

The Rails, The Hobo and The Java Enterprise platform

Or how to become a hobo player

First, installation

Installation of the dependancies

```
sudo gem install sqlite3-ruby -v=1.2.3
sudo apt-get install sqlite3
sudo gem install sqlite3-ruby
sudo gem install rails
```

If you are in front of some discrepancies during installation of sqlite3-ruby, you can try the following additional lines just before retrying installation.

```
sudo apt-get install libsqlite3-dev
```

Installation of the framework

Simply execute the standard gem following line:

```
sudo gem install hobo
```

First Application

Now we are going to create our first Hobo application

```
hobo MyLibrary
```

focus in the just create app directory

```
cd MyLibrary
```

And now create our first datamodel : a book modelisation

```
script/generate hobo_model_resource book title:string author:string
year:integer resume:text note:integer
```

nos, we are going to create this model persistence into the default sqlite3 database

```
script/generate hobo_migration
(choose the 'm' option)
```

Ok, done. now run the rails server (WebBrick):

```
script/server
```

browser on <http://localhost:3000/>

The screenshot shows the 'My Library' application's sign-up page. At the top, there's a blue header with the title 'My Library' and links for 'Guest', 'Log in', and 'Sign up'. Below the header, there are tabs for 'Home' and 'Books', and a search bar. The main content area has a yellow background and a 'Welcome to My Library' message. A blue box contains a congratulatory message: 'Congratulations! Your Hobo Rails App is up and running' with a link to 'edit app/views/front/index.dryml'. Below this, it says 'There are no user accounts - please provide the details of the site administrator'. There are four input fields: 'Name', 'Email Address', 'Password', and 'Password Confirmation'. At the bottom of the form is a 'Signup' button and a 'or Cancel' link.

Look at the title of the application : "My Library" ! Just because we call our project "MyLibrary" (without any space), Hobo kindly titled our welcome page with a beautiful "My Library" in a good old english.

Identify yourself by creating the first user account which will act as administrator for this new application.

And now start playing with your books collection !

The screenshot shows the 'My Library' application's home page after signing up. The top header is blue with the title 'My Library' and links for 'frederic.delorme@gmail.com', 'Logged in as frederic', 'Account', and 'Log out'. Below the header, there are tabs for 'Home' and 'Books', and a search bar. A dark blue box at the top of the main content area says 'Thanks for signing up!'. Below this, there's a 'Welcome to My Library' message. A blue box contains a congratulatory message: 'Congratulations! Your Hobo Rails App is up and running' with a link to 'edit app/views/front/index.dryml'.

Yes I am impressed to discover how the Hobo dev team has been very careful with all details.

Second, the hobo

We have just created the first step of our library.

Now we have hundreds of books, maybe we can dispatch all these books in some categories ?

Ok, let's add categories to our web app "My Library"

```
script/generate hobo_model_resource category name:string sorder:integer
```

And we have to update the database with our new model:

```
script/generate hobo_migration
```

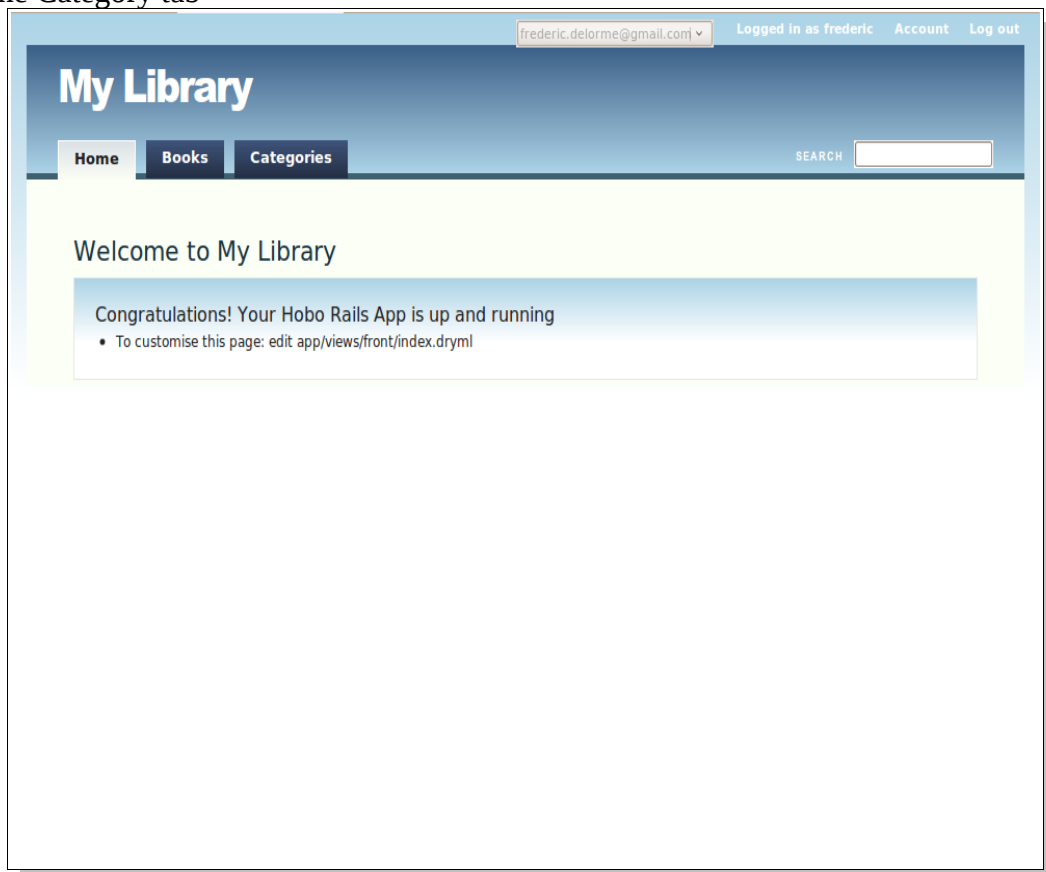
don't forget the 'm' option to run the migration right now !

ok, just for fun, run the application with a

```
script/server
```

browse to <http://localhost:3000/> and go to the "Categories" tab..

1. View the Category tab



2. and add a new "Novel" category

The screenshot shows a web application interface with a top navigation bar containing 'Home', 'Books', and 'Categories' tabs. A search bar is located on the right side of the navigation bar. The main content area is titled 'New Category' and contains two input fields: 'Name' with the value 'Novel' and 'Sorder' with the value '1'. Below the input fields are two buttons: 'Create Category' and 'Cancel'.

yes ok, nothing really new here. It's working just like Books.

Let's dye into the code

We are going to add the link between Book and Category, knwoing that a category can have many book, and one book belong to one category.

Open the file book.rb and just after fields declaration, add the following "belongs_to":

```
class Book < ActiveRecord::Base
  hobo_model # Don't put anything above this

  fields do
    title :string
    author :string
    year :integer
    resume :text
    note :integer
    timestamps
  end

  belongs_to :category
  ...
end
```

Open now the category.rb and add the small "has_many" property:

```
class Category < ActiveRecord::Base
  hobo_model # Don't put anything above this

  fields do
    name :string
    sorder :integer
    timestamps
  end

  has_many :books
  ...
end
```

we have to generate a small migration to add this field into the Book data model.

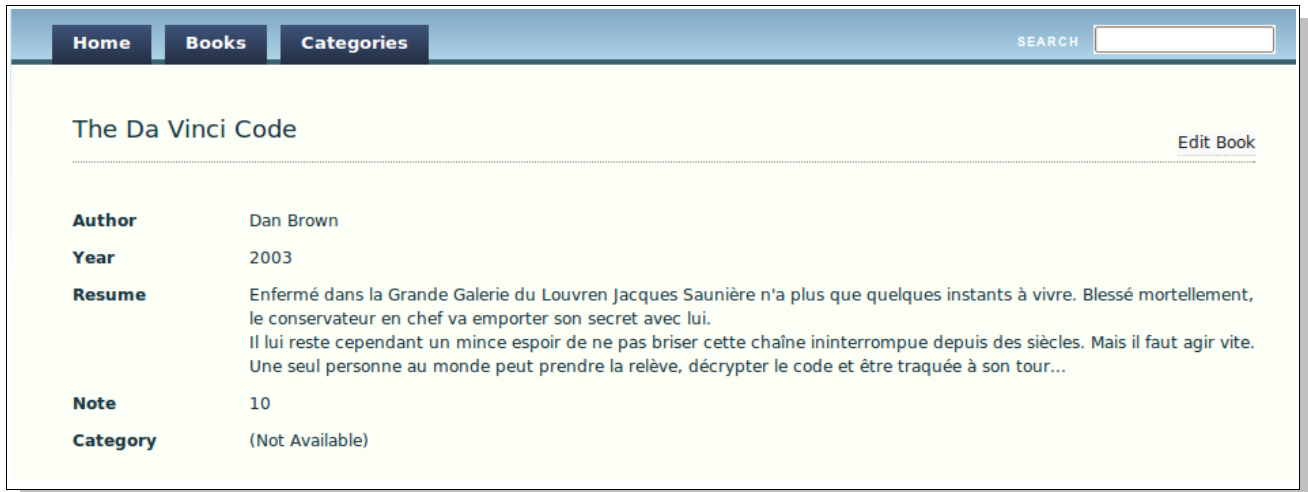
```
script/generate hobo_migration
```

choose the 'm' option to apply modifications.

and start the server to test !

script/server

let's browse the Books page

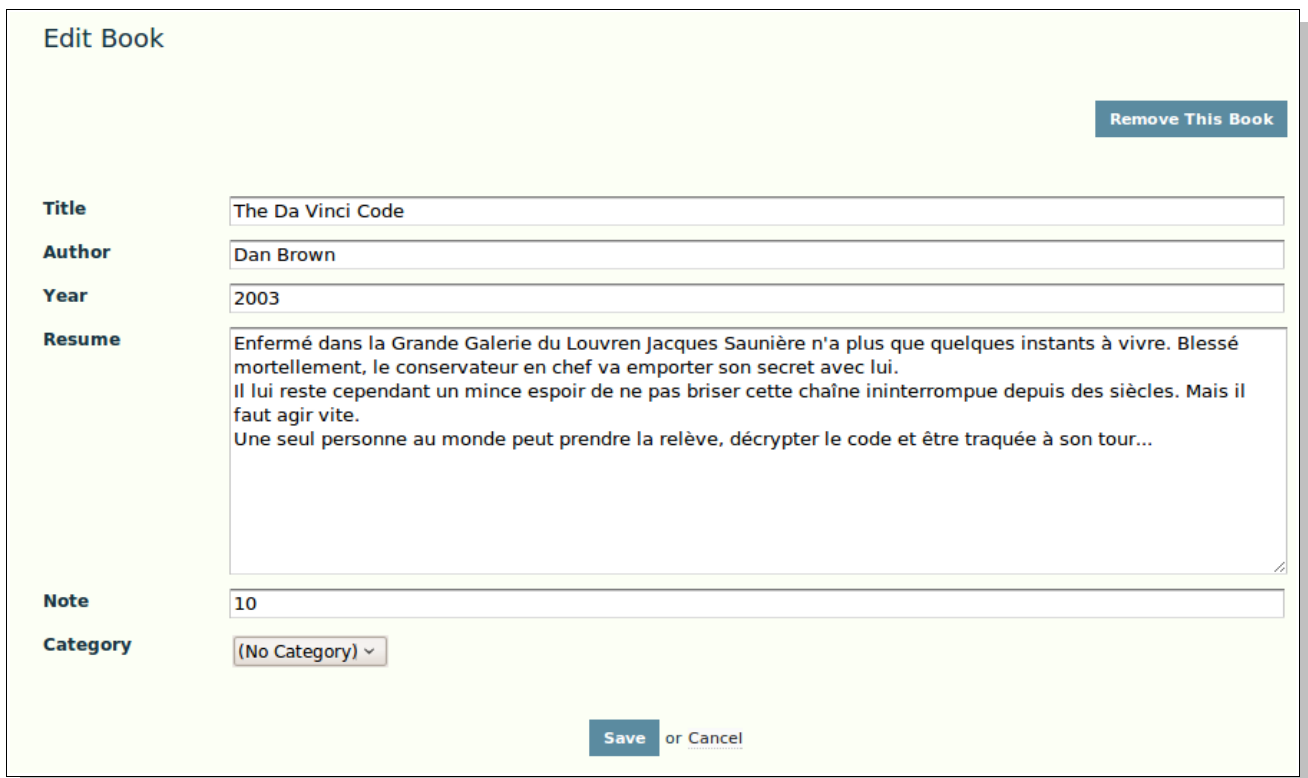


The screenshot shows a web application interface with a navigation bar at the top containing 'Home', 'Books', and 'Categories' tabs. A search bar is located on the right. The main content area displays the details for the book 'The Da Vinci Code'. On the right side of the book title, there is an 'Edit Book' link. The details are organized into a table-like structure with labels and values.

The Da Vinci Code		Edit Book
Author	Dan Brown	
Year	2003	
Resume	Enfermé dans la Grande Galerie du Louvre Jacques Saunière n'a plus que quelques instants à vivre. Blessé mortellement, le conservateur en chef va emporter son secret avec lui. Il lui reste cependant un mince espoir de ne pas briser cette chaîne ininterrompue depuis des siècles. Mais il faut agir vite. Une seule personne au monde peut prendre la relève, décrypter le code et être traquée à son tour...	
Note	10	
Category	(Not Available)	

And choose one of the created book, you can see the new "Category" field at end of page.

Click the "edit book" link and you will be invited to change the value for category which is currently set with the default "(No category)".



The screenshot shows the 'Edit Book' form. At the top right, there is a 'Remove This Book' button. The form contains several input fields for book details. The 'Category' field is a dropdown menu currently showing '(No Category)'. At the bottom, there are 'Save' and 'Cancel' buttons.

Edit Book

[Remove This Book](#)

Title: The Da Vinci Code

Author: Dan Brown

Year: 2003

Resume: Enfermé dans la Grande Galerie du Louvre Jacques Saunière n'a plus que quelques instants à vivre. Blessé mortellement, le conservateur en chef va emporter son secret avec lui.
Il lui reste cependant un mince espoir de ne pas briser cette chaîne ininterrompue depuis des siècles. Mais il faut agir vite.
Une seule personne au monde peut prendre la relève, décrypter le code et être traquée à son tour...

Note: 10

Category: (No Category) ▾

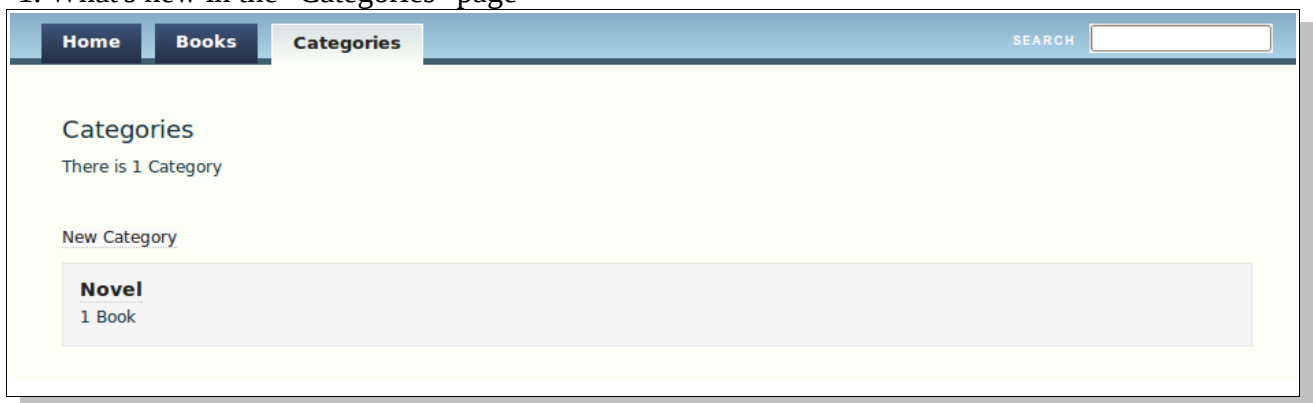
[Save](#) or [Cancel](#)

ok, back to your code editor and open the file `category_hints.rb`. and replace all the commented lines by the following single line.

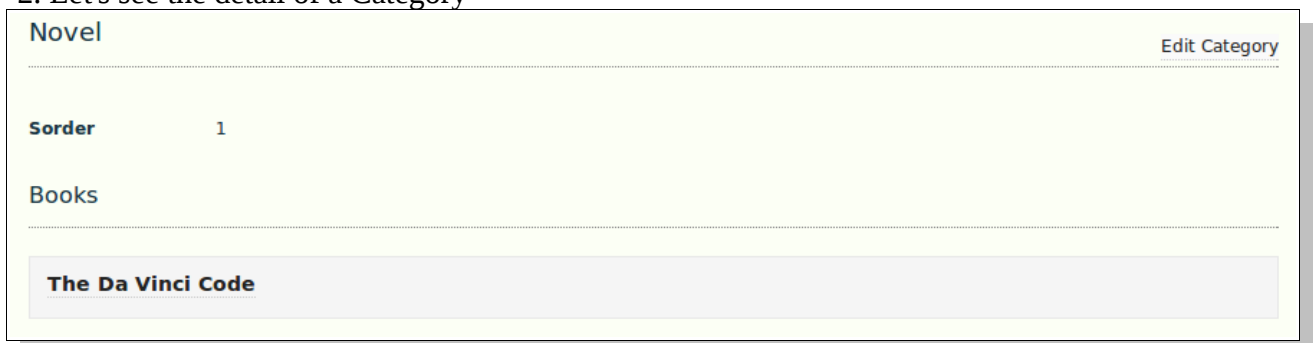
```
class CategoryHints < Hobo::ViewHints
  children :books
end
```

Telling that Category has Books has children will modify automatically the Category page, showing the number of book attached to a category.

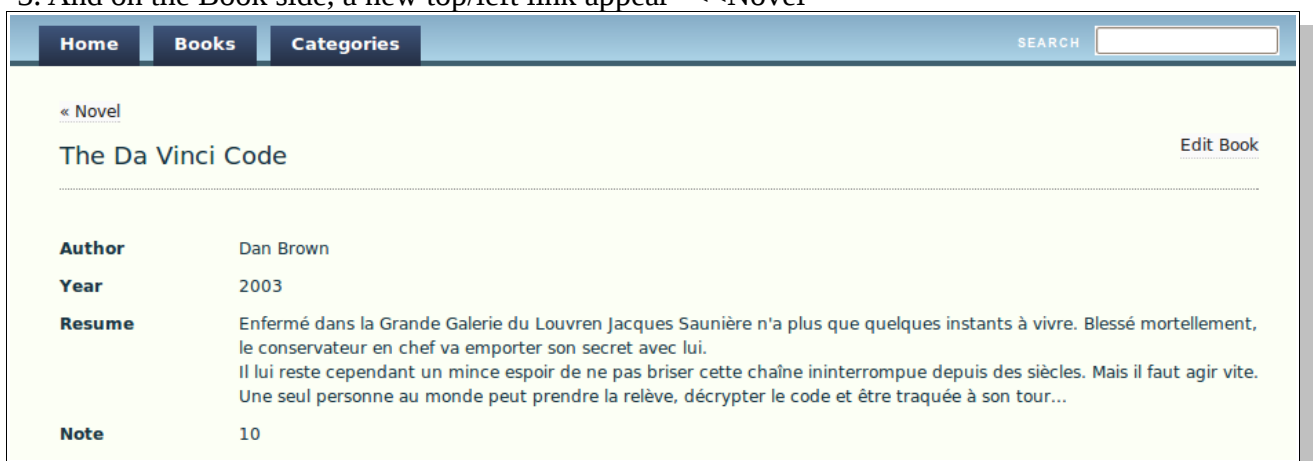
1. What's new in the "Categories" page



2. Let's see the detail of a Category



3. And on the Book side, a new top/left link appear "<<Novel"



you can see that with very small add to the generated code, you considerably inflect the default behaviour of the application and the links in the model.

Even if model link declaration and manipulation already exists in Rails, Hobo brings the corresponding mechanisms for the UI side.

Forth, Jruby to master Java

Fifth, Tomcat to rule them all