

Continuous Testing with Android

by Umar Nizamani

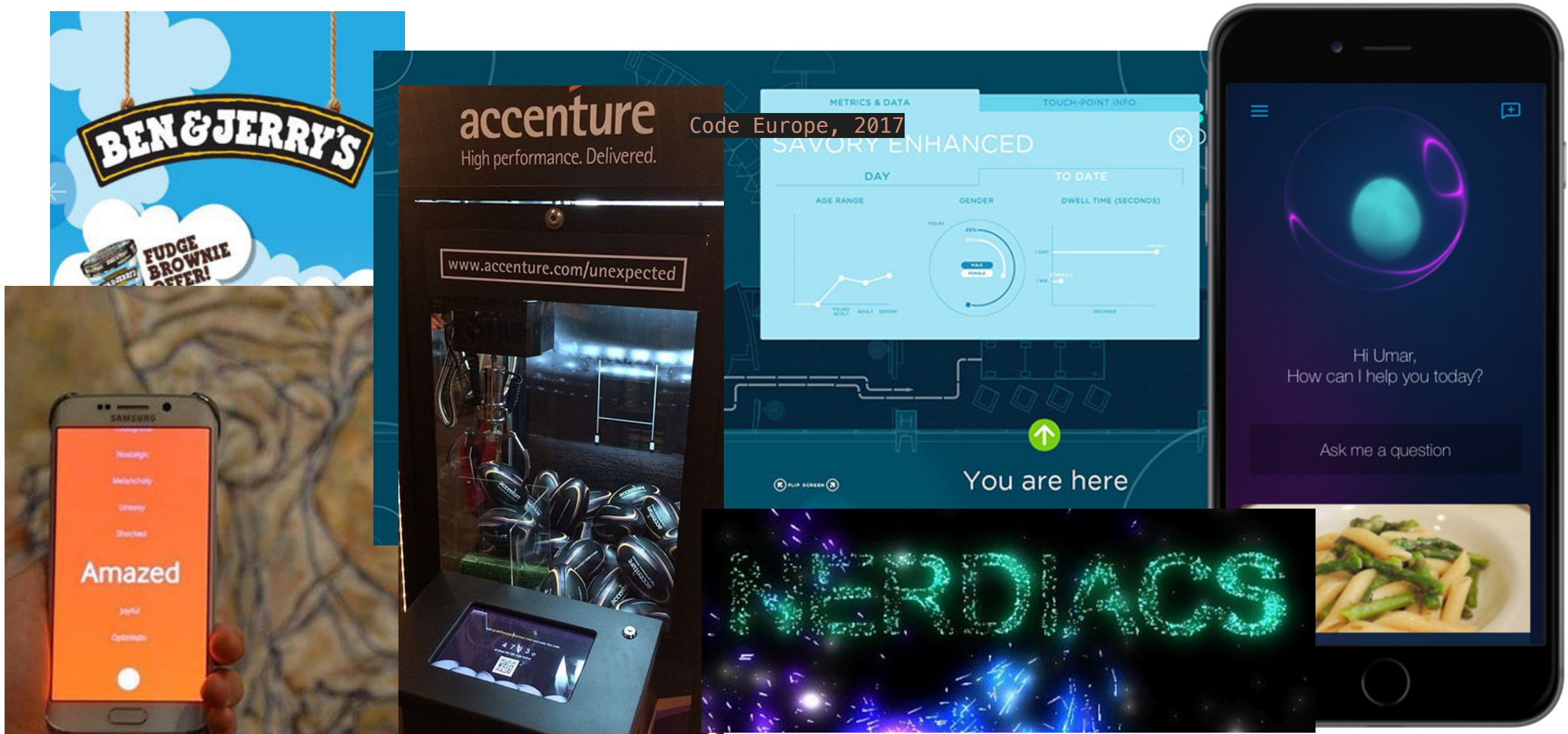
About me

Mobile Games



About me

Experiential Marketing



About me



Tech Lead



**Applications to monitor and coach you to
optimise your quality of life**

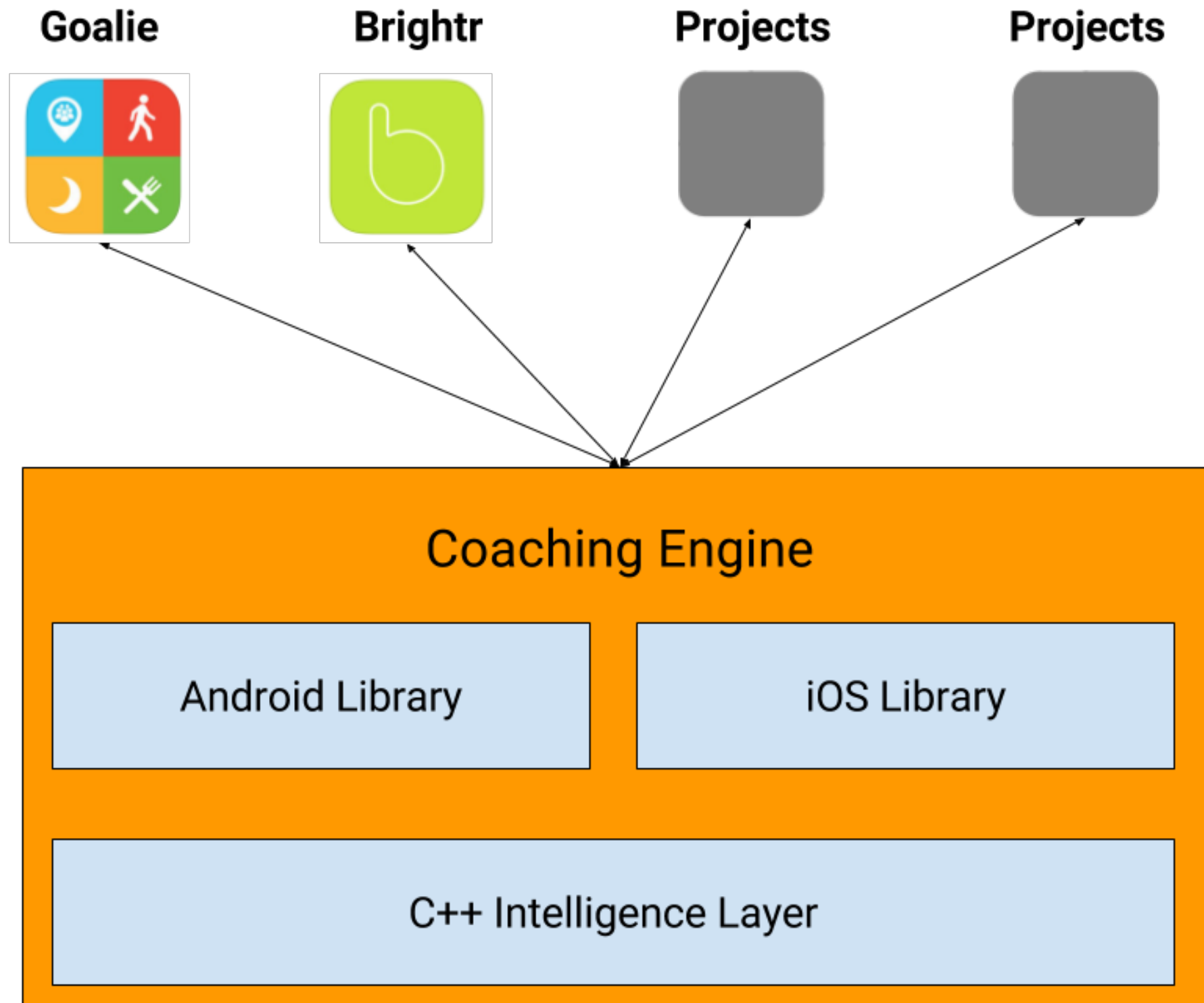


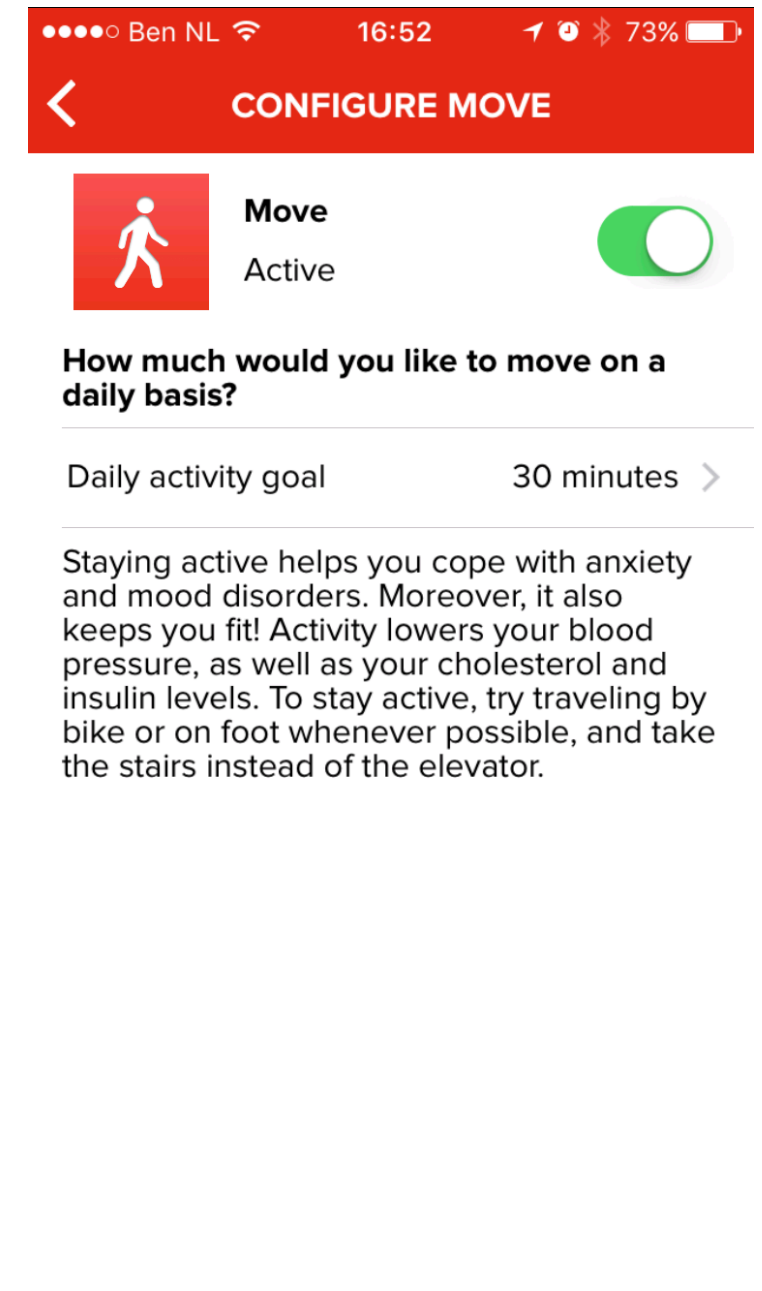
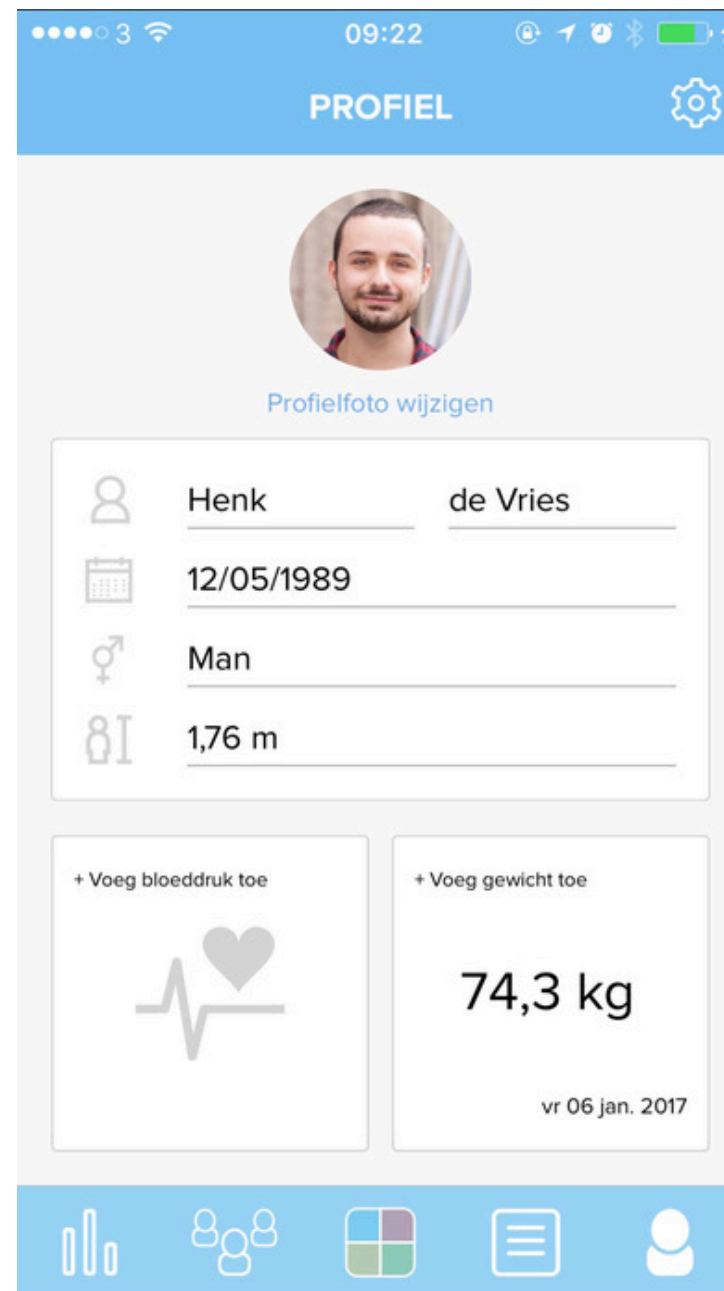
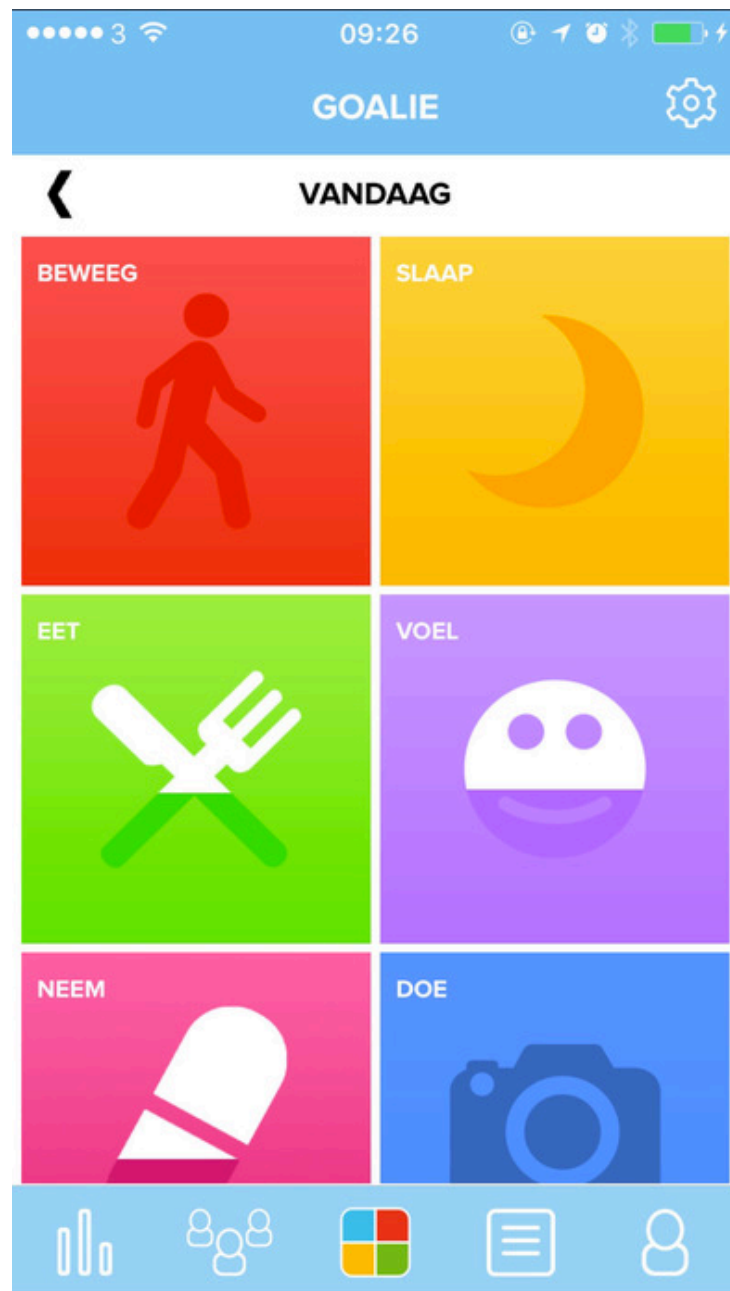
Goalie



brighter



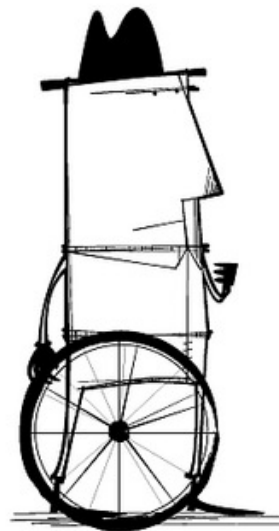




Problems

Technical Debt

ERRR...



**CAN'T STOP.
TOO BUSY!!**



Ownership



How does this work?



Startup



Image courtesy: [pinterest.com](https://www.pinterest.com)

Where do I start?

Quick wins first

If you don't have to understand the code to fix the problem its a quick win



- Android + iOS
- Null pointer exceptions
- Resource/Memory Leaks


```
class Infer {  
  
    String mayReturnNull(int i) {  
        if (i > 0) {  
            return "Hello, Infer!";  
        }  
        return null;  
    }  
  
    int mayCauseNPE() {  
        String s = mayReturnNull(0);  
        return s.length();  
    }  
  
}
```

Found 1 issue

Infer.java:12: error: NULL_DEREFERENCE

object s last assigned on line 11 could be null and is dereferenced at line 12

```
10.     int mayCauseNPE() {  
11.         String s = mayReturnNull(0);  
12. >         return s.length();  
13.     }  
14.
```

Lint

- Built into Android Studio
- End to end Android code problems
- Error ranging in severity from formatting to critical errors

Lint

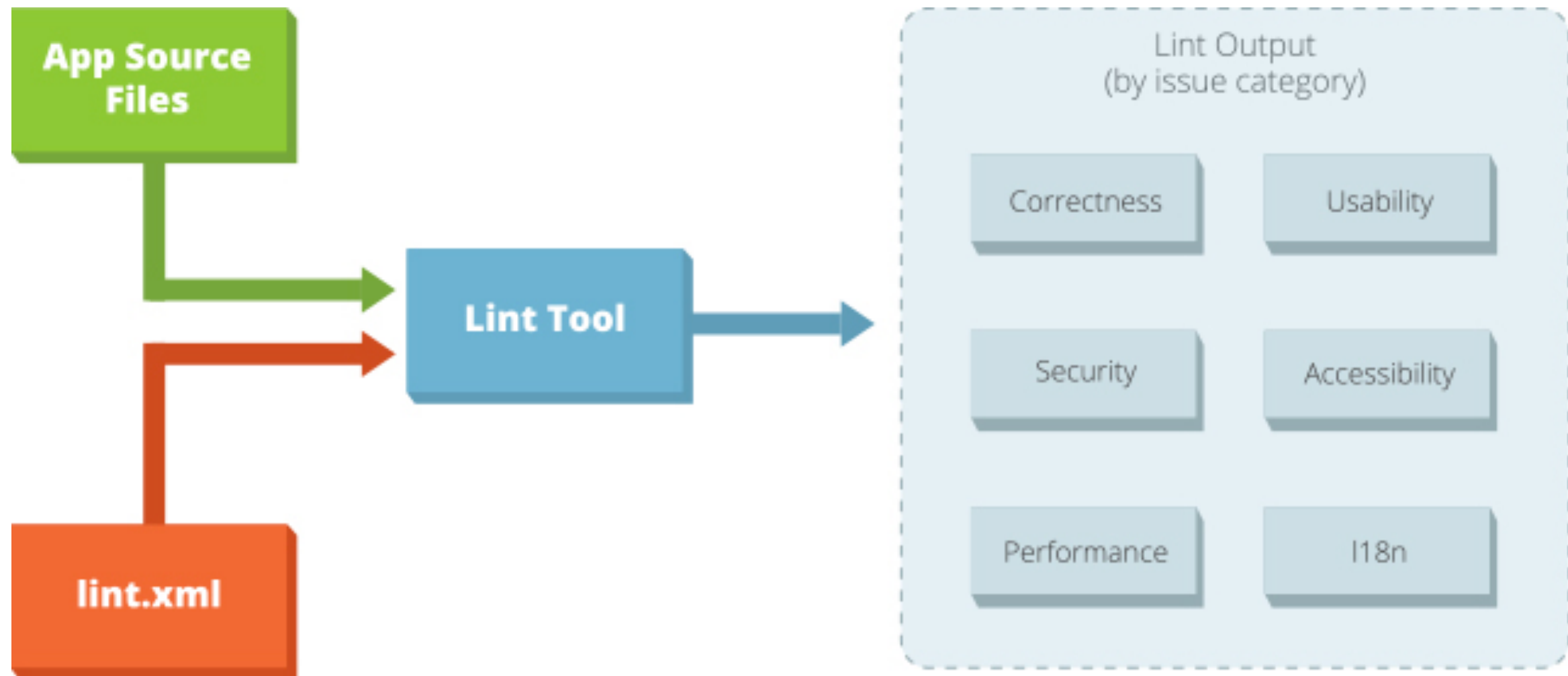


Image courtesy: zeroturnaround.com

Running on our
products



Infer

Android

NULL_DEREFERENCE: 79
CONTEXT_LEAK: 5
RESOURCE_LEAK: 4

Found 88 issues

22 errors in our code

iOS

DIRECT_ATOMIC_PROPERTY_ACCESS: 35
BAD_POINTER_COMPARISON: 4
REGISTERED_OBSERVER_BEING_DEALLOCATED: 3
MEMORY_LEAK: 2
STRONG_DELEGATE_WARNING: 2
NULL_DEREFERENCE: 2
PARAMETER_NOT_NULL_CHECKED: 1

Found 49 issues

Lint (Android Only)

- ~ 5000 warnings
- ~50 potential crashes
- ~ 80 resource warnings avoiding potential layouting problems
- ~ 20 translation problems

Time spent: 3 days

Potentially fatal bugs fixed: > 100

Lay-outing issues fixed: > 150

Across 4 projects (2 products x 2 platforms)

**Reducing technical
debt in 2 weeks**

Delete this repository

Once you delete a repository, there is no going back. Please be certain.

Delete this repository

Well,
thats one way to do it

Maximum gains with minimum effort

- Stable tests that the dev team relies on
- Remove points of failure
- Test like the end user will test
- Test core user interactions first

No mocking frameworks

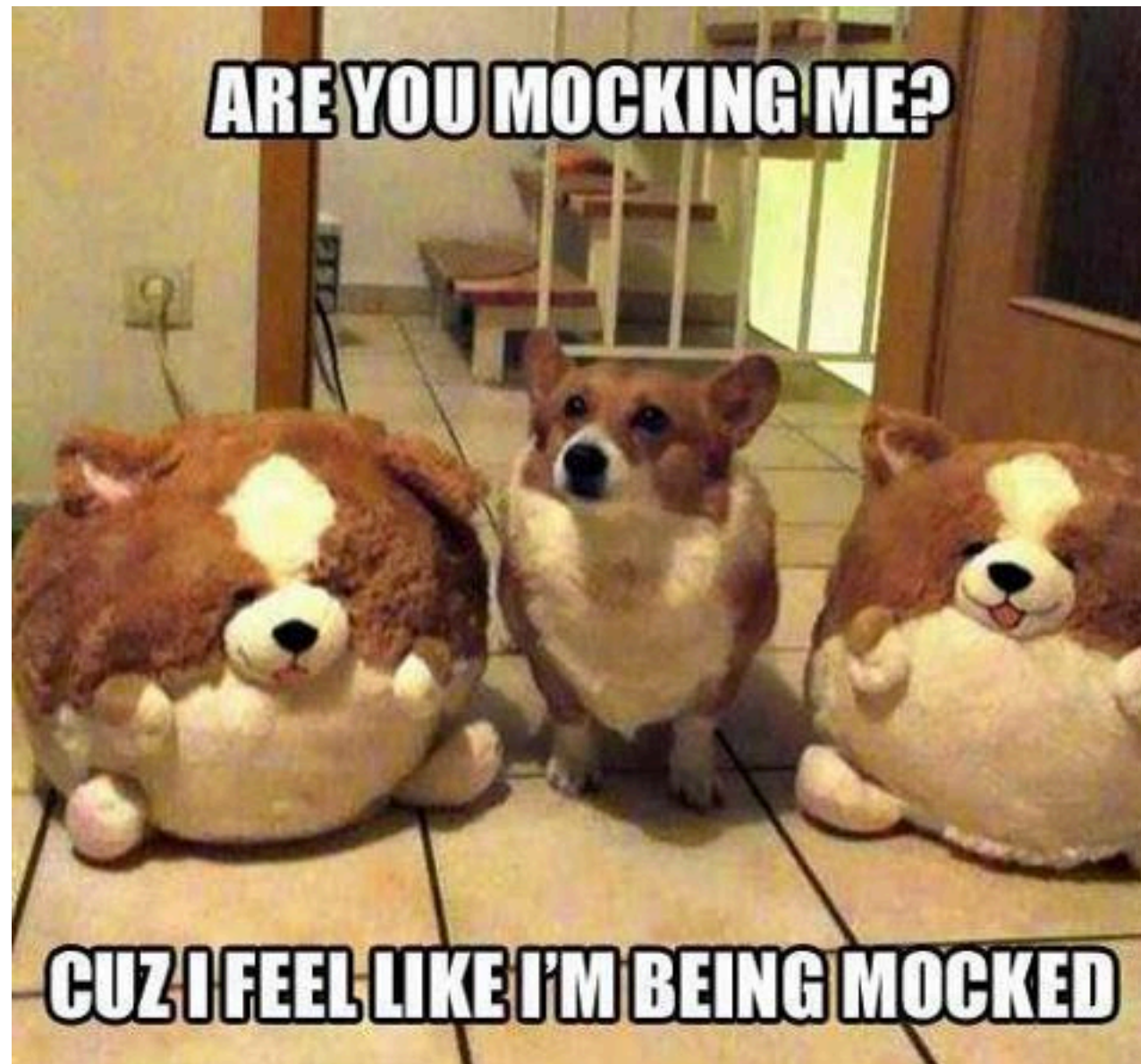


Image courtesy: imgur.com

Activity Test Rule

```
@Rule
public ActivityTestRule<BaseActivity> activityRule =
    new ActivityTestRule<>(
        BaseActivity.class,
        true,
        false);
```

Activity Test Rule

```
@Rule
public SenseLoggedOutActivityTestRule <BaseActivity> activityRule =
    new SenseLoggedOutActivityTestRule <>(
        BaseActivity.class,
        true,
        false);
```

Tests without Espresso

```
// Click Submit  
ClickViewWithId(R.id.sign_in_button);  
  
// Wait for API response  
Thread.sleep(15000);  
  
// Validate...
```

Tests without Espresso

```
// Wait for API response  
Thread.sleep(15000) ; (
```


Espresso

```
@Test
public void greeterSaysHello() {

    onView(withId(R.id.name_field))
        .perform(typeText("Steve"));

    onView(withId(R.id.greet_button))
        .perform(click());

    onView(withText("Hello Steve!"))
        .check(matches(isDisplayed()));

}
```

Espresso

```
// Fill email with null email
onView(withId(R.id.email))
    .perform(clearText(), closeSoftKeyboard());

// Fill password with valid password
onView(withId(R.id.password))
    .perform(clearText(), typeText("1Password"));

// Click Submit
onView(withId(R.id.sign_in_button))
    .perform(click());

// Check if the error is correct
onView(withId(R.id.email)).check(matches(TestHelpers.
    hasErrorText(R.string.error_field_required)));
```

Espresso

- No Thread.sleep
- Espresso can “wait” for async task
- View assertion for everything
- Custom matchers for custom views

Awesome!

- ✓ Native Activity Tests (**ActivityTestRule**)
- ✓ Expressive view assertion (**Espresso**)

But...

More technical debt

- App initialises everything from a single activity
- No support for deep links
- This means can't jump into activity to test
- Lots of click to get to the screen we want to test

Robot Pattern


```
// Get to main screen with a new user account  
gotoHomeScreenWithValidNewUser();
```

```
assertMainScreenIsShowing();
```

```
// Activate movement goal  
ConfigureGoalRobot.activateMoveGoal(activityRule, false);
```

gotoHomeScreenWithValidNewUser();

LoginRobot.gotoRegisterScreen(activityRule);

LoginRobot.registerUniqueNewUser(activityRule);

OnboardingRobot.skipOnboardingScreens(activityRule);

LoginRobot.gotoRegisterScreen()

```
// Force logout as we need to register with a specific user
if (!TestHelpers.onViewExists(R.id.toRegistration)) {
    LoginRobot.logout(activityRule);
}

if (TestHelpers.onViewExists(R.id.toRegistration)) {
    onView(withId(R.id.toRegistration)).
        perform(click());
}
```

Robot Pattern

- Clean and intuitive style
- Once created reusable for all tests
- Allows QA team to write extensive test cases
- Steps to reproduce bugs 1-1 with the test code

Sounds great!

- ✓ Native Activity Tests (**ActivityTestRule**)
- ✓ Expressive view assertion (**Espresso**)
- ✓ Natural language testing (**Robot Pattern**)

But...

Testing Problems

- Still ignoring UI and lay-outing issues
- Need to deploy to real device for testing
- Painful to deploy to multiple devices

Lets Spoon.

Spoon

- Run on multiple devices with a single command
- Capture screenshots for each screen
- Make animated GIFs
- Analytics for tests

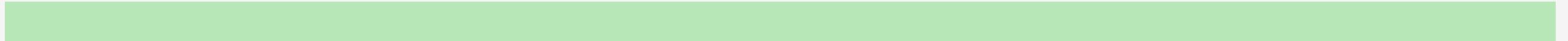
```
./gradlew spoon --debug --stacktrace.
```

workspace debugAndroidTest

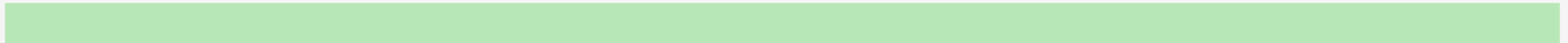
47 tests run across 2 devices with 47 passing and 0 failing in 10 minutes, 36 seconds at 2016-08-23 11:42 AM



Sony C6903



samsung GT-S7390



Sony C6903

24 tests run with 24 passing and 0 failing in 4 minutes, 21 seconds at 2016-08-23 11:42 AM

Running Android 4.4.4 (API 19) with locale en-US



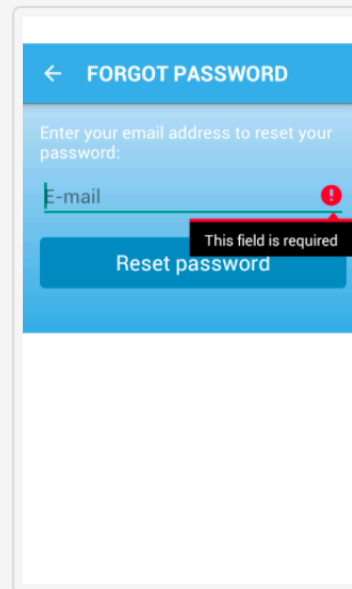
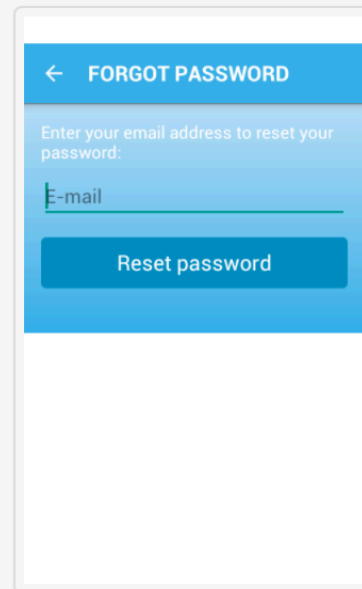
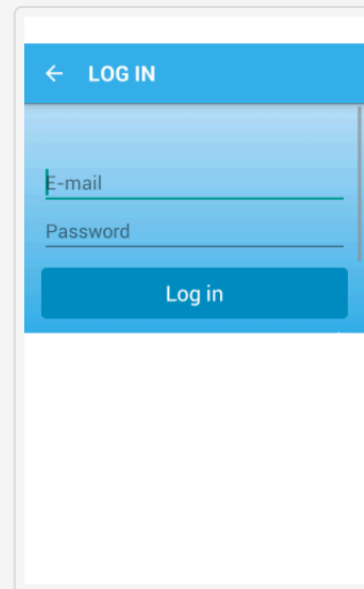
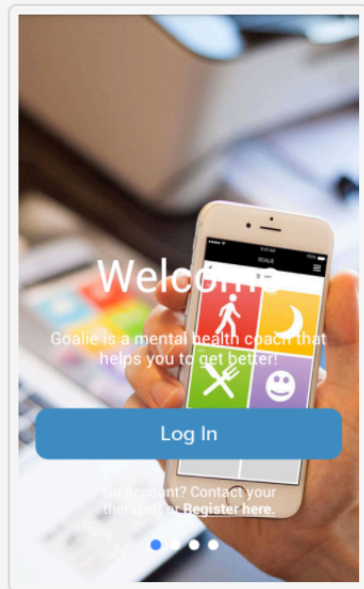
samsung GT-S7390

23 tests run with 23 passing and 0 failing in 3 minutes, 53 seconds at 2016-08-23 11:42 AM

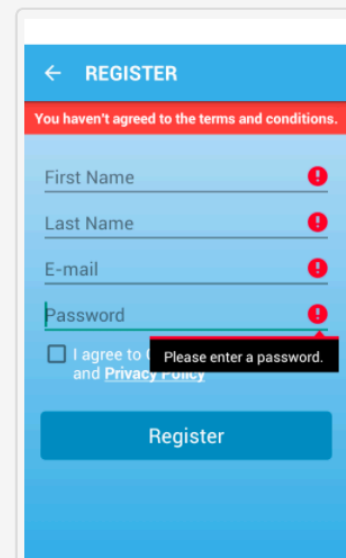
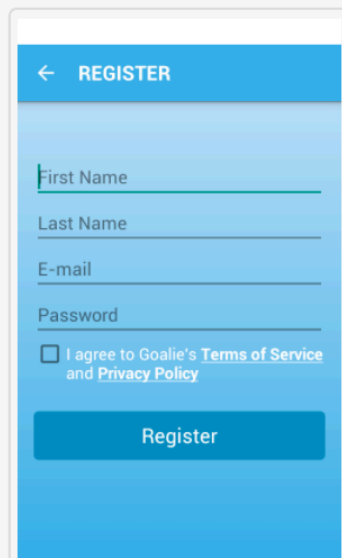
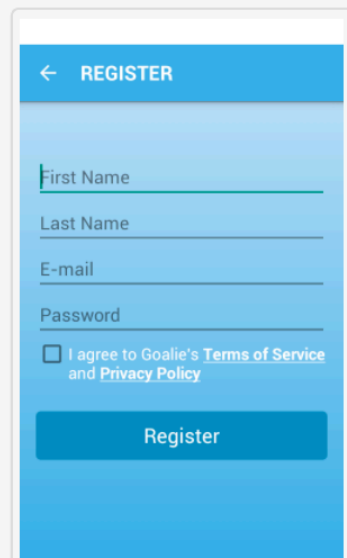
Running Android 4.1.2 (API 16) with locale en-GB



Forgot Password Empty Email



Register Empty All Field



Register Empty All Field

Ran on 2 devices with 2 passing and 0 failing in an average of 4 seconds



average of 4 seconds

Free performance metrics!

Timestamp	Level	Tag	Message
08-23 11:46:17.324	info	TestRunner	started: testRegisterEmptyAllField(GoalieTests.auth.RegisterUser)
08-23 11:46:17.324	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x51555780, 0x51555780) Thread: 31904
08-23 11:46:17.324	debug	HAWAII_EGL	eglMakeCurrent(NULL) Thread: 31904
08-23 11:46:17.334	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x5288af20, 0x5288af20) Thread: 31904
08-23 11:46:17.364	info	MonitoringInstrumentation	Activities that are still in CREATED to STOPPED: 1
08-23 11:46:17.384	debug	ActivityTestRule	Launching activity nl.sense_os.goalie.auth.RegisterActivity
08-23 11:46:17.404	debug	MonitoringInstrumentation	execStartActivity(context, ibinder, ibinder, activity, intent, int, bundle
08-23 11:46:17.454	warn	InputConnectionWrapper	showStatusIcon on inactive InputConnection
08-23 11:46:17.504	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@420c34f8 in: PRE_ON_CREATE
08-23 11:46:17.524	debug	ProgressBar	setProgress = 0
08-23 11:46:17.524	debug	ProgressBar	setProgress = 0, fromUser = false
08-23 11:46:17.524	debug	ProgressBar	mProgress = 0mIndeterminate = false, mMin = 0, mMax = 100
08-23 11:46:17.544	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@420c34f8 in: CREATED
08-23 11:46:17.544	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@420c34f8 in: STARTED
08-23 11:46:17.544	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@420c34f8 in: RESUMED
08-23 11:46:17.664	debug	HAWAII_EGL	Set SWAP INTERVAL 0
08-23 11:46:17.664	debug	HAWAII_EGL	eglCreateWindowSurface() surface: 0x529dbdd8, VC surface: 46, Thread: 31904
08-23 11:46:17.664	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x529dbdd8, 0x529dbdd8) Thread: 31904
08-23 11:46:17.674	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x5288af20, 0x5288af20) Thread: 31904
08-23 11:46:17.674	debug	HAWAII_EGL	eglMakeCurrent(NULL) Thread: 31904
08-23 11:46:17.674	debug	HAWAII_EGL	eglDestroySurface() surface: 0x5288af20, android window 0x51b29018, Thread: 31904
08-23 11:46:17.684	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x529dbdd8, 0x529dbdd8) Thread: 31904
08-23 11:46:17.744	debug	TestHelpers	setWifiState true
08-23 11:46:17.774	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@41da3910 in: STOPPED
08-23 11:46:17.774	debug	LifecycleMonitor	Lifecycle status change: nl.sense_os.goalie.auth.RegisterActivity@41da3910 in: DESTROYED
08-23 11:46:17.774	debug	HAWAII_EGL	eglMakeCurrent(0x51908188, 0x51555780, 0x51555780) Thread: 31904

Testing Complete!

- ✓ Native Activity Tests (**ActivityTestRule**)
- ✓ Expressive view assertion (**Espresso**)
- ✓ Natural language testing (**Robot Pattern**)
- ✓ UI testing by taking screenshots (**Spoon**)

**Lets make this
continuous.**

Build farm

- Old Mac Pro setup with Jenkins
- Connect android devices
- Apply required settings and forget about it (for a little while)




```
1  [||||| 100.0%]
2  [||||| 97.6 ]
3  [||||| 100.0%]
4  [||||| 97.6 ]
5  [||||| 99.4%]
6  [||||| 97.6 ]
7  [||||| 100.0%]
8  [||||| 97.0 ]
9  [||||| 100.0%]
10 [||||| 98.2 ]
11 [||||| 99.4%]
12 [||||| 98.2 ]
13 [||||| 99.4%]
14 [||||| 97.6 ]
15 [||||| 99.4%]
16 [||||| 97.6 ]
Mem[||||| 10772/12288M ]
Swp[||||| ]
```

Jenkins

- Nightly build and run on all devices
- Upload HTML reports for POs to view
- First test = take a screenshot. Runs nightly.
- POs and designers can see the development
- Ensure things are built correctly from the start

Build Steps

- Run a job on all branches with a new commit last night
- Remove all Sense Health apps from all phones
- Run test with Spoon
- Upload HTML report for each respective branch
- Post a link on Slack

Final Setup

- ✓ Native Activity Tests (**ActivityTestRule**)
- ✓ Expressive view assertion (**Espresso**)
- ✓ Natural language testing (**Robot Pattern**)
- ✓ UI testing by taking screenshots (**Spoon**)
- ✓ Continuous Testing (**Jenkins**)

Lessons Learnt

- Never rely on `waitForSeconds` in testing
- `FalconSpoon` to take screenshots
- Don't run more than 2 devices

Lessons Learnt

- UI tests should work like end users
- Automated UI positioning tests can be tricky
- Take screenshots and let the POs be the judge
- Make writing tests closer to natural language
- QA has added > 300 tests

**Lets make our apps
safe again**

Thank You!