RUBY ON RAILS - FRAMEWORK

http://www.tutorialspoint.com/ruby-on-rails/rails-framework.htm

A framework is a program, set of programs, and/or code library that writes most of your application for you. When you use a framework, your job is to write the parts of the application that make it do the specific things you want.

When you set out to write a Rails application, leaving aside the configuration and other housekeeping chores, you have to perform three primary tasks −

- **Describe and model your application's domain** − The domain is the universe of your application. The domain may be a music store, a university, a dating service, an address book, or a hardware inventory. So here you have to figure out what's in it, what entities exist in this universe and how the items in it relate to each other. This is equivalent to modeling a database structure to keep the entities and their relationship.

- **Specify what can happen in this domain** − The domain model is static; you have to make it dynamic. Addresses can be added to an address book. Musical scores can be purchased from music stores. Users can log in to a dating service. Students can register for classes at a university. You need to identify all the possible scenarios or actions that the elements of your domain can participate in.

- **Choose and design the publicly available views of the domain** − At this point, you can start thinking in Web-browser terms. Once you've decided that your domain has students, and that they can register for classes, you can envision a welcome page, a registration page, and a confirmation page, etc. Each of these pages, or views, shows the user how things stand at a certain point.

Based on the above three tasks, Ruby on Rails deals with a Model/View/Controller MVC framework.

Ruby on Rails MVC Framework

The **M**odel **V**iew **C**ontroller principle divides the work of an application into three separate but closely cooperative subsystems.

Model $ActiveRecord$

It maintains the relationship between the objects and the database and handles validation, association, transactions, and more.

This subsystem is implemented in ActiveRecord library, which provides an interface and binding between the tables in a relational database and the Ruby program code that manipulates database records. Ruby method names are automatically generated from the field names of database tables.

View $ActionView$

It is a presentation of data in a particular format, triggered by a controller's decision to present the data. They are script-based template systems like JSP, ASP, PHP, and very easy to integrate with AJAX technology.

This subsystem is implemented in ActionView library, which is an Embedded Ruby ERb based system for defining presentation templates for data presentation. Every Web connection to a Rails application results in the displaying of a view.

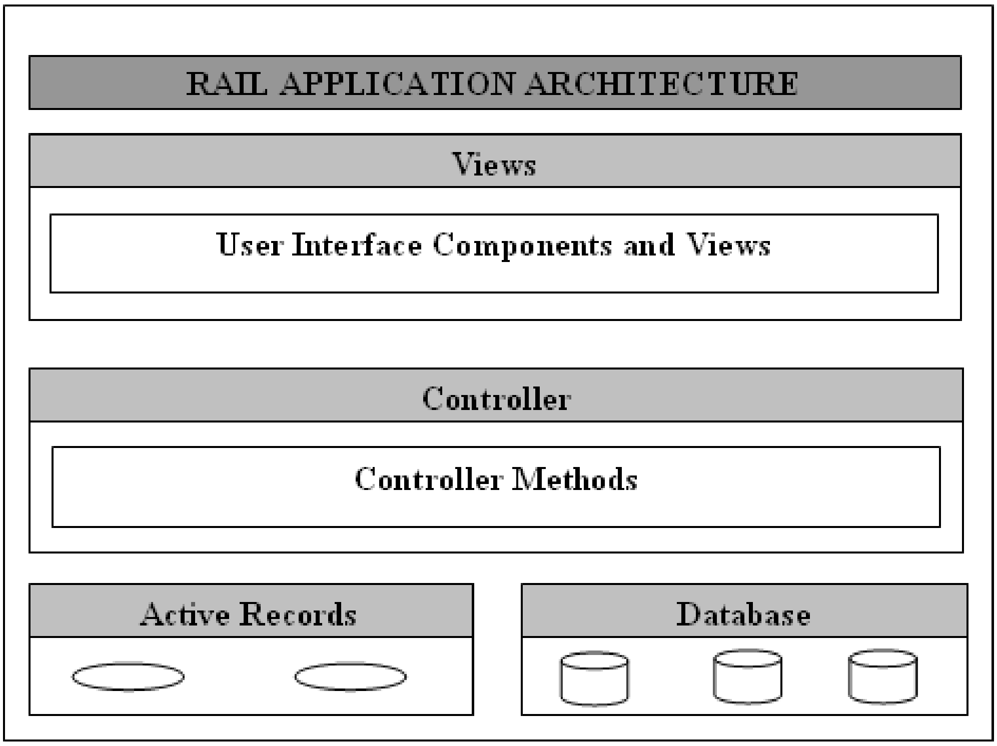
Controller $ActionController$

The facility within the application that directs traffic, on the one hand, querying the models for specific data, and on the other hand, organizing that data *searching*, *sorting*, *messagingit* into a form that fits the needs of a given view.

This subsystem is implemented in ActionController, which is a data broker sitting between ActiveRecord *thedatabaseinterface* and ActionView *thepresentationengine*.

Pictorial Representation of MVC Framework

Given below is a pictorial representation of Ruby on Rails Framework −


RAIL APPLICATION ARCHITECTURE
Views
User Interface Components and Views
Controller
Controller Methods
Active Records
Database

Directory Representation of MVC Framework

Assuming a standard, default installation over Linux, you can find them like this −

```
tp> cd /usr/local/lib/ruby/gems/2.2.0/gems
tp> ls
```

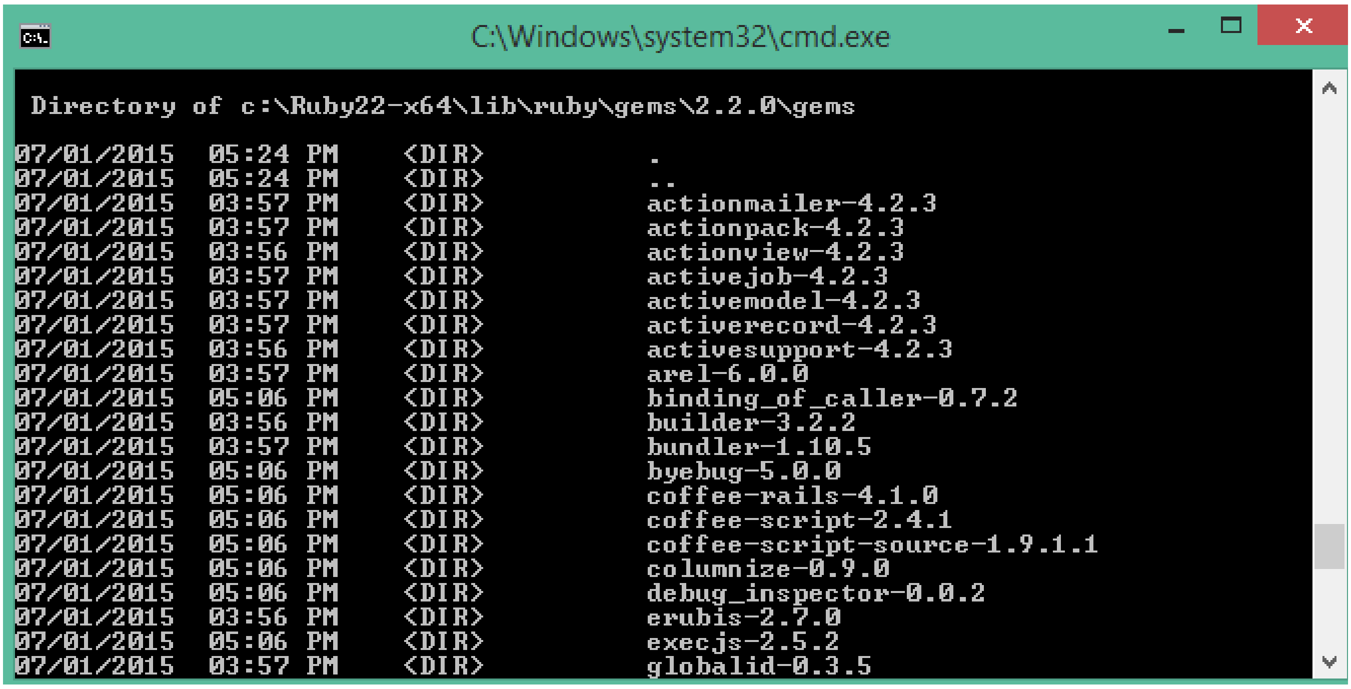
You will see subdirectories including *butnotlimitedto* the following −

- actionpack-x.y.z
- activerecord-x.y.z
- rails-x.y.z

Over a windows installation, you can find them like this −

```
tp>cd ruby\lib\ruby\gems\2.2.0\gems
ruby\lib\ruby\gems\2.2.0\gems\>dir
```

You will see subdirectories including *butnotlimitedto* the following −



```
 Directory of c:\Ruby22-x64\lib\ruby\gems\2.2.0\gems

07/01/2015  05:24 PM    <DIR>          .
07/01/2015  05:24 PM    <DIR>          ..
07/01/2015  03:57 PM    <DIR>          actionmailer-4.2.3
07/01/2015  03:57 PM    <DIR>          actionpack-4.2.3
07/01/2015  03:56 PM    <DIR>          actionview-4.2.3
07/01/2015  03:57 PM    <DIR>          activejob-4.2.3
07/01/2015  03:57 PM    <DIR>          activemodel-4.2.3
07/01/2015  03:57 PM    <DIR>          activerecord-4.2.3
07/01/2015  03:56 PM    <DIR>          activesupport-4.2.3
07/01/2015  03:57 PM    <DIR>          arel-6.0.0
07/01/2015  05:06 PM    <DIR>          binding_of_caller-0.7.2
07/01/2015  03:56 PM    <DIR>          builder-3.2.2
07/01/2015  03:57 PM    <DIR>          bundler-1.10.5
07/01/2015  05:06 PM    <DIR>          byebug-5.0.0
07/01/2015  05:06 PM    <DIR>          coffee-rails-4.1.0
07/01/2015  05:06 PM    <DIR>          coffee-script-2.4.1
07/01/2015  05:06 PM    <DIR>          coffee-script-source-1.9.1.1
07/01/2015  05:06 PM    <DIR>          columnize-0.9.0
07/01/2015  05:06 PM    <DIR>          debug_inspector-0.0.2
07/01/2015  03:56 PM    <DIR>          erubis-2.7.0
07/01/2015  05:06 PM    <DIR>          execjs-2.5.2
07/01/2015  03:57 PM    <DIR>          globalid-0.3.5
```

ActionView and ActionController are bundled together under ActionPack.

ActiveRecord provides a range of programming techniques and shortcuts for manipulating data from an SQL database. ActionController and ActionView provides facilities for manipulating and displaying that data. Rails ties it all together.