

Tree: **bcc3fdf072** ▾

airport_challenge / **feature_test_examples.rb**

Find file

Copy path


Fetching contributors...

 Cannot retrieve contributors at this time

34 lines (32 sloc) | 1.1 KB

```
1  require_relative "lib/airport.rb"
2
3  p "feature 1- ability to land a plane at an airport"
4  p "creating airport: my_airport = Airport.new"
5  p my_airport = Airport.new
6  p "creating plane"
7  p my_plane = Plane.new
8  p "land plane at airport: my_plane = Plane.new"
9  p my_airport.land my_plane
10 p "\n"
11 p "feature 2- ability to check which planes are currently in an airport"
12 p "list planes at airport: my_airport.planes"
13 p my_airport.planes
14 p "\n"
15 p "feature 3 -planes take off from airport"
16 p "release plane from airport: my_airport.release_for_takeoff my_plane"
17 p my_airport.release_for_takeoff my_plane
18 p "\n"
19 p "feature 4 -planes keep track of where they are"
20 p "get location of plane: my_plane.location"
21 p my_plane.location
22 p "\n"
23 p "for robustness planes can also be landed using the plane.land airport syntax without error"
24 p "my_plane.land my_airport"
25 p my_plane.land my_airport
26 p "my_airport.planes"
27 p my_airport.planes
28 p "\n"
29 p "likewise, planes can also be told to take off and will co-ordinate this with the airport"
30 p "my_plane.take_off"
31 p my_plane.take_off
32 p "my_airport.planes"
33 p my_airport.planes
```

Fetching contributors...

 Cannot retrieve contributors at this time

54 lines (44 sloc) | 1.08 KB

1

require_relative "plane"

2

require_relative "weather"

3

class Airport

4

DEFAULT_CAPACITY = 50

5

attr_reader :planes, :capacity

6

attr_writer :weather

7

8

def initialize(capacity: DEFAULT_CAPACITY, weather: Weather.new)

9

@planes = []

10

@capacity = capacity

11

@weather = weather

12

end

13

14

def land plane

15

raise "Airport Full, Cannot Land" if full?

16

raise "It's not safe to land at the moment!" unless safe_to_fly?

17

@expecting = plane

18

plane.land self

19

@planes << plane

20

@expecting = nil

21

end

22

23

def release_for_takeoff plane

24

raise "Plane Not Here" unless plane_in_airport? plane

25

raise "It's not safe to take off at the moment!" unless safe_to_fly?

26

@released_for_takeoff = plane

27

plane.take_off

28

@released_for_takeoff = nil

29

@planes.delete plane

30

end

31

32

def expecting? plane

33

@expecting == plane

34

end

35

36

def released_for_takeoff? plane

37

@released_for_takeoff == plane

38

end

39

40

private

41

42

def safe_to_fly?

43

@weather.conditions != :Stormy

44

end

45

46

def full?

47

planes.count == capacity

48

end

49

50

def plane_in_airport? plane

51

@planes.include? plane

52

end

53

end

Tree: **bcc3fdf072** ▾

airport_challenge / **lib** / **plane.rb**

Find file

Copy path

Fetching contributors...

 Cannot retrieve contributors at this time

32 lines (26 sloc) | 518 Bytes

```
1  class Plane
2    attr_reader :location
3
4    def initialize
5      @location = :Flying
6    end
7
8    def land airport
9      raise "Already Landed" unless flying?
10     if (airport.expecting? self) then
11       @location = airport
12     else
13       airport.land self
14     end
15   end
16
17   def take_off
18     raise "Already In Flight" if flying?
19     if (@location.released_for_takeoff? self) then
20       @location = :Flying
21     else
22       @location.release_for_takeoff self
23     end
24   end
25
26   private
27
28   def flying?
29     @location == :Flying
30   end
31 end
```

Tree: bcc3fdf072 ▼

airport_challenge / lib / weather.rb

Find file

Copy path


Fetching contributors...

 Cannot retrieve contributors at this time

14 lines (9 sloc) | 155 Bytes

```
1  class Weather
2
3    def conditions
4      rand_conditions
5    end
6
7    private
8
9    def rand_conditions
10     @conditions = rand(10) == 1 ? :Stormy : :Sunny
11   end
12
13 end
```

Fetching contributors...

 Cannot retrieve contributors at this time

69 lines (61 sloc) | 2.26 KB

1

require 'airport'

2

describe Airport do

3

4

let :plane {double(:plane, land: nil, take_off: nil)}

5

let :weather {double(:weather, conditions: :Sunny)}

6

subject {Airport.new(weather: weather)}

7

8

describe "#planes" do

9

it "should return a list planes in the airport" do

10

subject.land plane

11

subject.land plane

12

expect(subject.planes).to eq [plane, plane]

13

end

14

end

15

16

describe "#weather=" do

17

it "should be possible to set the weather" do

18

expect(subject.weather = weather).to eq weather

19

end

20

end

21

22

describe "#land" do

23

it "should be able to receive a plane" do

24

expect(subject).to respond_to(:land).with(1).argument

25

end

26

it "shouldn't be able to land any planes if the airport is full" do

27

subject.capacity.times {subject.land plane}

28

expect {subject.land(plane)}.to raise_error "Airport Full, Cannot Land"

29

end

30

it "shouldn't be able to land planes in story weather" do

31

allow(weather).to receive(:conditions) {:Stormy}

32

expect {subject.land(plane)}.to raise_error "It's not safe to land at the moment!"

33

end

34

end

35

36

describe "#release_for_takeoff" do

37

it "should return a plane, and the plane should no longer be in the airport" do

38

subject.land plane

39

expect(subject.release_for_takeoff plane).to eq plane

40

expect(subject.planes).to eq []

41

end

42

it "should cause the plane to take off" do

43

subject.land plane

44

expect(plane).to receive(:take_off)

45

subject.release_for_takeoff plane

46

end

47

it "should raise an error if trying to release a plane for takeoff that is not in the airport" do

48

expect{subject.release_for_takeoff plane}.to raise_error "Plane Not Here"

49

end

50

it "should not allow planes to take off when it is stormy" do

51

subject.land plane

52

allow(weather).to receive(:conditions) {:Stormy}

53

expect{subject.release_for_takeoff plane}.to raise_error "It's not safe to take off at the moment!"

54

end

55

end

56

57

describe "#expecting?" do

58

it "should return false when a plane is not expected for landing" do

59

expect(subject.expecting? plane).to eq false

60

end

61

end

62

63

describe "#released_for_takeoff?" do

64

it "should return false when a plane has not been released for takeoff" do

65

expect(subject.released_for_takeoff? plane).to eq false

66

end

67

end

68

end


Tree: bcc3fdf072 ▾

airport_challenge / spec / plane_spec.rb

Find file

Copy path

Fetching contributors...

 Cannot retrieve contributors at this time

60 lines (53 sloc) | 1.82 KB

1

require "plane"

2

describe Plane do

3

let :airport {double(:airport, expecting?: true, released_for_takeoff?: true)}

4

5

describe "#location" do

6

it "should be flying if not in an airport" do

7

expect(subject.location).to eq :Flying

8

end

9

10

it "should return an airport if in one" do

11

subject.land airport

12

expect(subject.location).to eq airport

13

end

14

end

15

16

describe "#land" do

17

it "shouldn't be able to land if not flying" do

18

subject.land airport

19

expect {subject.land airport}.to raise_error "Already Landed"

20

end

21

it "should check that the plane is expected before trying to land" do

22

expect(airport).to receive(:expecting?).with(subject)

23

subject.land(airport)

24

end

25

it "should ask the airport to land if not expected" do

26

allow(airport).to receive(:expecting?) {false}

27

expect(airport).to receive(:land).with(subject)

28

subject.land(airport)

29

end

30

it "should be at the airport after landing" do

31

subject.land airport

32

expect(subject.location).to eq airport

33

end

34

end

35

36

describe "#take_off" do

37

it "shouldn't be able to take off if already flying" do

38

expect {subject.take_off}.to raise_error "Already In Flight"

39

end

40

it "should be flying after take_off" do

41

subject.land(airport)

42

subject.take_off

43

expect(subject.location).to eq :Flying

44

end

45

it "should check that it has been released for takeoff before taking off" do

46

subject.land(airport)

47

expect(airport).to receive(:released_for_takeoff?).with subject

48

subject.take_off

49

end

50

it "should call released_for_takeoff if not released for takeoff" do

51

allow(airport).to receive(:released_for_takeoff?) {false}

52

subject.land(airport)

53

expect(airport).to receive(:release_for_takeoff).with subject

54

subject.take_off

55

end

56

57

end

58

59

end

Tree: bcc3fdf072 ▾

airport_challenge / [spec](#) / **weather_spec.rb**

Find file

Copy path

Fetching contributors...



Cannot retrieve contributors at this time

10 lines (8 sloc) | 197 Bytes

```
1  require "weather"
2
3  describe Weather do
4    describe "#conditions" do
5      it "returns current weather conditions" do
6        expect(subject.conditions).to eq(:Sunny).or(eq(:Stormy))
7      end
8    end
9  end
```