Business Natural Language (BNL)

ThoughtWorks
http://thoughtworks.com

Jay Fields
http://jayfields.com


wtf?

Domain Specifc Language for Domain Experts

distinguishing characteristics

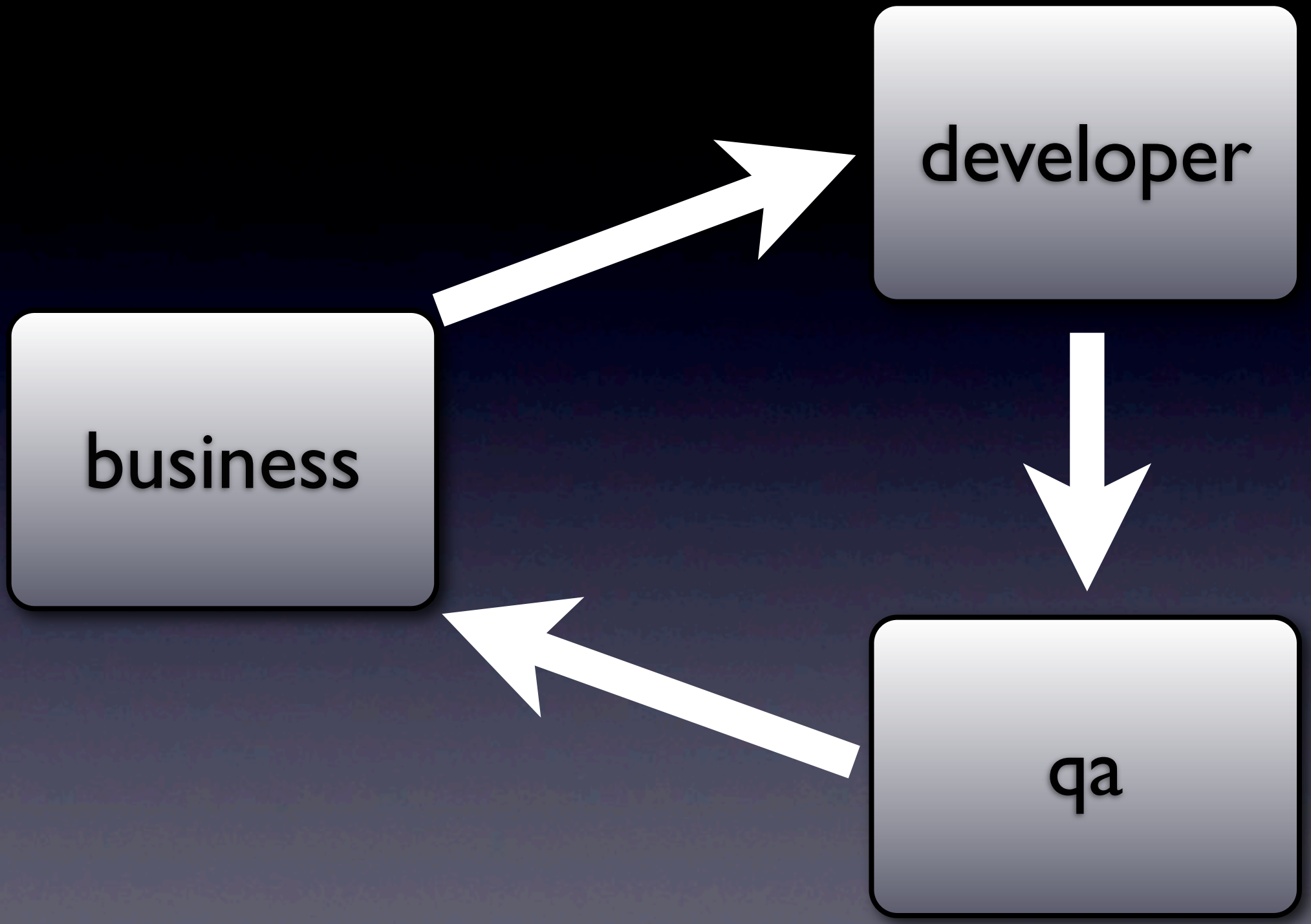
natural language

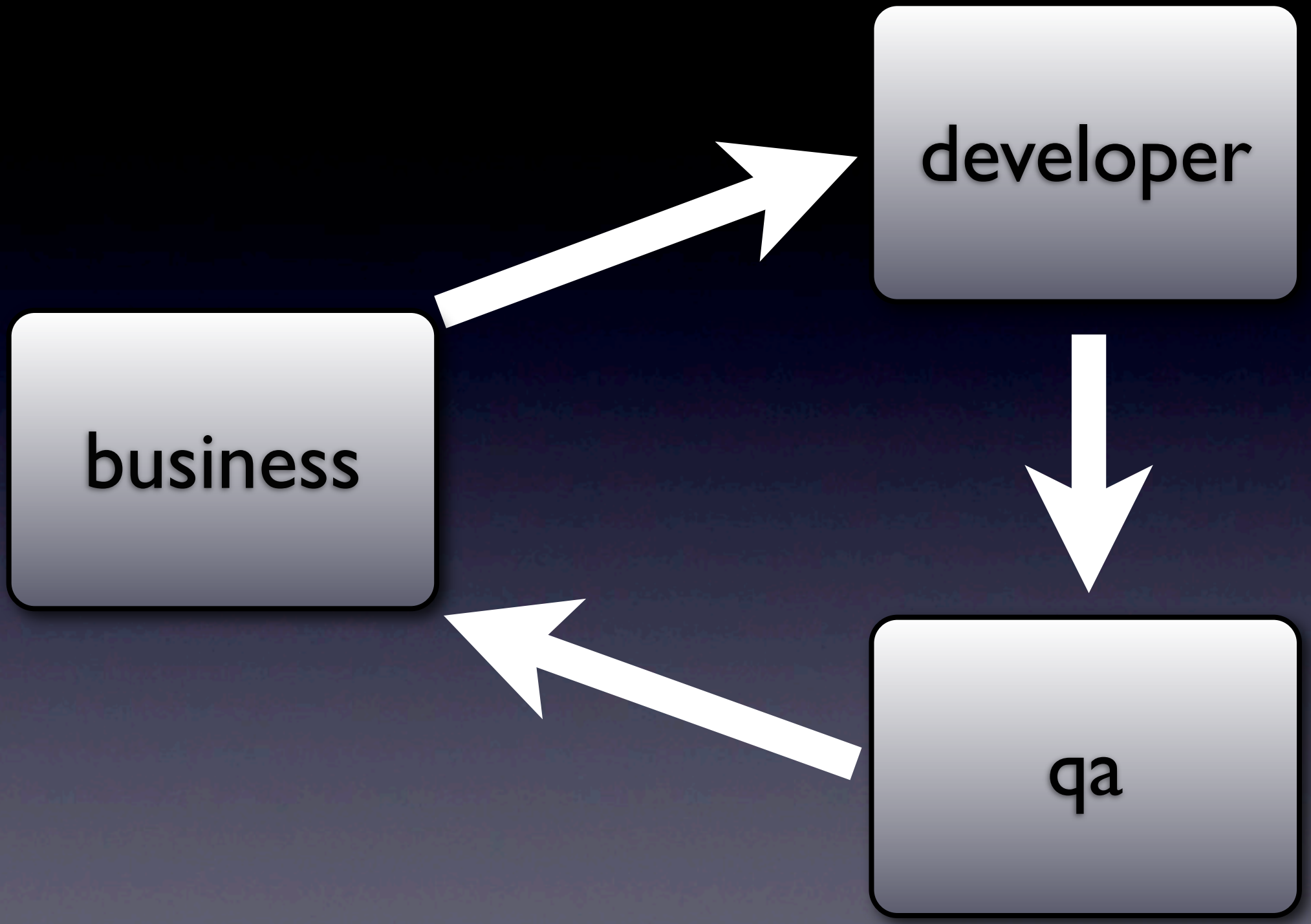
written by the business

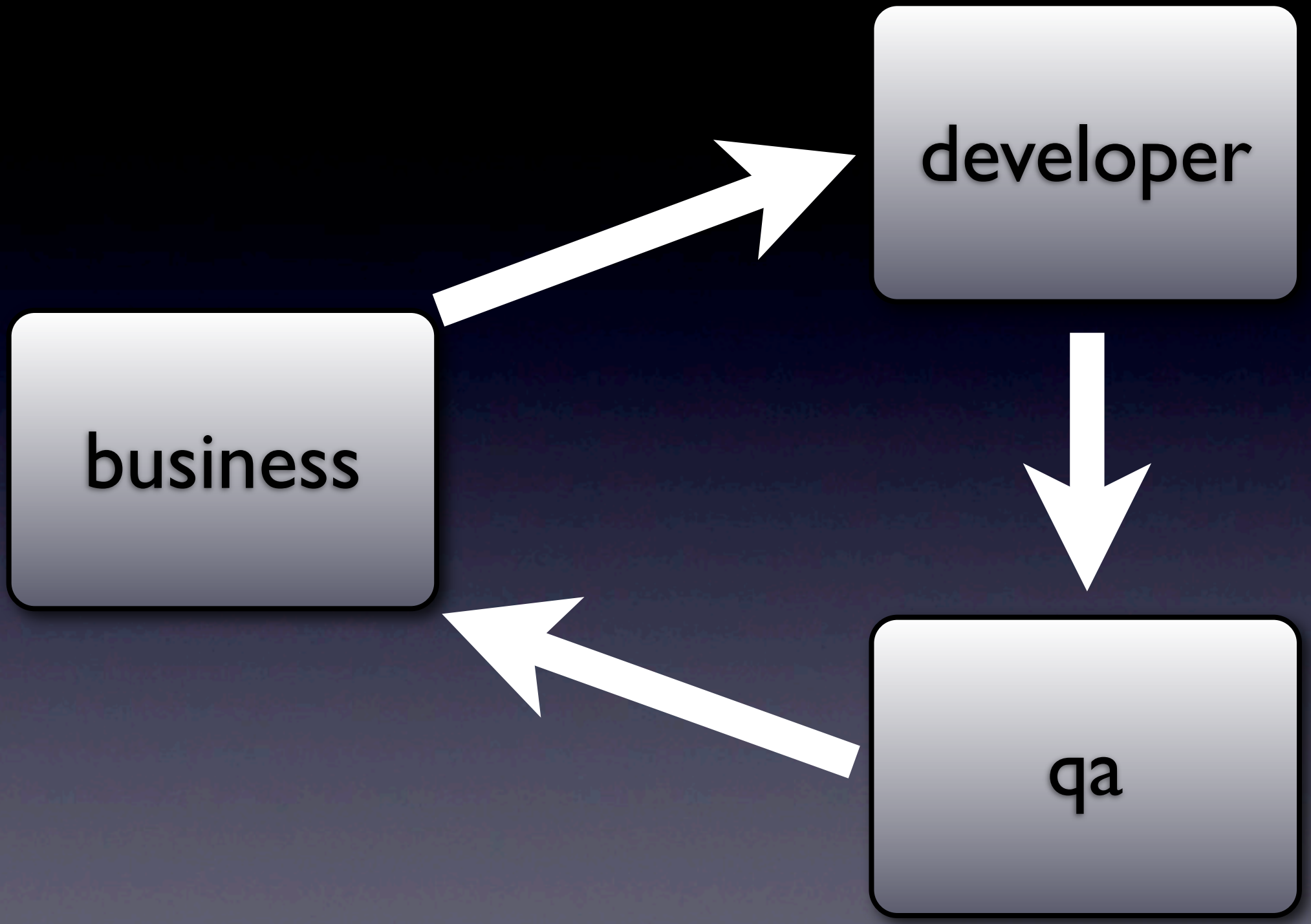
developer

business

qa


developer
business
qa


developer
business
qa


developer
business
qa

DSL

Benefits

Managers: Distribute Workload

Developers: Focus on Technical Issues

Business: Time To Market

htf?

(does it work)

external

First Class / Business Class / BusinessFirst / Full Fare Economy Class	A, C, D, J, R, Y, H Z	2
Economy Class	K, M, N, B, O, V, U	1

specification

Award 2 points if fare class is A, C, D, J, R, Y, H, Z

code

Award 2 points if fare class is A, C, D, J, R, Y, H, Z

```ruby
class FareRule
  def apply(segment)
    %w[a, c, d, j, r, y, h z].include?(segment.fare_class.downcase) ? 2 : 0
  end
end
```

```ruby
def self.create_rule(line)
  case line
  when /^award (\d+) points* if fare class is ([A-Za-z](, [A-Za-z])*)/i
    create_fare_rule(Object.new, $1, $2)
    # .. other whens
  end
end

def self.create_fare_rule(rule, amount, fare_classes)
  eval "def rule.apply(segment)
          %w[#{fare_classes.delete(",")}].include?(segment.fare_class.downcase) ? #{amount} : 0
        end"
  rule
end
```

code, specification,

code, specification,
and documentation

readable
documentation

Syntax OK

```
Award 1 point If fare class is B
Award 1 points if fare class is A
Award 1 point if flight is to SFO
Award 5 points if flight is from EWR
Award 8 points if flight is From las
```

Update

descriptive and maintainable phrases (damp)

maintainable

readable by any subject matter expert

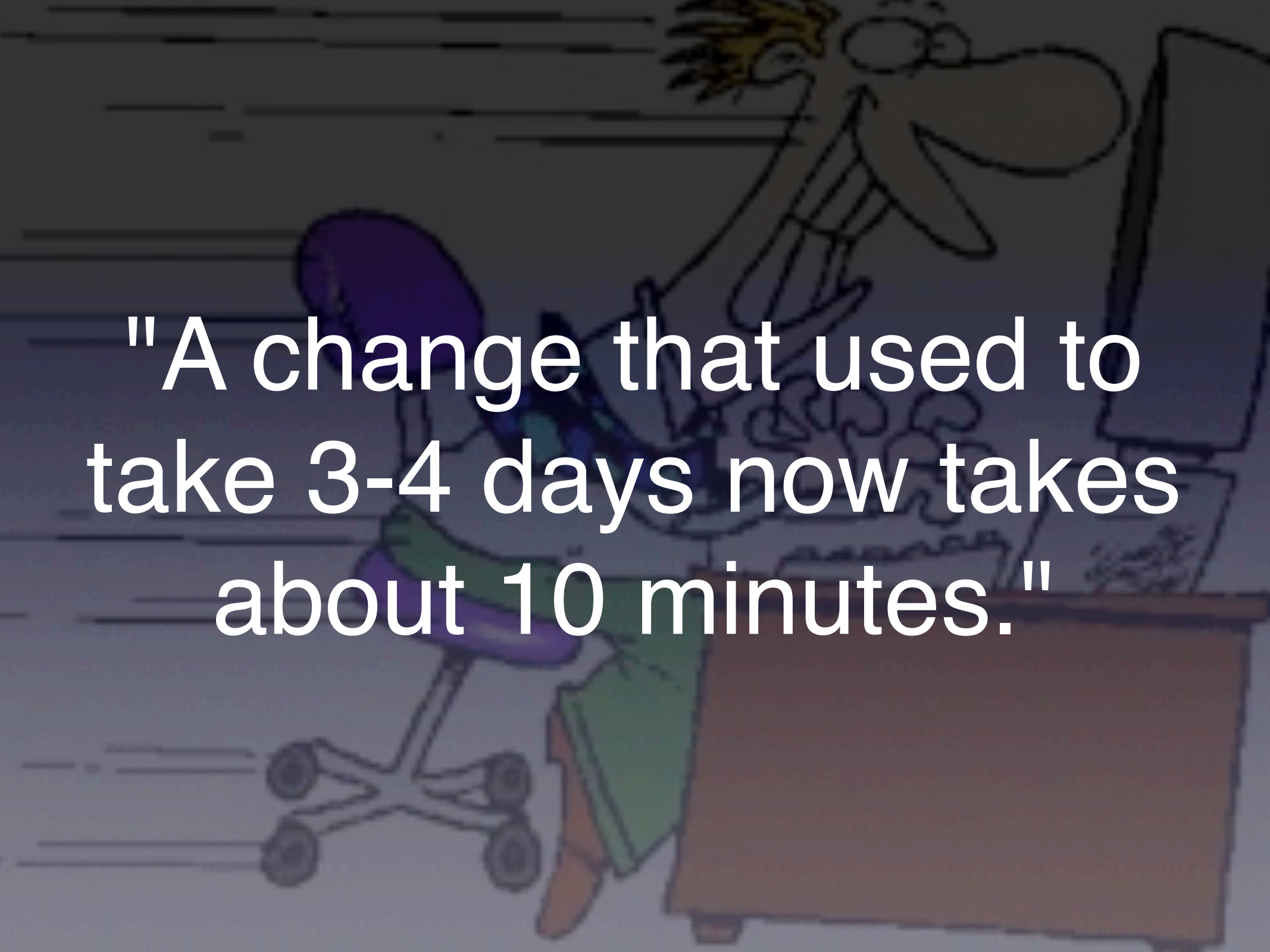

you
user

Syntax OK

```
Award 1 point If fare class is B
Award 1 points if fare class is A
Award 1 point if flight is to SFO
Award 5 points if flight is from EWR
Award 8 points if flight is From las
```

Update

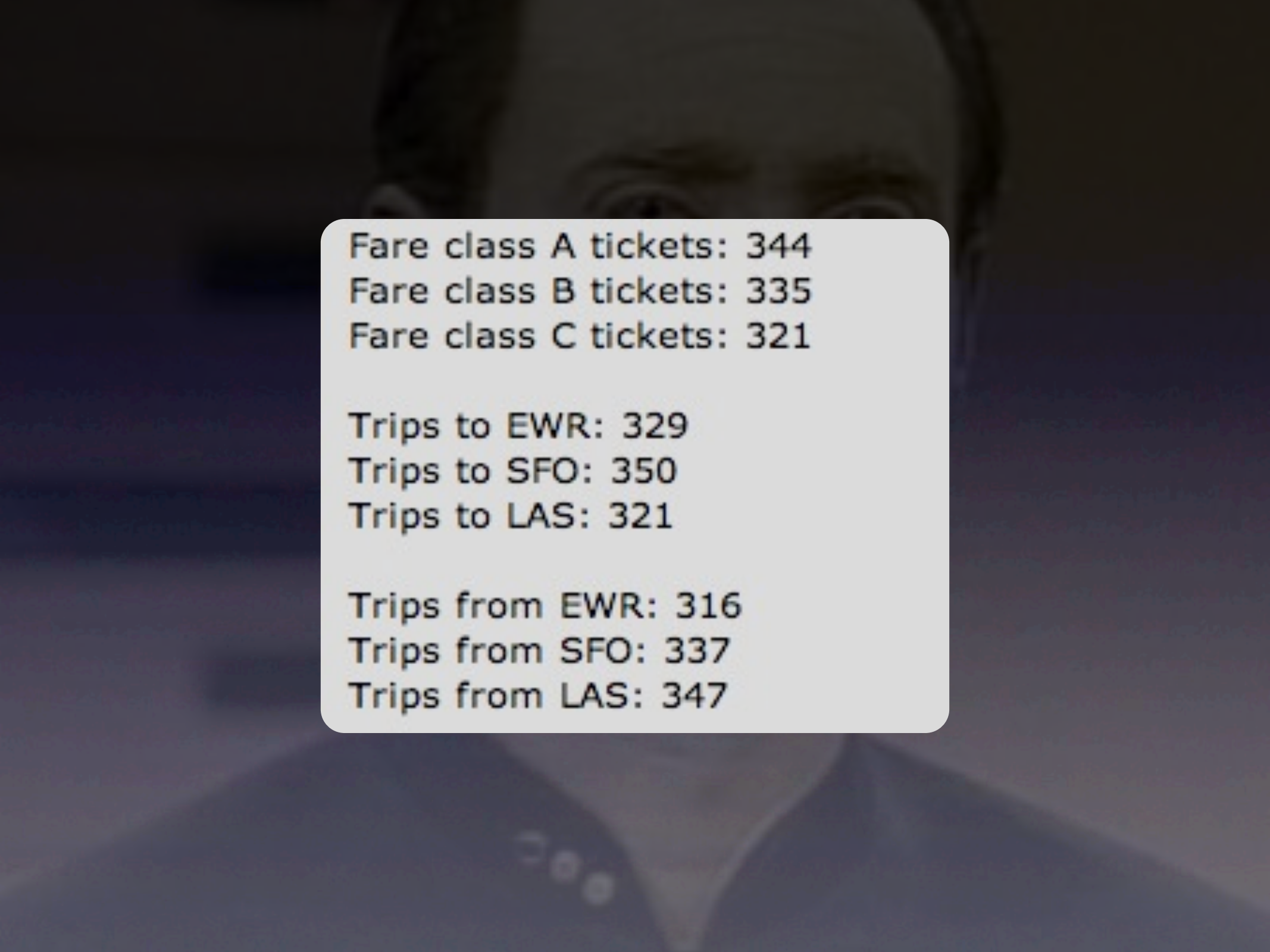

Training


"A change that used to take 3-4 days now takes about 10 minutes."

DEMO

Teachers
Talking
French

```
---
- :to: SFO
  :frequent_flyer_number: BK1126
  :from: EWR
  :fare_class: A
- :to: EWR
  :frequent_flyer_number: BK7670
  :from: LAS
  :fare_class: B
- :to: LAS
  :frequent_flyer_number: BK8783
  :from: SFO
  :fare_class: B
- :to: SFO
  :frequent_flyer_number: BK7107
  :from: LAS
  :fare_class: C
- :to: SFO
```

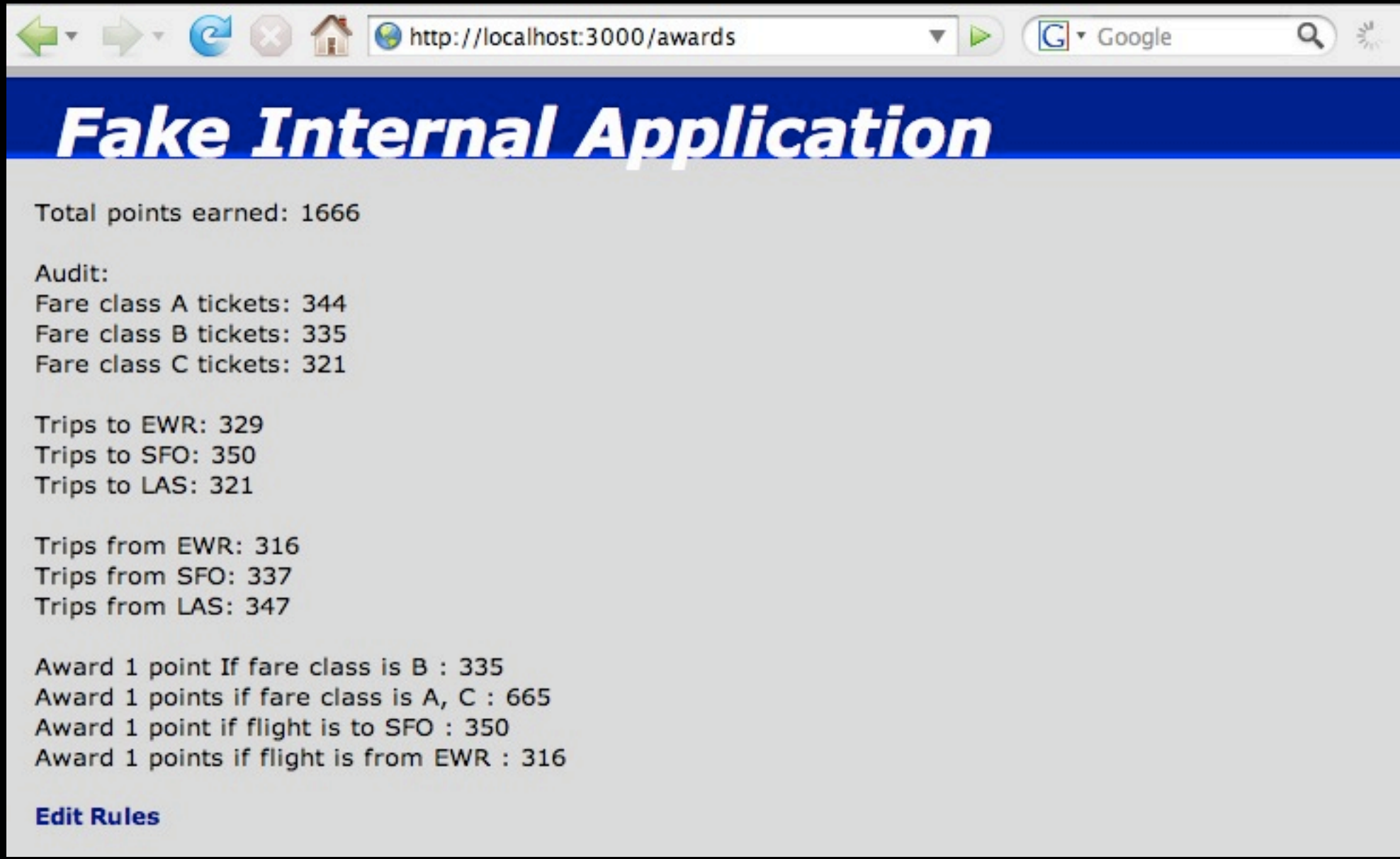

Fare class A tickets: 344
Fare class B tickets: 335
Fare class C tickets: 321
Trips to EWR: 329
Trips to SFO: 350
Trips to LAS: 321
Trips from EWR: 316
Trips from SFO: 337
Trips from LAS: 347

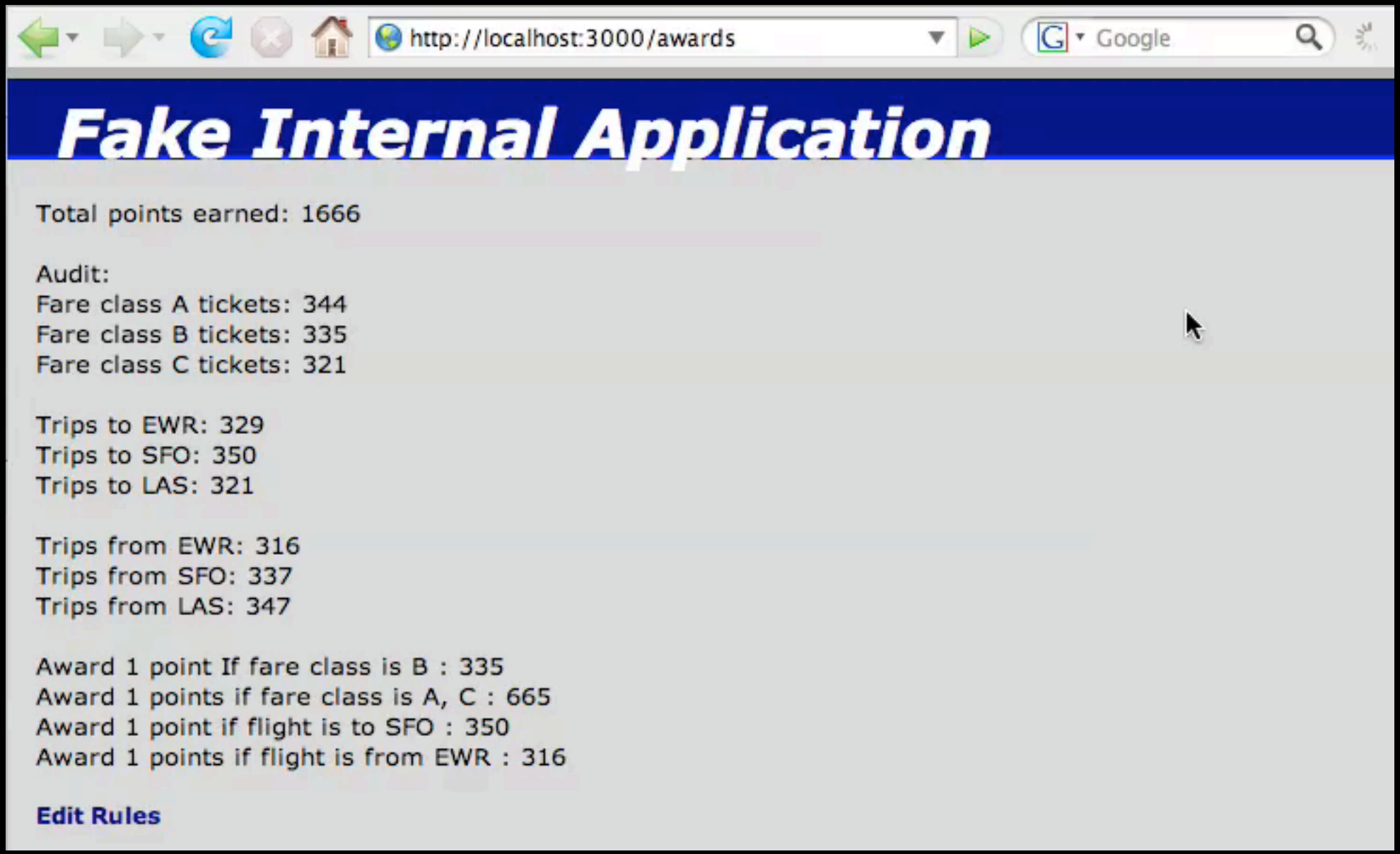
Award 1 point If fare class is B : 335
Award 1 points if fare class is A, C : 665
Award 1 point if flight is to SFO : 350
Award 1 points if flight is from EWR : 316

Fare class A tickets: 344
Fare class B tickets: 335
Fare class C tickets: 321

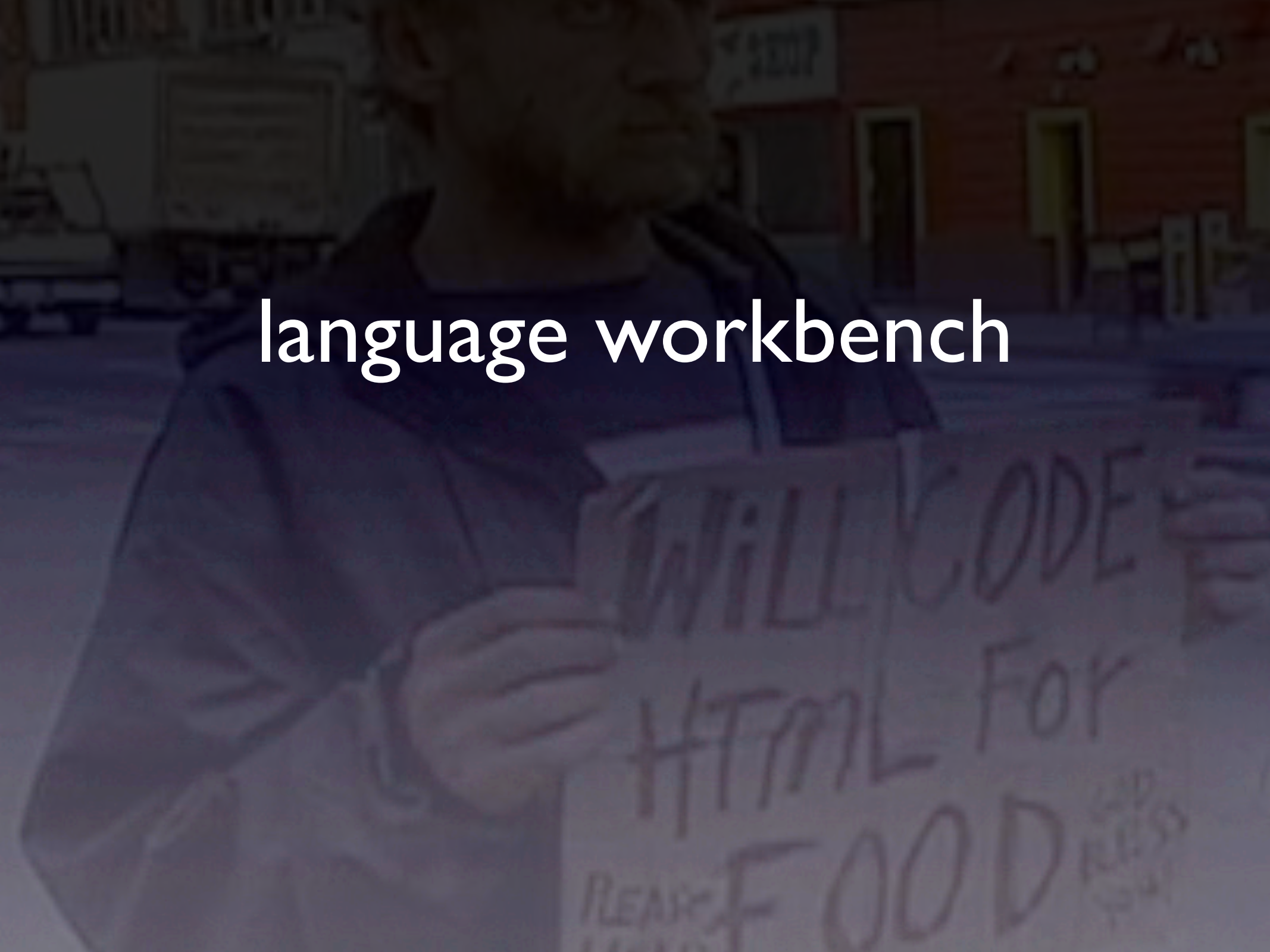
Trips to EWR: 329
Trips to SFO: 350
Trips to LAS: 321

Trips from EWR: 316
Trips from SFO: 337
Trips from LAS: 347

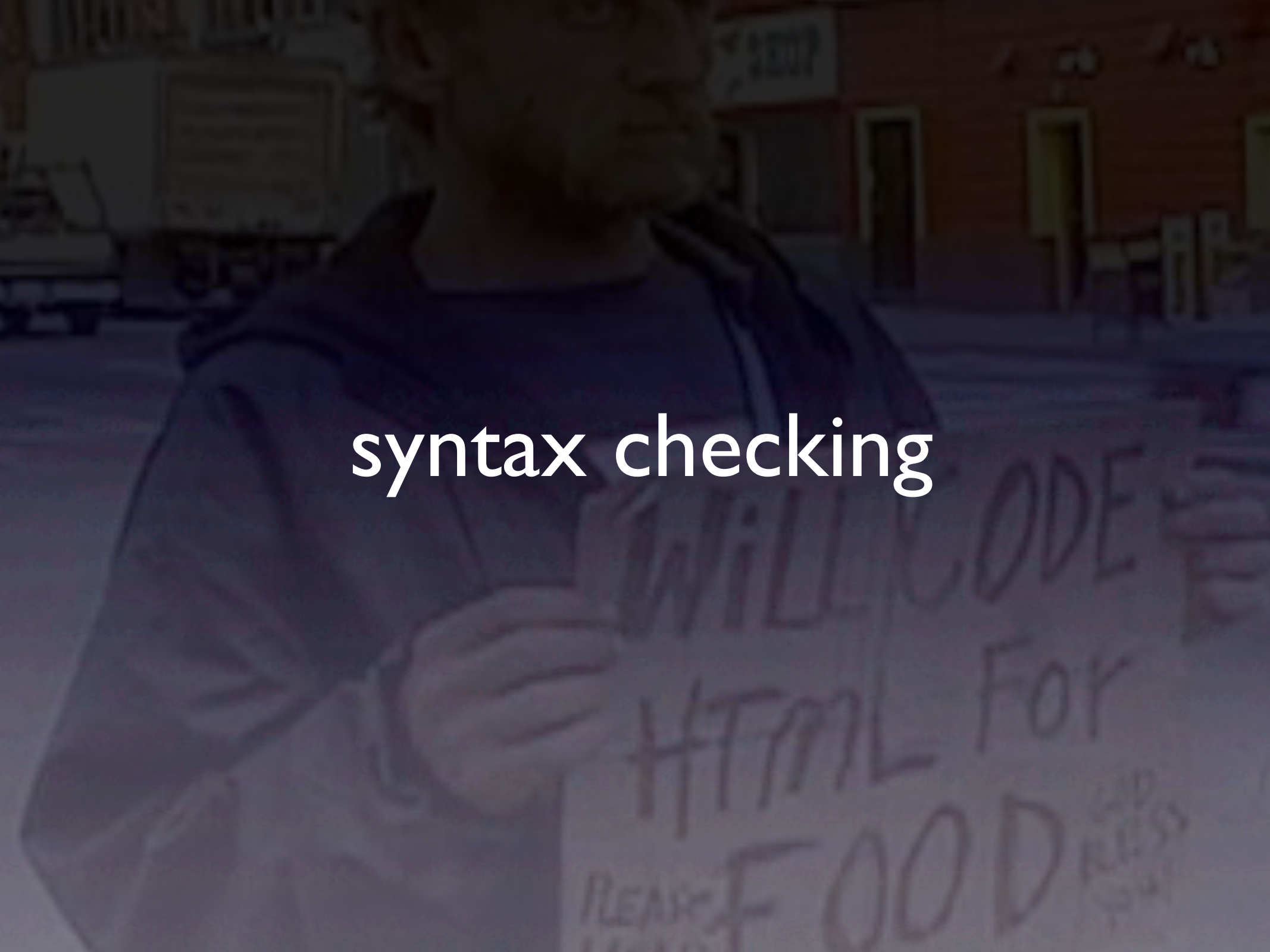

http://localhost:3000/awards
Google
Fake Internal Application
Total points earned: 1666
Audit:
Fare class A tickets: 344
Fare class B tickets: 335
Fare class C tickets: 321
Trips to EWR: 329
Trips to SFO: 350
Trips to LAS: 321
Trips from EWR: 316
Trips from SFO: 337
Trips from LAS: 347
Award 1 point If fare class is B : 335
Award 1 points if fare class is A, C : 665
Award 1 point if flight is to SFO : 350
Award 1 points if flight is from EWR : 316
Edit Rules


http://localhost:3000/awards
Google
Fake Internal Application
Total points earned: 1666
Audit:
Fare class A tickets: 344
Fare class B tickets: 335
Fare class C tickets: 321
Trips to EWR: 329
Trips to SFO: 350
Trips to LAS: 321
Trips from EWR: 316
Trips from SFO: 337
Trips from LAS: 347
Award 1 point If fare class is B : 335
Award 1 points if fare class is A, C : 665
Award 1 point if flight is to SFO : 350
Award 1 points if flight is from EWR : 316
Edit Rules

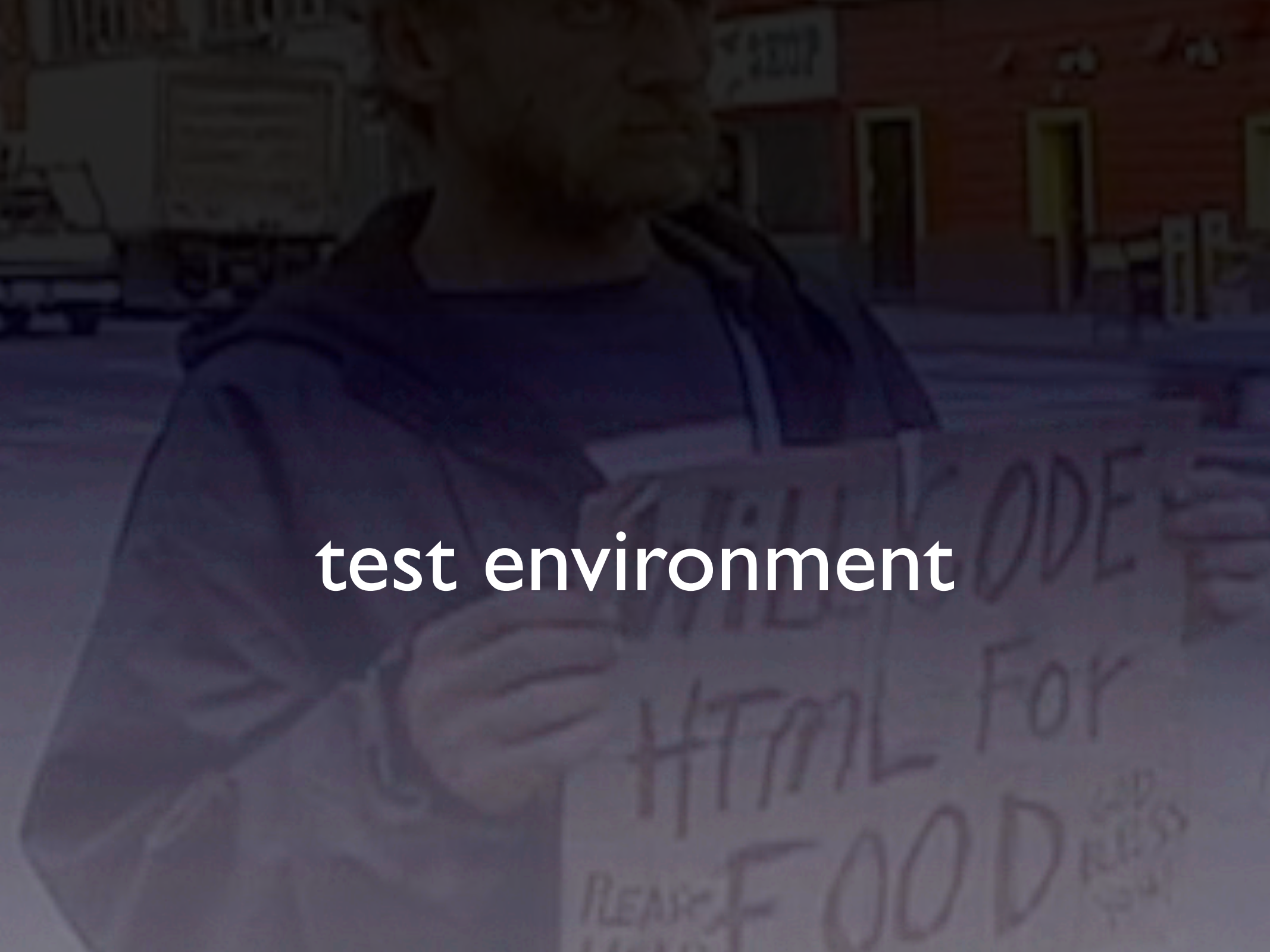

WiLL CODE
HTML For
FOOD

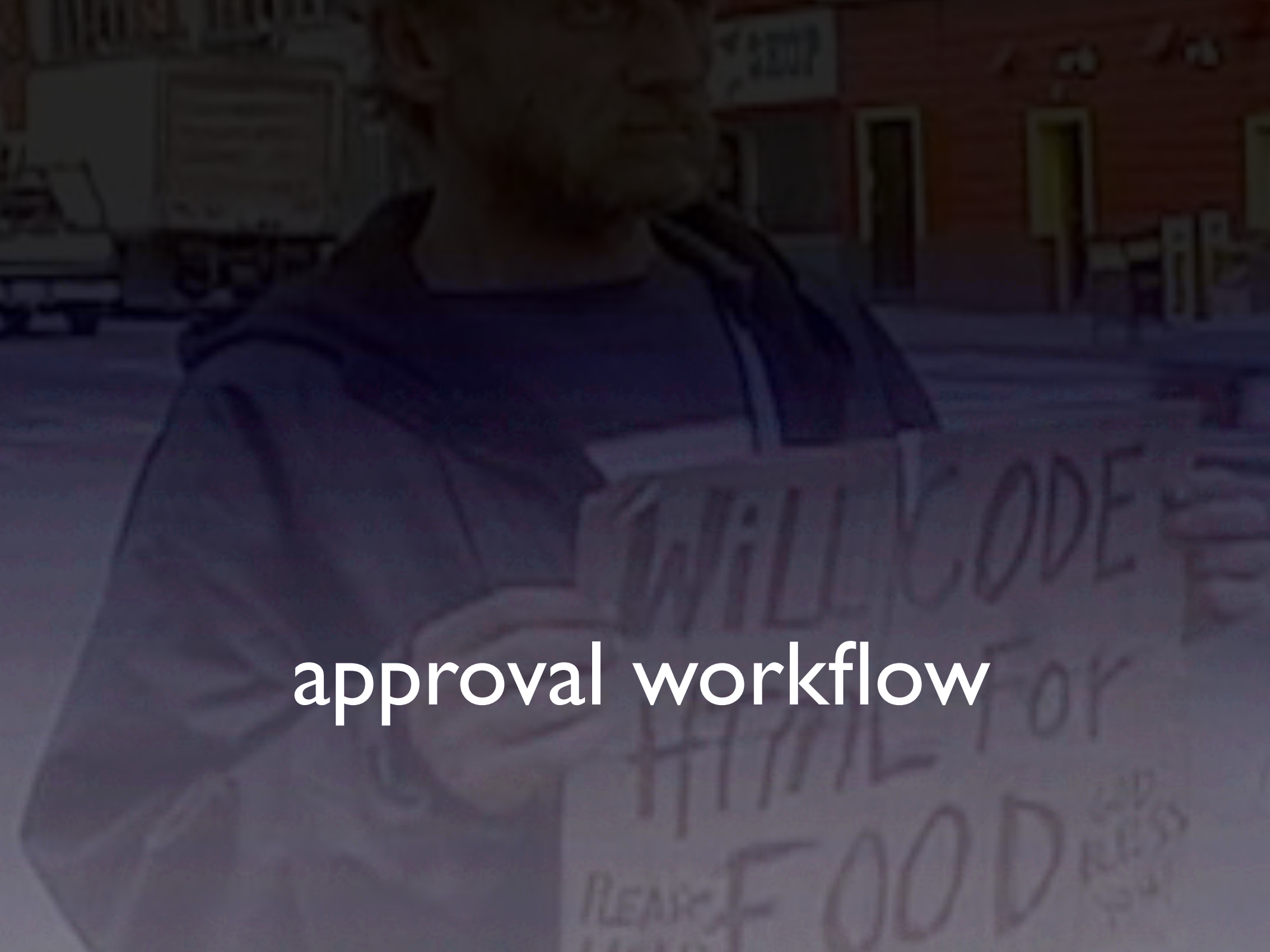


language workbench



syntax checking

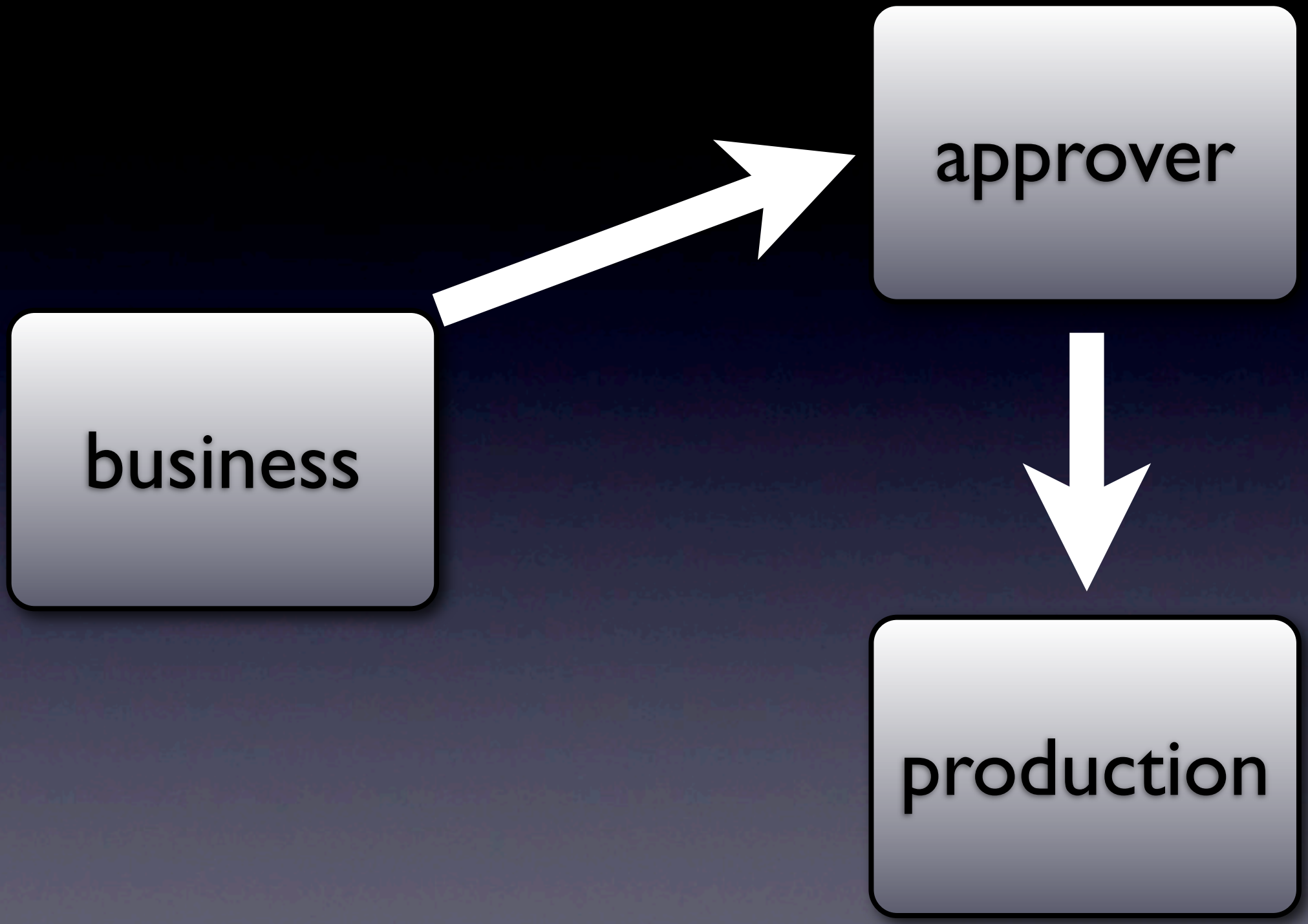

test environment



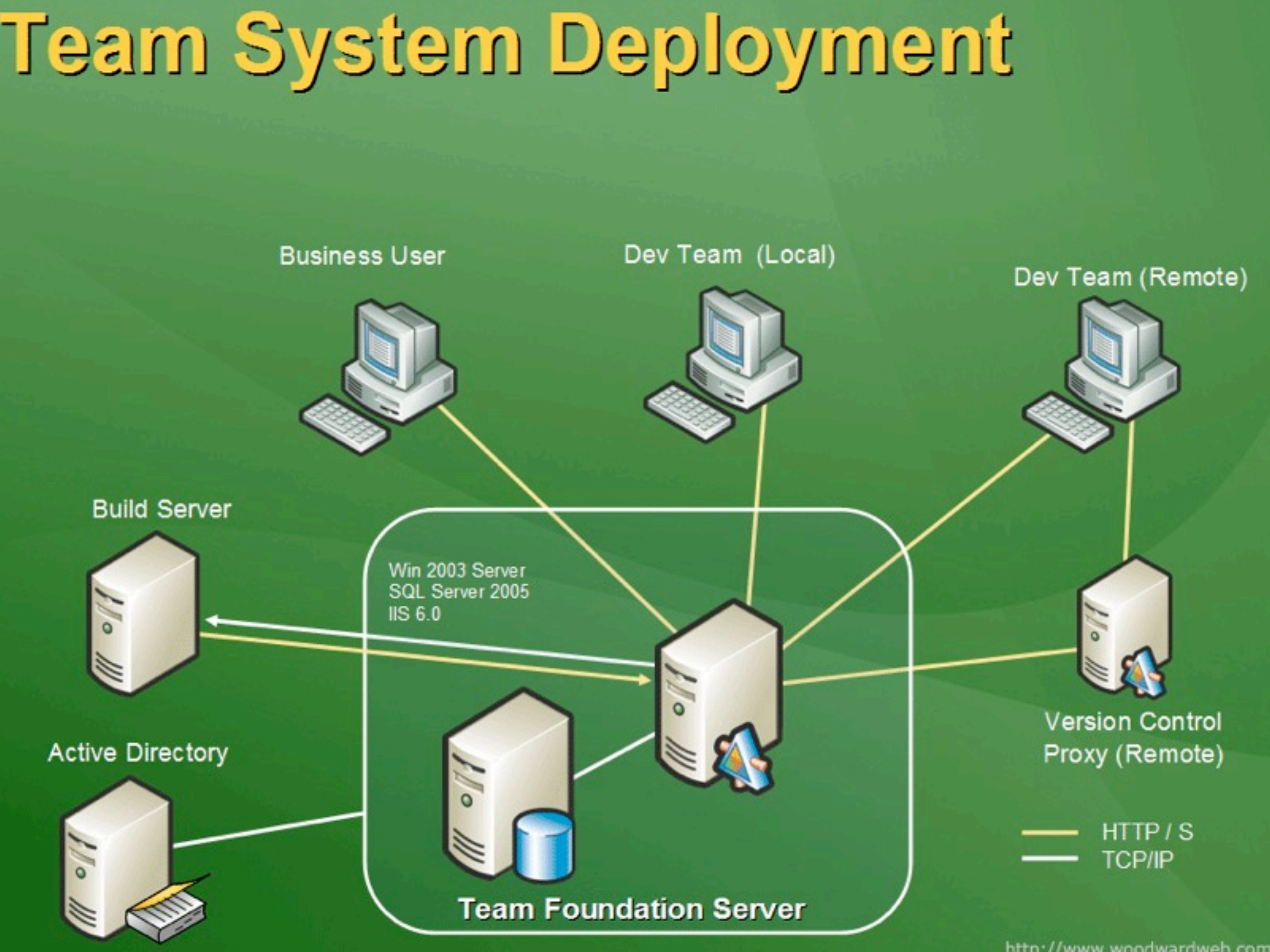
approval workflow

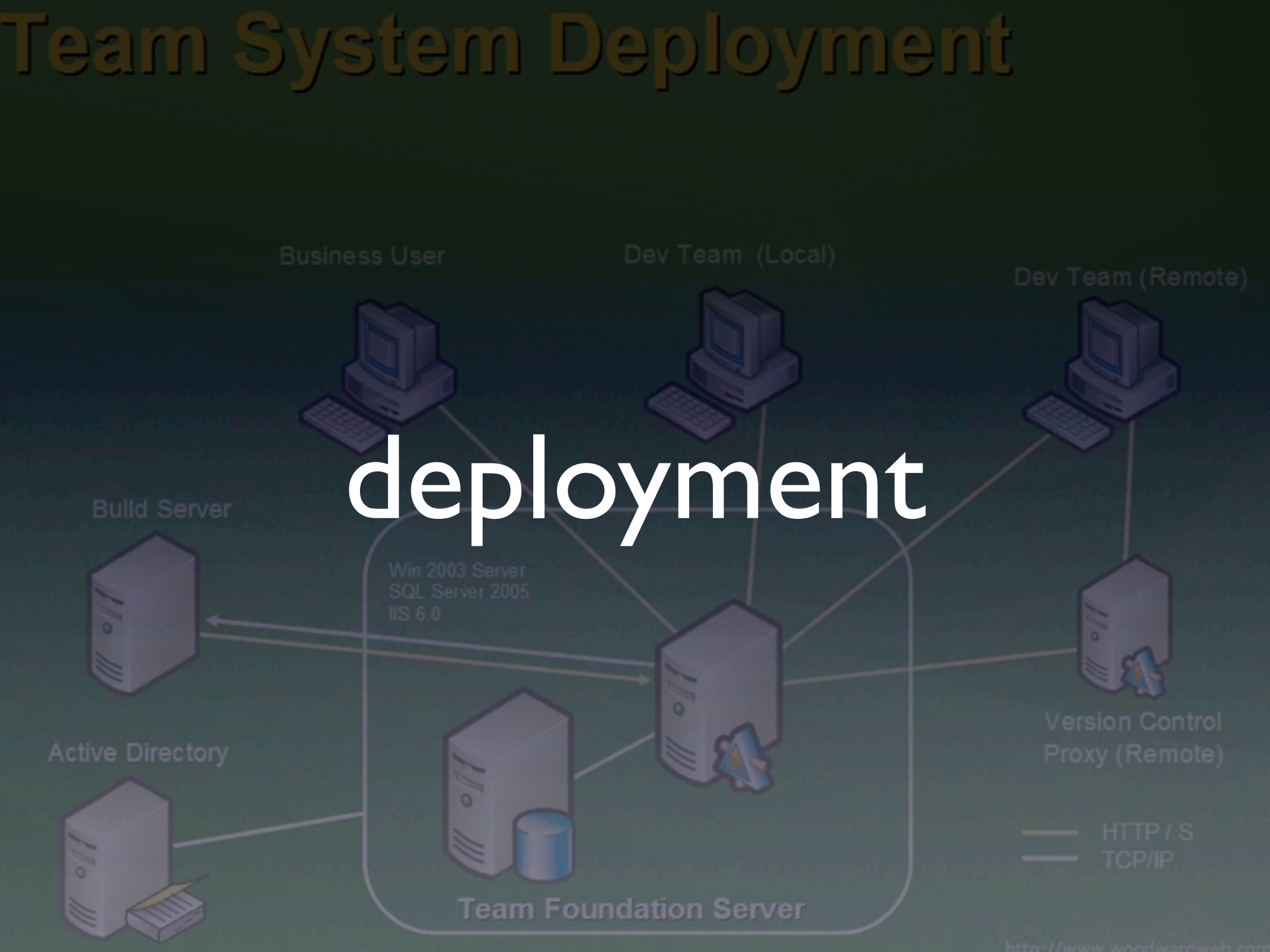

business

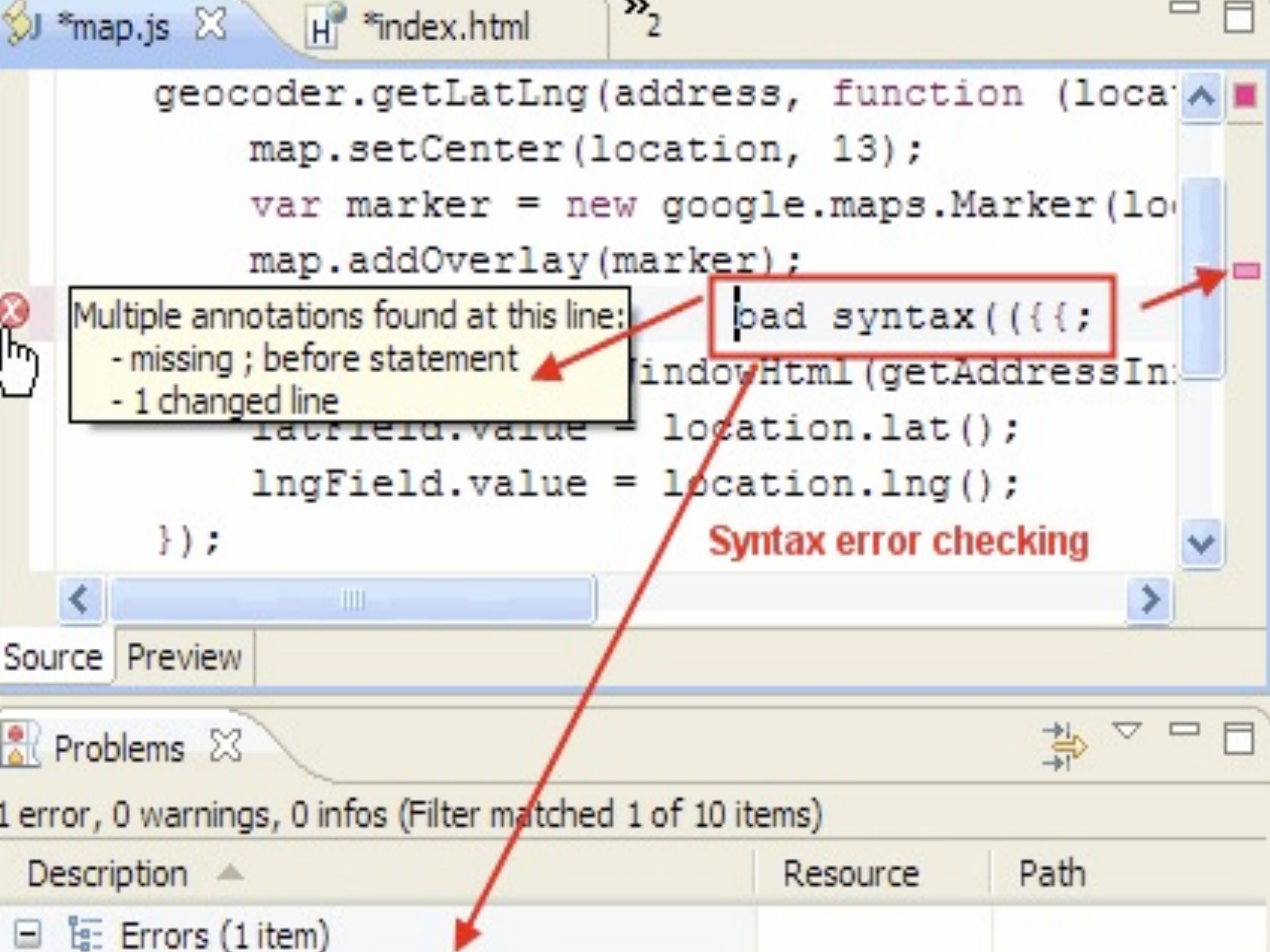

business

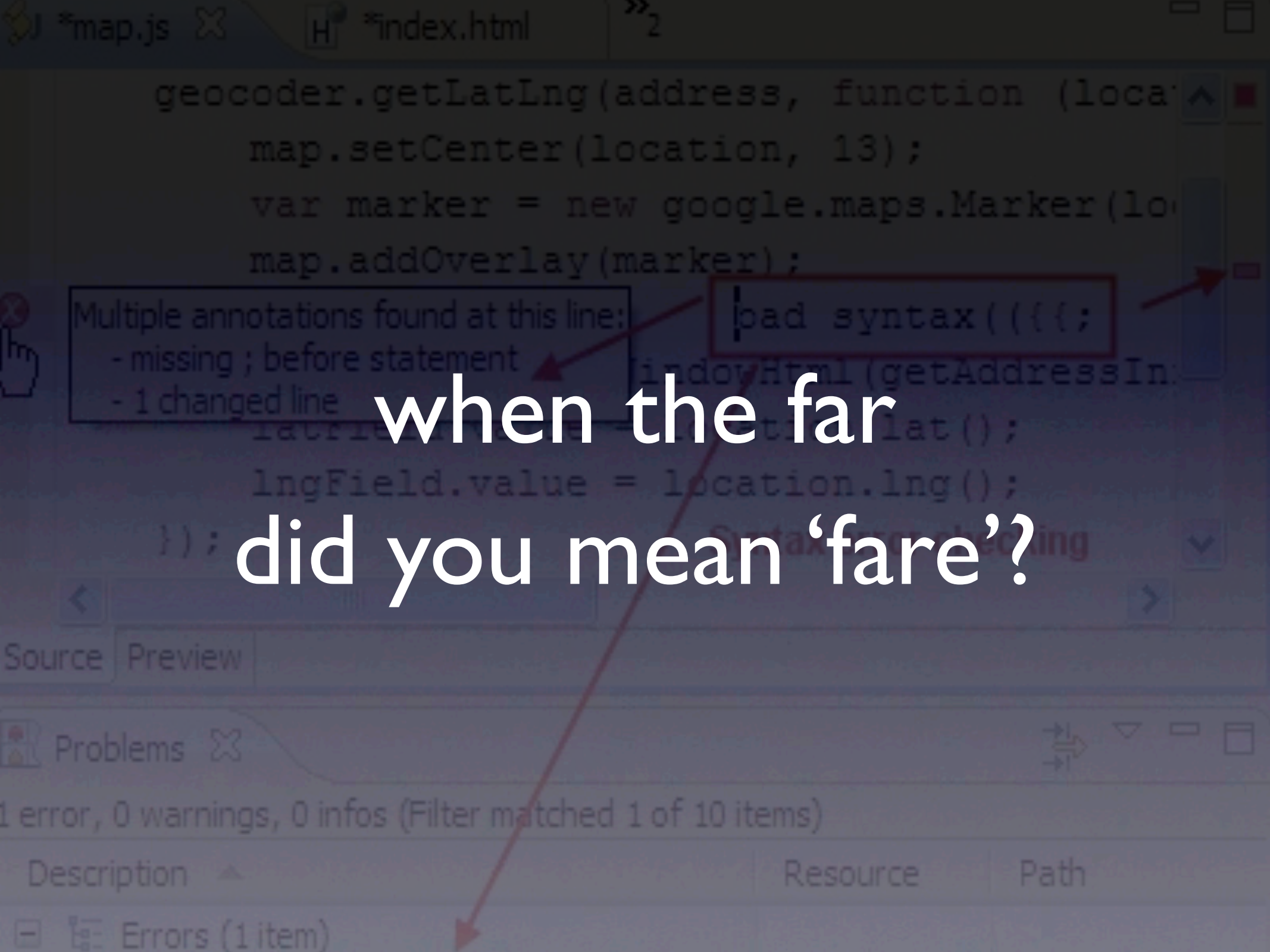

approver
business
production

Team System Deployment




Team System Deployment
Business User
Dev Team (Local)
Dev Team (Remote)
deployment
Build Server
Win 2003 Server
SQL Server 2005
IIS 6.0
Active Directory
Version Control
Proxy (Remote)
HTTP / S
TCP/IP
Team Foundation Server


*map.js
*index.html
geocoder.getLatLng(address, function (loca
map.setCenter(location, 13);
var marker = new google.maps.Marker(lo
map.addOverlay(marker);
bad syntax(({{;
Multiple annotations found at this line:
- missing ; before statement
- 1 changed line
indowHtml(getAddressIn
location.lat();
lngField.value = location.lng();
});
Syntax error checking
Source
Preview
Problems
1 error, 0 warnings, 0 infos (Filter matched 1 of 10 items)
Description
Resource
Path
Errors (1 item)



when the far

did you mean ‘fare’?

You entered pint. Award amount should be followed by 'point' or 'points' on line 3

```
Award 1 point If fare class is B
Award 1 points if fare class is A
Award 1 pint if flight is to SFO
Award 5 points if flight is from EWR
Award 8 points if flight is From las
```

Update

```ruby
class Syntax
  def self.parse(rules)
    find_errors(rules).any? ? find_errors(rules).join("\n") : "Syntax OK"
  end

  def self.find_errors(rules)
    rules.split(/\n/).enum_with_index.inject([]) do |errors, (line, index)|
      parse_line(errors, line, index + 1)
      errors
    end
  end

  def self.parse_line(errors, line, index)
    Line.new(line, index).check_line
  rescue Exception => ex
    errors << ex.message
  end
end
```

Syntax OK

```
Award 1 point If fare class is B
Award 1 points if fare class is A
Award 1 point if flight is to SFO
Award 5 points if flight is from EWR
Award 8 points if flight is From las
```

Update

```ruby
class Line
  attr_accessor :text, :line_number
  def initialize(text, line_number)
    self.text = text.squeeze(" ").strip
    self.line_number = line_number
  end

  def check_line
    check_pop(/^award /i, "Rules must start with 'Award'")
    check_pop(/^\d+ /i, "Rules must specify an award amount (e.g. 1, 23, 100)")
    check_pop(/^points* /i, "Award amount should be followed by 'point' or 'points'")
    check_pop(/^if /i, "Award point(s) should be followed by if")
  end

  def check_pop(regex, message_if_no_match)
    check(regex, message_if_no_match)
    pop(regex)
  end

  def pop(regex)
    text.gsub! regex, ""
  end

  def check(regex, message_if_no_match)
    raise "You entered #{text.gsub(/^(\S+).*/, "\\1")}. #{message_if_no_match}
           on line #{line_number}" unless text =~ regex
  end
end
```

```ruby
class Line
  attr_accessor :text, :line_number
  def initialize(text, line_number)
    self.text = text.squeeze(" ").strip
    self.line_number = line_number
  end
```

```ruby
def self.parse_line(errors, line, index)
  Line.new(line, index).check_line
rescue Exception => ex
  errors << ex.message
end
```

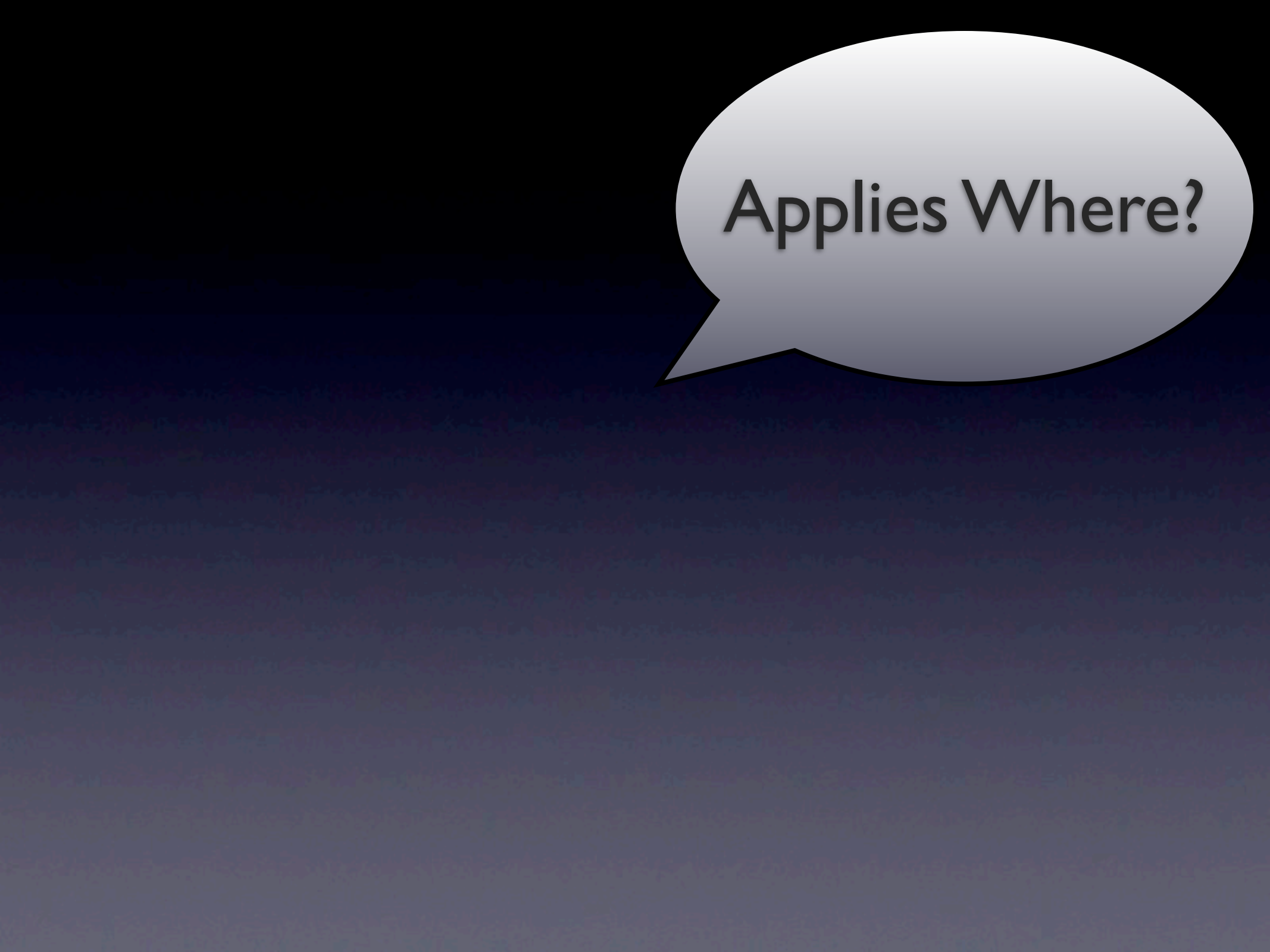
```ruby
class Line
  attr_accessor :text, :line_number
  def initialize(text, line_number)
    self.text = text.squeeze(" ").strip
    self.line_number = line_number
  end

  def check_line
    check_pop(/^award /i, "Rules must start with 'Award'")
    check_pop(/^\d+ /i, "Rules must specify an award amount (e.g. 1, 23, 100)")
    check_pop(/^points* /i, "Award amount should be followed by 'point' or 'points'")
    check_pop(/^if /i, "Award point(s) should be followed by if")
  end

  def check_pop(regex, message_if_no_match)
    check(regex, message_if_no_match)
    pop(regex)
  end

  def pop(regex)
    text.gsub! regex, ""
  end

  def check(regex, message_if_no_match)
    raise "You entered #{text.gsub(/^(\S+).*/, "\\1")}. #{message_if_no_match}
           on line #{line_number}" unless text =~ regex
  end
end
```

```ruby
def self.parse_line(errors, line, index)
  Line.new(line, index).check_line
rescue Exception => ex
  errors << ex.message
end
```

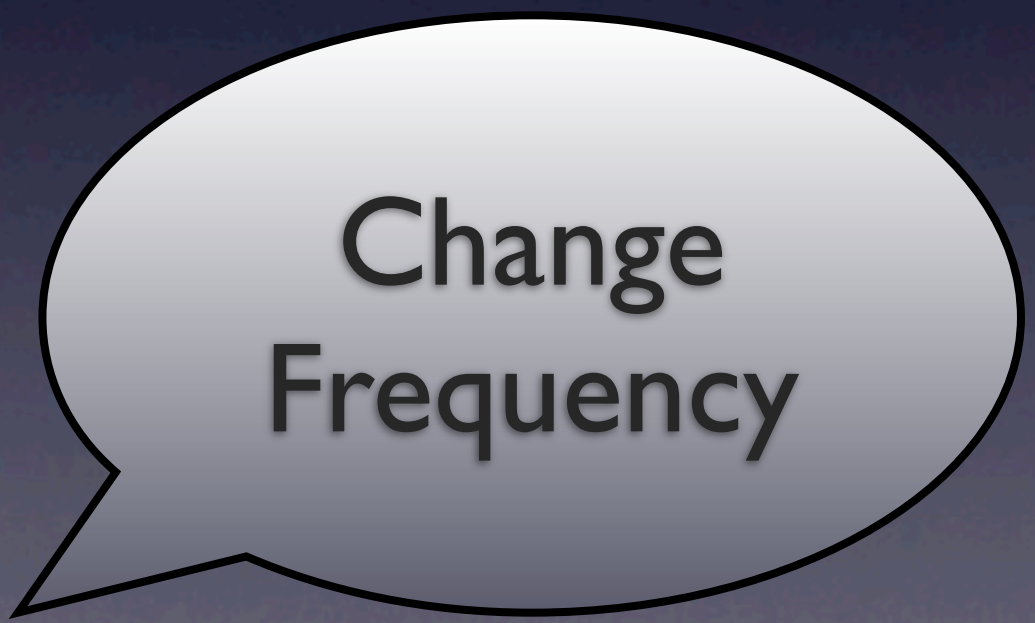
```ruby

  def check(regex, message_if_no_match)
    raise "You entered #{text.gsub(/^(\S+).*/, "\\1")}. #{message_if_no_match}
           on line #{line_number}" unless text =~ regex
  end
end
```

```ruby
class Line
  attr_accessor :text, :line_number
  def initialize(text, line_number)
    self.text = text.squeeze(" ").strip
    self.line_number = line_number
  end

  def check_line
    check_pop(/^award /i, "Rules must start with 'Award'")
    check_pop(/^\d+ /i, "Rules must specify an award amount (e.g. 1, 23, 100)")
    check_pop(/^points* /i, "Award amount should be followed by 'point' or 'points'")
    check_pop(/^if /i, "Award point(s) should be followed by if")
  end

  def check_pop(regex, message_if_no_match)
    check(regex, message_if_no_match)
    pop(regex)
  end

  def pop(regex)
    text.gsub! regex, ""
  end

  def check(regex, message_if_no_match)
    raise "You entered #{text.gsub(/^(\S+).*/, "\\1")}. #{message_if_no_match}
           on line #{line_number}" unless text =~ regex
  end
end
```

```
class Line
  def check_line
    check_pop(/^award /i, "Rules must start with 'Award'")
    check_pop(/^\d+ /i, "Rules must specify an award amount (e.g. 1, 23, 100)")
    check_pop(/^points* /i, "Award amount should be followed by 'point' or 'points'")
    check_pop(/^if /i, "Award point(s) should be followed by if")
    check(/^(fare|flight) /i, "If should be followed by either 'fare' or 'flight'")
    text =~ /^fare /i ? check_fare : check_flight
    raise "You entered #{text.gsub(/^(.*)/, "\\1").strip} at
           the end of line #{line_number}" unless text =~ /^$/
  end

  def check_fare
    pop(/^fare /i)
    check_pop(/^class /i, "Fare should be followed by 'class'")
    check_pop(/^is /i, "Class should be followed by 'is'")
    check_pop(/^[A-Za-z](, [A-Za-z])*/i, "Is should be followed by valid fare
                                          classes list (e.g. A or A, B, C)")
  end

  def check_flight
    pop(/^flight /i)
    check_pop(/^is /i, "Flight should be followed by 'is'")
    check_pop(/^(to|from) /i, "Is should be followed by either 'to' or 'from'")
    check_pop(/^[A-Za-z][A-Za-z][A-Za-z]/i, "Please enter a valid airport code")
  end
end
```


Applies Where?


Author?


Change
Frequency

commercial language workbenches

commercial language workbenches

MPS - JetBrains

Intentional


1


80/20 Rule

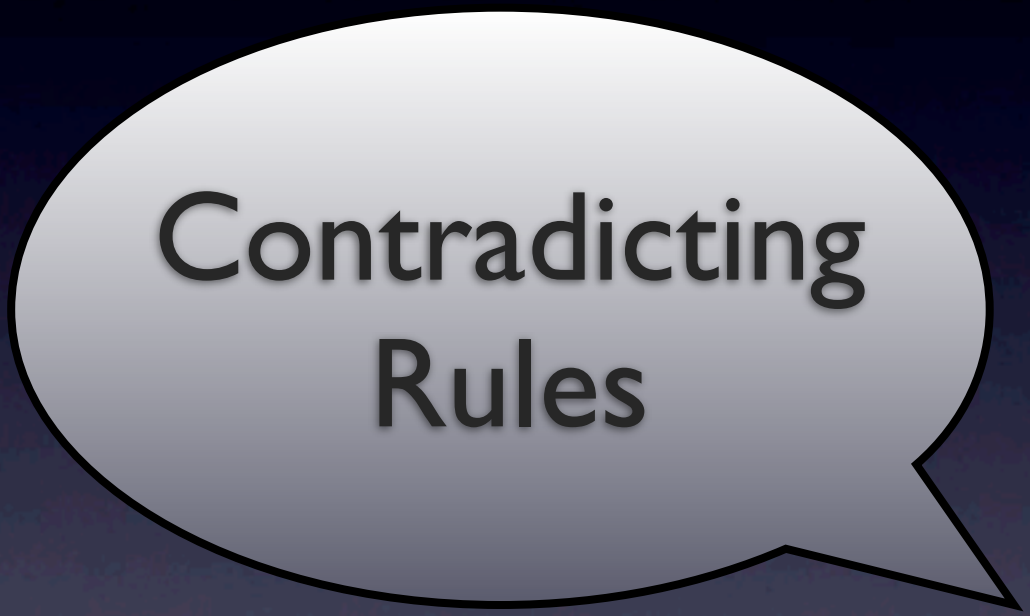
Treetop or Racc


Treetop or Racc

Treetop or Racc


More
complicated rules

Contradicting Rules


Contradicting
Rules

Contradicting Rules


Maturity

Questions

qcon@jayfields.com

ThoughtWorks
http://thoughtworks.com

Jay Fields
http://jayfields.com