

Configuring the Text Editor Atom

Stefano Balietti

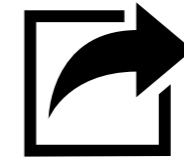
Center for European Social Science Research at Mannheim University (MZES)
Alfred-Weber Institute of Economics at Heidelberg University

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

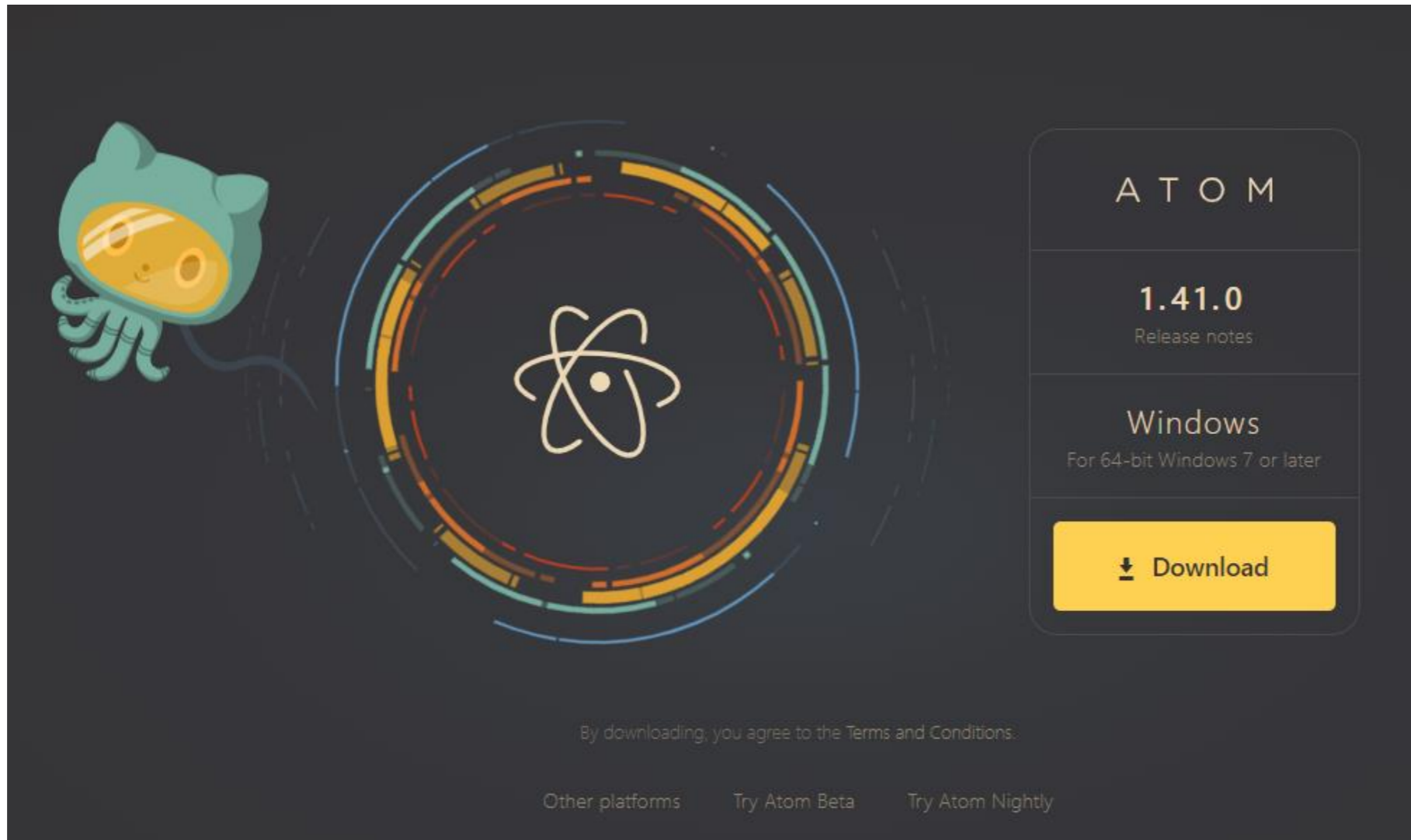


Design and Implementation of Online Experiments

ATOM



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



The image shows a dark-themed download page for the Atom editor. On the left, there is a large illustration of the Atom logo, which consists of a teal cat-like character with a yellow face and a white atomic symbol in the center of a circular, multi-layered ring. To the right of the illustration is a white-bordered box containing the following information:

- A T O M**
- 1.41.0**
Release notes
- Windows**
For 64-bit Windows 7 or later
- Download** (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." Below this, there are three links: "Other platforms", "Try Atom Beta", and "Try Atom Nightly".

Indentation

Indentation is nr. 1 feature of *high-quality* code

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 nr. 1 feature of *high-quality* code

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 nr. 1 feature of *high-quality* code

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);
```

Properly Indented

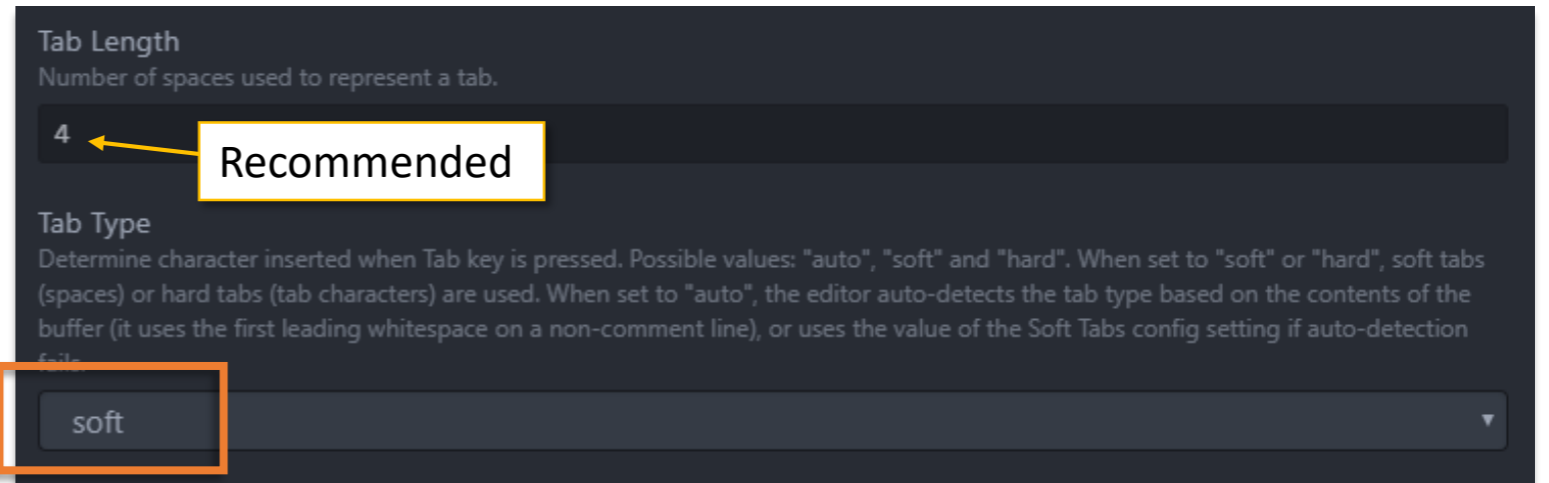
```
let a = 1234;
a++;
if ( a > 123 ) {
    a--;
}
else {
    a++;
}
console.log(a);
```

More Readable
Easier to catch errors

ATOM Editor Settings

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

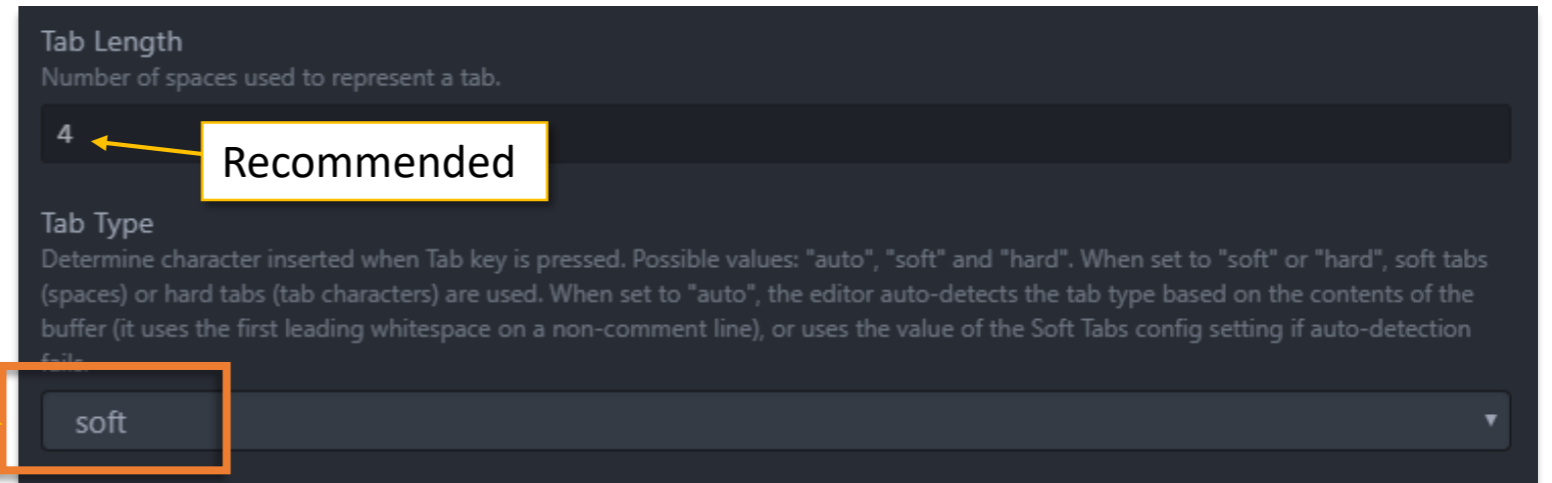
Make sure all your TABs are
automatically converted into spaces.



ATOM Editor Settings

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

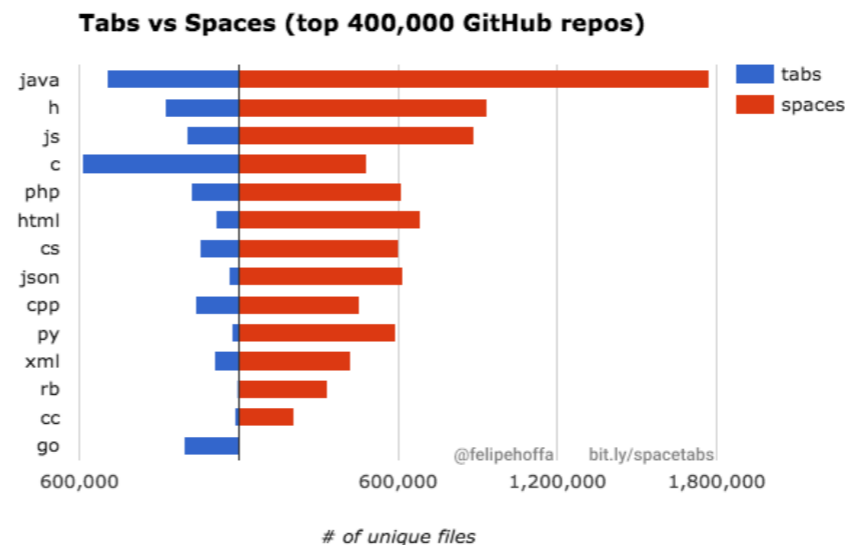
Make sure all your TABs are
automatically converted into spaces.



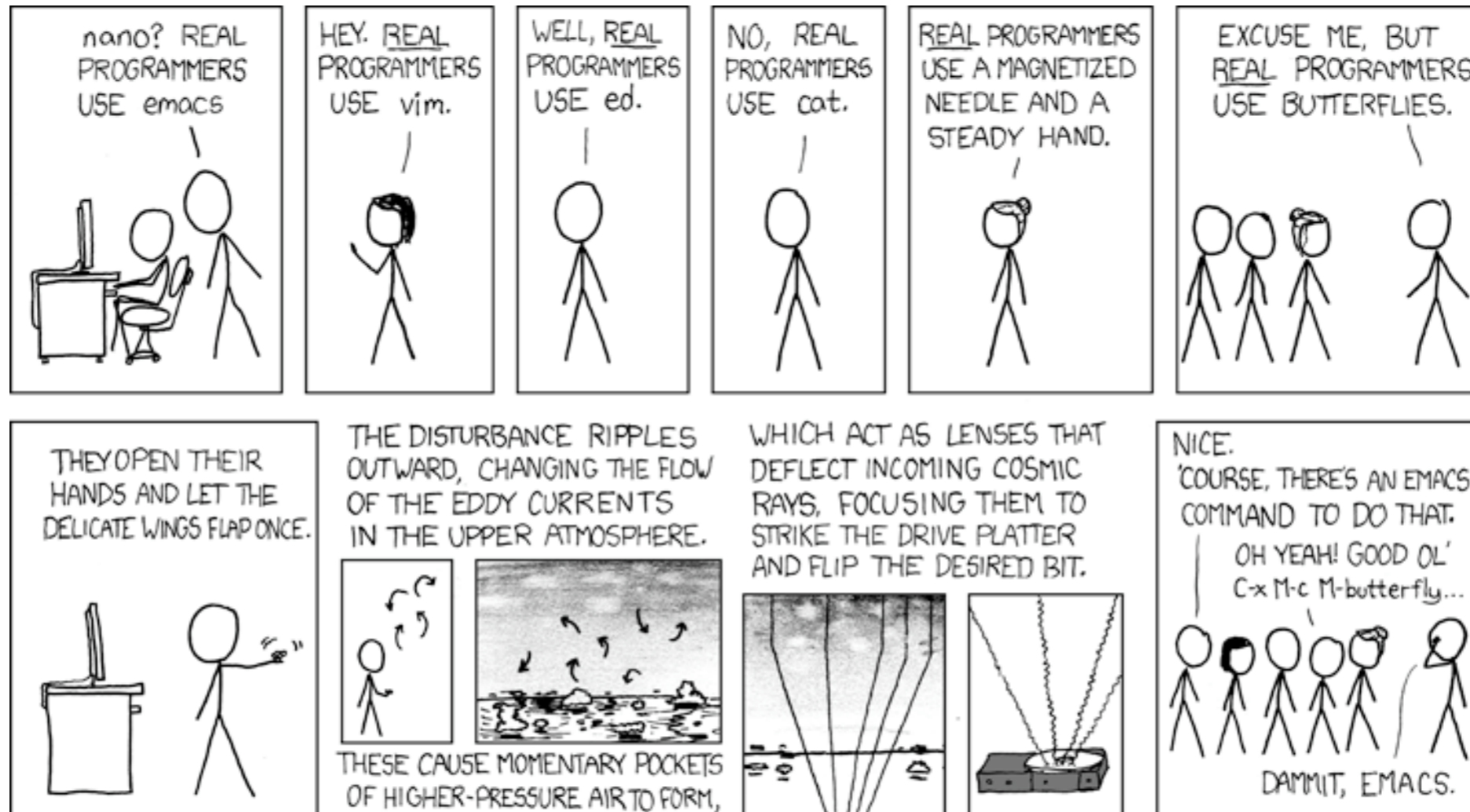
Why? Because TABs are evil!



**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 the value. The 'Tab Type' dropdown menu 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 this, the 'Editor Settings' section is expanded, showing a list of checked options. The 'Atomic Soft Tabs' option is highlighted with an orange box and a yellow arrow pointing from the text 'The best of the two worlds.'

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.' The 'Editor Settings' section has a yellow box around the 'Atomic Soft Tabs' checkbox, with a yellow arrow pointing from the text 'The best of the two worlds.' Other settings shown include 'Auto Indent', 'Auto Indent On Paste', and 'Confirm Checkout HEAD Revision', all of which are checked.

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

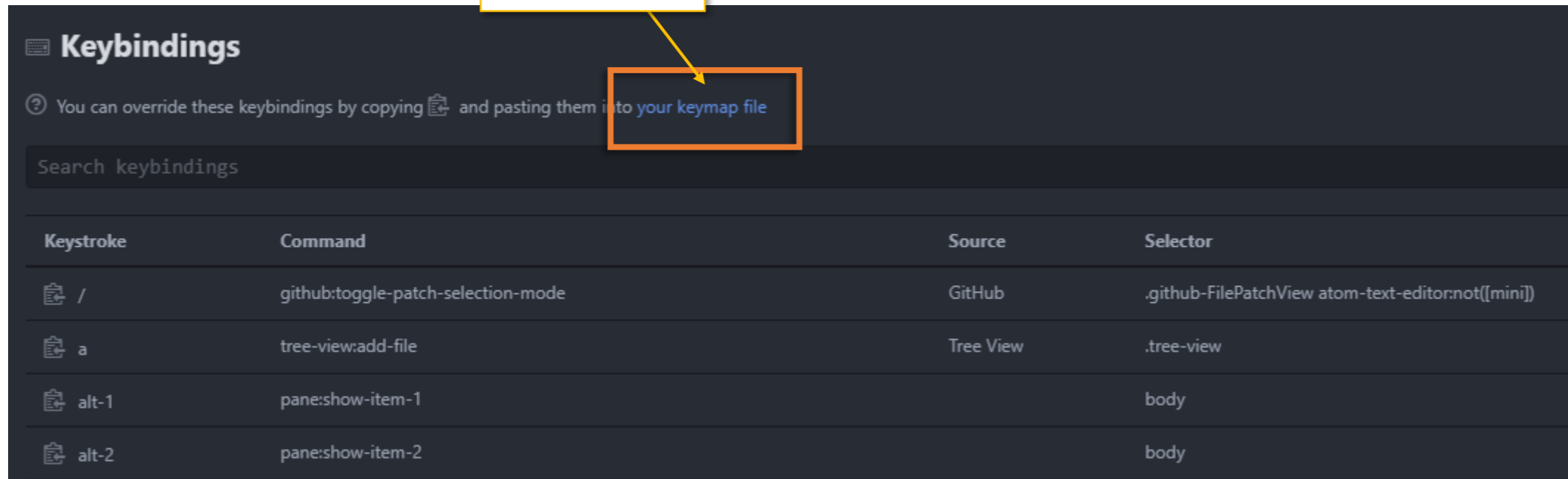
These settings are related to text editing. Some of these can be overridden on a per-language basis. Check language settings by clicking its package card in the Packages list.

- 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

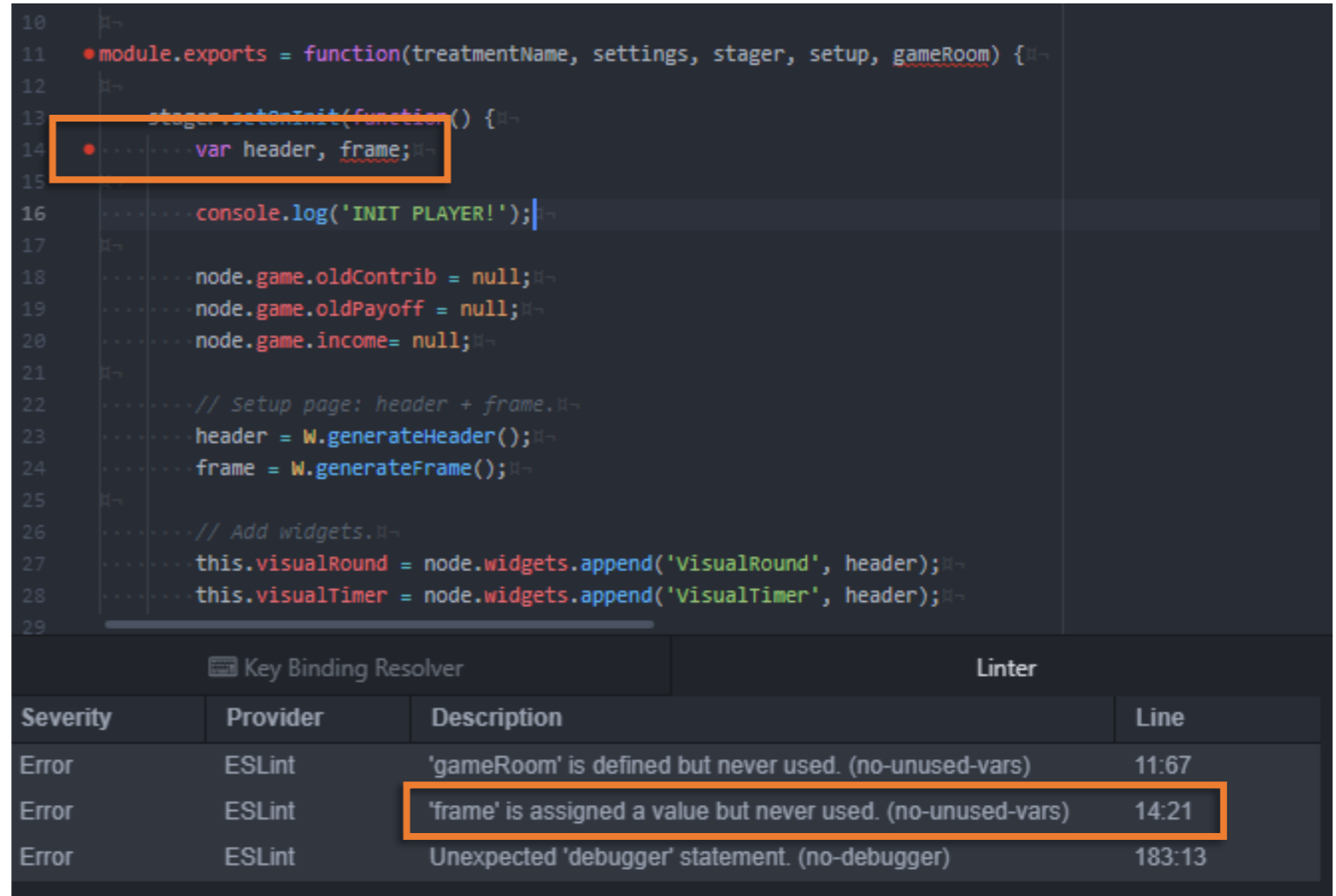
Red dot next to the line containing an error or warning.

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>

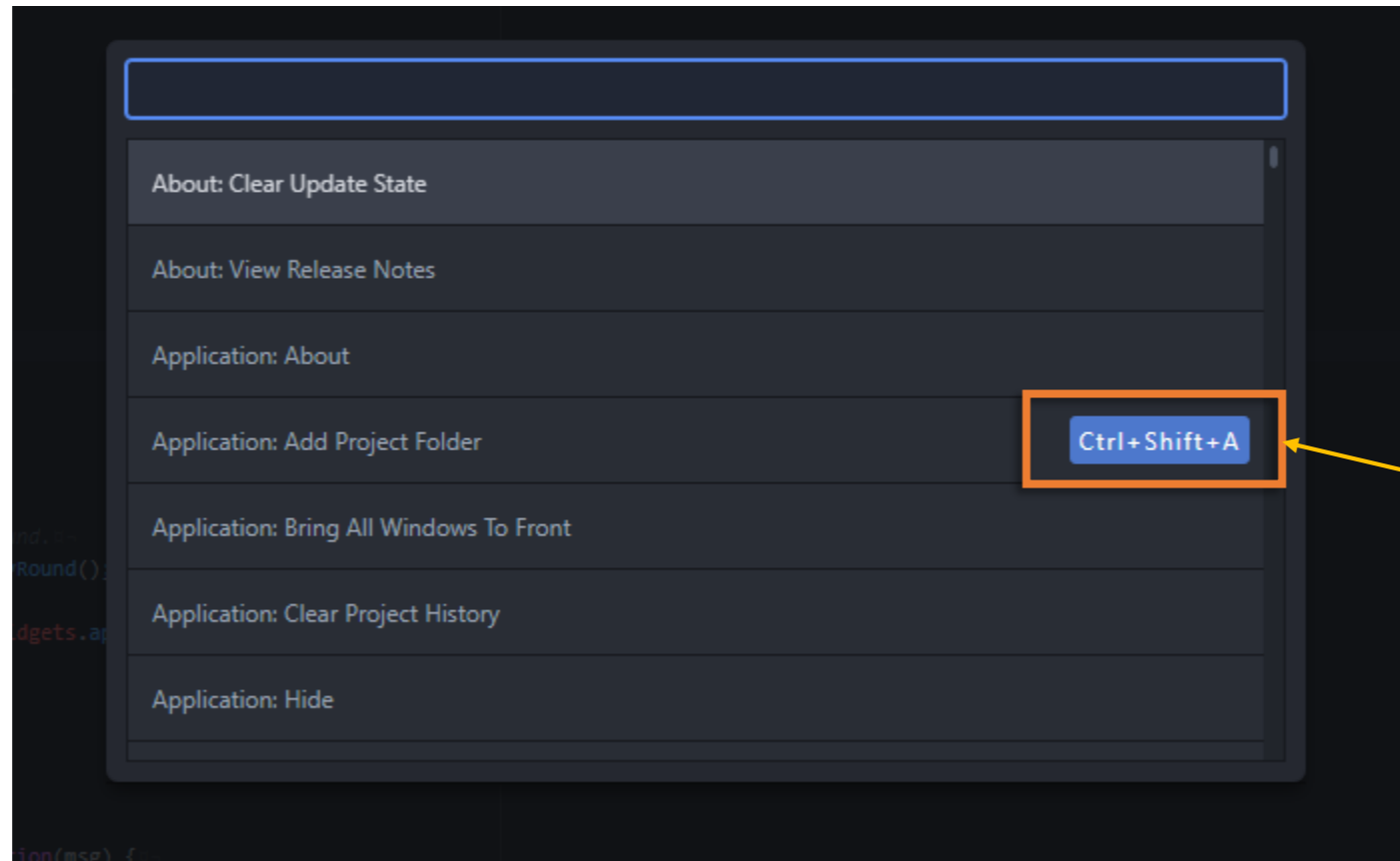


```
10 |
11 | • module.exports = function(treatmentName, settings, stager, setup, gameRoom) {
12 |
13 |   stager.setInit(function() {
14 |     • var header, frame;
15 |
16 |     console.log('INIT PLAYER!');
17 |
18 |     node.game.oldContrib = null;
19 |     node.game.oldPayoff = null;
20 |     node.game.income = 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

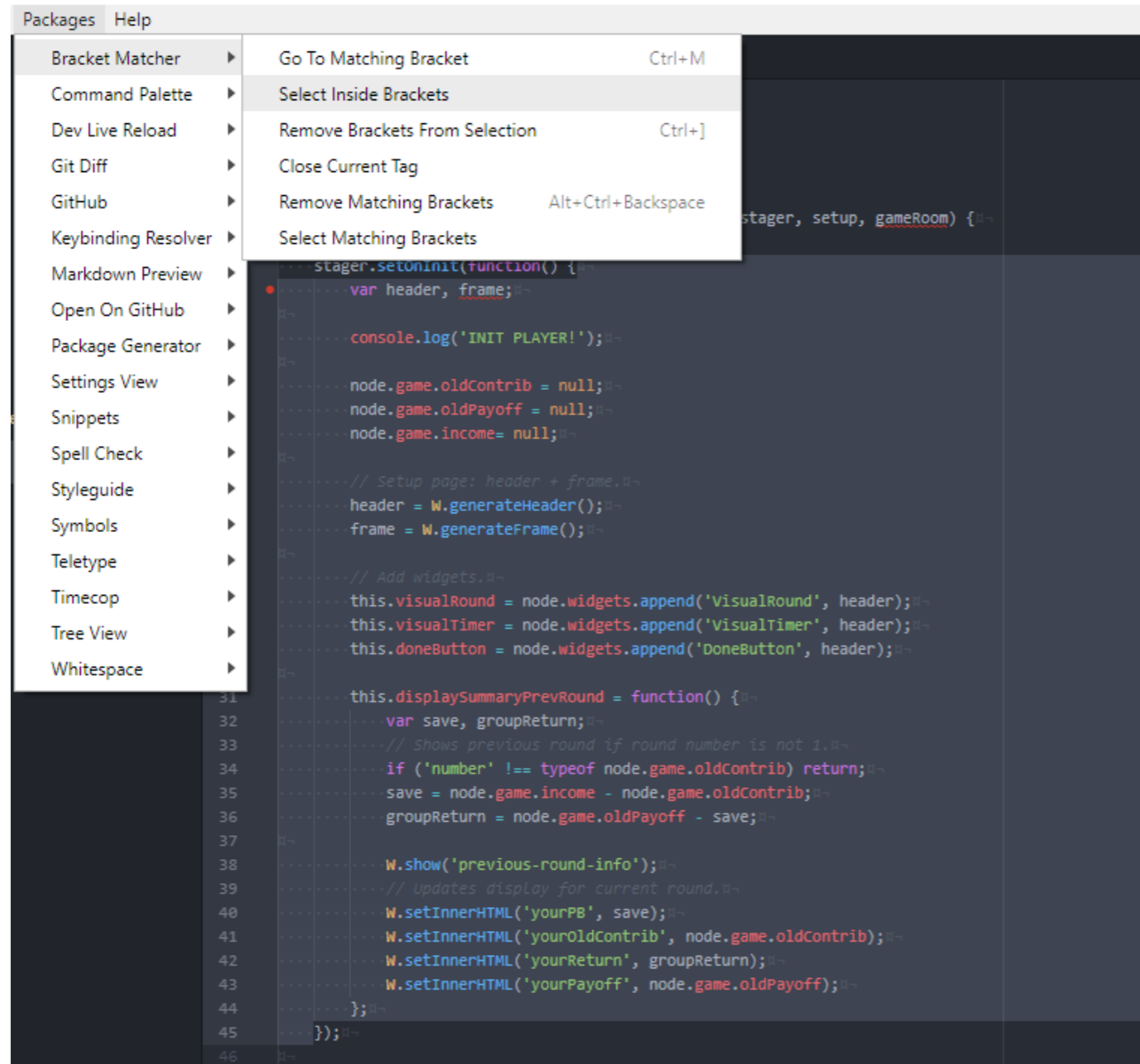
Learn New Commands and Shortcuts

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.