

# Emacs on Rails – Guia

## Índice

Requirements.....	1
Antes de começar.....	1
Instalação.....	2
Plugins utilizados.....	2
Cheatsheet.....	2
Movimentação.....	2
Busca.....	3
Seleção.....	3
Edição.....	3
Controle de janelas.....	4
Gerenciando arquivos (dired mode).....	4
Comandos do Rails.....	4
Comandos essenciais.....	5
Projectile.....	5
RSpec-mode.....	6
MaGIT.....	6
Dicas.....	6
Byebug no rails server \ rspec.....	6
Auto-complete avançado.....	6
Ver os comandos pelo emacs.....	6
Criando seu próprio snippet.....	7
Dando re-undo no emacs.....	7
Dando push da melhor maneira.....	7
Selecionando uma opção no code completion (company).....	7
Alterando tema.....	7
Alterando fonte.....	7
Usa docker na sua aplicação? Veja:.....	7

## Requirements

- Emacs 26
- silversearcher-ag (apt install silversearcher-ag)
- elpa-counsel
- Rbenv
- Linux

## Antes de começar

Recomendo que antes de iniciar, faça o tutorial que vem no Emacs. Assim terá uma maior noção dos comandos básicos.

# Instalação

1 -Bote os arquivos .emacs e .emacs.d na pasta home.

2 – Aperta M-x e execute: all-the-icons-install-fonts

3 - Abre o emacs, vá para a pasta do seu projeto com C-x C-f, use M-x para rodar um comando, e rode: projectile-discover-projects-in-directory.

## Plugins utilizados

- **projectile** – Gerencia projetos no Emacs. Comandos como abrir terminal, grep, etc.
- **projectile-rails** – Extensão do projectile, com comandos para Rails. Comandos como rails server, console, navegar entre models, etc
- **helm-projectile**– Helm é um plugin que estende as buscar do emacs, com ele, a gente consegue ter buscar bem mais poderosas no projeto \ buffers
- **rspec-mode** – Plugin pra executar RSPEC de maneira pratica no meio do código
- **yasnipet** – Snippets pro Emacs. Tu pode usar apertando tab(não recomendado) ou C-q (Recomendado)
- **expand-region** – Ferramenta top para selecionar coisas (Usa apertando M-2)
- **flycheck** – Avisa sobre erros no código e também mostra erros do rubocop
- **robe** – Integração com Ruby, diz o que os métodos fazem, etc
- **multiple-cursors** – Clássico, nem preciso falar nada
- **magit** – Coisa mais linda que existe, controle completo do GIT com integração direta com seu código
- **ace-window** – Para controlar as janelas

## Cheatsheet

### Movimentação

C-f	Move o ponteiro um caractere para frente
C-b	Move o ponteiro um caractere para tras
M-f	Move o ponteiro uma palavra pra frente
M-b	Move o ponteiro uma palavra pra trás
M-<	Vai para o inicio do arquivo
M->	Vai para o final do arquivo
C-p	Move o ponteiro um caractere para cima
M-p	Move o ponteiro um caractere para baixo
C-M-p	Vai para o inicio da sentença
C-M-n	Vai para o final da sentença
M-m	Vai para o primeiro caractere da linha

C-a	Vai para o começo da linha
C-e	Vai para o final da linha
C-o	Cria uma nova linha abaixo
C-S-o	Cria uma nova linha cima
C-S-; + letra	Vai para uma letra especifica em qualquer janela
M-.	Vai para a classe que o ponteiro está encima.

## Busca

C-s	Busca no arquivo atual indo para frente
C-r	Busca no arquivo atual indo para trás
C-c pss	Busca um texto em todo projeto (helm-projectile)
(F3) enquanto usa o C-c p ss	Salva todos os arquivos buscados em um buffer (helm-projectile)
(F4) enquanto usa o C-c p ss	Abre a lista de itens achados em 1 só buffer, para editar em todos os lugares ao mesmo tempo
C-c p f	Busca um arquivo no projeto atual (projectile)

## Seleção

C-SPC	Inicia a seleção
C-SPC C-SPC ou C-g	Termina seleção
M-2	Seleciona palavra → colchetes → método (Só ir repetindo o M-2) (expand-region) (Muito bom)
C - >	Multiple-cursors – Para baixo
C - <	Multiple-cursors – Para cima

## Edição

M-BACKSPACE	Deleta uma palavra para trás
BACKSPACE	Deleta um caractere para trás
C-d	Deleta um caractere para frente
M-d	Deleta uma palavra para frente
C-k	Deleta até o final da linha
C-a C-k	Deleta uma linha (2 comandos)
C-3 C-a C-k	Deleta 3 linhas (exemplo) C-3 = repete 3 vezes

C-S-k	Deleta até o fechamento de chaves, parênteses, etc, super útil.
C-c t	Troca o formato de do \ end para { }
C-c f	Cria um método com o texto que está no cursor atualmente

## Controle de janelas

C-x 3	Divide a janela verticalmente
C-x 2	Divide a janela horizontalmente
M-o	Vai para outra janela
C-x 1	Fecha todas as janelas, menos a atual
C-x 0	Fecha a janela atual
C-x k	Fecha janela atual

## Gerenciando arquivos (dired mode)

C-x C-f	Procura uma pasta para abrir
+	Cria uma nova pasta (no gerenciador de arquivos)
C	Copia um arquivo
R	Renomeia um arquivo
D	Marca um arquivo para ser deletado
x	Executa a ação de deletar
C-x b	Troca de buffer (arquivo)

## Comandos do Rails

Mais comandos em: <https://github.com/asok/projectile-rails>

C-c r m	Procura um model no projeto
C-c r M	Vai para o model referente ao resource aberto
C-c r c	Procura um controller
C-c r C	Vai para o controller referente ao resource aberto
C-c r T	Troca entre Arquivo \ Teste
C-c T	Cria um teste para o arquivo aberto atualmente
C-c r u	Procura por fixtures no projeto

C-c r r	Rails Console
C-c r R	Rails Server
C-c r ! g	Rails generate
C-c r ! d	Rails destroy
C-c r g s	Vai para o seeds
C-c r g r	Vai para o routes
C-c r x	Cria uma partial com o código selecionar
C-c r u	Procura pelas fixtures do projeto
C-c r a	Procura pelos locais
C-c r v	Procura por views
C-c r V	Procura por views relacionadas com o resource aberto.

## Comandos essenciais

C-x s	Salva o arquivo
C-=	Indentar arquivo atual
C-S-=	Remover whitespaces do arquivo atual
C-)	Roda o rubocop -a no arquivo atual
C-g	Cancela ação atual
C-x (	Começa a gravar um macro (Pesquise: Macro emacs guide)
C-x )	Termina de gravar um macro
C-x e	Executa o macro
C-3 C-x e	Executa o macro 3 vezes.
C-w	Recorta
M-w	Copia
C-y	Cola
M-y	Navega no histórico do clipboard
M-- M-y	Navega no histórico do clipboard (inverso)
C-x u OU C-/ OU C-;	Undo (Estilo CTRL + Z)

## Projectile

C-c p p	Busca projeto (Depois de selecionar, geralmente se usa o F ou o S)
C-c p f	Busca arquivo no projeto (Igual CTRL + P nos editores convencionais)
C-c p k	Fecha todos os arquivos do projeto

C-c p x s	Abre o terminal no projeto
-----------	----------------------------

## RSpec-mode

C-c , v	Roda os testes do arquivo atual
C-c , r	Roda os últimos testes executados.
C-c , s	Roda o teste de onde o ponteiro está.
C-c , a	Roda os testes de todo o projeto
C-c , f	Roda os testes que falharam na ultima execução

## MaGIT

C-c g	Abre o magit
cc	Criar um commit
h	Ver os comandos disponíveis
z	Stash
k	Remover modificação
s	Stage
S	Stage all
F	Pull
P	Push
f	Fetch
b	Branch (checkout, etc)

## Dicas

### Byebug no rails server \ rspec

Quando o byebug aparece, não é possível editar. Use C-x C-q para tirar o readonly e conseguir utilizar normalmente

### Auto-complete avançado

O Robe possui uma ferramenta avançada para auto-complete, para ela funcionar, é preciso estar com o console aberto. Basta usar M-x robe-start RET, para iniciar

### Ver os comandos pelo emacs

Se você quer saber mais comandos, na barra inferior há todos os plugins ativos, basta clicar neles que aparece os comandos disponíveis.

## Criando seu próprio snippet

Rode M-x yas-new-snippet, faça do jeito que quer, e de C-c C-c para salvar.

## Dando re-undo no emacs

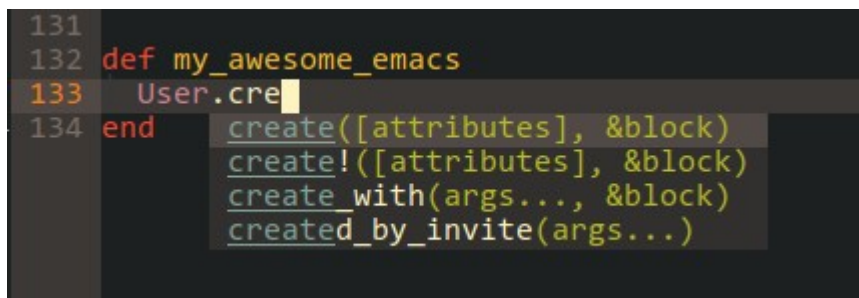
Por padrão, o Emacs não possui re-undo. Para fazer isso, após dar o UNDO, move um caractere para frente para “resetar o undo” e de undo novamente, dessa vez, vai dar como fosse re-undo dos editor convencionais.

## Dando push da melhor maneira

Ao usar o git push pelo Magit com o comando: P u, é perguntado o local que você deseja subir a branch. Ao apertar M-n duas vezes, é completado com **origin/nome\_da\_branch\_atual**.

## Selecionando uma opção no code completion (company)

Quando você se deparar com a seguinte situação:



```

131
132 def my_awesome_emacs
133   User.create
134 end
    
```

The screenshot shows a code completion menu in Emacs. The current line is 133, and the cursor is at the end of 'User.create'. The completion menu lists several options: 'create([attributes], &block)', 'create!([attributes], &block)', 'create\_with(args..., &block)', and 'created\_by\_invite(args...)'. The first option is highlighted.

Basta usar M-n e M-p para mudar a opção, é muito mais ergonômico que usar as setas.

## Alterando tema

Basta rodar M-x customize-themes.

## Alterando fonte

Basta ir em options → Set Default Font, e depois de trocar, vá em options → Save Options

## Usa docker na sua aplicação? Veja:

- <https://github.com/asok/projectile-rails#the-global-mode>
- <https://github.com/pezra/rspec-mode#docker>

Obs: recomendo desativar o flycheck, ou manter uma versão do ruby na maquina com o bundle atualizado.