



Joining up the dots

# SYNTHESIS

Stuart Caborn  
George Malamidis

© 2008 Stuart Caborn, George Malamidis



# “Synthesis”

the formation of something complex or coherent by combining simpler things

# Synthesized testing

Combine lightweight tests to build confidence that our system is complete and reduce the need for large, overarching tests



Test code is code, too



# Technical debt

# A story of shapes

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     # TODO - implement me
4   end
5 end
```

```
untitled
1 class ShapeShifter
2   def self.shift(*shapes)
3     really_complicated_shape_cooking(*shapes)
4   end
5 end
6
```

# Simple?

```
untitled 2
1 def test_makes_blue_star
2   assert_equal(:blue_star, ShapeExchange.blue_star)
3 end
```

# Hang on...

```
untitled 2
1 def test_makes_blue_star
2   # setup...
3   assert_equal(:blue_star, ShapeExchange.blue_star)
4 end
5
```



# Just a little setup

```
untitled 2
1 def test_makes_blue_star|
2   # setup...
3   connect_to_db
4
5   assert_equal(:blue_star, ShapeExchange.blue_star)
6 end
```

# Just a little more setup

```
untitled 2
1 def test_makes_blue_star
2   # setup...
3   connect_to_db
4   load_shape_definitions_from_sap
5
6   assert_equal(:blue_star, ShapeExchange.blue_star)
7 end
8
```

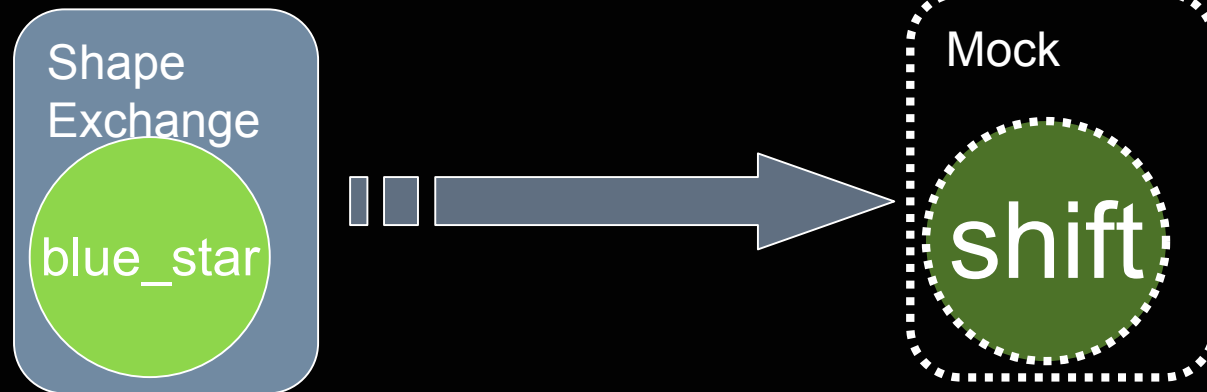
# You get the idea

```
untitled 2
1 def test_makes_blue_star
2   # setup...
3   connect_to_db
4   load_shape_definitions_from_sap
5   create_shape_dependency_1
6   #....
7
8   assert_equal(:blue_star, ShapeExchange.blue_star)
9 end
10
```



# Mock the interaction?

# Red



```
untitled 2
1 def test_makes_blue_star
2   ShapeShifter.expects(:shift)
3   ShapeExchange.blue_star
4 end
```

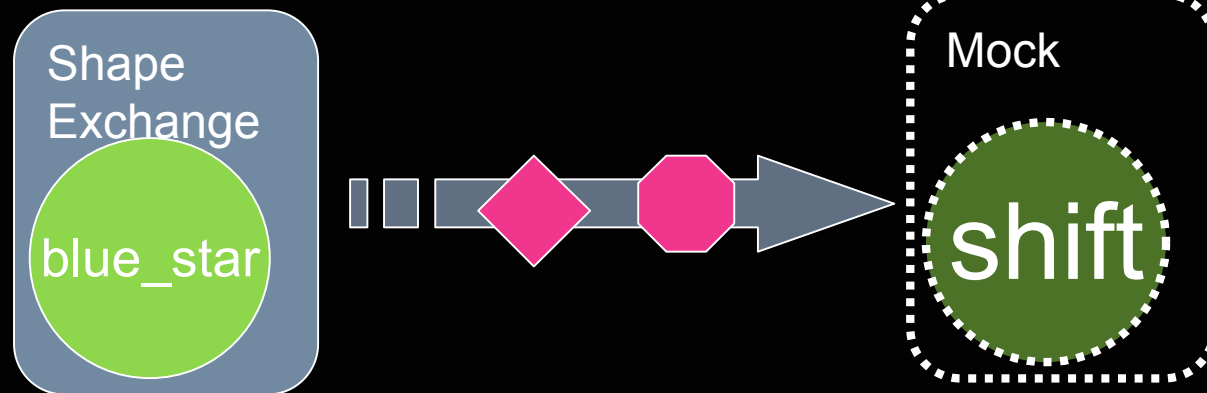
# Green

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     ShapeShifter.shift
4   end
5 end
6
7
```

# Crash

```
untitled
1 class ShapeShifter
2   def self.shift(*shapes)
3     really_complicated_shape_cooking(*shapes)
4   end
5 end
6
```

# Red



```
untitled 2
1 def test_makes_blue_star
2   ShapeShifter.expects(:shift).with(:diamond, :octagon)
3   ShapeExchange.blue_star
4 end
```



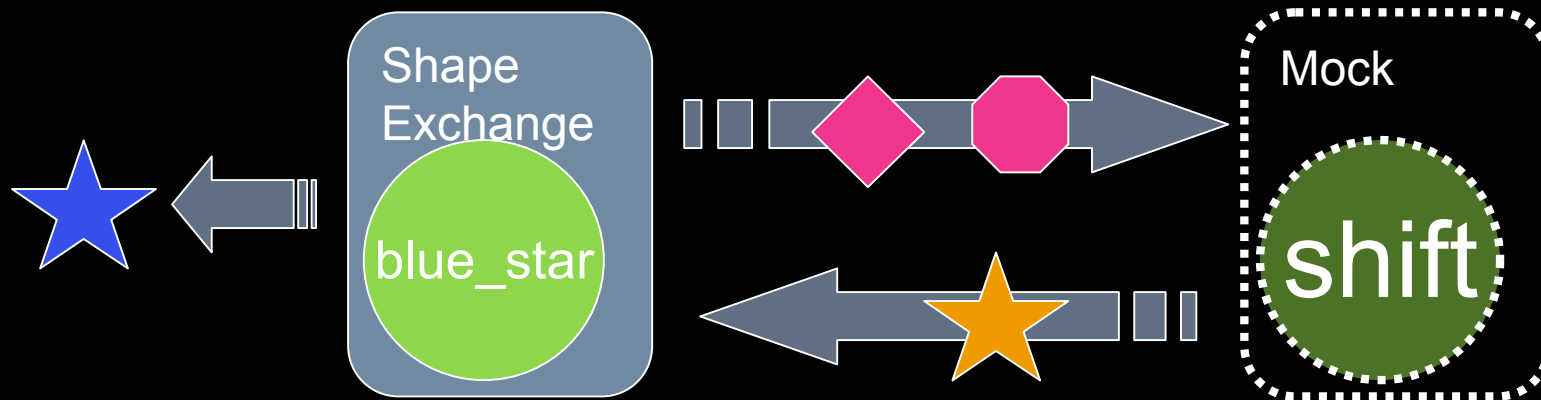
# Green

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     ShapeShifter.shift(:diamond, :octagon)
4   end
5 end
```

# Incomplete

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     ShapeShifter.shift(:diamond, :octagon)
4   end
5 end
```

# Red



```
untitled 2
1 def test_makes_blue_star
2   ShapeShifter.expects(:shift).with(:diamond, :octagon).returns(:star)
3   assert_equal(:blue_star, ShapeExchange.blue_star)
4 end
5
```

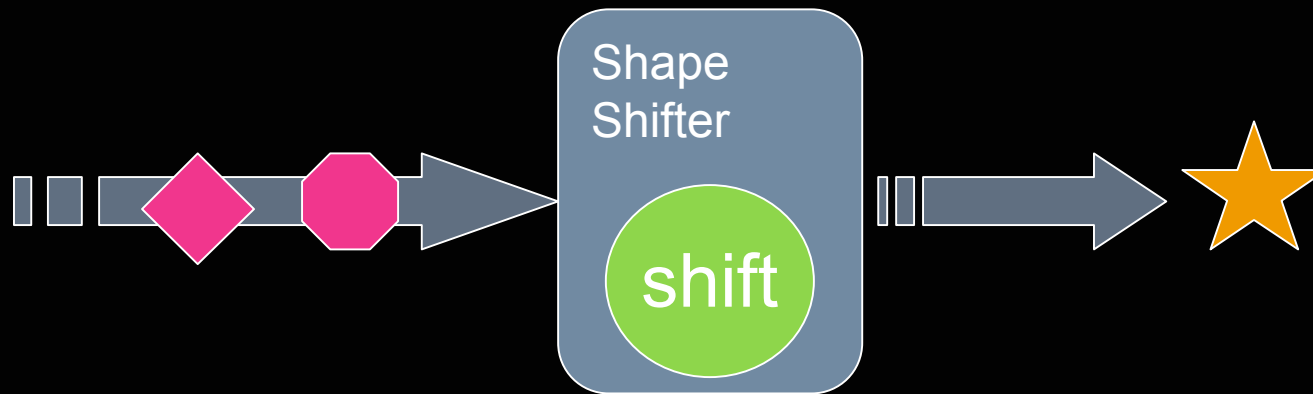
# Green

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     star = ShapeShifter.shift(:diamond, :octagon)
4     paint_blue(star)
5   end
6 end
```

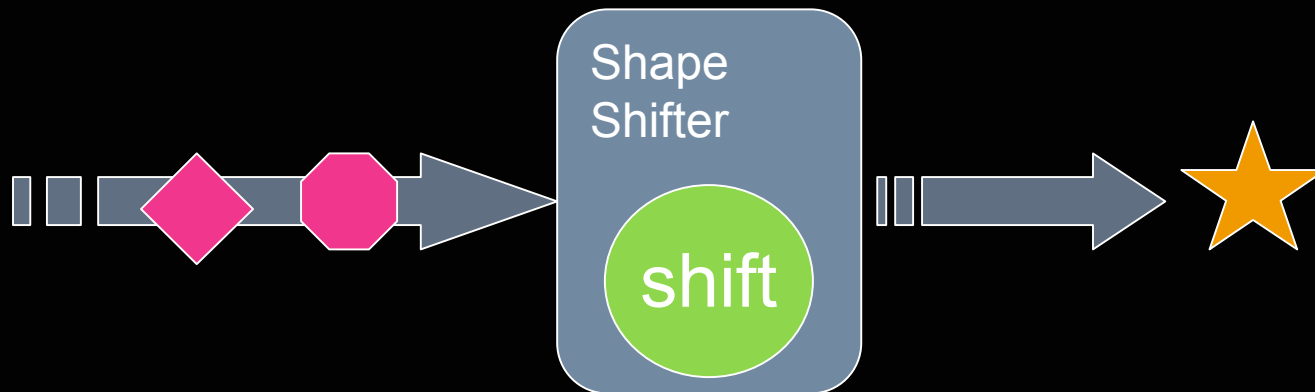
# Where?

```
untitled
1 class ShapeExchange
2   def self.blue_star
3     star = ShapeShifter.shift(:diamond, :octagon)
4     paint_blue(star)
5   end
6 end
```

# Can we prove this happens?



# Red




```
untitled 2
1 def test_shifts_start_from_diamond_and_octagon
2   assert_equal(:star, ShapeShifter.shift(:diamond, :octagon))
3 end
4
```

# Green

```
untitled
1 class ShapeShifter
2   def self.shift(*shapes)
3     really_complicated_shape_cooking(*shapes)
4   end
5 end
6
```

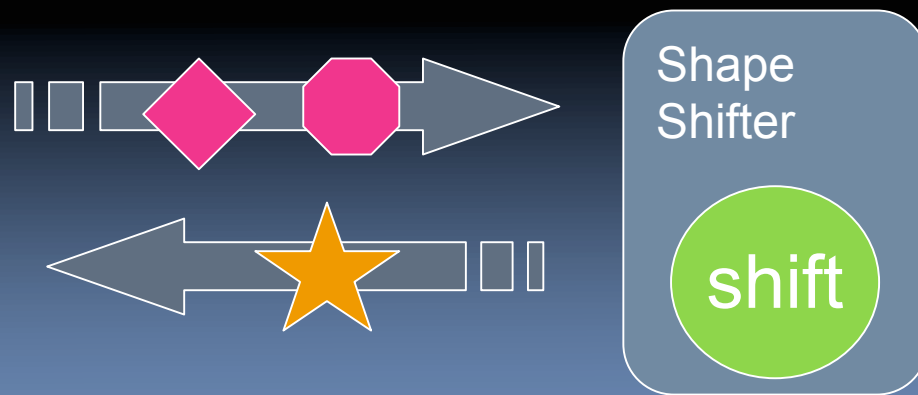
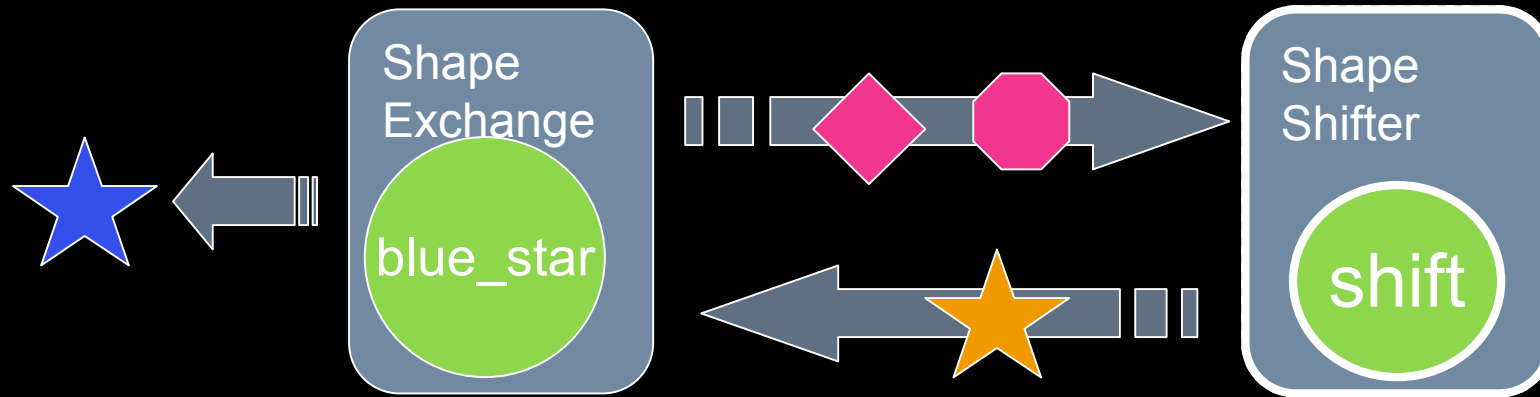


# Reduced technical debt

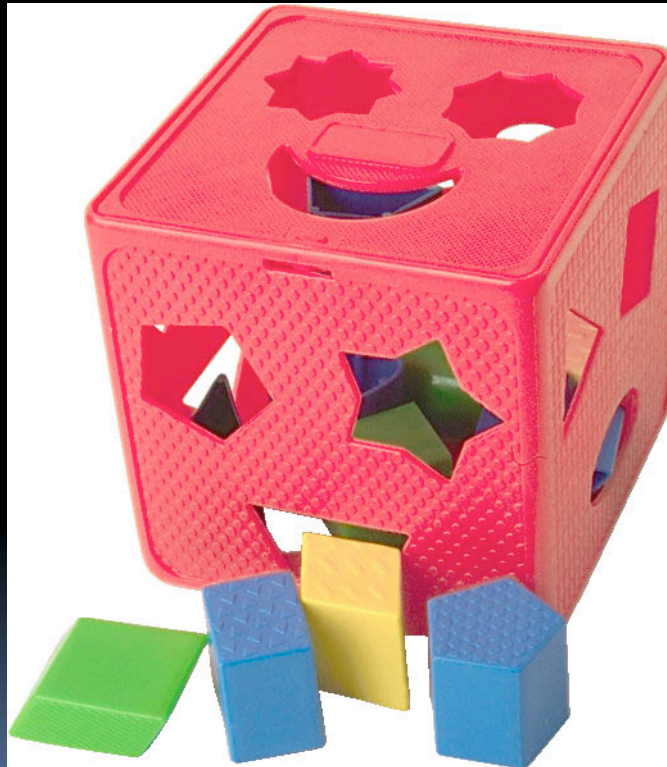


```
1 def test_makes_blue_star
2   # setup...
3   connect_to_db
4   load_shape_definitions_from_schema
5   create_shape_dependency_1
6   #....
7
8   assert_equal(:blue_star, ShapeExchange.blue_star)
9 end
10
```

# Synthesizing a bigger test



# Synthesis



# Works well with

## Existing tools

- RSpec
- Mocha
- zentest
- autotest

## Testing Practices

- TDD
- BDD

# Synthesis overview

Used through the Rake task `Synthesis::Task`

```
33 Synthesis::Task.new('synthesis:test:rspec') do |t|
34   t.adapter = :rspec
35   t.pattern = 'test_project/rspec/*_spec.rb'
36 end
```

# Synthesis overview

## First pass

Collect mocked expectations

## Second Pass

Correlate mocked expectations with REAL calls to REAL objects.

# Synthesis

- <http://synthesis.rubyforge.org/>
- [stuart.caborn@thoughtworks.com](mailto:stuart.caborn@thoughtworks.com)
- [george.malamidis@thoughtworks.com](mailto:george.malamidis@thoughtworks.com)