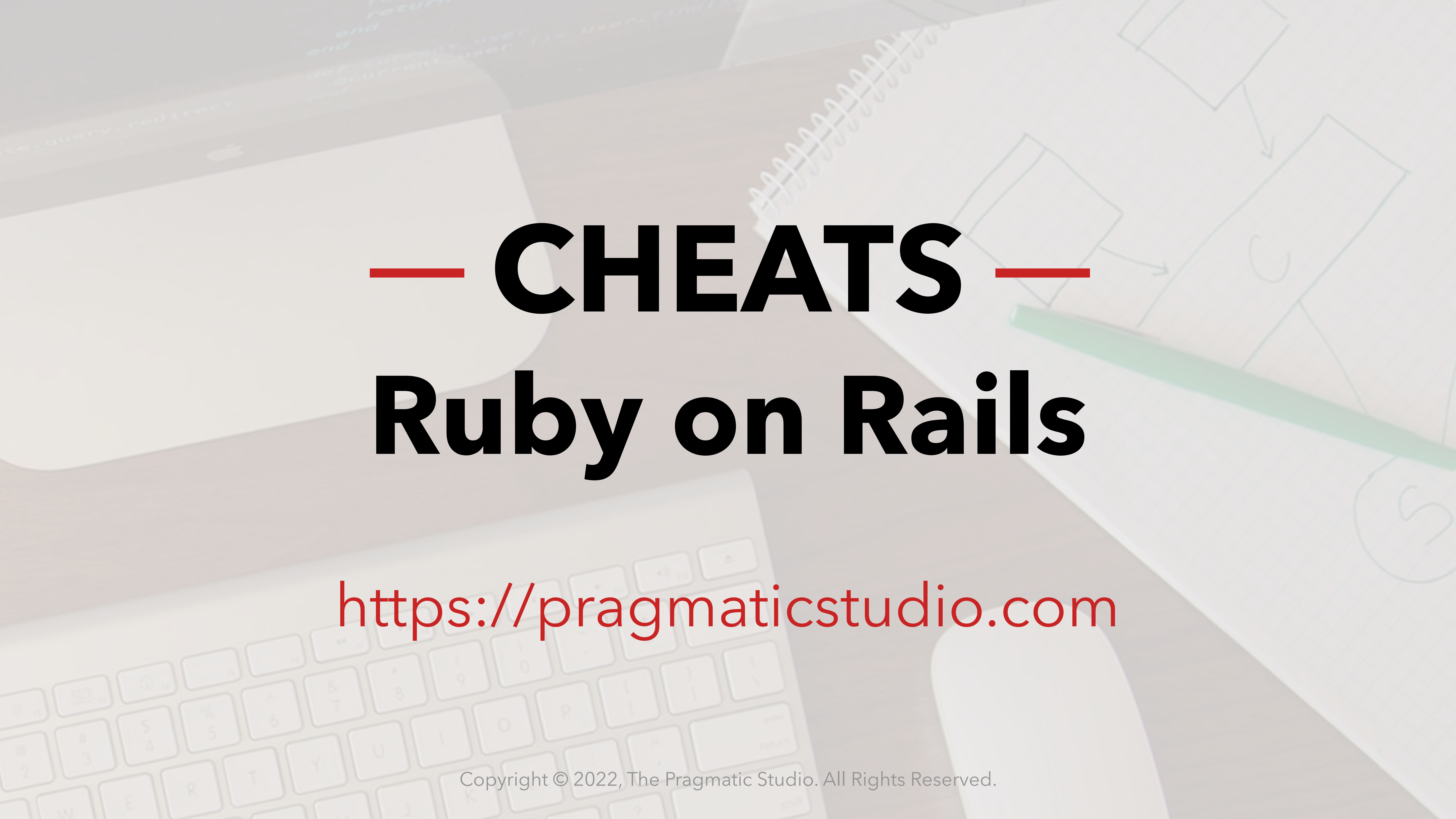


— CHEATS —

Ruby on Rails

https://pragmaticstudio.com

Create a New Rails App

```
rails new app_name
```

creates a new Rails application with a given name

example: rails new events

```
rails server (or rails s)
```

starts a web server on http://localhost:3000

```
rails console (or rails c)
```

starts a Rails console session

```
bundle install
```

installs all required gems and their dependencies

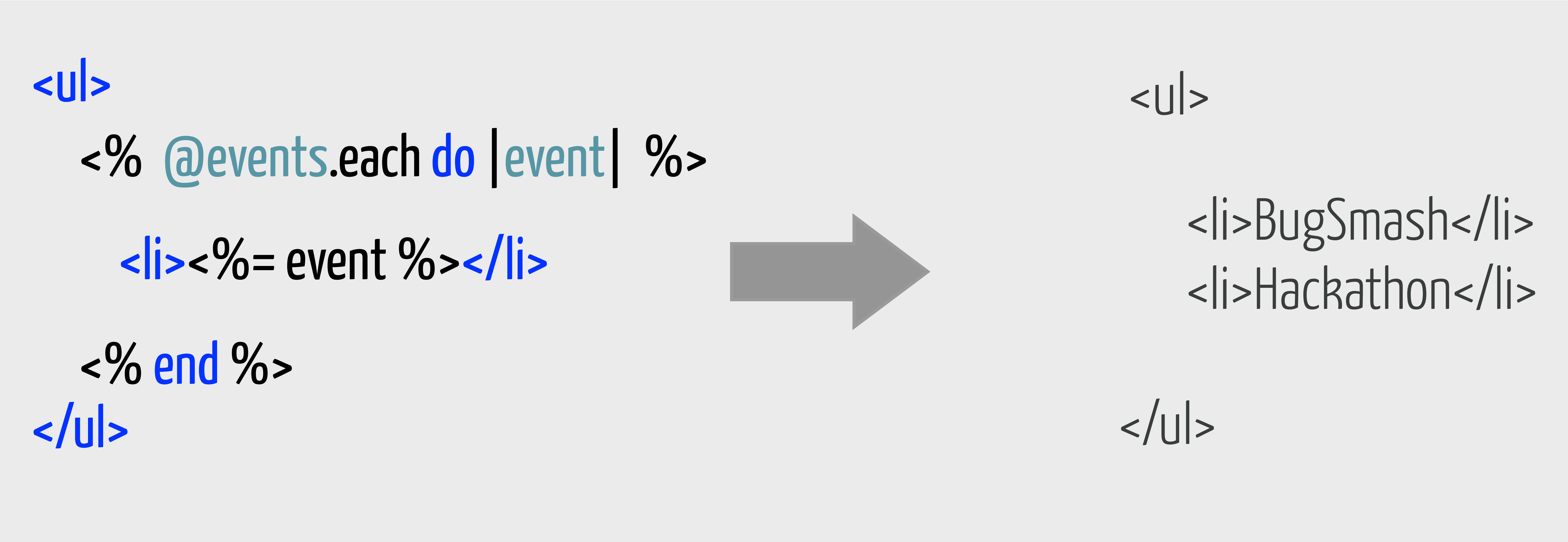
```
code .
```

opens the Visual Studio Code editor for the current working directory

ERb Tags

`<%=   %>` runs the Ruby code and **substitutes** result into template

`<%   %>` runs the Ruby code but **does not substitute** result into template



```
<ul>
  <% @events.each do |event| %>

    <li><%= event %></li>

  <% end %>
</ul>
```

```
<ul>

    <li>BugSmash</li>
    <li>Hackathon</li>

</ul>
```

Create a Controller Class

```
rails generate (or g) controller name_of_controller
```

creates a controller with the given name

example: rails g controller events

* controller names are always **plural**

```
rails destroy controller name_of_controller
```

cleans up what got generated

example: rails destroy controller events

Create a Model and Migration

```
rails g model name_of_model field:type field:type...
```

creates a model with the given name and a migration for creating a database table with the specified fields (columns) and types

example: rails g model event name:string location:string price:decimal

* model names are always **singular**

Migrations

The Pragmatic Studio

```
rails g migration name_of_migration field:type field:type...
```

creates a migration with a given name along with the specified fields and types

example: rails g migration AddFieldsToEvents starts_at:datetime description:text

tip: Name using the convention AddXXXToYYY to add the specified columns to the YYY table

```
rails db:migrate
```

runs any pending migration files in the db/migrate directory

```
rails db:migrate:status
```

checks the status of all migrations

```
rails db:rollback
```

reverses the last applied migration

```
rails db:migrate VERSION=XXXX
```

rolls back to any previous migration version

Common Migration Methods

```
create_table(name, options={}) do |t|
  t.column_type :column_name, options...
end
```

creates a new table and adds column definitions

```
change_table(table_name, options = {})
```

uses a block to change columns in the table

```
drop_table(name)
```

drops the table

```
rename_table(table_name, new_name)
```

renames a table

```
add_column(table_name, column_name, type, options = {})
```

adds a new column to the table

```
remove_column(table_name, column_name)
```

removes the column from the table

Column Types

```
:binary
:boolean
:date
:datetime
:decimal
:float
:integer
:primary_key
:string
:text
:time
:timestamp
```

Common Migration Methods (cont.)

`change_column(table_name, column_name, type, options = {})`
changes the column's definition according to the options
`rename_column(table_name, column_name, new_column_name)`
renames a column in the table
`add_index(table_name, column_name, options = {})`
adds a new index to the table
`remove_index(table_name, options = {})`
removes the given index from the table
`execute(sql)`
executes arbitrary SQL statements

Common Tasks

`rails –T`
see a list of all tasks
`rails –T db`
see a list of only the database-specific tasks
`rails routes`
see a list of all defined routes
`rails db:seed`
loads the seed data defined in db/seeds.rb
`rails db:reset`
drops and re-creates the database, applies all migrations, and loads the seed data defined in db/seeds.rb

Generate a Resource

```
rails g resource name_of_resource field:type field:type......
```

generates a resource with the given name along with fields and types

example: rails g resource registration name:string email:string how_heard:string event:references

generates:

- a migration for creating the registrations database table
- a Registration model with a belongs_to declaration
- an empty RegistrationsController
- a set of resource routes

Define Resource Routes

```
resources :name_of_resource
```

dynamically defines all the routes for a resource

example: resources :events

generates: 8 defined routes

Helper	HTTP Verb	URL Pattern	Controller#Action
events_path	GET	/events	events#index
event_path	GET	/events/:id	events#show
edit_event_path	GET	/events/:id/edit	events#edit
	PATCH	/events/:id	events#update
	PUT	/events/:id	events#update
	POST	/events	events#create
new_event_path	GET	/events/new	events#new
	DELETE	/events/:id	events#destroy

REST vs. SQL Actions

Actions	show	create	update	destroy
SQL	select	create	update	delete
REST	get	post	patch	delete

Render a Partial

Step 1

Extract common view code into a partial template

partial file names are always prefixed with an underscore

example: _form.html.erb

example: app/views/layouts/_header.html.erb

Step 2

```
<%= render “name of partial without underscore” %>
```

render the partial from another view

example: <%= render ‘form’ %>

example: <%= render ‘layouts/header’ %>

Frequently Used Built-In View Helpers

`truncate(text, options = {})`
truncates the given text to a default of 30 characters
`pluralize(count, singular, plural = nil)`
pluralizes the singular word unless the count is 1
`number_to_currency(number, options = {})`
1234567890.50 => $1,234,567,890.50
`number_to_percentage(number, options = {})`
100 => 100.000%
`number_to_phone(number, options = {})`
3035551212 => 303-555-1212
`time_ago_in_words(from_time)`
Time.now + 50.minutes => “about 1 hour”
`content_tag(name, content, options = {})`
returns a safe HTML tag of type name surrounding the content

Common Validation Methods

`validates :name, presence: true`
validates that the value of the specified attribute is not blank
`validates :description, length: { minimum: 25 }`
validates that the value of the specified attribute matches the specified length restrictions
`validates :price, numericality: { greater_than_or_equal_to: 0 }`
validates that the value of the specified attribute is numeric
`validates :email, format: { with: /(\S+)@(\S+)/ }`
validates that the value of the specified attribute has a format that matches the regular expression
`validates :password, confirmation: true`
validates that the value of the specified attribute matches a confirmation value

Common Validation Methods (cont.)

```
validates :how_heard, inclusion: { in: ["Blog", "Newsletter"] }
```

validates that the value of the specified attribute is available in the specified enumerable (array)

```
validates :role, exclusion: { in: ["Admin", "SuperUser"] }
```

validates that the value of the specified attribute is not available in the specified enumerable (array)

```
valid?
```

runs all the validations; automatically called when you try to create or save a model object

Populate the Database with Example Data

Step 1

Create records in the db/seeds.rb file

example:

```
Event.create!(
   name: 'BugSmash',
   location: 'Denver',
   price: 0.00,
   starts_at: 10.days.from_now,
   description: 'A fun evening of bug smashing!'
)
```

Step 2

```
rails db:seed
```

loads the seed data in db/seeds.rb into the current environment’s database

Adding RubyGems

Step 1

Add the gem (and optional version) to the Gemfile

```
gem 'bcrypt', '~> 3.1.7'

gem 'bootstrap', ‘~> 5.1’
```

Step 2

```
bundle install
```

installs any missing gems

Common Form Helper Methods

```
<%= form_for(@event) do |f| %>
```

generates a form for a single resource

```
<%= form_for([@event, @registration]) do |f| %>
```

generates a form for a nested resource

```
<%= f.label :name %>
```

returns a label tag for labelling an input field for the specified attribute

```
<%= f.text_field :name, autofocus: true %>
```

returns an input tag of the “text” type for accessing the specified attribute

```
<%= f.password_field :password %>
```

returns an input tag of the “password” type (masked) for accessing the specified attribute

Common Form Helper Methods (cont.)

`<%= f.password_field :password %>`
returns an input tag of the “password” type (masked) for accessing the specified attribute
`<%= f.text_area :description, cols: 40, rows: 7 %>`
returns a textarea tag set for accessing the specified attribute
`<%= f.number_field :price %>`
returns an input tag of type “number” for accessing the specified attribute
`<%= f.datetime_select :starts_at %>`
returns a set of select tags pre-selected for accessing the specified datetime-based attribute
`<%= f.select :how_heard, ["Blog", "Newsletter"] %>`
returns a select tag and a series of contained option tags for the specified attribute

Example has_many Methods

`event.registrations`
returns an array of the event's registrations (may be empty)
`event.registrations.new(attributes = {})`
instantiates a new registration for the event (event_id = event.id), but does not save it
`event.registrations.create(attributes = {})`
instantiates and saves a new registration for the event (event_id = event.id)
`event.registrations << registration`
associates the registration with the event (event_id = event.id) and saves the registration
`event.registrations.size`
returns the number of registrations associated with the event
`event.registrations.delete(registration)`
removes the registration from the event (if :dependent => :destroy, it will also delete the registration)
`event.registrations.clear`
removes all registrations from the event (if :dependent => :destroy, it will also delete the registrations)

Example belongs_to Methods

The Pragmatic Studio

`registration.event`
returns the event associated with the registration (may be nil)
`registration.event = an_event`
assigns the event to the registration

Example belongs_to Methods

Single Resource	Nested Resource
/events	/events/:event_id /registrations
events_path	event_registrations_path(event_id)
	/events/:event_id/registrations/new
new_event_path	new_event_registration_path(event_id)