

NSLOG? BUILD&RUN MUCH?

UND SONST SO?

PIT GARBE, 11.07.2013



UI DEBUGGING

- statt Logging und iterativem Anpassen des UI
- lieber den Helikopterblick anwenden
- teilweise in laufende App eingreifen
- (sehr) teilweise gewünschte Werte in einem Rutsch permanent übernehmen
- Versehen aufdecken, potentielle Performance-Probleme finden

WIE MACHT IHR DAS?

- üblicherweise hat man NSLog, oder fortschrittlicher: CocoaLumberjack (DDLog)
- für andere Logs muss oft neu gebaut werden
- für Logs von UI-Parametern (Frames, Farben, uvm.) ist das umständlich zu lesen (aber auch zu schreiben)
- teils recht nutzlos (Grafiken debuggen?)

NSLOGGER

- <https://github.com/fpillet/NSLogger>
- bietet tolle Performance, Log-Client, der auch Bilder, NSData, LogLevel, Marker, Remote Logging (ohne Kabel, woohoo!)
- Es gibt eine Bridge von DDLog auf NSLogger
- ist aber immer noch textlastig / kompilierbedürftig

Run 3 of 3

Application Runs

Preferences

Quick Search

Application Sets

Default Set

CommandFusion (iOS)

+
-

Filters for "Default Set"

All logs


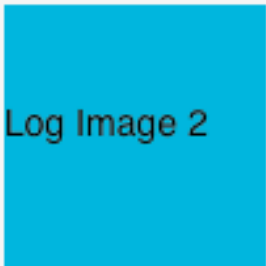
Data blocks

Images

Text messages

+
-

Client connected: NSLoggerClient 1.0 (iPhone OS 4.3.2)
Hardware: iPhone Simulator
UDID: 18BB9BAF-1BD3-53B4-B126-BAA32B7A6CAC

09:33:45.538	Main thread network	test log message 0 - Random characters follow:
09:33:45.738 +200ms	Main thread video	test log message 1 - Random characters follow: ,#6-6105!21736!)40447) *-7 (\$ 5*04!0(5\$8,8+!6:.("+4&!(9:30\$7
09:33:45.938 +199ms	Main thread main 1	test log message 2 - Random characters follow: 8\$'9*!+\$(*1*!#(-:\$ &'*177*4*,020,:& '52(6:78)9/7,&367!\$/8'(/.69)432# #:8&4)/4,6#., 31:4""95"".,\$((2
09:33:45.938 +0ms	Thread 3 transfers	message 3 from standalone thread
09:33:46.141 +203ms	Main thread image	
09:33:46.341 +199ms	Main thread image	
09:33:46.538 +197ms	Main thread main	test log message 4 - Random characters follow: #5:&)#.\$*6'" -#&0,:#1/ (&&# -/--'&(/\$. *8:7#'"*4/046.868#90:36&&\$, 4+9#&*)'(838,+35\$\$+-18:&(-# +8*8-6:317&0
09:33:46.738 +200ms	Main thread video 2	test log message 5 - Random characters follow: 35..8#2+:40!"9249#"# +&940>('73!83/!4:')(5",2(\$#*0,).3'5(0*))
Custom "marker" inserted in log flow		
09:33:46.938 +199ms	Main thread main 1	Raw data, 950 bytes: 0000: 61 70 3c 50 f9 03 c3 e9 85 b3 b4 8d 52 21 eb 3d 'ap<P R! =' 0010: 21 78 54 93 e3 53 38 63 c8 d3 65 a2 65 0f 90 b6 '!xT S8c e e ' 0020: 29 bd 4d 8c 2d f5 de 6f 88 9a b3 da 6b 42 ed db ') M - o kB ' 0030: 64 cd 07 d8 1f a2 34 30 03 42 7d 8e 3c 03 15 68 'd 40 B} < h' 0040: 67 ee 6a 24 d4 75 21 a6 46 f5 9c eh 3h 44 ed 55 'p i\$ u! F :D u'

+
-

|||

f

31 messages

All Levels | All Tags

DCINTROSPECT

- <https://github.com/domesticcatsoftware/DCIntrospect>
- etwas älterer Ansatz
- kann während der Ausführung im Simulator genutzt werden
- wird mit Tastatur und Touch-Eingabe gesteuert
- `pod 'DCIntrospect', '~> 0.0.2'`

DCINTROSPECT

In AppDelegate.m

```
#import <DCIntrospect/DCIntrospect.h>
```

in appDidFinishLaunching...,
nach makeKeyAndVisible:

```
#if TARGET_IPHONE_SIMULATOR  
    [[DCIntrospect sharedIntrospector] start];  
#endif
```

Der Rest wird mit On-Screen-Hilfe erläutert

DCINTROSPECT

- eher für Simulator gedacht, da Keyboardbedienung
 - geht aber auch auf Device per Geste
 - Highlighting, views origin & size anzeigen, samt Abstand zu Window
 - Bewegen und Größe ändern
 - Logging von Eigenschaften von Views (inkl. Subviews, Actions, Targets)
 - Logging von Accessibility-Eigenschaften
 - Manuell setNeedsDisplay, setNeedsLayout and reloadData ausführen
 - Highlight fürs alle Outlines
 - Highlight für alle transparenten Views
 - Warnungen für falsch ausgerichtete Views
 - View-Hierarchie ausgeben
-
- kostet nichts
 - jede Menge Forks auf GitHub, die viel jünger sind (keine Ahnung, ob sich da einer speziell lohnt)

DCINTROSPECT



PONYDEBUGGER

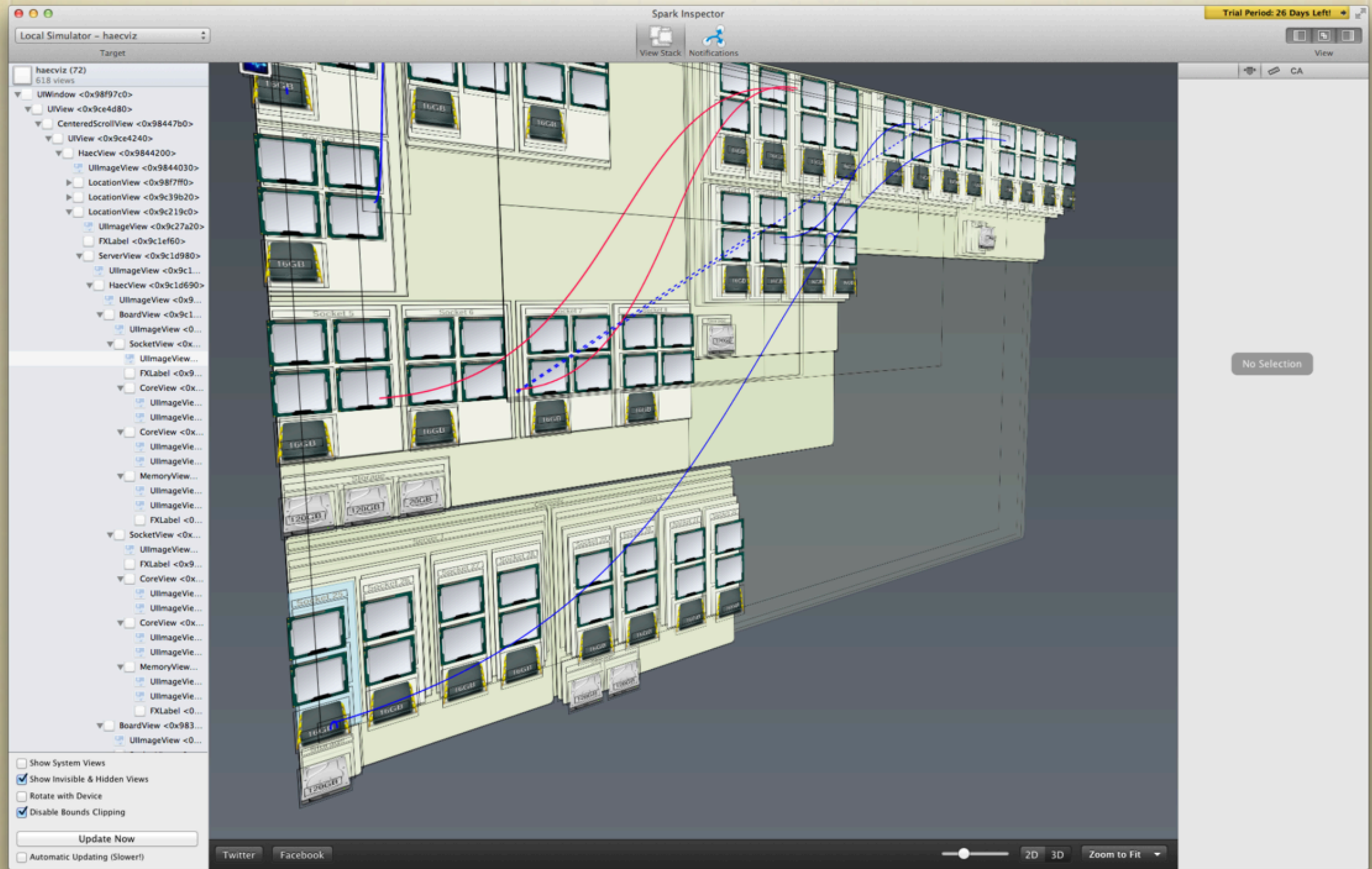
- <https://github.com/square/PonyDebugger>
- `pod 'PonyDebugger'`
- mit Chrome Developer Tools
- View Hierarchie (als XML)
- Netzwerkverkehr analysieren
- Remote Logging and Introspection (eine Art NSLog mit Objekten zum Aufklappen)
- Core Data Browser
- ab iOS 5
- kostenlos

SPARK INSPECTOR

- <http://www.sparkinspector.com/>
- 3D Anzeige der View-Hierarchie
- (alle) Notifications anzeigen, ggf. erneut senden (lässt sich filtern)
- Live-Aktualisierung der Anzeige
- Alle Views und deren Eigenschaften bis auf CALayer Transform herunter live änderbar
- arbeitet mit Swizzling (also lieber nicht in den App Store einliefern)
- OS X 10.8, iOS 5
- 32,19 € bzw. 30 Tage Trial

SPARK INSPECTOR

- Assistant hilft bei Einrichtung im Projekt (wer das nicht mag, nutzt CocoaPods)
- Device oder Simulator
- System Views lassen sich verstecken
- Rotation mit der iPhone-Neigung koppeln
- Clip To Bounds kann deaktiviert werden. D. h. man sieht außerhalb der Device-Grenze und auch, was gerendert wird aber nicht zur Anzeige kommt.



REVEAL

- <http://revealapp.com>
- `pod 'Reveal-iOS-SDK'`
- 2D/3D Anzeige der View-Hierarchie
- nicht live, sondern immer manueller Snapshot
- Live die Views ändern
- Subviews zusammenklappen für mehr Übersicht
- gut für Accessibility, VoiceOver Tests
- Device oder Simulator
- derzeit in Beta, kostenlos testen
- OS X 10.8, iOS 6



SPARK INSPECTOR VS REVEAL

	Spark Inspector	Reveal
ab iOS Version	iOS 5	iOS 6
3D	Ja	Ja
Abstand ändern	Ja	Ja
System Views verstecken	Ja	–
Notifications	Tracken, erneut senden	–
Clipping ausschalten	Ja	–
Live Displayaktualisierung	Ja	–
Nutzt Swizzling	Ja	Nein
Preis	32,19 € (30 Tage Trial)	Beta, kostenlos

XRAY EDITOR (+PROBE)

XRay Editor <http://mireus.com/xrayeditor/>

- Position, Farbe, Schrift, Ausrichtung, etc. in der laufenden App ändern (nur Design, muss dann noch übernommen werden)
- Design-Overlay einblendbar
- \$24.99 Launch Sale
- 14 Tage Trial

Lite Edition:

XRay Probe <http://mireus.com/xrayprobe/>

- Layout der laufenden App zeigen
- \$11.99 Launch Sale
- 14 Tage Trial

XRAY EDITOR (+PROBE)

```
pod 'XRay', '~> 1.2'
```

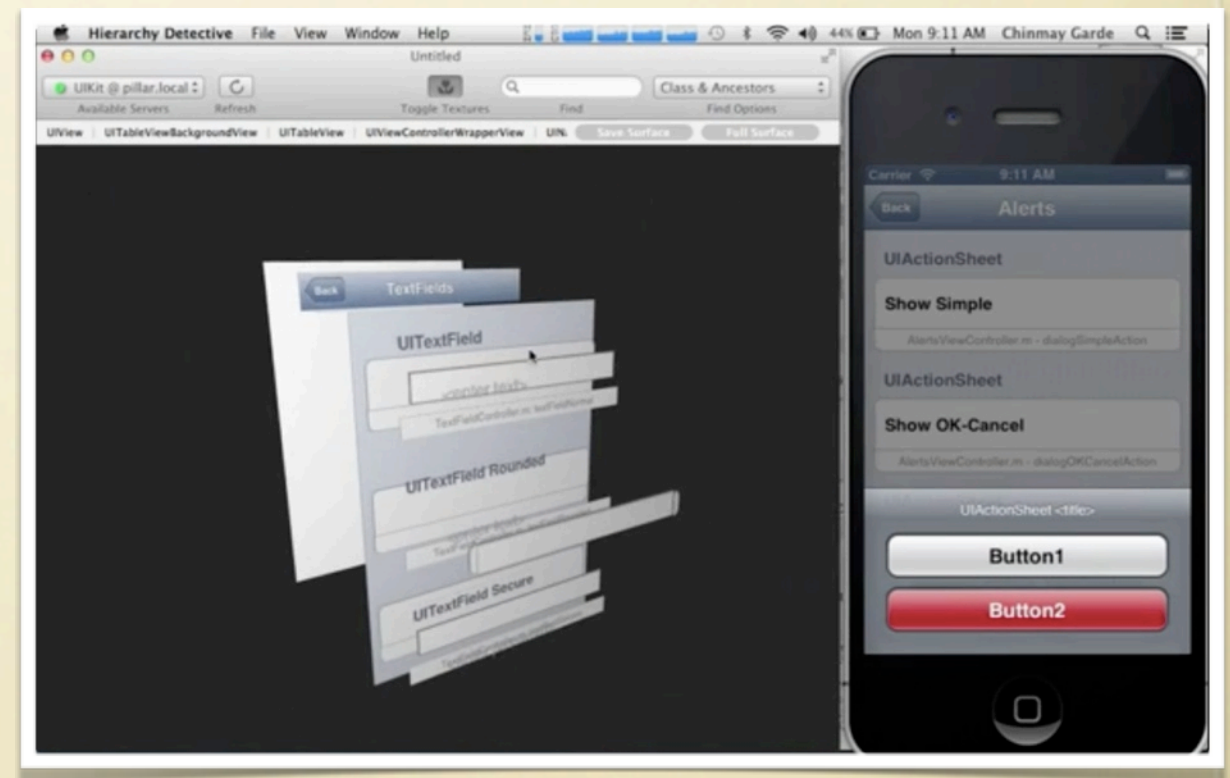
in AppDelegate einfügen:

```
#import <XRay/XRay.h>
```

- hat mit CocoaPods nicht funktioniert. Manuell geht.
- Inhalt scrollt leider ständig aus der Sicht bei UIScrollView und dergleichen.
- Funktioniert auf Device via WLAN
- ab iOS 5
- es gibt ein Xcode-Plugin, mit dem sich die Änderungen relativ einfach in den Code übernehmen lassen sollen

HIERARCHY DETECTIVE

- <http://hierarchydetective.com>
- <https://github.com/chinmaygarde/hierarchydetective>
- ist erweiterbar aufgebaut, unterstützt UIKit, CALayer, cocos2d
- Im Wiki steht, wie man neue View-Hierarchien unterstützen kann
- noch nicht probiert



DEMO