



OBJECTIFY THE SCREENPLAY:

A STORY OF PATTERNS

About Me

Aneta Petkova

Automation QA Lead,
Honeywell Aerospace

aneta.petkova@honeywell.com
[linkedin.com/in/apetkova/](https://www.linkedin.com/in/apetkova/)

Overview

Journey #1: Back in time

- The PageObject pattern

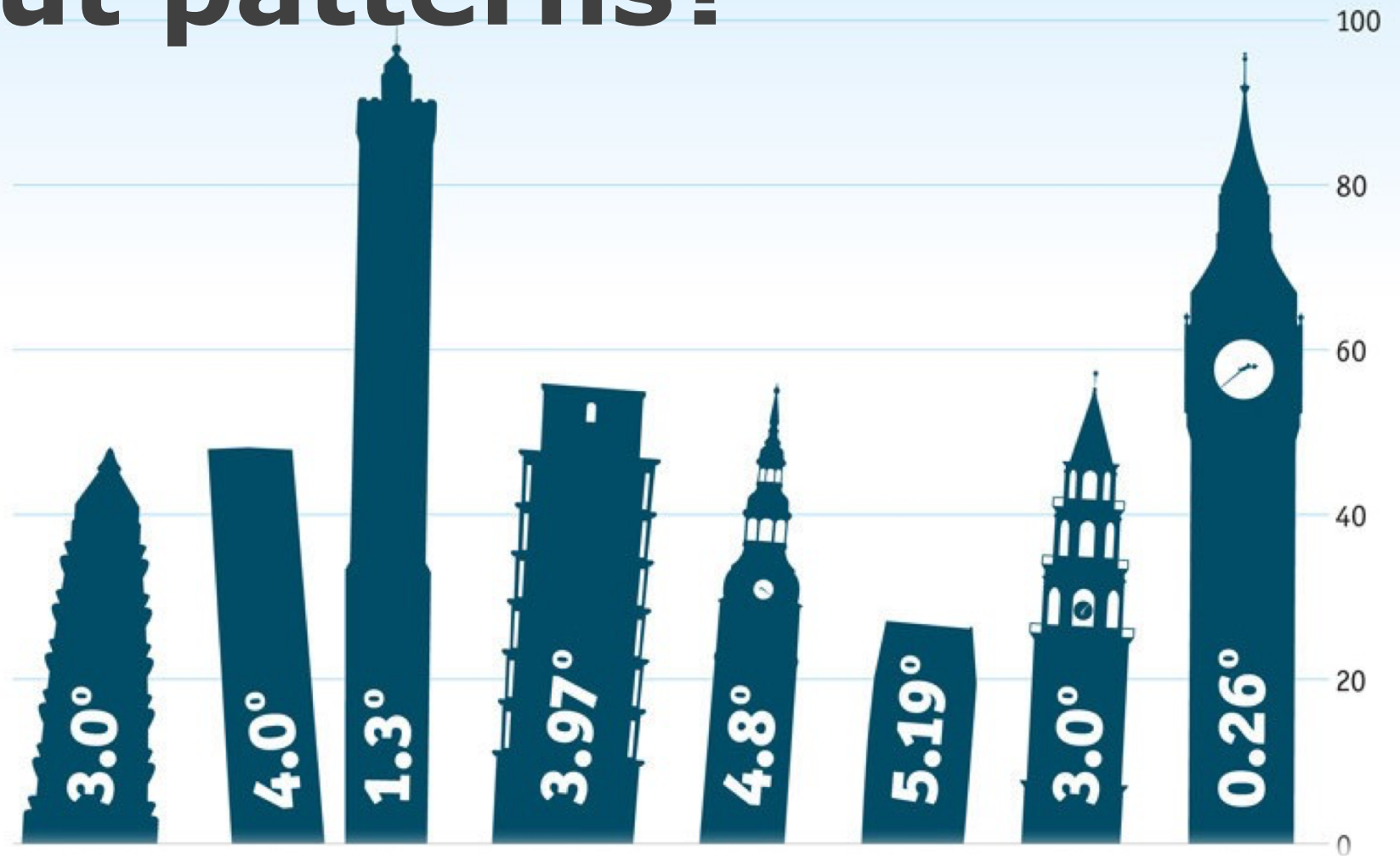
Journey #2: Into the now

- The Screenplay pattern

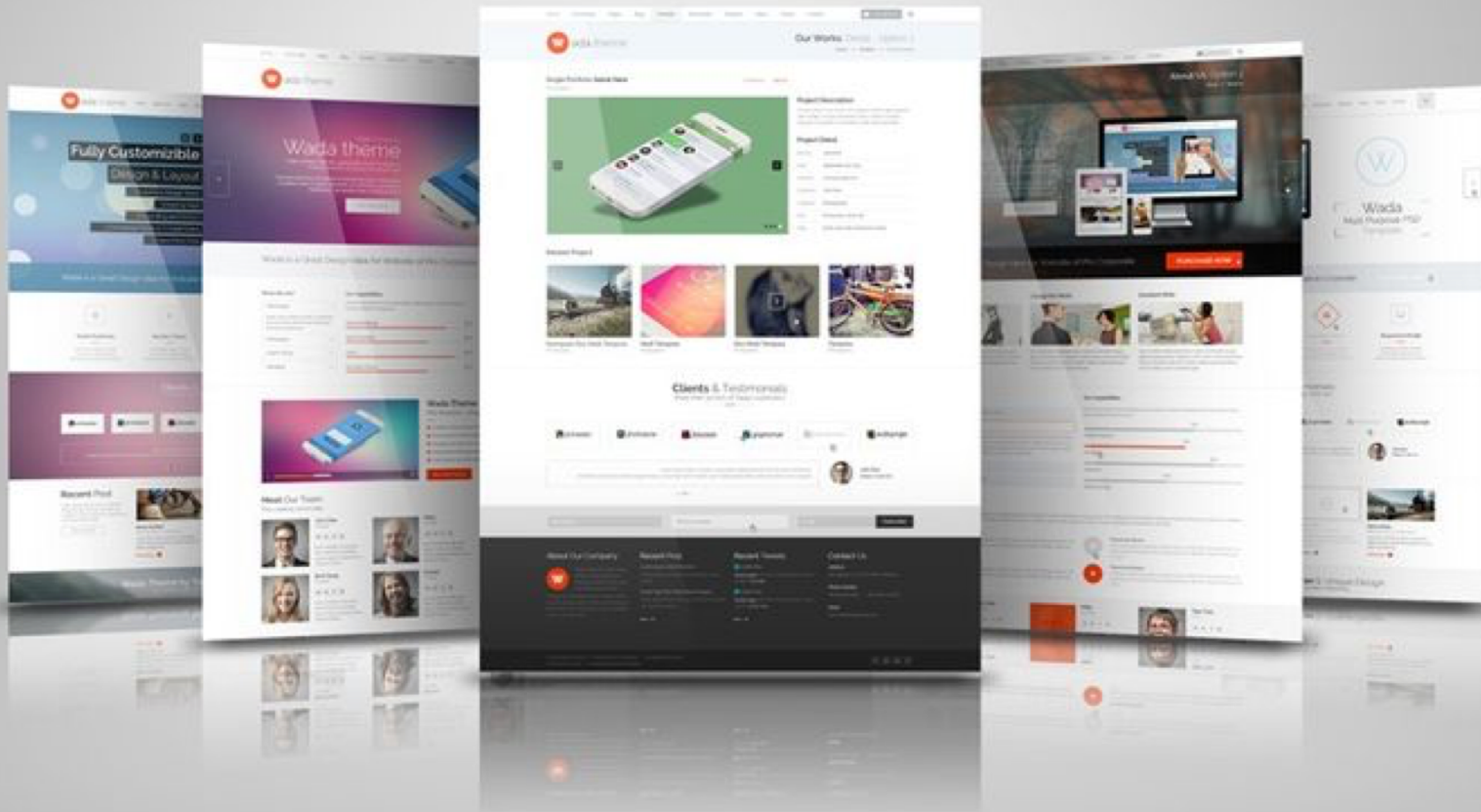
Journey #3: Looking at tomorrow

- No spoilers yet

Why should we care about patterns?



The PageObject

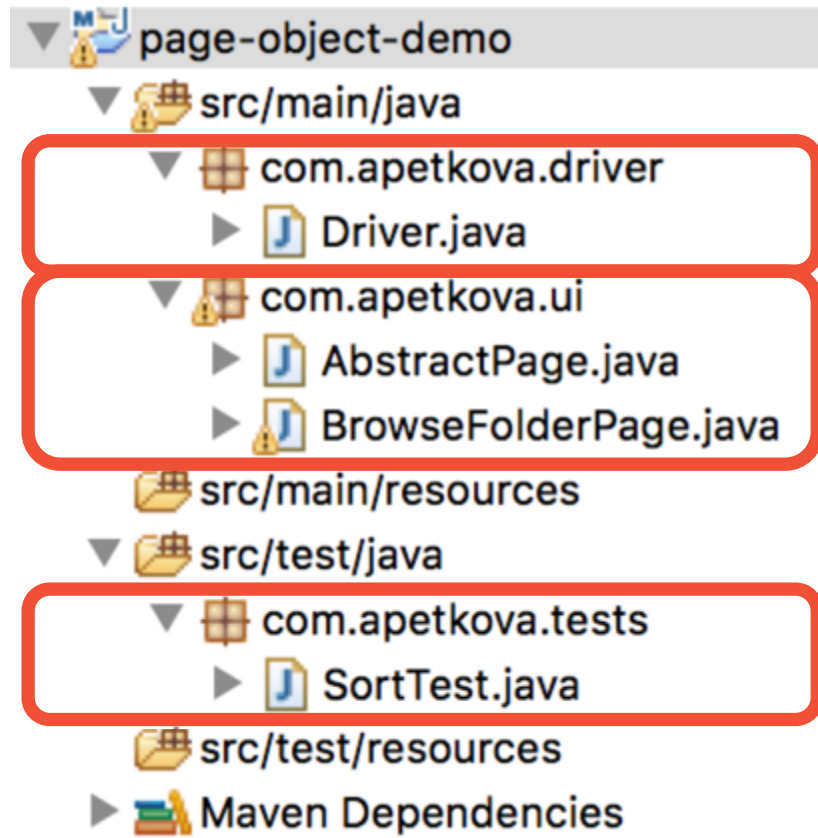


PageObject: concept

Separation: Keeping things tidy



PageObject: structure



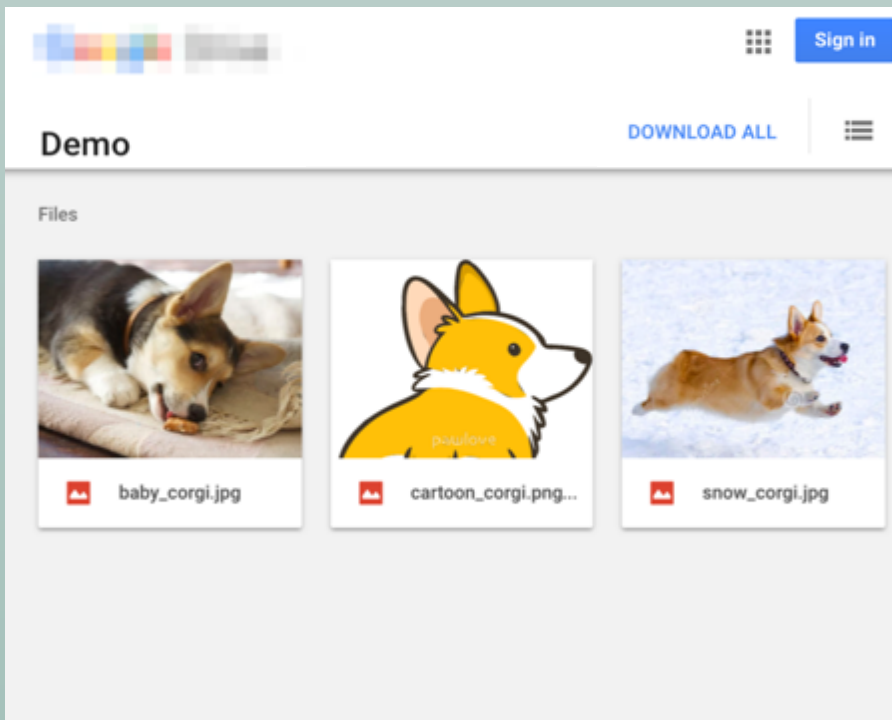
WebDriver config

Page Object classes

Test classes

PageObject

Web



Java

```
private WebElement  
    folderNameLabel;  
private WebElement  
    downloadAllLink;  
private List<WebElement>  
    files;  
  
public void sortByName();  
public void downloadAll();
```


PageObject

The test method

```
@Test
public void sortTest() {
    BrowseFolderPage page =
        new BrowseFolderPage();

    page.sortFilesByName();
    List<String> actualFileNames = page.getFileNames();
    List<String> expectedFileNames = new
        ArrayList<String>(actualFileNames);
    expectedFileNames.sort();
    Assert.assertEquals(actualFileNames, expectedFileNames,
        "File names do not match");
}
```

PageObject

The test method

```
@Test
public void sortTest() {
    BrowseFolderPage page = new BrowseFolderPage();
    page.sortFilesByName();
    List<String> actualFileNames = page.getFileNames();
    List<String> expectedFileNames = new
        ArrayList<String>(actualFileNames);
    expectedFileNames.sort();
    Assert.assertEquals(actualFileNames, expectedFileNames,
        "File names do not match");
}
```


PageObject

The t

```
@Test
public void sortTest() {
    BrowseFolderPage page =
    page.sortFilesByName();
    List<String> actualFileNames =
        page.getFileNames();

    List<String> expectedFileNames = new
        ArrayList<String>(actualFileNames);
    expectedFileNames.sort();
    Assert.assertEquals(actualFileNames, expectedFileNames,
        "File names do not match");
}
```



baby_corgi.jpg



cartoon_corgi.png...



snow_corgi.jpg

PageObject

The test method

```
@Test
public void sortTest() {
    BrowseFolderPage page = new BrowseFolderPage(driver);
    page.sortFilesByName();
    List<String> actualFileNames = page.getFileNames();
    List<String> expectedFileNames =
    new ArrayList<String>(actualFileNames);
    expectedFileNames.sort();
    Assert.assertEquals(actualFileNames, expectedFileNames,
        "File names do not match");
}
```

baby_corgi.jpg
cartoon_corgi.png
snow_corgi.jpg

PageObject

The test method

```
@Test
public void sortTest() {
    BrowseFolderPage page = new BrowseFolderPage();
    page.sortFilesByName();
    List<String> actualFileNames = page.getFileNames();
    List<String> expectedFileNames = new
        ArrayList<String>(actualFileNames);
    expectedFileNames.sort();
    Assert.assertEquals(actualFileNames,
        expectedFileNames, "File names do not match");
}
```

Payoff



Less code duplication

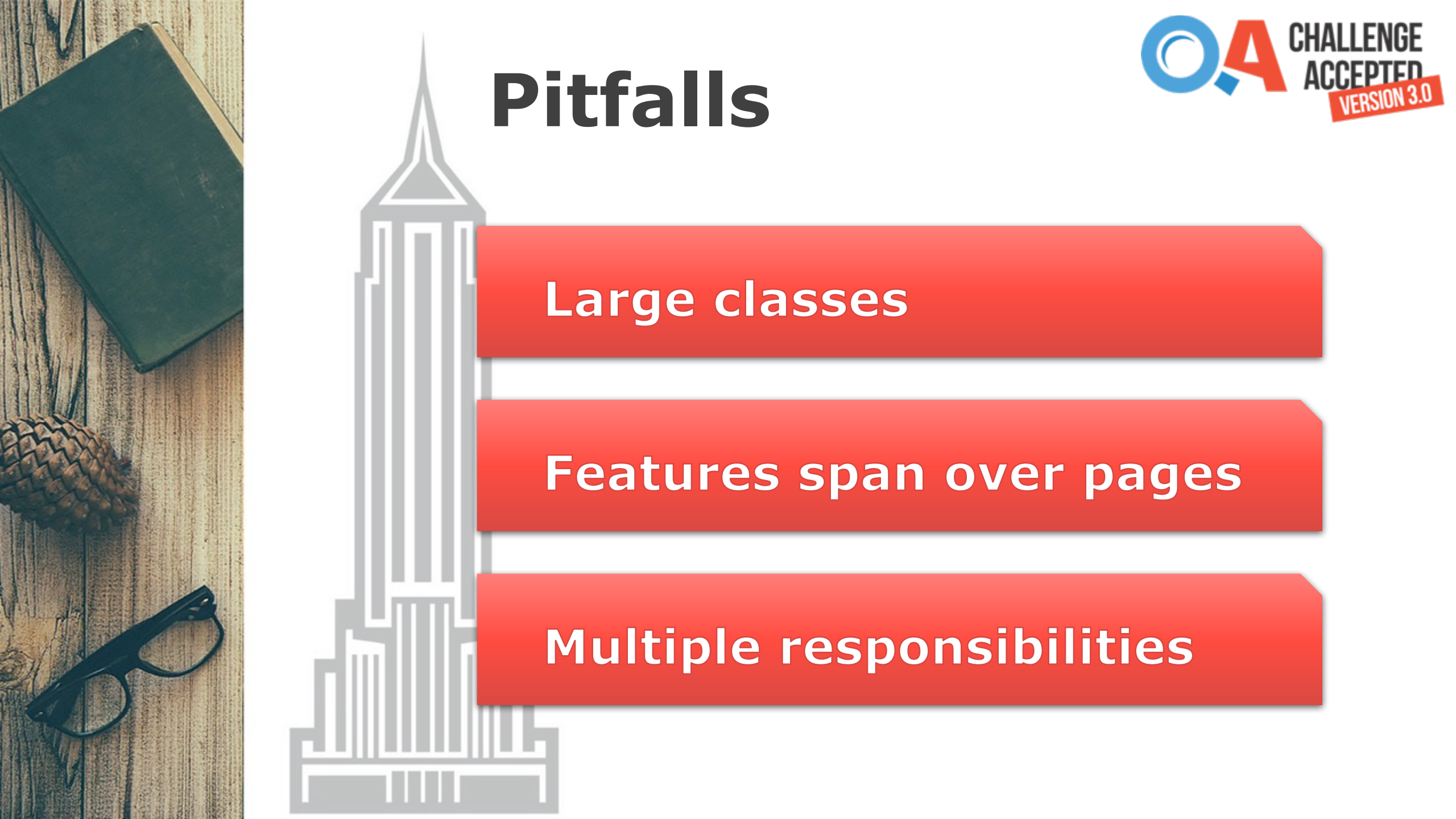


Robust and readable tests



Better maintainability

Pitfalls

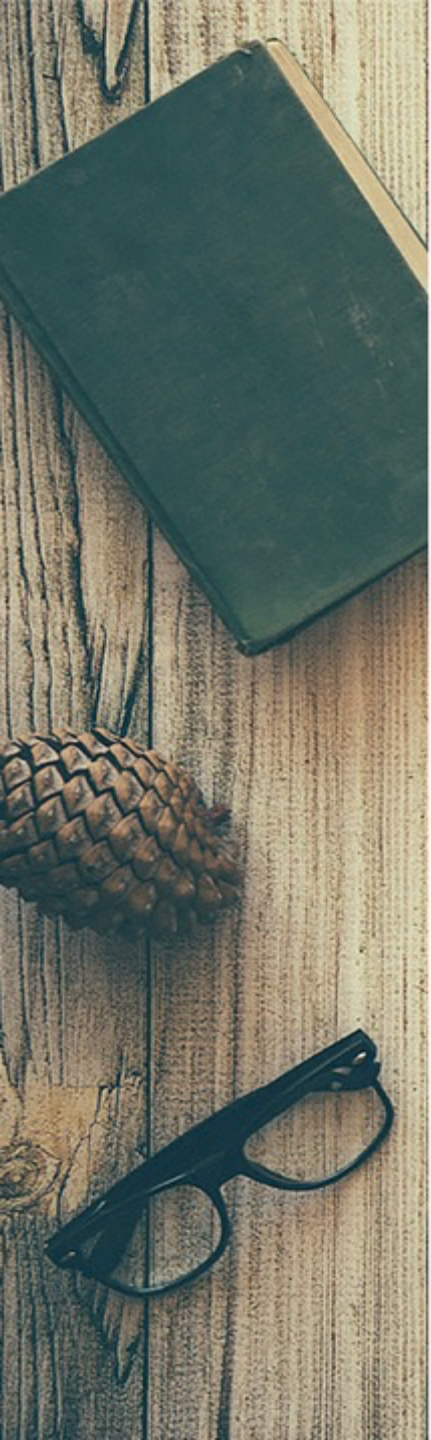


Large classes

Features span over pages

Multiple responsibilities

The PageObject



The Screenplay pattern

System functionality
before system
properties

Actions
before objects

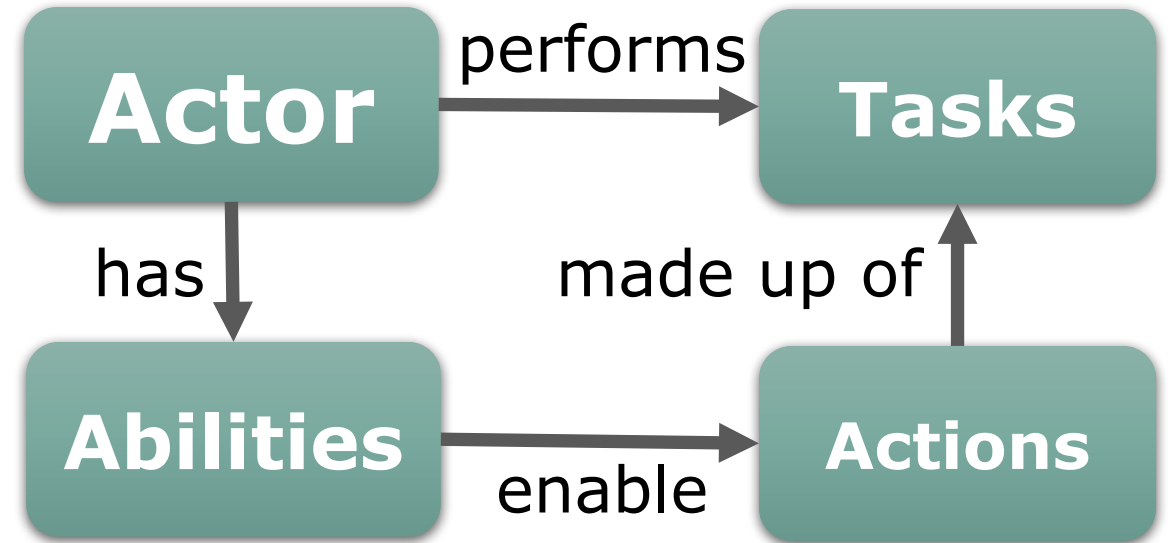


Screenplay: concept

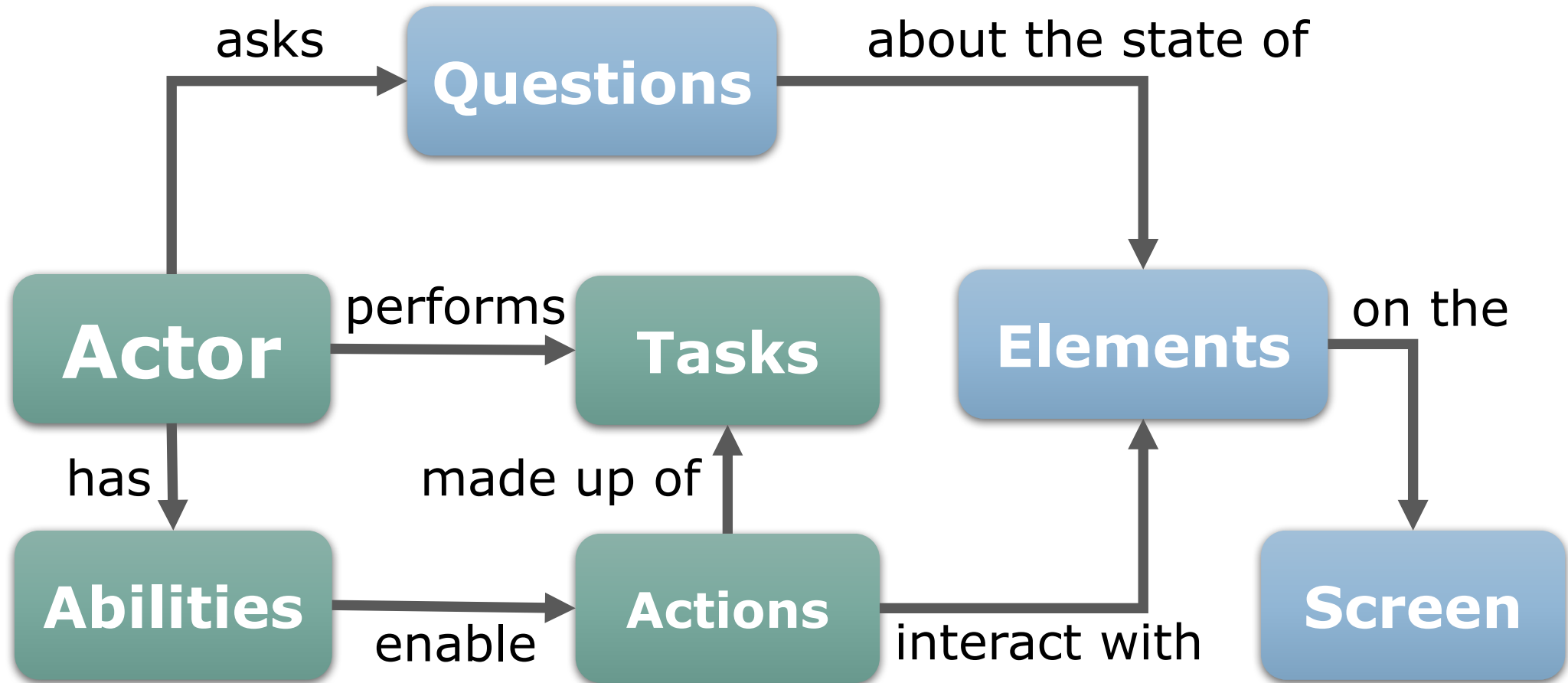


Actor

Screenplay: concept



Screenplay: concept



Screenplay: library



**“Automated Acceptance
Testing with Style”**

- and great reporting!

Screenplay: structure

- ▼  screenplay-gradle-demo
 - ▼  src/test/java
 - ▼  com.apetkova.features
 - ▶  SortFiles.java
 - ▼  com.apetkova.questions
 - ▶  FileList.java
 - ▼  com.apetkova.tasks
 - ▶  SortFilesInFolder.java
 - ▶  StartWith.java
 - ▼  src/main/java
 - ▼  com.apetkova.ui
 - ▶  BrowseFolderPage.java
 - ▶  JRE System Library [JavaSE-1.8]

Features -> tests

State checks

Tasks

Page Objects

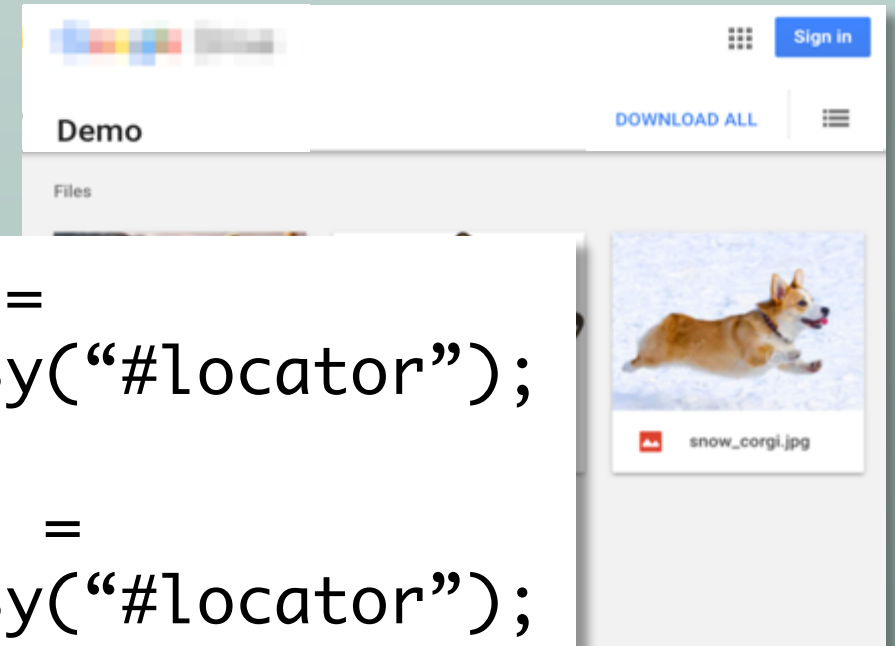
Screenplay

The PageObject class

```
public static Target FOLDER_NAME =  
the("Folder name label").locatedBy("#locator");
```

```
public static Target DOWNLOAD_ALL =  
the("Download All link").locatedBy("#locator");
```

```
public static Target FILES =  
the("List of files").locatedBy("#locator");
```



Screenplay

The test method

[illegible]

Screenplay

The test method

[illegible]

Screenplay

The test method

```
@Test
public void should_be_able_to_sort_his_files_by_name() {
    roland.wasAbleTo(StartWith.aPopulatedFolder());
    roland.attemptsTo(SortFilesInFolder.by("Name"));
    roland.should(seeThat(FileList.displayed(),
        is(FileList.orderedBy(system))));
}
```

Serenity report

Test Results: All Tests

1 test scenarios

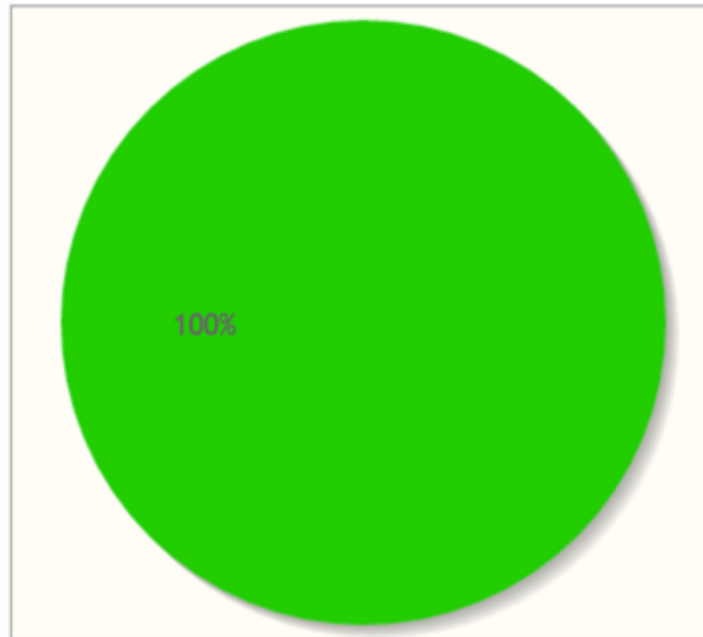
| 1 passed | 

Test Count

Weighted Tests

Distribution of tests (including rows in data-driven tests) by test result.

Show/Hide Pie Chart



Test Result Summary

Test Type	Total	Pass	Fail	Pending	Ignored
Automated	1	1 (100%)	0 (0%)	0 (0%)	0 (0%)
Manual	0	0 (0%)	0 (0%)	0 (0%)	0 (0%)
Total	1	1 (100%)	0 (0%)	0 (0%)	0 (0%)
Total Duration	9 seconds				

Related Tags

Stories

Sort Files

% Passed

100%



Test
count

1

Serenity report



Sort Files



Should Be Able To Sort His Files By Name

Steps	Screenshot	Outcome	Duration
 Roland starts with a populated folder		SUCCESS	6.67s
 Roland sorts files by Name		SUCCESS	2.11s
 Then file list should be ([baby_corgi.jpg, cartoon_corgi.png, snow_corgi.jpg])		SUCCESS	0.95s
SUCCESS 10.19s			

Payoff



Tests focus on features

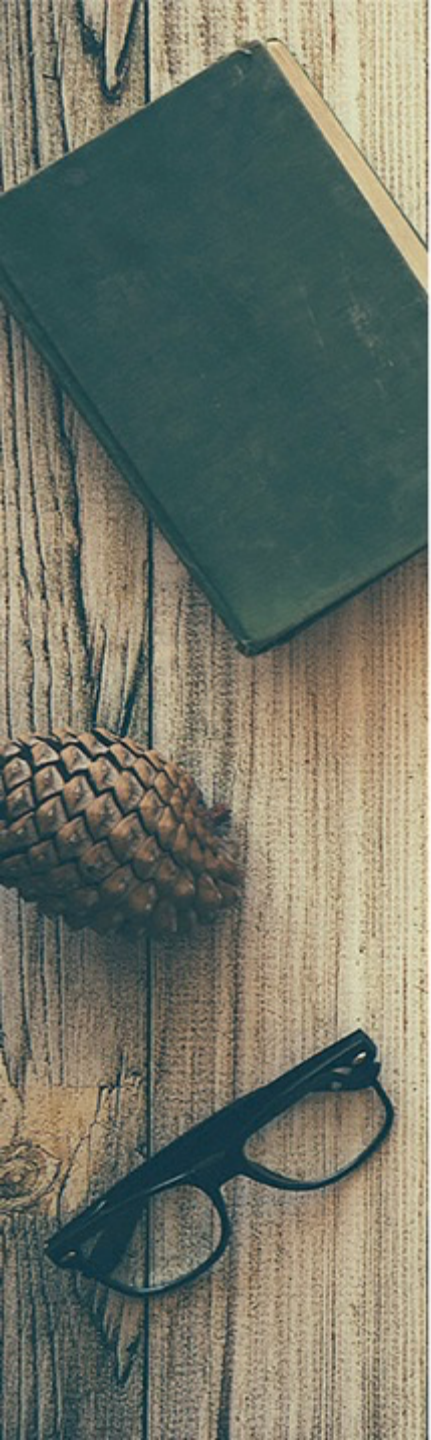
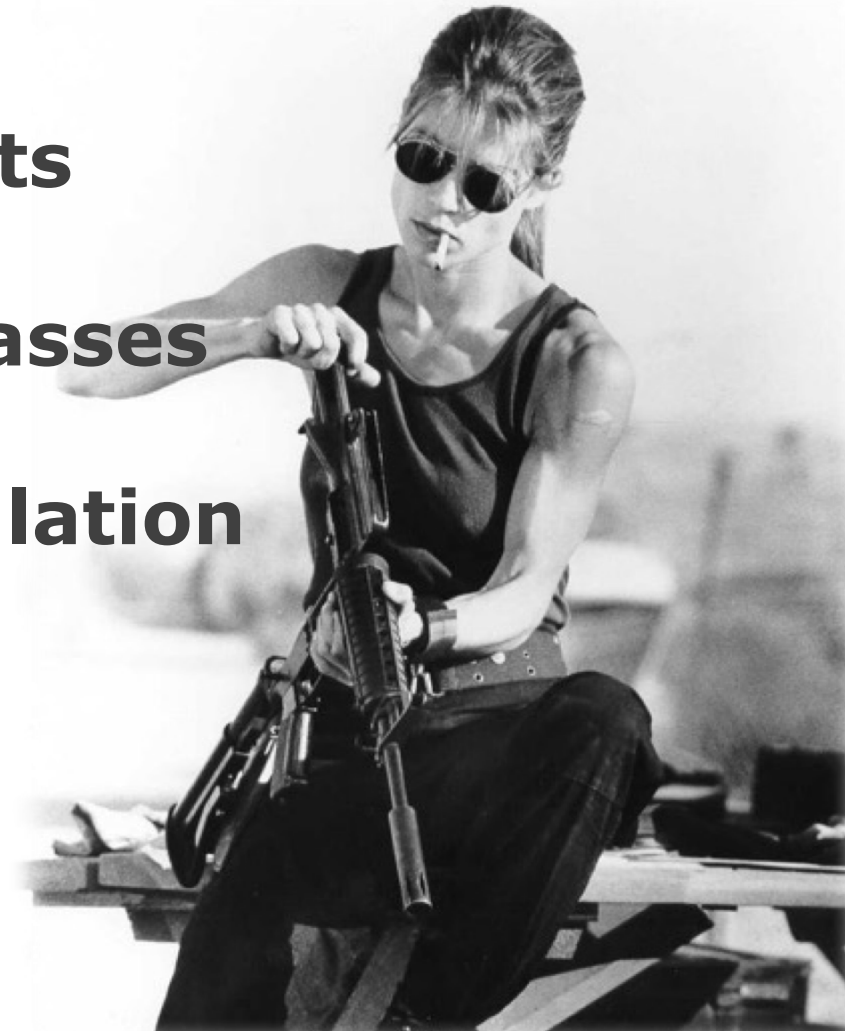
Single responsibility

Functional extensions

Great reporting

The middle ground

- **Focus on users, not asserts**
- **Keep tasks in separate classes**
- **Consider relaxed encapsulation**



**Define your own
journey!**





THANK YOU!

Q & A

aneta.petkova@honeywell.com
[linkedin.com/in/apetkova](https://www.linkedin.com/in/apetkova)