Ruby on Rails Interview Questions and Answers PDF download: Here are the top interview questions of Ruby on Rails. Be prepared for the interview. You can download the entire questions in a pdf format. so guys All the best for your interview.

**1) Explain what is Ruby on Rails?**

- **Ruby:** It is an object oriented programming language inspired by PERL and PYTHON.
- **Rails:** It is a framework used for building web application

**2) Explain what is class libraries in Ruby?**

Class libraries in Ruby consist of a variety of domains, such as data types, thread programming, various domains, etc.

**3) Mention what is the naming convention in Rails?**

- **Variables:** For declaring Variables, all letters are lowercase, and words are separated by underscores
- **Class and Module:** Modules and Classes uses MixedCase and have no underscore; each word starts with a uppercase letter
- **Database Table:** The database table name should have lowercase letters and underscore between words, and all table names should be in the plural form for example invoice_items
- **Model:** It is represented by unbroken MixedCase and always have singular with the table name
- **Controller:** Controller class names are represented in plural form, such that OrdersController would be the controller for the order table.

**4) Explain what is "Yield" in Ruby on Rails?**

A Ruby method that receives a code block invokes it by calling it with the "Yield".

**5) Explain what is ORM (Object-Relationship-Model) in Rails?**

ORM or Object Relationship Model in Rails indicate that your classes are mapped to the table in the database, and objects are directly mapped to the rows in the table.

**Question 6. What Is Orm In Rails?**

ORM tends for Object-Relationship-Model, where Classes are mapped to table in the database, and Objects are directly mapped to the rows in the table.

**Question 7. How Many Types Of Associations Relationships Does A Model Have?**

**Answer :**When you have more than one model in your rails application, you would need to create connection between those models. You can do this via associations. Active Record supports three types of associations:one-to-one: A one-to-one relationship exists when one item has exactly one of another item. For example, a person has exactly one birthday or a dog has exactly one owner.one-to-many: A one-to-many relationship exists when a single object can be a member of many other objects. For instance, one subject can have many books.many-to-many: A many-to-many relationship exists when the first object is related to one or more of a second object, and the second object is related to one or many of the first object.You indicate these associations by adding declarations to your models: has_one, has_many, belongs_to, and has_and_belongs_to_many.

**Question 8. What Are Helpers And How To Use Helpers In Ror?**

**Answer :**Helpers are modules that provide methods which are automatically usable in your view. They provide shortcuts to commonly used display code and a way for you to keep the programming out of your views. The purpose of a helper is to simplify the view.

**Question 9. What Are Filters?**

**Answer :**Filters are methods that run “before”, “after” or “around” a controller action. Filters are inherited, so if you set a filter on ApplicationController, it will be run on every controller in your application.

**Question 10. What Is Mvc? And How It Works?**

MVC tends for Model-View-Controller, used by many languages like PHP, Perl, Python etc. The flow goes like this:

Request first comes to the controller, controller finds and appropriate view and interacts with model, model interacts with your database and send the response to controller then controller based on the response give the output parameter to view.

**11) Explain what is Rails Active Record in Ruby on Rails?**

Ruby on Rails Interview Questions and Answers PDF download: Rails active record is the Object/Relational Mapping (ORM) layer supplied with Rails. It follows the standard ORM model as

- Table map to classes
- Rows map to objects
- Columns map to object attributes

**12) Explain how Rails implements Ajax?**

Ajax powered web page retrieves the web page from the server which is new or changed unlike other web-page where you have to refresh the page to get the latest information.

Rails triggers an Ajax Operation in following ways

- **Some trigger fires:** The trigger could be a user clicking on a link or button, the users inducing changes to the data in the field or on a form
- **Web client calls the server:** A Java-script method, XMLHttpRequest, sends data linked with the trigger to an action handler on the server. The data might be the ID of a checkbox, the whole form or the text in the entry field
- **Server does process:** The server side action handler does something with the data and retrieves an HTML fragment to the web client
- **Client receives the response:** The client side JavaScript, which Rails generates automatically, receives the HTML fragment and uses it to update a particular part of the current

**13) Mention how you can create a controller for subject?**

To create a controller for subject you can use the following command

```
C:\ruby\library> ruby script/generate controller subject
```

**14) Mention what is Rails Migration?**

Rails Migration enables Ruby to make changes to the database schema, making it possible to use a version control system to leave things synchronized with the actual code.

**15) List out what can Rails Migration do?**

Rails Migration can do following things

- Create table
- Drop table
- Rename table
- Add column
- Rename column
- Change column
- Remove column and so on

**16) Mention what is the command to create a migration?**

To create migration command includes

C:\ruby\application>ruby script/generate migration table_name

**17) Explain when self.up and self.down method is used?**

When migrating to a new version, **self.up** method is used while **self.down** method is used to roll back my changes if needed.

**18) Mention what is the role of Rails Controller?**

Ruby on Rails Interview Questions and Answers PDF download: The Rails controller is the logical center of the application. It faciliates the interaction between the users, views, and the model. It also performs other activities like

- It is capable of routing external requests to internal actions. It handles URL extremely well
- It regulates helper modules, which extend the capabilities of the view templates without bulking of their code
- It regulates sessions; that gives users the impression of an ongoing interaction with our applications

**19) Mention what is the difference between Active support's "HashWithIndifferent" and Ruby's "Hash" ?**

The **Hash** class in Ruby's core library returns value by using a standard **"= ="** comparison on the keys. It means that the value stored for a **symbol** key cannot be retrieved using the equivalent string. While the **HashWithIndifferentAccess**treats Symbol keys and String keys as equivalent.

**20) Explain what is Cross-Site Request Forgery (CSRF) and how Rails is protected against it?**

CSRF is a form of attack where hacker submits a page request on your behalf to a different website, causing damage or revealing your sensitive data. To protect from CSRF attacks, you have to add **"protect_from_forgery"** to your **ApplicationController**. This will cause Rails to require a CSRF token to process the request. CSRF token is given as a hidden field in every form created using Rails form builders.

**21) Explain what is Mixin in Rails?**

Mixin in Ruby offers an alternative to multiple inheritances, using mixin modules can be imported inside other class.

**22) Explain how you define Instance Variable, Global Variable and Class Variable in Ruby?**

- Ruby Instance variable begins with — **@**
- Ruby Class variables begin with — **@@**
- Ruby Global variables begin with — **$**

**23) Explain how you can run Rails application without creating databases?**

You can execute your application by uncommenting the line in environment.rb

path=> rootpath conf/environment.rb

config.frameworks = [ action_web_service, :action_mailer, :active_record]

**24) Mention what is the difference between the Observers and Callbacks in Ruby on Rails?**

- **Rails Observers:** Observers is same as Callback, but it is used when method is not directly associated to object lifecycle. Also, the observer lives longer, and it can be detached or attached at any time. For example, displaying values from a model in the UI and updating model from user input.
- **Rails Callback:** Callbacks are methods, which can be called at certain moments of an object's life cycle for example it can be called when an object is validated, created, updated, deleted, A call back is short lived. For example, running a thread and giving a call-back that is called when thread terminates

**25) Explain what is rake in Rails?**

Ruby on Rails Interview Questions and Answers PDF download: Rake is a Ruby Make; it is a Ruby utility that substitutes the Unix utility 'make', and uses a 'Rakefile' and '.rake files' to build up a list of tasks. In Rails, Rake is used for normal administration tasks like migrating the database through scripts, loading a schema into the database, etc.

**Question 26. How Many Types Of Relationships Does A Model Has?**

has_one

belongs_to

has_many

has_many :through

**Question 27. Difference Between Render And Redirect?**

**Answer :**render example: render :action, render :partial etc. redirect example: redirect_to :controller => 'users', :action => 'new'

**Question 28. What Is Active Record?**

**Answer :**Active Record are like Object Relational Mapping(ORM), where classes are mapped to table and objects are mapped to columns in the table.

**Question 29. Ruby Supports Single Inheritance/multiple Inheritance Or Both?**

**Answer :**Ruby Supports only Single Inheritance

**Question 30. What Is Bundler?**

**Answer :**Bundler is a new concept introduced in Rails3, which helps to you manage your gems for the application. After specifying gems in your Gemfile, you need to do a bundle install. If the gem is available in the system, bundle will use that else it will pick up.

**Question 31. What Is The Newest Approach For Find(:all) In Rails 3?**

**Answer :**Model.where(:activated => true)

**Question 32. What Is The Role Of Mvc Architecture In Ruby On Rails?**

**Answer :**Ruby on Rails Interview Questions and Answers PDF download: MVC (Model-View-Controller) is the architecture that provides flexibility and scalability of the applications.It is almost having the same concept in any other language like PHP, Perl or Python. It is one of the major used architecture involved today due to its simplicity.Controller is the main part in this kind of architecture where it handles the request that is coming from another controller.Controller contacts the view and passes on the request to the view and it also interacts with the model to define the type of request.Model is responsible for interacting with the database and provides the responses to the controller.
Controller takes the response and gives the response in the output form to the user that has made the request.

## You might also like: *Top 20 MongoDB Interview Questions and Answers*

**Question 33. What Are The Components Defined In The Model From Mvc Architecture?**

**Answer :**The components involved in defining the model are as follows:**Validations:** this is one of the very essential components and it defines the validations that are being put up on the input type of stream like validate_presence_of, format_of, etc.**Relationship:** this is another type of component that describe the relationship between different types of components and it shows the relationship in the form of has_one, has_many, etc.**Callbacks:** this is essential when it comes to respond after the failure and it allows the application to have certain functionality during failure. This can be given as before_save, after_save, etc.**Validation group settings:** allow users to define the installed plugin settings.**Active record association relationship:** allows current records to be actively having the relationship between one another.

**Question 34. Choosing Between Belongs_to And Has_one?**

**Answer :**If you want to set up a one-to-one relationship between two models, you'll need to add belongs_to to one, and has_one to the other. How do you know which is which?The distinction is in where you place the foreign key (it goes on the table for the class declaring the belongs_to association), but you should give some thought to the actual meaning of the data as well. The has_one relationship says that one of something is yours - that is, that something points back to you. For example, it makes more sense to say that a supplier owns an account than that an account owns a supplier.

**Question 35. Render Vs. Redirect_to In Ruby On Rails ?**

**Answer :**render will render a particular view using the instance variables available in the action.For example if a render was used for the new action, when a user goes to /new, the new action in the controller is called, instance variables are created and then passed to the new view. Rails creates the html for that view and returns it back to the user's browser. This is what you would consider a normal page load.redirect_to will send a redirect to the user’s browser telling it to re-request a new URL as 302 redirect response. Then the browser will send a new request to that URL and it will go through the action for that URL, oblivious to the fact that it was redirected to. None of the variables created in the action that caused the redirect will be available to the redirected view. This is what happens when you click on ‘Create’ in a form and the object is created and you’re redirected to the edit view for that object.

**Question 36. What Is The Difference Between Save And Save?**

**Answer :**Save! performs all validations and callbacks. If any validation returns false, save! throws an error and canceles the save.Save does not throw any error in the case above, but canceles the save. Also, the validators can be bypassed.

**Question 37. What Is The Difference Between Delete And Destroy ?**

**Answer :**The delete method essentially deletes a row (or an array of rows) from the database. Destroy on the other hand allows for a few more options. First, it will check any callbacks such as before_delete, or any dependencies that we specify in our model. Next, it will keep the object that just got deleted in memory; this allows us to leave a message saying something like “ 'Object_name' has been deleted.” Lastly, and most importantly, it will also delete any child objects associated with that object!

**Question 38. What Are Filters? And How Many Types Of Filters Are There In Ruby ?**

**Answer :**Filters are methods that are run before, after or "around" a controller action.Filters are inherited, so if you set a filter on ApplicationController, it will be run on every controller in your application.Filter can take one of three forms: method reference (symbol), external class, or inline method (proc).after_filter append_after_filter append_around_filter append_before_filter around_filter before_filter filter_chain prepend_after_filter prepend_around_filter prepend_before_filter skip_after_filter skip_before_filter skip_filter

**Question 39. What Is The Flash?**

**Answer :**The flash is a special part of the session which is cleared with each request. This means that values stored there will only be available in the next request, which is useful for storing error messages etc. It is accessed in much the same way as the session, like a hash.flash is a object of Actiondispatch class.

**Question 40. Difference Between Argument And Parameter In Ruby On Rails ?**

**Answer :**A parameter represents a value that the method expects you to pass when you call it.An argument represents the value you pass to a method parameter when you call the method. The calling code supplies the arguments when it calls the method.In simple words, parameters appear in method definitions; arguments appear in method calls.For example, in below method, variables param1 and param2 are the parametersdef foo_method(param1, param2): ....... endwhile calling the method, arg1 and arg2 are the argumentsfoo_method(arg1, arg2)