

メディア情報応用 第8週

Advanced Topics in Media Informatics Week 08

金沢工業大学 情報フロンティア学部
メディア情報学科

山岸 芳夫

Yoshio Yamagishi

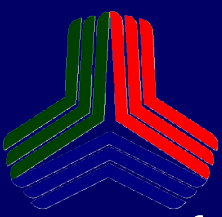
Dept. of Media Informatics

Col. of Informatics and Human Communication

Kanazawa Inst. Tech.

Web フレームワーク

Web Framework

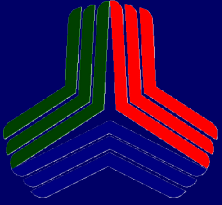
- 
- framework（枠組み）...様々なアプリケーション開発の工程で共通と思われる部分を予め用意したもの。その枠組みの中で開発を行えば少ない工程で開発できる

A collection of the common part in the processes of several application developments. The development steps can be reduced drastically when the development is based on the framework.

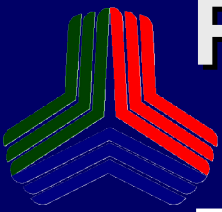
- 特に、開発環境だけではなくデータベースや実行環境まで全て揃えたものはフルスタックフレームワークと呼ばれる
The framework that includes not only development environment but also database and execution environment is called as a full stack framework.

Web フレームワーク

Web Framework



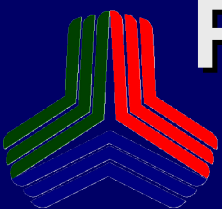
- 便利だが、まだ発展途上の技術なので流行り廃れも早い
While frameworks are very convenient, those trend has been changed rapidly because a framework is still developing technology.
- Web フレームワークの例 / Examples of web framework
 - Ruby on Rails (Ruby)
 - CakePHP(PHP)
 - Django(Python)
 - Spring(Java)



Ruby

- 日本人のまつもとゆきひろが開発した、Python同様さまざまな用途に用いることができるスクリプト系言語。真珠のPearlと同じ発音のPerlに対抗して名付けられた。文法はPythonに似ている（行末のセミコロンなし、ブロックはインデントで）

A Python-like script language for general purpose. Ruby is developed by Japanese programmer Yukihiro Matsumoto, who named "Ruby" as an analogy to a "Perl", which is pronounced same as "pearl". The syntax is like a Python, which requires no semicolons at the end of the line and represents "block" in terms of the indent instead of curly bracket.



Ruby

- **gem**というパッケージ管理システムを備えている。依存関係は**bundle**で検証できる

Ruby has a package management system named **gem**. The dependency between each package can be checked by a program **bundle**.

- Python のSimpleServer同様、Ruby も WEBrick という Webサーバを自前で用意している。もちろんApache httpd など他のWebサーバ上で動作させることもできる

Ruby has own Web server called WEBrick, which is similar to "SimpleServer" of Python. It is also available to connect Ruby to the other web server such as Apache httpd.

Rails

- デンマークのDavid Heinemeier Hanssonによって開発された、RubyによるMVCフルスタックWebアプリケーションフレームワーク

A Ruby-based MVC full-stack web application framework developed by David Heinemeier Hansson in Denmark

- RubyがWebアプリケーションの開発に用いられる場合はほぼRailsフレームワークと共に使われることが多い（だからRuby on Rails）

It is quite common that the web application development based on Ruby is using Rails framework together(that's why it is called "Ruby on Rails").

- 後発のCakePHPなどに大きな影響を与えた

Ruby on Rails greatly impacts on lately-emerged frameworks such as CakePHP.

Proxyの設定

Proxy Configuration for Ruby on Rails

- 演習環境でRuby on Railsを使って開発を行う場合、Proxyの設定が必要になる。ターミナルを開いたら以下の命令を入力しておくこと。

If you use Ruby on Rails in KIT LAN environment, it is necessary to configure the web proxy. Open terminal and the type below command:

```
export http_proxy=http://wwwproxy.kanazawa-it.ac.jp:8080
```

- 後述するrails,gem,bundleといった命令は、上記の命令を打ち込んだターミナル上で実行すること。

If you want to execute the commands such as "rails", "gem" and "bundle", you should type them in the terminal that you have input above command.



MVC

- Model-View-Controller の略。GUIアプリケーションを開発する上での考え方の一つ

An abbreviation of Model-View-Controller. One of the concept to develop GUI application.

- **Model**...データの操作、保存。データベースサーバなどのバックエンドに対応

means the storage and management of the data. it corresponds to the backend such as database server.

- **View**...実際に画面に情報を表示するフロントエンドに対応
corresponds to the frontend which displays information on the screen.

- **Controller**...モデルとビューの間を取り持つロジックの部分
corresponds to the logic part which intermediates the "Model" and the "View".



MVC

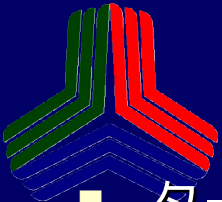
- Ruby on RailsのようなMVCフレームワークでは、これら
がなるべく独立して作成できるようにしてアプリケーション
を開発する

In the application development with MVC framework such
as Ruby on Rails, each coding process of Model, View
and Controller should be independent from each other.

Ruby on Railsの命令

Commands of Ruby on Rails

- **rails**...基本的な操作を行う。モデル、ビュー、コントローラーの作成、WEBrickサーバの起動など
Basic command of Ruby on Rails. It works to generate Model, View and Controller, start WEBrick server and so on.
- よく使われる rails 命令/ Common rails commands
 - **rails new** ...新規アプリケーション作成/create new application.
 - **rails g (generate)** ... モデルやコントローラー、スキャフォールドなどの作成/generate Model, Controller, Scaffold or something like that.
 - **rails s (server)**...WEBrickサーバの起動/start WEBrick server.
 - **rails destroy**...モデルやコントローラー、スキャフォールドなどの削除/remove Model, Controller , Scaffold or something like that.
- **rake**...主にデータベース操作やテストに用いられる
manages mainly database operation and test.



Hello World

- ターミナルを開いてコマンドラインから以下のように入力する
Open terminal, then type in below command.

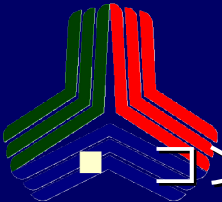
```
rails new helloworld
```

- helloworld/というディレクトリが作成され、さらにその中に様々なファイル、ディレクトリが作成されているのがわかる。このディレクトリの中に入り、helloworldというコントローラーと、index というメソッドを作成する。

The "helloworld/" directory is created. You can see several files and directories in the directory. Enter the "helloworld/" directory, then generate "helloworld" controller and "index" method.

```
cd helloworld
```

```
rails g controller helloworld index
```



Hello World

- コントローラーを作成すれば自動的にビューも作成される。今回は表示だけなのでモデルは作成しない

The "View" is automatically generated when the "Controller" is generated. We do not generate "Model" because this time we only need to display strings on the screen.

- Webサーバを起動する/Start web server.

`rails s`

- <http://localhost:3000/helloworld/index> にアクセスしてみよう。ページが表示されるはず。この内容は `app/views/helloworld/index.html.erb` に書かれているので、それを書き換えれば内容を変更できる

Try to access <http://localhost:3000/helloworld/index> and make sure it works. The HTML is described in the file `app/views/helloworld/index.html.erb` so the contents can be changed if the file is modified.

Ruby on Rails チャット

Ruby on Rails Chat

- 予めweek08.tar.gz をMoodleからダウンロードしておくこと。ホームディレクトリに戻り、また新規でアプリケーションronrchatを作成する。

First of all, download week08.tar.gz from 8th topic of Moodle course. Go back to the home directory, then create new application named "ronrchat".

```
cd
```

```
rails new ronrchat
```

Ruby on Rails チャット

Ruby on Rails Chat

- 今回はモデルを作成する。MySQLも使えるが、今回はデフォルトのデータベースである sqlite を用いる事にする。ここではモデル名を chat とする

This time we generate "Model" named "chat". Although MySQL is available, we take default database of Ruby on Rails "sqlite" .

```
cd ronrchat
```

```
rails g model chat name:string mesg:text
```


- この段階ではまだデータベースが作成されていないため、rakeを使って作成する。

At this time the database does not exist. So create it using rake.

```
rake db:migrate
```

Ruby on Rails チャット


Ruby on Rails Chat

- 
- 作成後は **rails db** で sqlite コマンドラインを開いてデータベースの操作が出来る。今回テーブル名は chats となる。**.schema chats** でテーブル構造を見ると、モデル作成時に指定した name, mesg というカラムに加え、id と created_at、updated_at というタイムスタンプが自動的に加わっているのがわかる。sqlite のコマンドラインの他の命令は **.help** で見る事ができる

The database management can be done by sqlite command line, which is opened by "**rails db**" command. "**.schema chat**" command shows the table structure of the database. You can see the columns "name" and "mesg", and also some other columns such as "id", "created_at" and "updated_at", which are added automatically. The list of the commands of sqlite can be shown with **.help** command.

Ruby on Rails チャット

Ruby on Rails Chat

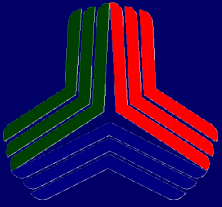
- 
- 続いてコントローラーを作成する。今回も表示と入力と同じ一つの画面なので index メソッドだけを使う。モデルを使う場合**コントローラー名は複数形にする**のがRailsの暗黙のルール

Generate "Controller" as below. This time we take "index" method only. The **name of the "Controller" must be plural** due to the Rails convention whenever it is based on the "Model".

```
rails g controller chats index
```


Ruby on Rails チャット

Ruby on Rails Chat



- week08.tar.gz を展開し、chats_controller.rb を app/controllers/、index.html.erb を app/views/chats/、application.rb と routes.rb を config/ に上書きコピーして、web サーバを起動しよう。

Extract week08.tar.gz, then copy chats_controller.rb to app/controllers/, index.html.erb to app/views/chats/ and application.rb and routes.rb to config/, respectively. After all, start web server.

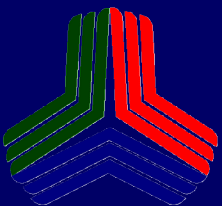
rails s

- <http://localhost:3000> にアクセスして、これまで作ってきたチャットとほぼ同仕様のアプリケーションが動作していることを確かめよう。

Access <http://localhost:3000> and make sure that the chat application, which looks like the one we have built so far, works.

スキヤフォールド

Scaffold



- scaffold (土台) ... CRUD (Create, Read, Update, Deleteといった、一般的なWebアプリケーションの4大機能)システムを、**ほとんどコーディングすることなしに**、いち早く作成できるRailsの機能

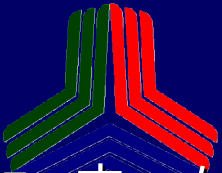
realizes **almost coding-less development** of CRUD(four typical functions of general web application: Create, Read, Update and Delete) system.

- モデル、コントローラー、ビューの作成が自動的に行われる。もちろん作成後に修正することも可能

The "Model", "Controller" and "View" are generated automatically with "scaffolding". It is also available to modify MVC after automatic generation of them.

スキヤフォールドの作成

Scaffold Generation



- ホームディレクトリに戻り新たにアプリケーションおよびスキヤフォールドを作成する。

Go back to the home directory, then create new application and its scaffold.

```
cd
rails new scaffold_chat
cd scaffold_chat
rails g scaffold chat name:string mesg:text
```

- データベースを作成し、サーバを起動してから<http://localhost:3000/chats> で動作を確認しよう。

Generate a database, then start web server and access <http://localhost:3000/chats> to check the operation.

```
rake db:migrate
rails s
```

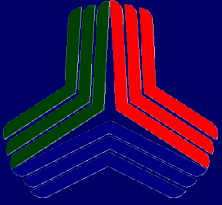


twitter Bootstrap

- twitter社が開発しオープンソースで公開しているWebフロントエンドフレームワーク。HTML,CSS,Javascriptで構成されており、レスポンシブデザイン（端末の画面の大きさに合わせて動的にデザインが変わり、モバイル/PCで別々にコンテンツを作成する必要がない）に対応している

An open-source web frontend framework, which is originally developed by twitter. It consists of a set of HTML, CSS and Javascript, and enables a responsive design (the design is dynamically changed corresponding to the browsing device. There is no need to build contents for PC and mobile device separately)

twitter Bootstrap



- MVCフレームワークでビューが完全にコントローラー、モデルといったロジックと切り離されている場合、簡単に導入が可能

Bootstrap can be easily introduced into MVC framework if the "View" is completely separated from the logic part such as "Controller" and "Model" .

- Ruby on Railsのスキヤフォールドにも非常に簡単に導入できる

It is very easy to introduce Bootstrap into the "Scaffold" of Ruby on Rails.

Bootstrap の導入

Introducing Bootstrap

- scaffold_chatにBootstrapを導入してみよう。Gemfileを開き、一番下の end の上に以下の3行を追加する。行頭にタブが一つ入っていることに注意

Introduce Bootstrap into scaffold_chat. Open "Gemfile" and append following three lines above the "end" at the bottom of the file. Note that a tab is inserted at the start of each line.

```
gem 'therubyracer'  
gem 'less-rails'  
gem 'twitter-bootstrap-rails'
```

- これらのgemを依存関係も含めてインストールする
Install these gems preserving dependencies.

```
bundle install
```

Bootstrap の導入

Introducing Bootstrap

- 続いて現在の scaffold_chat にBootstrapを反映させる

Install Bootstrap on scaffold_chat.

```
rails g bootstrap:install
```

- これだけでも若干画面が変わったが、さらに画面をBootstrapらしくするためにテーマを導入する。画面が洗練されたデザインになる。なお、上書きしていいか、という質問には全てyで答えて良い

While the "View" is slightly changed by these processes only, we introduce a "theme" to make the "View" more Bootstrap-oriented. After below treatment, the design of the "View" becomes drastically sophisticated. Note: it is OK to answer "y" for all of the questions, which has been asked during below process.

```
rails g bootstrap:themed Chats
```