Rails from the Ground Up!

with Jared Richardson
RoleModel Software

Objects
Part One
Thinking in Objects

Object Oriented Thinking

- Knowledge
- Behavior

Four Questions

What's it responsible for?

What question does it answer?

What commands does it execute?

What information does it manage?

Deck of Cards

Answer the 4 questions (3x5 card)

Discuss your solution with classmates

Revise/refactor

Implement

http://www.flickr.com/photos/lainamarie/6193896626/

Do You Have?

reset

shuffle

more_cards?

int get_card

Testing

Deeply embedded in the Ruby culture

Runit ships with Ruby

```
> ruby card_test.rb
```

runit

```
require 'test/unit'

require 'CardDeck'

class TestCardDeck < Test::Unit::TestCase

  def test_num_cards

    num_cards = card_deck.num_cards

    assert_equal num_cards, 52, "Should have had 52 cards."

  end

end
```

assertions

```
assert(<condition>, message)

assert_equal( expected, actual, msg)

assert_not_equal( expected, actual, msg)

assert_not_nil(<variable>, msg>

assert_repond_to( dog, :bark, "Dogs
should bark!")
```

Revisit the Cards

Simple tests

Each method

What will you discover?

Ruby Gems

Package management for Ruby

```
gem --version
```

Practical Example

```
gem install simplecov

require 'simplecov'

SimpleCov.start

re-run your tests

look in the coverage directory
```

War!

Two players

Each draws a card

High card wins

Use Your CardDeck

Code a console application

Runs war for "two" players (A & B)

Play the entire deck

```
> ruby war.rb
A:7, B:3 => Player A wins!
A:4, B:9 => Player B wins!
A:8, B:11 => Player B wins!
```

but ...

First, Think!

What's it responsible for?

What question does it answer?

What commands does it execute?

What information does it manage?

Rake

Scripting for Ruby

Used for builds, testing, deployment

Files are valid Ruby

tasks

```
task :name
```

tasks with pre-requisites

```
task :name => :prereq_1

task :name => [:prereq_]

task :name => [:prereq_1, :prereq_2]
```

tasks with actions

```
task :name do |t|
  # do stuff (may reference t)
end
```

Rakefile

Default file is Rakefile

```
rake

rake -f <other Rakefile>

rake -T
```

File & Directory Tasks

```
directory "tmp"
file "hello" => "tmp" do
  sh "echo 'Hello' >> 'tmp/hello.tmp'"
end
```

A Script Example

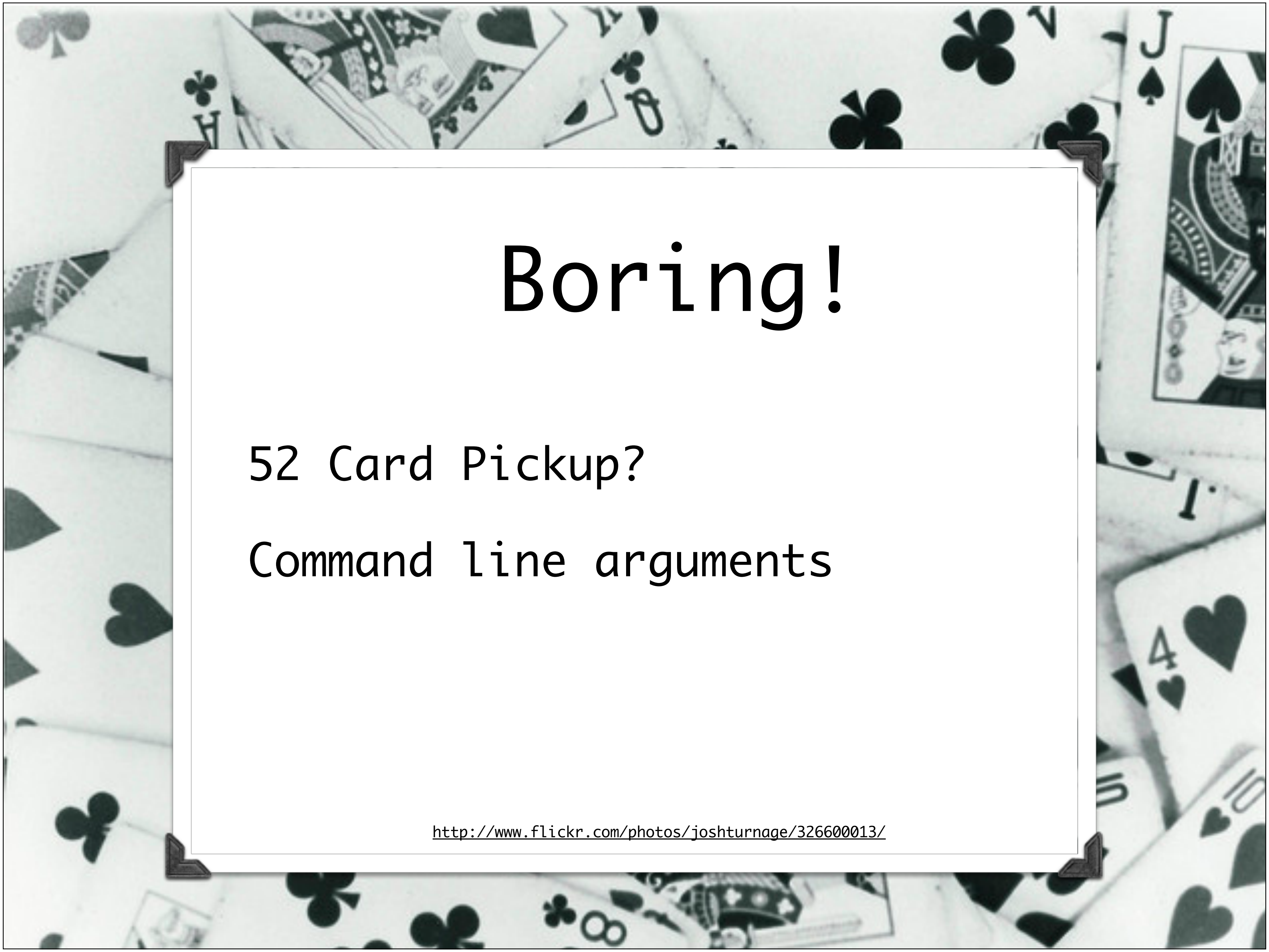
```
SERVER_DIR = /path/to/server/directory


task :start do
  Dir.chdir(SERVER_DIR) do
    sh "./startServer.sh >out.log &"
  end
end
```

Your Turn

Write your own Rakefile

rake client

rake server



Boring!

- 52 Card Pickup?
- Command line arguments

http://www.flickr.com/photos/joshturnage/326600013/

Optparse

Bundled Ruby library

Parses command line options

```
require 'optparse'

options={}   # holds all the parsed options

optparse = OptionParser.new do |opts|

  opts.banner = "Usage: ruby war.rb ...."

  # now add options until you have all you need

  options[:verbose] = false

  opts.on( '-v', '--verbose', "More info") do

    options[:verbose] = true

  end

end

optparse.parse!
```

Use the Results

```
if( options[:verbose])
  puts "The verbose flag was set"
end
```

Note:

ARGV still contains unparsed options

```
ARGV.each do | arg |
  puts arg
end
```

War Options

Number of hands to play

```
ruby war.rb 5
```

http://www.flickr.com/photos/idfonline/6869569255/

Next...

Networking in Ruby