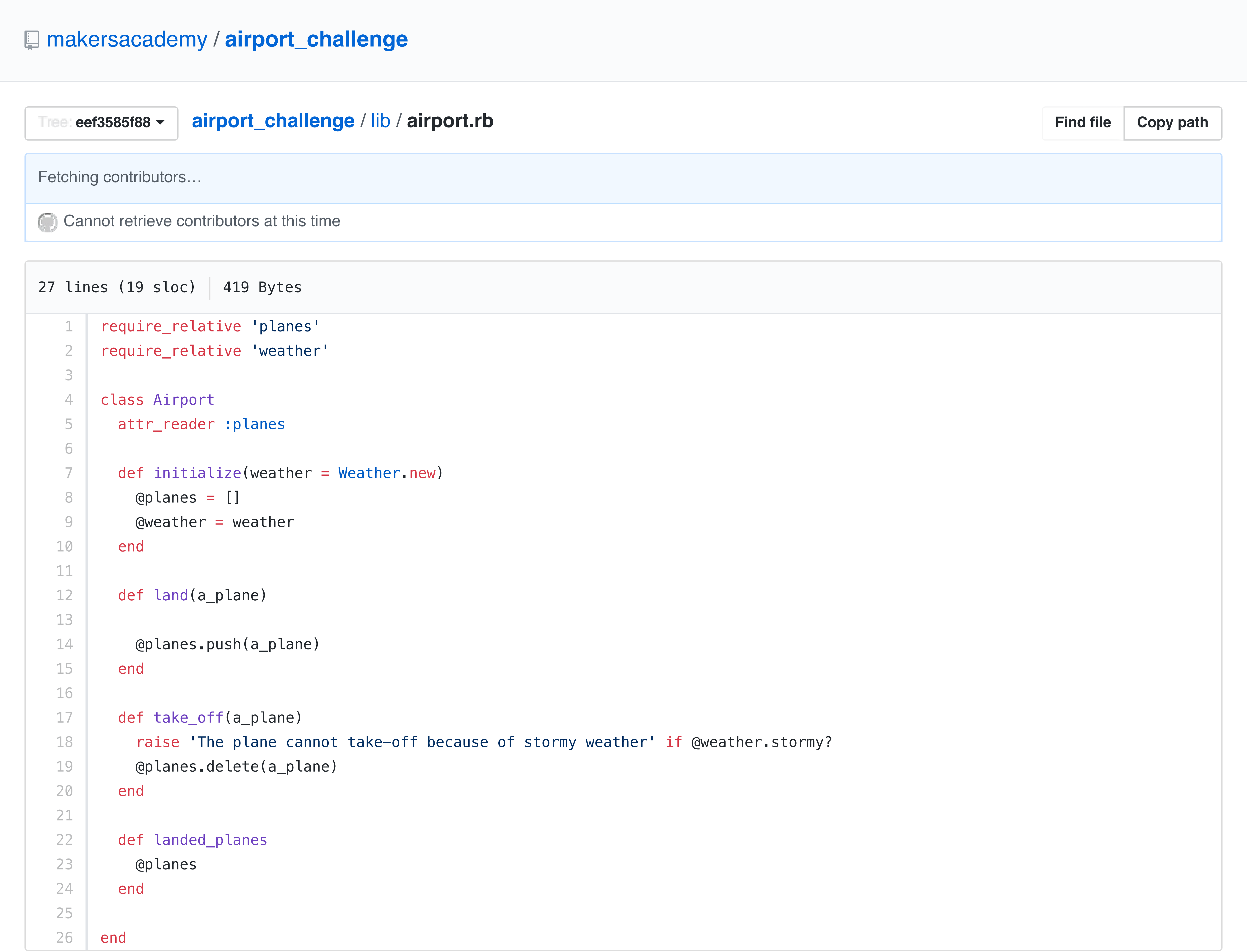


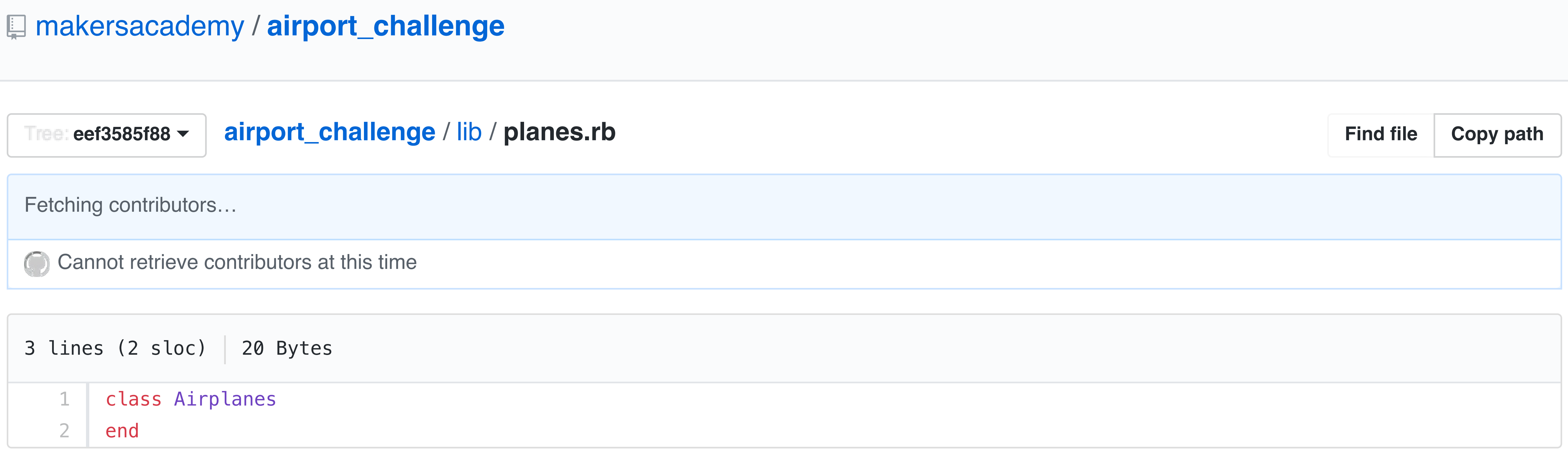
airport_challenge / lib / airport.rb

Fetching contributors…

Cannot retrieve contributors at this time

27 lines (19 sloc) | 419 Bytes

```ruby
require_relative 'planes'
require_relative 'weather'

class Airport
  attr_reader :planes

  def initialize(weather = Weather.new)
    @planes = []
    @weather = weather
  end

  def land(a_plane)

    @planes.push(a_plane)
  end

  def take_off(a_plane)
    raise 'The plane cannot take-off because of stormy weather' if @weather.stormy?
    @planes.delete(a_plane)
  end

  def landed_planes
    @planes
  end

end
```



makersacademy / airport_challenge

Tree: eef3585f88

airport_challenge / lib / planes.rb

Find file

Copy path

Fetching contributors…

Cannot retrieve contributors at this time

3 lines (2 sloc) | 20 Bytes

```ruby
class Airplanes
end
```

makersacademy / **airport_challenge**

Tree: eef3585f88 ▾ **airport_challenge** / lib / **weather.rb**

Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

17 lines (9 sloc) | 242 Bytes

```ruby


class Weather

conditions = ['sunny', 'overcast', 'raining', 'mild', 'stormy']


  def random_weather
    conditions = ['sunny', 'overcast', 'raining', 'mild', 'stormy'].sample
  end

  def stormy?
    random_weather == 'stormy'
  end

end
```

makersacademy / **airport_challenge**

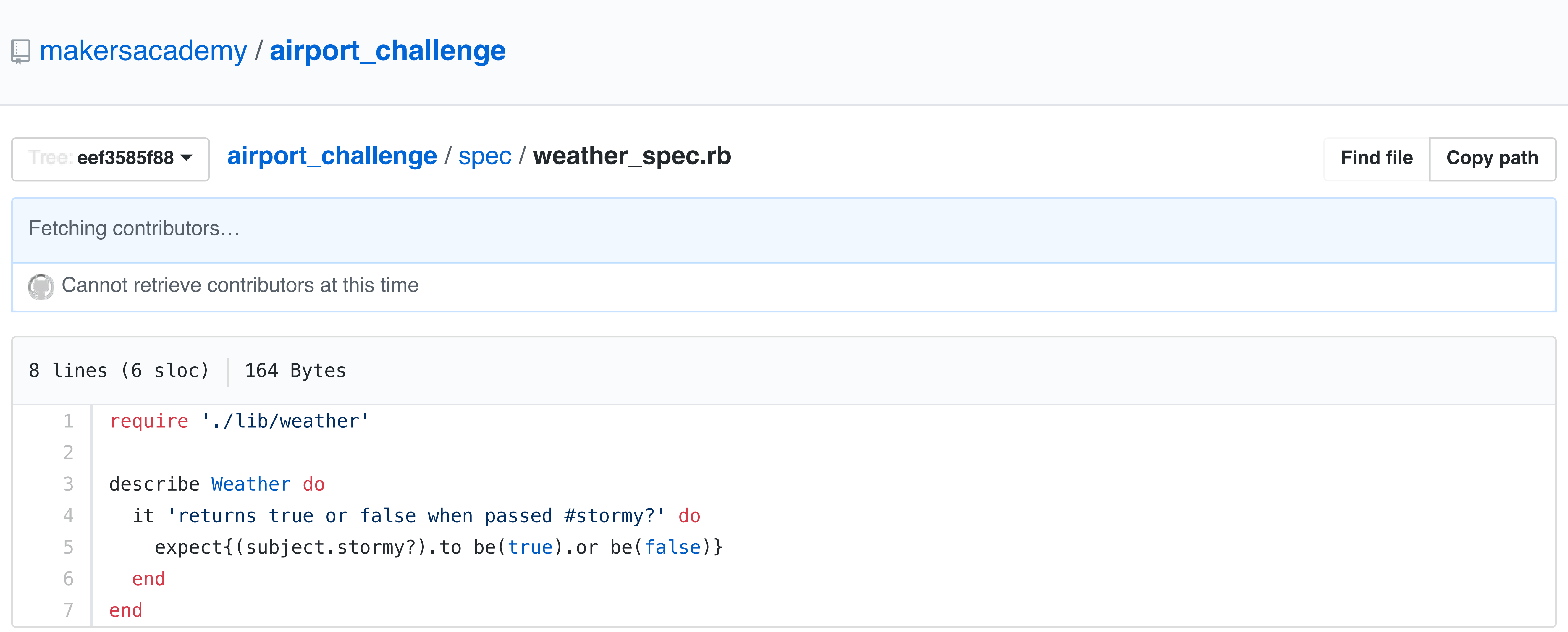
Tree: eef3585f88 ▾ **airport_challenge** / spec / **airport_spec.rb** Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

40 lines (27 sloc) | 956 Bytes

```
require './lib/airport'

describe Airport do

  describe 'Landing' do

    it 'should allow a plane to land at an airport' do
      expect{subject.to respond.to(:land).with(1).argument}
    end

    it 'should raise an error when the weather is too stormy to land' do
      expect{(weather.stormy?).to be(true)}
      plane = Airplanes.new
      expect{(subject.land(plane)).to raise_error('weather too stormy to land')}
    end
  end


  describe 'take_off' do

    it 'should allow a plane to take-off' do
      expect{subject.to respond.to(:take_off).with(1).argument}
    end

  end

    it 'should raise an error when the weather is stormy' do
      expect{(weather.stormy?).to be(true)}
      plane = Airplanes.new
      subject.land(plane)
      expect{(subject.take_off(plane)).to raise_error('Weather too stormy for take-off')}
    end

  it 'should see which planes are in the airport' do
    expect{subject(:landed_planes).to eq []}
  end


end
```



makersacademy / airport_challenge

airport_challenge / spec / weather_spec.rb

Fetching contributors…

Cannot retrieve contributors at this time

8 lines (6 sloc) | 164 Bytes

```ruby
require './lib/weather'

describe Weather do
  it 'returns true or false when passed #stormy?' do
    expect{(subject.stormy?).to be(true).or be(false)}
  end
end
```