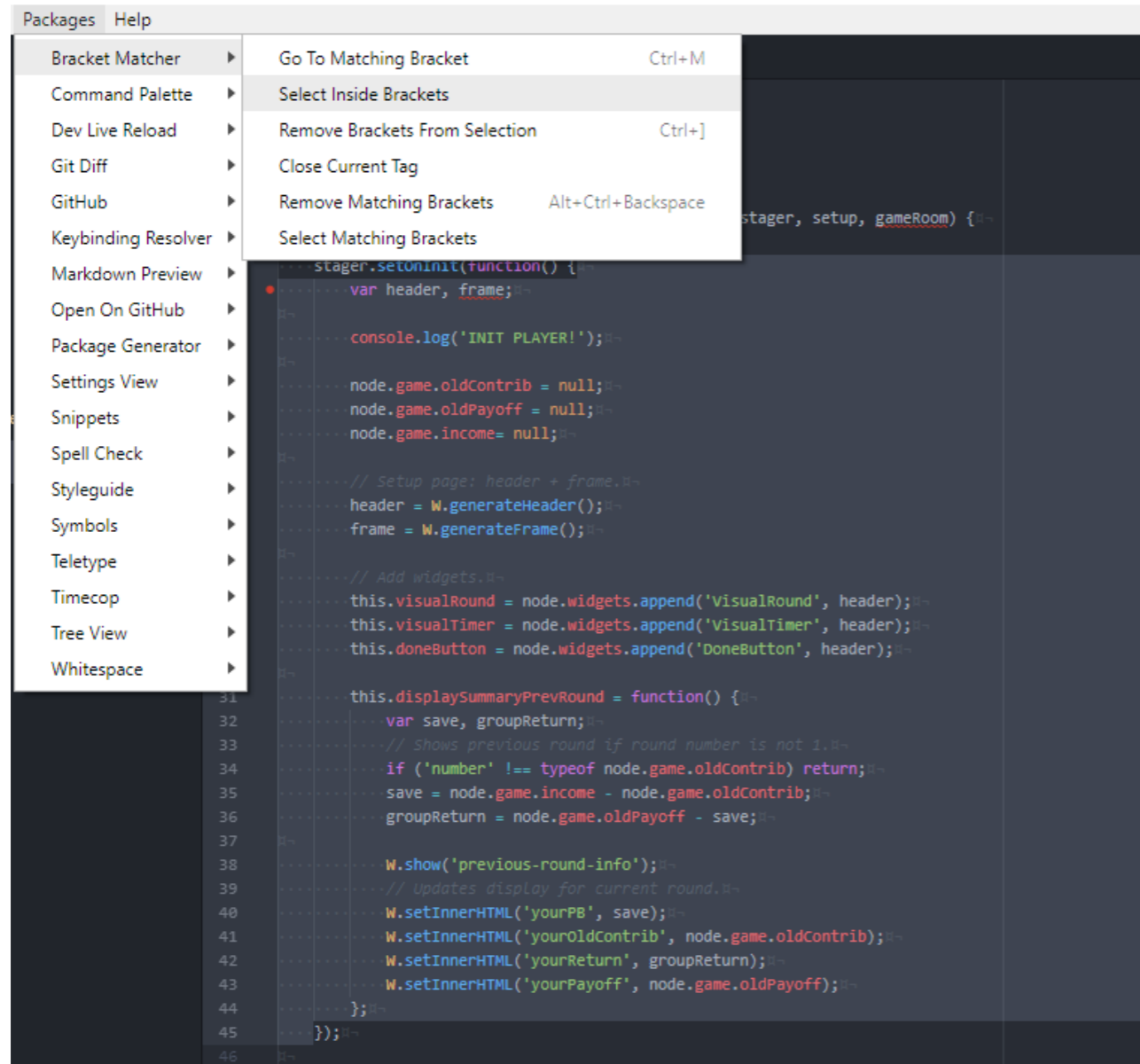
Learn New Commands and Shortcuts

Hold keys **CTRL-SHIFT-P** to open a new menu with all available commands and shortcuts (when available)



Try this shortcut to open a new project

Bracket Matcher



It is really to miss closing a parenthesis, but it is really hard to find it!

The Bracket Matcher package is here for you to help!

Hint: If you use this command a lot, what about creating a shortcut? Try it yourself following the instructions in the Keybinding slides.

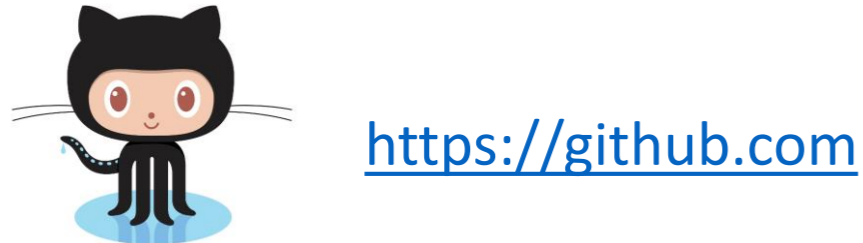
Git and GitHub Integration

If you work in a team, but also if you are a lone developer, you will need:

- version controlling system,



- an online repository



You are lucky!

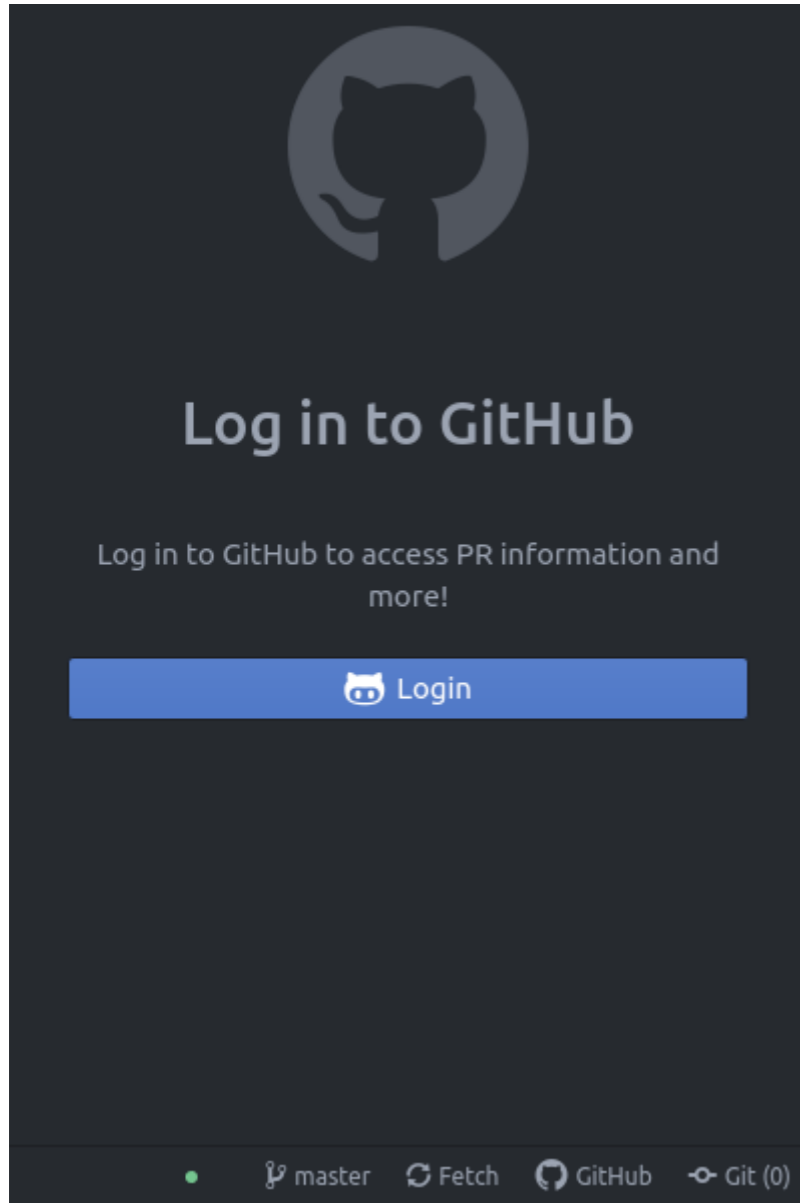
ATOM offers native **integration** with both!



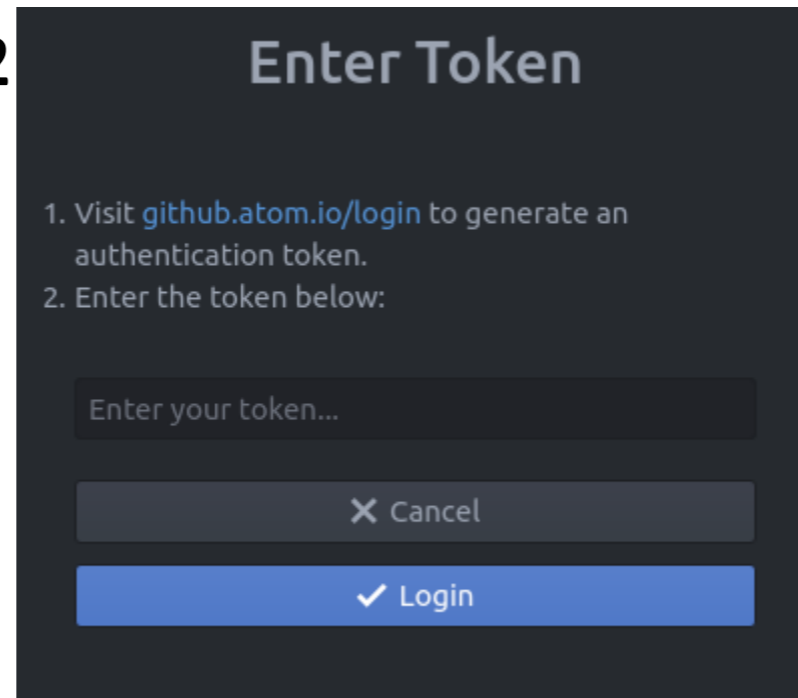
Let's learn how to use it.

Git and GitHub Integration

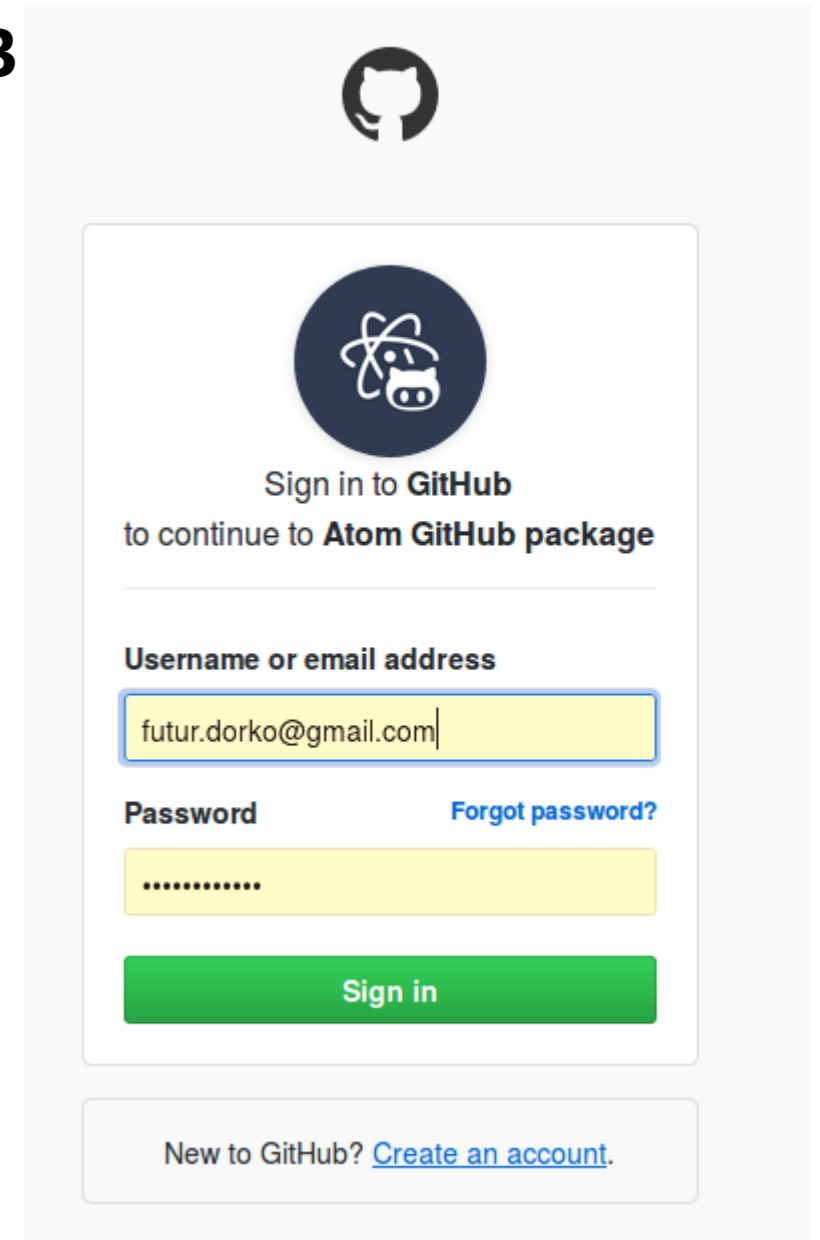
1



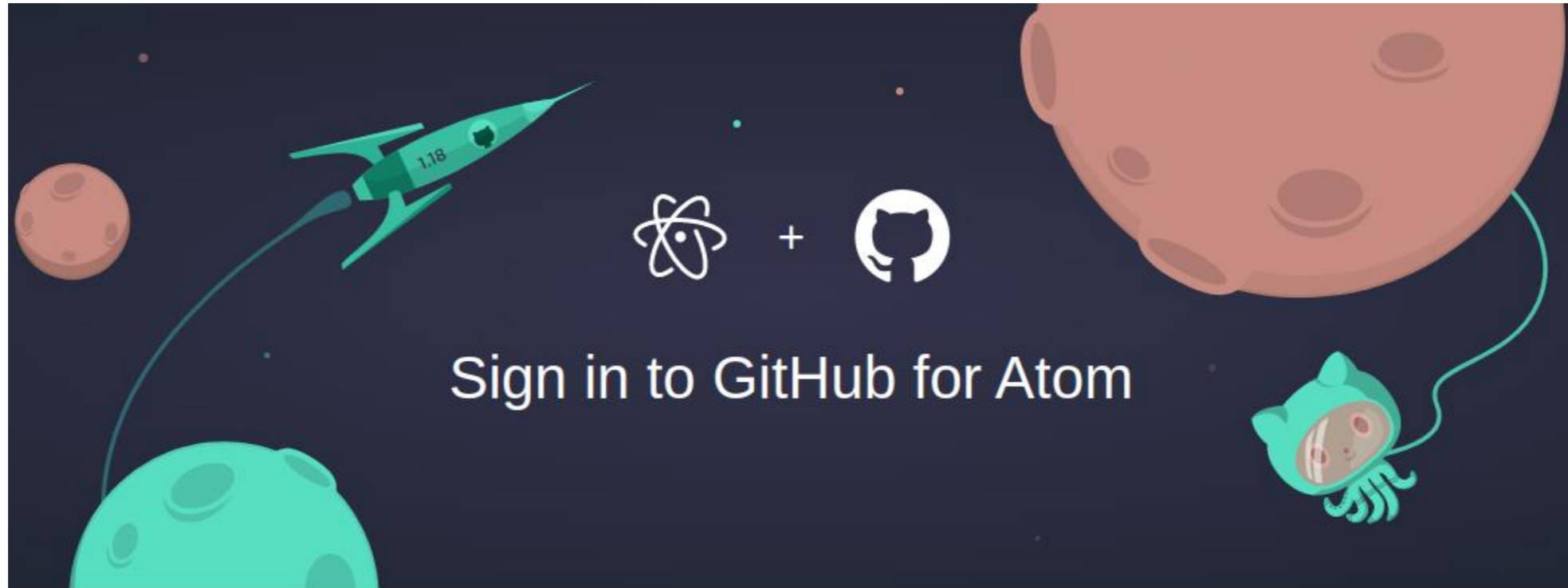
2



3



Git and GitHub Integration



Your GitHub token

9f8acec8627e3995937d49XXXXXXXXXXXX

Token copied!

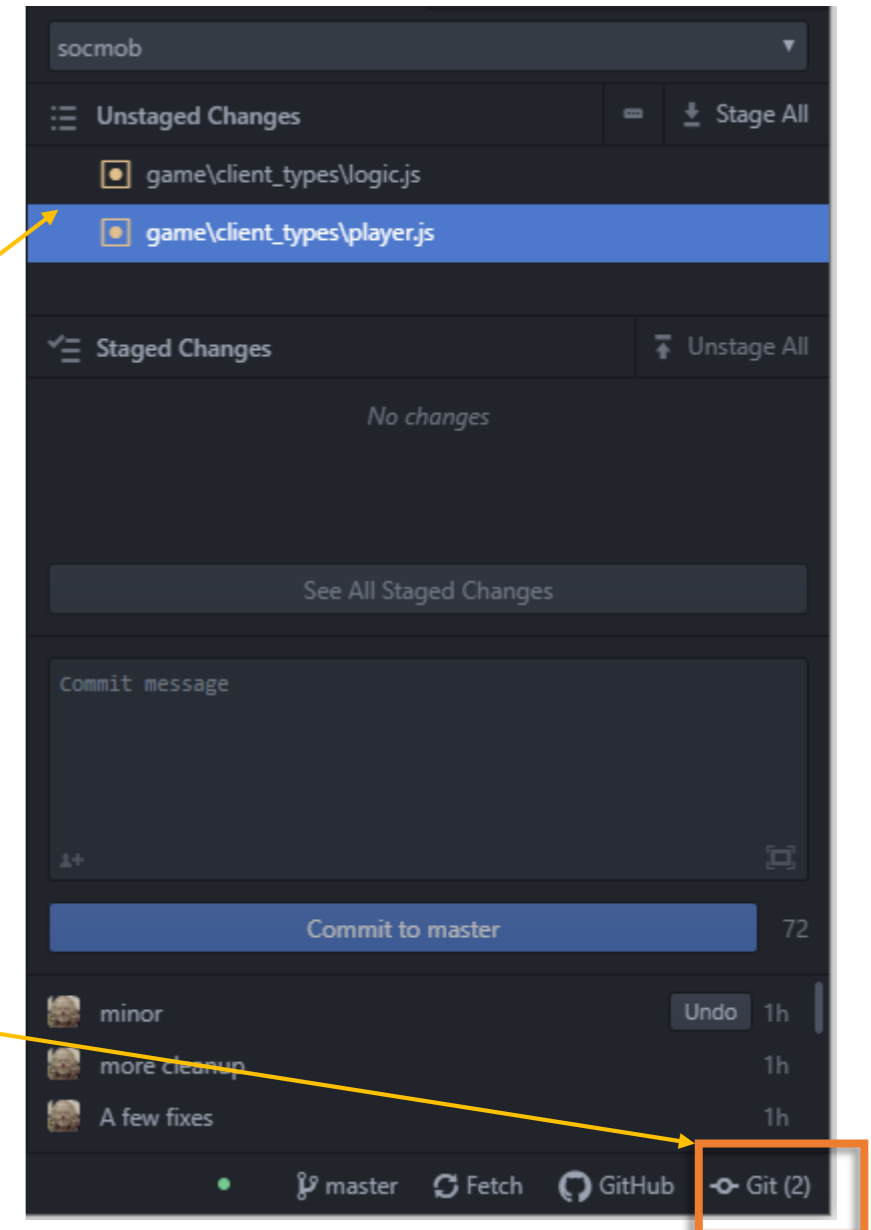
To finish signing in, copy your token and paste it into the sign-in form in Atom.

Git and GitHub Integration

1. Open the Git Pane clicking on the bottom right icon.
2. Click on a file to verify the changes

These are the files that contain changes.

The number in parenthesis counts how many files are modified ("unstaged").



Git and GitHub Integration

In file player.js we removed a line that contained a debugger statement ("red") and replaced it with an empty line ("green")

If we are happy with the changes, click on "Stage All" to add all files to the index.

You can also double click on a single file to select only some of the changed files to be added to the index.

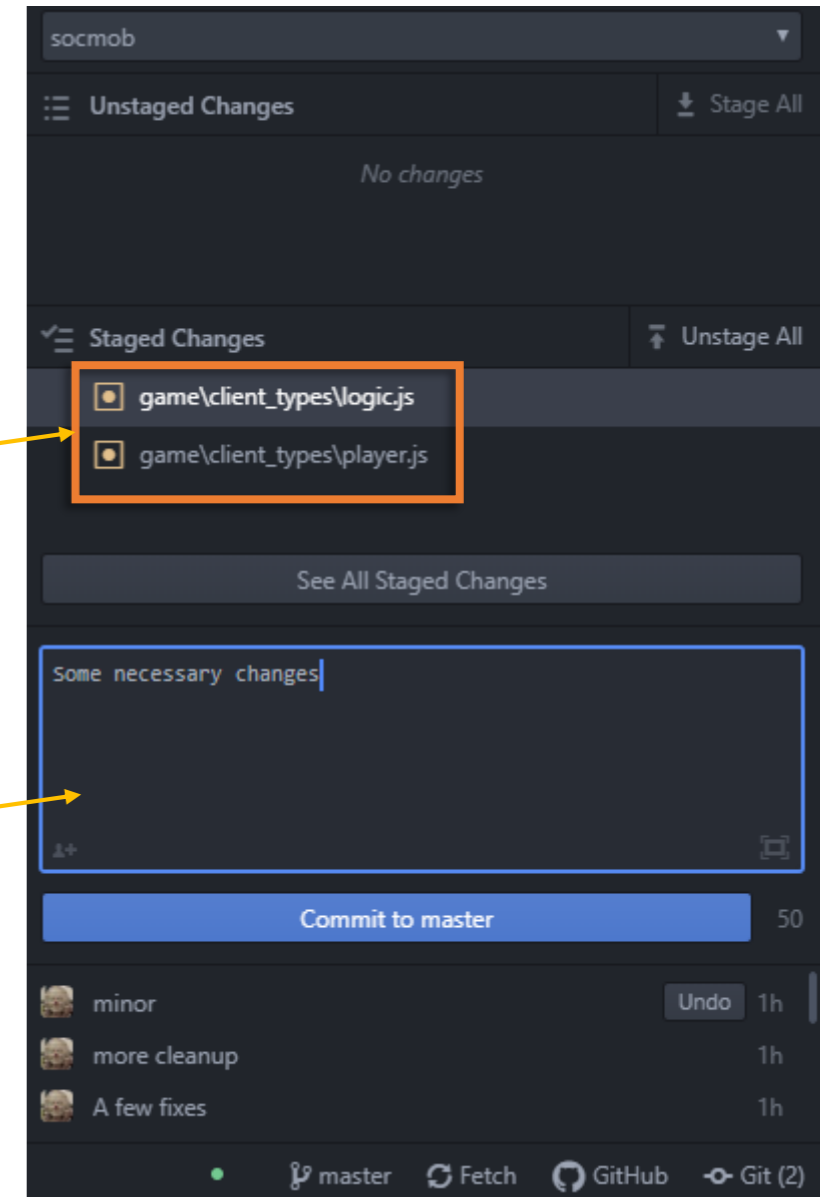
The screenshot shows the Visual Studio Code interface with a code editor and a Git sidebar. The code editor displays a diff for the file 'game\client_types\player.js'. The diff shows a red line for the removed 'debugger' statement and a green line for the added empty line. The Git sidebar shows 'Unstaged Changes' with 'game\client_types\player.js' selected, and a 'Stage All' button highlighted in an orange box. The commit message field is empty, and the 'Commit to master' button is visible.

Git and GitHub Integration

1. Open the Git Pane clicking on the bottom right icon.
2. Click on a file to verify the changes
3. Click on Stage All
4. Add a commit message

Here are the files ready to be "committed" to the index.

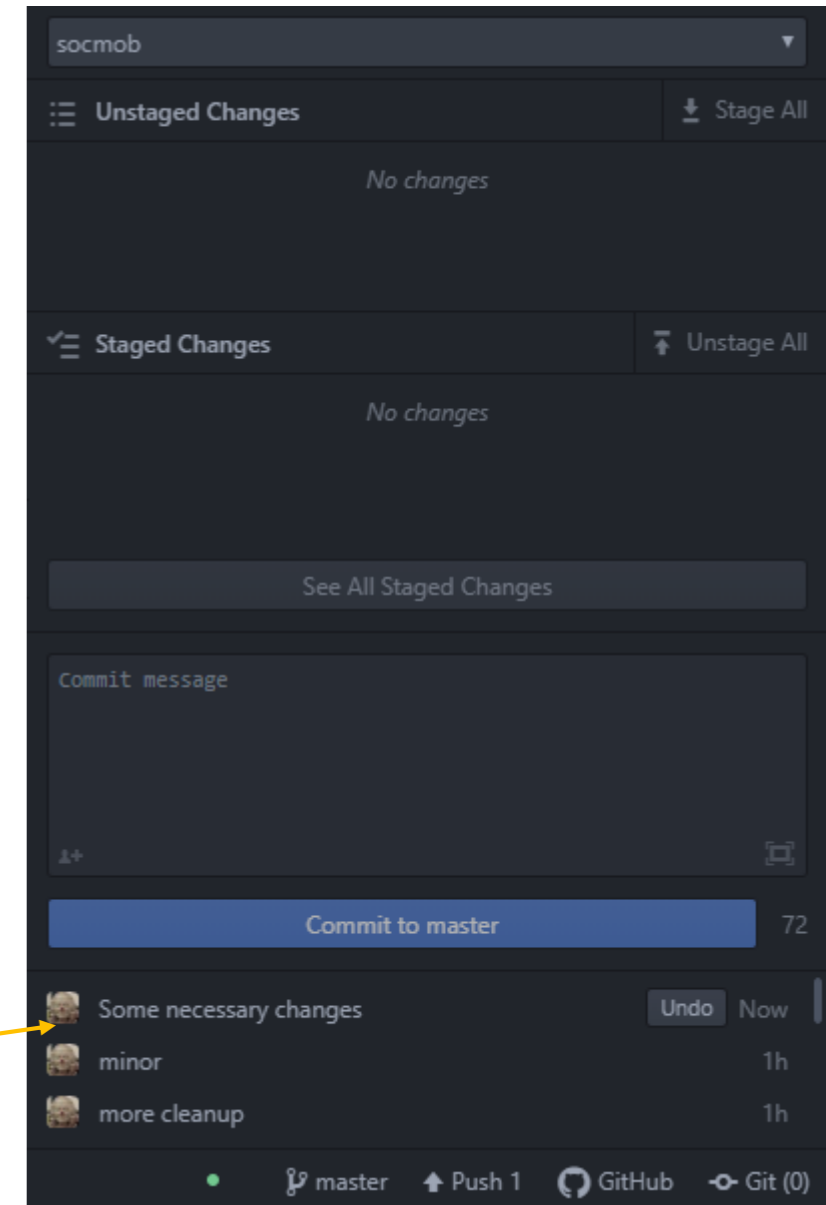
Here is the commit message: a short text describing what you changed.



Git and GitHub Integration

1. Open the Git Pane clicking on the bottom right icon.
2. Click on a file to verify the changes
3. Click on Stage All
4. Add a commit message
5. Click "Commit to master"

Here is your last commit has been added to your **local** index.



Git and GitHub Integration

1. Open the Git Pane clicking on the bottom right icon.
2. Click on a file to verify the changes
3. Click on Stage All
4. Add a commit message
5. Click "Commit to master"
6. Click to "Push" to upload your staged files to the online Github repository.

Note! The Github repository is available if you have a Github account and you do either:

- Follow online instructions to link an online repository to your local repository
- Clone an online repository.

See the GitHub guide for more info.

