

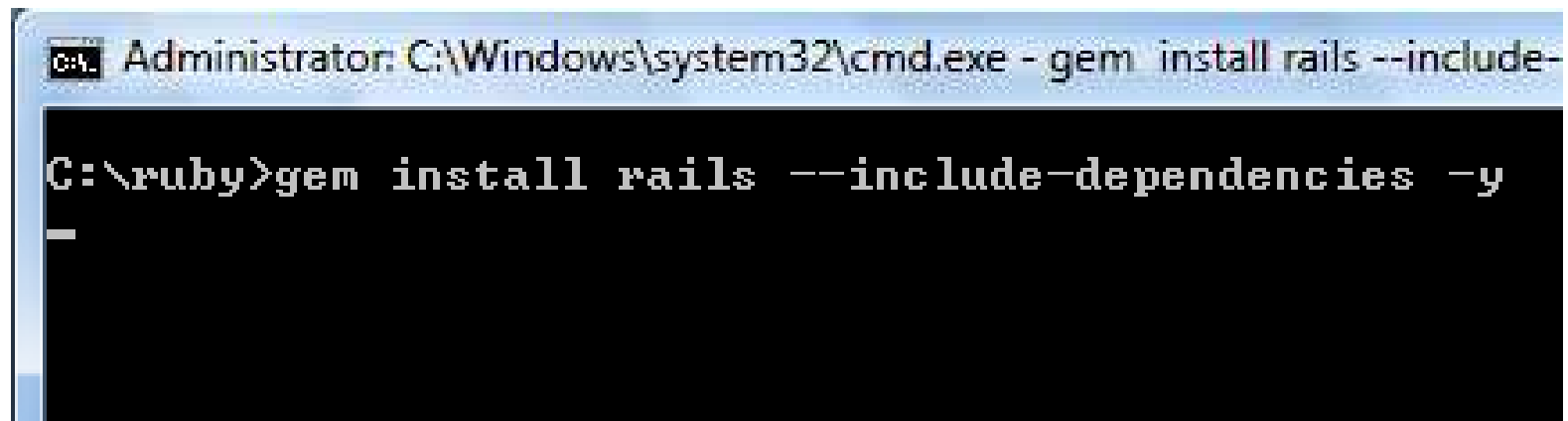
Rails 2.0

Barry Burd
Barry@BurdBrain.com

This slide presentation © 2008 Barry Burd
Copyrights for Ruby, Rails and other material are held by their
respective owners.

Install Ruby and Rails

- Ruby - <http://www.ruby-lang.org/en/downloads/>
- Rails -



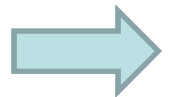
```
Administrator: C:\Windows\system32\cmd.exe - gem install rails --include-de-  
C:\ruby>gem install rails --include-dependencies -y  
-
```

Install SQLite

- <http://www.sqlite.org/download.html>

sqlite-3.5.4.zip (207.94 KiB)	A command-line program for accessing and modifying SQLite databases. See the documentation for additional information.
tclsqlite-3.5.4.zip (277.29 KiB)	Bindings for Tcl/Tk. You can import this shared library into either tclsh or wish to get SQLite database access from Tcl/Tk. See the documentation for details.
sqlitedll-3.5.4.zip (206.33 KiB)	This is a DLL of the SQLite library without the TCL bindings. The only external dependency is MSVCRT.DLL.
sqlite3_analyzer-3.5.4.zip (540.03 KiB)	An analysis program for database files compatible with SQLite version 3.5.4 and later.

Copy the uncompressed files to your ruby\bin directory



Install SQLite (continued)

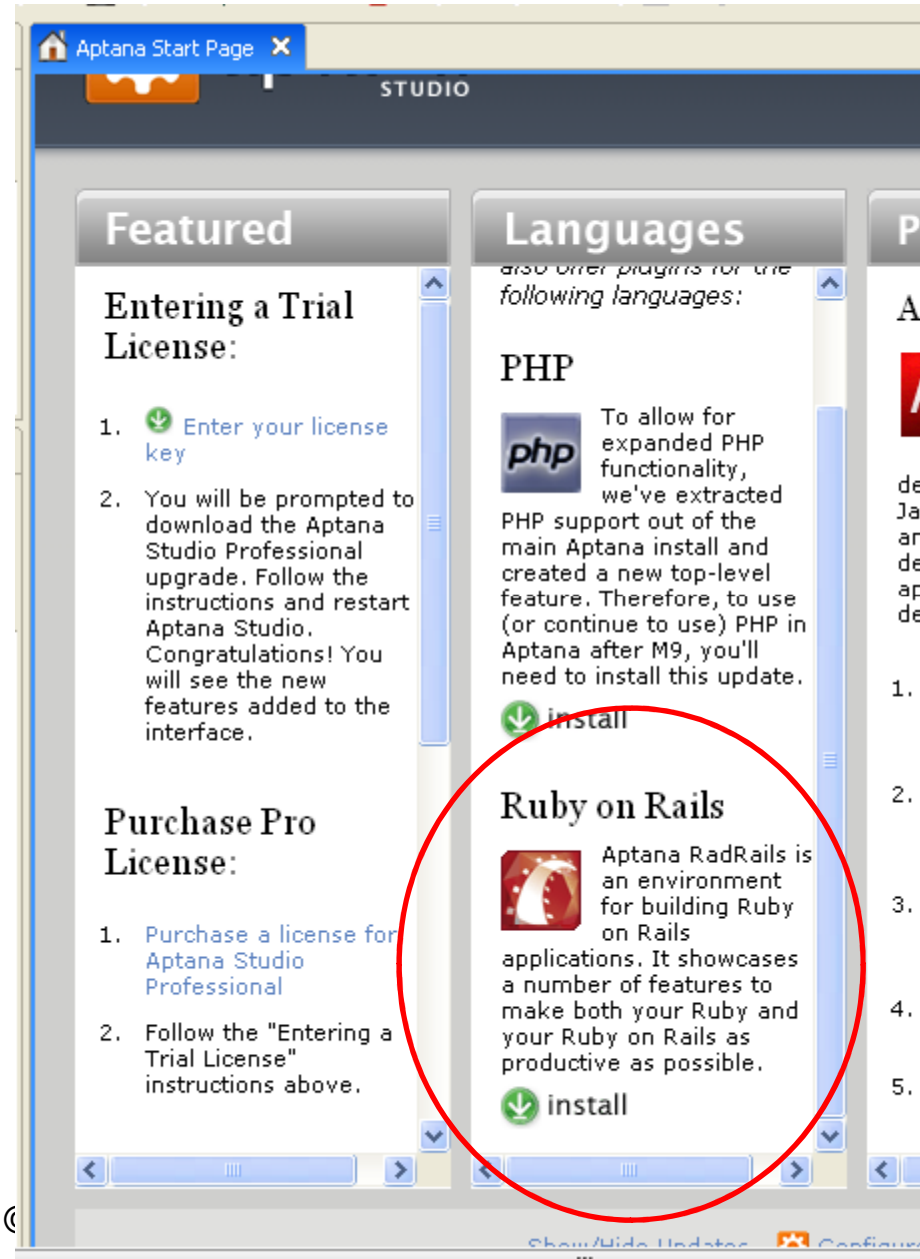


```
Administrator: C:\Windows\system32\cmd.exe
C:\ruby>gem install sqlite3-ruby --include-dependencies -y
```

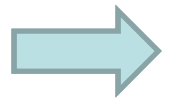
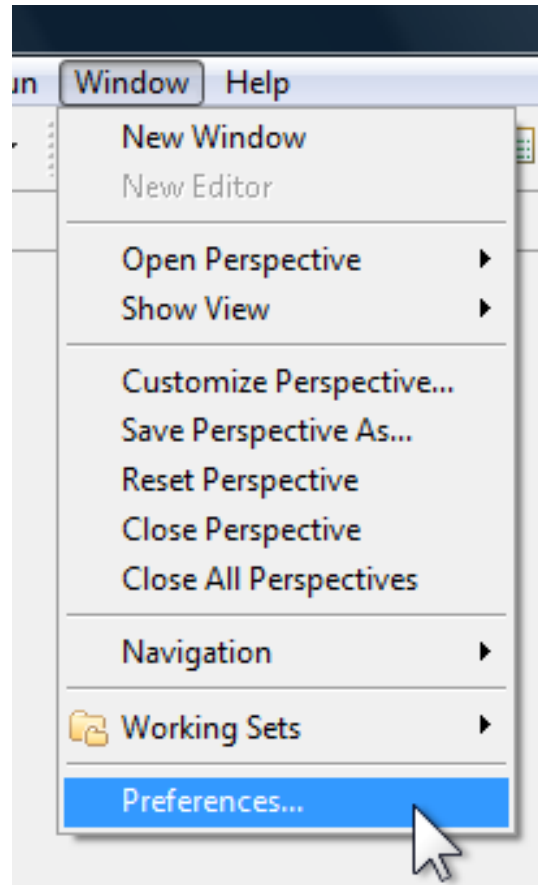
Install the Aptana IDE

- You need the Java Runtime -
<http://java.sun.com/javase/downloads/index.jsp>
- Aptana - <http://www.aptana.com/download/index.php>

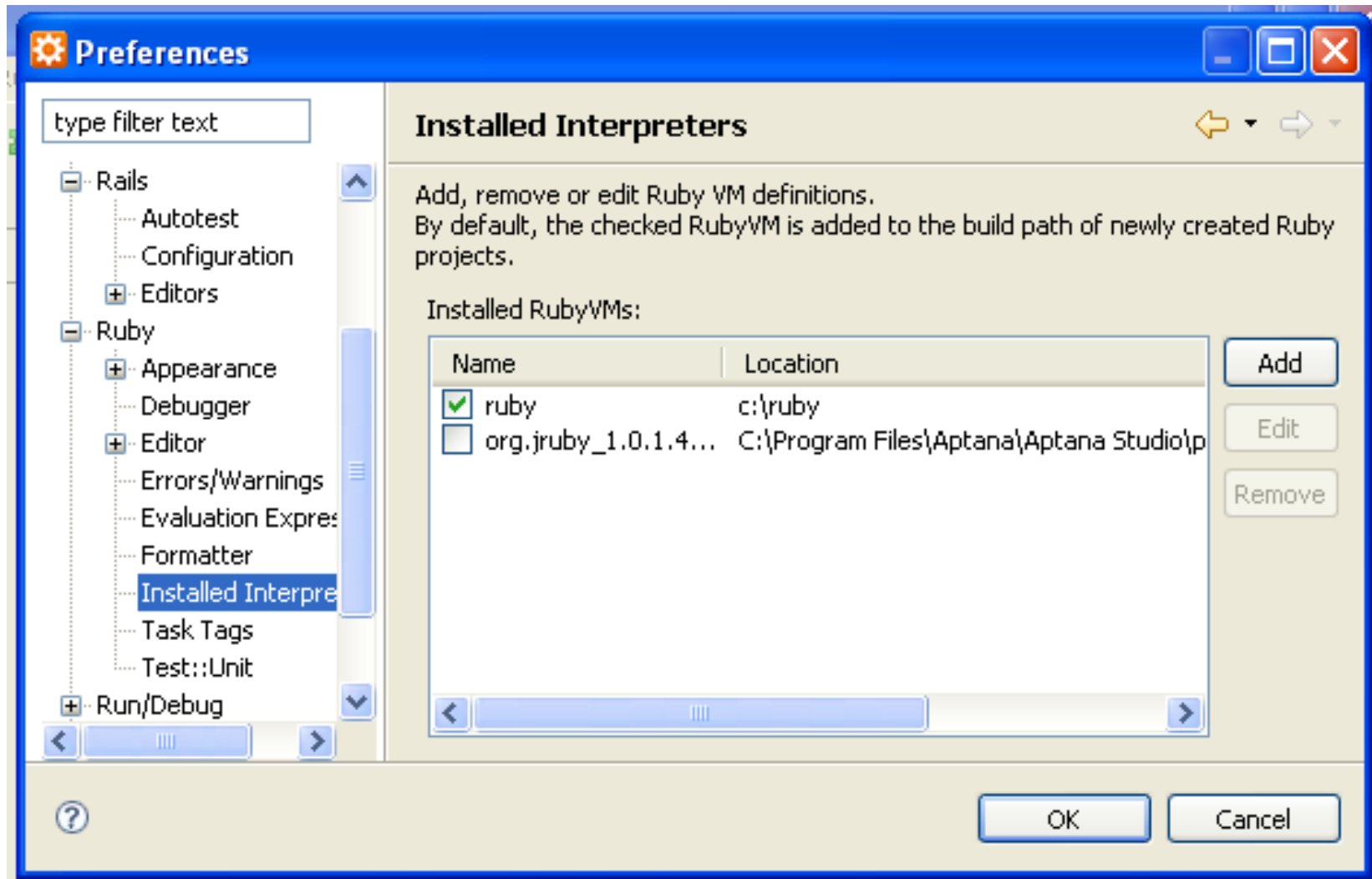
Install the RadRails plugin within Aptana



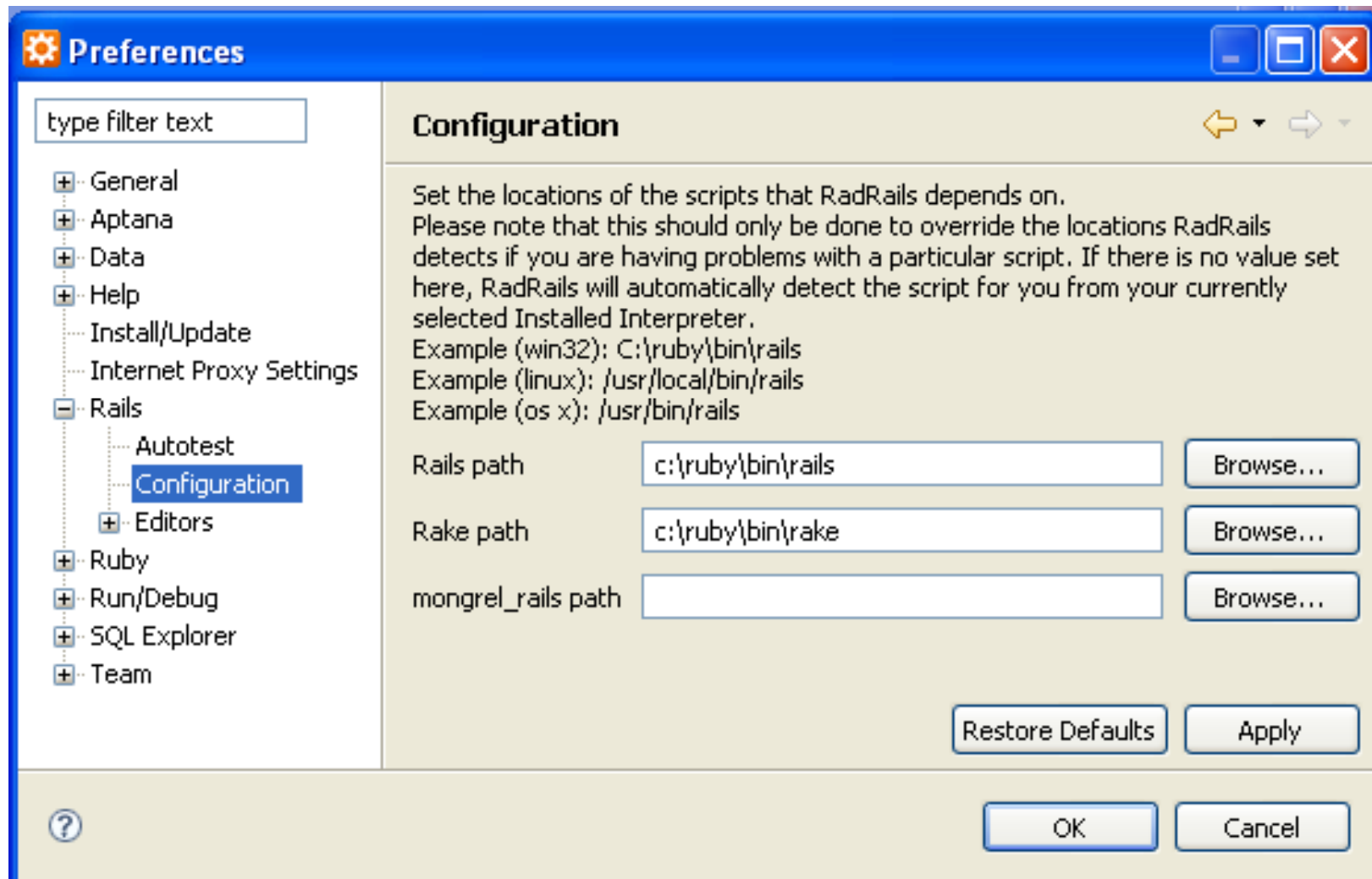
Configure RadRails



Configure RadRails



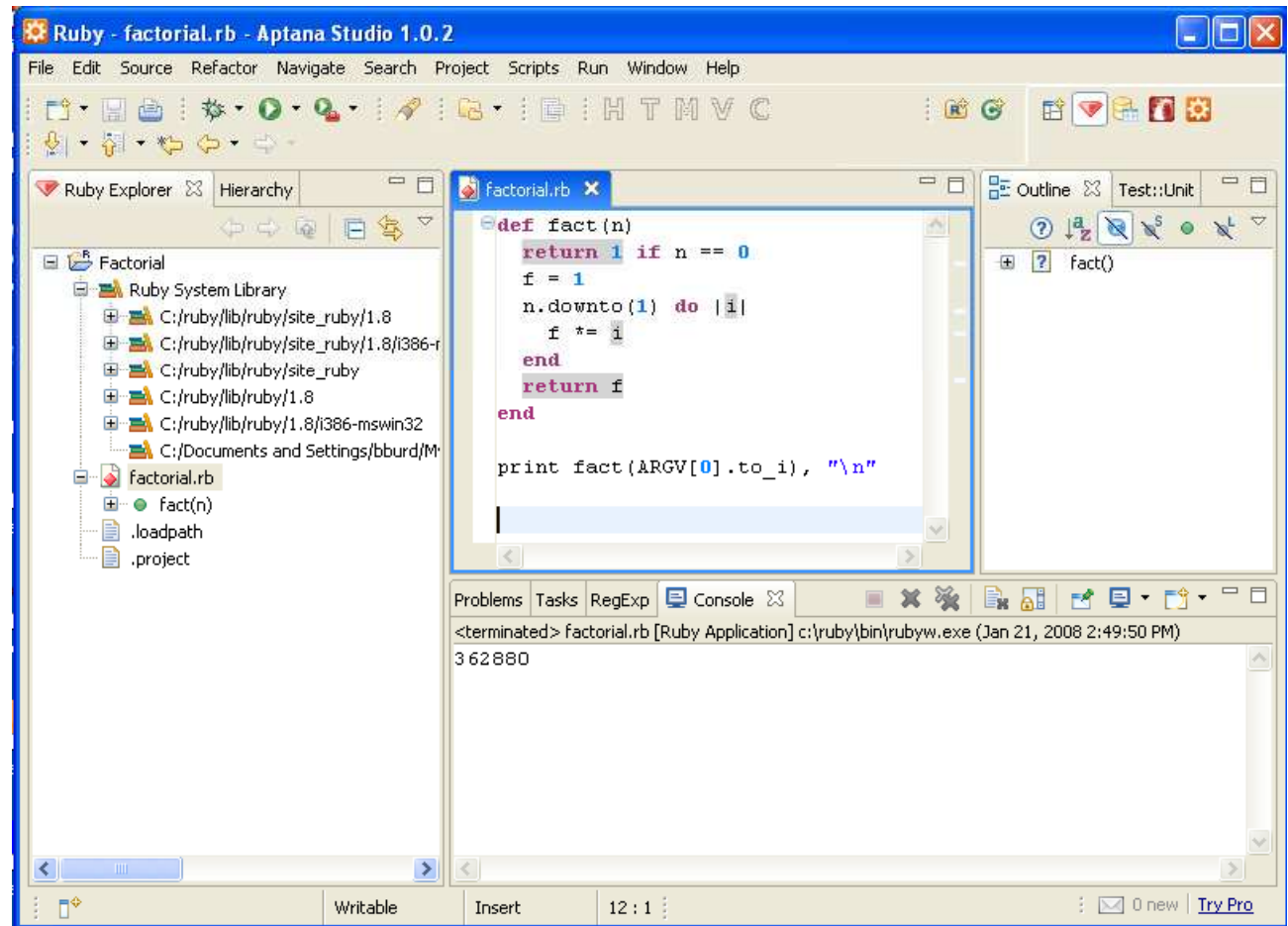
Configure RadRails

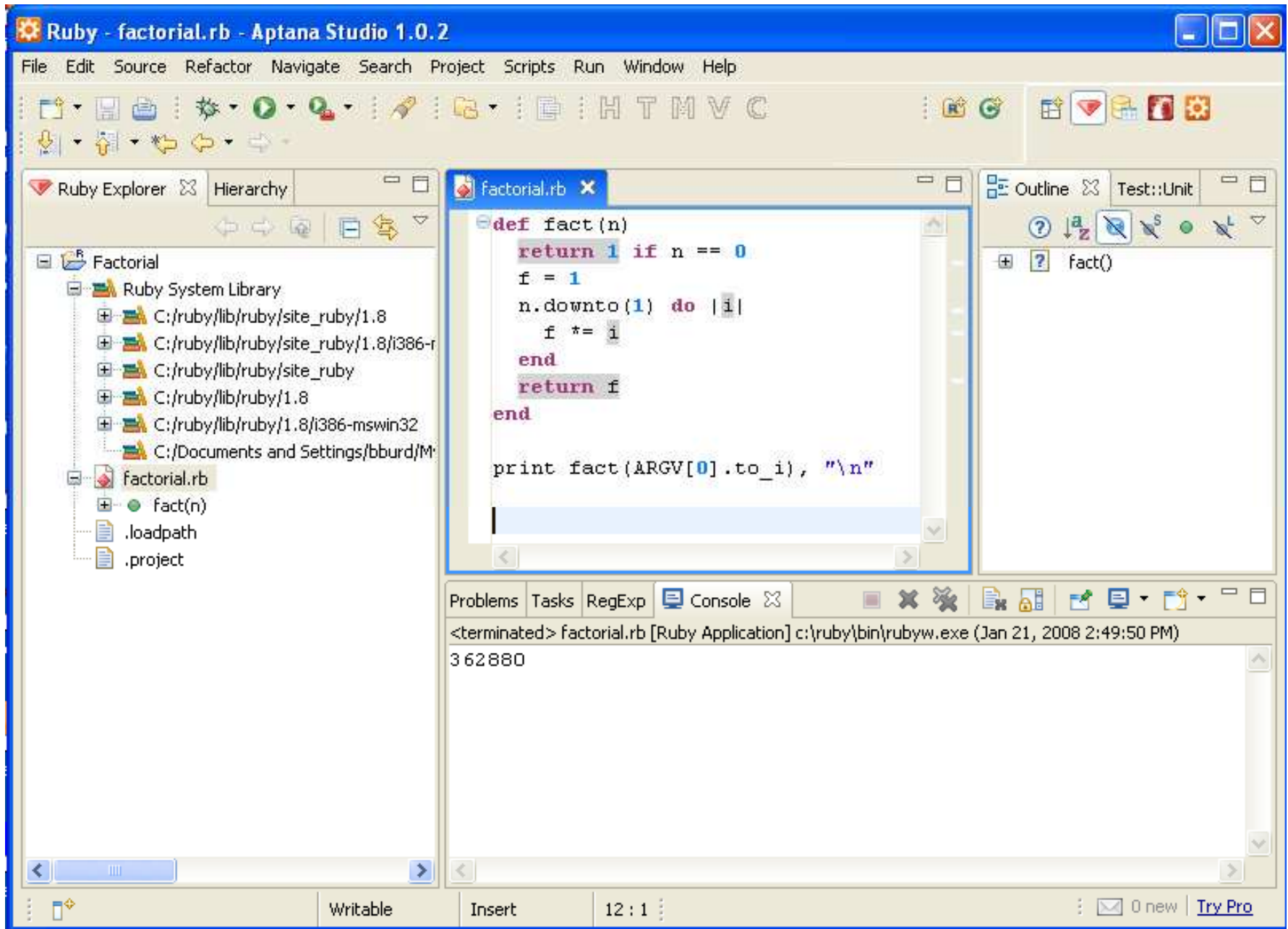


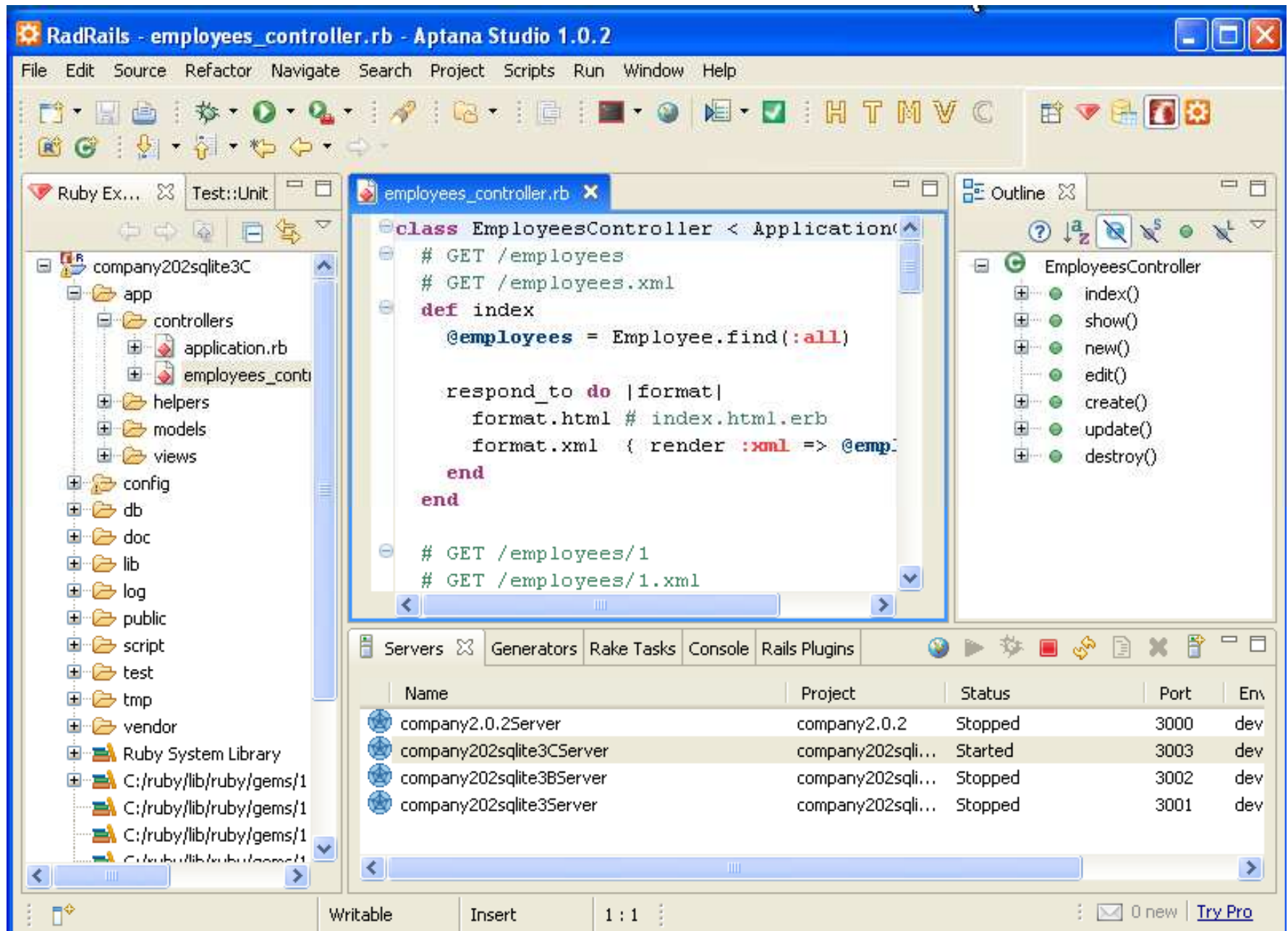
The Aptana IDE

- Based on Eclipse

- View
- Editor
- Part
- Area
- Perspective

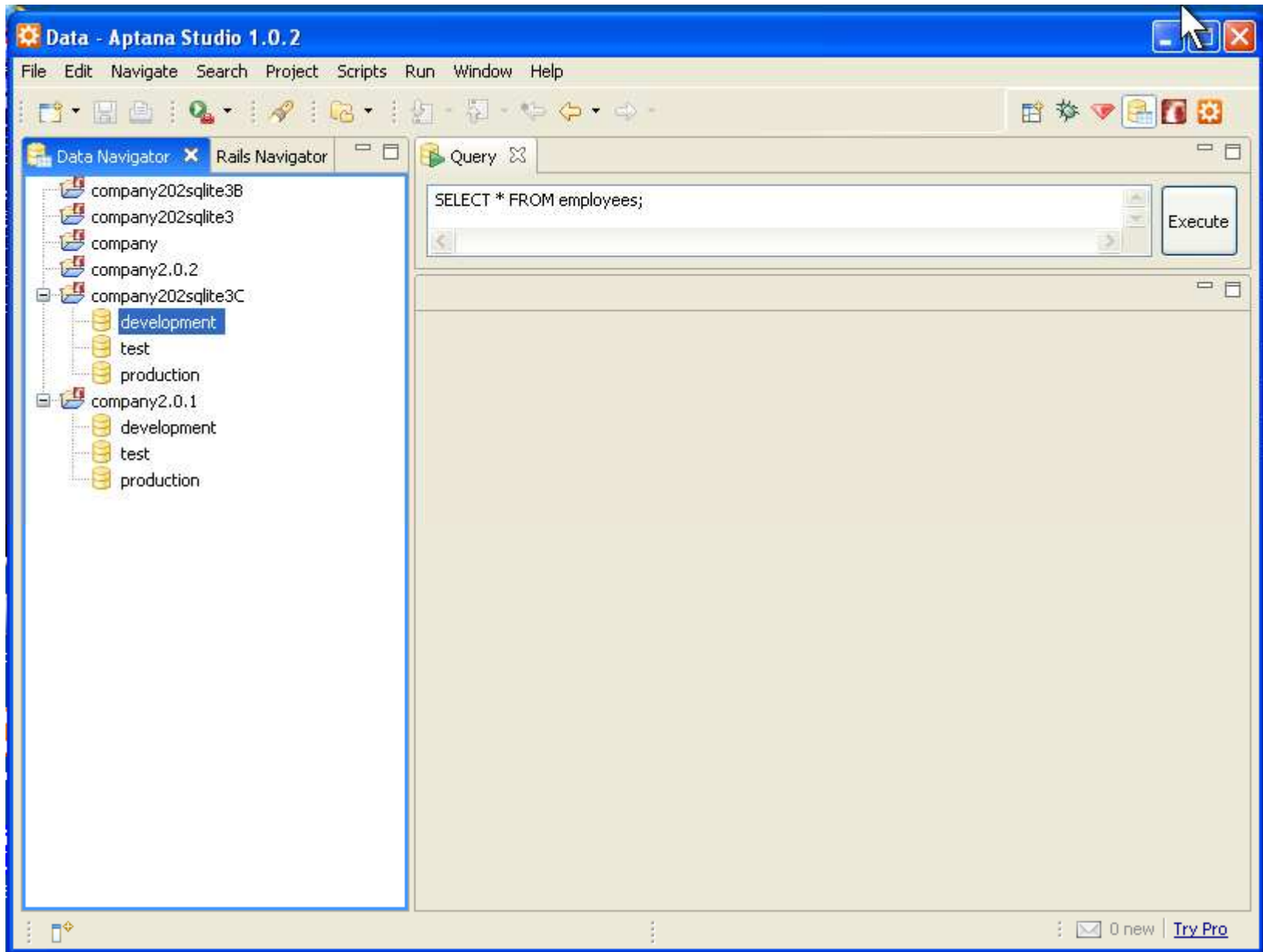


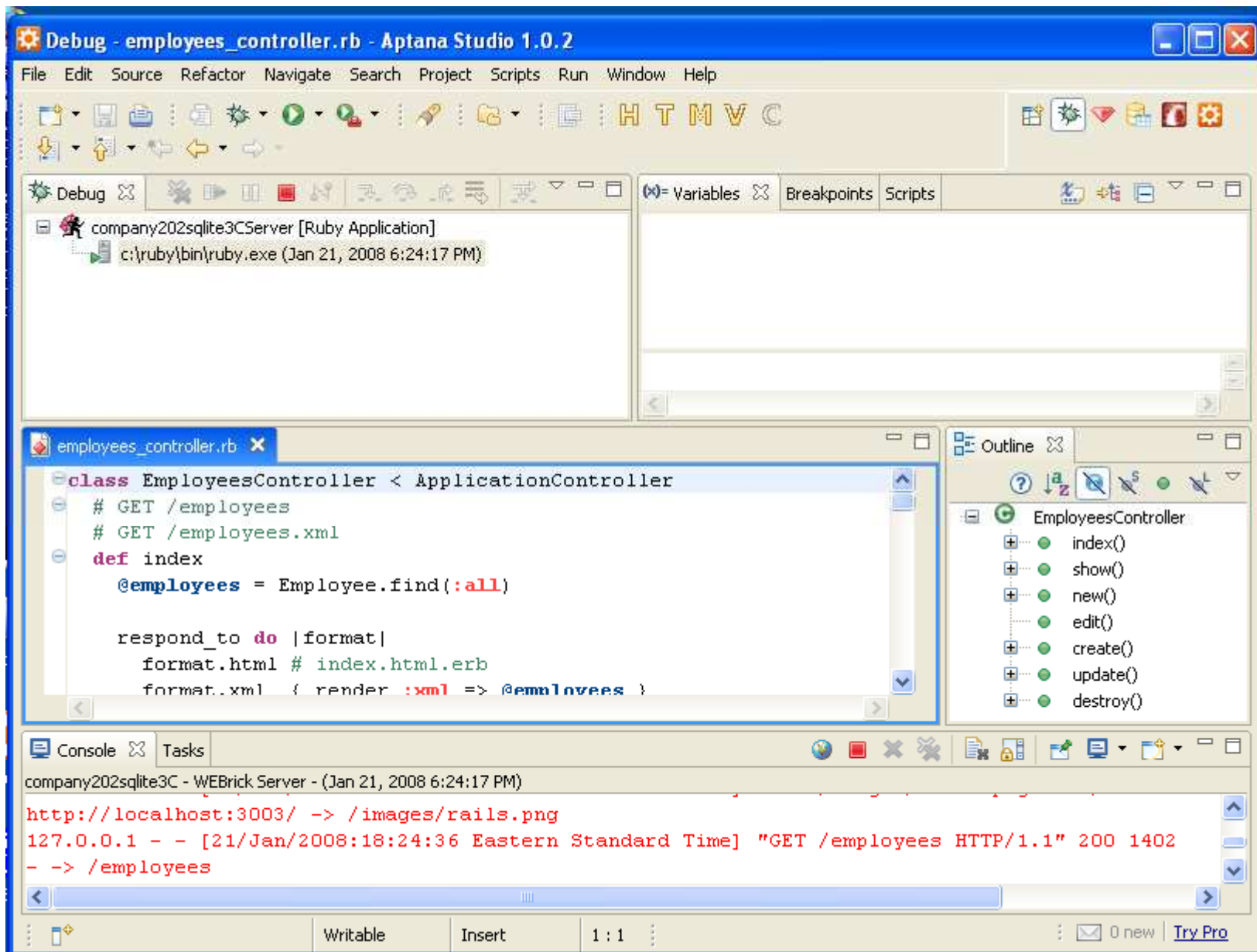




D
a
t
a

P
e
r
s
p
e
c
t
i
v
e





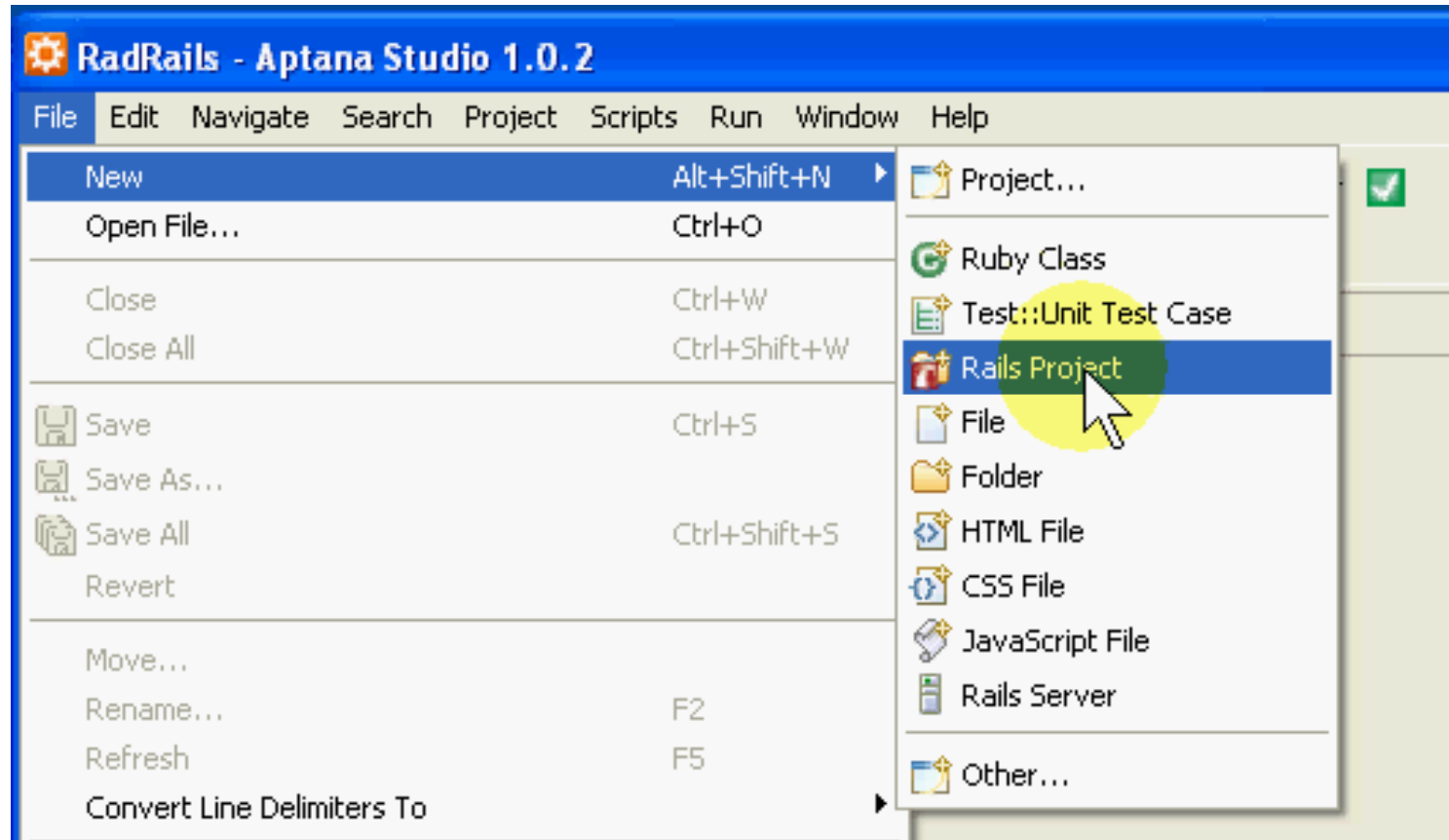
Watch a movie...

<http://www.burdbrain.com/RubyOnRails/Rails2.wmv>

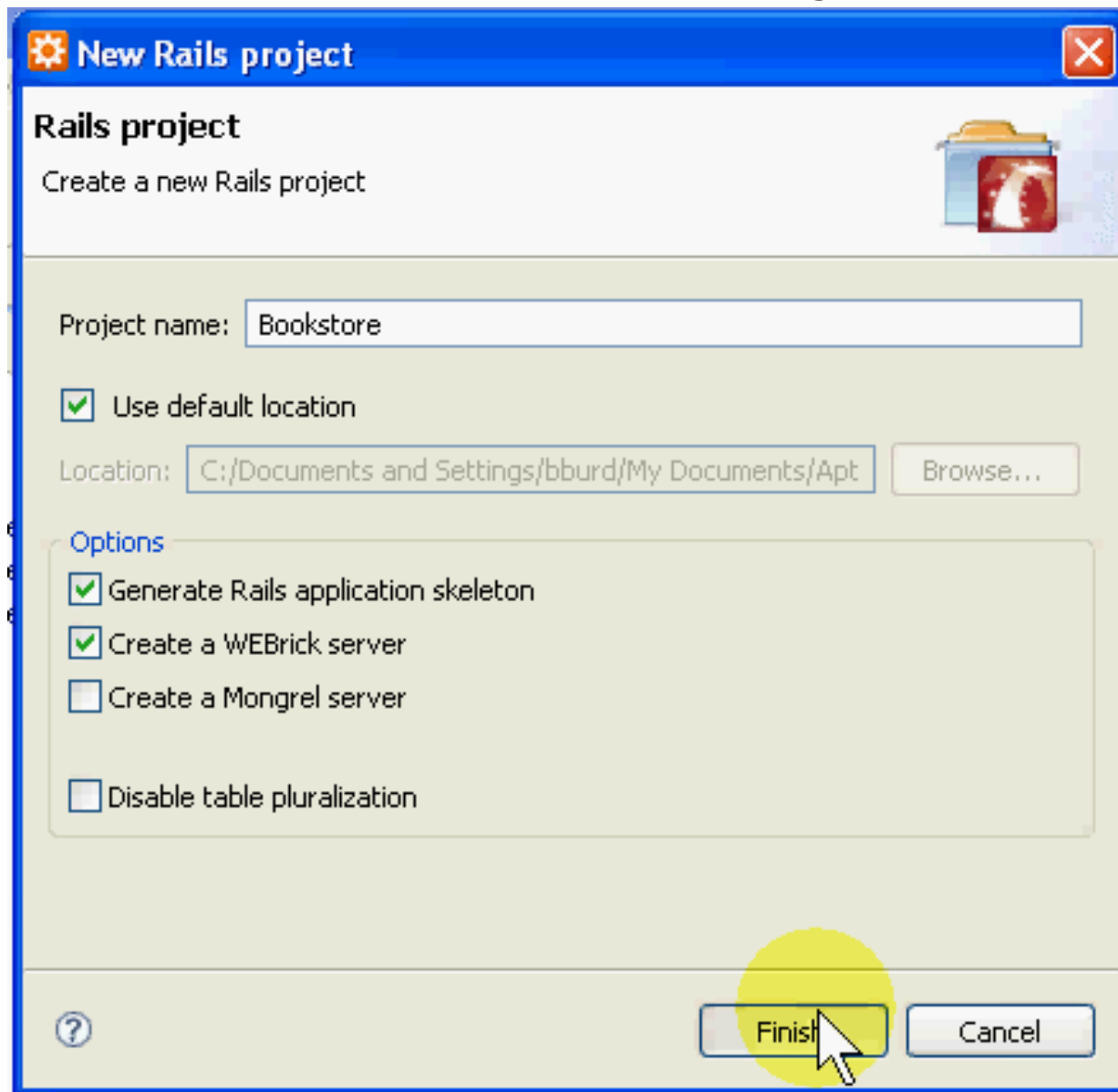
RadRails Perspective



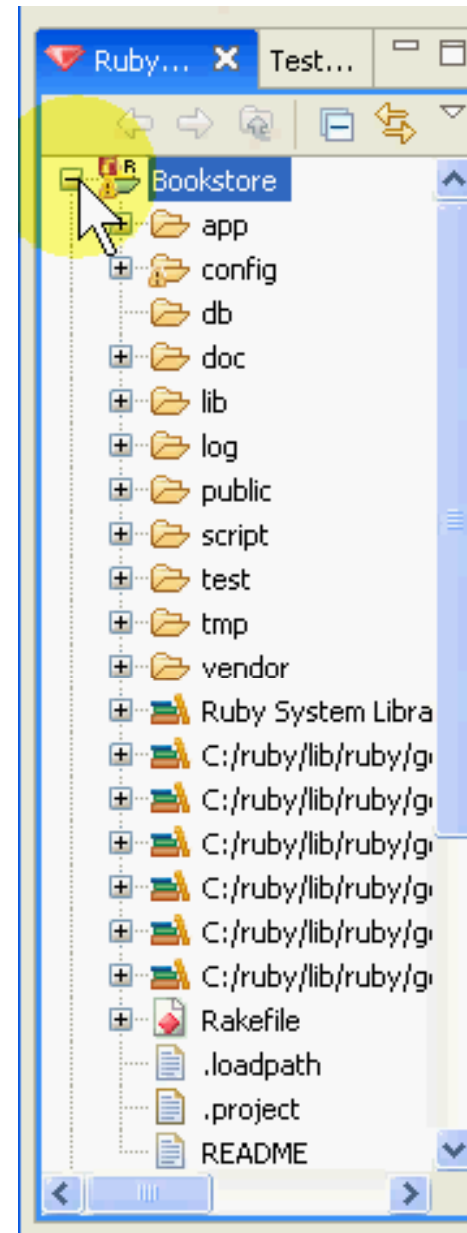
Create a new Rails Project



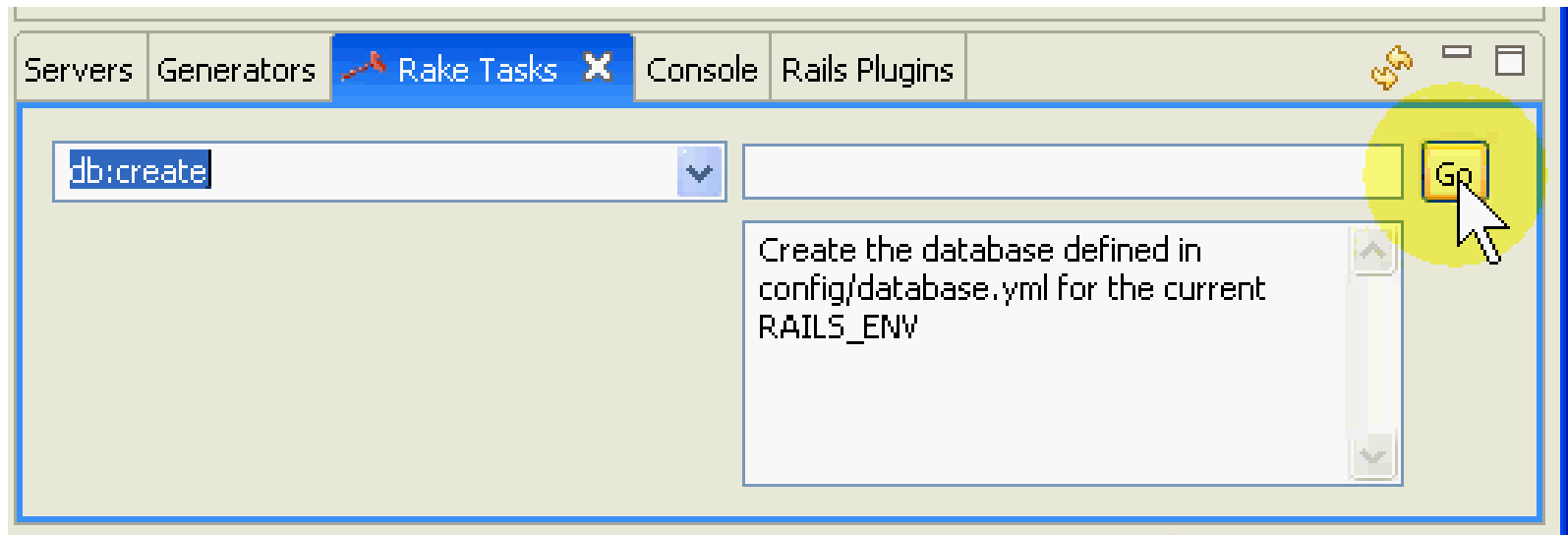
Name the project



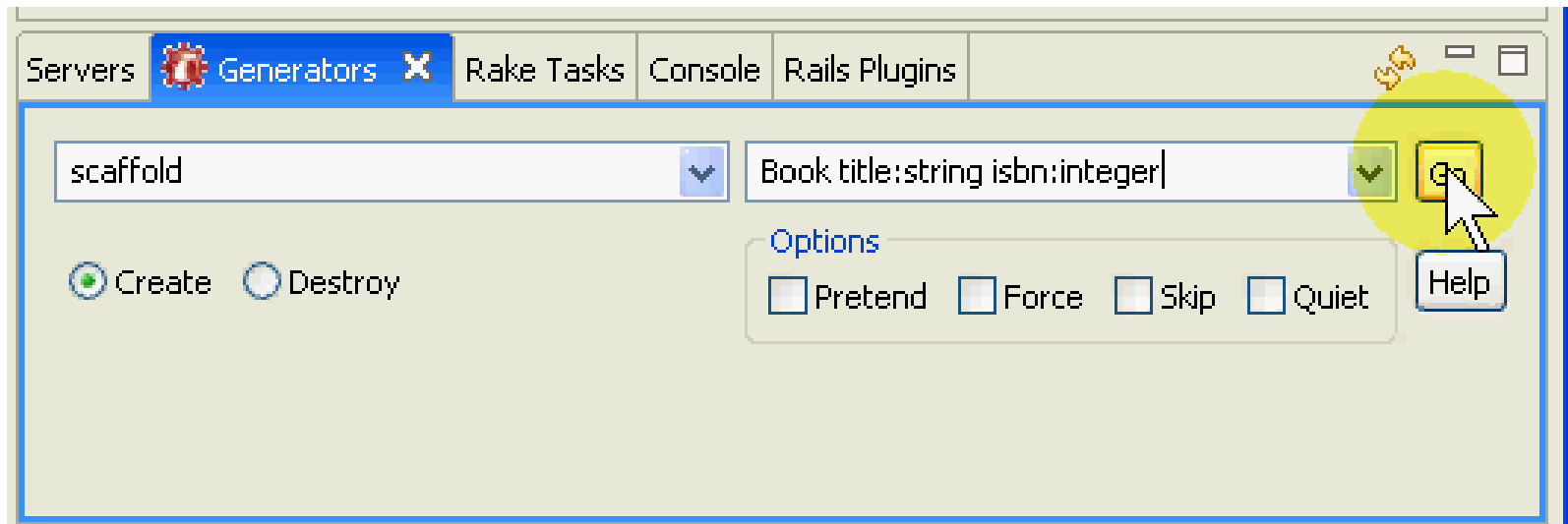
Rails creates folders and files



Create a database

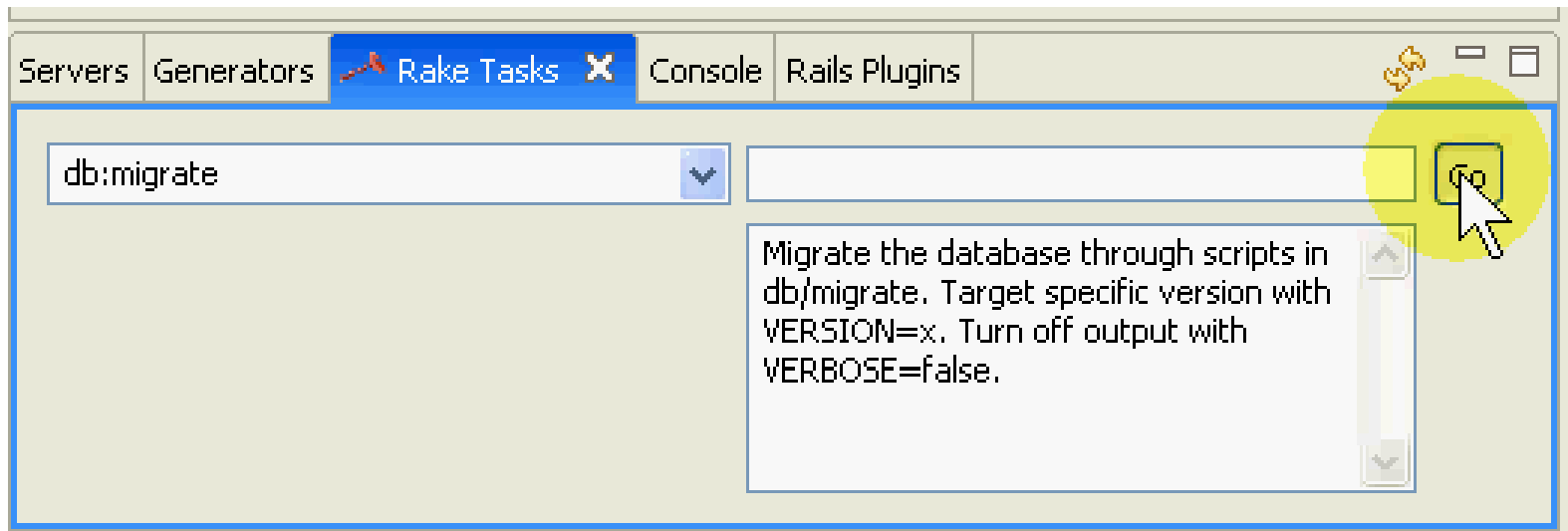


Generate a scaffold

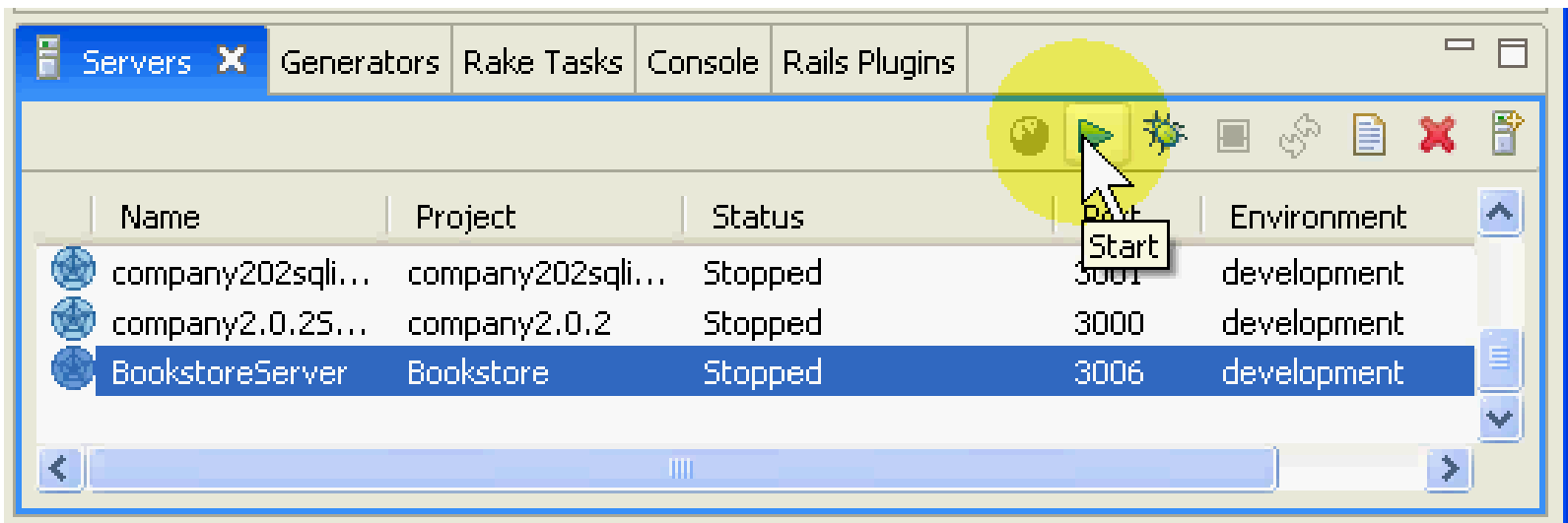


Migrate the database

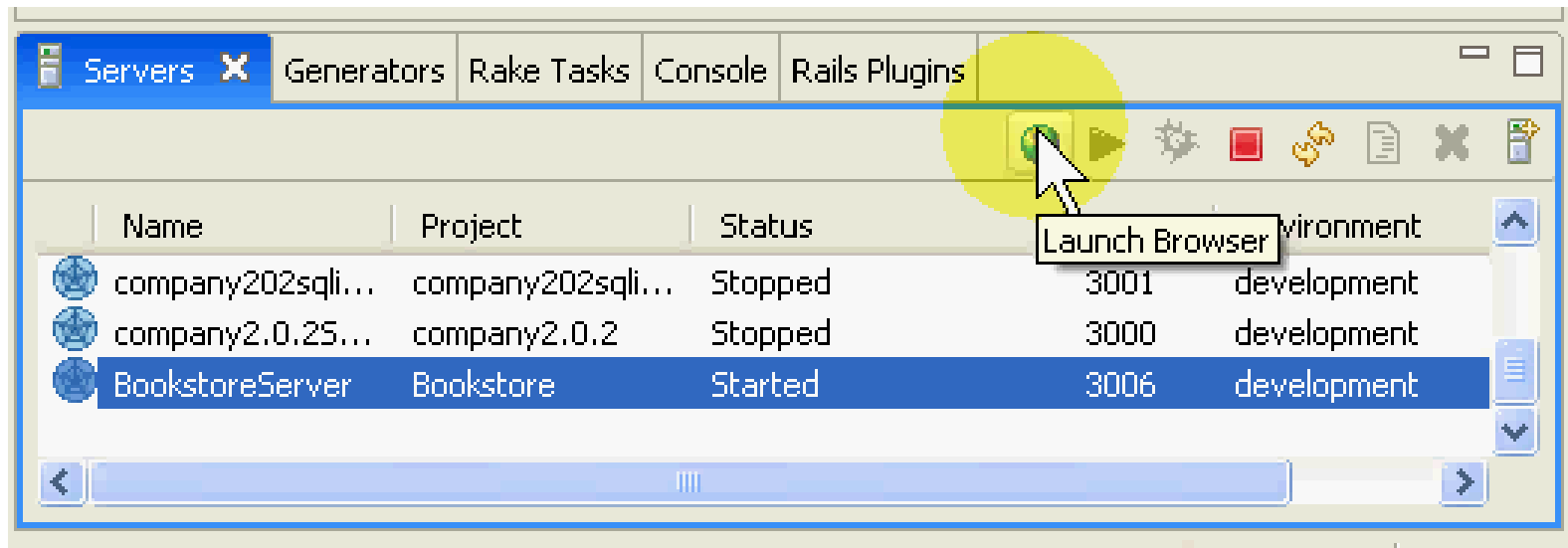
Create table(s) and column(s)



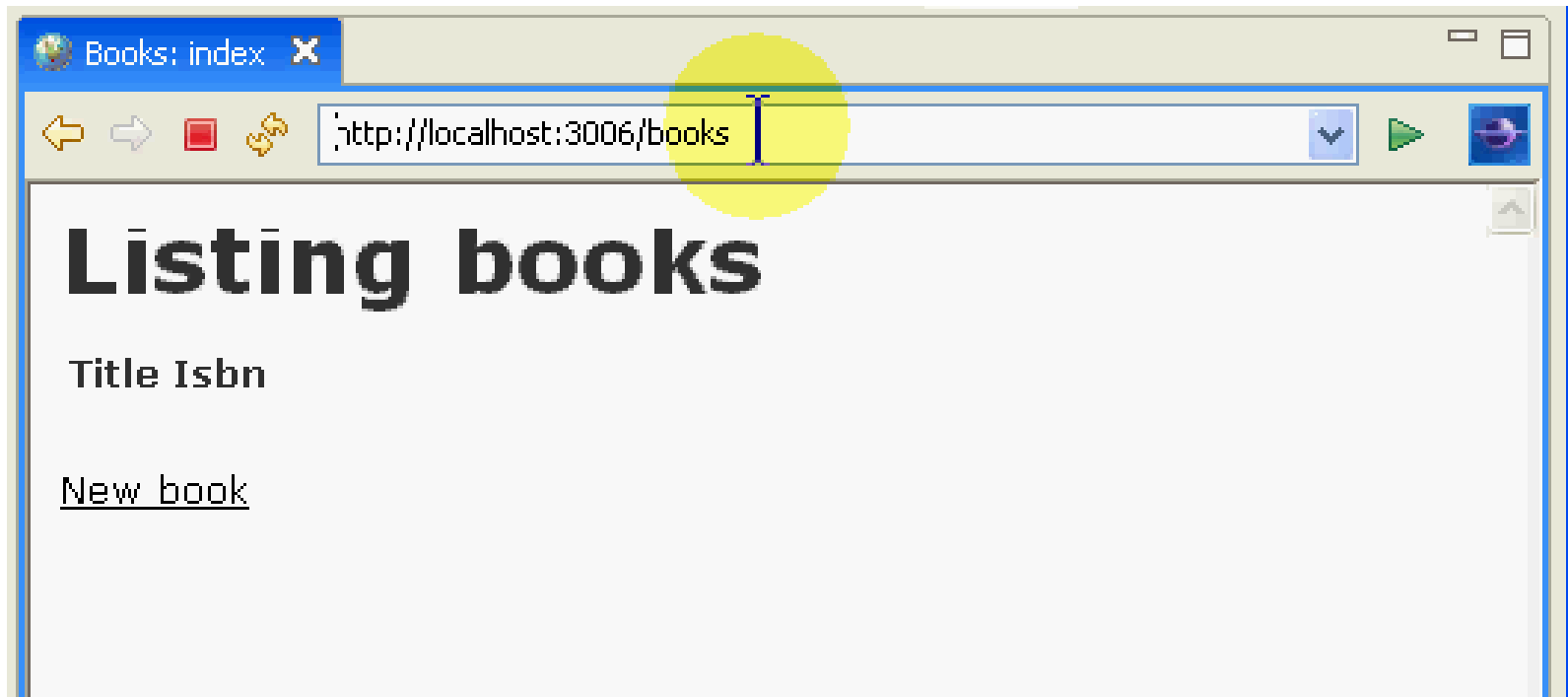
Start the project's Web server



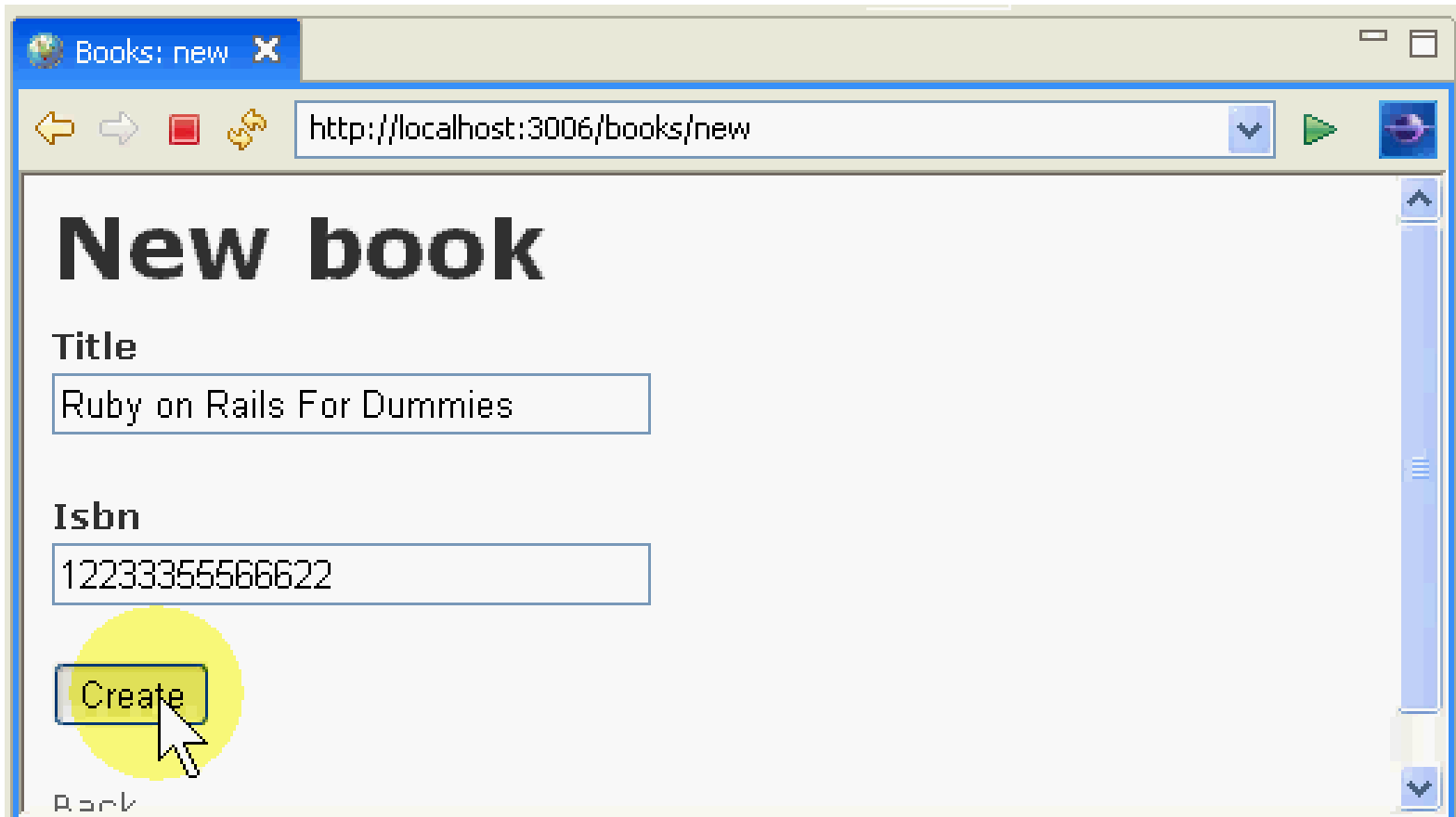
Visit the Web server's Welcome page



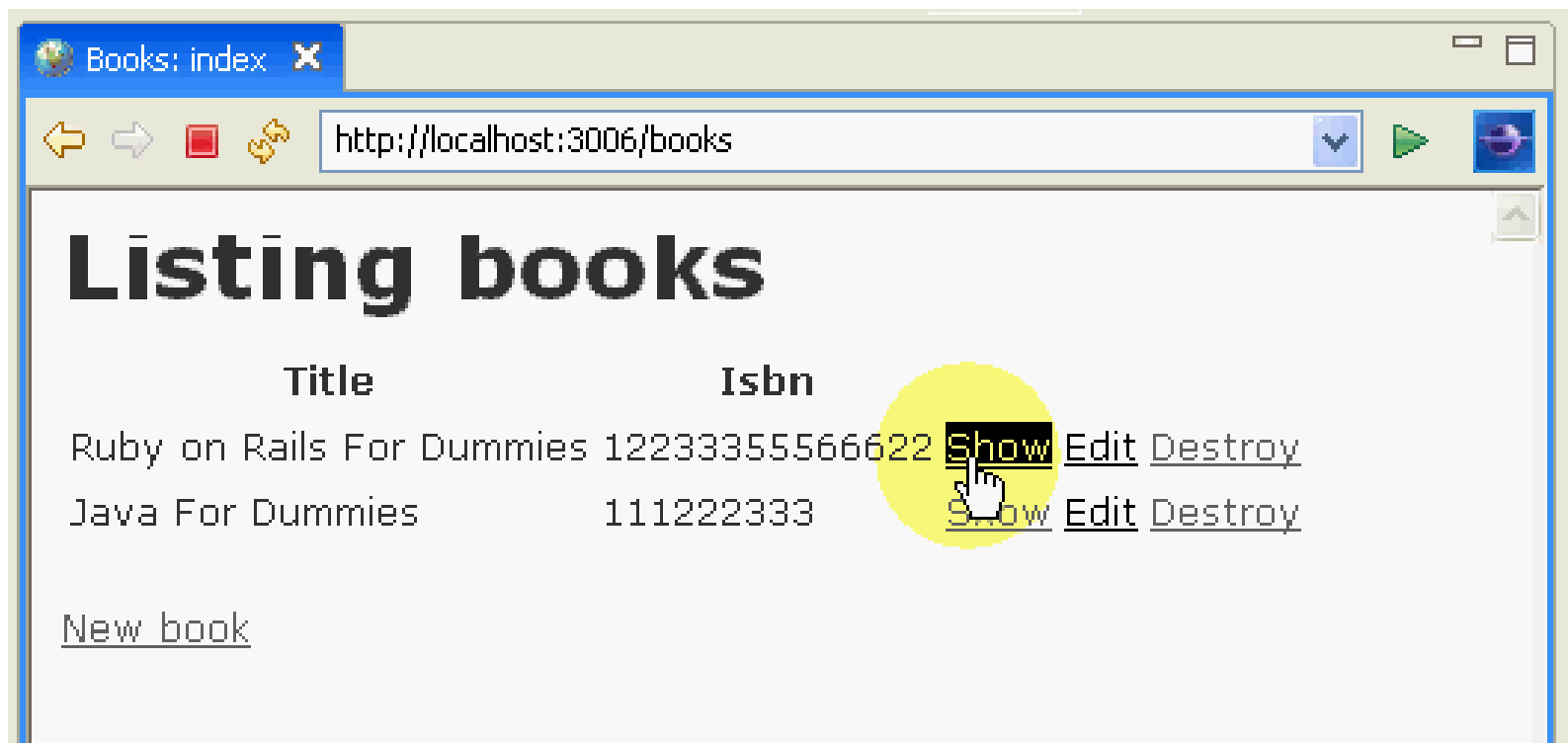
Visit the “list” page



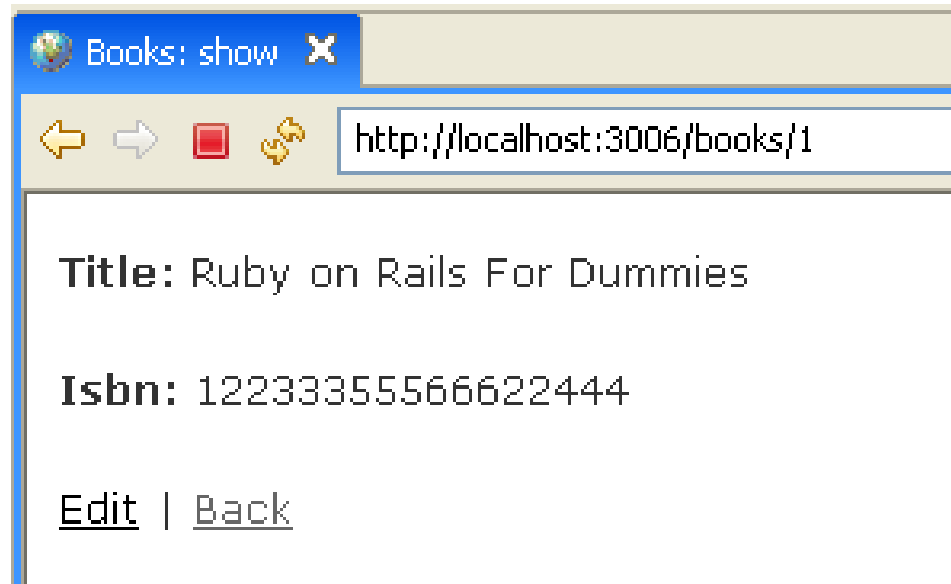
Add a new record



The screenshot shows a web browser window with the title 'Books: new'. The address bar contains the URL 'http://localhost:3006/books/new'. The main content area displays a form titled 'New book'. The form has two input fields: 'Title' with the text 'Ruby on Rails For Dummies' and 'Isbn' with the text '12233355566622'. A yellow circle highlights the 'Create' button, which has a mouse cursor pointing to it. Below the 'Create' button, the word 'Rank' is partially visible.



Show an existing record



Edit an existing record

Books: edit

← → ⏹ ↻ http://localhost:3006/books/1/edit

Editing book

Title

Isbn

[Show](#) | [Back](#)

Several types of fields



The screenshot shows a web browser window with the following elements:

- Tab: list.rhtml Employees: new
- Address bar: http://localhost:3013/employees/new
- Form Title: **New employee**
- Form Fields:
 - Name: Text input field containing "Barry Burd"
 - Salary: Text input field containing "100000000.00"
 - Fulltime: Dropdown menu with "False" selected
 - Level: Text input field containing "10"
 - Hiredate: Three dropdown menus for year (2006), month (August), and day (22)
- Buttons: "Create" button (highlighted with a mouse cursor) and a "Back" link.

The Migration File

Bookstore\db\migrate\001_create_books.rb

```
001_create_books.rb x
class CreateBooks < ActiveRecord::Migration
  def self.up
    create_table :books do |t|
      t.string :title
      t.integer :isbn

      t.timestamps
    end
  end

  def self.down
    drop_table :books
  end
end
```

Bookstore\config\database.yml

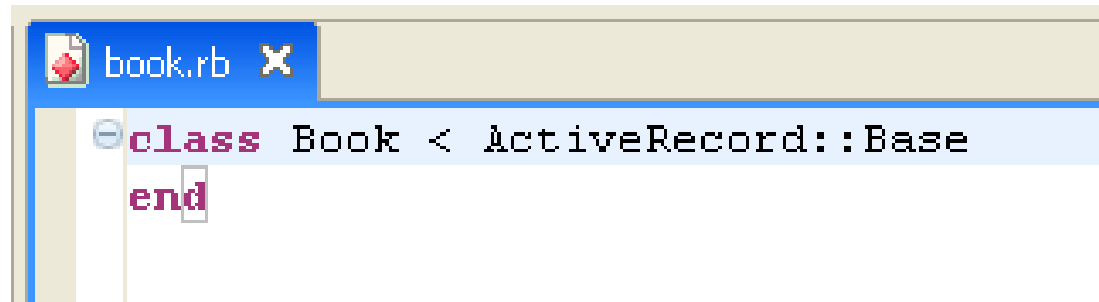
```
database.yml x
# SQLite version 3.x
#   gem install sqlite3-ruby (not necessary on
development:
  adapter: sqlite3
  database: db/development.sqlite3
  timeout: 5000

# Warning: The database defined as 'test' will
# re-generated from your development database
# Do not set this db to the same as developmen
test:
  adapter: sqlite3
  database: db/test.sqlite3
  timeout: 5000

production:
  adapter: sqlite3
  database: db/production.sqlite3
  timeout: 5000
```


The Model Code

Bookstore\app\models\book.rb



```
class Book < ActiveRecord::Base
end
```

The Controller Code

Bookstore\app\controllers\books_controller.rb

```
books_controller.rb x
class BooksController < ApplicationController
  # GET /books
  # GET /books.xml
  def index
    @books = Book.find(:all)

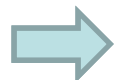
    respond_to do |format|
      format.html # index.html.erb
      format.xml { render :xml => @books }
    end
  end

  # GET /books/1
  # GET /books/1.xml
  def show
    @book = Book.find(params[:id])

    respond_to do |format|
      format.html # show.html.erb
      format.xml { render :xml => @book }
    end
  end
end
```

"if the client wants HTML in response to this action, just respond as we would have before, but if the client wants XML, return them the list of people in XML format." (Rails determines the desired response format from the HTTP Accept header submitted by the client.)"

Let the instance variable (the attribute) `@book` be the record you get when you look for a record whose id matches the incoming id parameter.



The Controller Code

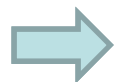
Bookstore\app\controllers\books_controller.rb

```
books_controller.rb x
```

```
# GET /books/new
# GET /books/new.xml
def new
  @book = Book.new

  respond_to do |format|
    format.html # new.html.erb
    format.xml { render :xml => @book }
  end
end

# GET /books/1/edit
def edit
  @book = Book.find(params[:id])
end
```

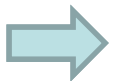


The Controller Code

Bookstore\app\controllers\books_controller.rb

```
# POST /books
# POST /books.xml
def create
  @book = Book.new(params[:book])

  respond_to do |format|
    if @book.save
      flash[:notice] = 'Book was successfully created.'
      format.html { redirect_to(@book) }
      format.xml { render :xml => @book, :status => :created, :location => @book }
    else
      format.html { render :action => "new" }
      format.xml { render :xml => @book.errors, :status => :unprocessable_entity }
    end
  end
end
end
```



The Controller Code

Bookstore\app\controllers\books_controller.rb

```
books_controller.rb x
# PUT /books/1
# PUT /books/1.xml
def update
  @book = Book.find(params[:id])

  respond_to do |format|
    if @book.update_attributes(params[:book])
      flash[:notice] = 'Book was successfully updated.'
      format.html { redirect_to(@book) }
      format.xml { head :ok }
    else
      format.html { render :action => "edit" }
      format.xml { render :xml => @book.errors, :status => :unprocessable_entity }
    end
  end
end

# DELETE /books/1
# DELETE /books/1.xml
def destroy
  @book = Book.find(params[:id])
  @book.destroy

  respond_to do |format|
    format.html { redirect_to(books_url) }
    format.xml { head :ok }
  end
end
end
```

The Index Page

Bookstore\app\views\books\index.html.erb

```
index.html.erb x
<h1>Listing books</h1>

<table>
  <tr>
    <th>Title</th>
    <th>Isbn</th>
  </tr>

  <% for book in @books %>
    <tr>
      <td><%=h book.title %></td>
      <td><%=h book.isbn %></td>
      <td><%= link_to 'Show', book %></td>
      <td><%= link_to 'Edit', edit_book_path(book) %></td>
      <td><%= link_to 'Destroy', book, :confirm => 'Are you sure?', :method => :delete %></td>
    </tr>
  <% end %>
</table>

<br />

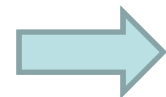
<%= link_to 'New book', new_book_path %>
```

View Source



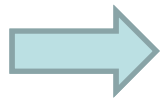
View Source

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0  
  Transitional//EN"  
    "http://www.w3.org/TR/xhtml1/DTD/xhtml1-  
  transitional.dtd">  
  
<html xmlns="http://www.w3.org/1999/xhtml"  
  xml:lang="en" lang="en">  
<head>  
  <meta http-equiv="content-type"  
    content="text/html; charset=UTF-8" />  
  <title>Books: index</title>  
  <link href="/stylesheets/scaffold.css?1201328122"  
    media="screen" rel="stylesheet" type="text/css" />  
</head>
```



View Source

```
<body>
<p style="color: green"></p>
<h1>Listing books</h1>
<table>
  <tr>
    <th>Title</th>
    <th>Isbn</th>
  </tr>
  <tr>
    <td>Ruby on Rails For Dummies</td>
    <td>12233355566622444</td>
    <td><a href="/books/1">Show</a></td>
    <td><a href="/books/1/edit">Edit</a></td>
    <td><a href="/books/1" onclick="if (confirm('Are
you sure?')) { var f = ... (some code omitted)
```



View Source

```
<body>
<p style="color: green"></p>
<h1>Listing books</h1>
<table>
  <tr>
    <th>Title</th>
    <th>Isbn</th>
  </tr>
  <tr>
    <td>Ruby on Rails For Dummies</td>
    <td>12233355566622444</td>
    <td><a href="/books/1">Show</a></td>
    <td><a href="/books/1/edit">Edit</a></td>
    <td><a href="/books/1" onclick="if (confirm('Are
you sure?')) { var f = ... (remaining code omitted)</td>
```

The Edit Page

Bookstore\app\views\books\edit.html.erb

```
edit.html.erb x
<h1>Editing book</h1>

<%= error_messages_for :book %>

<% form_for(@book) do |f| %>
  <p>
    <b>Title</b><br />
    <%= f.text_field :title %>
  </p>

  <p>
    <b>Isbn</b><br />
    <%= f.text_field :isbn %>
  </p>

  <p>
    <%= f.submit "Update" %>
  </p>
<% end %>

<%= link_to 'Show', @book %> |
<%= link_to 'Back', books_path %>
```

The New Page

Bookstore\app\views\books\new.html.erb

```
new.html.erb x
<h1>New book</h1>

<%= error_messages_for :book %>

<% form_for(@book) do |f| %>
  <p>
    <b>Title</b><br />
    <%= f.text_field :title %>
  </p>

  <p>
    <b>Isbn</b><br />
    <%= f.text_field :isbn %>
  </p>

  <p>
    <%= f.submit "Create" %>
  </p>
<% end %>

<%= link_to 'Back', books_path %>
```

The Show Page

Bookstore\app\views\books\show.html.erb

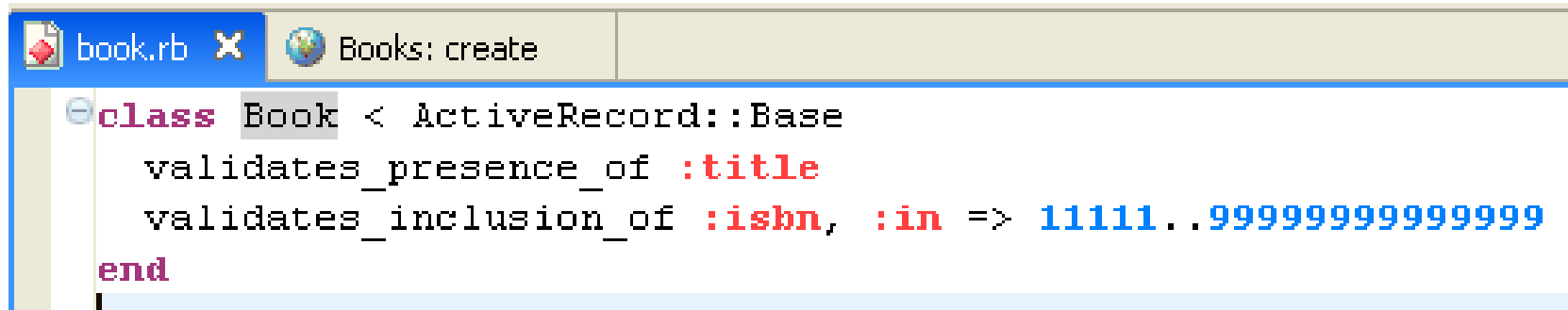
```
show.html.erb x
<p>
  <b>Title:</b>
  <%=h @book.title %>
</p>

<p>
  <b>Isbn:</b>
  <%=h @book.isbn %>
</p>

<%= link_to 'Edit', edit_book_path(@book) %> |
<%= link_to 'Back', books_path %>
```

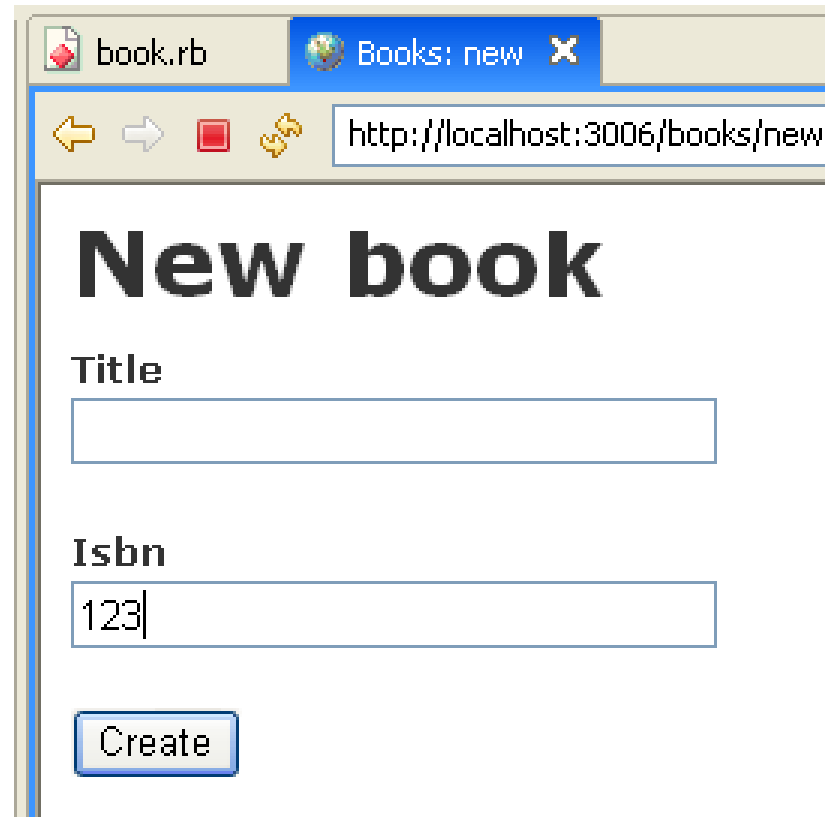
Fine tuning a Web application

Adding validation to the model code



```
book.rb x Books: create  
class Book < ActiveRecord::Base  
  validates_presence_of :title  
  validates_inclusion_of :isbn, :in => 1111..9999999999999999  
end
```

Attempting to create a new record, but...



book.rb Books: new X

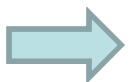
← → ⏏ 🔄 http://localhost:3006/books/new

New book

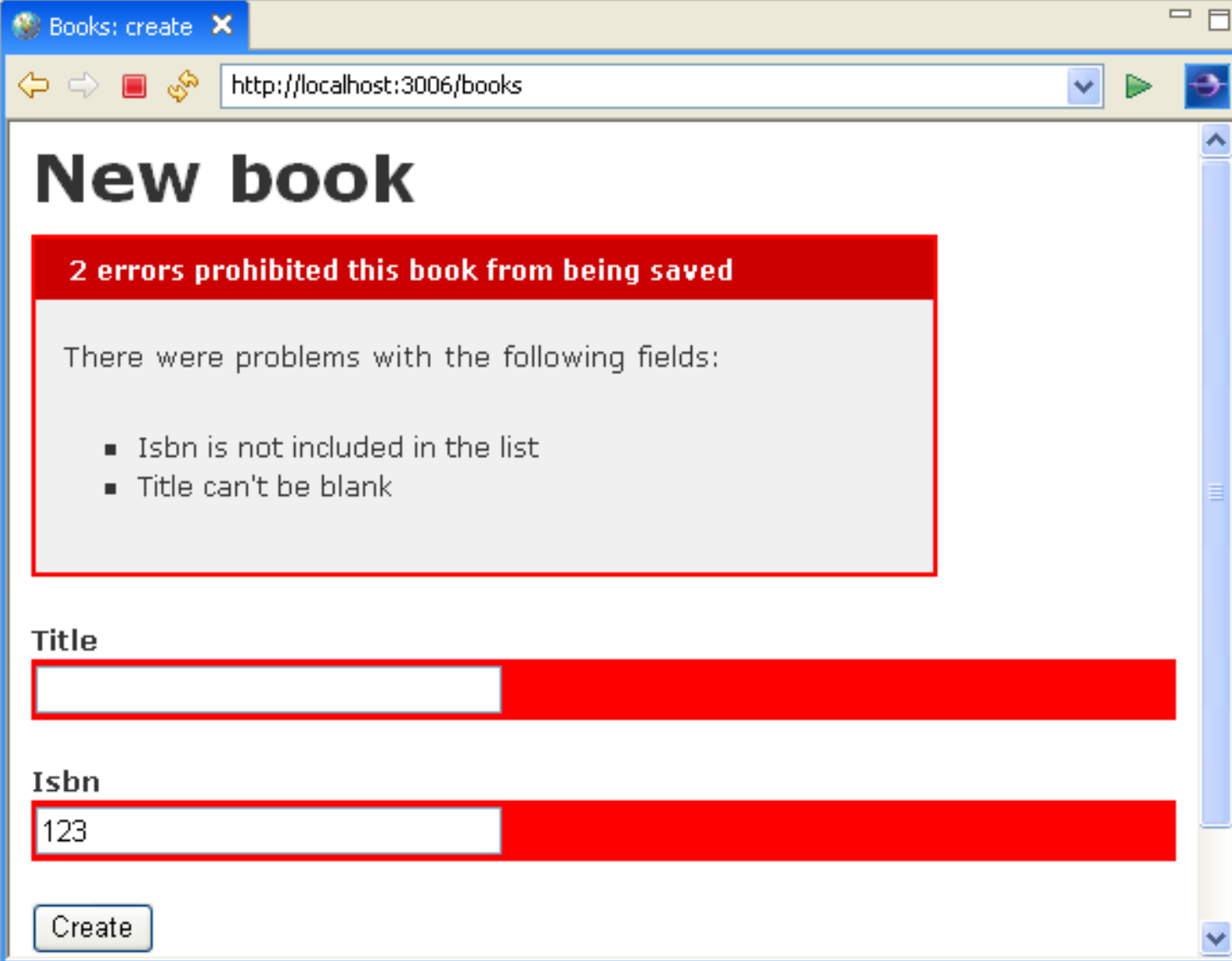
Title

Isbn

Create



...validation errors



The screenshot shows a web browser window titled "Books: create" with the address bar displaying "http://localhost:3006/books". The page content includes a heading "New book" and a red error message box that reads "2 errors prohibited this book from being saved". Below this, a grey box states "There were problems with the following fields:" followed by a bulleted list: "Isbn is not included in the list" and "Title can't be blank". The form fields for "Title" and "Isbn" are visible, with the "Title" field being empty and the "Isbn" field containing "123". A "Create" button is located at the bottom left of the form.

New book

2 errors prohibited this book from being saved

There were problems with the following fields:

- Isbn is not included in the list
- Title can't be blank

Title

Isbn

Create

The view's code takes notice of errors

Books: create x
http://localhost:3006/books

New book

2 errors prohibited this book from being saved

There were problems with the following fields:

- isbn is not included in the list
- Title can't be blank

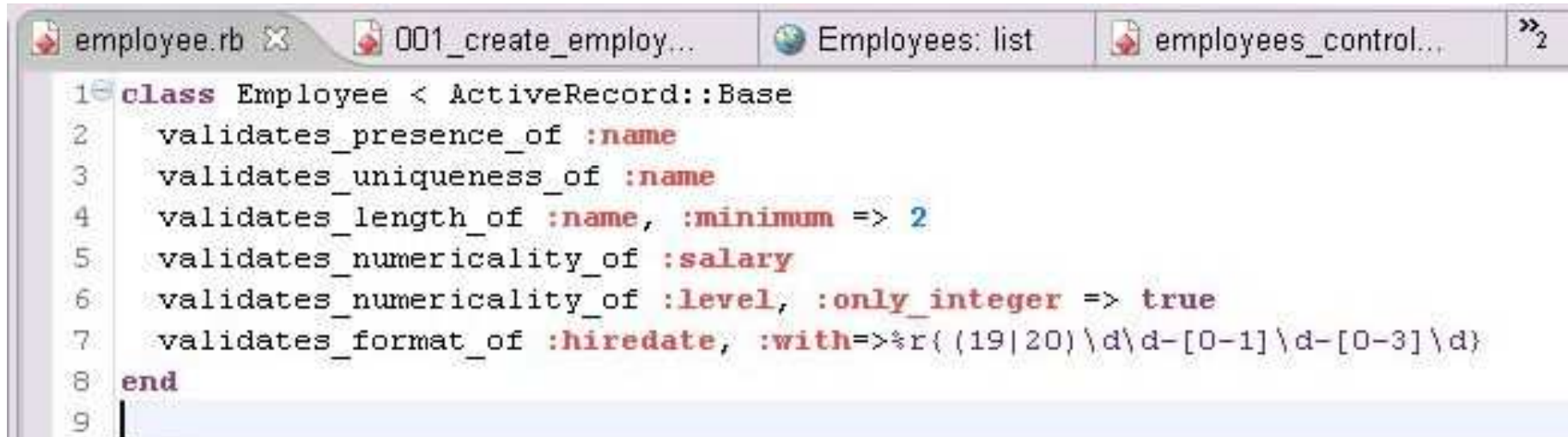
Title

isbn

Create

```
new.html.erb  
<h1>New book</h1>  
<%= error_messages_for :book %>  
  
<% form_for(@book) do |f| %>  
  <p>  
    <b>Title</b><br />  
    <%= f.text_field :title %>  
  </p>  
  
  <p>  
    <b>Isbn</b><br />  
    <%= f.text_field :isbn %>  
  </p>  
  
  <p>  
    <%= f.submit "Create" %>  
  </p>  
<% end %>  
  
<%= link_to 'Back', books_path %>
```

Rails has several `validates` methods



```
employee.rb x 001_create_employ... Employees: list employees_control... »2
1 class Employee < ActiveRecord::Base
2   validates_presence_of :name
3   validates_uniqueness_of :name
4   validates_length_of :name, :minimum => 2
5   validates_numericality_of :salary
6   validates_numericality_of :level, :only_integer => true
7   validates_format_of :hiredate, :with=>%r((19|20)\d\d-[0-1]\d-[0-3]\d)
8 end
9 |
```

Relationships between tables

The screenshot shows the Rails Generators interface. The 'Generators' tab is active, displaying a scaffold for a 'Review' model. The scaffold is defined as 'Review book_id:integer reviewer:string body:text'. Below the scaffold definition, there are radio buttons for 'Create' (selected) and 'Destroy'. To the right, there is an 'Options' section with checkboxes for 'Pretend', 'Force', 'Skip', 'Quiet', and 'Backtrace', and a 'Help' button. A 'Go' button is located to the right of the scaffold definition.

Servers Generators X Rake Tasks Rails Plugins Console Search

scaffold Review book_id:integer reviewer:string body:text Go

Create Destroy

Options

Pretend Force Skip Quiet Backtrace Help

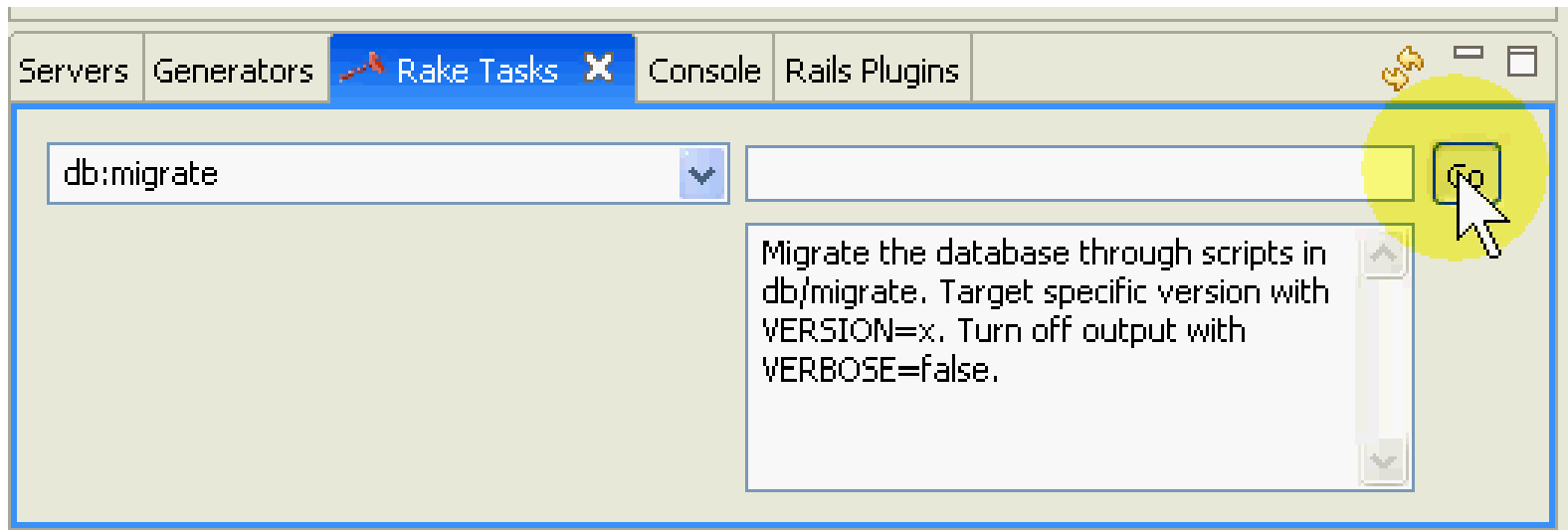
Bookstore\db\migrate\002_create_reviews.rb

```
002_create_reviews.rb X
class CreateReviews < ActiveRecord::Migration
  def self.up
    create_table :reviews do |t|
      t.integer :book_id
      t.string :reviewer
      t.text :body

      t.timestamps
    end
  end

  def self.down
    drop_table :reviews
  end
end
```

Run the most recent migration

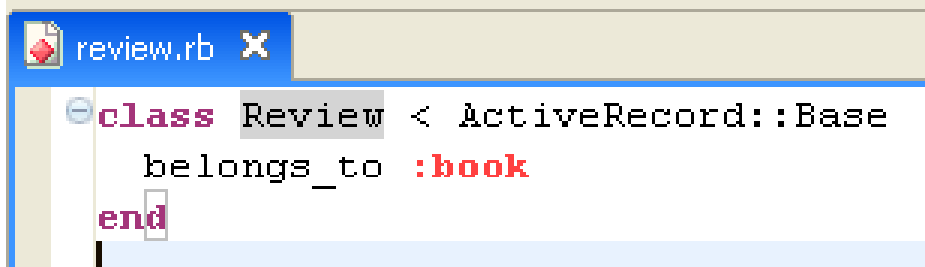


Bookstore\app\models\book.rb

```
book.rb x
class Book < ActiveRecord::Base
  validates_presence_of :title
  validates_inclusion_of :isbn, :in => 11111..9999999999999999

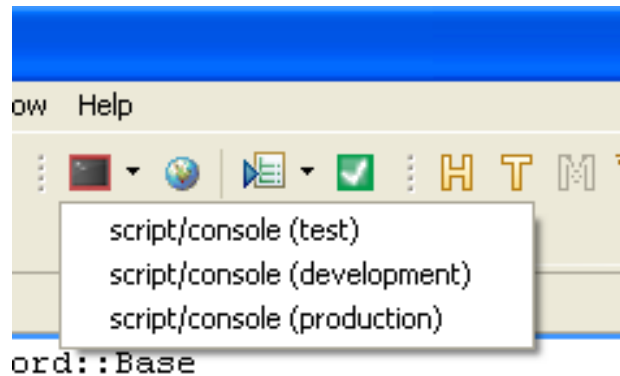
  has_many :reviews
end
```

Bookstore\app\models\review.rb

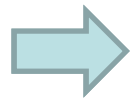
A screenshot of a code editor window titled 'review.rb'. The code defines a class 'Review' that inherits from 'ActiveRecord::Base' and has a 'belongs_to :book' association. The code is as follows:

```
class Review < ActiveRecord::Base
  belongs_to :book
end
```

The Rails Console



```
Servers Generators Rake Tasks Rails Plugins Console X Search
console development [Ruby Application] c:\ruby\bin\ruby.exe (Jan 26, 2008 1:22:37 AM)
Loading development environment (Rails 2.0.2)
>> Book.find(:first).reviews.create :reviewer => "John Q. Public", :body => "Great!"
Book.find(:first).reviews.create :reviewer => "John Q. Public", :body => "Great!"
=> #<Review id: 1, book_id: 1, reviewer: "John Q. Public", body: "Great!", created_at:
>>
```





Add code to

Bookstore\app\views\books\show.html.erb

```
show.html.erb x
<p>
  <b>Title:</b>
  <%=h @book.title %>
</p>

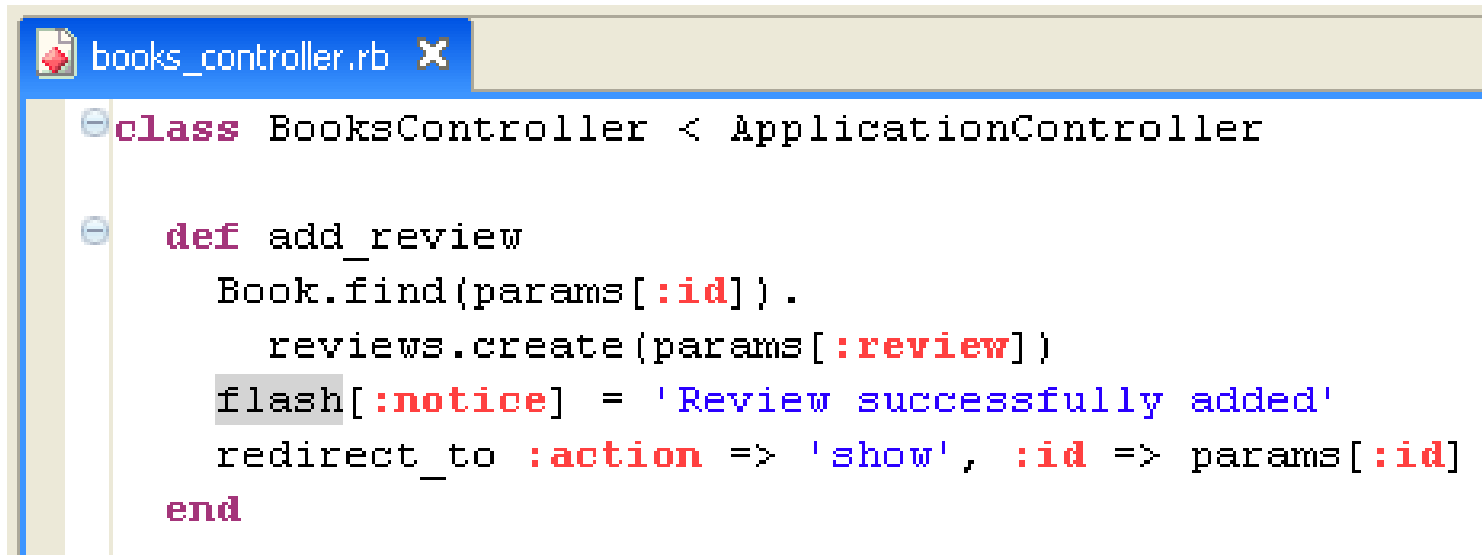
<p>
  <b>Isbn:</b>
  <%=h @book.isbn %>
</p>

<p>
  <b>Reviews:</b><br />
  <% if @book.reviews.empty? %>
    (None)
  <% else %>
    <% for review in @book.reviews %>
      <%= review.reviewer %> writes &quot;<%= review.body %>&quot;<br />
    <% end %>
  <% end %>
</p>

<% form_tag :action => "add_review", :id => @book do %>
  <b>Add a review:</b><br />
  Reviewer: <%= text_field "review", "reviewer" %><br />
  <%= text_area "review", "body", "rows" => 5 %><br />
  <%= submit_tag %>
<% end %>
```

Add a method to

Bookstore\app\controllers\books_controller.rb



```
books_controller.rb x
class BooksController < ApplicationController

  def add_review
    Book.find(params[:id]).
      reviews.create(params[:review])
    flash[:notice] = 'Review successfully added'
    redirect_to :action => 'show', :id => params[:id]
  end
end
```

