Ruby on Rails Interview Questions for Freshers

1. What is Ruby on Rails?

- Ruby is a high-level, interpreted programming language that supports a variety of programming paradigms.
- Rails, on the other hand, is a framework that can be used to build web applications.

2. Explain the naming convention in Rails.

- **Variables:** All letters are lowercase, and words are separated by underscores.
- **Class and Modules:** Mixed case, no underscore, each word starts with an uppercase.
- **Database Table:** The database table name should have lowercase letters and underscore between words; all table names should be in the plural form.
- **Model:** Mixed-case, always has singular with the table name.
- **Controller:** Represented in plural form like OrdersController.

3. What is ORM(Object-Relationship-Model).

In Rails, the ORM or Object Relationship Model indicates that your classes are mapped to the database table, and objects are directly mapped to the table rows. The attributes and relationships of objects in an application may be easily stored and retrieved from a database using ORM, which requires less overall database access code and does not require writing SQL queries directly.

4. What is the difference between false and nil in Ruby?

- True and false are boolean values in Ruby that signify yes and no, respectively. True is a TrueClass object, and False is a FalseClass object.
- Nil is a special value in Ruby that represents the lack of any value. Nil is a NilClass object. Nothing or void is referred to as nil in Ruby.

5. What is the difference between String and Symbol?

Strings are any text typed between quote marks ("this is a string," "so is this," "this too!") and symbols are text that starts with a colon (: symbol). Strings and symbols, on the other hand, have diverse functions that make them valuable for different programming tasks.

"Scalar value objects used as identifiers, mapping immutable strings to fixed internal values," according to the Ruby definition. This essentially indicates that symbols are immutable strings.

6. What command can you use to create a controller for the subject?

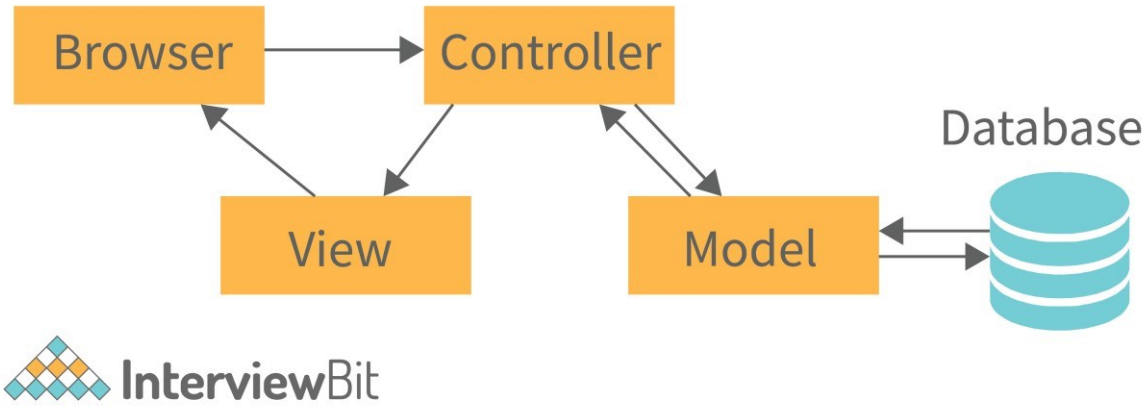
Use the following command to build a controller for the subject:

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

7. What do subdirectory app/controllers and app/helpers do?

- **apps/controller:** The controller classes are found in the app/controllers subfolder, which Rails searches for. A user's web request is handled by a controller.
- **app/helpers:** Any helper classes needed to assist the model, view, and controller classes are stored in the app/helpers subdirectory. This keeps the model, view, and controller code clean, simple, and focused.

8. What is a Rails Controller?



Your application's logical heart is the Rails controller. It orchestrates the user's interaction with the views and the model. A number of key ancillary functions are also housed in the controller, including:

It is in charge of directing external requests to internal processes.

- It controls caching, which can improve application performance by orders of magnitude.
- It handles helper modules, which add functionality to view templates without adding code to them.

-

- It keeps track of sessions, creating the impression that users are still interacting with our apps.

9. What is Rails Active Record in Ruby on Rails?

The Object/Relational Mapping (ORM) layer included with Rails is called Active Record. It roughly resembles the conventional ORM model, which goes like this:

- Tables correspond to classes.
- Rows correspond to objects, and
- The columns correspond to the attributes of the objects.

Active Records in Rails provide a connection and interface between relational database tables and Ruby computer code that manipulates database records.

10. Explain Rails Migration.

A Rails migration is a tool for altering the database schema of an application. Instead of handling SQL scripts, you use a domain-specific language to define database modifications (DSL). Because the code is database-agnostic, you can quickly port your project to a different platform. Migrations can be rolled back and managed alongside your application source code.

11. Explain how rail implements Ajax?

The way Rails supports Ajax operations is simple and consistent. Different user actions force the browser to display a new web page (like any regular web application) or initiate an Ajax activity after the original web page has been produced and displayed.

- **Some trigger is fired-** This could be a user clicking on a button or link, a user changing data on a form or in a field, or just a recurring trigger (based on a timer).
- **The server is contacted by the web client-** The XMLHttpRequest JavaScript function transmits data associated with the trigger to a server action handler. The data could be the checkbox ID, the text in an entry field, or the entire form.

- **The server performs the processing-** The server-side action handler (Rails controller action) manipulates the data and sends an HTML fragment to the web client.
- **The HTML fragment is received by the client-side JavaScript**- The client-side JavaScript which Rails generate automatically is used to alter a specific area of the current page's HTML, usually the content of an element.

12. What command is used to create a migration?

To create migration command includes:

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

13. What does garbage collection do in Ruby on Rails?

Garbage collection is a technique for controlling the amount of memory used by computer programs. Garbage collection and other memory management techniques, like reference counting, work by having the language keep track of which objects are in use by a program rather than the developer. This allows the programmer to concentrate on the business logic or other challenge at hand rather than the intricacies of memory allocation and release. This also aids program stability and security, as improper memory management can cause crashes, and memory management bugs account for a major fraction of security bugs.

14. Explain Cross-Site Request Forgery (CSRF). How is Rails protected against it?

Cross-Site Request Forgery (CSRF) is a typical online application attack that compromises a victim's authenticated session. This attack entails duping a target into executing unwanted actions on a website to which they have been authenticated.

You must add "protect from forgery" to your ApplicationController to protect against CSRF attacks. Rails will now require a CSRF token in order to complete the request. Every form built using Rails forms builders includes a hidden field called CSRF token.

15. How to define Instance Variable, Global Variable, and Class Variable in Ruby?

Instance Variable in Ruby begins with– **@**

- Class variable in Ruby begins with– **@@**
- The global variable in Ruby begins with– **$**

•

16. What is the role of load and require in Ruby?

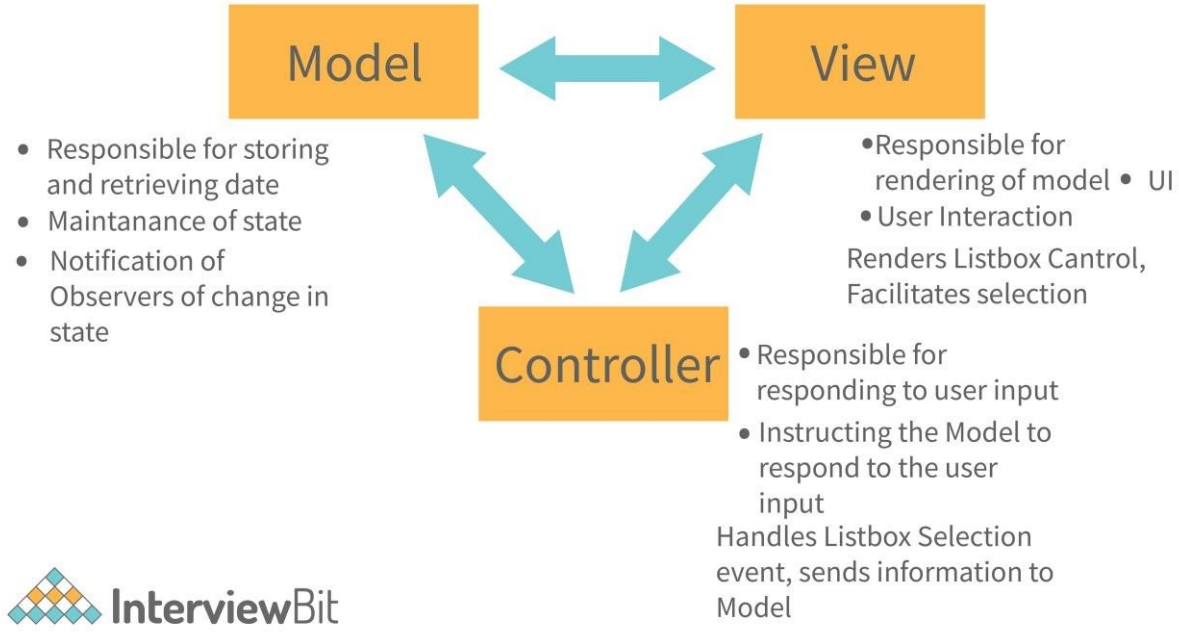
- **load( )-** We use load to execute code.
- **require( )-** We use require to import libraries.

17. What are access modifiers in Ruby?

1. **Public:** In this, all members are available to everyone to modify.
2. **Private:** In this, only functions inside the class can access members.
3. **Protected:** In this, the members can only be accessed by functions inside the subclass.

18. What is MVC and how does it work?

The **Model-View-Controller** (MVC) architectural pattern divides an application into three logical components: model, view, and controller. Each of these components is designed to handle specific parts of application development.



The request initially goes to the controller, who then chooses a suitable view and interacts with the model, who then interacts with your database and sends the response to the controller, who then gives the output parameter to the view based on the response.

Ruby on Rails Interview Questions for Experienced

1. What is the difference between observers and callbacks in Ruby on rails?

- Callback methods in Rails can only be called at specific moments in an object's life cycle, such as validation, creation, updating, deletion, and so on. The rails callback, unlike the rails observers, is only active for a brief time.
- Rails observers are similar to callbacks, but they're used when a method isn't directly related to the object's life cycle. It can be attached or detached at any time and lives for a longer period of time.

2. What is the purpose of the rakefile available in the demo directory in Ruby?

This brief questionnaire is designed to ensure that a developer is familiar with testdriven development. This file may be unfamiliar to a newbie. The rakefile, which is analogous to the makefile in Unix, is used to package and test Rail's code. The rake utility, which comes with the Ruby installation, makes use of it.

3. Explain the difference between ActiveSupport's "HashWithIndifferentAccess" and Ruby's "Hash"?

The "HashWithIndifferentAccess" class treats symbol and string keys as equivalent, whereas Ruby's "Hash" class uses a tighter = = comparison on keys: thus a comparable string key will not get the value for a given symbol key.

4. What is the difference between string and text in Rails?

- Both string and text save information of the "string-type" that you can freely write in. The number of characters you can enter in these fields differs

between the two. The character limit for a string field is 255 characters, while the character limit for a text field is 30,000 characters.

- If you wish to store data like addresses, names, or basic custom data, a string field is an excellent option. When you want to store information from a comment box on a form, or if you're importing a huge block of text, a text area field is an excellent solution.

5. Explain the difference between dynamic and static scaffolding.

Dynamic Scaffolding	Static Scaffolding
At runtime, it generates all of the content and user interface.	To produce the data with their fields, it takes explicit entry in the command.
It allows you to create new, delete, and modify methods for usage in your application.	In static scaffolding, It is not necessary for such generation to occur.
It does not require synchronization with a database.	It necessitates the migration of the database.

6. What are strong parameters? Explain in brief.

Many Rails apps employ Strong Parameters, also known as Strong Params, to strengthen the security of data supplied through forms. Strong parameters allow developers to determine which parameters are accepted and used in the controller. Any superfluous or potentially hazardous params will be ignored and successfully filtered out by allowing only the expected params. This is especially crucial when using Active Model bulk assignments, as numerous params might be provided at the same time.

7. Does Ruby Support Single Inheritance/Multiple Inheritance Or Both?

Ruby only supports single inheritance. It does not support multiple inheritance directly, but it supports something similar- mixins.

8. What are the limits of Ruby on Rails?

Ruby on Rails is a framework for building MVC-based CRUD web applications. Other programmers may find Rails unusable as a result of this. The following are some of the functionalities that Rails does not support.

- Databases with foreign keys.
- Linking to many databases at the same time.
- Web services for soap.
- Multiple database servers are connected at the same time.

Ruby on Rails Coding Interview Questions

1. Write a function in Ruby to check if the string is a palindrome.

```
def palindrome?(str)
  str == str.reverse
end
```

2. What will be the outp ut of the given code snippet?

```
x, y, z = 12, 36, 72
   puts "The value of x is #{ x }."
   puts "The sum of x and y is #{ x + y }."
   puts "The average was #{ (x + y + z)/3 }."
```

Output:

```
The value of x is 12.
The sum of x and y is 48.
The average was 40.
```

3. What is the problem with the following controller code? How would you fix it?

```
class MyController < ApplicationController
def options       options = {}
      available_option_keys = [:first_option, :second_option, :third_option]       all_keys =
params.keys.map(&:to_sym)
      set_option_keys = all_keys & available_option_keys
set_option_keys.each do |key|         options[key] =
params[key]       end       options      end




   end
```

Converting user-supplied arguments to symbols is risky since symbols in Ruby are not garbage collected. An attacker may submit a series of requests containing random keys that would be converted to symbols, quickly draining your server's memory and bringing your site down.

There are two possible solutions to this problem. The first is to use a slice to filter out non-option-key values from the params hash. This would seem as follows:

```
params.slice(*available option keys)
```

Using String keys for your selections is an alternate, and some would argue a better solution. Unless you have a very high number of viable choice keys, using Symbol keys instead will not save you any memory.

4. What paths (HTTP verb and URL) will be defined by the following snippet in config/routes.rb?

```
resources :posts do
  member do
    get 'comments'
  end
  collection do
    post 'bulk_upload'
  end
end
```

Using the resource method to define routes will automatically generate routes for the standard seven restful actions:

- GET /posts
- POST /posts
- GET /posts/new
- GET /posts/:id/edit
- GET /posts/:id
- PATCH/PUT /posts/:id
- DELETE /posts/:id

5. Given this input:

```
x = [{"a" => 10},{"b" => 20},{"c" => 30}]
```

How will you obtain the following:

- **One array containing all keys**
- **Another containing all values**

The code will look something like this:

```
y = x[0].merge(x[1]).merge(x[2])

y.keys              # will return all keys
y.values            # will return all values
```

FAQ'S

1. What does a Ruby on Rails developer do?

A Ruby on Rails developer is in charge of building Ruby-based server-side web application logic for the Rails framework. Roles and responsibilities of a Ruby on rails developer include

- New web application design and development.
- Existing web apps' maintenance and troubleshooting. • Writing and maintaining dependable Ruby code
- Bringing data storage options together.
- Back-end component development.
- Finding and eliminating constraints and bugs.
- Additional web servers are connected to apps.
- API management.

2. Is Ruby on Rails in demand?

Ruby on Rails has been around for a long time and has proven reliable. Although its popularity has waned over time, hundreds of developers continue to trust and enjoy working with it.

3. How much does a Ruby on Rails developer make?

According to the ambition box, the average yearly income for a Ruby on Rails Developer in India is 6.5 lakhs, with salaries ranging from 1.9 lakhs to 15.0 lakhs.

4. Is Ruby on rails good for freelancing?

With all the frameworks and libraries available (such as Rails, Merb and Sintara), Ruby is definitely great for freelancing. Its focus on clean code and object-oriented syntax, and a strong community will also help you a lot in your freelancing career.

5. What is something you like about working with Rails?

Ruby on Rails is fantastic because it simplifies basic web development processes, allowing developers to concentrate on their app's functionality. To do simple procedures, you no longer need to create endless database queries.

6. Is Ruby front end or backend?

Ruby On Rails is a front-end and back-end framework. Because Ruby on Rails covers both the frontend and the backend, you may consider yourself a true fullstack developer.