makersacademy / **airport_challenge**

Tree: facd73180c ▾ **airport_challenge** / lib / **airport.rb**

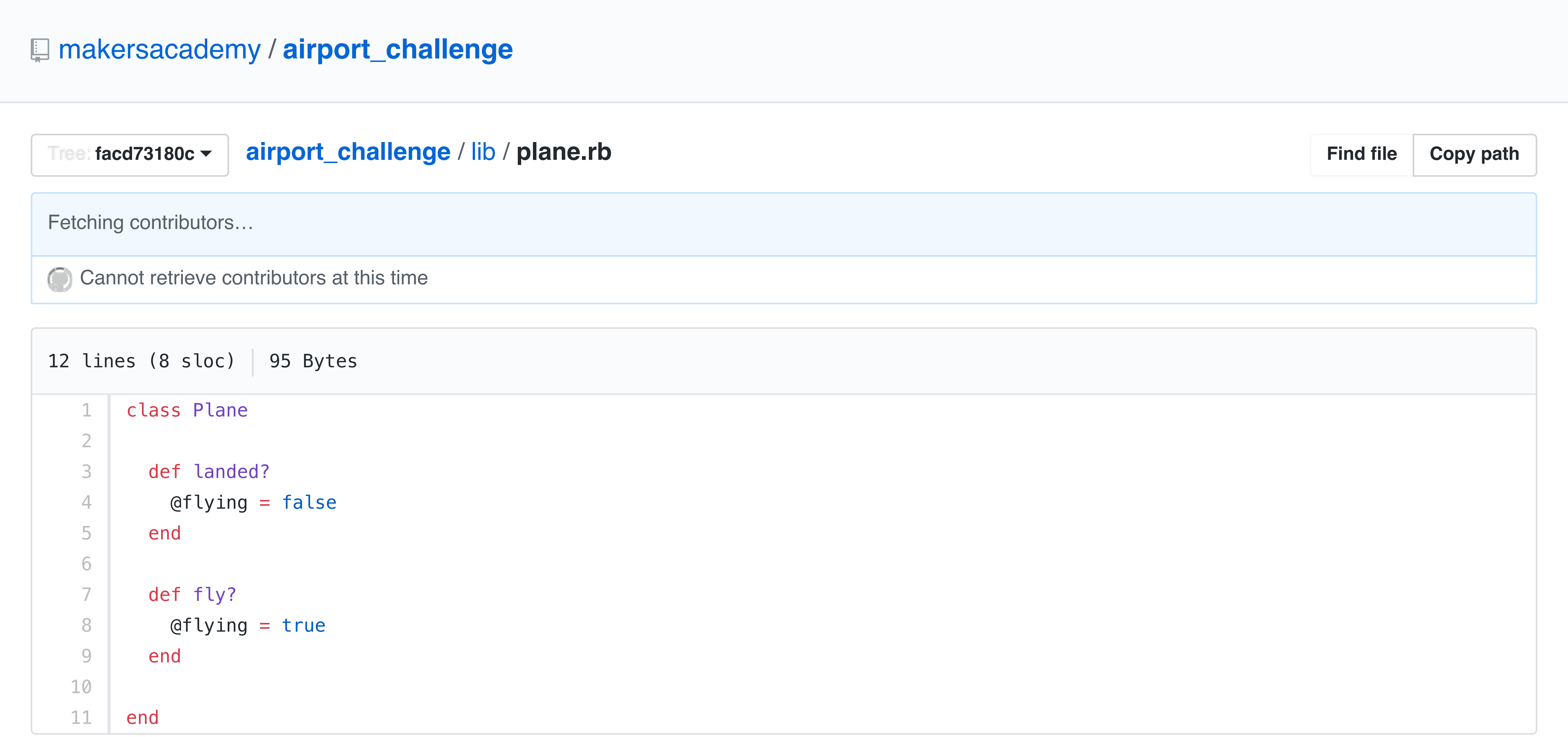
Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

37 lines (29 sloc) | 725 Bytes

```ruby
require './lib/plane.rb'
class Airport
  attr_reader :capacity
  attr_reader :planes_at_airport

  def initialize(capacity = 3)
    @capacity = capacity
    @planes_at_airport = []
  end

  def land_plane(plane)
    raise('Cannot land weather is Stormy') if stormy?
    raise('no more landing space') if airport_full?
    plane.landed?
    @planes_at_airport.push(plane)
  end

  def take_off(plane)
    raise('Cannot take off weather is Stormy') if stormy?
    plane.fly?
    @planes_at_airport.delete(plane)
  end

  def stormy?
    @stormy = weather
  end

  def weather
    ['Sunny', 'Rain', 'Snow', 'Stormy'] == 'Stormy' ? true : false
  end

  def airport_full?
    @planes_at_airport.length >= @capacity
  end

end
```



makersacademy / **airport_challenge**

airport_challenge / lib / **plane.rb**

Fetching contributors…

Cannot retrieve contributors at this time

12 lines (8 sloc) | 95 Bytes

```ruby
class Plane

  def landed?
    @flying = false
  end

  def fly?
    @flying = true
  end

end
```

makersacademy / **airport_challenge**

Tree: facd73180c ▾ **airport_challenge** / spec / **airport_spec.rb**

Find file | Copy path

Fetching contributors…

Cannot retrieve contributors at this time

38 lines (31 sloc) | 1.19 KB

```ruby
require "airport.rb"
require "plane.rb"
describe Airport do
  let(:airport) { Airport.new }
  let(:plane) { Plane.new }

  it 'should allow developer to change capacity when needed' do
    expect(Airport.new(5).capacity).to eq 5
  end

  describe '#land_plane(plane)' do
    it "should be able to land plane" do
      expect(subject.land_plane(plane)).to eq subject.planes_at_airport
    end

    it "should be able to prevent landing if weather is stormy" do
      allow(subject).to receive(:stormy?).and_return(true)
      expect { subject.land_plane(plane) }.to raise_error('Cannot land weather is Stormy')
    end

    it "should'nt be able to land planes if airport full" do
      subject.capacity.times { subject.land_plane(plane) }
      expect { subject.land_plane(plane) }.to raise_error('no more landing space')
    end
  end

  describe '#take_off(plane)' do
    it "should be able to tell plane to take off" do
      expect(subject.take_off(plane)).to eq nil
    end

    it "should be able to prevent take off if weather is stormy" do
      allow(subject).to receive(:stormy?).and_return(true)
      expect { subject.take_off(plane) }.to raise_error('Cannot take off weather is Stormy')
    end
  end
end
```

makersacademy / **airport_challenge**

Tree: facd73180c ▾ **airport_challenge** / spec / **plane_spec.rb** Find file Copy path

Fetching contributors…

Cannot retrieve contributors at this time

22 lines (18 sloc) | 449 Bytes

```ruby
require "airport.rb"
require "plane.rb"

describe Plane do
  let(:airport) { Airport.new }
  let(:plane) { Plane.new }

  describe '#landed?' do
    it 'should update status of plane when landed' do
      airport.land_plane(plane)
      expect(plane.landed?).to eq false
    end
  end

  describe '#fly?'do
    it 'should update status of plane when taking off' do
      airport.take_off(plane)
      expect(plane.fly?).to eq true
    end
  end
end
```