Spain.js

7pm: @ggalmator leads a REFACTORING DOJO

@ggalmazor
Hola!

BuntPlanet

DreamingMinds

AgileSpain

BuntPlanet

DreamingMinds

Katayunos

AgileSpain

BuntPlanet
DreamingMinds

Merendojos

Katayunos

AgileSpain

BuntPlanet
DreamingMinds

The Mêlée

Merendojos

Coding Dojo

A Coding Dojo is a meeting where a bunch of coders get together to work on a programming challenge. They are there have fun and to engage in DeliberatePractice in order to improve their skills.

http://codingdojo.org

Coding Kata

A code kata is an exercise in programming which helps a programmer <u>hone</u> their <u>skills</u> through <u>practice and repetition</u>

Remember that the point of the kata is not arriving at a correct answer. <u>The point is the stuff you learn along the way</u>.

http://codekata.pragprog.com
Dave Thomas

Refactoring

"Disciplined technique for restructuring an existing body of code, altering its internal structure without changing its external behavior" undertaken in order to improve some of the nonfunctional attributes of the software

http://en.wikipedia.org/wiki/Code_refactoring

Benefits of refactoring your code

reduced complexity

improved code readability

improved maintainability

improved expressiveness

improved extensibility

Typical refactoring targets

duplicated/boilerplate code

bad naming

huge piles o' code

fuzzy concerns/responsibilities

Things to have in mind while refactoring

SOLID

DRY

Design Patterns

Good naming

Things to have in mind while refactoring

How are you going to guarantee that you won't break anything?

How do you tell if you're changing only the desing, not the function?

Never trust humans

Cover your code

(test it!)

SOLID desing cheatsheet

S: Single responsibility principle

O: Open (for extension) closed (for modification) principle

L: Liskov substitution (with subtypes) principle

I: Interface segregation principle
(many client-specific better than one all-purpose)

D: Dependency (upon abstractions, not concretions) inversion principle

http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata

Hi and welcome to team Gilded Rose. As you know, we are a small inn with a prime location in a prominent city ran by a friendly innkeeper named Allison. We also buy and sell only the finest goods. Unfortunately, our goods are constantly degrading in quality as they approach their sell by date. We have a system in place that updates our inventory for us. It was developed by a no-nonsense type named Leeroy, who has moved on to new adventures. Your task is to add the new feature to our system so that we can begin selling a new category of items. First an introduction to our system:

- All items have a SellIn value which denotes the number of days we have to sell the item
- All items have a Quality value which denotes how valuable the item is
- At the end of each day our system lowers both values for every item

http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata

Pretty simple, right? Well this is where it gets interesting:

 - Once the sell by date has passed, Quality degrades twice as fast
 - The Quality of an item is never negative
 - "Aged Brie" actually increases in Quality the older it gets
 - The Quality of an item is never more than 50
 - "Sulfuras", being a legendary item, never has to be sold or decreases in Quality
 - "Backstage passes", like aged brie, increases in Quality as it's SellIn value approaches; Quality increases by 2 when there are 10 days or less and by 3 when there are 5 days or less but Quality drops to 0 after the concert

We have recently signed a supplier of conjured items. This requires an update to our system:

 - "Conjured" items degrade in Quality twice as fast as normal items

http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata

Feel free to make any changes to the UpdateQuality method and add any new code as long as everything still works correctly. However, do not alter the Item class or Items property as those belong to the goblin in the corner who will insta-rage and one-shot you as he doesn't believe in shared code ownership (you can make the UpdateQuality method and Items property static if you like, we'll cover for you). Your work needs to be completed by Wednesday, July 11, 2012 08:00:00 AM PST.

Just for clarification, an item can never have its Quality increase above 50, however "Sulfuras" is a legendary item and as such its Quality is 80 and it never alters.

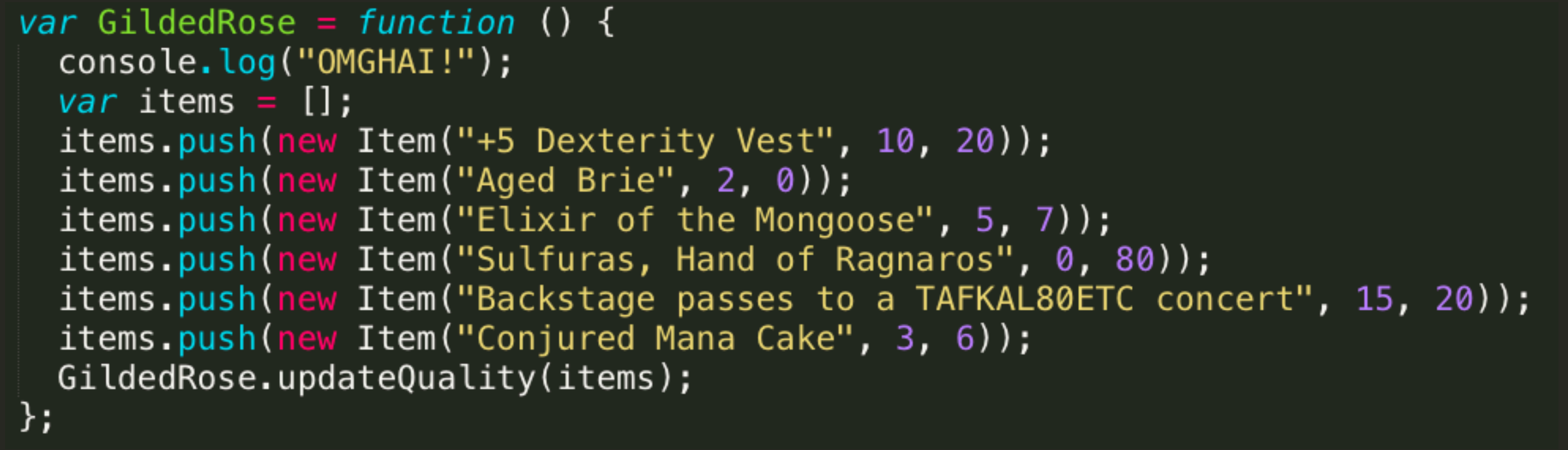
http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata

```
var Item = function (name, sellIn, quality) {
  this.name = name;
  this.sellIn = sellIn;
  this.quality = quality;
};
```

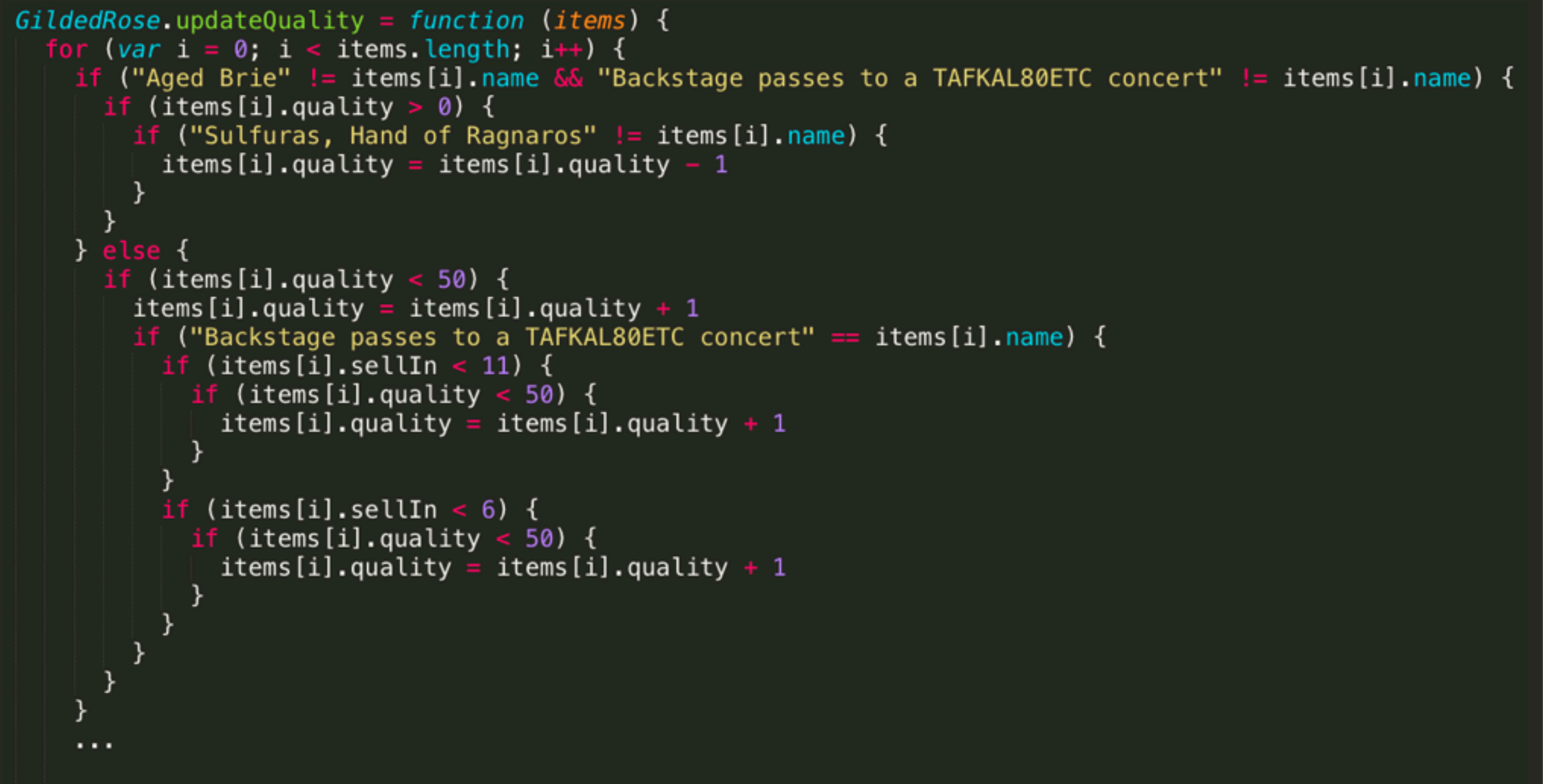
http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata


```
var GildedRose = function () {
  console.log("OMGHAI!");
  var items = [];
  items.push(new Item("+5 Dexterity Vest", 10, 20));
  items.push(new Item("Aged Brie", 2, 0));
  items.push(new Item("Elixir of the Mongoose", 5, 7));
  items.push(new Item("Sulfuras, Hand of Ragnaros", 0, 80));
  items.push(new Item("Backstage passes to a TAFKAL80ETC concert", 15, 20));
  items.push(new Item("Conjured Mana Cake", 3, 6));
  GildedRose.updateQuality(items);
};
```

http://github.com/ggalmazor/GildedRose_Jasmine

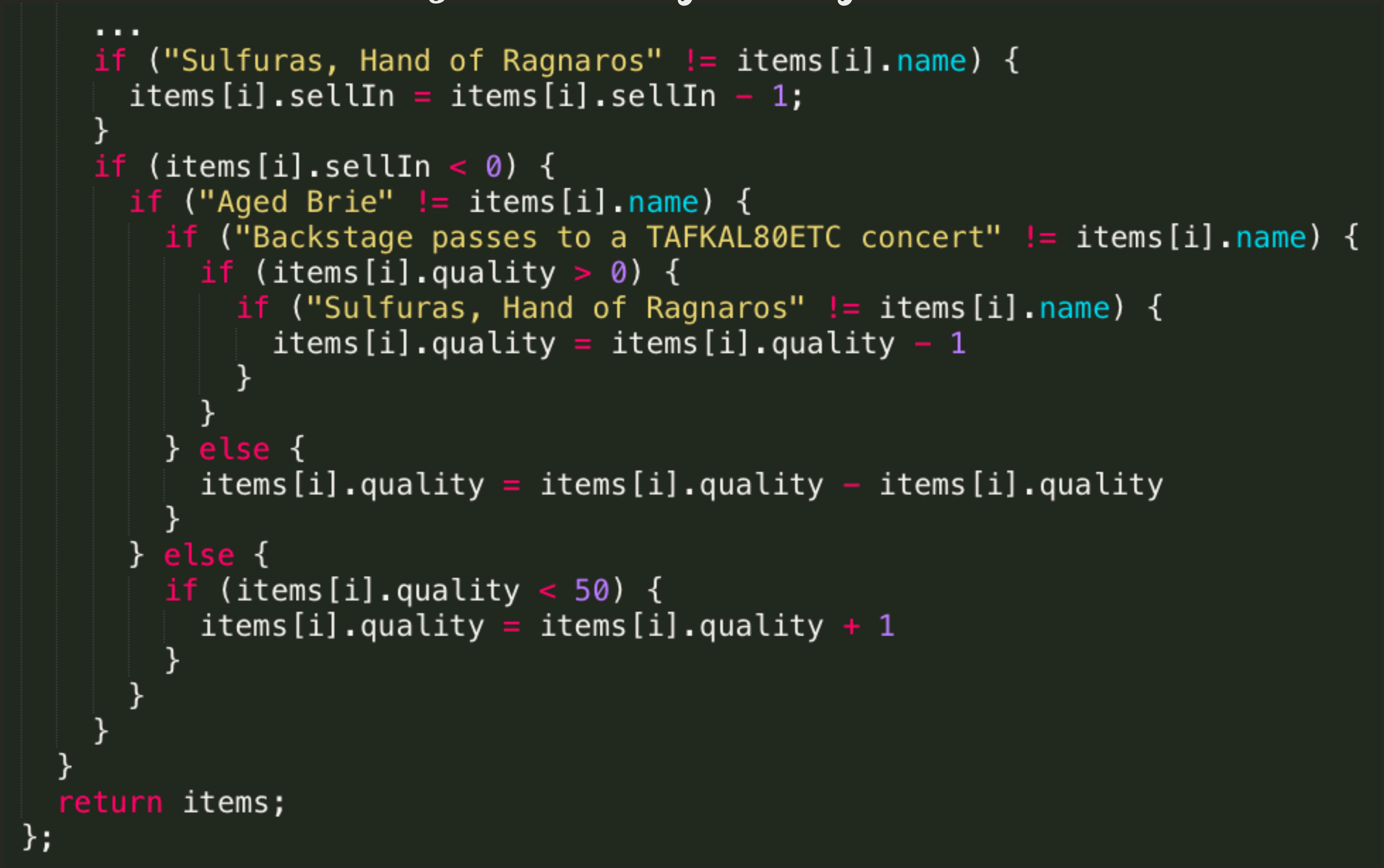
GildedRose Refactoring Kata



```
GildedRose.updateQuality = function (items) {
  for (var i = 0; i < items.length; i++) {
    if ("Aged Brie" != items[i].name && "Backstage passes to a TAFKAL80ETC concert" != items[i].name) {
      if (items[i].quality > 0) {
        if ("Sulfuras, Hand of Ragnaros" != items[i].name) {
          items[i].quality = items[i].quality - 1
        }
      }
    } else {
      if (items[i].quality < 50) {
        items[i].quality = items[i].quality + 1
        if ("Backstage passes to a TAFKAL80ETC concert" == items[i].name) {
          if (items[i].sellIn < 11) {
            if (items[i].quality < 50) {
              items[i].quality = items[i].quality + 1
            }
          }
          if (items[i].sellIn < 6) {
            if (items[i].quality < 50) {
              items[i].quality = items[i].quality + 1
            }
          }
        }
      }
    }
    ...
```

http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata



```
    ...
    if ("Sulfuras, Hand of Ragnaros" != items[i].name) {
      items[i].sellIn = items[i].sellIn - 1;
    }
    if (items[i].sellIn < 0) {
      if ("Aged Brie" != items[i].name) {
        if ("Backstage passes to a TAFKAL80ETC concert" != items[i].name) {
          if (items[i].quality > 0) {
            if ("Sulfuras, Hand of Ragnaros" != items[i].name) {
              items[i].quality = items[i].quality - 1
            }
          }
        } else {
          items[i].quality = items[i].quality - items[i].quality
        }
      } else {
        if (items[i].quality < 50) {
          items[i].quality = items[i].quality + 1
        }
      }
    }
  }
  return items;
};
```

http://github.com/ggalmazor/GildedRose_Jasmine

GildedRose Refactoring Kata

```
Some of code smells that we spotted:
  - Lots of magic numbers, magic strings
  - Code repetition:
    - items[i]
    - items[i].quality = items[i].quality + 1
    - items[i].quality = items[i].quality - 1
    - items[i].quality < 50
    - items[i].quality > 0
    - items[i].sellIn < 0
    - "somestring" != items[i].name
    - etc.
  - If block cascading
  - Uncached items.length on for loop
  - GildedRose.updateQuality is not in GildedRose.prototype
  - The code on the constructor simply does not make sense
```

GildedRose Refactoring Kata

Proposed list of refactors:

- Cache items.length in first sentence on for loop
- Extract local var item = items[i] inside loop
- Replace magic strings and numbers with variables with meaningful names: maxQuality, cheesyItem, etc.
- Extract repeated quality modifying sentences into functions that receive the item as an argument
- Extract those sentences that check on item properties that are used in if conditions into functions that receive the item as an argument
- Simplify if blocks, remove else blocks
- Fix the constructor function to receive a list of items
- Fix the constructor function to not call updateQuality
- Extract everything inside the for loop into a function called updateItem that receives the item as an argument
- Deal with the Feature Envy antipattern we should have produced by now applying polymorphism, or aggregation (remember not to change Item object)

Extra ball

- ask yourself -

We know that your boss wants to sell Conjured items ASAP… what would you do?

Would you focus on refactoring counting that you will need more time to finish your job?

Would you do just enough to release the feature and then improve the code?

Randori how to

- the roles -

Driver

She's coding on a laptop connected to a large screen or proyector. She is focused on just one refactor at a time and runs the tests after every change to the code

Copilot

She takes notes in a large pad or blackboard with the suggestions that the Audience makes. She's responsible to arrange proposed refactors according to their size, from smaller to larger

Audience

Thinks and discusses about code smells, possible solutions and proposes refactors to the Copilot
Lets the Driver code :)

Randori how to

- the process -

Start by taking a look to the code and discuss it some minutes. Try to understand the shop's rules, etc. and get some refactors listed out.

And then, every five or ten minutes, test in green, etc.

1 The Driver returns to the Audience

2 The Copilot takes the lead as Driver

3 A person from the Audience steps up to be the Copilot

Randori how to

- remember -

Baby steps

Always try to make the smallest possible move

Run the tests

After each change, run them!

Arrange your refactor backlog

Always engage on the smallest listed refactor

When in doubt...

Tell the Copilot to write it on the backlog and take care of it on your turn.
Try not to endlessly discuss some particular topic (we love that, don't we?)

Ideas if you enjoyed the Refactoring Dojo

Repeat after some weeks with your friends

Repeat it at work with your colleagues

Make up another kata and post it somewhere we can try it

Engage with some random legacy code from work

Search for refactoring targets on projects at Bitbucket or GitHub

Ideas if you enjoyed the Refactoring Dojo

Organize a Dojo within your local development community

http://katayunos.com

http://meetup.com/madruby

http://madridrb.jottit.com

Thanks for watching!