Quality assurance for mobile applications

Case studies for GUI test automation

Alexandra Schladebeck
Agenda

- The history
- The new questions
- The candidates
- Our experiences
- Results and conclusion
The (hi)story
The (hi)story
The aims

- Evaluate feasibility of mobile GUI test automation
  Functional and technical perspective
- Find out how continuous integration and testing can be implemented
- Analyse cross-platform testing in a mobile context
  Write once, run anywhere
- Make informed decisions on mobile testing strategy
Candidate 1: Customer application

- Datawarehouse for competition analysis
  Enterprise desktop application

- Go mobile
  Conference catalog that uses data
  Extra comfort and features (bookmarking, quick navigation)
  Target group = managers at automobile congresses

- iOS only
  Version 5 → 6 → 7

- Release date *must* be kept
Candidate 2: Continuous Integration

- We test our test tool with our test tool...

- Testing support for widgets requires example applications
  Android, iOS, Windows 8 (modernUI)
  Perform actions and checks

- Functionally uninteresting

- Candidates for evaluating continuous integration
Candidate 3: In-house application

- In-house application
  - Completely new development
- For taking minutes in meetings
- Cross-platform development (Xamarin)
- Cross-platform automated testing
Experiences: Customer (iOS) project

Process
- Manual test scripts specified as automated tests
- Automated test started manually
- Testable version of the app provided manually
- New productive data available very late

Functional aspects
- Required features could be tested with automated test
- Regression testing for new versions (iOS 6)

Technical aspects
- Some modifications necessary due to custom components
Demo
Experiences: Continuous Integration

Linux

... Hudson Build Deploy setup

Mac OSX

Win8

Win Vista

Win7

Eclipse Testing Day 2013
Experiences: Continuous Integration

Prepare Environment
- "reset" backend
- setup test db
- build AUT

Prepare (virtual) test instance
- start
- cleanup
- deploy install
- configure

Run test(s)
- start AUT-Agent
- testexec

Finalize
- gather AUT logs
- tear down AUT & env & tests
Experiences: Continuous Integration

- **iOS**
  - Mac machine
  - XCode
  - Simulator / Device
  - Communication
  - Deploy application (platform specific)
  - Start AUT

- **Windows 8 (modernUI)**
  - Windows 8 machine
  - Visual studio 2012 / 2013
  - Rights & signing
  - AUT automatically startable

Eclipse Testing Day 2013
Experiences: Continuous Integration

- Sandbox
- Platform-dependent scripts necessary
- Idiosyncrasies
- Simulators alone not enough
- Functional testing of
  - Orientation changes
  - GPS
  - Battery
  - Internet connection
  - Access to external applications
Test results

Execution Stack

- caa_ios_FULLTEST_ipad_portrait_5.1 - TC (ok) - 0:13:16.360
- caa_ios_FULLTEST_ipad - TC (ok) - 0:13:35.62
- FULLTEST - TC (ok) - 0:13:16.360

**SPECIFIC**
- TC (ok) - 0:00:00:00
- iOS.Button - TC (ok) - 0:01:43.908
- iOS.AddContact - TC (ok) - 0:00:18.358
  - iOS.AddContact - Check Enablement - TC (ok) - 0:00:02.398
  - iOS.AddContact - Check Existence - TC (ok) - 0:00:02.437
- iOS.AddContact - Check Selection - TC (ok) - 0:00:02.703
- iOS.AddContact - Click - TC (ok) - 0:00:05.966
- iOS.AddContact - Store Value - TC (ok) - 0:00:02.270
- iOS.AddContact - Wait for Component - TC (ok) - 0:00:02.932
- iOS.DetailDisclosure - TC (ok) - 0:00:17.951
- iOS.InfoDark - TC (ok) - 0:00:17.943
- iOS.InfoLight - TC (ok) - 0:00:17.657
- iOS.RoundedRect - TC (ok) - 0:00:20.966
- iOS.RoundedRectBackground - TC (ok) - 0:00:05.522
- iOS.RoundedRectForeground - TC (ok) - 0:00:05.509
- iOS.Switch - TC (ok) - 0:00:20.234
- iOS.UISlider - TC (ok) - 0:00:10.641
- iOS.UISegmentedControl - TC (ok) - 0:00:45.425
  - iOS.UISegmentedControl - Check Enablement - TC (ok) - 0:00:01.922
  - iOS.UISegmentedControl - Check Existence - TC (ok) - 0:00:02.428
  - iOS.UISegmentedControl - Click - TC (ok) - 0:00:05.520
  - iOS.UISegmentedControl - Wait for Component - TC (ok) - 0:00:02.903
  - iOS.UISegmentedControl - Check Existence of Tab - TC (ok) - 0:00:02.784
  - iOS.UISegmentedControl - Check Enablement of Tab by Index - TC (ok) - 0:00:02.239
  - iOS.UISegmentedControl - Check Enablement of Tab by Value - TC (ok) - 0:00:02.307
  - iOS.UISegmentedControl - Check Existence of Tab - TC (ok) - 0:00:02.736
  - iOS.UISegmentedControl - Check Selection of Tab by Index - TC (ok) - 0:00:04.617
  - iOS.UISegmentedControl - Check Selection of Tab by Value - TC (ok) - 0:00:04.624
  - iOS.UISegmentedControl - Check Text of Tab by Index - TC (ok) - 0:00:02.204
  - iOS.UISegmentedControl - Select Tab by Index - TC (ok) - 0:00:04.513
  - iOS.UISegmentedControl - Select Tab by Value - TC (ok) - 0:00:04.648
- iOS.UITextField - TC (ok) - 0:08:17.703
- iOS.UITextField - Click - TC (ok) - 0:00:05.606
- iOS.UITextField - Check Existence - TC (ok) - 0:00:01.912
Experiences: Cross-Platform Project

- Cross-platform development
- Test: write once, run anywhere
Cross-Platform

- SWT
- Swing
- mobile
- .NET
- HTML

- RCP & GEF
- iOS
- Android
- Windows Phone
Cross-Platform

Abstract

Concrete

 SWT
 Swing
 mobile
 .NET
 HTML

 RCP & GEF
 iOS
 Android
 Windows Phone
Cross-Platform

abstract

concrete

Textual components

Graphical components

Buttons

Text fields

Labels

Tables

Lists

…

SWT

Swing

mobile

.NET

HTML

RCP & GEF

iOS

Android

Windows Phone

Gestures
Cross-Platform

Test specification: AUT-independent

Test execution: Correct UI-Driver

Graphical components

Textual components

Buttons
Labels
Text fields
Lists
Tables
…

Textual components

SWT
Swing
mobile
.NET
HTML

RCP & GEF
iOS
Android
Windows Phone

Eclipse Testing Day 2013
23.09.2013
The good news...

- Different mobile UI toolkits use same/similar functional components
  - Buttons, textfields, lists, tabbed components, combo components
Example 1: Edit list

1. topic
   excellent meeting

2.

Exercise

1. Topic
   Excellent meeting

2.
Example 2: Choose minute taker
Example 3: Enter participants

- Titel: pr1
- Ort: 
- Datum: 29.08.2013
- Teilnehmer: Alex, Felix, Michael

Protokoll:
- Alex
- Felix
- Michael
Cross-platform differences are desired

- Differences between desktop versions = minimal
- Differences between mobile clients = huge
  - Features
  - Workflows
  - Use of hardware / software features
  → Users are used to / expect these differences.
- Differences between versions often large
  - iOS 6/7
  - Android 2/4
Experiences: Cross-Platform Project

- Cross-platform development
- Test: write once, run anywhere

- Technically possible
- Unlikely to be used…
Should we automate mobile tests?

Multiplication factor over devices
New OS versions appear regularly
No choice or control over runtime environment
Quick feedback whether new version / feature is total fail
Short release cycles for own software
Changes happen that developers have no control over
Results: Writing automated tests for mobile

- Apps are testable!
- Similar concepts, fewer components
- Extra hurdles
  - More input from development team
  - Additional tests necessary (GPS, Battery, Internet)
  - Environment more complex
Results: Continuous integration for mobile

Feedback → Develop

Analyse → Build

Test → Feedback
Results: Cross platform

- Write once, run anywhere is possible
- But seems unlikely
- How much should / can development be influenced by test factors?
Results: Mobile test strategy
Thank you!

Alexandra Schladebeck
alexandra.schladebeck@bredex.de
@alex_schl
http://testing.bredex.de
blog.bredex.de
www.bredex.de
www.guidancer.de
www.eclipse.org/jubula