

Tim Wingfield



Central OH Wendy Boys Kona Hockey

local/regional speaker

### THE FOLLOWING PREVIEW HAS BEEN APPROVED FOR ALL AUDIENCES

BY THE MOTION PICTURE ASSOCIATION OF AMERICA, INC.

www.filmratings.com

www.mpaa.org

















new-ish, 1.0 on April 12, 1.1 July 16 rails use with existing .net code





new-ish, 1.0 on April 12, 1.1 July 16 rails use with existing .net code

### Why IronRuby?



### Why IronRuby?



Wednesday, July 28, 2010

new-ish, 1.0 on April 12, 1.1 July 16 rails use with existing .net code

# What is?





1.8.6



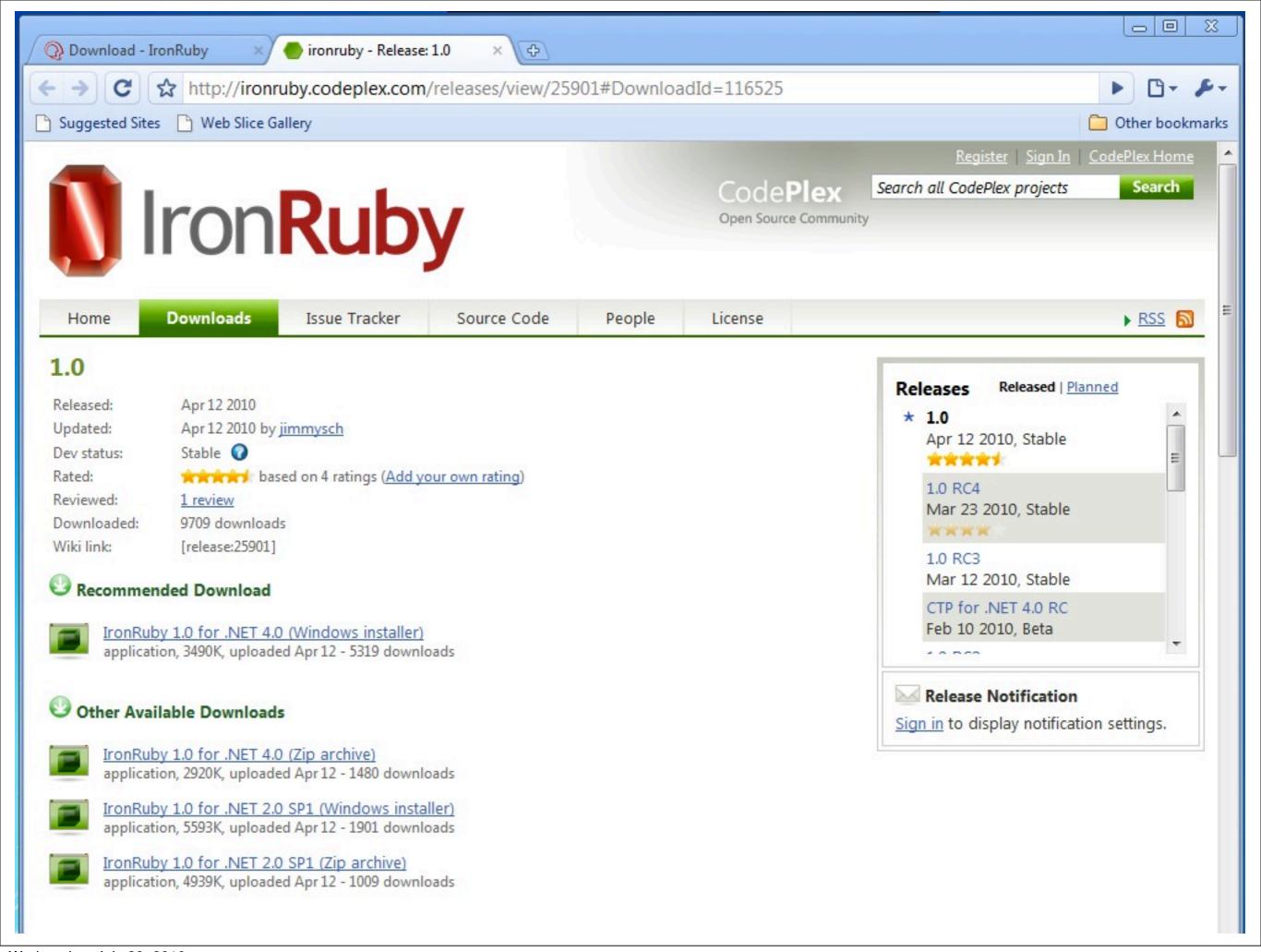
#### Dynamic Language Runtime



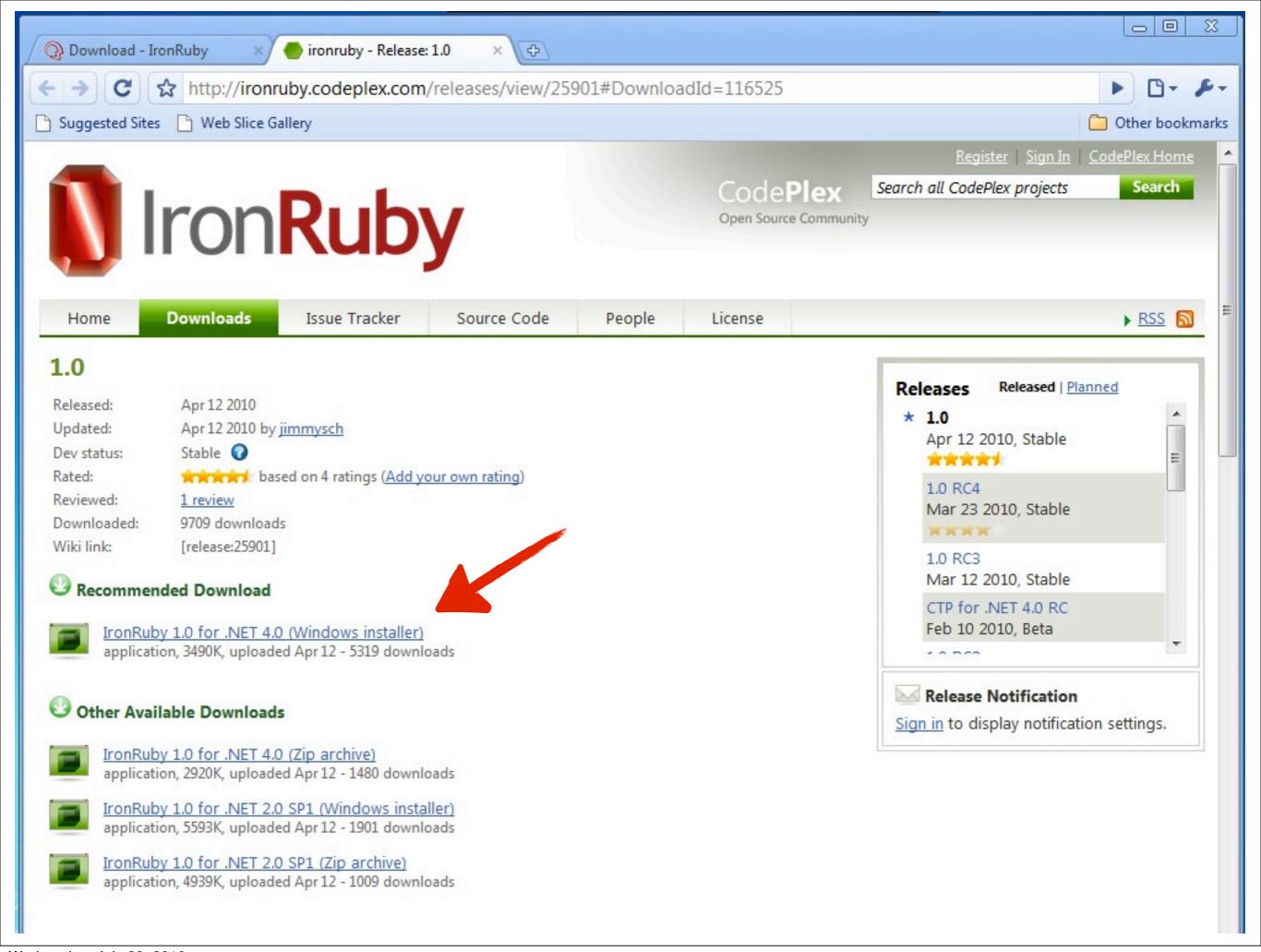
Common Language Runtime

Wednesday, July 28, 2010

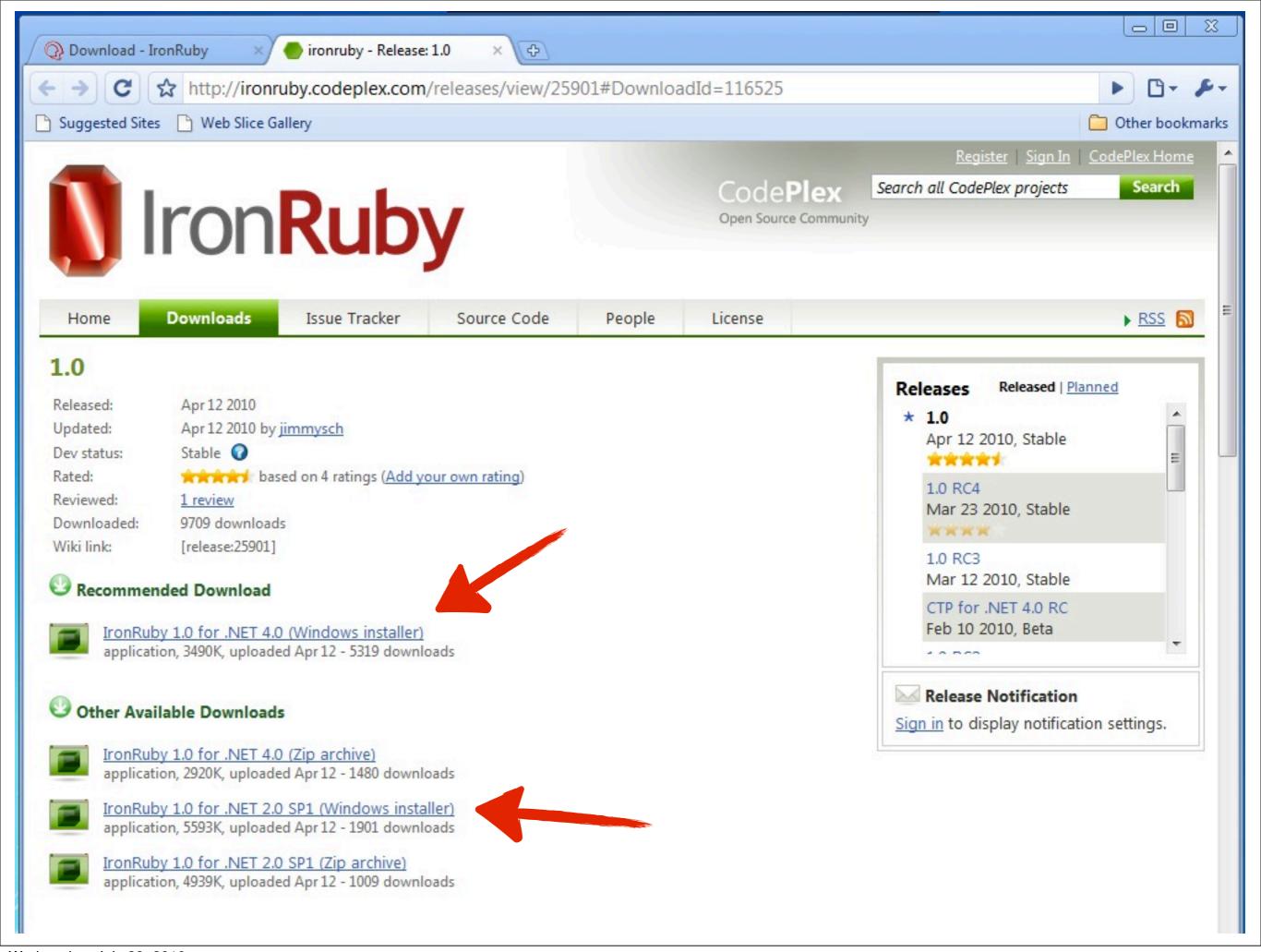
other DLR languages IronPython F#



start: IronRuby.net



start: IronRuby.net



start: IronRuby.net

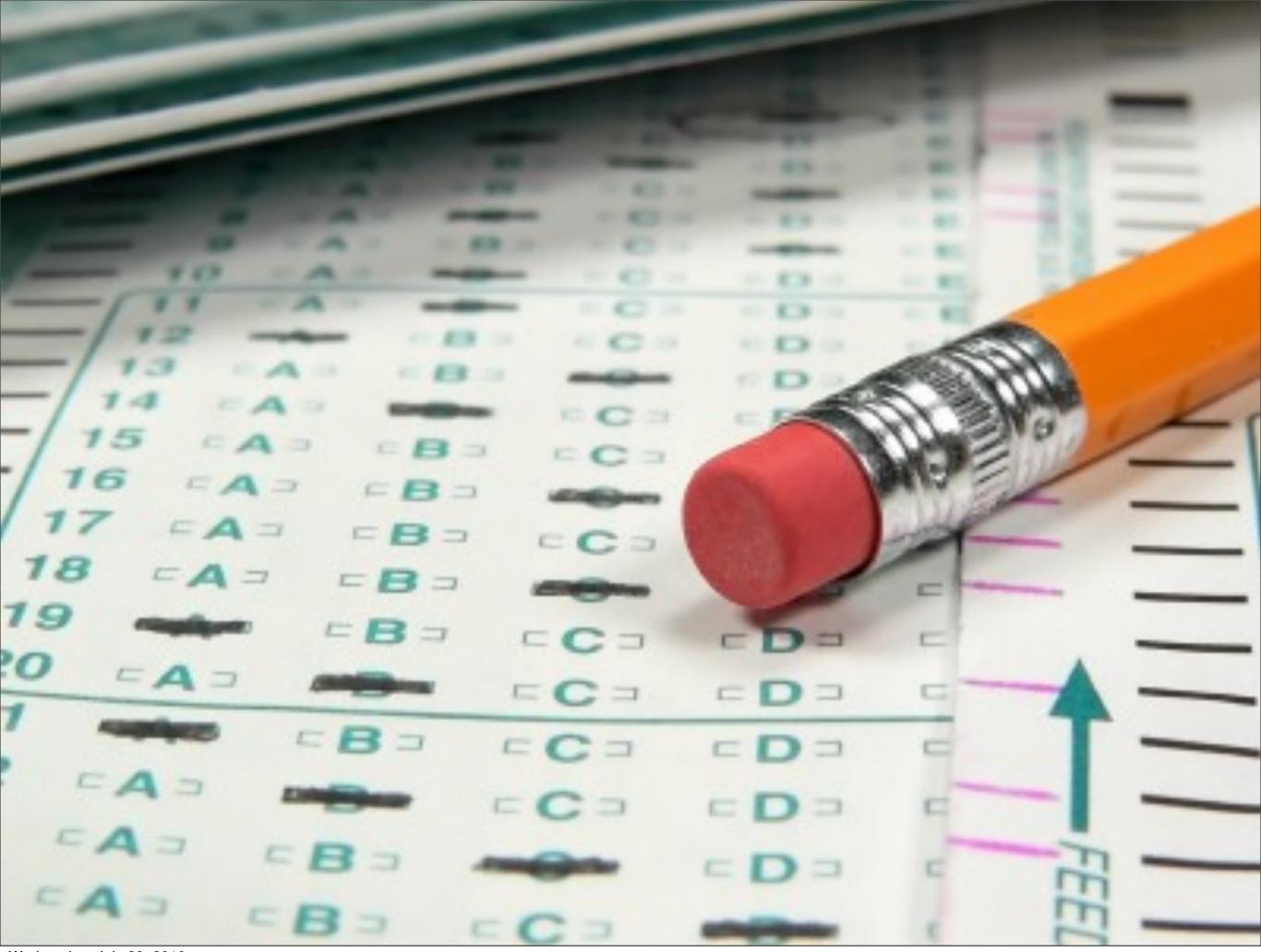


prepend an i => igem, iirb, ir instead of ruby performance not awesome rails/sinatra need rack for IIS C extensions (bson in mongo) no need to get to the CLR, don't use it

## <code>

Wednesday, July 28, 2010

first bit of code loading datetime loading monopoly object monkey patch something



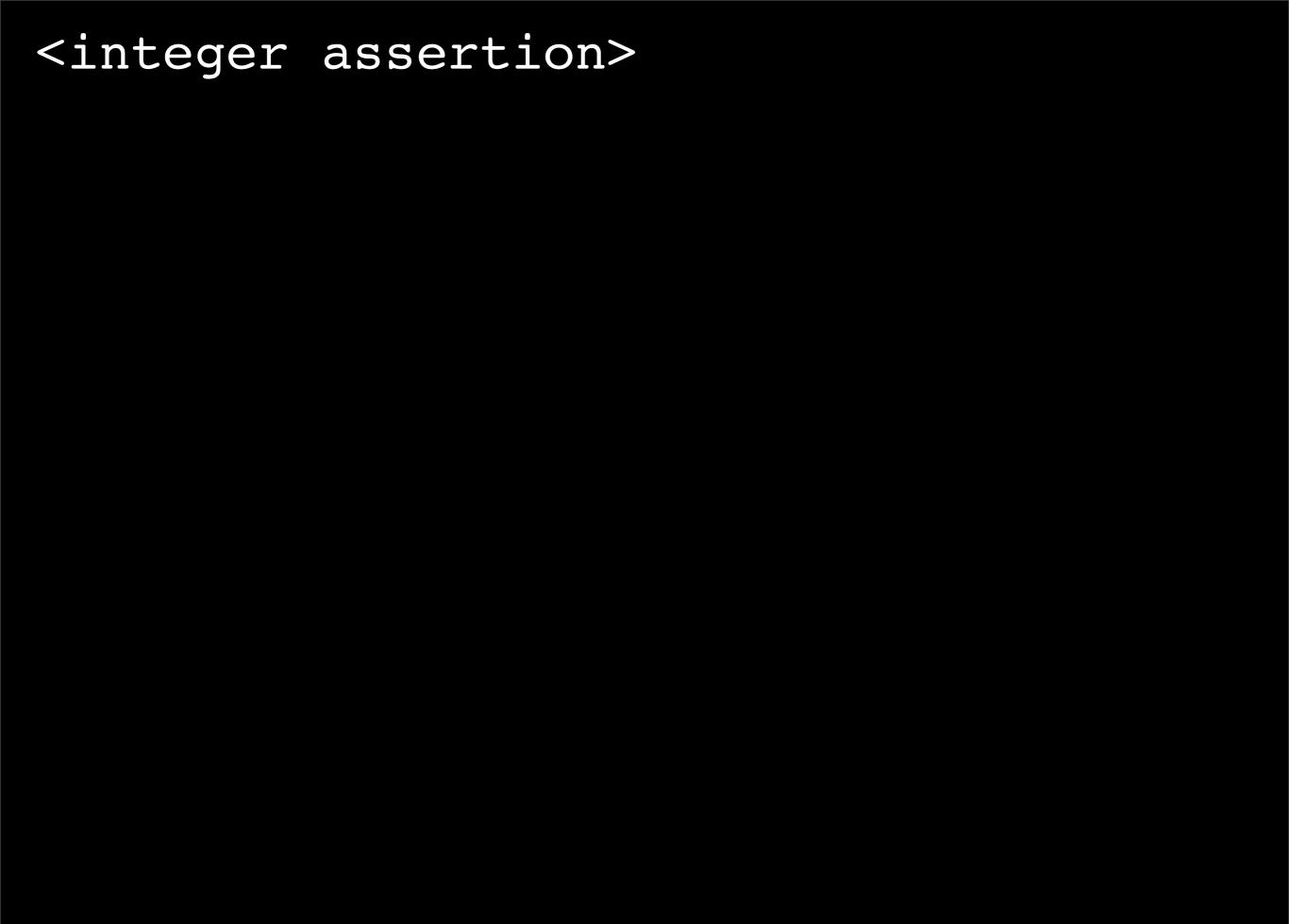
essence v. ceremony (neal ford)
Raiders of the Lost Ark
zero based arrays?
c#/kpl example w/Brendan



```
public class When_verifying_the_nunit_environment
  Player p;
  [SetUp]
  public void Because()
    p = new Player();
  [Test]
  public void then_the_player_should_not_be_null()
    Assert.That(p, Is.Not.Null);
```

```
describe 'when verifying the environment' do
before :each do
@p = Player.new
end
```

```
it 'should create a CLR object from the Monopoly project' do @p.should_not be_nil end end
```



<integer assertion>

Assert.That(\_player.AccountBalance, ls.EqualTo(125));

Wednesday, July 28, 2010

<integer assertion>

Assert.That(\_player.AccountBalance, ls.EqualTo(125));

@player.account\_balance.should be(125)





Assert.That(\_message, Is.EqualTo("Payment successful"));

Wednesday, July 28, 2010

<string assertion>

Assert.That(\_message, Is.EqualTo("Payment successful"));

@message.should == "Payment successful"



```
public class MockRepository: IBankRepository
  public bool PutMoneyInTheBankWasCalled { get; set; }
  public void PutMoneyInTheBank(int amount)
    this.PutMoneyInTheBankWasCalled = true;
  public bool TakeMoneyFromTheBank(int amount)
    return amount < 500;
  public int CheckFundsAvailable()
    throw new NotImplementedException();
```

#### class MockRepository include Monopoly::IBankRepository attr accessor:put money in the bank was called def take money from the bank (amount) amount < 500 end def put\_money\_in\_the\_bank(amount) @put money in the bank was called = true end

end

```
private Player _player;
private BankerService _service;
private MockRepository _repo;

[SetUp]
public void before_each()
{
    __player = new Player { AccountBalance = 875 };
    _repo = new MockRepository();
    _service = new BankerService(_repo);
}
```

#### before :each do

- @player = Player.new
- @player.account\_balance = 875
- @repo = MockRepository.new
- @service = BankerService.new @repoend

### <mocks v2>

```
private Player _player;
private BankerService _service;
private Mock<IBankRepository> _mock;
[SetUp]
public void before_each()
   _player = new Player { AccountBalance = 875 };
  mock = new Mock<|BankRepository>();
   _mock.Setup(m => m.TakeMoneyFromTheBank(550)).Returns(<mark>false</mark>);
   _mock.Setup(m => m.TakeMoneyFromTheBank(<mark>lt</mark>.IsAny<<mark>int</mark>>()))
           .Returns(true);
   _service = new BankerService(_mock.Object);
```

```
before :each do

@player = Player.new
@player.account_balance = 875

@isolation = Isolation.for IBankRepository
@isolation.when_receiving(:take_money_from_the_bank).return(true)
@isolation.when_receiving(:take_money_from_the_bank)
.with(550).return(false)

@service = BankerService.new @isolation
end
```



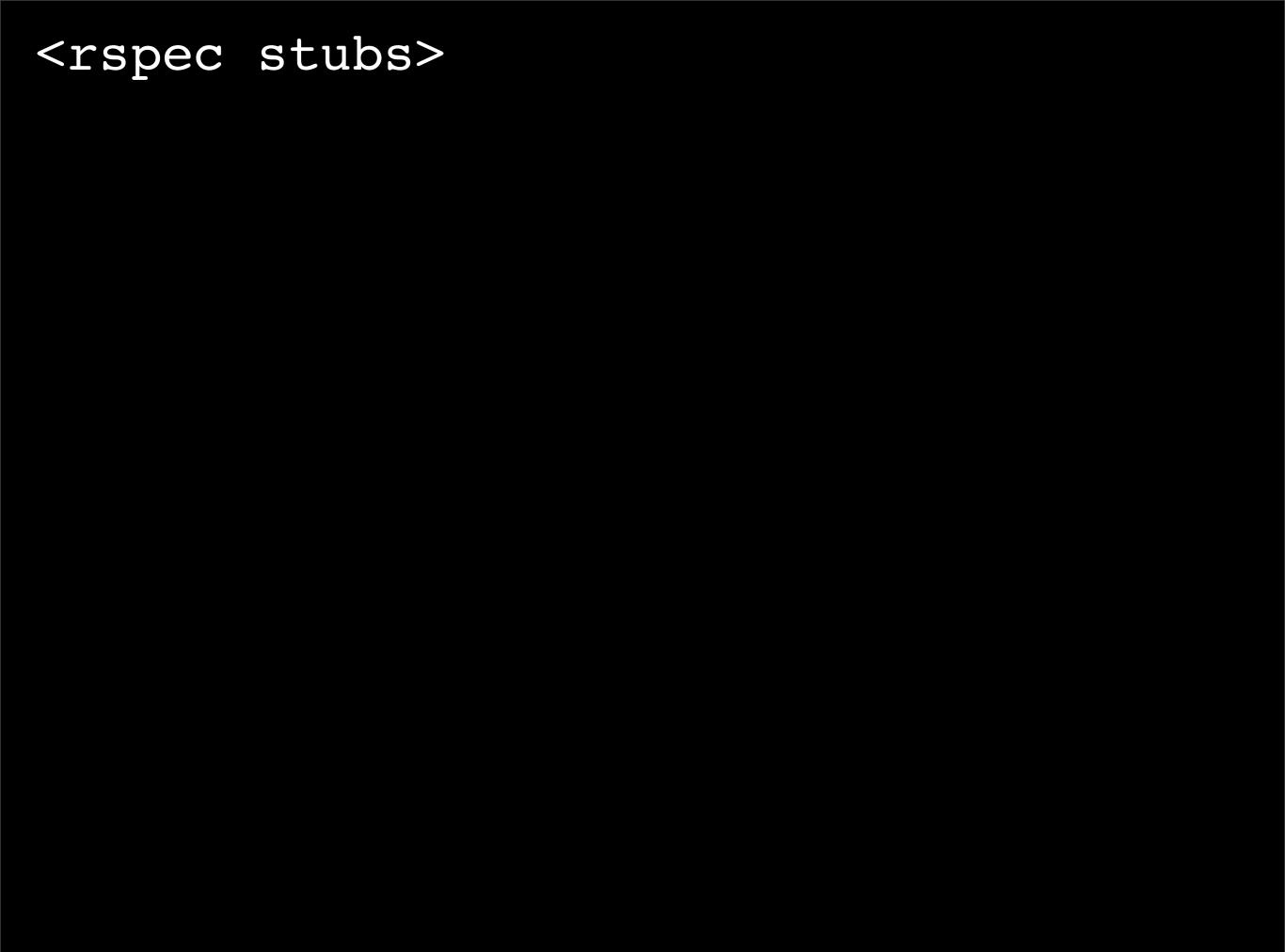
<mock verification>

\_mock.Verify(m => m.PutMoneyInTheBank(It.IsAny<int>()), Times.AtLeastOnce());

<mock verification>

\_mock.Verify(m => m.PutMoneyInTheBank(It.IsAny<int>()),
Times.AtLeastOnce());

@isolation.did\_receive?(:put\_money\_in\_the\_bank).with(50)



rspec stub

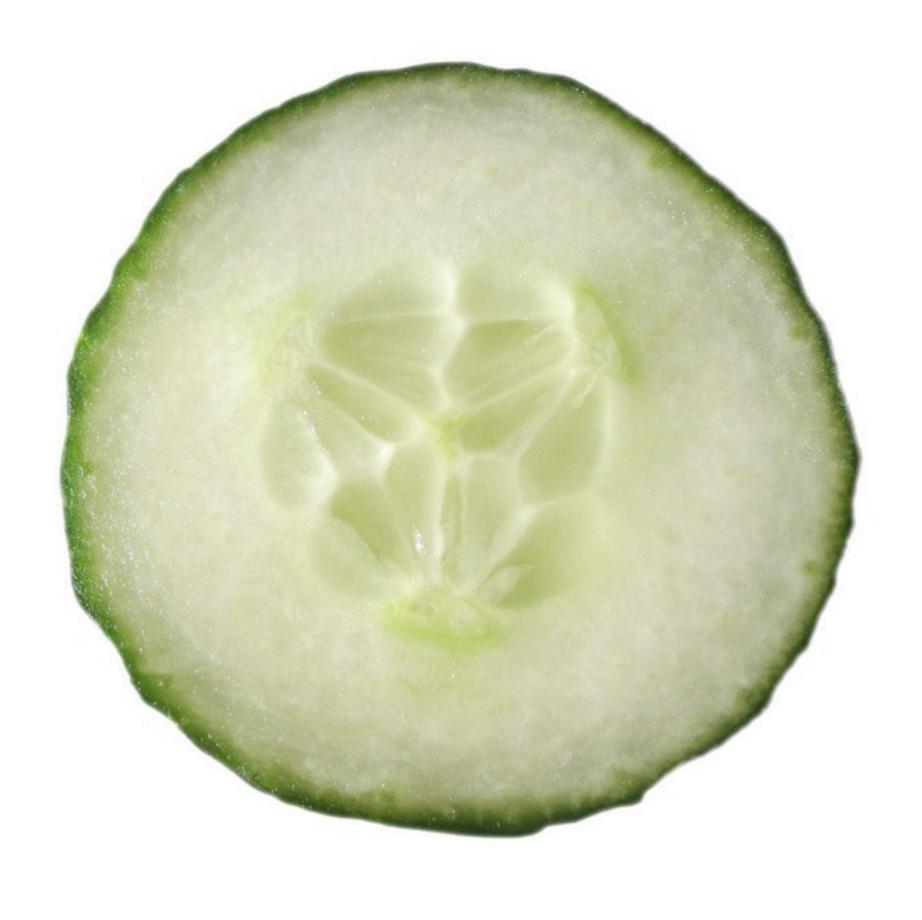
## <rspec stubs>

class StubbedRepo include Monopoly::IPropertyRepository end

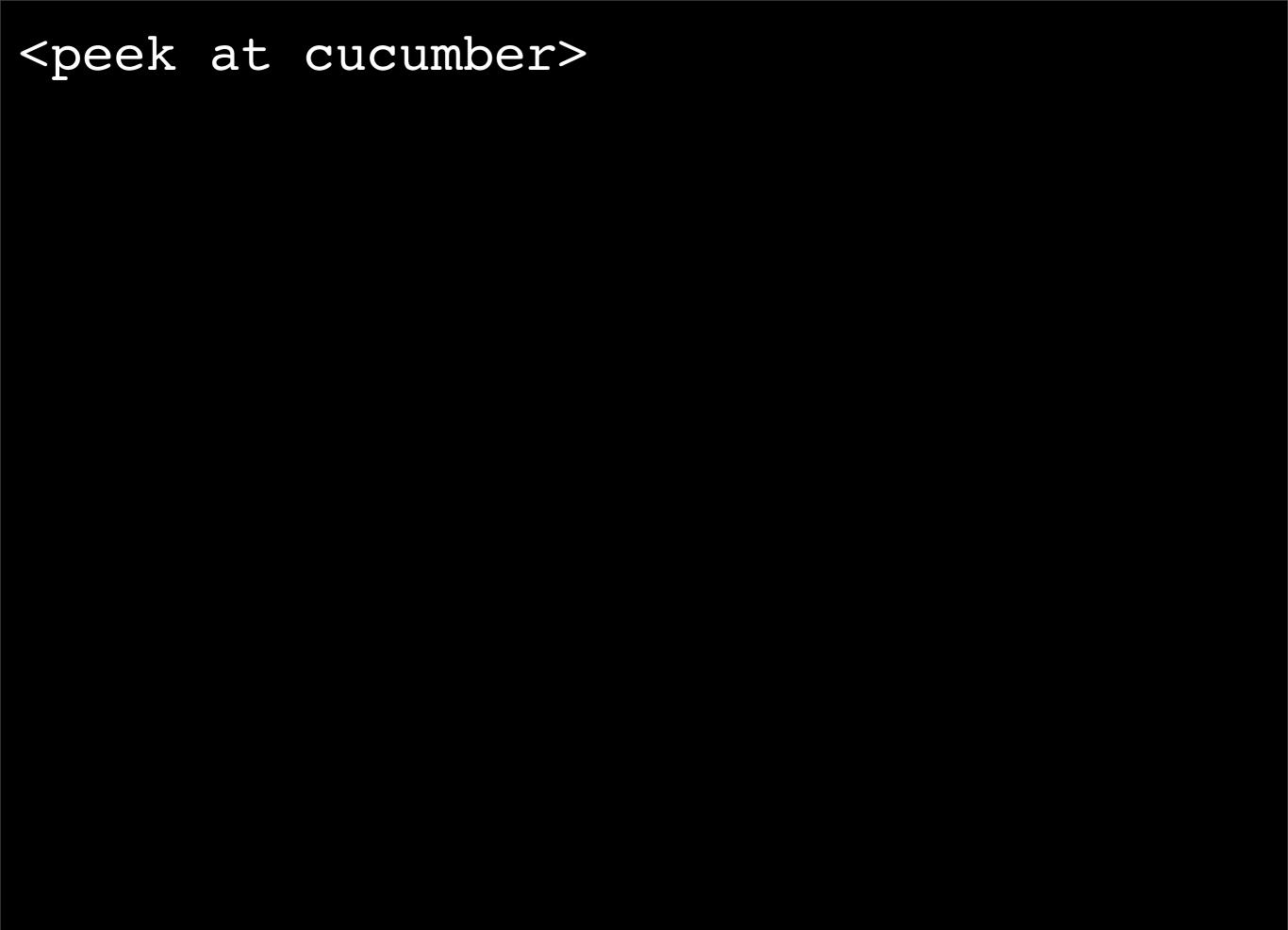
<rspec stubs>

class StubbedRepo include Monopoly::IPropertyRepository end

@repo = StubbedRepo.new
@repo.stub!(:get\_all\_properties).and\_return(@all\_properties)



use older version of Cucumber (0.6.4) not cuke for nuke



## <peek at cucumber>

Scenario: Checking my environment
Given the player "Car" is initialized to start a game
When I check the player balance
Then the account balance should be \$"2000"

## <peek at cucumber>

```
Scenario: Checking my environment
Given the player "Car" is initialized to start a game
When I check the player balance
Then the account balance should be $"2000"
```

```
Given /^the player "([^\"]*)" is initialized to start a game$/ do |player_name|
@player = Player.new
@player.name = player_name
end

When /^I check the player balance$/ do
@balance = @player.account_balance
end

Then /^the account balance should be "([^\\"]*)"$/ do |balance|
@balance.should be(balance)
end
```







## Tim Wingfield

tim@timwingfield.com

blog.timwingfield.com

@timwingfield

github.com/timwingfield/TestingSeaPound