



【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

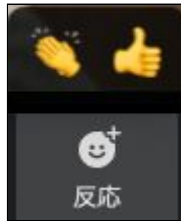
■ 1 - 0 : 講座を受ける前に

【ZOOMの使い方】

リアクションについて

ZOOM画面下部に「反応」というボタンがありますので、

- ・ 講師が皆さんの進捗を伺いますので、何も問題ない場合には、右側のいいねマーク 
- ・ 何かわからない点・つまづいた点がありましたら、左側の拍手マーク を押してください。



チャット機能について

わからないことがあれば、基本的にチャットで質問してください。

チャット機能で改行する方法

Windows → Shift + Enter.

Mac → control + Return (Enter)

質問の仕方について

下記のように、質問内容・入力したコマンドの行全体・出力結果をチャットしてくださると助かります。
ex)

下記のエラーが出ます。どうすればいいですか？

```
ec2-user:~/environment/contact_app/techgym_rails_course01 (lesson4) $ git checkout lesson4
```

```
Already on 'lesson4'
```

```
Your branch is up-to-date with 'origin/lesson4'.
```

ミュートについて

基本的にはミュートしててください。

チャットでは、質問しづらい内容がありましたら、ミュートを解除し、発言してください。

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

1回目： / 分 2回目： / 分 3回目： / 分 4回目： / 分 5回目： / 分

サンプルソースの公開場所：https://github.com/techgymjp/techgym_rails_course01

☆ 実行環境はCloud9(<https://aws.amazon.com/jp/cloud9/>)を使用する。

☆ 対象のgithubリポジトリをクローンする。

■ 1-0-1：実行環境を整えよう

【各種バージョン】

Rails: 5.2.4.2

Ruby: 2.6.3

Linux: Amazon Linux AMI release 2018.03

bundle: 1.17.3

【手順】

環境設定として下記のコマンド入力してください。

コマンドは、Terminalに下記図のように入力し、EnterキーまたはReturnキーを押してください。

※ \$マークは、すでに入力されているため、\$より後ろを入力してください。

コマンドを実行しても、何も表示されない場合がありますが、問題ありません。

```
ec2-user:~/environment $ mkdir techgym_rails
```

techgym_railsという名前のフォルダを作成する。

```
$ mkdir techgym_rails
```

techgym_railsフォルダに移動する。

```
$ cd techgym_rails
```

対象のgithubリポジトリをクローンする。

クローン： github上のプロジェクトをカレントディレクトリに複製する。

```
$ git clone https://github.com/techgymjp/techgym\_rails\_course01.git
```

techgym_rails_course01フォルダに移動する。

```
$ cd techgym_rails_course01
```

プロジェクトに必要なプログラムをインストールする。

```
$ bundle install --path vendor/bundle
```

※ postgresqlがエラーが発生した場合

```
An error occurred while installing pg (1.2.3), and Bundler cannot continue.  
Make sure that `gem install pg -v '1.2.3' --source 'https://rubygems.org/'` succeeds before bundling.
```

必要なパッケージをインストールする。

```
$ sudo yum install postgresql postgresql-server postgresql-devel postgresql-contrib -y
```

データベースの初期化

```
$ sudo service postgresql initdb
```

データベースサーバーの起動

```
$ sudo service postgresql start
```

```
$ bundle install --path vendor/bundle
```

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

データベースをセットアップする。

```
$ bundle exec rake db:setup
```

※ データベース作成時にpostgresqlのエラーが発生した場合

```
FATAL: role "ec2-user" does not exist
Couldn't create 'contact_app_development' database. Please check your configuration.
rake aborted!
```

ユーザーの作成

```
$ sudo -u postgres createuser -s ec2-user
```

```
$ bundle exec rake db:setup
```

※ 下記のエラーが発生した場合

```
Could not find public_suffix-4.0.4 in any of the sources
```

```
Run `bundle install` to install missing gems.
```

```
$ bundle install --path vendor/bundle
```

```
$ bundle exec rake db:setup
```

Railsのサーバーを起動する。

```
$ bundle exec rails server
```

【実行結果】

URL: /

ex) <https://f24e3029423e4xxxxxx38c8888d4.vfs.cloud9.ap-northeast-1.amazonaws.com/>

 お問い合わせ

下記フォームよりお問い合わせください。

件名

サービスについて

氏名

Eメール

内容

送信する

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

1回目： / 分 2回目： / 分 3回目： / 分 4回目： / 分 5回目： / 分

■ 1 - 1 : データを確認しよう : lesson1

【はじめに】

```
$ git checkout -b lesson1 remotes/origin/lesson1
```

【問題】

フォームから送信されたデータを表示しよう。

【修正する内容】

ファイル : app/controllers/contacts_controller.rb

メソッド : create

追加する機能 : フォームから送信されたデータを表示する。

【実行結果】

フォームを入力して「送信する」のボタンを押すと、下記のデータが表示されます。

```
{"title"=>"job", "name"=>"テックジム", "email"=>"test@test.test", "content"=>"テストです。"}
```

【ヒント】

- createメソッド内に `render plain: "hello, world!"` と記述すると、hello, world!と表示された画面が表示されます。
- フォームから送信された値はparamsという変数に格納されています。(createメソッド内)
- ContactController内のcontact_paramsはフォームから送信された値の内必要な値のみを取り出す。(ストロングパラメータ)

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

1回目： / 分 2回目： / 分 3回目： / 分 4回目： / 分 5回目： / 分

■ 1 - 2 : データを作成しよう : lesson2

【はじめに】

```
$ git add .
```

```
$ git commit -m "フォームのデータ確認"
```

```
$ git checkout -b lesson2 remotes/origin/lesson2
```

【問題】

フォームから送信されたデータを保存し、サンクスページに遷移させましょう。

※ サンクスページは元々で作成してあり、すでにルーティングを設定してある。

※ ルーティング設定箇所はconfig/routes.rbファイル内

【修正する内容】

ファイル : app/controllers/contacts_controller.rb

メソッド : create

追加する機能 : データを作成し、サンクスページへリダイレクトされる。

【実行結果】

サンクスページに遷移した後、下記画面が表示される

URL: /contacts/thanks

ex)

<https://f24e3029423e4xxxxxx38c8888d4.vfs.cloud9.ap-northeast-1.amazonaws.com/contacts/thanks>

お問い合わせいただきありがとうございます

確認メールを送信致しました。

2営業日以内に担当者からご連絡いたします。

お問い合わせフォームに戻る

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

【ヒント】

- `Contact.new()`は、引数にハッシュを受け取り、カラム名に対応した値が格納されたインスタンスを返す。
- `Contact.name`は、格納された値の内、`name`の値を返す。
- `p`関数 または `puts`関数を使用すると、rails serverコマンドで起動したコンソールに値が表示されます。
- `Contact.new()`によって返されたオブジェクトは`save()`メソッドが使える、`save()`メソッドはオブジェクトに格納されたデータを保存します。
- `redirect_to root_path` と`root`のPrefixを指定すると、トップページ(/)にリダイレクトされる。
- サンクスページのPrefixは、`/rails/info/routes`にアクセスすると確認することができる。

<https://f24e3029423e4xxxxxx38c8888d4.vfs.cloud9.ap-northeast-1.amazonaws.com/rails/info/routes>

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

1回目： / 分 2回目： / 分 3回目： / 分 4回目： / 分 5回目： / 分

■ 1 - 3 : データを確認しよう : lesson3

【はじめに】

```
$ git add .
```

```
$ git commit -m "フォームのデータ作成"
```

```
$ git checkout -b lesson3 remotes/origin/lesson3
```

【問題】

管理者画面を作成し、実際にデータが保存されているかどうかを確認しましょう。

【入力するコマンド】

AdminControllerを生成する。

```
$ bundle exec rails generate controller admin
```

→ app/controllers/admin_controller.rbというファイルが生成されます。

Admin::ContactsControllerをscaffoldで作成する。

```
$ bundle exec rails generate scaffold_controller admin/contacts title:enum name:string email:string content:text status:enum
```

※ 本プログラムではscaffoldのプログラムをカスタマイズしており、実際の挙動とは少し異なります。

→ app/controllers/adminとapp/views/adminというフォルダが生成され、各フォルダ内に必要なファイルが生成される。

【実行結果】

URL: /admin/contacts

<https://f24e3029423e4xxxxxx38c8888d4.vfs.cloud9.ap-northeast-1.amazonaws.com/admin/contacts>

お問い合わせ管理					
ID	件名	氏名	Eメール	内容	ステータス
1	お仕事のご相談・ご依頼について	テックジム	test@test.test	テストです。	未対応

新規作成表示編集削除

【番外】

□ \$ bundle exec rails destroy controller admin

```
$ bundle exec rails destroy scaffold_controller admin/contacts
```

というコマンドにより、生成されたファイルを削除することができます。

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

1回目 : / 分 2回目 : / 分 3回目 : / 分 4回目 : / 分 5回目 : / 分

■ 1 - 4 : 管理者画面に認証機能を実装しよう : lesson4

【はじめに】

```
$ git add .
```

```
$ git commit -m "管理者画面作成"
```

```
$ git checkout -b lesson4 remotes/origin/lesson4
```

【問題】

管理者画面を開いた際に、Basic認証が行われるようにしましょう。

【修正する内容】

ファイル : `app/controllers/admin_controller.rb`

メソッド : `basic`

追加する機能 : `AdminController`を継承したコントローラーを読み込む直後に、`basic`関数を呼び出しbasic認証をかける。 ※ 新しく`basic`という関数を`private`で作成してください。

【実行結果】

お問い合わせ管理ページにアクセスした際に、下記画像のようなポップアップが出現し、設定したユーザー名・パスワードを正しく入力すると、お問い合わせ管理者ページが見られる。

URL: `/admin/contacts`

`https://f24e3029423e4xxxxxx38c8888d4.vfs.cloud9.ap-northeast-1.amazonaws.com/admin/contacts`



ログイン

ユーザー名

パスワード

【テックジム】Railsコース 第1章 「問い合わせフォームを作ろう」

【ヒント】

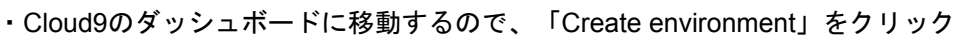
- `before_action :basic` と記述することでコントローラーを読み込んだ直後に`basic`関数を呼び出すことができます。
- 下記のように記述すると、入力したユーザー名とパスワードが、`rails server`コマンドで起動したコンソールにそれぞれの値が表示されます。また、`false`を記入することで、どんな値を入れても認証が通らない関数となり、`true`を記述すると必ず認証が通る関数になります。つまり、下記関数内で`user`と`pass`という変数を用い、論理演算を行います。

```
authenticate_or_request_with_http_basic do |user, pass|  
  p user  
  p pass  
  false  
end
```

回答はlesson5

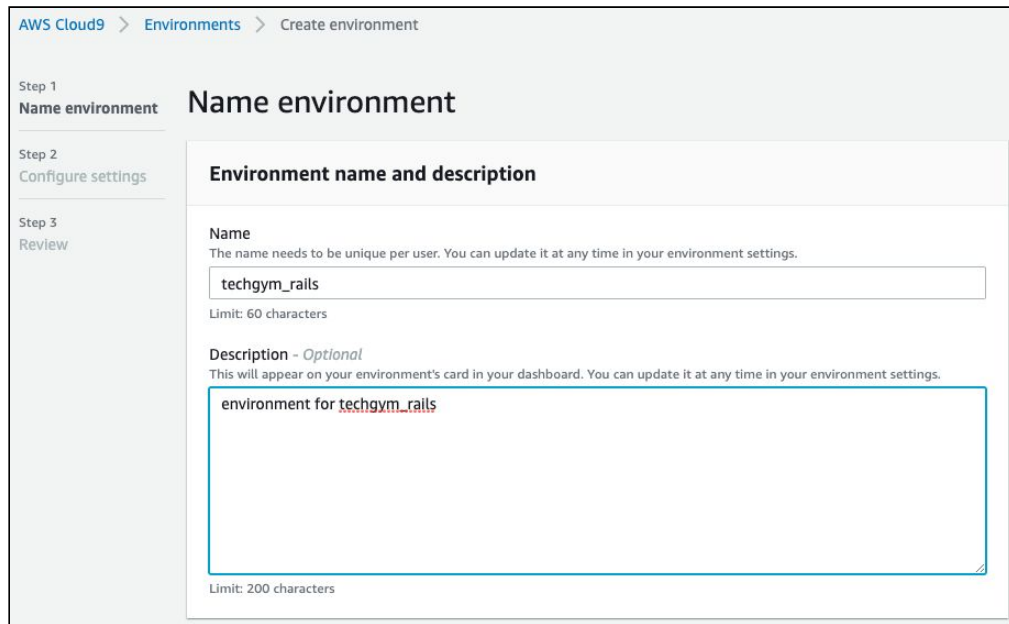
■ Cloud9の立ち上げ方

・AWS(<https://aws.amazon.com/jp/>)にログインして、フッターの「サービス」をクリックし、検索フォームにCloud9と入力してます。すると、「Cloud9」の項目が出てくるので、クリックしてください。



【テックジム】Railsコース 第1章 「問い合わせフォームを作ろう」

- ・ Step 1 「Name environment」では、好きな名前を入力し、任意で説明を入力してます。



AWS Cloud9 > Environments > Create environment

Step 1
Name environment

Step 2
Configure settings

Step 3
Review

Name environment

Environment name and description

Name
The name needs to be unique per user. You can update it at any time in your environment settings.

techgym_rails

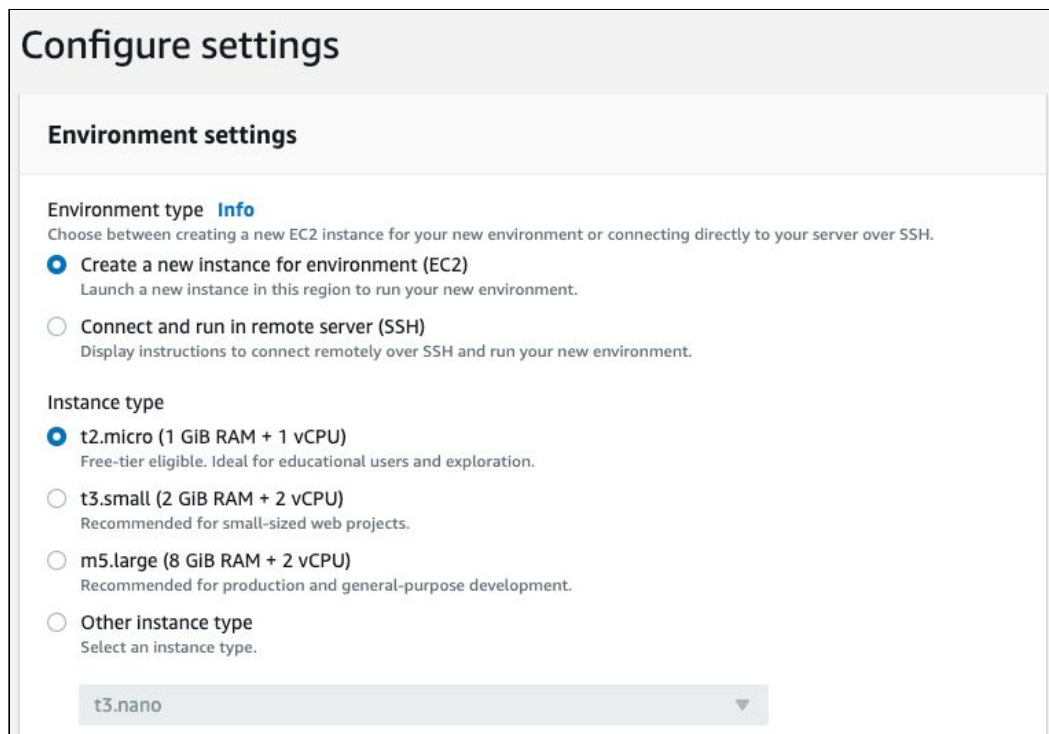
Limit: 60 characters

Description - Optional
This will appear on your environment's card in your dashboard. You can update it at any time in your environment settings.

environment for techgym_rails

Limit: 200 characters

- ・ Step 2 「Configure settings」では、下記の内容を選択し、「Next step」をクリックして下さい。
Environment type : Create a new instance for environment(EC2)
Instance type : t2.micro(1 GiB RAM + 1 vCPU)
Platform : Amazon Linux
Cost-saving setting : After 30 minutes (default)



Configure settings

Environment settings

Environment type [Info](#)
Choose between creating a new EC2 instance for your new environment or connecting directly to your server over SSH.

☒ Create a new instance for environment (EC2)
Launch a new instance in this region to run your new environment.

☐ Connect and run in remote server (SSH)
Display instructions to connect remotely over SSH and run your new environment.

Instance type

☒ t2.micro (1 GiB RAM + 1 vCPU)
Free-tier eligible. Ideal for educational users and exploration.

☐ t3.small (2 GiB RAM + 2 vCPU)
Recommended for small-sized web projects.

☐ m5.large (8 GiB RAM + 2 vCPU)
Recommended for production and general-purpose development.

☐ Other instance type
Select an instance type.

t3.nano ▼

【テックジム】Railsコース 第1章 「問い合わせフォームを作ろう」

Platform

☒ Amazon Linux


☐ Ubuntu Server 18.04 LTS

Cost-saving setting

Choose a predetermined amount of time to auto-hibernate your environment and prevent unnecessary charges. We recommend a hibernation settings of half an hour of no activity to maximize savings.

After 30 minutes (default) ▼

IAM role

AWS Cloud9 creates a service-linked role for you. This allows AWS Cloud9 to call other AWS services on your behalf. You can delete the role from the AWS IAM console once you no longer have any AWS Cloud9 environments. [Learn more](#) 

AWSServiceRoleForAWSCloud9

► **Network settings (advanced)**

No tags associated with the resource.

Add new tag

You can add 50 more tags.

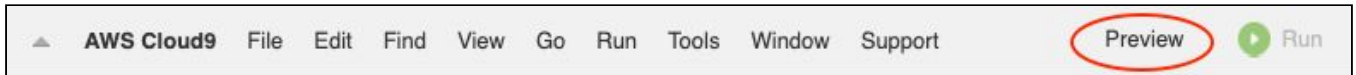
Cancel Previous step Next step

- ・ Step 3 「Review」 では、内容を確認し「Create environment」をクリックして下さい。

【テックジム】Railsコース 第1章 「問い合わせフォームを作ろう」

■ Cloud9でブラウザを立ち上げる

- ・ ページ上部の「Preview」をクリックし、「Preview Running Application」をクリック。



- ・ Cloud9の画面上で、仮想的なブラウザが表示されますので、ブラウザ上部のBrowserの右隣にあるボタンをクリックしてください。すると、新規ブラウザが表示され、bundle exec rails serverで立ち上げたページを表示することができます。



■ Oops VFS connection does not exist と表示された場合

ブラウザが問題を起こしている可能性が高いので、ブラウザを変えていただく(講師はChromeを使用しています)か、シークレットモードで再度AWS・Cloud9にログインしていただけますと、エラーがなくなると思います。

【テックジム】 Railsコース 第1章 「問い合わせフォームを作ろう」

■ gitについて

【前提知識】

- ・ 修正： gitではファイルを修正すると、自動で修正部分・新規追加ファイルを認識します。
- ・ コミット： いくつかの修正をひとまとまりにしたものです。
- ・ ブランチ： コミットを順番にまとめたものです。

【コマンド】

ブランチの一覧を表示する。

```
$ git branch
```

特定のブランチ(lesson1)に切り替える

```
$ git checkout lesson1
```

修正・新規ファイルの一覧を表示する。

```
$ git status
```

特定のファイル(app/controllers/contacts_controller.rb)をコミットできる状態にする。

```
$ git add app/controllers/contacts_controller.rb
```

カレントディレクトリ内の全てのファイルをコミットできる状態にする。

```
$ git add .
```

コミットできる状態にした修正・新規ファイルを名前(フォーム送信機能 追加)をつけてコミットする

```
$ git commit -m "フォーム送信機能 追加"
```

コミットを順番に表示する。

```
$ git log
```

特定のファイル(app/controllers/contacts_controller.rb)を修正する前の状態に戻す

```
$ git checkout app/controllers/contacts_controller.rb
```