Rails Interview Training  
  
  
**Rails Evaluation**

* Which Architecture is used in rails? MVC
* How MVC works in rails
* Which problems are covered in MVC?
* Models for handling data and business logic and database communication
* Controllers for handling the user interface and application
* Views for handling graphical user interface objects and presentation
* Model role/functions in MVC and model derived from ActiveRecord::Base
* establishing a connection to the database server
* retrieving data from a table
* storing new data in the database
* Controller role/function in MVC and controllers are derived from base ActionController::Base
* deciding how to handle a particular request.
* retrieving data from the model to be passed to the view
* gathering information from a browser request and using it to create or update data in the model
* View contain presentation logic only
* What kind of reponses can be returned by controller
* returning HTML, but Rails controllers can also generate XML, JSON, PDFs, mobile-specific views, and more
* What is Action Dispatch?
* In Rails, incoming requests are routed by Action Dispatch to an appropriate controller. Action Dispatch and Action Controller are bundled together in Action Pack.
* Full form of .erb is embedded ruby codes with <%= ruby\_code %> in html message.
* 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
* What is ORM**(**Object-Relationship-Model)
* 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
* What are default environments in rails? testing, development, and production
* What is **app/helpers** directory holds any helper classes used to assist the view, model and controller classes.
* What is Rails Active Record in Ruby on Rails
* 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
* Directory Structure <https://www.sitepoint.com/a-quick-study-of-the-rails-directory-structure/>
* Mention what is the difference between a gem and a plugin in Ruby?
* **Gem:** A gem is a just ruby code. It is installed on a machine, and it’s available for all ruby applications running on that machine.
* **Plugin:** Plugin is also ruby code, but it is installed in the application folder and only available for that specific application.
* Gemfile specify dependencies for a Ruby application
* What is Gemfile.lock?
* It contains records of exact versions of gems installed. This is so the same versions can be installed if another machine clones the project.
* In contrast, specifying a gem in a Gemfile without pegging to a specific version will just install the latest version of a gem.
* What is the meaning of “Fat models, skinny controllers”?
* What is the meaning of “skinny controllers, skinny models”?
* Models can be made skinnier by keeping the single responsibility principle in mind and moving logic out of models, and into other design patterns like service objects.
* Is ruby strongly or weekly typed?
* Ruby is strongly typed. An error will be thrown if you try to calculate "hello" + 3.
* What are strong parameters
* ActionController::ParameterMissing exception in strong parameters
* ActiveModel::ForbiddenAttributesError exception in strong parameters
* params.require(:person).permit(:name, :age)
* Permitted Scalar Values in strong parameters
* params.permit(**:id**)
* rails db:reset & rails db:setup & rails db:seed
* When the **database already created** rails db:seed would be the best option.
* When the database is not ready yet, in other words, we want to create the database first, then we can go with rails db:setup to first run migrations and then **seeding**.
* And the last option is when we want to **reset the database**, rails db:reset will drop the database, create again, and seed the application
* Types of scaffolding
* Dynamic Scaffolding is used to give the model name that is being used with the function like: scaffold: model\_name.
* Whereas, Static Scaffolding requires manual entry in the command and it is given as Script/generate scaffold Post title: string content:text category\_id: integer
* Dynamic Scaffolding automatically generates the entire content and user interface at runtime. Whereas, Static Scaffolding requires the insertion of the command to, generate the data with their fields.
* Dynamic Scaffolding doesn’t require the database to be integrated as it gets created in runtime. Whereas, Static Scaffolding requires the database to, be migrated.
* Dynamic Scaffolding allows the generation of new, edit and deletes methods for the use in application. Whereas, Static Scaffolding doesn’t allow any such generation to, take place.
* What are rake tasks used to run tasks like rake db:create
* What are Turbolinks and how does it works
* *cache busting*
* how Rails keep track of which migrations have run for a database? using **schema\_migrations** table
* What are dirty objects? ones who have changed their attributes but not changed yet
* person = Person.new
* What following does?
* person.changed? returned true/false
* person.changed ["first\_name"]
* person.changed\_attributes {"first\_name"=>nil}
* person.changes {"first\_name"=>[nil, "First Name"]}
* How to check validations on Model? person.valid?
* Include vs Extend'include' is for adding methods only to an instance of a class and 'extend' is for adding methods to the class but not to its instance Lord.**new**.geeks (include), Star.geeks (extend)
* What do persisted? method do
* Override Active Record Naming Convention
* How to specify different db table name self.table\_name = "my\_products"
* Override primary column name by default id self.primary\_key = "product\_id"
* What are fixtures
* What are CRUDs
* find\_by vs find
* destroy vs destroy\_all
* save and update return false if validation fails
* save vs save!
* update vs update!
* What are migrations
* db:migrate vs db:rollback
* database-agnostic code that runs on Mysql Postgres
* which columns are created by default with tables created\_at and updated\_at
* up vs down methods
* rails generate migration AddPartNumberToProducts
* rails generate migration AddPartNumberToProducts part\_number:string:index (to add index)
* What is ActionCable
* open source directories
* lib vs vendor
* What is finger printing
* Active Model vs Active Support vs Active Record
* How routing works
* How if we want to make index and show route only resources :products, only: [:index, :show]
* What is DSL
* resources :products will create which routes
* does sequence of routes matters in routing? Yes
* resource vs resources
* what are namespace in routing and its use
* namespace vs scope vs module
* Nested Routes
* Shallow Nesting
* Concerns in routing
* member vs collection routes
* count vs length vs size
* difference bewteen class method and scope
* default scope
* how to unscope default scope? using unscoped
* presents? (selects all columns) vs exits? (does not selects columns)
* find vs exits?
* find vs find\_by
* sort vs order
* match keyword in routing to transfer all put post on a single route
* What are associations
* how many types of associations are there in rails? belongs\_to, has\_one, has\_many, has\_many :through, has\_one :through and has\_and\_belongs\_to\_many
* belongs\_to associations *must* use the singular term
* belongs\_to vs has\_one association
* usages of has\_many through other than many\_to\_many association
* The has\_many :through association is also useful for setting up “shortcuts” through nested has\_many associations
* has\_one :through associations vs has\_one
* What is has\_and\_belongs\_to\_many
* What is the use of Destructive Method? one that mutates the object it is called on. For example, the << method
* Render vs Redirect
* How to check if a specific key is present in a hash or not? session.key?("user")
* things you should know to make efficient use of Active Record associations in your Rails applications:
* Controlling caching
* Avoiding name collisions
* Updating the schema
* Controlling association scope
* Bi-directional associations
* when reload on model is important
* How to add foreign key to a table
* t.references **:author**
* add\_reference :books, :author
* webserver vs app server
* which server you are using, puma
* puma is web server or app server?
* lifecycle of web request cycle (where do lie app server or web server )
* https vs http
* what is ssl
* what is csrf attack? (token verification?)
* what is fishing?
* protect\_from\_forgery?
* MVC request cycle
* active record vs active model
* actionPack
* actioncontroller
* how background works?
* does background worker run in separate server or same server?
* load distribute
* redis
* action cable
* action cable broadcasting on a specific channel
* model callbacks
* after save when run in database? create , update
* after commit run
* after create
* where do gem install
* if we install latest version of ruby, do we have accessible old gems of old ruby ? No
* is rails gem?
* Chrone job
* serializers
* alternate to serializers
* how to invalid a true object? using errors.all
* what is and how to do STI
* eager load vs pre load in includes
* User.find\_by([1,2]) vs User.find([1,2])
* headers request type

sitepoint.com**sitepoint.com**

[**A Quick Study of the Rails Directory Structure — SitePoint**](https://www.sitepoint.com/a-quick-study-of-the-rails-directory-structure/)

Read A Quick Study of the Rails Directory Structure and learn with SitePoint. Our web development and design tutorials, courses, and books will teach you HTML, CSS, JavaScript, PHP, Python, and more.

**Written by**

Vinoth

Feb 29th, 2016

[9:09](https://devsinc.slack.com/archives/C04N54F4A2V/p1675354199878369)

Rich Text File

**[File from iOS (1)](https://files.slack.com/files-pri/T0A1N9R2S-F04MSF68SQ3/download/interviewnotes.rtf?origin_team=T0A1N9R2S)**

[Rich Text File](https://files.slack.com/files-pri/T0A1N9R2S-F04MSF68SQ3/download/interviewnotes.rtf?origin_team=T0A1N9R2S)

[9:11](https://devsinc.slack.com/archives/C04N54F4A2V/p1675354313442429)

```

rails intro

active support---???-(extended funtionality)

rake kya hta ??

gemfil lock

data seeding

open source librarires use krty hwy kya dehan rkhty h ??

activejob/sidekiq difference

turbolinks

jquery function

shallow routes

member routes/ collection

resource/resources

n+1 query problem

Order of call backs when an object is created, deleted, updated and destroyed

pre -oading

Why icludes is used?

Find and find\_by difference.

kin scenaiors mei preload ki trha beehve krta h include or kin senerio mei eagar load ki trhakrt h

before save/before create

proc/lamda ??

action mei return lga hwa h to proc??

proc ko kitny treeko sy call kr skty h??

services/concerns

lib k folders mei kya htanh

serializers

git

select

kis case mei select use hoga??

further data base processing mein use hta h

partials

locals??

collection kysy pass krty hain partials mei??

form

remote ajax..

comtroller kysa response returns

exits? present?

layout folder.

page ka base structure

mailer view..

asset pipeline ni likhty in mein sari alag sy css krty h.

session vs cookies

callbacks/validations mei purani value get krni ho toh

builtin validations mei

single table inheritancle in database

has many through

pre compilations kya hti h

devlopmnt hti h ?

pre comile comand

finger print..

minification

forgery protection??

form\_with/ form\_tag / form\_for??

call back skip

class method/scope??

```

[9:13](https://devsinc.slack.com/archives/C04N54F4A2V/p1675354399325319)

```

Rails Questions

Select vs Pluck

Eager loading/ pre loading

joins

aggregate functions

exist vs present

indexes

before/after\_commit vs before/after\_create

before/after\_validation, around\_save etc

Get old values of attributes in callbacks

polymorphic associations

find vs find-by

Join, includes, left outer join

fingerprinting

load paths

Devise extensions

Data seeding

rake command

Libraries

bin folder

Ajax

joins vs include

Whats is Rails

What are Rails components

What is Rails versioning policy

Define Rails directory structure, lib folder, bin folder, public folder

Background jobs in ruby

ActiveModel vs ActiveRecord

Polymorphic Associations

What does it mean by convention over configuration

What does has and belongs to many do?

Update vs Update!

Delete vs Destroy

Delegate

Single Table Inheritance

Self association

Observers in rails

Webpacker, Action Cable, Asset Pipeline

=======================

Inject example by Ahmad:

INTERPOLATION\_MAPPING = {

'Stage Name' => 'stage.name',

'Created On' => 'created\_at',

'Updated On ' => 'updated\_at'

}.freeze

context = {}

INTERPOLATION\_MAPPING.each do |key, attribute|

context[key] = public\_send(attribute) attribute.split('.').inject(self) { |obj, fun| obj.send(fun) }

end

here self is a model object lets say User.first and user belongs\_to stage.

====================

+————————-+———————————+

| Name Begins With | Variable Scope |

+————————-+———————————+

| $ | A global variable |

| @ | An instance variable |

| [a-z] or \_ | A local variable |

| [A-Z] | A constant |

| @@ | A class variable |

+————————-+———————————+

=====================================================================

class NotificationService

def self.initialize

end

def call

puts "Hello World!"

end

end

->

suppose we have a service

class WajtayNiHou

def call

puts "Hello World!"

end

end

We need to execute call method without initializing this class.

NotificationService.new().call

=====================================================================

find\_no\_of\_pairs([1,2,3,4,2,3,4,2,3,5,6,7, 3])

def find\_no\_of\_pairs(array)

res = {}

array.each do |a|

if res[a].present?

res[a] += 1

else

res[a] = 1

end

end

res.map {|key, value| puts "#{key}: #{value/2}" }

end

```

[9:14](https://devsinc.slack.com/archives/C04N54F4A2V/p1675354445437869)

```

ROR Interview Questions => American

1) What is Rails?

2) What is MVC?

3) Talk about Rails and security.

4) Describe convention over configuration, and its effect on security. How does it affect upgrading?

5) What is a controller?

6) What is a model?

7) How do models work?

8) What is an abstract model?

9) What is STI?

10) How does STI work? How does Rails implement it?

11) What is polymorphism? How does it work in Rails? Give an example. What does HABTM mean? How does Rails implement this internally?

12) Can you include methods from one class in another class? How about from modules? What are these called?

13) What is a concern? How are they used?

14) What is a migration? How do you write them? How do you run them? How do you revert them?

15) What is an irreversible migration?

16) What is an index? Why should you use them?

17) What is schema.rb for?

18) What are seeds?

19) What are rake tasks? Where do they live?

20) Describe how routing works. What are the verbs? What are they for? How does nesting work? How do the params work?

21) How does pluralization work in Rails? Which areas use which forms?

22) What is a controller filter? Which filters are there? How do you apply/restrict them?

23) What are strong params? How do you use them?

24) Describe the difference between public, private, and protected methods

25) What is method\_missing? How is it useful? Can you give some examples of how Rails uses it internally?

26) What is ActiveRecord?

27) What is an ARel table?

28) What is a helper?

29) What are view helpers specifically?

30) How do you use a helper to create a link? a select? a form?

31) How do nested (children) objects work? e.g. allowing a user to create a Person record with `n` associated Contact records from the same form?

32) How do you prevent creating blank children? How do you destroy them along with the parent?

33) What is a hash? array? HashWithIndifferentAccess?

34) What is the difference between a symbol and a string? When (and why) would you use one over the other?

35) How do you store arbitrary precision decimals?

36) How do you interact with the Ruby GC? Why would you?

37) What are specs? What is RSpec?

38) What are fixtures? factories?

39) When would you use them?

40) What is code coverage?

41) What is feature coverage?

42) Which is more important?

43) What is CI?

44) How do you run specs in parallel?

45) What are shared specs?

46) What is an integration spec?

47) Can you test routing? helpers? markup? css?

47) What is Capybara?

48) What is ActiveSupport? What features does it provide?

49) What is ActionCable? How does it work?

50) What is the Rails logger? How do you use it?

51) What is a layout? What is a view? What is a partial? How do you pass data between them? How does Rails make this work?

52) What is an API?

53) How do you version APIs?

54) What is exponential backoff?

55) What is the asset pipeline?

56) What is cachebusting?

57) How do you deploy a Rails application?

58) What is the Rails cache? How does it work? List three ways you can use it, and why/when it's a good idea.

59) What is Unicorn / Passenger? Why are they useful?

60) What is the splat operator? When do you use it?

61) What is the difference between a Proc and a Block?

62) What is the differene between a Proc and a lambda?

63) What is the spaceship operator?

64) How do you safe-chain methods? When is this a good idea? When is it a bad idea?

65) What is the difference between safe-chaining and .try()?

66) What does "Foo#bar" mean in documentation?

67) Why should you not use sequential IDs in anything public-facing?

68) In migrations, how do you limit data types? Why would you? Give examples.

69) What is a Regex?

70) What is a case statement? Why and when should you use these in place of if/else's?

71) How does Ruby error handling work?

72) What are the different types of Errors and Exceptions? What is the difference?

73) What is middleware? Can you give an example?

74) What is Rack specifically?

75) Why did you pick Rails?

76) Have you ever used Ruby by itself?

77) What is the difference between `&&` and `and`? `||` and `or`? `!` and `not`?

78) What is the differene between `if !` and `unless`?

79) How do you force integer division? float division?

80) Give an example of syntatical sugar, and an example of when using it can be a good idea. Give an example of when using it is a bad idea.

81) What is the difference between `Object::Thing` and `Object.thing`?

82) How do a class method and instance method differ?

83) What is an instance variable? What is a class variable? What is a global?

84) When do variables fall out of scope, specifically with procs/blocks?

85) Is Ruby pass by value or reference? When?

86) Describe the difference between `each`, `map`, and inject

87) How do you assign to multiple variables at once?

88) What is an implicit return? When is using it a good idea? When is using it a bad idea?

89) What are questionmark methods?

90) What are bang methods?

91) What are scopes?

92) What are validations? What types are there? How do you use them e.g. on forms?

93) What is the point of the Application controller?

94) What is CSRF? How does Rails protect against it?

95) What is a JWT? What is it useful for?

96) How does `render` work? How do you render different actions? different layouts? nothing?

97) How do you redirect?

98) How do you send files? streams? inline PDFs?

99) How do you schedule off-thread background jobs?

~~100) Who is Matz?~~

ROR => Sendoso Interview

TTD k mutaliq kaafi questions thy. Rspec, FactoryBot, Traits.

SecureRandom.uuid k mutaliq ik question tha.

Database related kuch questions thy, indexing kya hoti hai kitni types ki.

Ik question algo ka tha uss ka code screen share kar k likhwaya tha. Question nai yaad a raha abhi.

Resources vs Resource routes.

What are RESTful routes.

Migrations ka kya maqsad hai.

Module vs concerns