

**MAKE** **it** **REAL**  
           .camp

- 1- Consejos antes de empezar*
- 2- Editor de Texto*
- 3- Línea de Comandos*

# Código Binario Descifrado

---

*Quien lo hizo?*

## *Las 3 habilidades más importantes que debes aprender*

- 1- Leer y escribir el código - (mostrar atom)*
- 2- Prestar atención a los detalles - (mostrar detalles)*
- 3- Detectar diferencias (mostrar github)*

## Consejos personales

- 1- *Ponerse una meta diaria de horas para programar. Llevar la contabilidad de horas con togg*
- 3- *Apagar el celular mientras codificas*
- 4- *Por el CPU no entrar a redes mientras codificas*
- 5- *No salir a cafés, enfoque! (problemas de red, distractores)*
- 6- *Evitar reuniones cara a cara, manejalas por hangout, o skype*
- 7- *Habláte a ti mismo de forma positiva (aumenta lo positivo, disminuye lo negativo)*
- 8- *Apuntar problemas en papel y tratar de resolverlos ahí primero*
- 9- *Pocas ventanas en el navegador*
- 10- *Siempre escucho música suave*
- 11- *Mucho café, aromática o redbull*

# Porqué es tan difícil aprender?

---

## *Otras sugerencias*

- 1- No copiar y pegar*
- 2- Encontrar cosas en internet*
- 3- Leer un código*
- 4- Lidar con el código de otros programadores*

# Editor de Texto - Qué es?

---

*Quien descargó Atom?*

# Editor de Texto - Qué es?

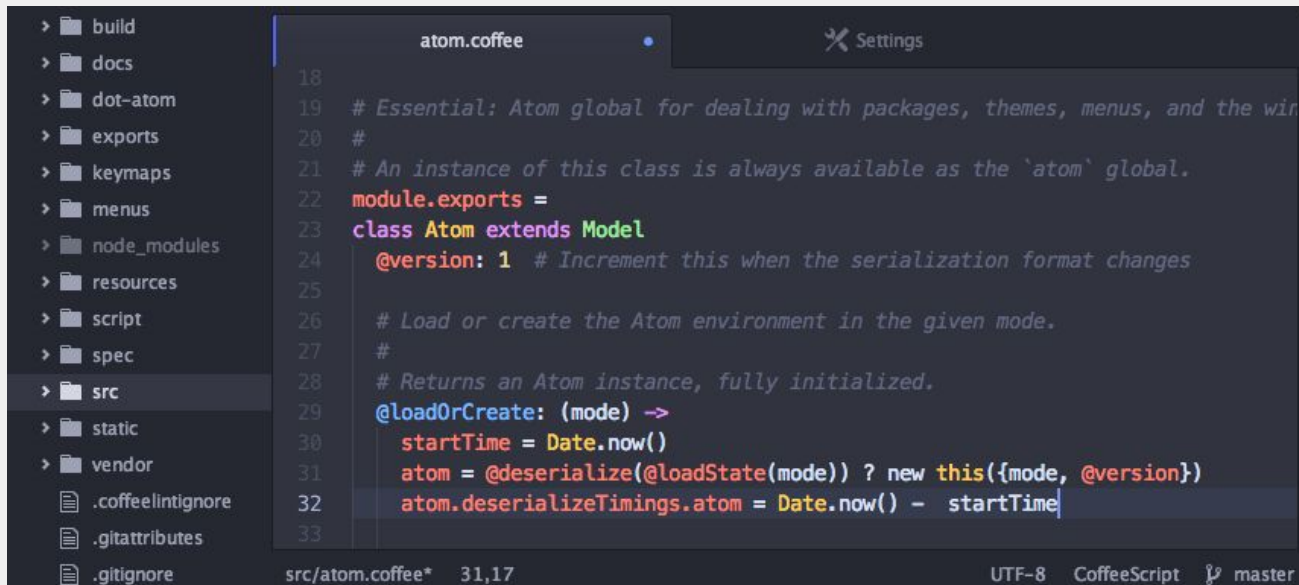
---

*Ejercicio:*

*Abramos el bloc de notas de su sistema operativo*



# Editor de Texto - Qué es?



The screenshot shows the Atom text editor interface. On the left is a sidebar with a file tree containing folders like 'build', 'docs', 'dot-atom', 'exports', 'keymaps', 'menus', 'node\_modules', 'resources', 'script', 'spec', 'src' (selected), 'static', 'vendor', and files like '.coffeelintignore', '.gitattributes', and '.gitignore'. The main editor area displays the 'atom.coffee' file with CoffeeScript code. The code includes comments about the Atom global, a class definition for 'Atom' extending 'Model', and a '@loadOrCreate' function that initializes the Atom environment. The status bar at the bottom indicates the file path 'src/atom.coffee\*', line and column '31,17', encoding 'UTF-8', language 'CoffeeScript', and branch 'master'.

```
18
19 # Essential: Atom global for dealing with packages, themes, menus, and the win
20 #
21 # An instance of this class is always available as the `atom` global.
22 module.exports =
23 class Atom extends Model
24   @version: 1 # Increment this when the serialization format changes
25
26   # Load or create the Atom environment in the given mode.
27   #
28   # Returns an Atom instance, fully initialized.
29   @loadOrCreate: (mode) ->
30     startTime = Date.now()
31     atom = @deserialize(@loadState(mode)) ? new this({mode, @version})
32     atom.deserializeTimings.atom = Date.now() - startTime
33
```


src/atom.coffee\* 31,17 UTF-8 CoffeeScript master


- *Colores*
- *Orden e indentación*
- *Carpetas y subcarpetas*
- *Depuradores de errores*
- *Atajos y formas rápidas*

# Editor - Descarga

<https://atom.io>

Packages Themes Documentation Blog Discuss [Sign In](#)

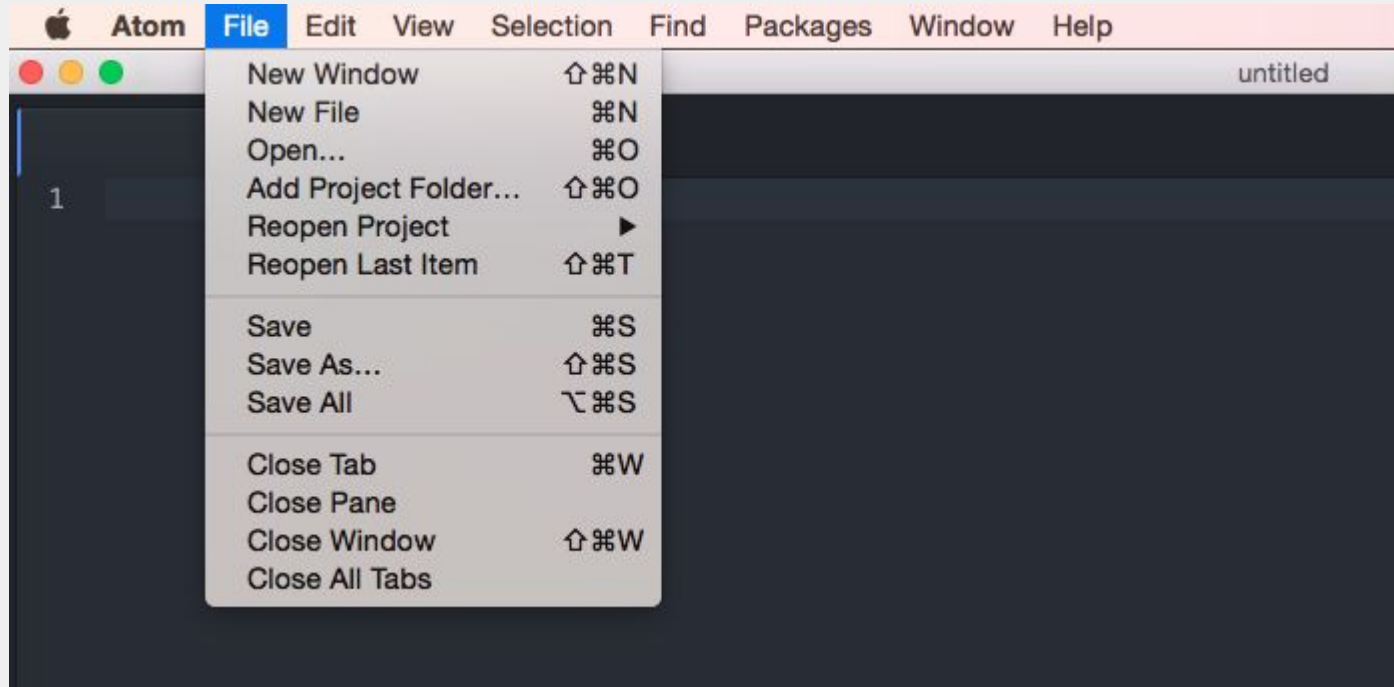
 **ATOM**  
A hackable text editor  
for the 21st Century



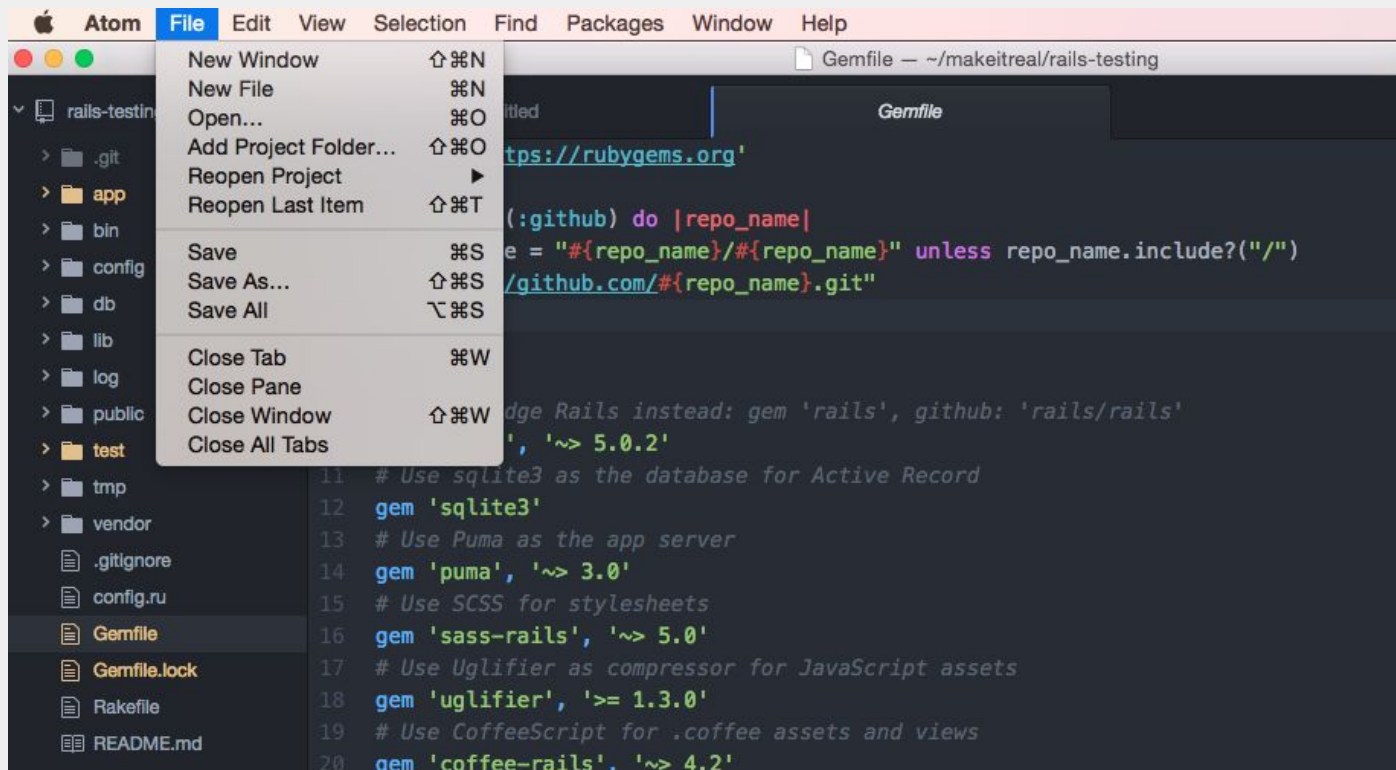
[Download For Mac](#)

For macOS 10.8 or later. Other platforms - Beta releases

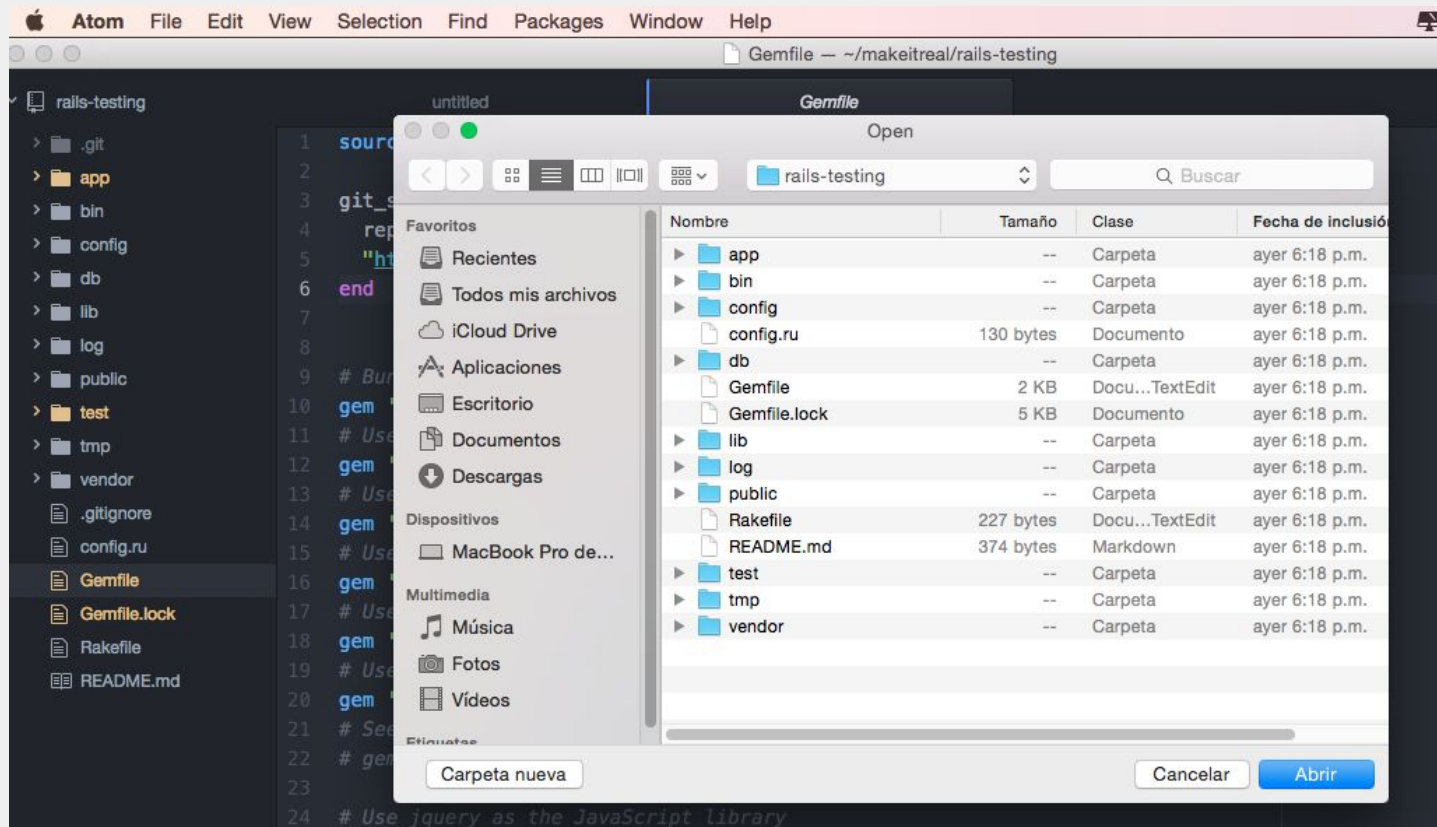
# Editor - Crear Archivo



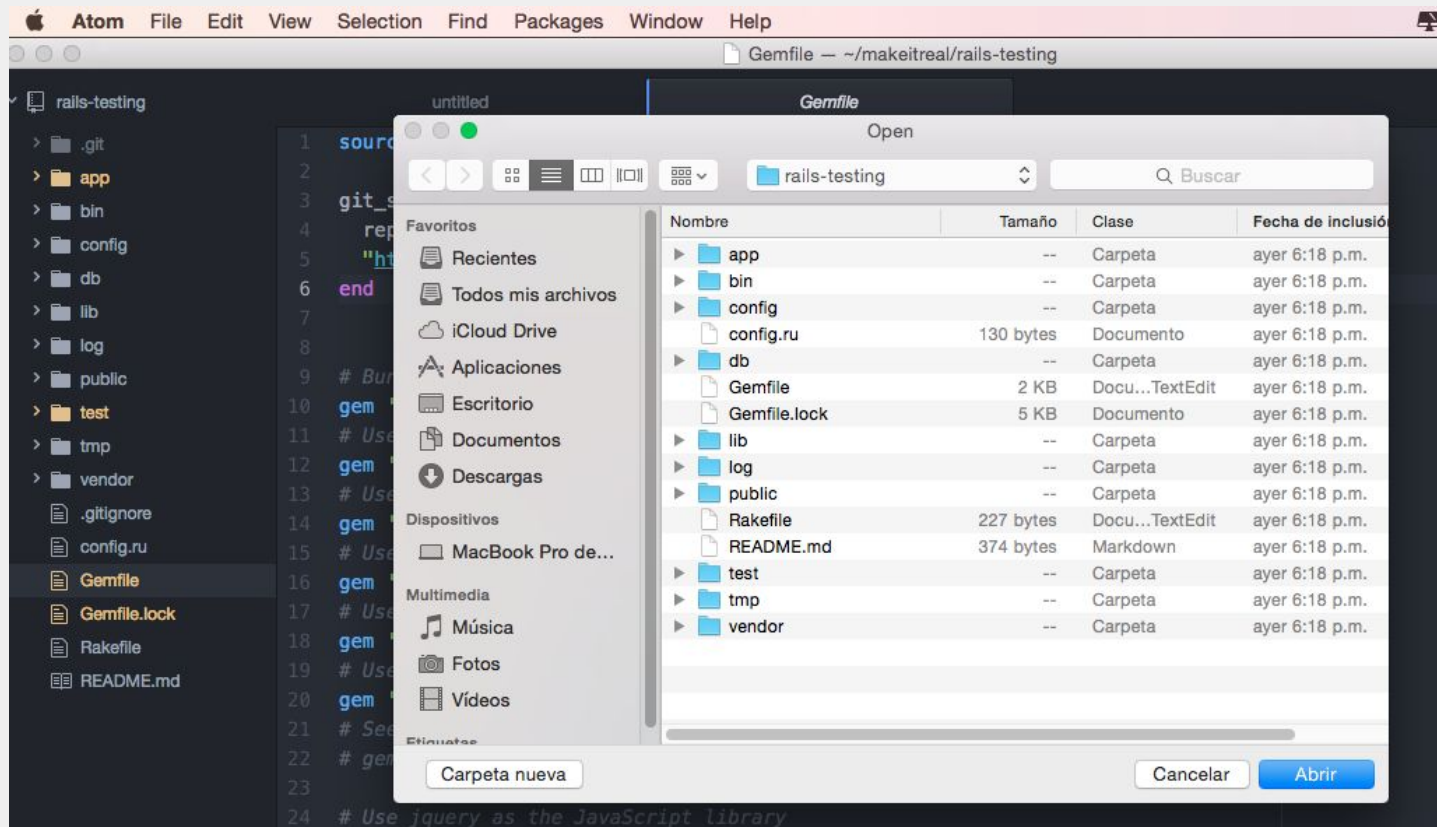
# Editor - Guardar Archivo



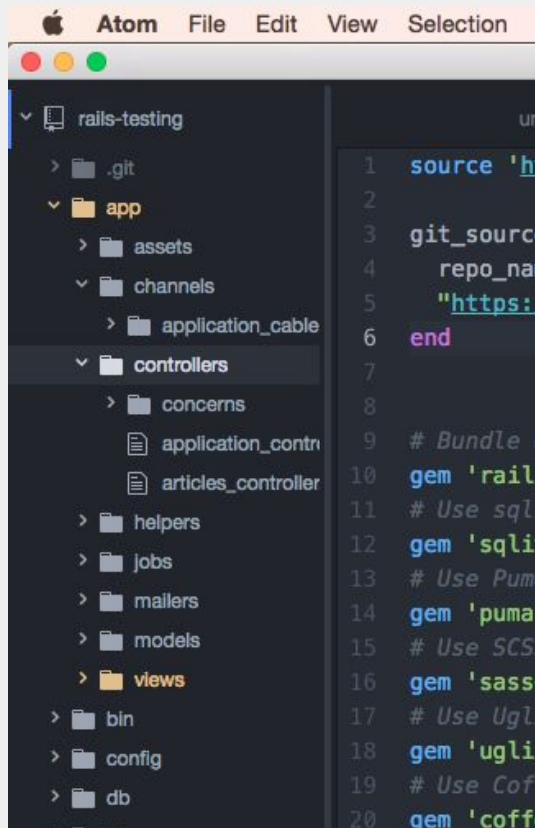
# Editor - Abrir Archivo



# Editor - Abrir Carpeta



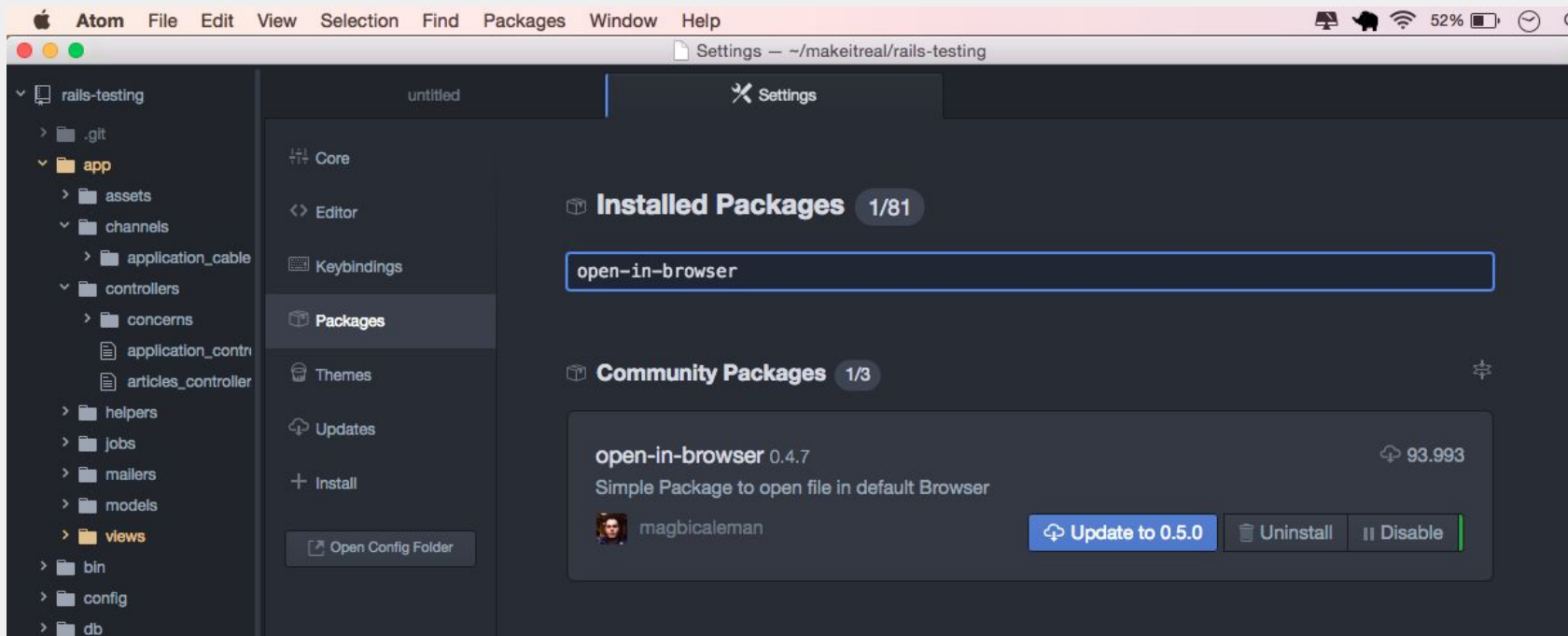
# Editor - Manejar Archivos





# Editor - Paso Avanzado

*Open in Browser (atom/preferences/install/open-in-browser)*



# Editor - Ejercicios

---

- 1- *Crear una carpeta llamada ruby*
- 2 - *Crear dentro de esa carpeta un archivo llamado (hola-mundo.html)*
- 3- *Escribir dentro de este archivo = “Hola mamá desde Atom”*
- 4- *Guardar archivo*
- 5- *Open in browser*
- 6- *Cerrar y salir*

# Editor - Ejercicios (2)

---

## Ejercicios

---

Que creen una carpeta en el escritorio, la abran con Atom y creen la siguiente estructura:

- mi-proyecto
  - archivo1.txt
  - imagenes
    - archivo2.txt

# Editor - Shortcuts / Cheatsheet

*Shortcuts (<https://github.com/nwinkler/atom-keyboard-shortcuts>)*

## General

Some general keyboard shortcuts that I use frequently.

Command	macOS	Windows	Linux	Description
Preferences/Settings	cmd-,	ctrl-,	ctrl-,	Opens the Preferences/Settings view
Command Palette	shift-cmd-p	shift-ctrl-p	ctrl-shift-p	Opens & closes the command palette
Open File (Fuzzy)	cmd-p	ctrl-p or ctrl-t	ctrl-p	Opens the Fuzzy Finder palette in which you can search and open files
Browse Open Files	cmd-b	ctrl-b	ctrl-b	Browse tabs within the window
Grammar Selector	ctrl-shift-l	ctrl-shift-l	ctrl-shift-l	Selects the language the file is in
Markdown Preview	ctrl-shift-m	ctrl-shift-m	ctrl-shift-m	Previews the file in the Markdown format
Key Binding Resolver	cmd-.	ctrl-.	ctrl-.	Shows what keybindings the pressed key combination resolves to

# Editor - Shortcuts / Cheatsheet

*Shortcuts (<https://github.com/nwinkler/atom-keyboard-shortcuts>)*

## Window Management

Command	macOS	Windows	Linux	Description
New File	cmd-n	ctrl-n	ctrl-n	Opens an empty file in a new tab
New Window	shift-cmd-n	ctrl-shift-n	ctrl-shift-n	Opens a new editor window
Open	cmd-o	ctrl-o	ctrl-o	Shows the <i>Open File</i> dialog, which lets you select a file to open in the editor
Open Folder	cmd-shift-o	ctrl-shift-o	ctrl-shift-o	Shows the <i>Open Folder</i> dialog, which lets you select a folder to add to the editor's Tree View
Save	cmd-s	ctrl-s	ctrl-s	Saves the currently active file
Save As	shift-cmd-s	ctrl-shift-s	ctrl-shift-s	Saves the currently active file under a different name
Save All	alt-cmd-s			Saves all changed files
Close Tab	cmd-w	ctrl-w	ctrl-w	Closes the currently active tab
Close Window	shift-cmd-w	ctrl-shift-w	ctrl-shift-w	Closes the currently active editor window

# Editor - Shortcuts / Cheatsheet

*Shortcuts (<https://github.com/nwinkler/atom-keyboard-shortcuts>)*

## Editing

Command	macOS	Windows	Linux	Description
Duplicate Lines	shift-cmd-d	ctrl-shift-d	ctrl-shift-d	Duplicates the line of the current cursor position and creates a new line under it with the same contents
Delete Line	ctrl-shift-k	ctrl-shift-k	ctrl-shift-k	Deletes the current line
Move Line Up	ctrl-cmd-up	ctrl-up	ctrl-up	Moves the contents of the current cursor position up one line. If there is a line above with content, the current lines content will swap with the one above it.
Move Line Down	ctrl-cmd-down	ctrl-down	ctrl-down	Moves the contents of the current cursor position down one line. If there is a line below with content, the line's content will swap with the one below it.
Find/Replace	cmd-f	ctrl-f	ctrl-f	Opens up the Find/Replace panel
Find Next	cmd-g	F3	F3	Toggles forward through the results of the current buffer in the file while the Find/Replace panel is active
Find Previous	shift-cmd-g	shift-F3	shift-F3	Toggles backward through the results of the current buffer in the file while the Find/Replace panel is active

# Editor - Packages, Plugins, AddOns

---

*Buscar: best atom packages*

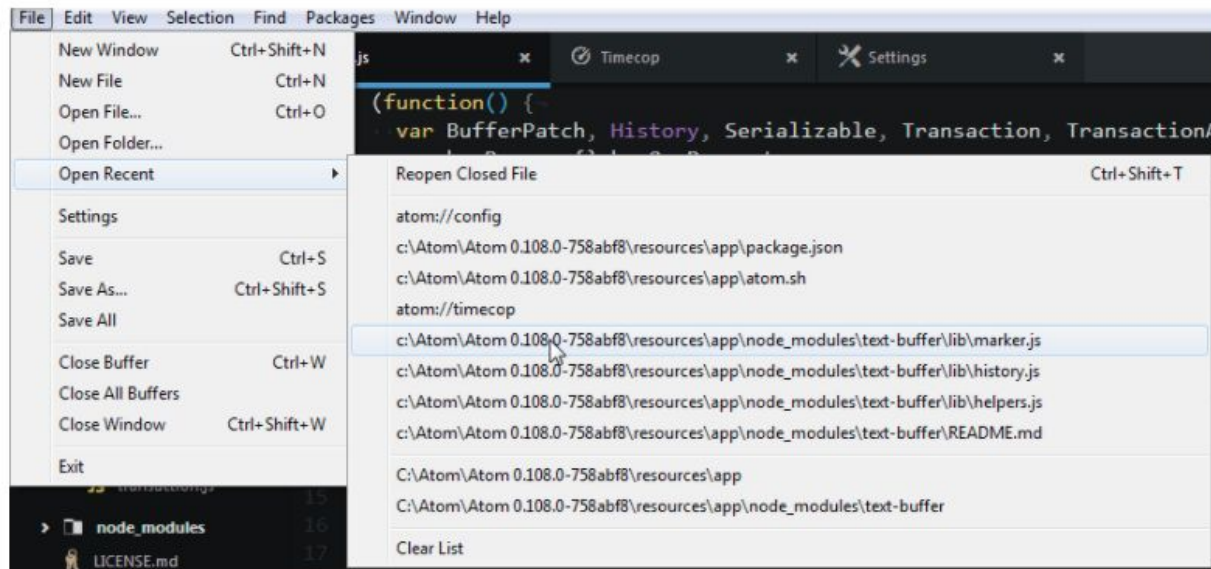
Package installation is simple. You can either:

- 1 open the **+ Install** panel in the **Settings** tab then search for a package by name, or
- 2 install from the command line using `apm install <package-name>`.

# Editor - Packages, Plugins, AddOns

<https://atom.io/packages/open-recent>

## Screenshots



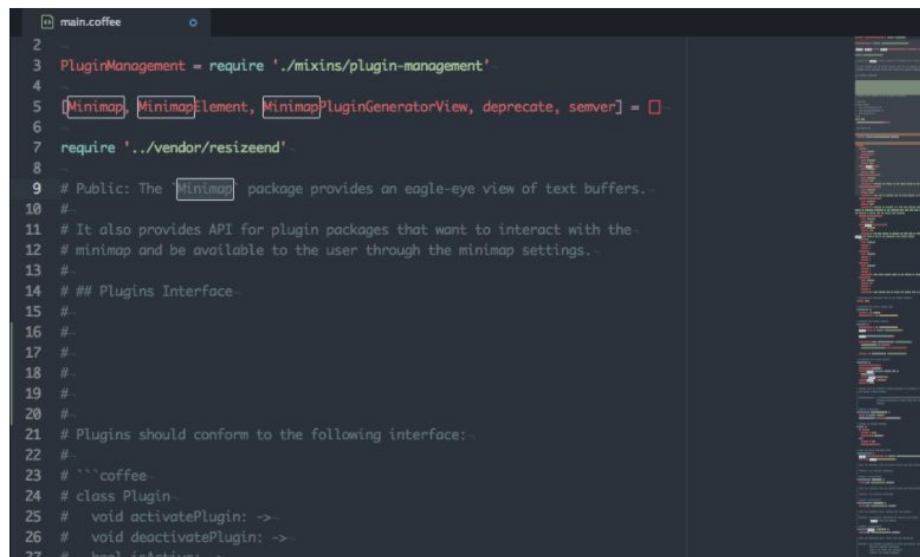


# Editor - Packages, Plugins, AddOns

<https://atom.io/packages/minimap>

## Minimap package

A preview of the full source code.

A screenshot of the Atom text editor showing the source code for the 'minimap' package. The editor has a dark theme. The file is named 'main.coffee'. The code is a CoffeeScript file that defines the package's interface and dependencies. It includes comments in Chinese and English. The code is as follows:

```
1 main.coffee
2 ..
3 PluginManagement = require './mixins/plugin-management'
4
5 [Minimap, MinimapElement, MinimapPluginGeneratorView, deprecate, semver] = []
6
7 require './vendor/resizeend'
8
9 # Public: The 'Minimap' package provides an eagle-eye view of text buffers..
10 #
11 # It also provides API for plugin packages that want to interact with the
12 # minimap and be available to the user through the minimap settings..
13 #
14 ## Plugins Interface..
15 #
16 #
17 #
18 #
19 #
20 #
21 # Plugins should conform to the following interface:..
22 #
23 ```coffee
24 # class Plugin
25 #   void activatePlugin: ->
26 #   void deactivatePlugin: ->
27 #   bool isActive: ->
```

# Editor - Packages, Plugins, AddOns

<https://atom.io/packages/highlight-selected>

```
minimap-highlight-select... x
1  {View, EditorView} = require 'atom'
2
3  module.exports = ->
4    highlightSelectedPackage =
5      • atom.packages.getLoadedPackage('highlight-selected')
6      minimapPackage = atom.packages.getLoadedPackage('minimap')
7
8      minimap = require (minimapPackage.path)
9      highlightSelected = require (highlightSelectedPackage.path)
10     HighlightedAreaView = require (highlightSelectedPackage.path +
11     • '/lib/highlighted-area-view')
12
13   class MinimapHighlightSelectedView extends HighlightedAreaView
14     constructor: ->
15       super
16       @paneView = @editorView.getPane()
17       @subscribe @paneView.model.$activeItem, @onActiveItemChanged
18
19     attach: ->
20       @subscribe @editorView, "selection:changed", @handleSelection
21
22     minimapView = @getMinimap()
```

# Editor - Packages, Plugins, AddOns

---

*Buscamos otros y los instalamos*

## *Classroom Questions*

## *Línea de comandos vs Interfaz Gráfica*



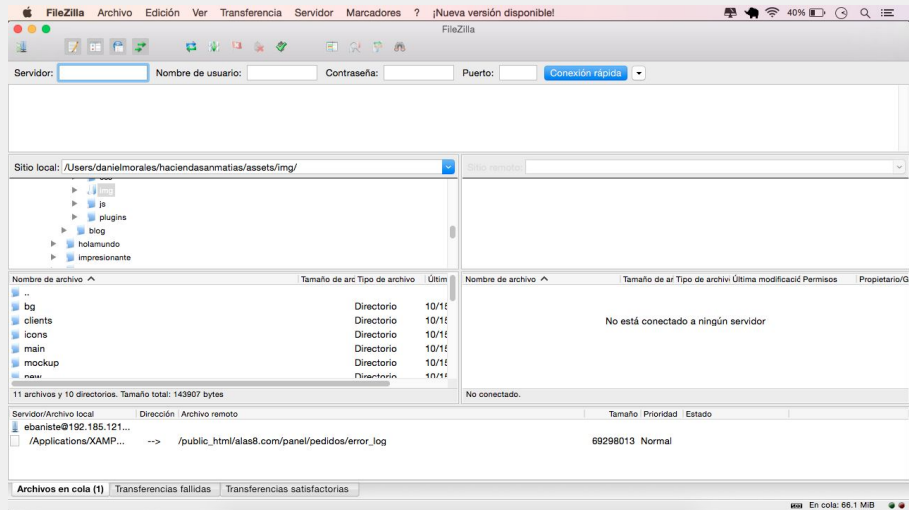
# Línea de Comandos

## *Línea de comandos vs Interfaz Gráfica*

```
financelas — rails s — rails — ruby — 80x24

crud-products      product-categories  sinatra-reto2
cv                 producthunt         sinatra-reto3
dudas_clases.rb    puzzle             sinatra-reto4
ex3.rb             rails-testing       sinatra-reto5
index.rb           reto-fiscalia       sinatra-reto6
jaguar            reto5.rb            sinatra-reto7
make-it-real-blog  retos-17feb         sinatra-reto8
many_to_many       retos-javascript    sinatra-reto9
mi_app             retos-jquery         store
notas.rb           retos-rails          superblog
notas.txt          ruby                 test_rails
notes             ruby-intro           testing
password_validation.rb ruby-server          wikilinks

→ makeitreal git:(master) ✕ cd ..
→ ~ git:(master) ✕ cd financelas
→ financelas git:(master) rails s
=> Booting WEBrick
=> Rails 4.2.3 application starting in development on http://localhost:3000
=> Run 'rails server -h' for more startup options
=> Ctrl-C to shutdown server
[2017-04-29 14:18:55] INFO  WEBrick 1.3.1
[2017-04-29 14:18:55] INFO  ruby 2.3.0 (2015-12-25) [x86_64-darwin14]
[2017-04-29 14:18:55] INFO  WEBrick::HTTPServer#start: pid=3561 port=3000
```



## *Desventajas*

- 1- Poco intuitiva*
- 2- Comandos poco obvios*
- 3- No es rica visualmente*
- 4- No es amistoso para los novatos*

## *Ventajas*

- 1- Control total del sistema*
- 2- Potente para muchas tareas*
- 3- Menos memoria*
- 4- Conciso y poderoso*
- 5- Fácil de automatizar*
- 6- Preferido por expertos*
- 7- Te ves como programador Pro!*



*¿Que es?*

*Aplicación que nos permite escribir y ejecutar comandos sobre el sistema operativo sin necesidad de usar el Mouse.*

## *¿Que se puede hacer?*

La **línea de comandos** es una aplicación en la que puedes escribir y ejecutar comandos para realizar tareas como:

- Navegar por las carpetas de tu computador.
- Manipular archivos (crear, editar, copiar, mover y eliminar).
- Conectarte a servidores remotos.
- Crear **scripts** que te ayuden con tus tareas diarias.

Y mucho más!

## *¿Cómo abrirla?*

### **Windows**

Ir al menú Inicio → Todos los programas → Accesorios → Command Prompt

### **Mac OS X**

Aplicaciones → Servicios → Terminal

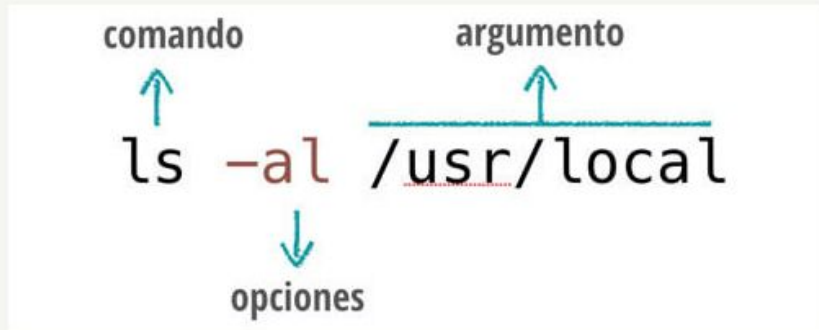
### **Linux**

Está probablemente en Aplicaciones → Accesorios → Terminal, pero eso depende de tu distribución. Si no lo encuentras, Googlealo :)

## Anatomía de un comando

Todos los comandos se componen de:

- Un **nombre** con el que se invoca el comando.
- **Opciones** que modifican el comportamiento del comando. Son opcionales.
- **Argumentos** sobre los que actúa el comando. También opcionales.



Anatomía del comando `ls` que lista las carpetas y archivos (en este caso de la carpeta `/usr/local`). La opción `-a` muestra los archivos ocultos y `-l` muestra los detalles de cada carpeta/archivo.

# Línea de Comandos

## *Prompt*

### Prompt

Ahora deberías ver una ventana blanca o negra que está esperando tus órdenes.

Si estás en Mac o Linux, probablemente verás `$` , así:

```
$
```

En Windows, es un signo así `>` , como este:

```
>
```

Cada comando será precedido por este signo y un espacio, pero no tienes que escribirlo. Tu computadora lo hará por ti :)

Sólo una pequeña nota: en tu caso, tal vez hay algo como `C:\Users\ola>` o `Olas-MacBook-Air:~`  
`ola$` antes del prompt y eso es 100% correcto. En este tutorial lo simplificaremos lo más posible.

## *Tu Primer Comando*

### Tu primer comando (¡YAY!)

Vamos a empezar con algo simple. Escribe este comando:

```
$ whoami
```

o

```
> whoami
```

Y luego oprime la tecla Enter. Este es el resultado:

```
$ whoami olasitarska
```

Como puedes ver, la computadora sólo te presentó tu nombre de usuario. Bien, ¿eh? :)

Trata de escribir cada comando, no copies y pegues. ¡Te acordarás más de esta manera!

# Línea de Comandos

## Directorio actual

Sería bueno saber dónde estamos ahora, ¿cierto? Vamos a ver. Escribe este comando y oprime Enter:

```
$ pwd  
/Users/olasitarska
```

Si estás en Windows:

```
> cd  
C:\Users\olasitarska
```

Probablemente verás algo similar en tu máquina. Una vez que abres la línea de comandos generalmente empiezas en el directorio home de tu usuario.

Nota: 'pwd' significa 'print working directory' - en español, 'mostrar directorio de trabajo'.

# Línea de Comandos

## Lista de archivos y directorios

¿Qué hay aquí? Sería bueno saber. Veamos:

```
$ ls
Applications
Desktop
Downloads
Music
...
```

Windows:

```
> dir
Directory of C:\Users\olasitarska
05/08/2014 07:28 PM <DIR> Applications
05/08/2014 07:28 PM <DIR> Desktop
05/08/2014 07:28 PM <DIR> Downloads
05/08/2014 07:28 PM <DIR> Music
...
```



# Línea de Comandos

## Cambia el directorio actual

¿Quizás podemos ir a nuestro escritorio?

```
$ cd Desktop
```

Windows:

```
> cd Desktop
```

Comprueba si realmente ha cambiado:

```
$ pwd  
/Users/olasitarska/Desktop
```

Windows:

```
> cd  
C:\Users\olasitarska\Desktop
```

# Línea de Comandos - Rutas

---

## Rutas relativas y absolutas

Para las rutas de las carpetas puedes utilizar rutas relativas o absolutas.

Una **ruta relativa** toma como referencia la carpeta actual.

Una **ruta absoluta** inicia con `/` y se refiere a la ruta completa desde la carpeta raíz de tu sistema.

Por ejemplo, si nos encontramos en la carpeta `/Users/germanescobar`, la **ruta relativa** `Projects/images` se refiere a la **ruta absoluta** `/Users/germanescobar/Projects/images`.

# Línea de Comandos - Rutas

## La carpeta del usuario

Cuando ingresas a la **línea de comandos** te vas a encontrar en la carpeta raíz de tu usuario (generalmente en `/Users/tu_nombre_de_usuario`).

Puedes volver a esta carpeta en cualquier momento utilizando el comando:

```
$ cd ~
```

Ese caracter  es difícil de encontrar en el teclado a veces. Si no lo encuentras pide ayuda!

# Línea de Comandos - Rutas

## La carpeta padre

Existen dos nombres de carpeta especiales que se utilizan para referirse a la carpeta actual y a la carpeta padre.:

- `.` se refiere a la carpeta actual
- `..` se refiere a la carpeta padre.

Por ejemplo, si nos encontramos en la ruta `/Users/germanescobar/Projects` y ejecutamos:

```
$ cd ..
```

La nueva ubicación será `/Users/germanescobar`.

Puedes utilizar `..` en tus **rutas relativas**. Por ejemplo, si te encuentras en una carpeta `images` y quieres ir a una carpeta `fonts` que está al lado de `images` puedes ejecutar:

```
$ cd ../fonts
```

Esa ruta `../fonts` lo que está diciendo es: vaya a la carpeta padre y después ingrese a la carpeta `fonts`.

# Línea de Comandos

## Crear directorio

¿Qué tal si creamos un directorio de Django Girls en tu escritorio? Puedes hacerlo de esta manera:

```
$ mkdir.djangogirls
```

Windows:

```
> mkdir.djangogirls
```

Este pequeño comando creará una carpeta con el nombre `djangogirls` en tu escritorio. ¡Puedes comprobar si está allí buscando en tu escritorio o ejecutando el comando `ls` (si estás usando Mac o Linux) o `dir` (si estás usando Windows)! Inténtalo :)

Pro tip: Si no quieres escribir una y otra vez los mismos comandos, prueba oprimiendo la `flecha arriba` y `flecha abajo` de tu teclado para ver recientes comandos utilizados.

# Línea de Comandos

## Abriendo archivos y carpetas en Atom

**Atom** viene con una aplicación de la **línea de comandos** llamada `atom`. Puedes utilizar ese comando para abrir un archivo o una carpeta en **Atom**.

Por ejemplo, el siguiente comando abre el archivo `archivo1.txt`:

```
$ atom archivo1.txt
```

Para abrir la carpeta actual:

```
$ atom .
```

## *Crear archivo*

*Comando:*

*\$ touch index.html*

*-> touch index.html*

## Moviendo archivos y carpetas

Para mover un archivo de ubicación utiliza el comando `mv` seguido del nombre del archivo y la ubicación donde lo quieres mover (separándolos por espacio).

Por ejemplo, el siguiente comando movería el archivo `archivo1.txt` a la carpeta `mi-carpeta` en la ruta actual:

```
$ mv archivo1.txt mi-carpeta
```

Mover una carpeta es muy parecido a mover un archivo, utiliza el comando `mv` seguido del nombre de la carpeta que quieres mover y la carpeta destino.

Por ejemplo, el siguiente comando movería la carpeta `mi-carpeta` a la carpeta `otra-carpeta` en la misma ubicación:

```
$ mv mi-carpeta otra-carpeta
```



## Copiando archivos y carpetas

Para copiar un archivo a otra ubicación utiliza el comando `cp` seguido del nombre del archivo y la ubicación a la que quieres copiar el archivo.

Por ejemplo, asumiendo que `archivo1.txt` y `mi-carpeta` existen en la ubicación actual puedes utilizar el siguiente comando para copiar `archivo1.txt` a `mi-carpeta`:

```
$ cp archivo1.txt mi-carpeta
```

Para copiar una carpeta a otra ubicación utiliza el comando `cp -r` seguido del nombre de la carpeta que quieres mover y la ruta a donde la quieres mover.

Por ejemplo, el siguiente comando **copia** la carpeta `mi-carpeta` a la carpeta `otra-carpeta` en la misma ubicación:

```
$ cp -r mi-carpeta otra-carpeta
```

## *Eliminar*

Ahora es hora de eliminar el directorio `djangoirls` .

**Atención:** Eliminar archivos utilizando `del` , `rmdir` o `rm` hace que no puedan recuperarse, lo que significa que los *archivos borrados desaparecerán para siempre* Debes ser muy cuidadosa con este comando.

```
$ rm -r djangoirls
```

Windows:

```
> rmdir/s djangoirls  
djangoirls, ¿Estás seguro <Y/N>? Y
```

Hecho! Asegurémonos que en verdad fueron borrados, vamos a ver:

```
$ ls
```

Windows:

```
> dir
```

## Trucos y consejos

---

Si oprimas la flecha arriba **↑** te van a aparecer los últimos comandos que has ingresado. Muy útil para repetir un comando que ya has ingresado. Si te pasas te puedes devolver con la flecha hacia abajo:

Puedes autocompletar las rutas ingresando las primeras letras y oprimiendo la tecla **tab** . Ahorra mucho tiempo!

# Línea de Comandos

## Salida

¡Esto es todo por ahora! Ahora puedes cerrar la línea de comandos sin problemas. Vamos a hacerlo al estilo hacker, ¿bien? :)

```
$ exit
```

Windows:

```
> exit
```

Genial, ¿no? :)

# Línea de Comandos

## Índice

Aquí hay una lista de algunos comandos útiles:

Comando (Windows)	Comando (Mac OS / Linux)	Descripción	Ejemplo
exit	exit	Cierra la ventana	<b>exit</b>
cd	cd	Cambia el directorio	<b>cd test</b>
dir	ls	Lista directorios/archivos	<b>dir</b>
copy	cp	Copia de archivos	<b>copy c:\test\test.txt c:\windows\test.txt</b>
move	mv	Mueve archivos	<b>move c:\test\test.txt c:\windows\test.txt</b>
mkdir	mkdir	Crea un nuevo directorio	<b>mkdir testdirectory</b>
del	rm	Elimina archivos/directorios	<b>del c:\test\test.txt</b>

## *Ejercicio (Parte 1)*

- 1- Entra a la terminal*
- 2- Crea una carpeta llamada makeitreal*
- 3- Dentro de esa carpeta, crea una nueva carpeta llamada dia\_2*
- 4- Dentro de la carpeta dia\_2, crea dos archivos llamados:  
hola.html y chao.html*

## *Ejercicio (Parte 2)*

*1- Ve a la carpeta dia\_2 y elimina el archivo chao.html*

*2- Sal de la consola o línea de comandos*

*Nota: sigue usando la carpeta /makeitreal para añadir todos los recursos de las clases*

## *Ejercicio (Parte 3)*

### Ejercicios:

- Crear una carpeta con la siguiente estructura en el escritorio:

```
- mi-proyecto
  - archivo1.txt
  - imagenes
    - archivo2.txt
```

También que muevan el `archivo1.txt` a `imagenes` y el `archivo2.txt` una carpeta atrás. Después que **copien** el archivo `archivo1.txt` en `imagenes` y el `archivo2.txt` una carpeta atrás.

Que cierren la consola, la vuelvan a abrir y vayan a la carpeta que acabaron de crear, y la abran en Atom. (Esto lo van a repetir mucho durante el curso y necesitamos que lo hagan rápido).



# Usos de Editor y CLI

---

*Será una constante usar el editor y la línea de comandos (CLI). Practica y familiarízate!*

# Usos de Editor y CLI

---

*Será una constante usar el editor y la línea de comandos (CLI). Practica y familiarízate!*

## *Classroom Questions*

# Git y Github

---

*<https://git-scm.com/book/es/v1>*