

かなりすごい発表（かなり）

supermomonga

自己紹介



supermomonga



supermomonga

<http://blog.supermomonga.com>

twitter.com/supermomonga

github.com/supermomonga



supermomonga

フリーランスでRailsとかやってる

作ったプラグインの紹介

作ったプラグイン (1/2)

音や通知に関するものが多め

- Shaberu.vim ... Vimがしゃべるやつ (音声合成)
- jazzradio.vim ... Vimでジャズが聴ける (ネットラジオ)
- skyfm.vim ... Skyrimではないやつ (Skyrimではない)
- mplayer.vim ... mplayerを操作するライブラリ (未公開)
- ThingsPast.vim ... 通知センター (Mac OS X inspired)

作ったプラグイン (2/2)

VimShellの拡張プラグインもある

- vimshell-inline-history ... VimShellが便利になるやつ
- vimshell-kawaii ... VimShellがかわいくなるやつ
- vimshell-wakeup ... VimShellが便利になるやつ
- vimshell-pure ... VimShellが便利になるやつ

おわり (おわり)

今日のテーマ

Vimは音でもっと便利になる

Sound makes vim more benrily



(Vimが便利である様子)

どういふことか

「通知」に関する例

我々がプログラミングをしている時
身体のI/Oはどうなっているでしょうか

インプット = ディスプレイ => 両目

アウトプット = 両手 => キーボード

両手と両目をフル稼働させている



これは両手と両目をフル稼働させている人です

このとき

もっと多くの情報を
Vimから得たいとしたら？

例えば「通知」

For example

- TweetVim ... リプライやDMなどの受信を通知
- J6uil.vim ... チャットルームでの新規発言を通知
- Watchdogs ... RSpecを実行して、エラーがあれば通知
- VimShell ... git clone などの時間がかかるコマンドの終了を通知

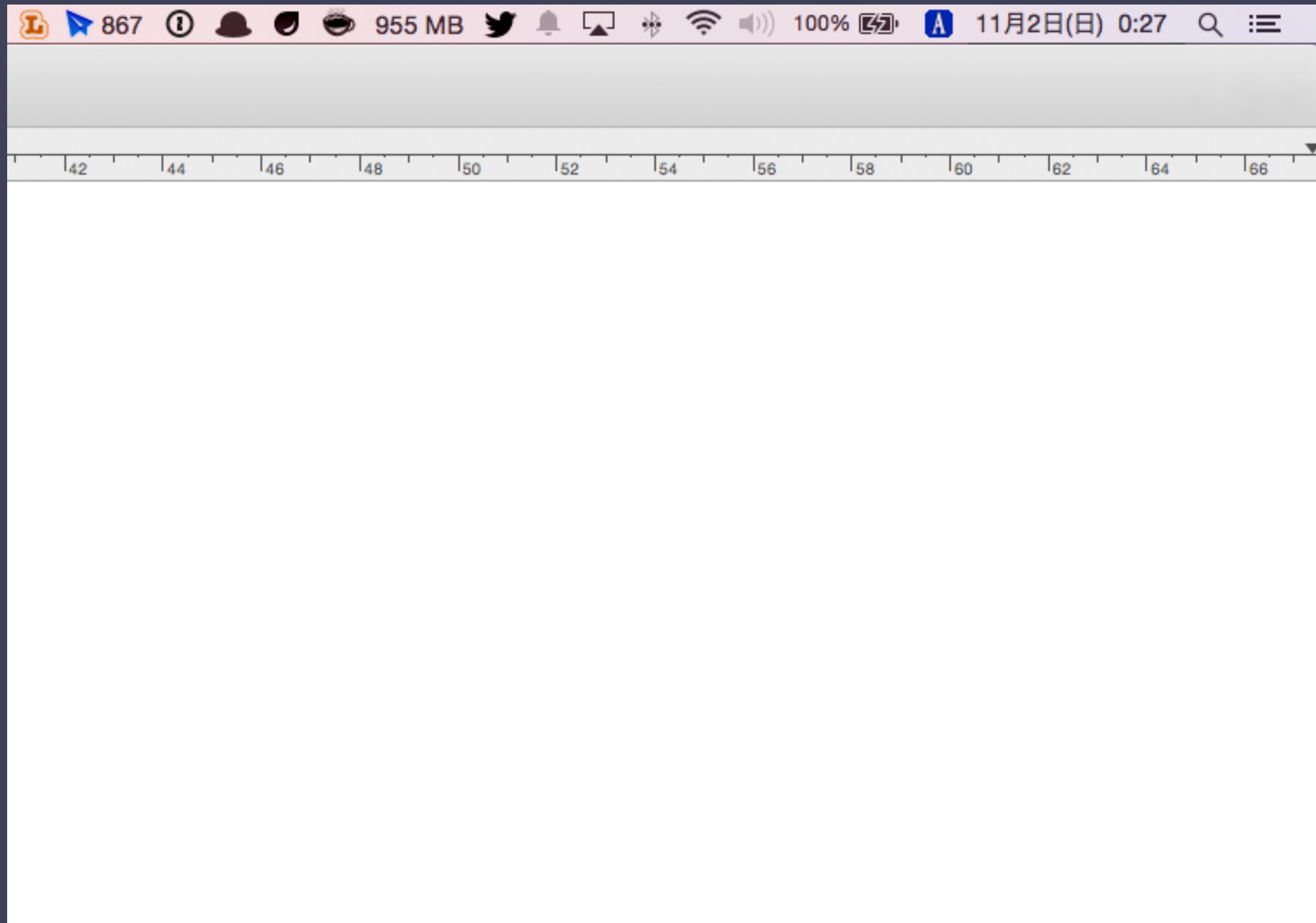
そこで

「通知」を一元管理するプラグイン
「ThingsPast.vim」をつくった

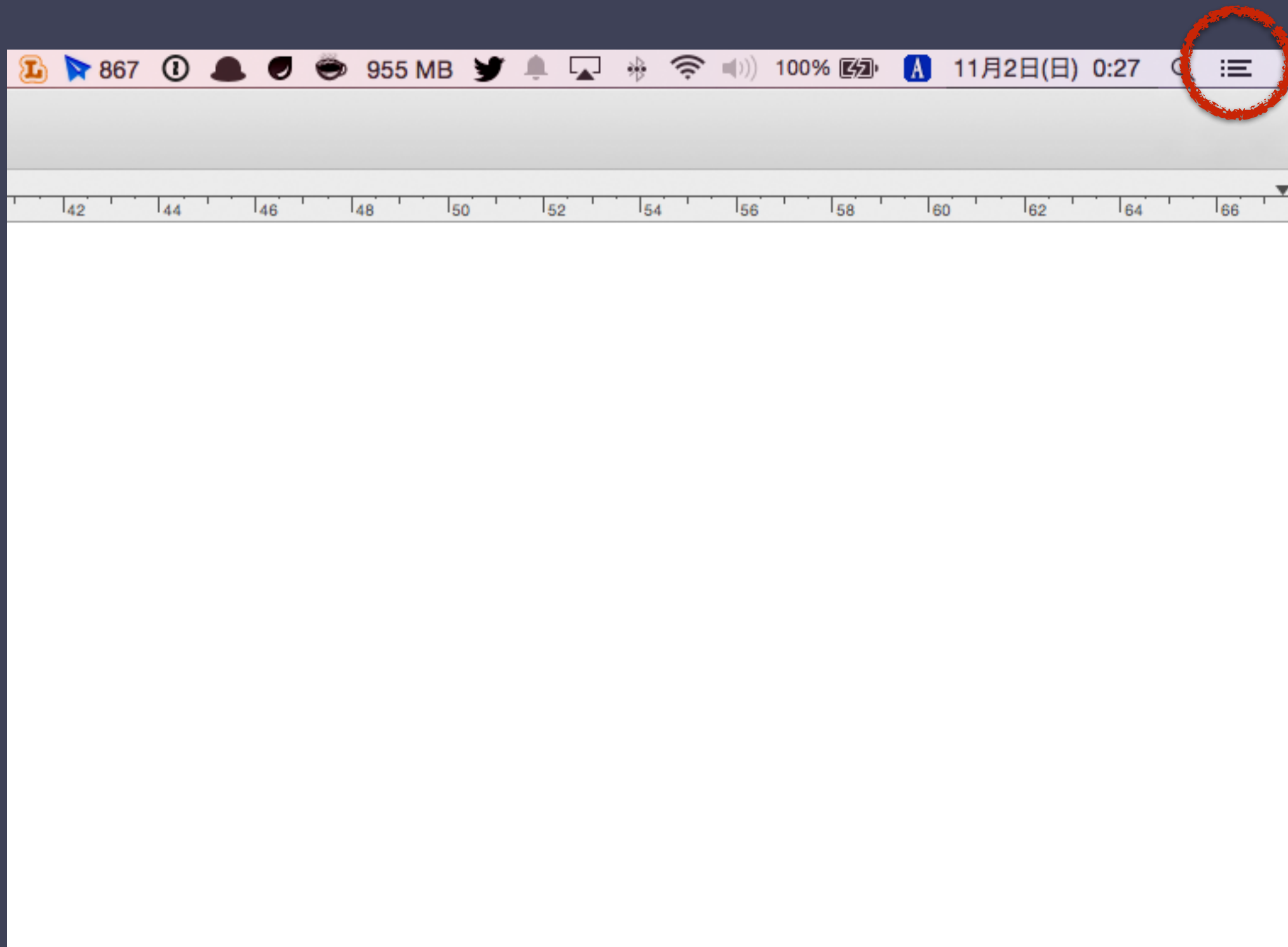
ThingsPast とは

Mac OS Xの「通知センター」を Vim内で再現するプラグイン

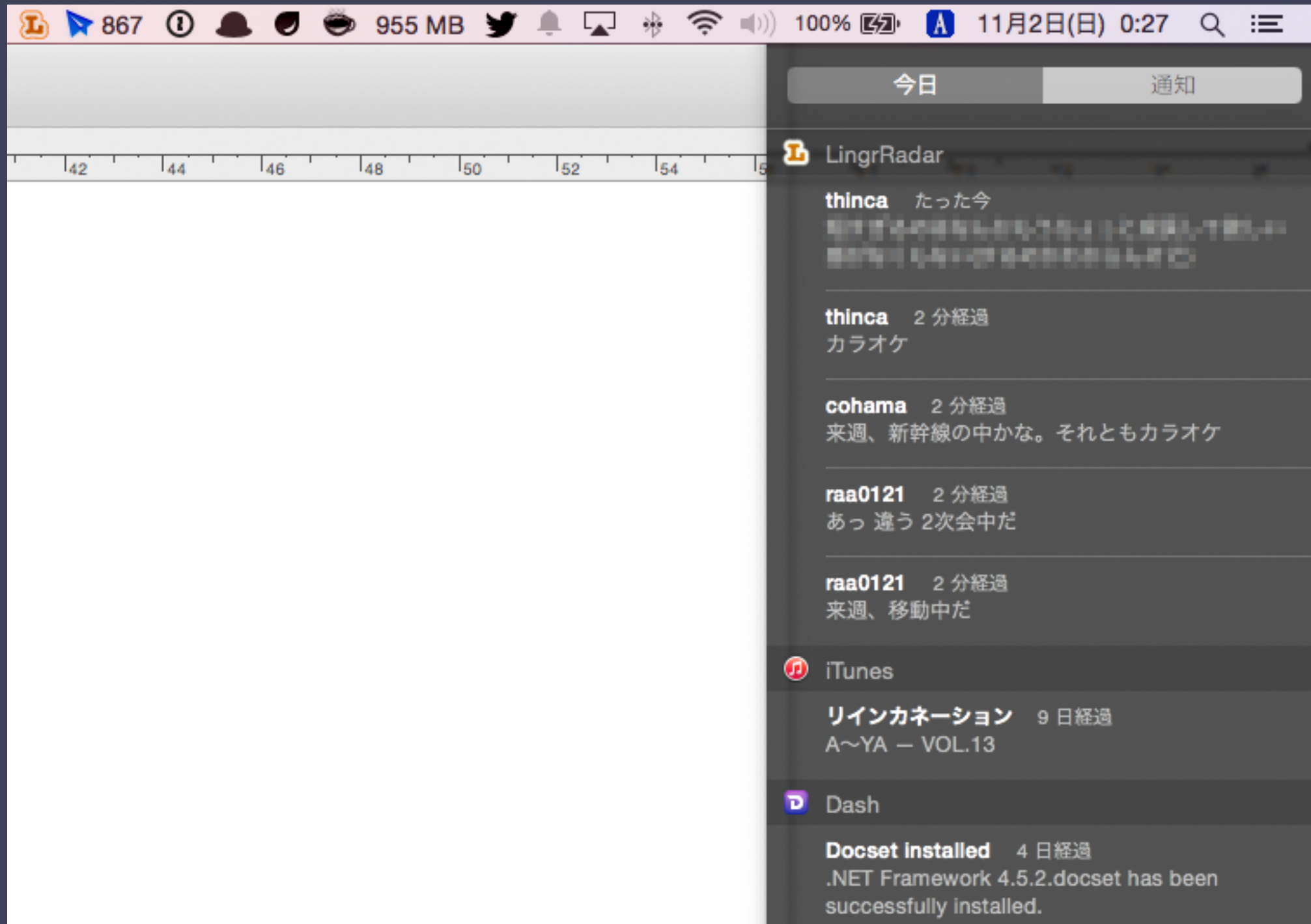
Mac OS X 通知センター



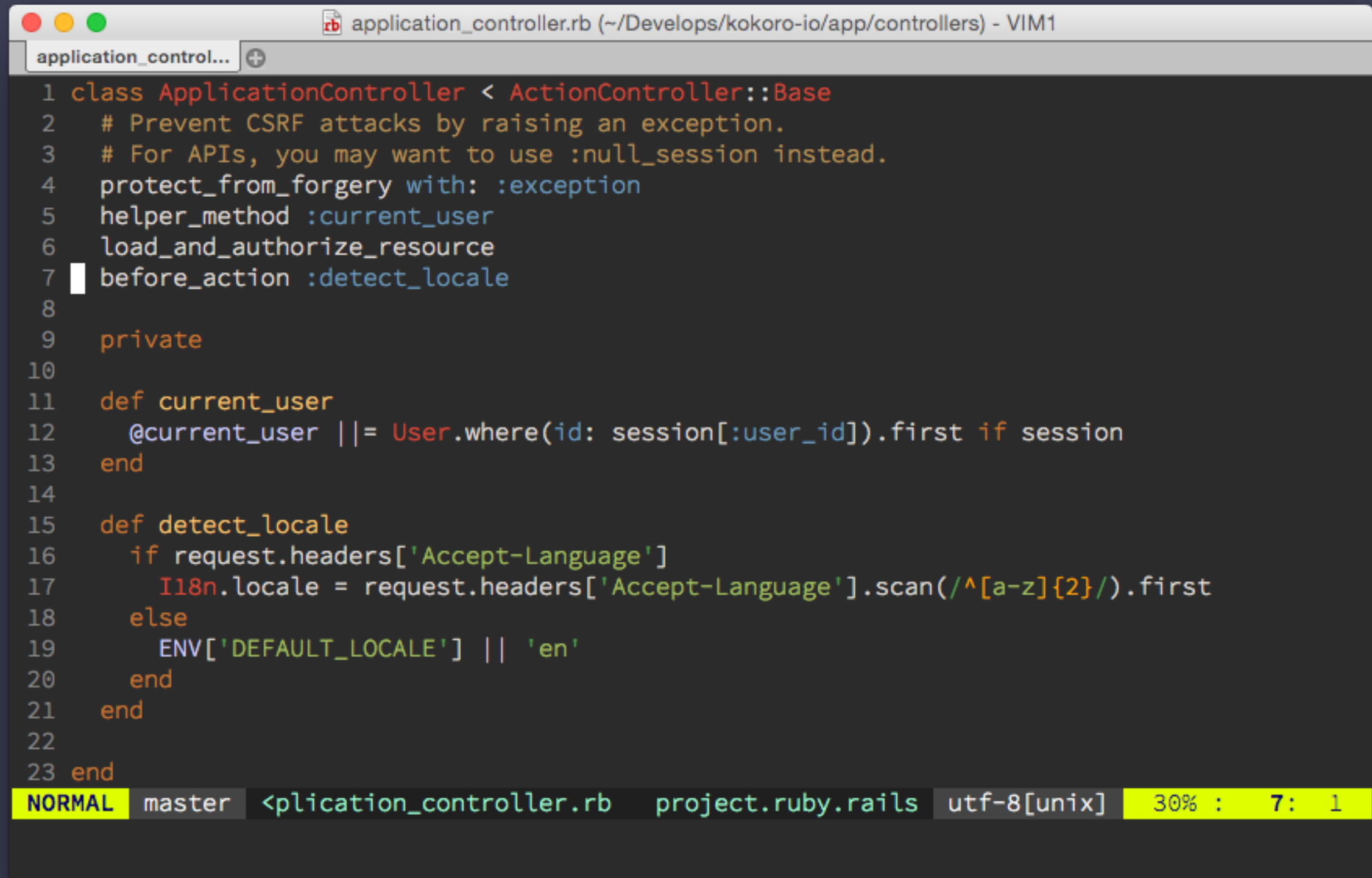
Mac OS X 通知センター



Mac OS X 通知センター



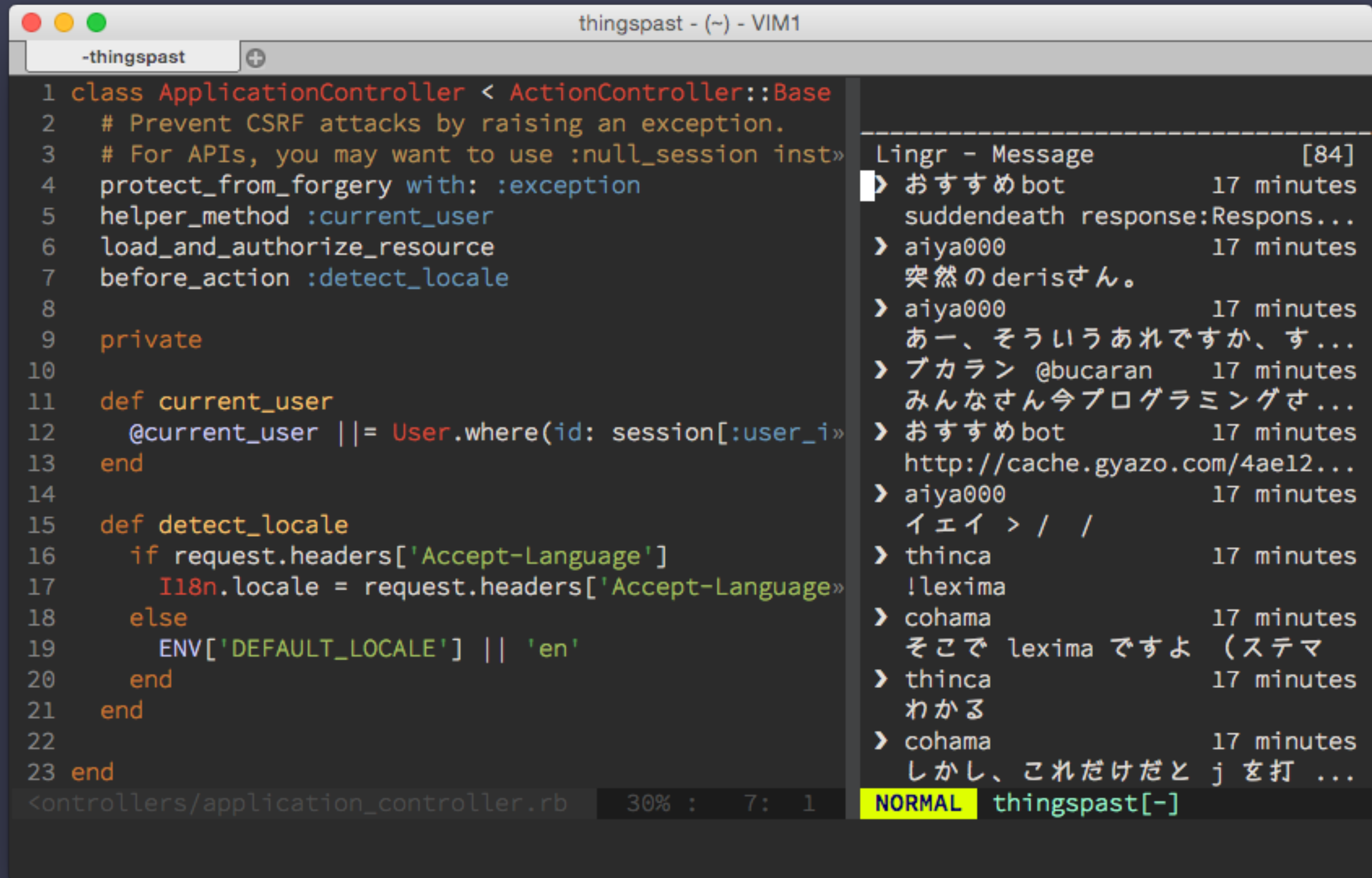
ThingsPast.vim



```
1 class ApplicationController < ActionController::Base
2   # Prevent CSRF attacks by raising an exception.
3   # For APIs, you may want to use :null_session instead.
4   protect_from_forgery with: :exception
5   helper_method :current_user
6   load_and_authorize_resource
7   before_action :detect_locale
8
9   private
10
11   def current_user
12     @current_user ||= User.where(id: session[:user_id]).first if session
13   end
14
15   def detect_locale
16     if request.headers['Accept-Language']
17       I18n.locale = request.headers['Accept-Language'].scan(/^[a-z]{2}/).first
18     else
19       ENV['DEFAULT_LOCALE'] || 'en'
20     end
21   end
22
23 end
```

NORMAL master <plication_controller.rb project.ruby.rails utf-8[unix] 30% : 7: 1

ThingsPast.vim



The screenshot shows a Vim editor window titled "thingspast - (~) - VIM1". The left pane displays a Ruby file named "application_controller.rb" with the following code:

```
1 class ApplicationController < ActionController::Base
2   # Prevent CSRF attacks by raising an exception.
3   # For APIs, you may want to use :null_session inst»
4   protect_from_forgery with: :exception
5   helper_method :current_user
6   load_and_authorize_resource
7   before_action :detect_locale
8
9   private
10
11   def current_user
12     @current_user ||= User.where(id: session[:user_i»
13   end
14
15   def detect_locale
16     if request.headers['Accept-Language']
17       I18n.locale = request.headers['Accept-Language»
18     else
19       ENV['DEFAULT_LOCALE'] || 'en'
20     end
21   end
22
23 end
```

The right pane shows a message log titled "Lingr - Message [84]". It contains a list of messages with timestamps and user names:

- おすすめ bot 17 minutes
- suddendeath response: Respons...
- aiya000 17 minutes
- 突然の deris さん。
- aiya000 17 minutes
- あー、そういうあれですか、す...
- ブカラン @bucaran 17 minutes
- みんなさん今プログラミングせ...
- おすすめ bot 17 minutes
- http://cache.gyazo.com/4ae12...
- aiya000 17 minutes
- イエイ > / /
- thinca 17 minutes
- !lexima
- cohama 17 minutes
- そこで lexima ですよ (ステマ
- thinca 17 minutes
- わかる
- cohama 17 minutes
- しかし、これだけだと j を打 ...

The status bar at the bottom shows the file path "<ontrollers/application_controller.rb", the cursor position "30% : 7: 1", and the mode "NORMAL thingspast[-]".

やっていること

各プラグインのhookを利用して

ThingsPastに通知を流し込んでいる

やっていること

この時、通知オブジェクトにcallback関数を
持たせることができるので、通知選択時に
任意の処理を実行可能

たとえば

「J6uilからの“新規発言受信”の通知」を

ThingsPast上で選択(<CR>)すると

J6uil.vimを起動

これによって
通知の一元管理が実現

Vimで動くアプリケーションを開発する際
通知関連の実装をThingsPastに丸投げできる
(プラグイン側はhookを用意しておくだけ)

ユーザーは統一したインターフェイスで
通知を開覧し、通知元へ移動することができる

便利（かなり）

ここで問題が発生

我々はVimを使っている時
両目をフル稼働させている



```
1 class ApplicationController < ActionController::Base
2   # Prevent CSRF attacks by raising an exception.
3   # For APIs, you may want to use :null_session instead.
4   protect_from_forgery with: :exception
5   helper_method :current_user
6   load_and_authorize_resource
7   before_action :detect_locale
8
9   private
10
11   def current_user
12     @current_user ||= User.where(id: session[:user_id]).first if session
13   end
14
15   def detect_locale
16     if request.headers['Accept-Language']
17       I18n.locale = request.headers['Accept-Language'].scan(/^[a-z]{2}/).first
18     else
19       ENV['DEFAULT_LOCALE'] || 'en'
20     end
21   end
22
23 end
```

thingspast - (~) - VIM1

-thingspast

+

1 class ApplicationController < ActionController::Base

2 # Prevent CSRF attacks by raising an exception.

3 # For APIs, you may want to use :null_session inst»

4 protect_from_forgery with: :exception

5 helper_method :current_user

6 load_and_authorize_resource

7 before_action :detect_locale

8

9 private

10

11 def current_user

12 @current_user ||= User.where(id: session[:user_i»

13 end

14

15 def detect_locale

16 if request.headers['Accept-Language']

17 I18n.locale = request.headers['Accept-Language»

18 else

19 ENV['DEFAULT_LOCALE'] || 'en'

20 end

21 end

22

23 end

<ontrollers/application_controller.rb

30% : 7: 1

Lingr - Message [84]

➤ おすすめ bot 17 minutes
suddendeath response:Respons...

➤ aiya000 17 minutes
突然のderisさん。

➤ aiya000 17 minutes
あー、そういうあれですか、す...

➤ ブカラン @bucaran 17 minutes
みんなさん今プログラミングせ...

➤ おすすめ bot 17 minutes
http://cache.gyazo.com/4ae12...

➤ aiya000 17 minutes
イエイ > / /

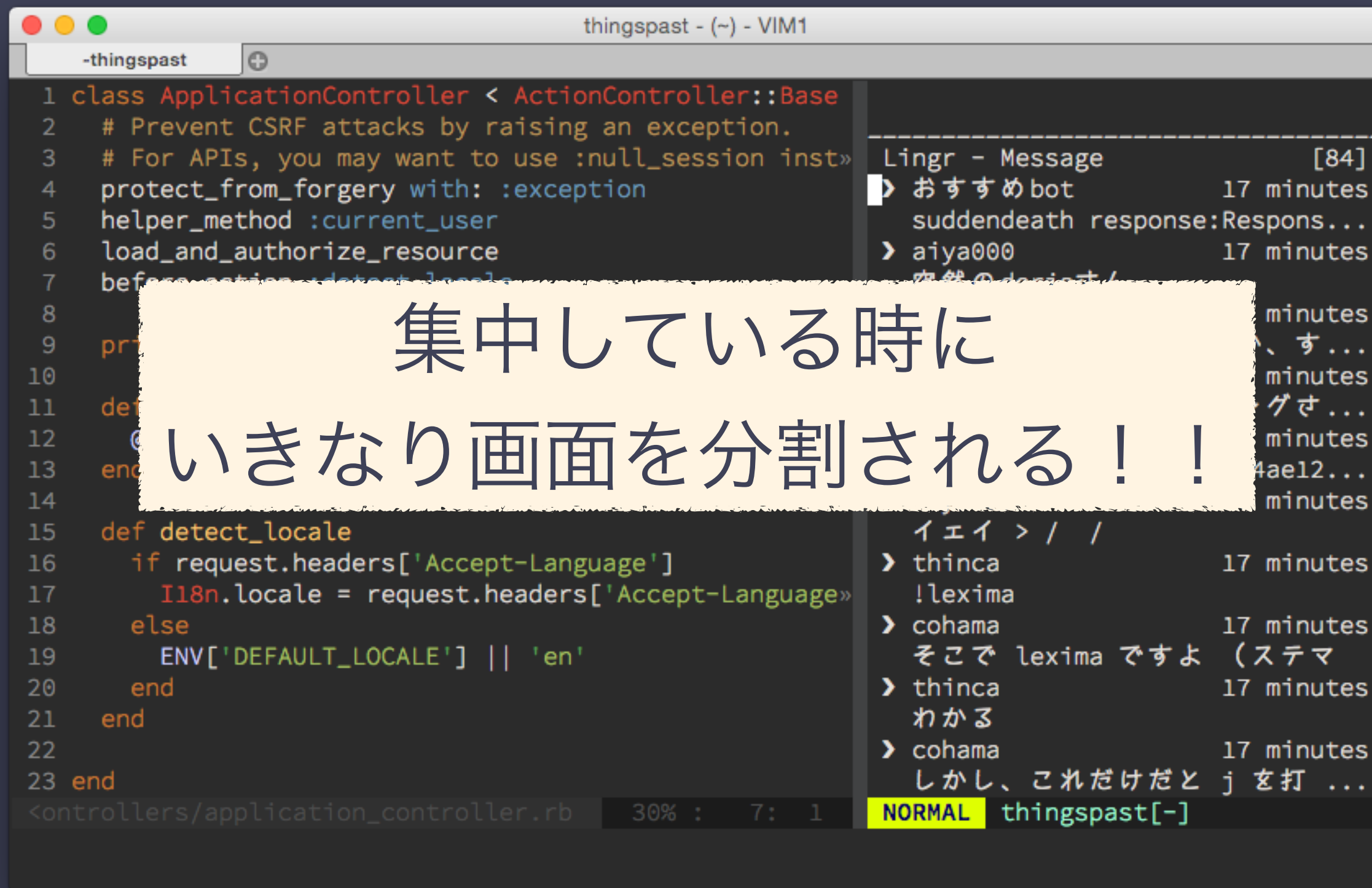
➤ thinca 17 minutes
!lexima

➤ cohama 17 minutes
そこで lexima ですよ (ステマ

➤ thinca 17 minutes
わかる

➤ cohama 17 minutes
しかし、これだけだと j を打 ...

NORMAL thingspast[-]



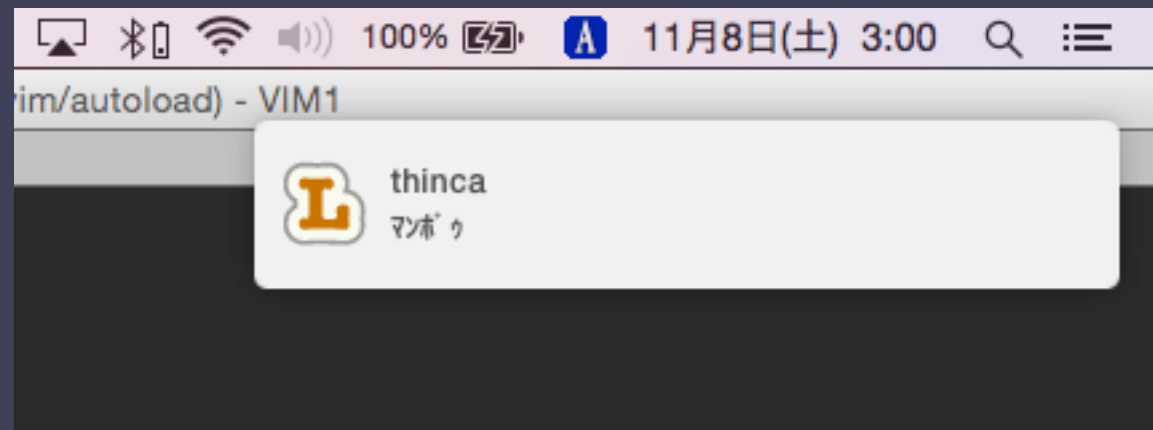
集中している時に
いきなり画面を分割される！！

とてもよくない（とても）

もっと「静かに」通知させたい

もっと「静かに」通知させる方法...

通知センターの「バルーン通知」を、
バッファにオーバーレイさせて実現する？



そうじゃない

そもそも

集中して視覚を活用してコード書いてる時に
視覚のリソースを少しでも奪うのがダメ

そんなのは「静か」じゃない

そんなのはやない



そこで音です

脳に情報を入力できる器官は目だけじゃない

視覚のリソースが足りないなら
聴覚のリソースを使えばいい

聴覚へ通知

=音で通知

= 音声合成で通知

=Shaberu.vim

Shaberu.vim

Vimプラグイン処女作

音声合成エンジンのラッパーライブラリ

1. Shaberu.vimで音声通知

2. 詳しく見たくなったらThingsPastペインを表示

これで視覚をいきなり遮られることがない！

(デモ)

<https://vimeo.com/111277527>

かなり便利（かなり）

まとめます

今日、通知の話を用いて
伝えなかったこと

聴覚というリソースをもっと活用しよう

おまけ

その他の **Shaberu.vim**活用例

autocmd

様々なautocmdと連動させることで
Vimがすごいべんりになる

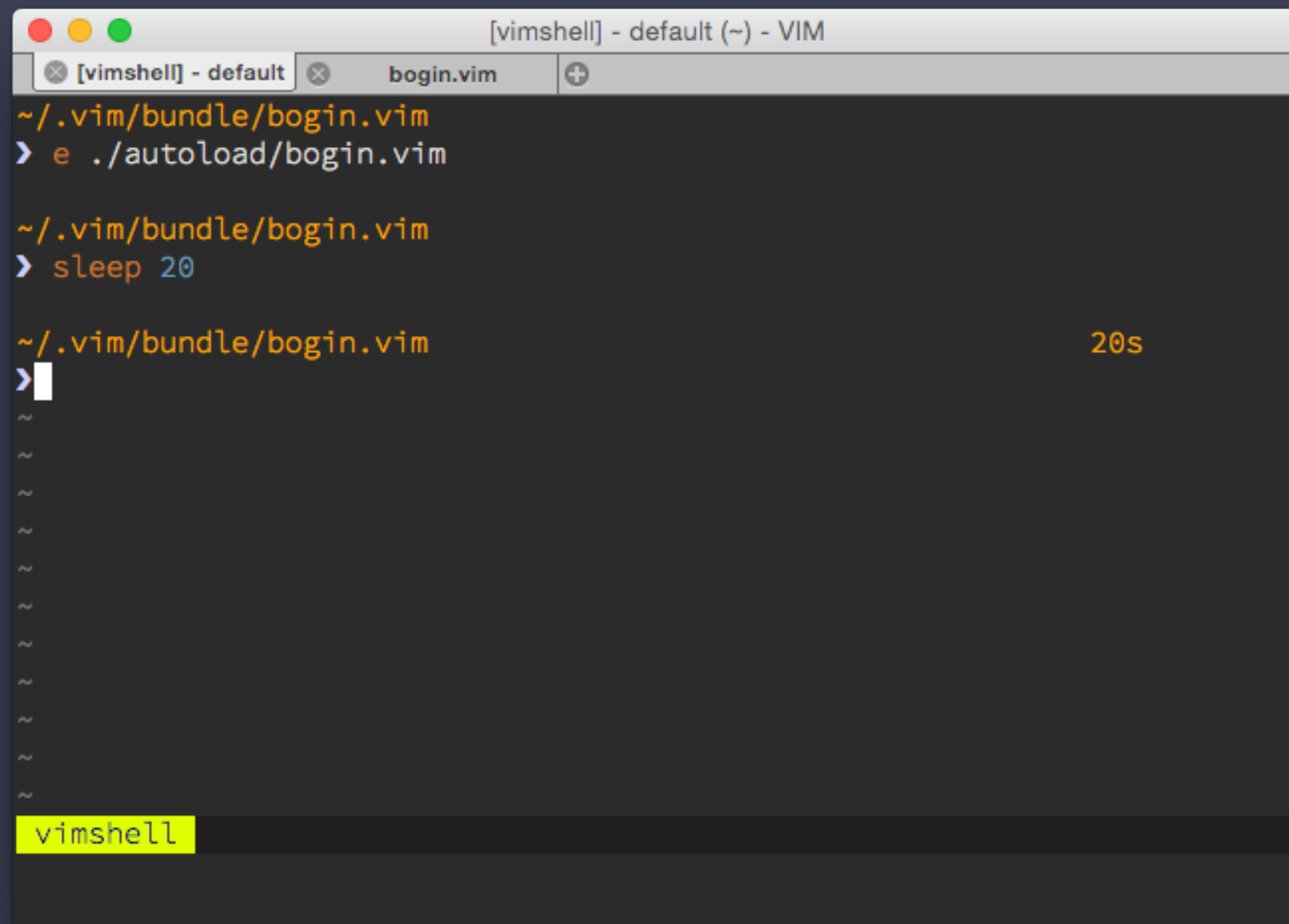
起動/終了時に進捗を促してくれる

```
au MyAutoCmd VimEnter * ShaberuSay '進捗どうですか'
```

```
au MyAutoCmd VimLeave * ShaberuSay '進捗どうですか'
```

vimshell-wakeup.vim

VimShellで「時間がかかるコマンド」の実行が終了したとき、音声で通知してくれるプラグイン



```
[vimshell] - default (~) - VIM
[vimshell] - default  bogin.vim
~/vim/bundle/bogin.vim
> e ./autoload/bogin.vim

~/vim/bundle/bogin.vim
> sleep 20

~/vim/bundle/bogin.vim                                     20s
>
~
~
~
~
~
~
~
~
~
~
~
~
vimshell
```

VimShellで「時間がかかるコマンド」の実行が終了したとき、音声で通知してくれるプラグイン

- ・ 設定した秒数以上の時間がかかったコマンドが終了した時に発動
(herokuへのデプロイとか)
- ・ コマンド終了時、VimShellバッファを開いていた場合は自明なので発動しない
- ・ ただし、VimShellバッファを開いていても、Vim以外のアプリケーションがアクティブになっていた場合は発動

VimShellで「時間がかかるコマンド」の実行が終了
したとき、音声で通知してくれるプラグイン

VimShellの仕様上、通知されない場合がある（改善したい）

VimShell hooks

VimShellでは様々なタイミングに
hook関数を呼ぶことができる

エラー発生時に音声で教えてくれる

```
au MyAutoCmd FileType vimshell
  \ call vimshell#hook#add(
    \ 'emptycmd',
    \ 'my_vimshell_emptycmd',
    \ reti#lambda(":call shaberu#say('コマンドを入力してください') | return a:1"))
```

```
au MyAutoCmd FileType vimshell
  \ call vimshell#hook#add(
    \ 'notfound',
    \ 'my_vimshell_notfound',
    \ reti#lambda(":call shaberu#say('コマンドが見つかりません') | return a:1"))
```


移動 (cd) したときに 「よっこいしょ」と喋る

```
au MyAutoCmd FileType vimshell
\ call vimshell#hook#add(
\ 'chpwd' ,
\ 'my_vimshell_chpwd' ,
\ reti#lambda(":ShaberuSay 'よっこいしょ'"))
```

VimShell aliases

VimShellではコマンドエイリアスを
定義することができる

time?コマンドで 時間を教えてくれる

```
call vimshell#set_alias(  
  \ 'time?',  
  \ ':call shaberu#say(strftime("はいっ。今は%H時%M分です"))')
```

lsを打ち間違えた (sl) ときに 「きしゃぽっぽ」と喋る

```
call vimshell#set_alias(  
  \ 'sl',  
  \ ':call shaberu#say("きしゃぽっぽ。きしゃぽっぽ。ぽぽ")')
```


とてもべんり（とても）

まとめ

Vimがしゃべると便利

Vimがしゃべるとかわいい (重要)

Shaberu.vimは
任意の音声合成エンジンを使用可能

OpenJTalkを使うとかなりかわいい（かなり）

もう一度言います

聴覚というリソースをもっと活用しよう

ありがとうございました

イラスト：

いらすとや ([irasutoya.com](https://www.irasutoya.com))

音声：

OpenJTalk (mei voice)