CHEAT SHEETS HERO

Ruby on Rails Cheatsheet

A comprehensive cheat sheet covering essential Ruby on Rails commands, helpers, and best practices for efficient web development.

Rails Basics & Setup

Project Setup

`rails new my_app` - Create a new Rails application named 'my_app'.

`rails server` or `rails s` - Start the Rails server.

Access it via `http://localhost:3000`.

`rails console` or `rails c` - Start the Rails console for interacting with the application.

`rails db:create` - Create the database defined in `config/database.yml`.

`rails db:migrate` - Run pending database migrations.

`rails db:seed` - Load the seed data from `db/seeds.rb`.

Generators

`rails generate model ModelName attribute:type attribute2:type2 ...` - Generate a model with specified attributes and types.

Example: `rails generate model Product name:string price:decimal`

`rails generate controller ControllerName action1 action2 ...` - Generate a controller with specified actions.

Example: `rails generate controller Products index show new create edit update destroy`

`rails generate migration AddColumnToTable column:type` - Generate a migration to add a column to a table.

Example: `rails generate migration AddPriceToProducts price:decimal`

`rails generate resource ResourceName attribute:type ...` - Generate a model, controller, and routes for a resource.

Example: `rails generate resource Product name:string price:decimal`

Basic Commands

`rails routes` - List all defined routes in the application.

`rails test` - Run all tests.

`rails assets:precompile` - Precompile assets for production.

Models & Database

ActiveRecord Basics

`Model.all` - Retrieve all records from the table.

`Model.find(id)` - Find a record by its ID.

`Model.new(attributes)` - Create a new model instance.

`model.save` - Save the model instance to the database.

`model.update(attributes)` - Update the attributes of the model instance.

`model.destroy` - Delete the model instance from the database.

Associations

`has_one`	A model has one of another model. Example: `has_one :profile`
`belongs_to`	A model belongs to another model. Example: `belongs_to :user`
`has_many`	A model has many of another model. Example: `has_many :comments`
`has_many :through`	A model has many of another model through an association. Example: `has_many :appointments, through: :physician`

Validations

`validates :attribute, presence: true` - Ensures the attribute is present.

`validates :attribute, uniqueness: true` - Ensures the attribute is unique.

`validates :attribute, length: { minimum: 5, maximum: 20 }` - Validates the length of the attribute.

`validates :attribute, format: { with: /regex/ }` - Validates the format of the attribute using a regular expression.

`validates :attribute, numericality: true` - Ensures the attribute is a number.

Controllers & Views

Controller Actions

`index` - Display a list of all records.

`show` - Display a specific record.

`new` - Display a form to create a new record.

`create` - Create a new record.

`edit` - Display a form to edit an existing record.

`update` - Update an existing record.

`destroy` - Delete a record.

Views & Templates

ERB (Embedded Ruby) templates are used to generate HTML views.

`<%= @variable %>` - Output the value of a variable.

`<% code %>` - Execute Ruby code.

`<%= link_to 'Link Text', path %>` - Create a link to a specified path.

`<%= form_with(model: @model) do |form| %> ... <% end %>` - Create a form for a model.

Layouts & Partials

Layouts provide a consistent look and feel across multiple pages.

Partials are reusable view templates.

`<%= render 'partial_name' %>` - Render a partial.

Use `yield` in layouts to insert content from views.

Routing & Assets

Routes

`get 'path', to: 'controller#action'` - Define a GET route.

`post 'path', to: 'controller#action'` - Define a POST route.

`resource :resource_name` - Define RESTful routes for a resource.

`resources :resource_name` - Define multiple RESTful routes for a resource.

`root 'controller#action'` - Define the root route.

Asset Pipeline

The asset pipeline manages CSS, JavaScript, and image assets.

Assets are located in the `app/assets` directory.

Use Sprockets directives (e.g., `require`, `require_tree`) to manage asset dependencies.

`<%= asset_path('image.png') %>` - Generate the path to an asset.

Helpers

`number_to_currency(number)` - Formats a number as currency.

`date.strftime('%m/%d/%Y')` - Format a date.

`time_ago_in_words(time)` - Show how long ago a time was.

`pluralize(count, 'item')` - Pluralize a word based on the count.