



OBJECT ORIENTED WEB PROGRAMMING USING RUBY

Day 4: 10/May/2012

Locale and Internationalization,
Test Driven Development

Our Goal for this semester

This lecture is for graduate students, and the lecture is assumed as “advanced” learning of WEB and Database.

Let’s set up our goal of the Project; that is

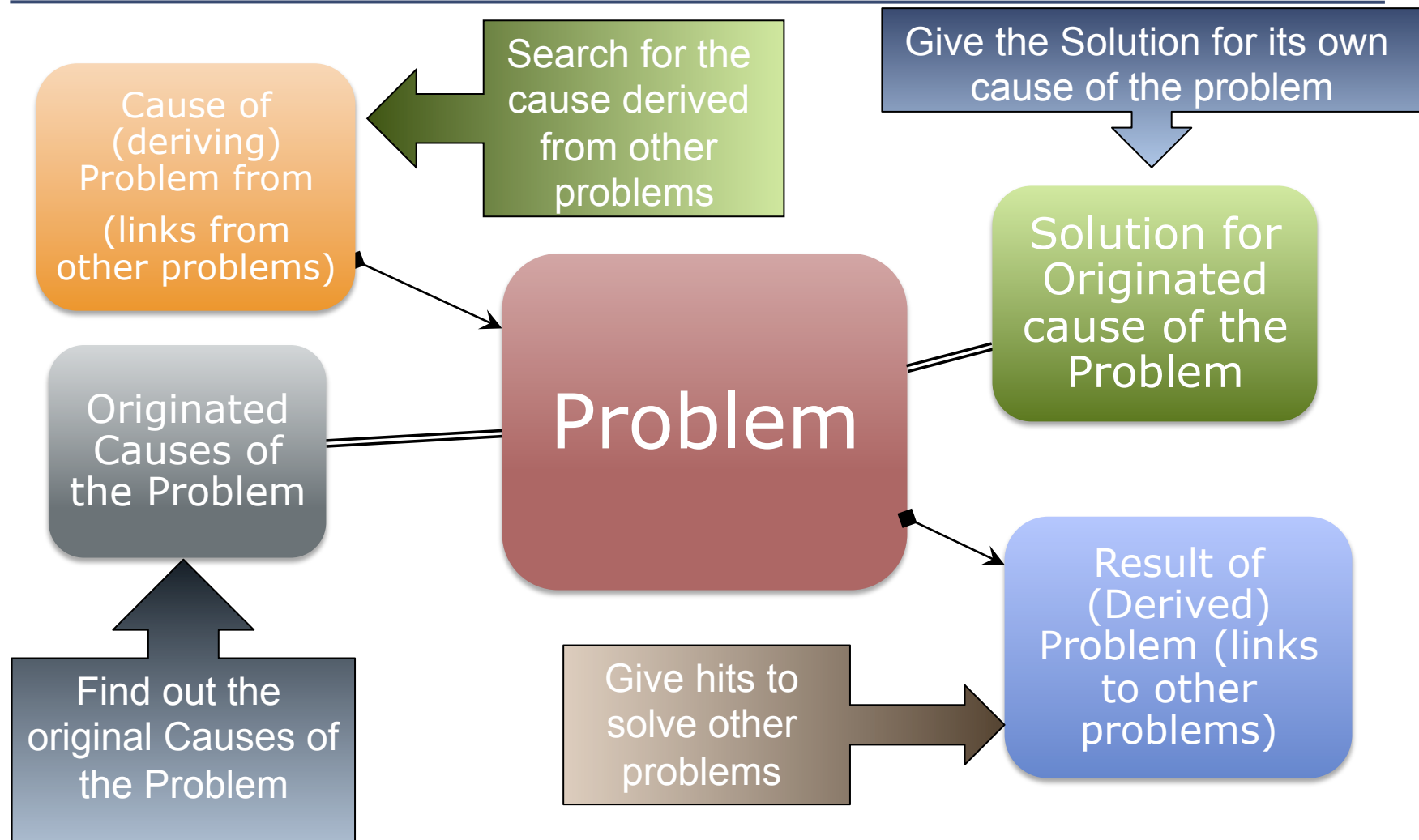
“Problem Solving Engine”

When we face any “problem,” we often try to find “solution,” asking our friends or search WEB pages in the internet environment.

Give the “Network Structure” to the “Problems” and “Solutions”, then solve the source of the problem from its causes.

We design the WEB based Database System for the “Solutions of problems,” and by developing the system, we learn [how to write programs, and design the system.](#)

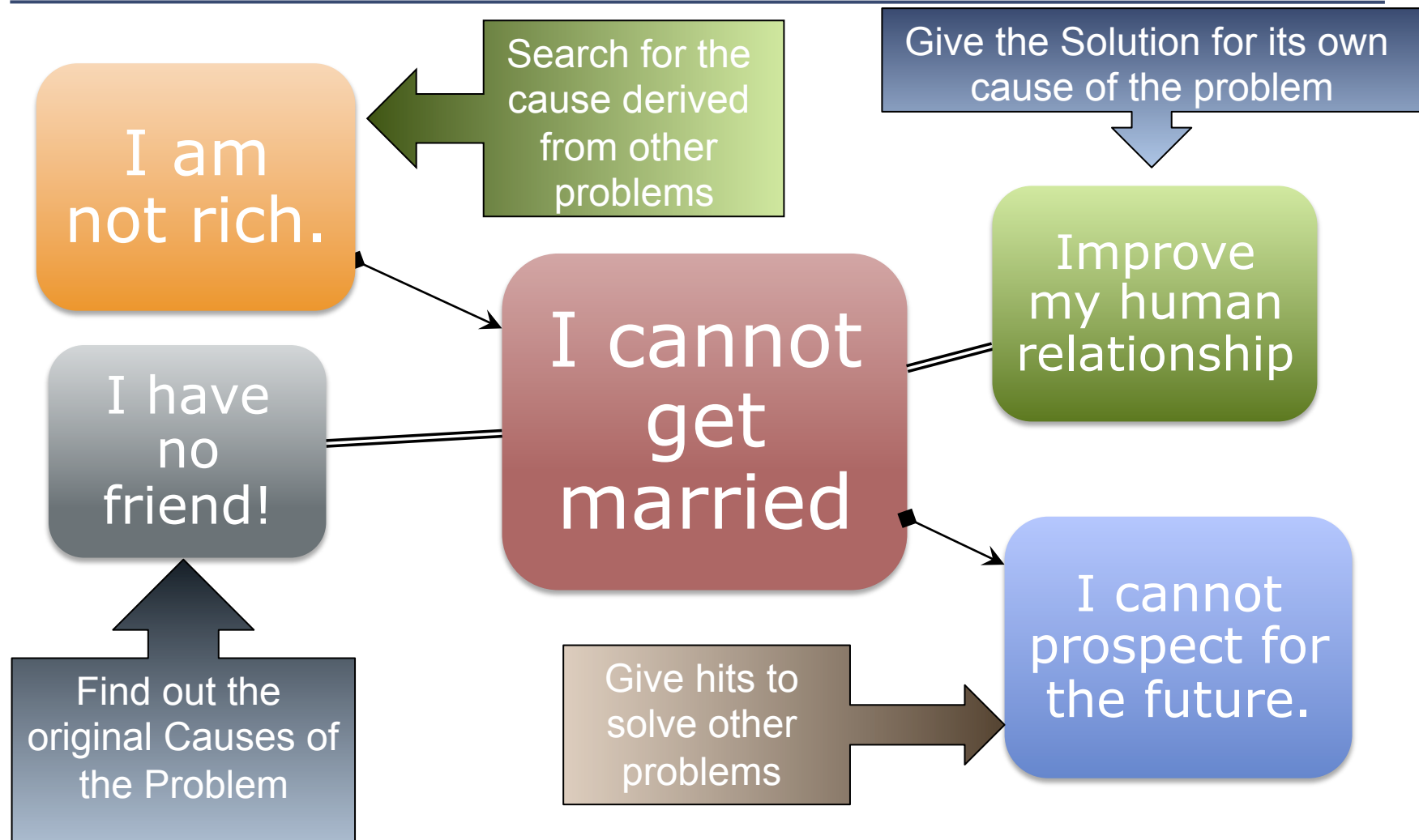
Design Image of the system



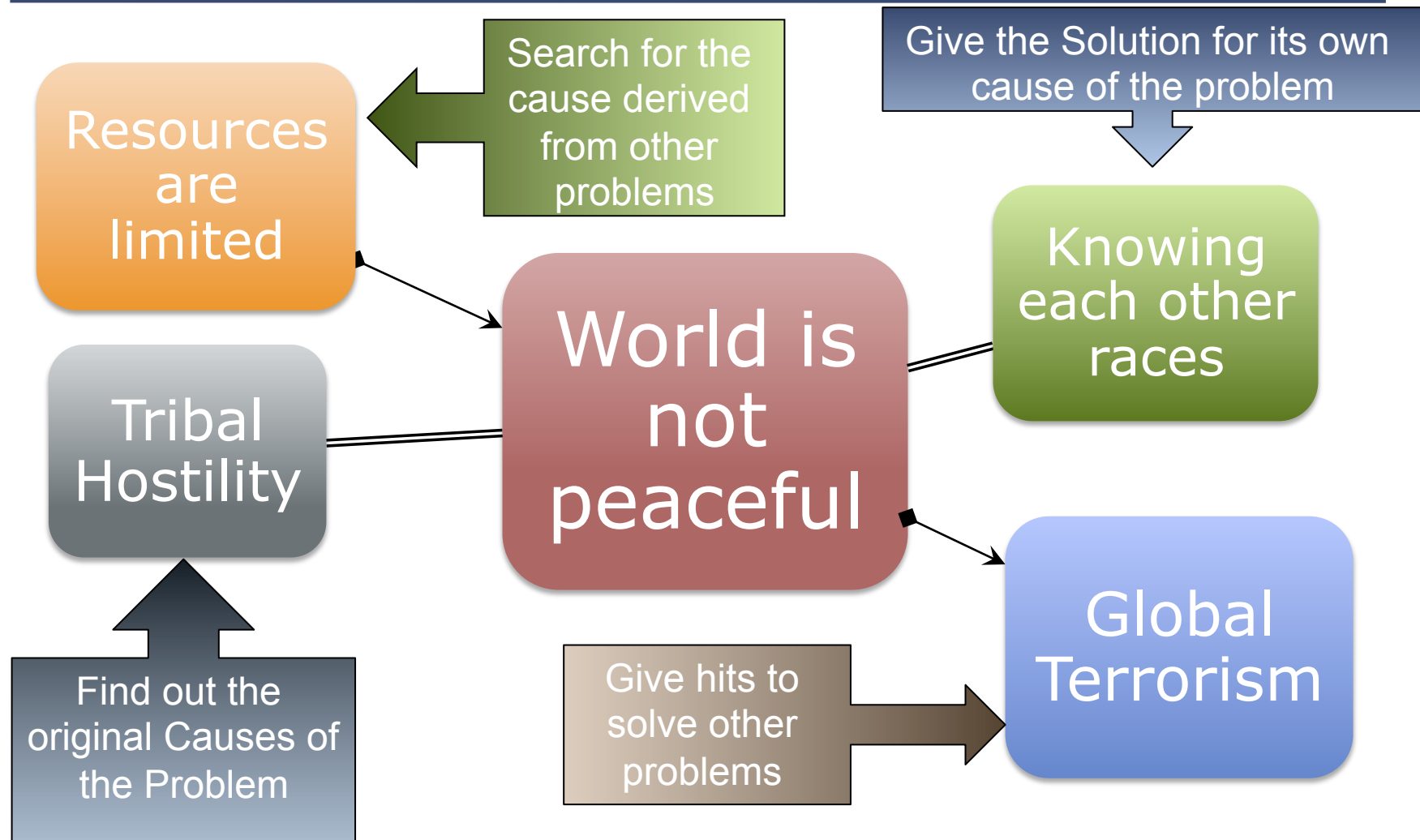
How the system help solving problems?

- Give some links to a "Problem."
- The links are;
 - "Cause" of the Problem.
 - Originated "cause"
 - Linked from other "problem"
 - "Solution" for the originated "cause"
 - "results" of the problem
 - Other problems could derive from this problem.
 - Some other acceptable results.

Example 1 of Problem Solution



Example 2 of Problem Solution



Aim for the next “Facebook”

By Integrating people's idea,
we can solve any problems we face,
to improve the future!

Build the WEB system open to the public to
join, and contribute for the world!

The Project name

- Let's give the project name for this Problem Solving Engine.

- Proposal 1 : PSE – very straight naming;
 - Problem Solving Engine

 - Proposal 2 : Lucas –
 - Let Us Conclude by Assuming Solutions
 - Proposal 3 : Spielberg –
 - Solving Problems with Integrated Engine for Listing Beginning Essences of Resulting Gripes ... or whatever ...

Today's Topics

Let us build the PSE system in the last 10 lessons of this lecture.

Today, we introduce multi lingual support.

This year again, we spent 3 weeks to set the environment ready. The content of syllabus was been shifted for one week.

To begin with “Workspace”

Create “Workspace,”

which contains all project directories in,

Make the path easy to access in the shell,

which means the name does not contain spaces

- C:\Users\admin\Aptana3Work
 - And such would be better
 - cd ~
 - Mkdir Aptana3Work

```
kobayashi-ikuo-no-MacBook:~ kobayashi$ cd ~
kobayashi-ikuo-no-MacBook:~ kobayashi$ mkdir Aptana3Work
kobayashi-ikuo-no-MacBook:~ kobayashi$ pwd
/Users/kobayashi
kobayashi-ikuo-no-MacBook:~ kobayashi$ cd Aptana3Work/
kobayashi-ikuo-no-MacBook:Aptana3Work kobayashi$ pwd
/Users/kobayashi/Aptana3Work
kobayashi-ikuo-no-MacBook:Aptana3Work kobayashi$ █
```

Generate Project

Let's give the project name:

(for me) spielberg

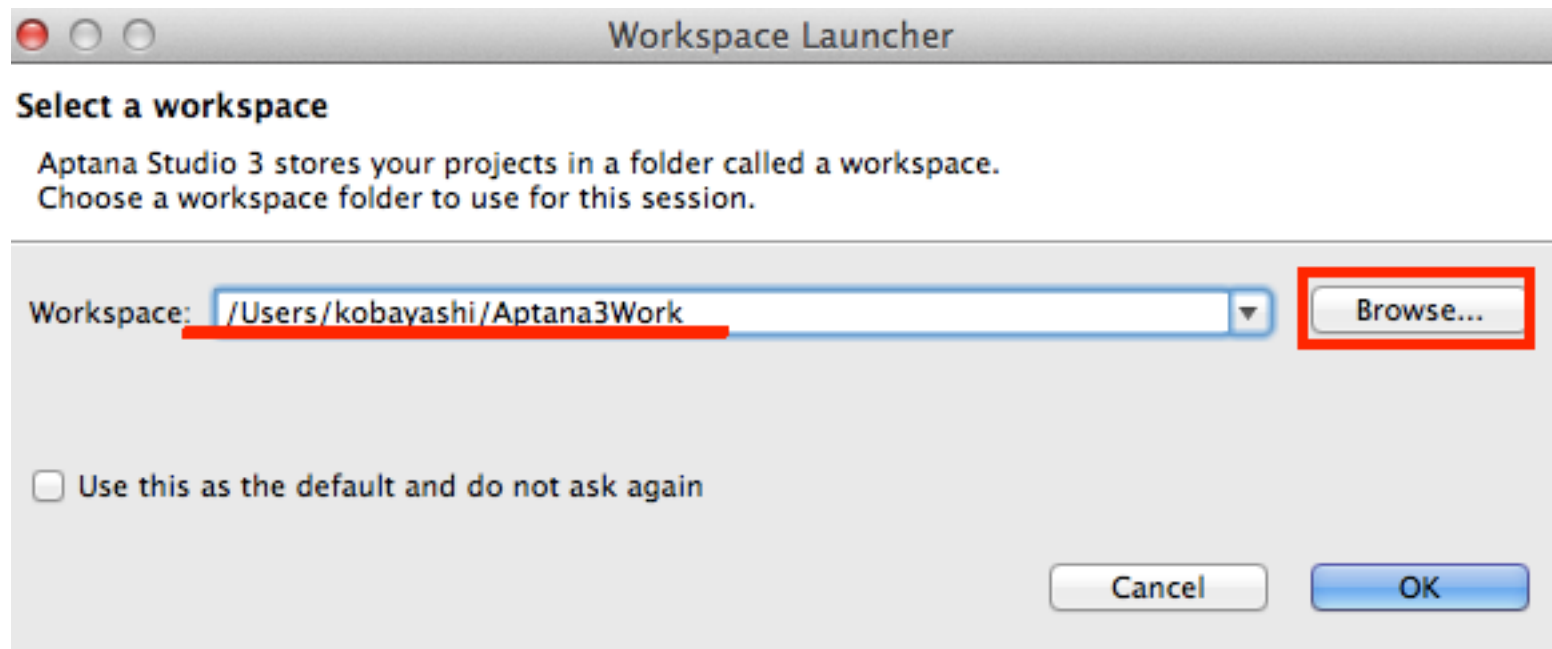
(Give the project name whatever you like.)

`rails new spielberg`

```
kobayashi-ikuo-no-MacBook:Aptana3Work kobayashi$ rails new spielberg
create
create README.rdoc
create Rakefile
create config.ru
create .gitignore
create Gemfile
create app
create app/assets/images/rails.png
create app/assets/javascripts/application.js
create app/assets/stylesheets/application.css
create app/controllers/application_controller.rb
create app/helpers/application_helper.rb
create app/mailers
create app/models
create app/views/layouts/application.html.erb
create app/mailers/.gitkeep
create app/models/.gitkeep
create config
create config/routes.rb
```

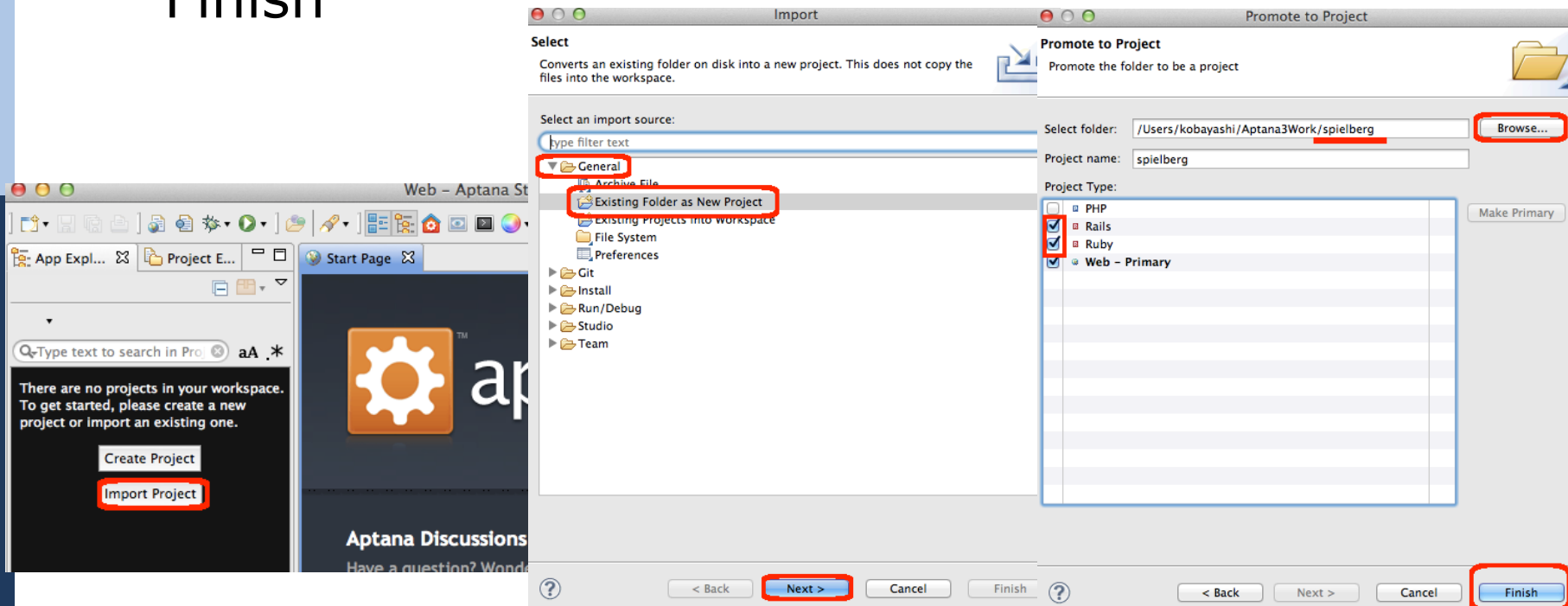
Set Workspace to Aptana

Start Aptana with created Workspace.



Import the project into Aptana

- 1) Click Import Project
- 2) Choose the filter: "Existing Folder as New Project"
- 3) Browse to get name, check Rails and Ruby, and Finish



Scaffolding

- Provide the “Base” table of database
 - “Problem” is the main database table
- Create the table with the following fields,
 - title:string
 - content:text
- Later on, we add “cause” tables and “Solution” tables as links to this table

rails generate scaffold problem title:string content:text

```
kobayashi-ikuo-no-MacBook:spielberg kobayashi$ rails generate scaffold problem
title:string content:text
invoke active_record
create db/migrate/20120504065255_create_problems.rb
create app/models/problem.rb
invoke test_unit
create test/unit/problem_test.rb
create test/fixtures/problems.yml
route resources :problems
invoke scaffold_controller
create app/controllers/problems_controller.rb
invoke erb
create app/views/problems
create app/views/problems/index.html.erb
.
```

Cancelation of Scaffolding

If you mistype the table name or the field name of the table of database, then cancel the scaffolding.

Just type

```
rails destroy scaffold problem
```

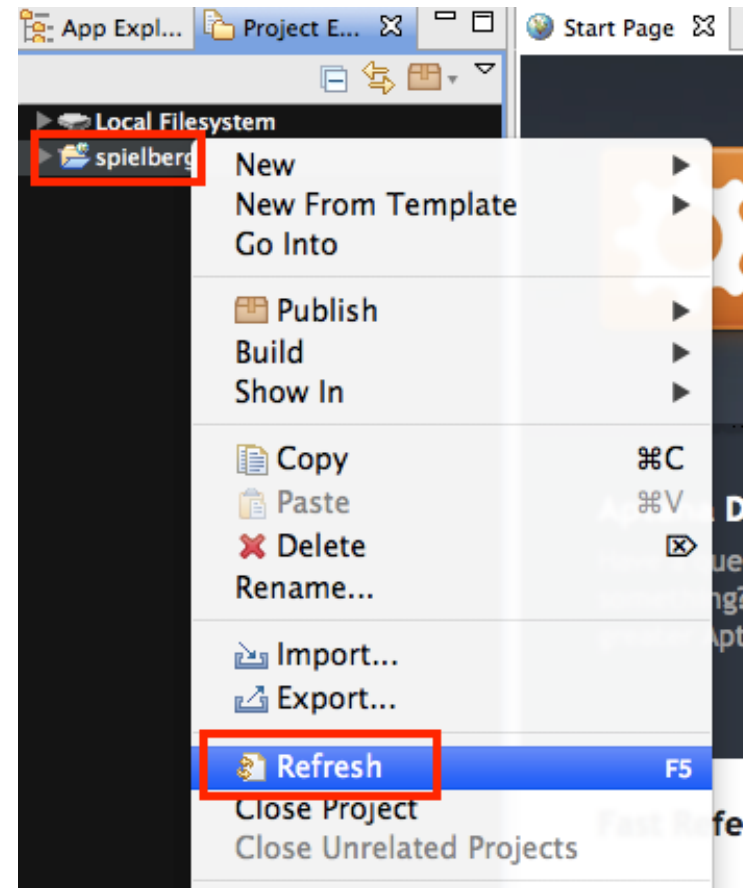
to cancel the generation of

```
rails generate scaffold problem title:string content:text
```

If you have to cancel the migration also, cancel the migration prior to this cancelation.

Refresh

Right click the Project name, then choose “refresh” to reflect the changes into Aptana.



Migration

Migration is the implementation of the table schema of database.

Type

`rake db:migrate`

If you cancel the migration, type

`rake db:rollback`

```
kobayashi-ikuo-no-MacBook:spielberg kobayashi$ rake db:migrate
== CreateProblems: migrating =====
-- create_table(:problems)
   -> 0.0015s
== CreateProblems: migrated (0.0016s) =====

kobayashi-ikuo-no-MacBook:spielberg kobayashi$ rails server
```

Edit Gemfile

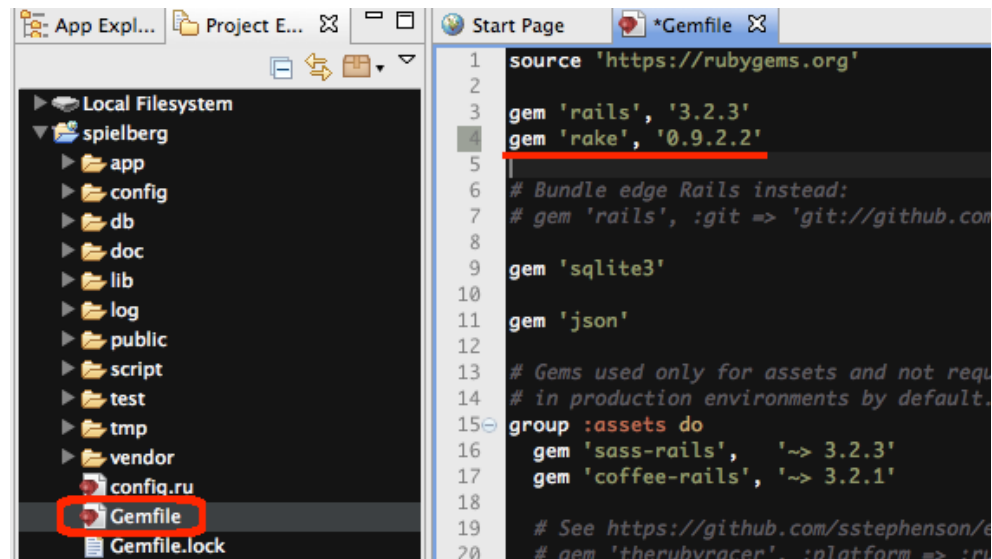
- This could be the patch for the previous bugs which had been already fixed, but...

In Gemfile, just below

`gem 'rails', '3.2.3'`

Add

`gem 'rake', '0.9.2.2'`



```
1 source 'https://rubygems.org'
2
3 gem 'rails', '3.2.3'
4 gem 'rake', '0.9.2.2'
5
6 # Bundle edge Rails instead:
7 # gem 'rails', :git => 'git://github.com:rails/rails.git'
8
9 gem 'sqlite3'
10
11 gem 'json'
12
13 # Gems used only for assets and not required
14 # in production environments by default.
15 group :assets do
16   gem 'sass-rails', '~> 3.2.3'
17   gem 'coffee-rails', '~> 3.2.1'
18 end
19
20 # See https://github.com/sstephenson/rubygems-ssl for details
21 # gem 'therubyracer', :platform => :ruby
```

Gem update

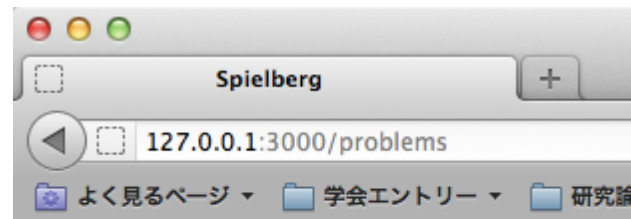
To make the change effective, type
`bundle update`

```
kobayashi-ikuo-no-MacBook:spielberg kobayashi$ bundle update
Fetching source index for https://rubygems.org/
Enter your password to install the bundled RubyGems to your system:
Using rake (0.9.2.2)
Using i18n (0.6.0)
Using multi_json (1.3.4)
Using activesupport (3.2.3)
Using builder (3.0.0)
Using activemodel (3.2.3)
Using erubis (2.7.0)
Using journey (1.0.3)
Using rack (1.4.1)
Using rack-cache (1.2)
Using rack-test (0.6.1)
Using hike (1.2.1)
Using tilt (1.3.3)
Using sprockets (2.1.3)
Using actionpack (3.2.3)
```

Invoke project on the Server

- Type rails server
- To start server. Now we should have the screen for “problems,” type

<http://127.0.0.1:3000/problems/>



Listing problems

Title Content

[New Problem](#)



Internationalization

- Official Site
 - <http://edgeguides.rubyonrails.org/i18n.html>
- Specify “locales”, register the translations for each “keywords,” to get the message in various languages.
- We have to provide own dictionary.
- Let’s modify the Entry Screen.

Code for your language

<http://www.w3.org/International/>

http://en.wikipedia.org/wiki/ISO_3166-1_alpha-2

http://en.wikipedia.org/wiki/List_of_ISO_639-1_codes

For Japanese people, the language code is 'ja', and the country code is 'JP.' Use 'ja_JP'

For Tamil language, the code is 'ta.' The country code of 'LT' is for Sri Lanka.

Preparation of Dictionary File

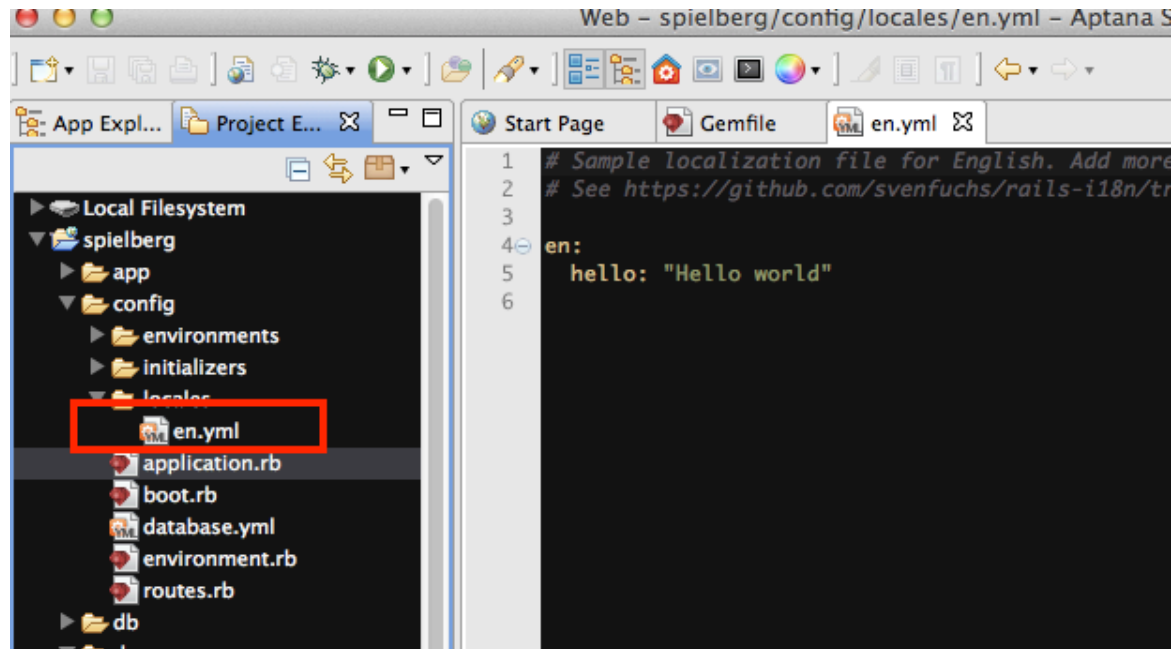
- Find the locale files under config/locales
 - (locale).yml

- What locales are supported?
 - For Java:
 - http://lab.moyo.biz/recipes/java/locale/format_1.5.0_05.xsp
 - For Windows
 - <http://msdn.microsoft.com/ja-jp/library/cc392381.aspx>

- Automatically generated file include en.yml. This is for “English” locale.

Locale Description Sample

- Open
 - spielberg\config\locales\en.yml
- In “en” locale,
The Symbol “hello” appears, the message will be
“Hello world”



The screenshot shows an IDE window with the title "Web - spielberg/config/locales/en.yml - Aptana S". The left sidebar displays a file explorer for the "Local Filesystem" with the following structure:

- Local Filesystem
 - spielberg
 - app
 - config
 - environments
 - initializers
 - locales
 - en.yml (highlighted with a red box)
 - application.rb
 - boot.rb
 - database.yml
 - environment.rb
 - routes.rb
- db
- des

The main editor area shows the content of the en.yml file:

```
1 # Sample localization file for English. Add more
2 # See https://github.com/svenfuchs/rails-i18n/tr
3
4 en:
5   hello: "Hello world"
6
```


Grammar of Yml(YAML)

- For detailed information see the page below:
<http://en.wikipedia.org/wiki/YAML>
- Array(Lists)
 - item1
 - item2
- Hash (Associated Arrays)
 - Key: Value
 - After colon(:) **you need to put space(s)**.
- # is for comment
- Data type expressions are the same with C and Java language.

Changes from Rails 3.0 to 3.2

XML based → JSON based

JSON: JavaScript Object Notation

- We had to learn the grammar of XML and YAML, but now we have to learn the grammar of JSON and YAML.

YAML: YAML Ain't Markup Language

Let rails know the languages

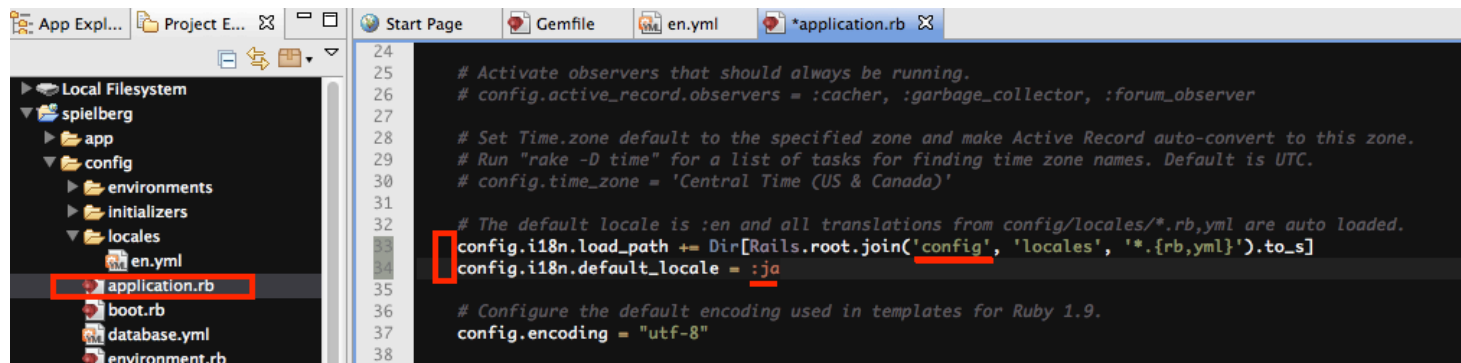
Edit the file

spielberg\config\application.rb

in line number 33 and 34

```
config.i18n.load_path += Dir[Rails.root.join('config', 'locales', '*.rb,*.yml').to_s]
config.i18n.default_locale = :ja
```

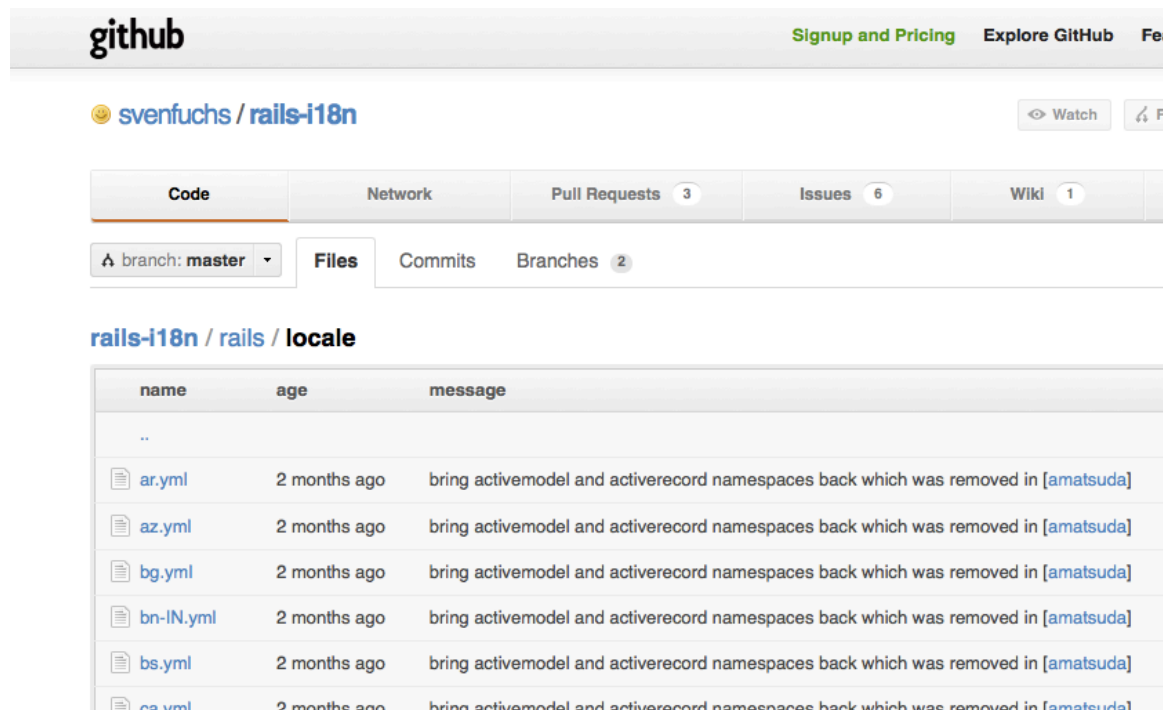
- Uncomment the line (remove “#” at the top of the line,) then replace “my” to “config”, and change default locale into ja (or en.)



```
24
25 # Activate observers that should always be running.
26 # config.active_record.observers = :cacher, :garbage_collector, :forum_observer
27
28 # Set Time.zone default to the specified zone and make Active Record auto-convert to this zone.
29 # Run "rake -D time" for a list of tasks for finding time zone names. Default is UTC.
30 # config.time_zone = 'Central Time (US & Canada)'
31
32 # The default locale is :en and all translations from config/locales/*.rb,yml are auto loaded.
33 config.i18n.load_path += Dir[Rails.root.join('config', 'locales', '*.rb,*.yml').to_s]
34 config.i18n.default_locale = :ja
35
36 # Configure the default encoding used in templates for Ruby 1.9.
37 config.encoding = "utf-8"
38
```

Prepare the language dictionary

- Let us get the ja.yml file from the follogin site:
- <https://github.com/svenfuchs/rails-i18n/tree/master/rails/locale>

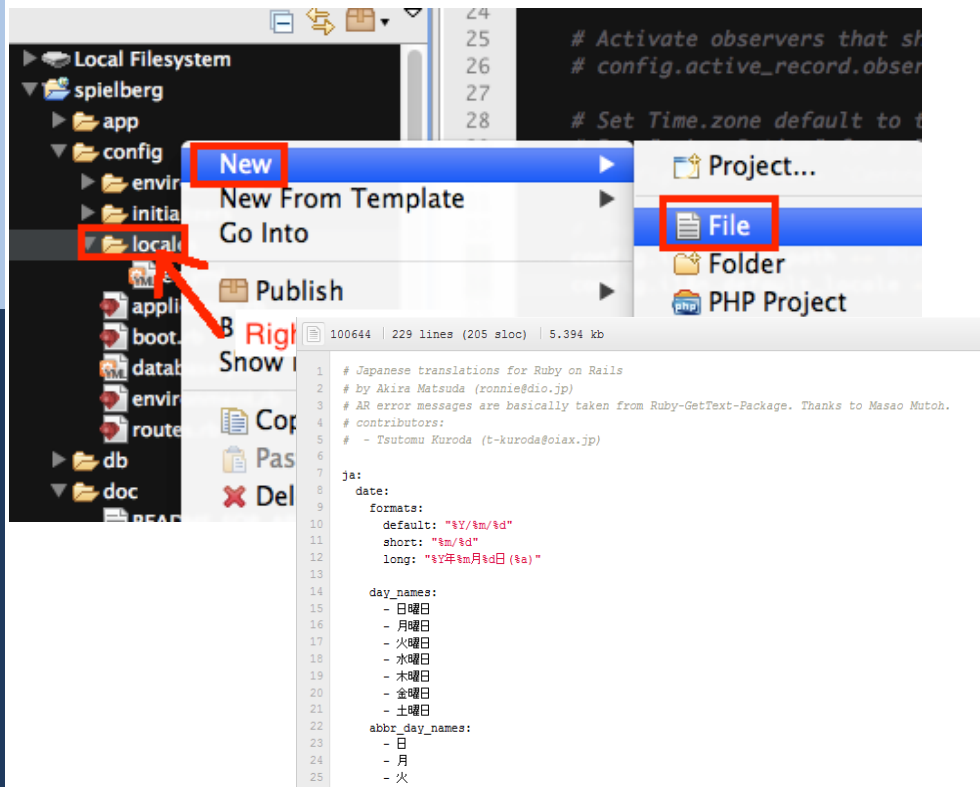


The screenshot shows the GitHub interface for the repository `svenfuchs/rails-i18n`. The repository is on the `master` branch. The navigation tabs include Code, Network, Pull Requests (3), Issues (6), and Wiki (1). The current view is the file browser for the `rails-i18n / rails / locale` directory. The file list shows several locale files, each with a document icon, a name, an age of 2 months ago, and a message indicating they were removed in a previous commit.

name	age	message
..		
ar.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]
az.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]
bg.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]
bn-IN.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]
bs.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]
ca.yml	2 months ago	bring activemodel and activerecord namespaces back which was removed in [amatsuda]

ja.yml

- ❑ Create ja.yml file in config/locales.
- ❑ Then, copy and paste the file contents.
- ❑ Also, you can update en.yml, or support other languages.



```
errors:
  format: "%{attribute}%{message}"
```

```
messages: :errors_messages
inclusion: "Iは一覧にありません。"
exclusion: "Iは予約されています。"
invalid: "Iは不正な値です。"
confirmation: "Iが一致しません。"
accepted: "Iを受諾してください。"
empty: "Iを入力してください。"
blank: "Iを入力してください。"
too_long: "Iは#{count}文字以内で入力してください。"
too_short: "Iは#{count}文字以上で入力してください。"
wrong_length: "Iは#{count}文字で入力してください。"
not_a_number: "Iは数値で入力してください。"
not_an_integer: "Iは整数で入力してください。"
greater_than: "Iは#{count}より大きい値にしてください。"
greater_than_or_equal_to: "Iは#{count}以上の値にしてください。"
equal_to: "Iは#{count}にしてください。"
less_than: "Iは#{count}より小さい値にしてください。"
less_than_or_equal_to: "Iは#{count}以下の値にしてください。"
odd: "Iは奇数にしてください。"
even: "Iは偶数にしてください。"
```

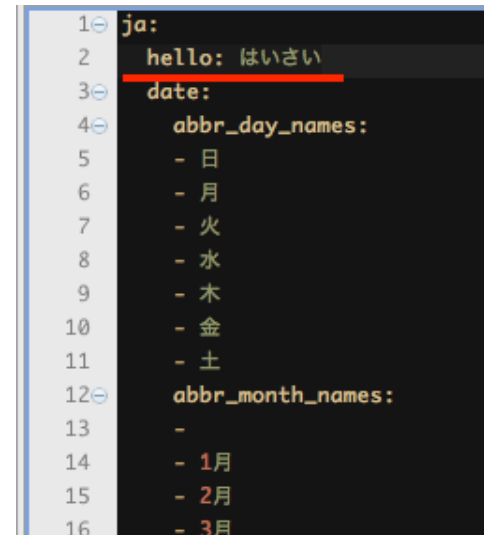
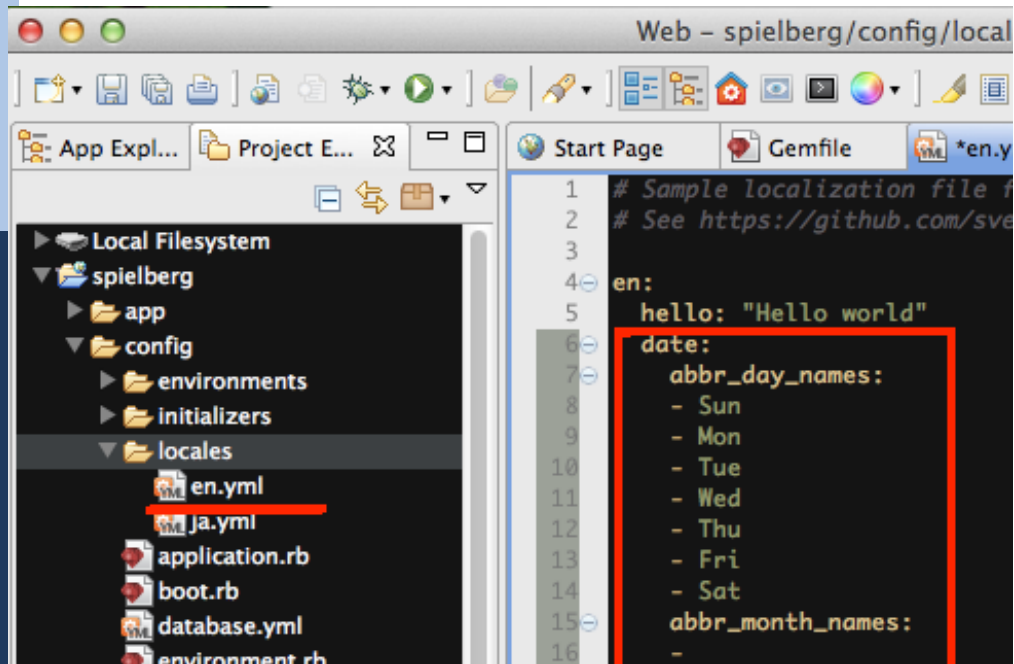
Edit ja.yml

In the file

spielberg\config\locales\

Add hello string.

IMPORTANT: items in the same depth have same indentation.



Edit the target file to display

Open

spielberg/app/views/problems/new.html.erb

file, and see the top line. `<h1>` is a tag to show "Heading Level 1", and the content of the heading is "New problem," the string literal.

We replace this literal into the "symbol" of `'new_problem'` and to let rails pick up the local languages for the symbol `'new_problem.'` To do so, we replace "New problem" into

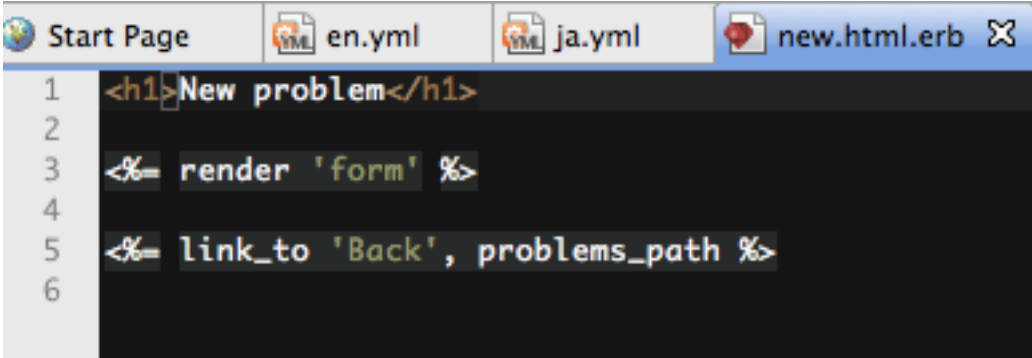
`t :new_problem`

or

`t 'new_problem'`

(Both the same)

T is short for 'translate.'



```
Start Page en.yml ja.yml new.html.erb X
1 <h1>New problem</h1>
2
3 <%= render 'form' %>
4
5 <%= link_to 'Back', problems_path %>
6
```

Embedded Ruby

This file new.html.erb is a HTML based file, with EEmbedded RuBy.

Ruby expressions are embedded with the following code:

```
<% (Ruby Expression) %>
```

Especially when the ruby variables or the ruby executed results are displayed, the code is:

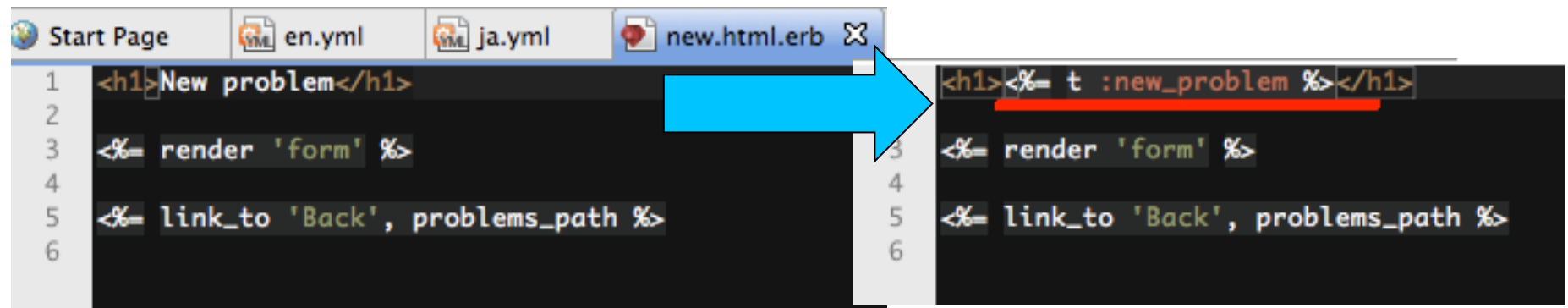
```
<%= (Ruby Expression) %>
```

To show the Translation result, the code is

```
<%= t :symbol %> or <%= t `symbol' %>
```


Replace Literal to symbols

- 1) Replace String Literal to (t symbol).
- 2) Prepare the translation in all dictionaries.



The screenshot shows a code editor with three tabs: 'en.yml', 'ja.yml', and 'new.html.erb'. The 'new.html.erb' tab is active and shows two versions of the code. A blue arrow points from the original code on the left to the modified code on the right. In the original code, the first line is `<h1>New problem</h1>`. In the modified code, it is `<h1><%= t :new_problem %></h1>`. The rest of the code, `<%= render 'form' %>` and `<%= link_to 'Back', problems_path %>`, remains unchanged.

```
4 en:  
5   new_problem: "New problem"  
6   hello: "hello world"  
7   date:  
8     abbr_day_names:  
9       - Sun  
10      - Mon  
11      - Tue
```

```
1 ja:  
2   new_problem: "問題の登録"  
3   hello: "はいさい"  
4   date:  
5     abbr_day_names:  
6     - 日
```

Try first, to see the current screen

- Most of our computers have the default local as ja_JP. So, anyway try the following two URLs.

<http://127.0.0.1:3000/problems/new?locale=ja>

or

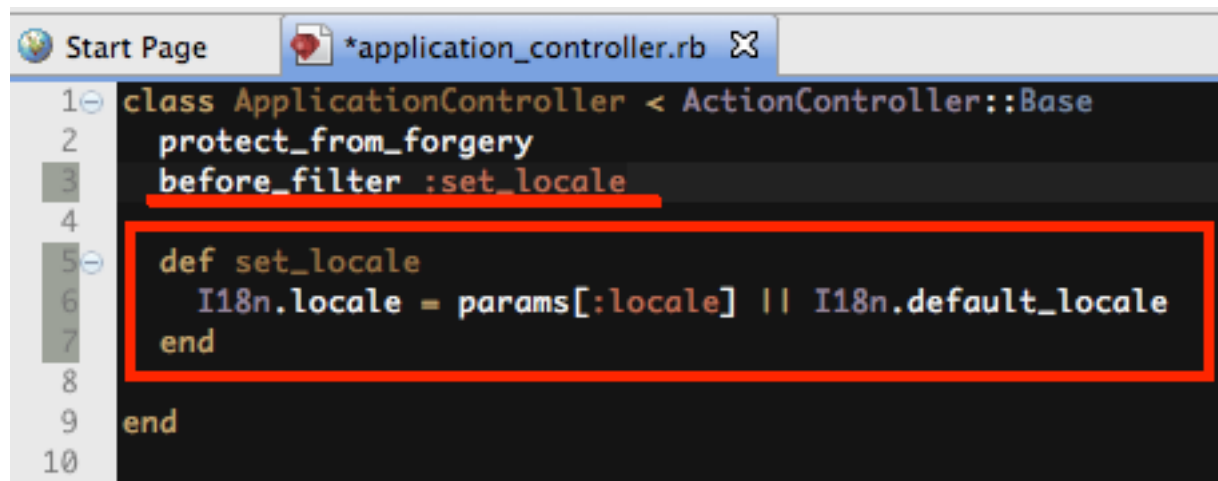
<http://127.0.0.1:3000/problems/new?locale=en>

- Both should be the same.

Tell the controller to preprocess

To tell the system to handle the explicitly set locale when specified, edit the file:

`spielberg\app\controllers\application_controller.rb`



```
1 class ApplicationController < ActionController::Base
2   protect_from_forgery
3   before_filter :set_locale
4
5   def set_locale
6     I18n.locale = params[:locale] || I18n.default_locale
7   end
8
9 end
10
```

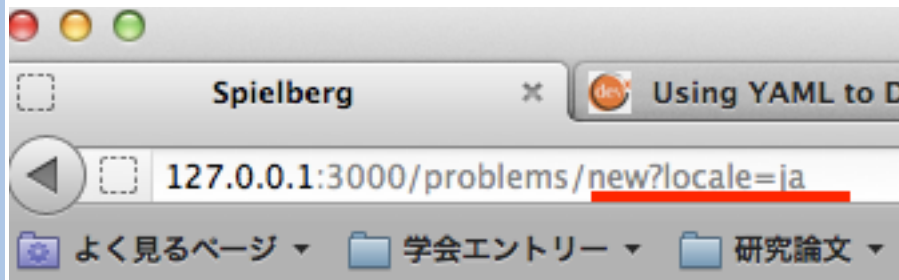
application_controller.rb

```
class ApplicationController < ActionController::Base
  protect_from_forgery
  before_filter :set_locale

  def set_locale
    I18n.locale = params[:locale] || I18n.default_locale
  end
end
```

If the locales are switched,

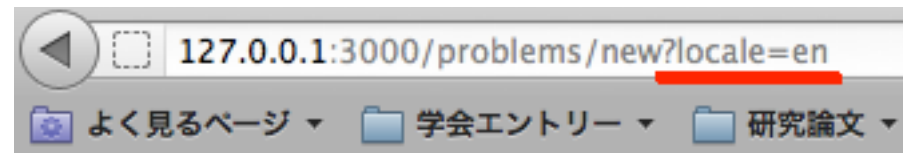
- We have done today's practice!



問題の登録

Title

Content



New problem

Title

Content



Today's report theme

none



Prepare for the next week

We will learn Test Driven Development, and the “Validation” together with “Verification.”