IIC 2143 - Ingeniería de Software

Asociaciones

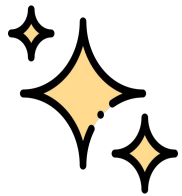
M. Trinidad Vargas
mtvargas1@uc.cl

Actividad con décimas

Completa los controles 1 y 2 en canvas sobre metodologías y un poco de Ruby on Rails

- De forma **individual**
- 3 intentos sin límite de tiempo
- **No se permiten preguntas** sobre el contenido al cuerpo docente
- 1 décima por cada control si tienen 8 o más preguntas correctas

Disponible hasta el miércoles 22:00 hrs



Active Record

Es un ORM (object relational mapping) que facilita la interacción entre la aplicación y la base de datos

Active Record

- all recupera todos los registros de la tabla del modelo
- first recupera el primer registro según la clave primaria (id)
- find(id) recupera el registro según su clave principal
- where(field: value) filtra según condiciones
- order(:field) por defecto ascendente o order(field: desc)

create, update, destroy, count, sum, average y muchos más

https://guides.rubyonrails.org/active_record_querying.html

Asociaciones

Define las relaciones entre diferentes modelos, reflejando las conexiones entre las tablas de tu base de datos

Al establecer las asociaciones se gestiona los datos asociados de forma eficaz y se puede acceder usando métodos simples

Asociaciones

Asociaciones disponibles

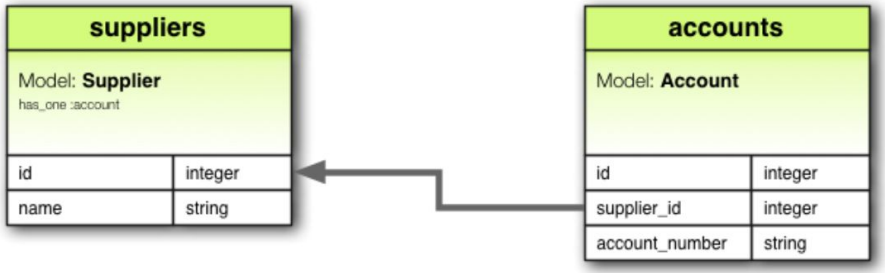
- belongs_to
- has_one
- has_many
- has_many :through
- has_one :through
- has_and_belongs_to_many

https://guides.rubyonrails.org/association_basics.html

Otras asociaciones

Asociación 1 a 1

- Una cuenta tiene un Proveedor
- en Rails: has_one

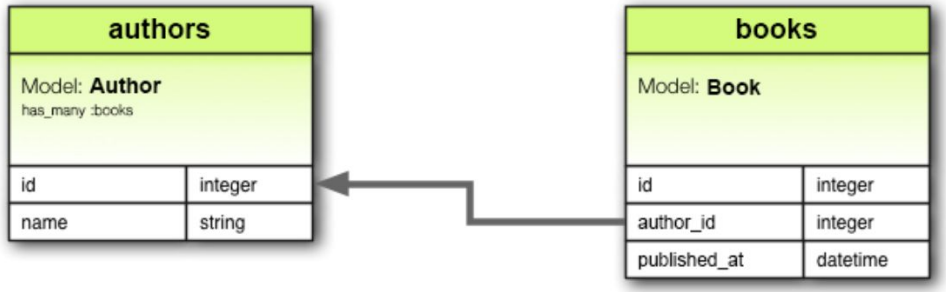



```
class Supplier < ApplicationRecord
  has_one :account
end
```

Otras asociaciones

Asociación 1 a muchos

- Un autor tiene muchos libros
- en Rails: has_many

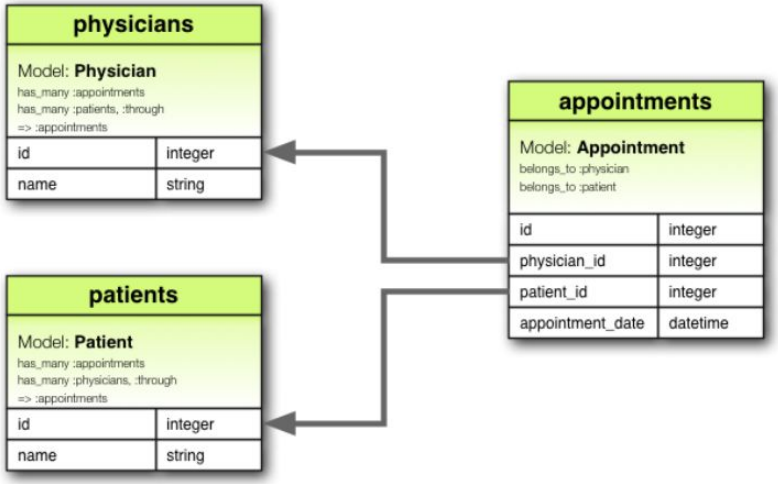



```
class Author < ApplicationRecord
  has_many :books
end
```

Otras asociaciones

Asociación muchos a muchos

- Un paciente tiene muchos médicos a través de horas agendadas
- en Rails: has_many :through




```ruby
class Physician < ApplicationRecord
  has_many :appointments
  has_many :patients, :through => :appointments
end

class Appointment < ApplicationRecord
  belongs_to :physician
  belongs_to :patient
end

class Patient < ApplicationRecord
  has_many :appointments
  has_many :physicians, :through => :appointments
end
```

Asociaciones

Ejercicio: Crear un mvp de un e-commerce

Necesitamos los siguientes modelos

- Productos
- Órdenes
- Ítems de la orden

```
rails new ecommerce --database=postgresql --api

rails db:create
```

Asociaciones

Productos

```
rails generate model Product name:string units_available:integer
price:decimal
```

Asociaciones

Órdenes

```
rails generate model Order status:string discount_amount:decimal
amount_paid
```

Ítems de las Órdenes

```
rails generate model OrderItem units:integer price:decimal order:references
product:references
```

```
rails db:migrate
```

Asociaciones

```
tshirt = Product.create(name: "Black T-shirt", price: 8500, units_available: 20)

socks = Product.create(name: "Socks", price: 3500, units_available: 5)

order = Order.create(status: "in_creation")

item = OrderItem.create(order: order, product: tshirt, units: 1, price: 8500)
```

Asociaciones

```
rails c

OrderItem.column_names
```

```
3.3.6 :003 > OrderItem.column_names
 => ["id", "units", "price", "order_id", "product_id", "created_at", "updated_at"]
```

* A las referencias Active Record agrega `_id` al nombre de la columna

Asociaciones

OrderItem pertenece al producto y a la orden

```
class OrderItem < ApplicationRecord
 belongs_to :order
 belongs_to :product
end
```

Por defecto podemos acceder al producto y a la orden de un OrderItem

Asociaciones

Agregamos las asociaciones en Producto y Orden

```
class Product < ApplicationRecord
 has_many :order_items
end

class Order < ApplicationRecord
 has_many :order_items, dependent: :destroy
end
```

Ahora podemos acceder a todos los items vendidos de un producto y a todos los ítems de una orden

Asociaciones

Ahora podemos acceder a todos los items vendidos de un producto y a todos los ítems de una orden

`order.order_items`

También podemos crear un item desde la orden

`order.order_items.create(product: socks, price: 3000)`

```
3.3.6 :004 > order.order_items.create(product: socks, price: 3000)
  TRANSACTION (0.1ms)  BEGIN
  OrderItem Create (1.2ms)  INSERT INTO "order_items" ("units", "p
[["units", nil], ["price", "3000.0"], ["order_id", 1], ["product_i
  TRANSACTION (3.0ms)  COMMIT
 =>
#<OrderItem:0x0000000126a52dd8
 id: 2,
 units: nil,
 price: 0.3e4,
 order_id: 1,
 product_id: 2,
 created_at: "2025-03-24 11:50:07.494816000 +0000",
 updated_at: "2025-03-24 11:50:07.494816000 +0000">
3.3.6 :005 > order.order_items
  OrderItem Load (2.9ms)  SELECT "order_items".* FROM "order_items
 =>
[#<OrderItem:0x0000000120a103d0
  id: 2,
  units: nil,
  price: 0.3e4,
  order_id: 1,
  product_id: 2,
  created_at: "2025-03-24 11:50:07.494816000 +0000",
```

Asociaciones

¿Cómo obtener todos los Productos de una orden? ¿Y todas las órdenes donde he vendido un producto?

Usamos la asociación **`has_many :through`** que nos permite relacionar dos modelos a través de un tercero

```
class Product < ApplicationRecord
 has_many :orders, through: :order_items
end

class Order < ApplicationRecord
 has_many :products, through: :order_items
end
```

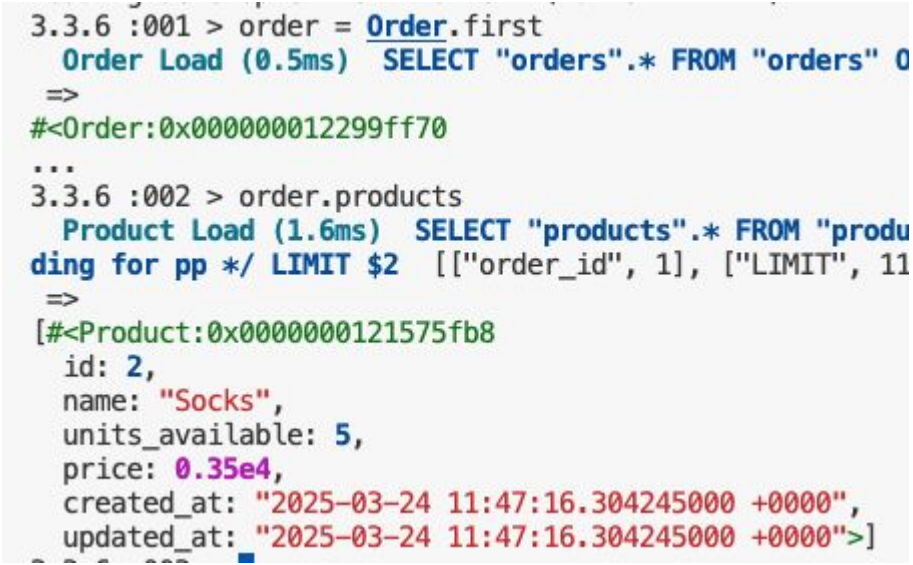
Asociaciones

Podemos obtener todos los productos de la orden

`order.products`

Y todas las órdenes donde se ha vendido un producto

`socks.orders`



Asociaciones

Vamos a crear el envío de la Orden

- Una Orden tiene solo un envío
- El envío no existe sin la orden

```
rails generate model Shipping address:string date_of_arrival:date
order:references
```

```
rails db:migrate
```

Asociaciones

La asociación has_one nos permite indicar que la orden tiene solo un envío.

```
class Shipping < ApplicationRecord
 belongs_to :order
end

class Order < ApplicationRecord
 has_one :shipping, dependent: :destroy
end
```

Asociaciones

```
order.create_shipping(address: "Vicuna Mackenna 123", date_of_arrival:
Date.today + 5.days)
```

```
3.3.6 :004 > order.create_shipping(address: "Vicuna Mackenna 123", date_of_arrival: Date.today + 3.days)
  TRANSACTION (0.3ms)  BEGIN
  Shipping Create (2.5ms)  INSERT INTO "shippings" ("address", "date_of_arrival", "order_id", "created_at"
s", "Vicuna Mackenna 123"], ["date_of_arrival", "2025-03-27"], ["order_id", 1], ["created_at", "2025-03-24
  TRANSACTION (4.5ms)  COMMIT
  TRANSACTION (0.1ms)  BEGIN
  Shipping Destroy (0.4ms)  DELETE FROM "shippings" WHERE "shippings"."id" = $1  [["id", 1]]
  TRANSACTION (0.2ms)  COMMIT
 =>
#<Shipping:0x00000001256d7588
 id: 2,
 address: "Vicuna Mackenna 123",
 date_of_arrival: "2025-03-27",
 order_id: 1,
 created_at: "2025-03-24 12:11:38.792149000 +0000",
 updated_at: "2025-03-24 12:11:38.792149000 +0000">
```

* Si intento crear un segundo envío para una misma orden, elimina el anterior

Asociaciones

no puedo crear un envío sin una orden

```
shipping = Shipping.new(address: "Calle 123")

shipping.save
```

```
3.3.6 :006 > shipping = Shipping.new(address: "Calle 123")
 => #<Shipping:0x00000001256d7088 id: nil, address: "Calle 123", date_of_arrival: nil, order_id: nil, created_at: nil, updated_at: nil>
3.3.6 :007 > shipping.save
 => false
3.3.6 :008 >
```

Asociaciones

Ejercicio: Crea el modelo Customer

- Los clientes pueden tener varias órdenes
- Agrega la columna customer a Order usando una nueva migración

Tips de debugeo

Uso de debugger

- Es una herramienta muy poderosa
- Requiere dedicarle tiempo a la instalación y adaptación pero ahorra mucho tiempo
- Te permite corregir errores y aprender rápidamente

Tips de debugeo

Configuración del debugger

Cada editor de texto tiene su propia guía de instalación

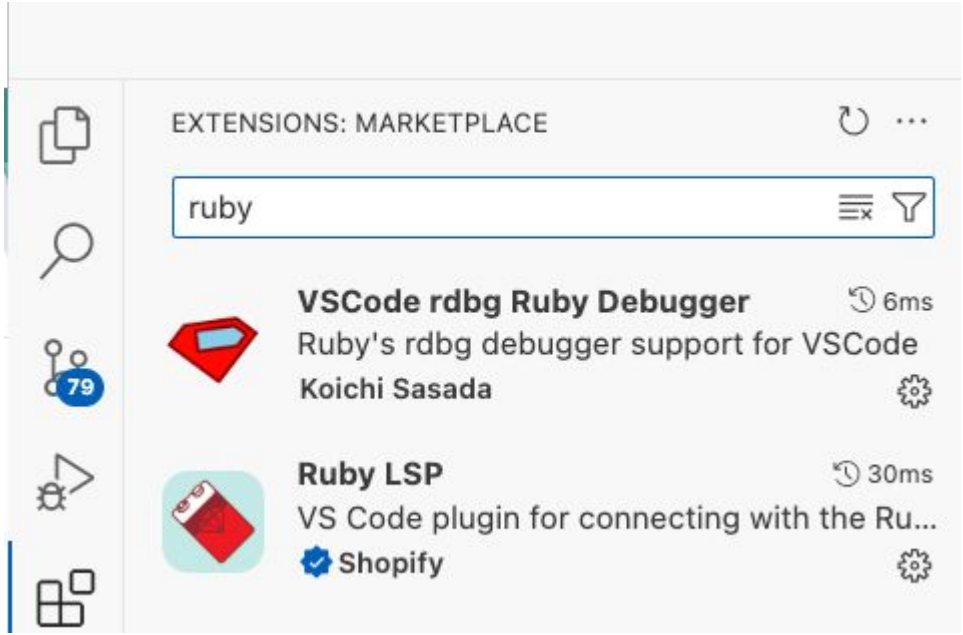
Haremos el ejemplo con VSCode por ser el más utilizado

Si usas otro, busca en internet como configurarlo ¡Existen miles de blogs con el paso a paso!

Tips de debugeo

VSCode

1. Instala las extensiones **rdbg** y **Ruby LSP**



Tips de debugeo

2. Instala la gema **debug**

`gem install debug`

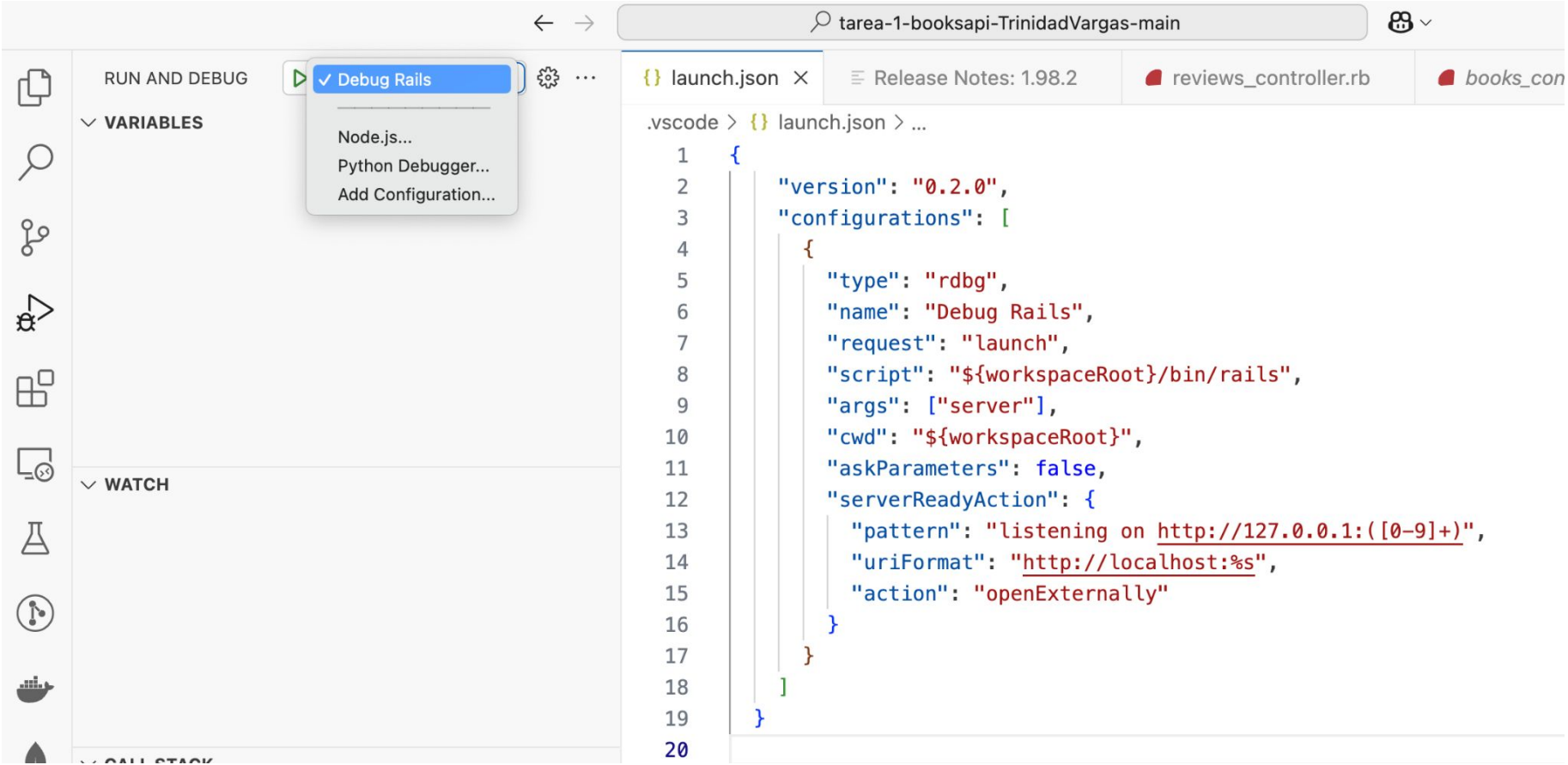
```
trinidadvargas@MacBook-Pro-de-Trinidad-2 tarea-1-booksapi-TrinidadVargas-main % gem install debug
Building native extensions. This could take a while...
Successfully installed debug-1.10.0
Parsing documentation for debug-1.10.0
Installing ri documentation for debug-1.10.0
Done installing documentation for debug after 0 seconds
1 gem installed

A new release of RubyGems is available: 3.5.22 → 3.6.6!
Run `gem update --system 3.6.6` to update your installation.
```

Tips de debugeo

3. Crea el archivo launch.json

```
{
   "version": "0.2.0",
   "configurations": [
     {
       "type": "rdbg",
       "name": "Debug Rails",
       "request": "launch",
       "script": "${workspaceRoot}/bin/rails",
       "args": ["server"],
       "cwd": "${workspaceRoot}",
       "askParameters": false,
       "serverReadyAction": {
         "pattern": "listening on http://127.0.0.1:([0-9]+)",
         "uriFormat": "http://localhost:%s",
         "action": "openExternally"
       }
     }
   ]
 }
```


tarea-1-booksapi-TrinidadVargas-main
RUN AND DEBUG
Debug Rails
Node.js...
Python Debugger...
Add Configuration...
VARIABLES
WATCH
launch.json
Release Notes: 1.98.2
reviews_controller.rb
books_con
.vscode > launch.json > ...
{
"version": "0.2.0",
"configurations": [
{
"type": "rdbg",
"name": "Debug Rails",
"request": "launch",
"script": "${workspaceRoot}/bin/rails",
"args": ["server"],
"cwd": "${workspaceRoot}",
"askParameters": false,
"serverReadyAction": {
"pattern": "listening on http://127.0.0.1:([0-9]+)",
"uriFormat": "http://localhost:%s",
"action": "openExternally"
}
}
]
}

Tips de debugeo

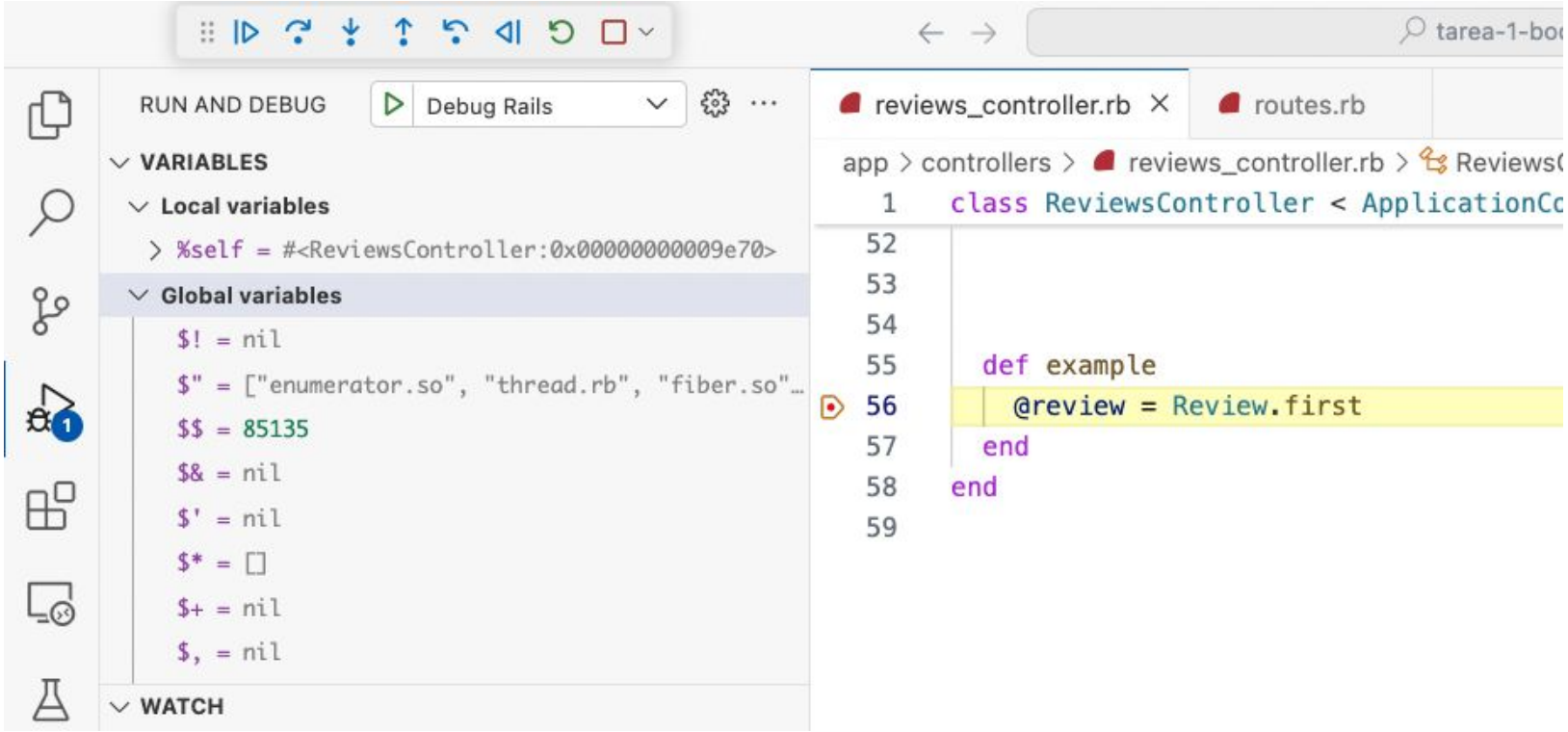
4. Inicia el debugger

Tips de debugeo

5. Agrega breakpoints

(puntos rojos)

Puedes ver los valores de las variables en cada línea



Puedes moverte a la línea siguiente con las flechas o teclas

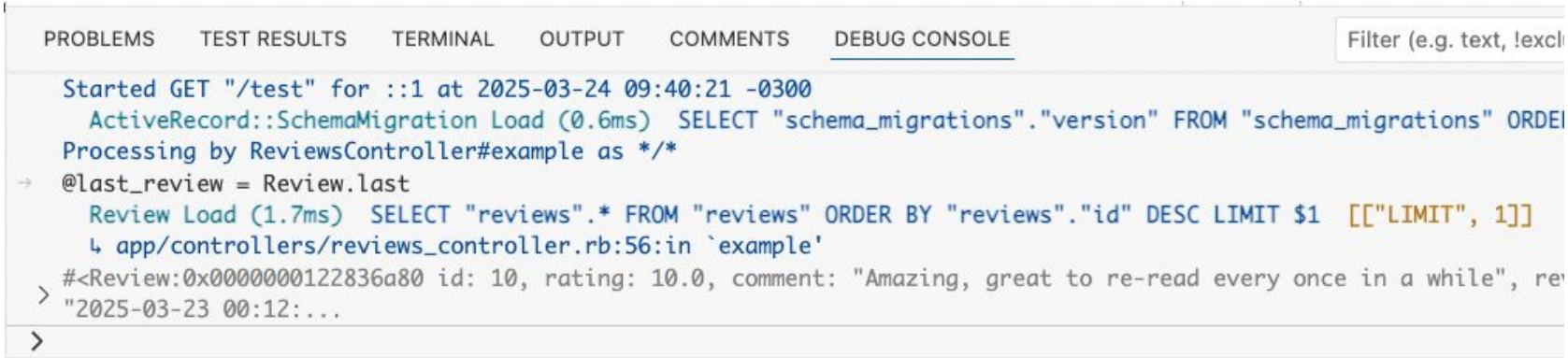
F5: siguiente breakpoint

F10: siguiente línea (Step Over)

y más como F11, F12

Puedes usar la consola de debugeo

- Tiene el contexto de la línea en pausa
- Puedes revisar los valores de tus variables
- Crear nuevas variables
- Modificar el valor de las variables existentes



¡Mientras antes aprendes a usarlo, antes aprovechas sus ventajas!