makersacademy / **airport_challenge**

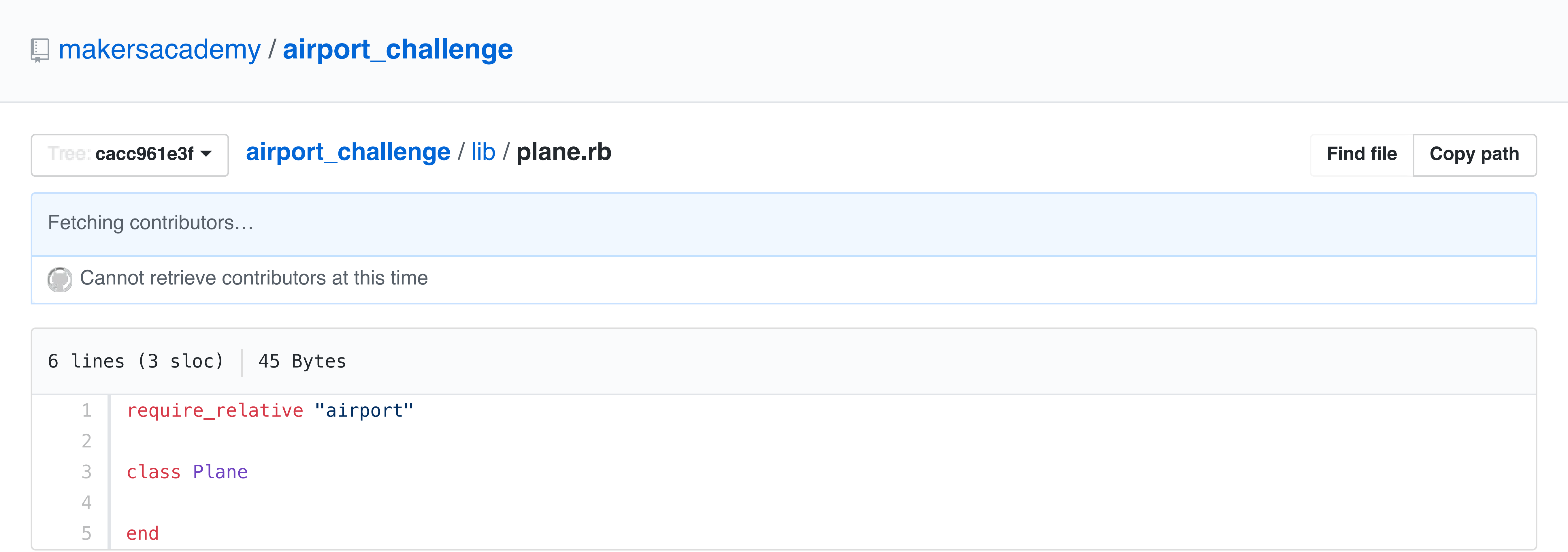
Tree: cacc961e3f ▾ **airport_challenge** / lib / **airport.rb** Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

40 lines (32 sloc) | 832 Bytes

```ruby
require_relative "plane"
require_relative "weather"

class Airport
  DEFAULT_CAPACITY = 10
  attr_reader :hangar
  attr_accessor :capacity

  def initialize
    @hangar = []
    @capacity = DEFAULT_CAPACITY
    @weather = Weather.new
  end

  def land(plane)
    fail "Plane already landed" if @hangar.include?(plane)
    fail "Airport at full capacity, plane unable to land" if full?
    fail "Stormy weather, plane unable to land" if stormy?
    @hangar << plane
    "Plane has landed"
  end

  def take_off(plane)
    fail "Plane already taken off" unless @hangar.include?(plane)
    fail "Stormy weather, plane unable to take off" if stormy?
    @hangar.delete(plane)
    "Plane has taken off"
  end

  protected

  def full?
    hangar.length >= @capacity
  end

  def stormy?
    @weather.current_weather == :stormy
  end
end
```



makersacademy / **airport_challenge**

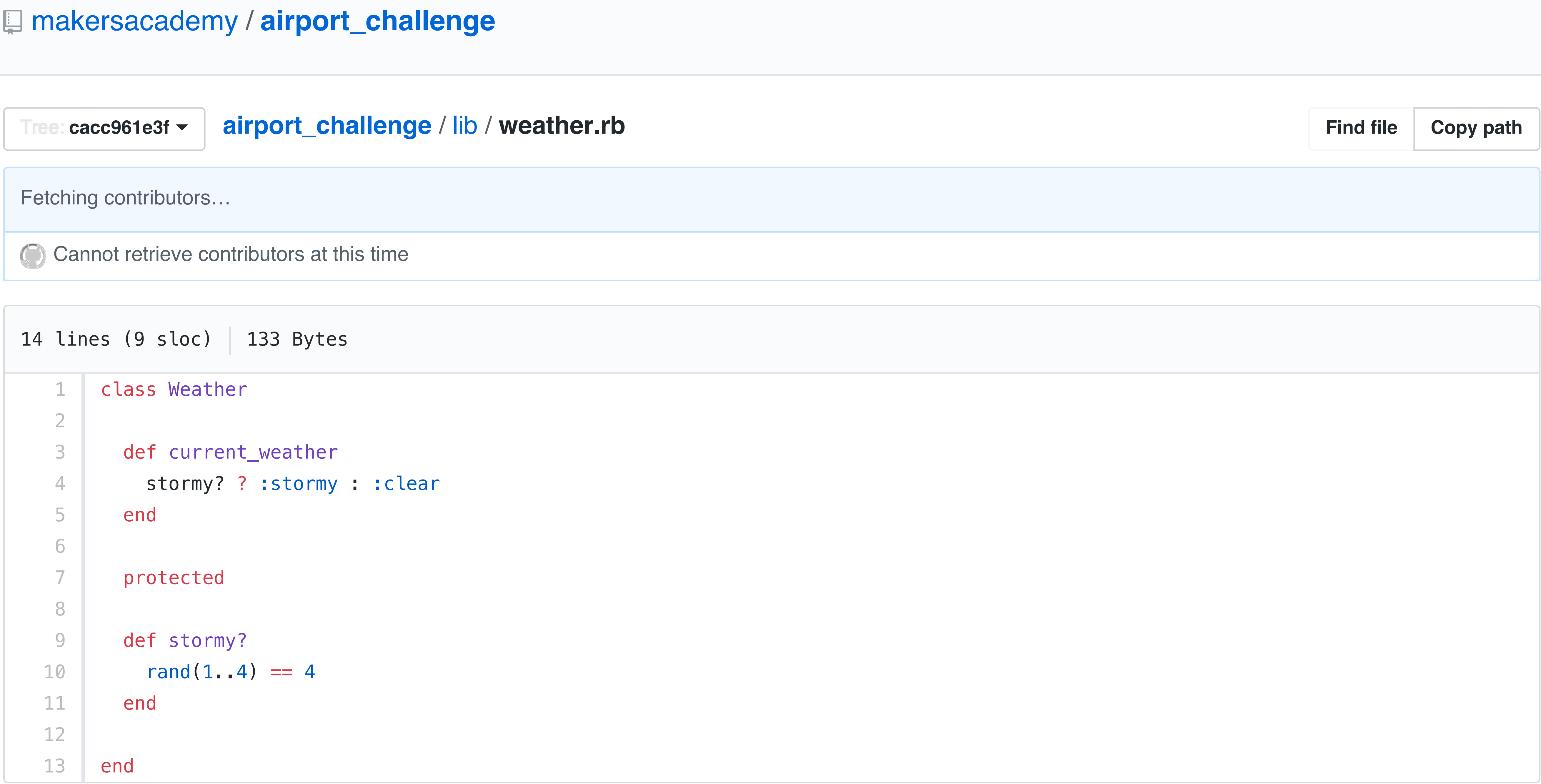
Tree: cacc961e3f ▾ **airport_challenge** / lib / **plane.rb**

Fetching contributors…

Cannot retrieve contributors at this time

6 lines (3 sloc) | 45 Bytes

```ruby
require_relative "airport"

class Plane

end
```



airport_challenge / lib / weather.rb

Fetching contributors…

Cannot retrieve contributors at this time

14 lines (9 sloc) | 133 Bytes

```ruby
class Weather

  def current_weather
    stormy? ? :stormy : :clear
  end

  protected

  def stormy?
    rand(1..4) == 4
  end

end
```

Tree: cacc961e3f ▾ airport_challenge / spec / airport_spec.rb

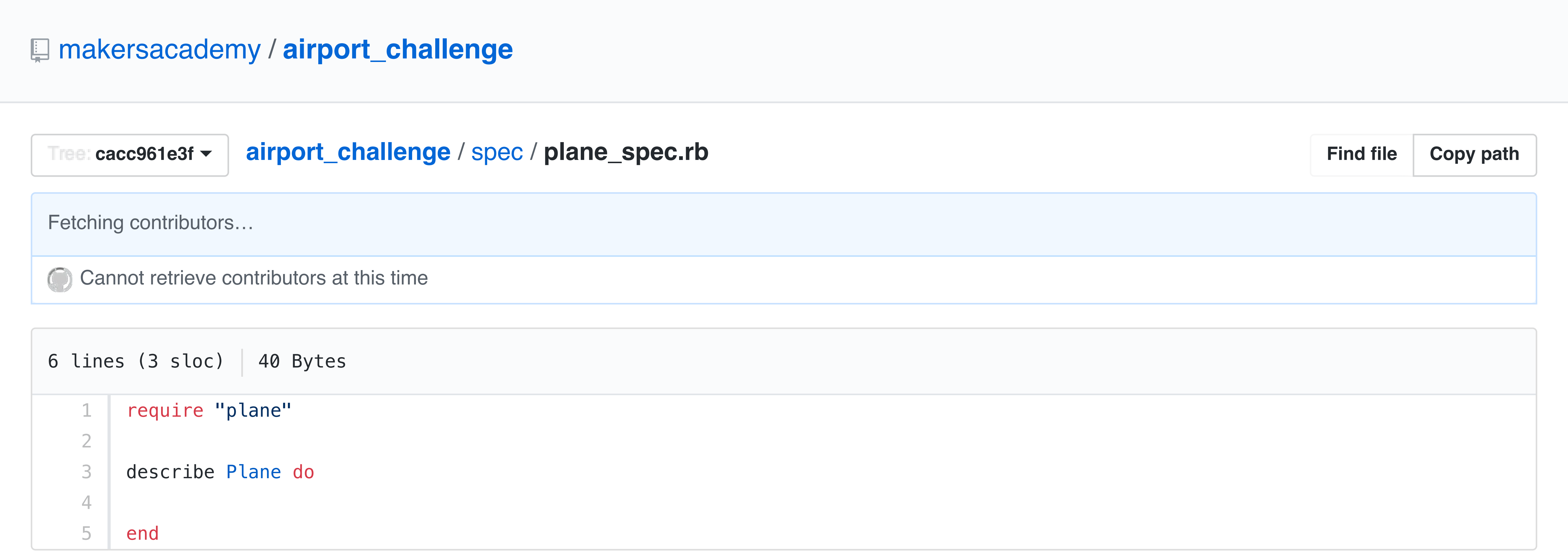
Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

69 lines (65 sloc) 2.41 KB

```ruby
require "airport"
require "weather"

describe Airport do
  plane = Plane.new
  before(:each) do
    allow(subject).to receive(:stormy?).and_return(false)
  end
  describe "#land" do
    it "allows the plane to land" do
      expect(subject.land(plane)).to eq "Plane has landed"
    end
    it "raises an error if plane has already landed" do
      subject.land(plane)
      expect { subject.land(plane) }.to raise_error "Plane already landed"
    end
    it "raises an error if plane cannot land because airport at default capacity" do
      Airport::DEFAULT_CAPACITY.times { subject.land(Plane.new) }
      message = "Airport at full capacity, plane unable to land"
      expect { subject.land(Plane.new) }.to raise_error message
    end
    it "checks if airport capacity is default capacity" do
      expect(subject.capacity).to eq Airport::DEFAULT_CAPACITY
    end
    it "sets the airport capacity to 5" do
      subject.capacity = 5
      expect(subject.capacity). to eq 5
    end
    it "raises an error if airport full with non-default capacity" do
      subject.capacity = 5
      subject.capacity.times { subject.land(Plane.new) }
      message = "Airport at full capacity, plane unable to land"
      expect { subject.land(Plane.new) }.to raise_error message
    end
    it "doesn't allow plane to land due to stormy weather" do
      allow(subject).to receive(:stormy?).and_return(true)
      message = "Stormy weather, plane unable to land"
      expect { subject.land(plane) }.to raise_error message
    end
  end

  describe "#take_off" do
    it "allows the plane to take off" do
      subject.land(plane)
      expect(subject.take_off(plane)).to eq "Plane has taken off"
    end
    it "raises an error if plane has already taken off" do
      subject.land(plane)
      subject.take_off(plane)
      expect { subject.take_off(plane) }.to raise_error "Plane already taken off"
    end
    it "confirms plane taken off is no longer in hangar" do
      subject.land(plane)
      subject.take_off(plane)
      expect(subject.hangar).not_to include plane
    end
    it "confirms plane is no longer in airport" do
      expect(subject.hangar).to be_empty
    end
    it "doesn't allow plane to take off due to stormy weather" do
      subject.land(plane)
      allow(subject).to receive(:stormy?).and_return(true)
      message = "Stormy weather, plane unable to take off"
      expect { subject.take_off(plane) }.to raise_error message
    end
  end

end
```



makersacademy / airport_challenge

airport_challenge / spec / plane_spec.rb

Fetching contributors…

Cannot retrieve contributors at this time

6 lines (3 sloc) | 40 Bytes

```ruby
require "plane"

describe Plane do

end
```

makersacademy / **airport_challenge**

Tree: cacc961e3f ▾ **airport_challenge** / spec / **weather_spec.rb** Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

16 lines (13 sloc) | 435 Bytes

```ruby
require "weather"

describe Weather do
  describe "#current_weather" do
    it "should return stormy weather when random value is 4" do
      allow(subject).to receive(:rand).and_return(4)
      expect(subject.current_weather).to eq :stormy
    end
    it "should return clear weather when random value is 1" do
      allow(subject).to receive(:rand).and_return(1)
      expect(subject.current_weather).to eq :clear
    end
  end

end
```