03.21.18

Crossing The Streams

Multi – Platform Framework Design

Jason Edstrom

Magenic // Fast Forward

1

About This Guy

- Magenic Technologies
  - 4 years
- Automation Lead
- Mobile Enthusiast
  - Mobile Hardware Repair
  - Mobile Developer
  - Mobile Automation
- Chronic Early Adopter
- Notorious Edge Case

2

Mobile Application Types

- Native
  - iOS
    - Objective C
    - Swift
  - Android
    - Kotlin
    - Java
- Translated
  - React Native, Xamarin, Flutter
- Hybrid
  - Some or All HTML in an app package
  - PhoneGap, Cordova, Ionic
- Web

Criteria

- Shared/Single Test Class
- Cross Platform
- Application Divergence
- Performance
- Shared Code

Stretch Goals

- Readability
  - Diagramming
  - Test
  - Page Object
- Design Consistency Enforcement
- Use Object Oriented Principles
- Traceability
  - All errors are happening in the class context that is relevant
- Not just Application divergence, but device type divergence

Design Scenarios

- Split Framework
- Conditional Page Models
- Page Factory
- Abstract Parent

Split Framework

"You stay on your side of the room and I'll stay on my side"

7

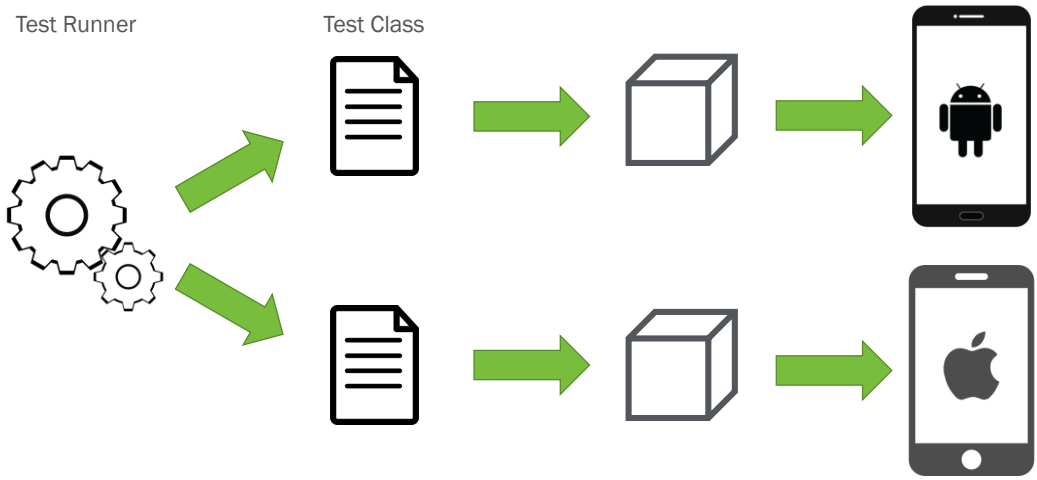
Split Framework - Single Test Case

Android Test Class

iOS Test Class

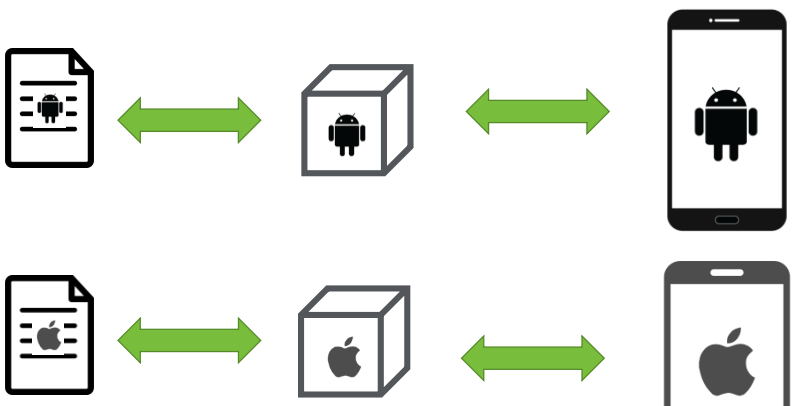
8

Split Framework – Cross Platform



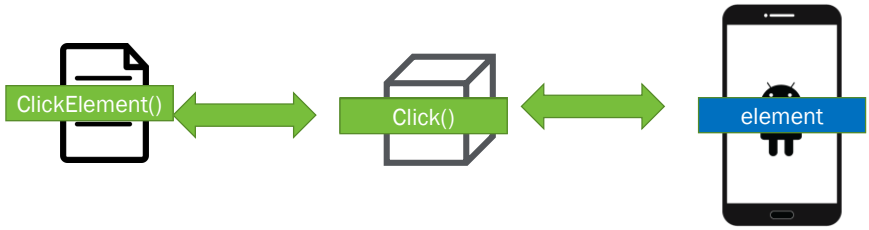
9

Split Framework – Application Divergence



10

Split Framework - Performance

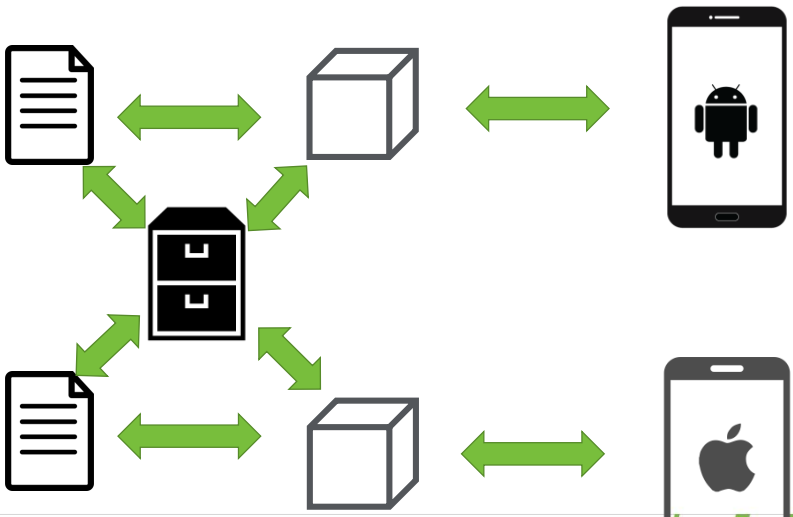


Magenic Fast Forward

11

11

Split Framework – Shared Code



Magenic Fast Forward

12

12

Split Framework - Benefits

- Out of the Box Platform Specific Tests
- Abstraction is not required for UX divergence
- Enforced Single Responsibility on Test and Page Object level

Split Framework - Consequences

- Test Coverage Desync
- Reporting the test results becomes a separate test point
  - Test Point – attributes of test
    - Platform
    - Version
    - App Version
- Code Sharing is painful
  - High Number of static methods
  - Duplicated code everywhere

Split Framework Checklist

Single Test Class - No
Cross Platform - Meh
App Divergence - Yes
Performance - Yes
Shared Code - No

Conditional Page Model

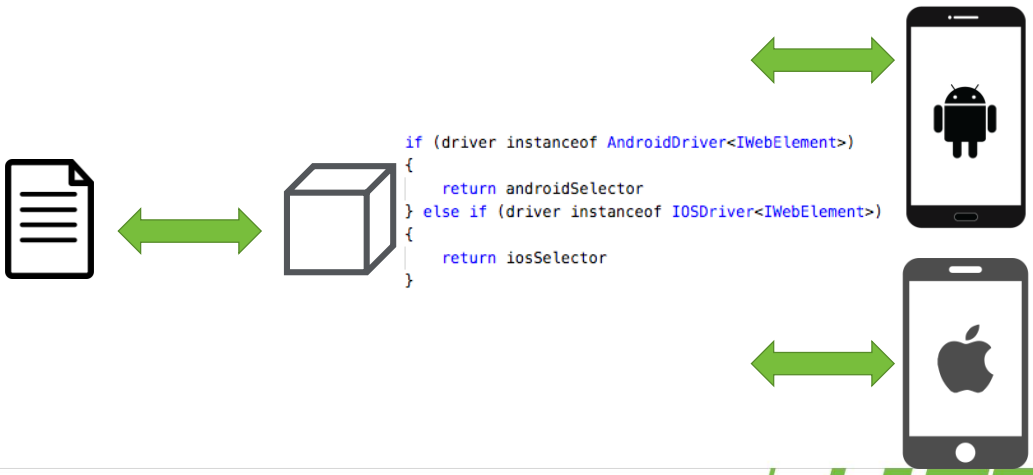
"10 gallons of stuff in a 5 gallon bucket"

Conditional Page Model - Single Test Case

iOS Test Class
Android Test Class

17

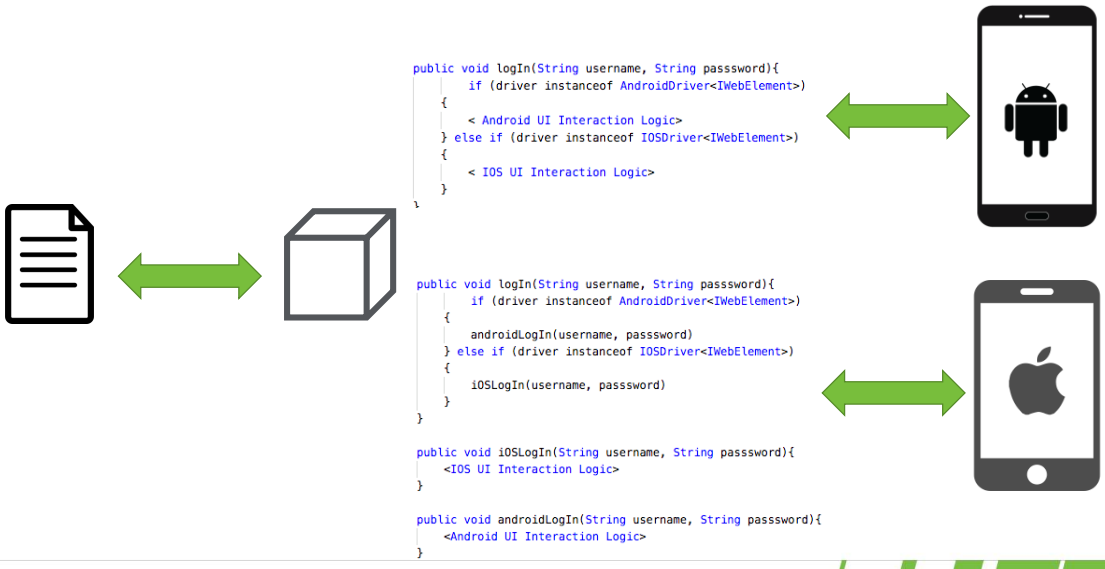
Conditional Page Model - Cross Platform



```
if (driver instanceof AndroidDriver<IWebElement>)
{
    return androidSelector
} else if (driver instanceof IOSDriver<IWebElement>)
{
    return iosSelector
}
```


18

Conditional Page Model - Application Divergence



```
public void logIn(String username, String passsword){
        if (driver instanceof AndroidDriver<IWebElement>)
    {
        < Android UI Interaction Logic>
    } else if (driver instanceof IOSDriver<IWebElement>)
    {
        < IOS UI Interaction Logic>
    }
}
```

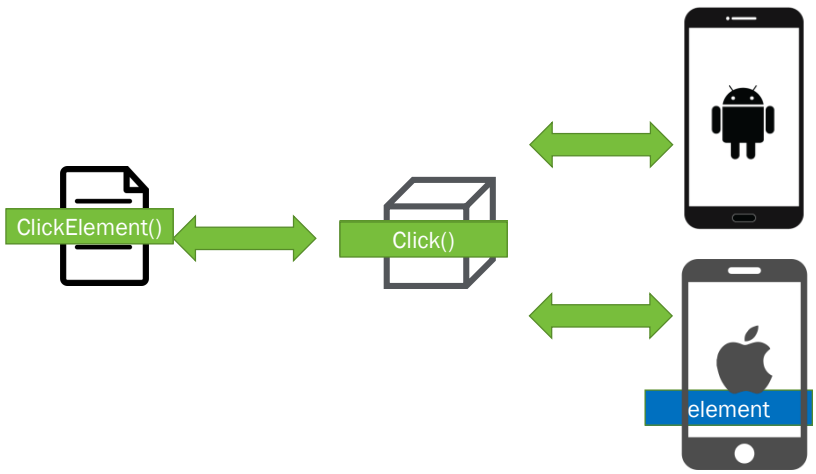
```
public void logIn(String username, String passsword){
        if (driver instanceof AndroidDriver<IWebElement>)
    {
        androidLogIn(username, passsword)
    } else if (driver instanceof IOSDriver<IWebElement>)
    {
        iOSLogIn(username, passsword)
    }
}

public void iOSLogIn(String username, String passsword){
    <IOS UI Interaction Logic>
}

public void androidLogIn(String username, String passsword){
    <Android UI Interaction Logic>
}
```

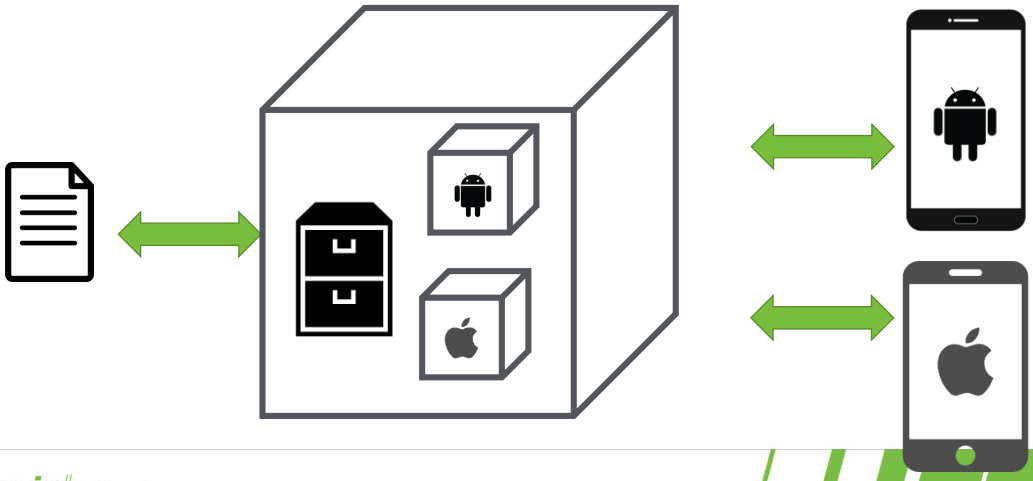

19

Conditional Page Model - Performance



20

Conditional Page Model - Shared Code



21

Conditional Page Model - Benefits

- Test reporting becomes a run configuration detail
- Page Object Models have more shared code
- Page Object Models can primarily use the parent driver
- Test Coverage stays consistent across platforms

22

Conditional Page Model - Consequences

- Extra work needed for Platform Specific tests
- Page Object Models stop respecting Single Responsibility
  - Platform Specific drivers
  - Platform Specific methods
  - Platform Specific selectors
- Conditional statement at every instance of UX Divergence
  - Which has the high probability of being every selector

Conditional Page Model

- Single Test Class - Yes
- Cross Platform - Yes
- App Divergence - Meh
- Performance - Yes
- Shared Code - Meh

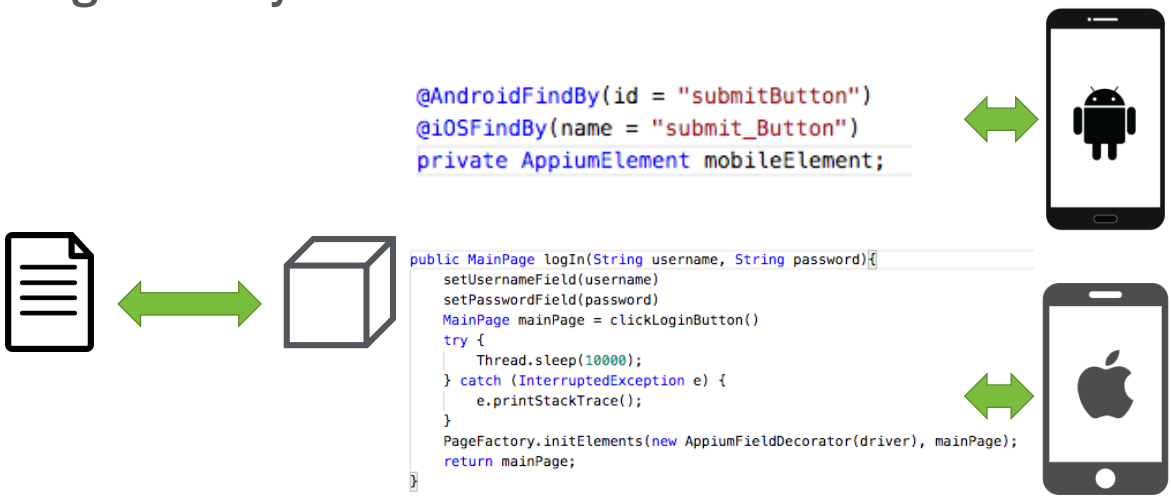
Page Factory

"Buying a Papa Johns store when all you want is a slice of pizza"

Page Factory - Single Test Case

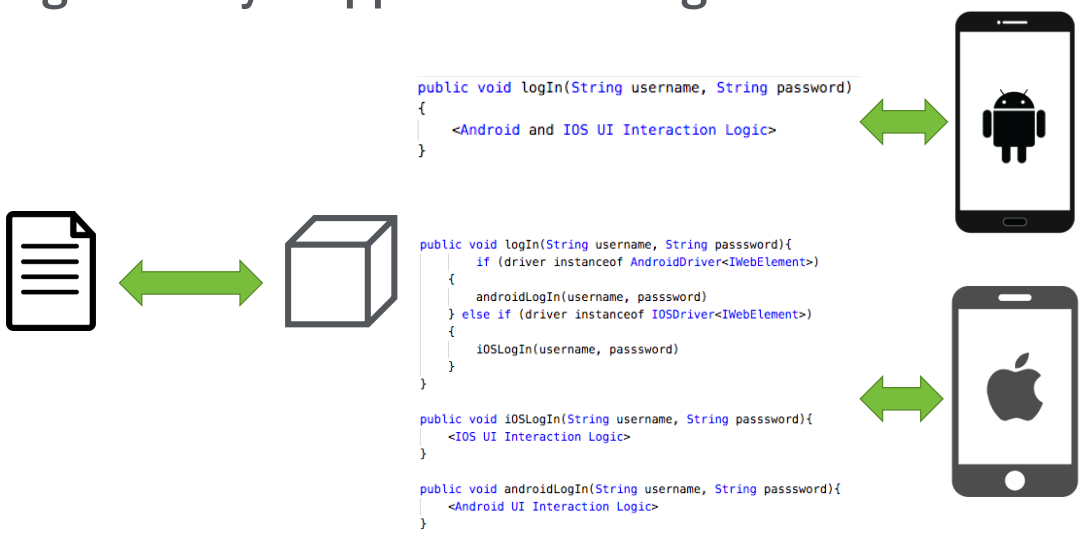
iOS Test Class
Android Test Class

Page Factory - Cross Platform



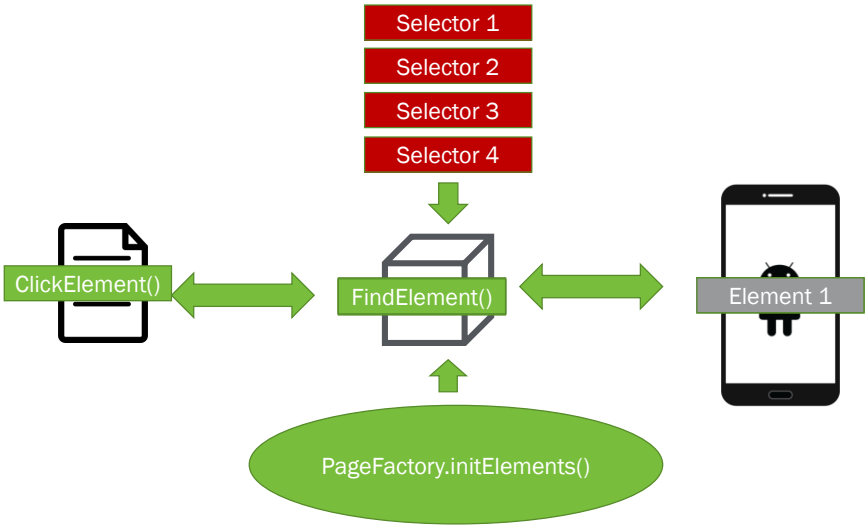
27

Page Factory – Application Divergence



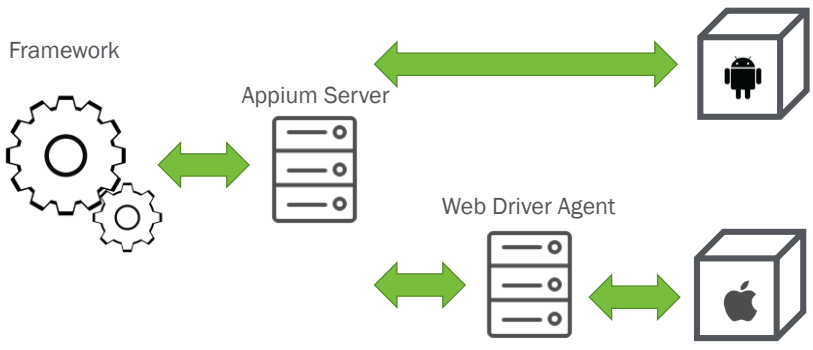
28

Page Factory - Performance



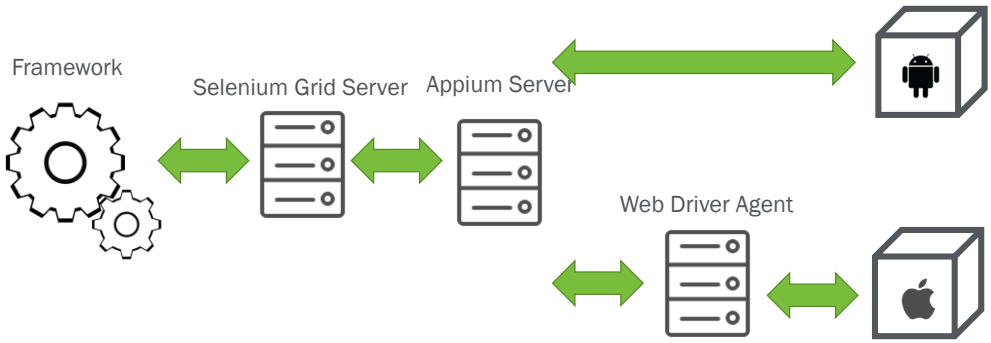
29

Page Factory – Performance - Local



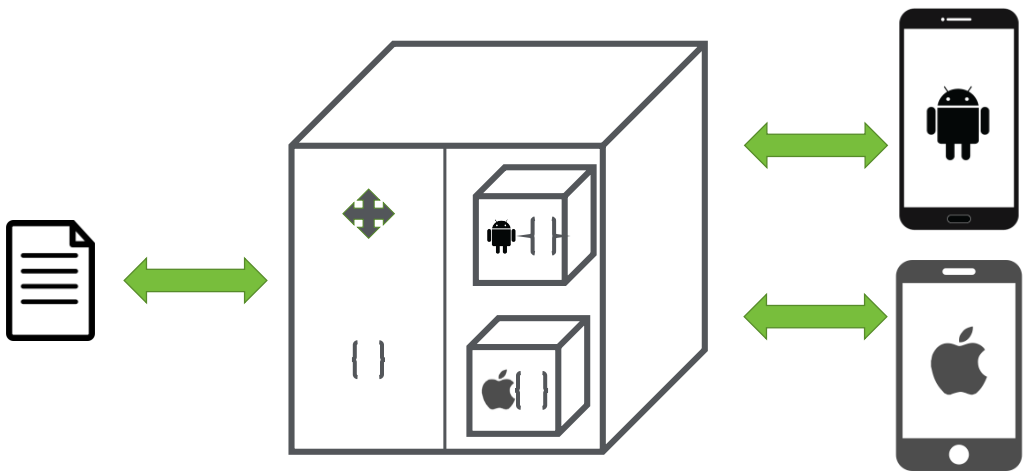
30

Page Factory – Performance - Cloud



31

Page Factory – Shared Code



32

Page Factory - Benefits

- Test reporting stays a run configuration detail
- Page Object Models continue to have more shared code
- Page Object Models rely on web elements more than the driver
  - Elements start being agnostic
- Test Coverage stays consistent across platforms

Page Factory - Consequences

- Extra work needed for Platform Specific tests
- Conditional statement at every instance of UX Divergence
  - But mostly in functional workflow
- Dynamic UI Changes will cause high maintenance or inflated execution time
  - Stale Elements
  - Thread Sleeps
- Execution time becomes bloated because of how Page Factory operates
- Page Factory uses complex method calls to provide elements
- Device divergence becomes difficult.

Page Factory

- Single Test Class - Yes
- Cross Platform - Yes
- App Divergence – Meh
- Performance - No
- Shared Code – Meh

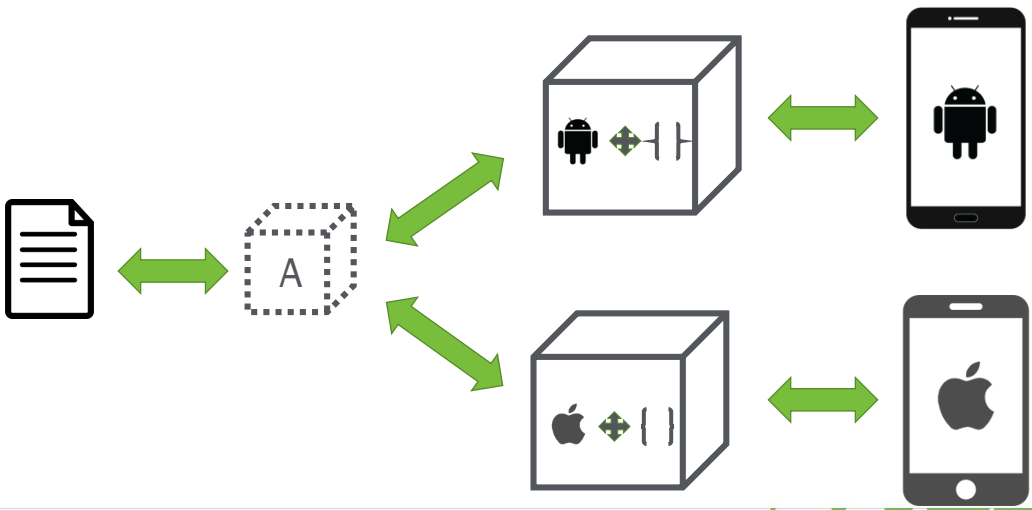
Abstract Parent

"Pay no attention to the man behind the curtain"

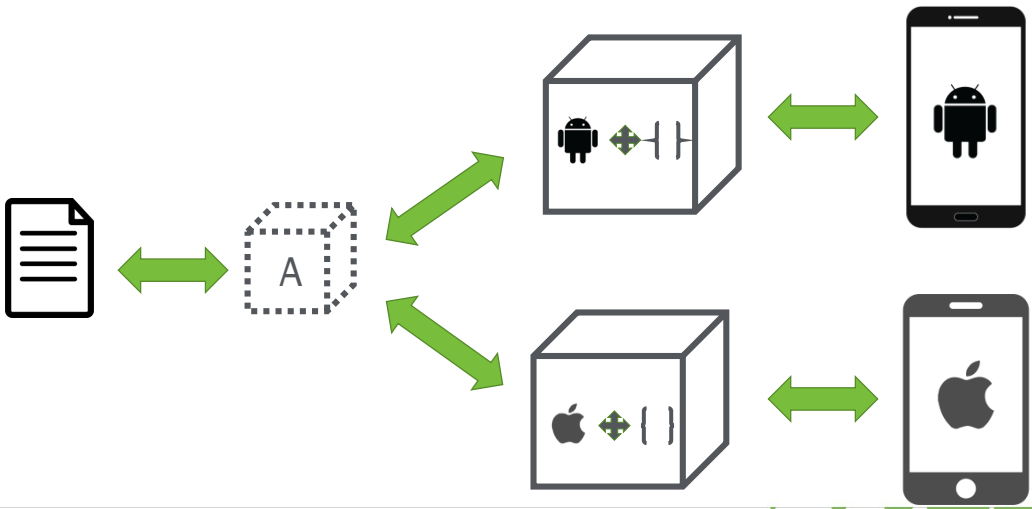
Abstract Parent - Single Test Case

iOS Test Class
Android Test Class

Abstract Parent - Cross Platform

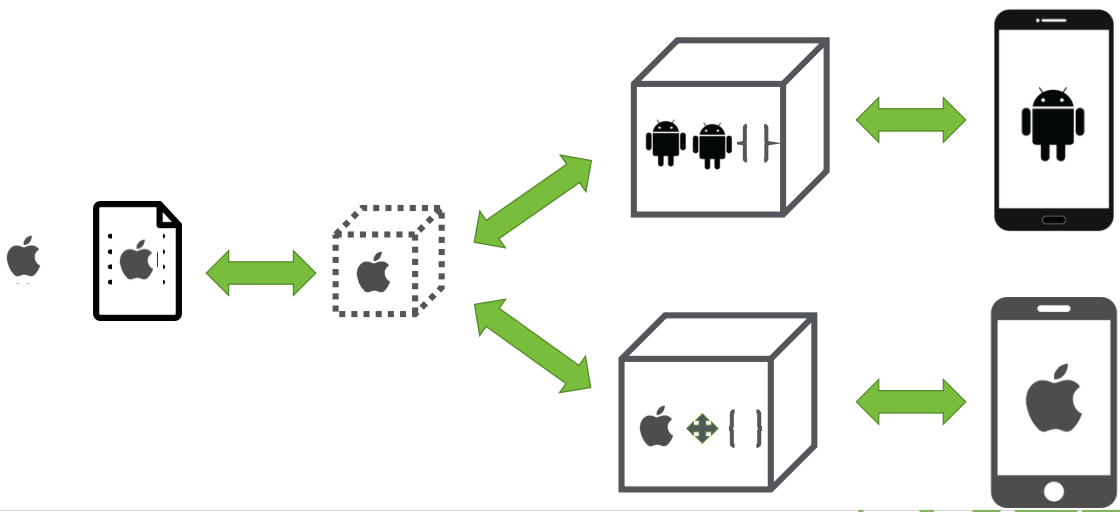


Abstract Parent - Application Divergence



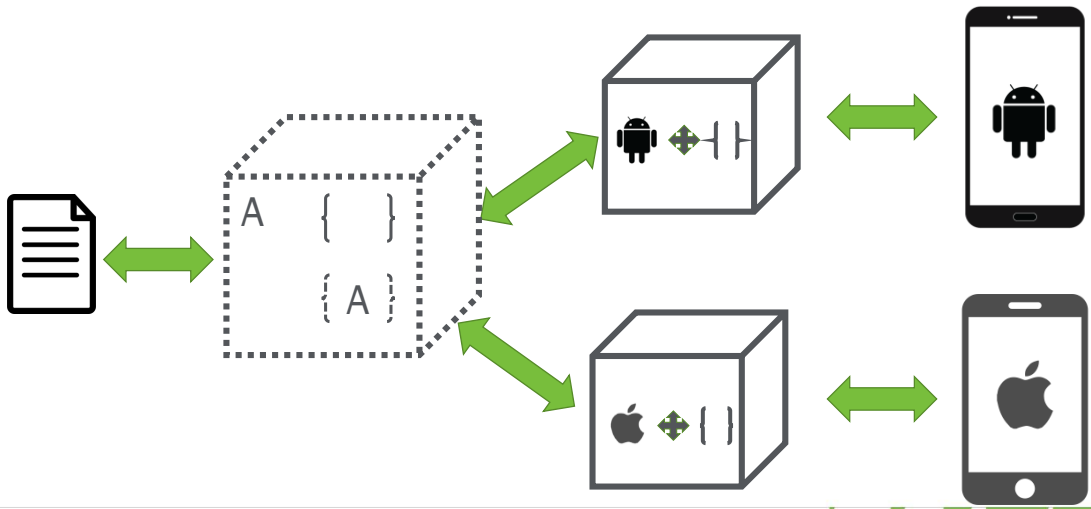
39

Abstract Parent - Performance



40

Abstract Parent - Shared Code



41

Abstract Parent- Benefits

- Test reporting stays a run configuration detail
- Page Object Model flow becomes an app blueprint
  - Parent abstract class holds shared code
  - Platform specific code remains in the child classes
- Test Coverage stays consistent across platforms
- Single Responsibility is respected
- Abstraction allows for App and Device divergence
  - Readability is improved by taking some tricks from BDD tools
- Maintenance is reduced because errors will lead you to whether issue is platform specific or common workflow
- Abstract methods help enforce consistency

42

Abstract Parent- Consequences

- Extra work needed for Platform Specific tests
- Larger framework asset count because of using at least 3 page models per view
- Depending on language used, sharing code in the parent can create a small anti-pattern.
  - Specifically, scripting vs compiled languages

Abstract Parent

- Single Test Class - Yes
- Cross Platform - Yes
- App Divergence - Yes
- Performance - Yes
- Shared Code - Yes

Q & A Time!

45

THANK YOU

Magenic // Fast Forward

46