バグは続くよどこまでも

～いかにしてバグと戦うか～

第９ビジネス・ディビジョン　鈴木寿尚

SQUARE ENIX

今回のターゲット

・Objective-C、C++など
　メモリをプログラマが管理する

アプリ開発を主に対象とします。

弊社アプリの多くは上記に該当。

ひさなお

テクニカルディレクター

せいべつ　：おとこ
ねんれい　：あらふぉー
けいけんち：ゲーム業界１９年

ぎょうむ

E　ゲーム開発
E　開発会社技術支援
E　プロジェクト支援

わざ

アミューズメント基盤
PlayStation®
PlayStation®2
GameBoy Advance®
Wii®，ガラケーアプリ
Nintendo 3DS™
スマートフォンアプリ
ブラウザアプリ

携わったタイトル数は
３００くらい

SQUARE ENIX®

アジェンダ

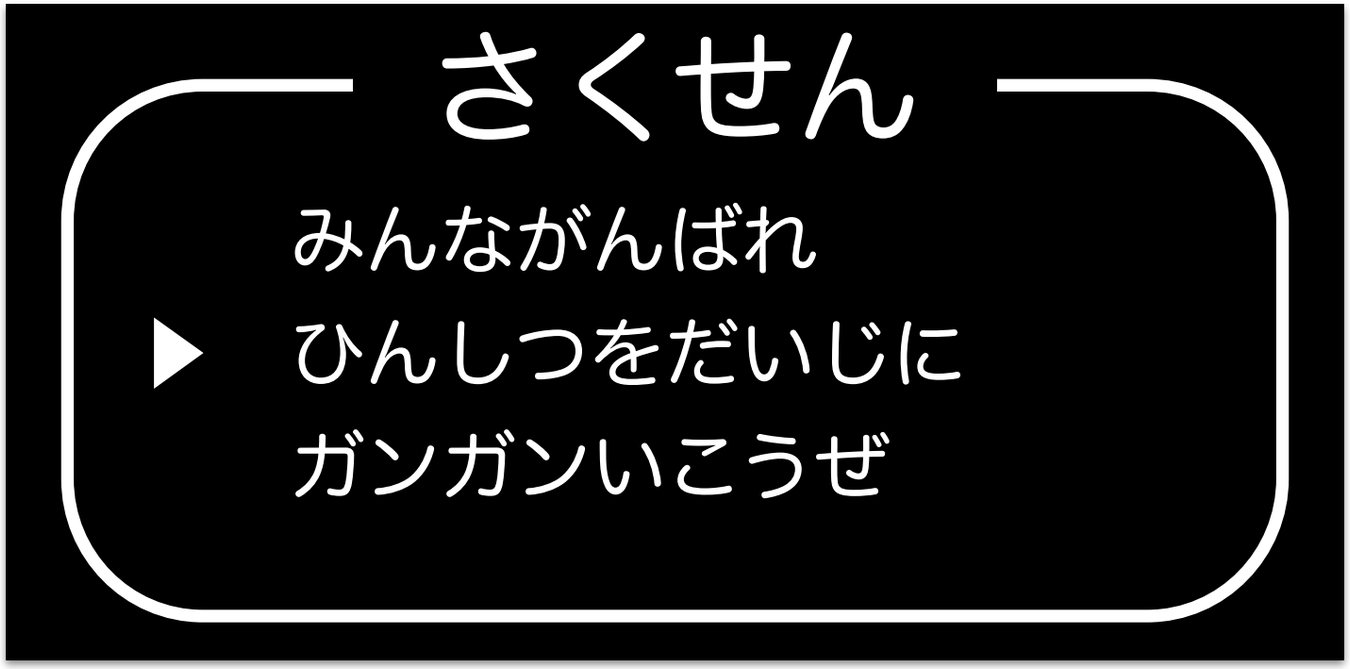
バグを減らすために

バグを見つけるために

バグを追うために

SQUARE ENIX

第一章
～バグを減らすために～



バグを減らす為にできること

バグの多くはコーディングによって紛れ込む。

日頃からバグを発見できる開発体制を整える

ことが重要。

SQUARE ENIX

バグを減らす為に　その１

静的解析の導入

有償

Coverity
Klocwork
C++Test
…

無償

FindBug
cppcheck
Clang(Xcode)
…

バグを減らす為に　その２

ユニットテストの導入

ツール

CppUnit
JUnit
C++Test
Xcode

ロジック関連に効果あり！

描画関連に不向き…

バグを減らす為に　その３

コードレビューの導入

バグ混入を防止

- コード設計の確認
- 脆弱性の発見
- パフォーマンス低下の防止
- エンジニア成熟度の確認

SQUARE ENIX

バグを減らす為に　その４

自動動作テストの導入

シナリオに従ってアプリを自動再生

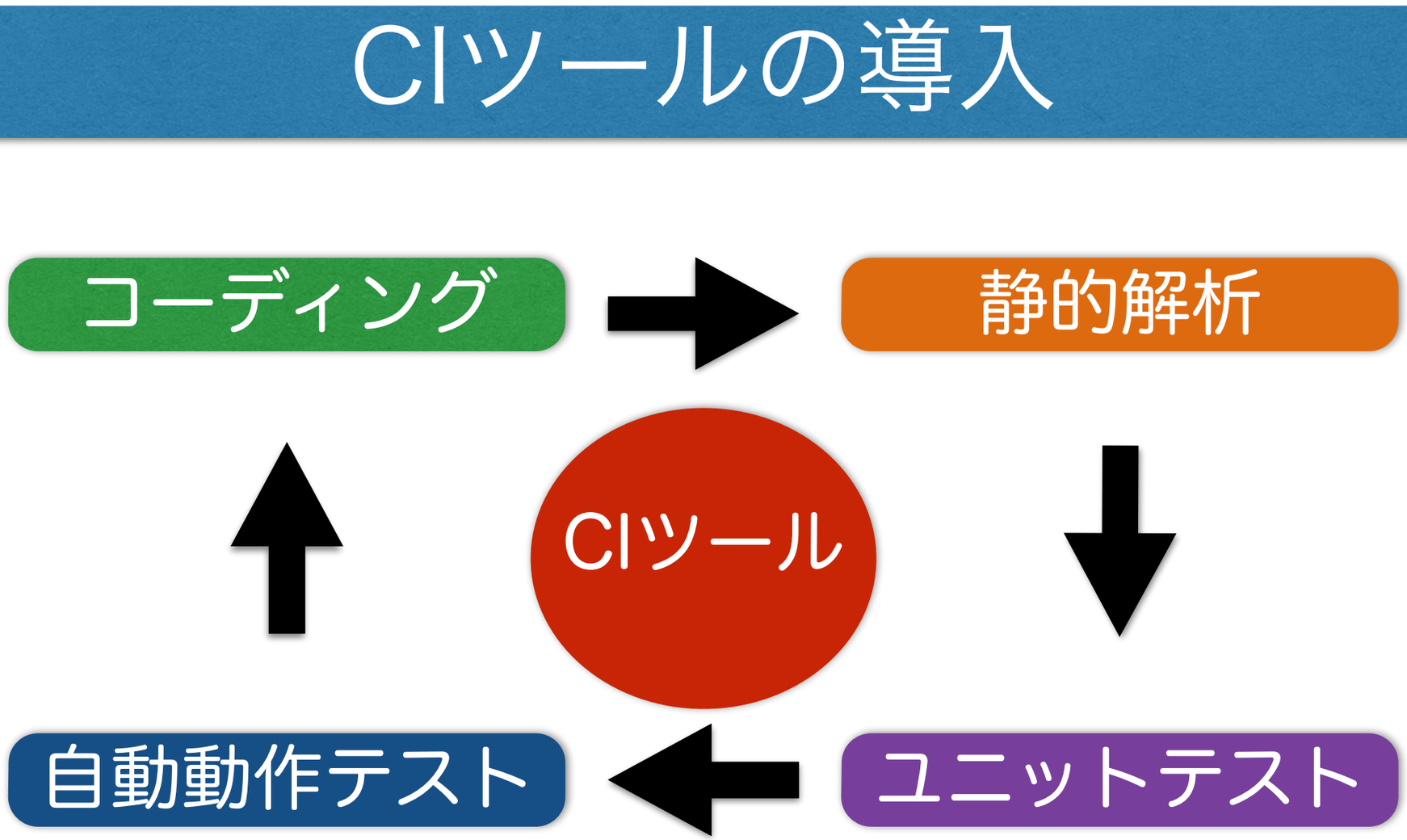
基本動作確認（そもそも動作する？）

- メモリ問題の発見
- パフォーマンスチェック
- リソース整合性チェック
- 全画面遷移チェック

SQUARE ENIX

バグを減らす為に　その５

CIツールの導入



SQUARE ENIX

バグを減らす為に

- 静的解析の導入
- ユニットテストの導入
- コードレビューの導入
- 自動動作テストの導入
- CIの導入

SQUARE ENIX

バグを減らす為に

静的解析の導入

…などなど。

他にもできることは沢山ありますが

バグ混入率が低下すればエンジニアは幸せに！

CIの導入

SQUARE ENIX

第二章
～バグを見つけるために～

エンジニアは　バグからにげだした
しかし　まわりこまれてしまった

バグを見つける為に

Instrumentsなど有益なツールは存在するもの
QAチェック時（特に外部委託の場合）に上記ツールを
利用できるとは限らない。

バグ発見率を上げる為
アプリへ有益な機能を実装しておく必要がある。

想定外の値は例外かアサート

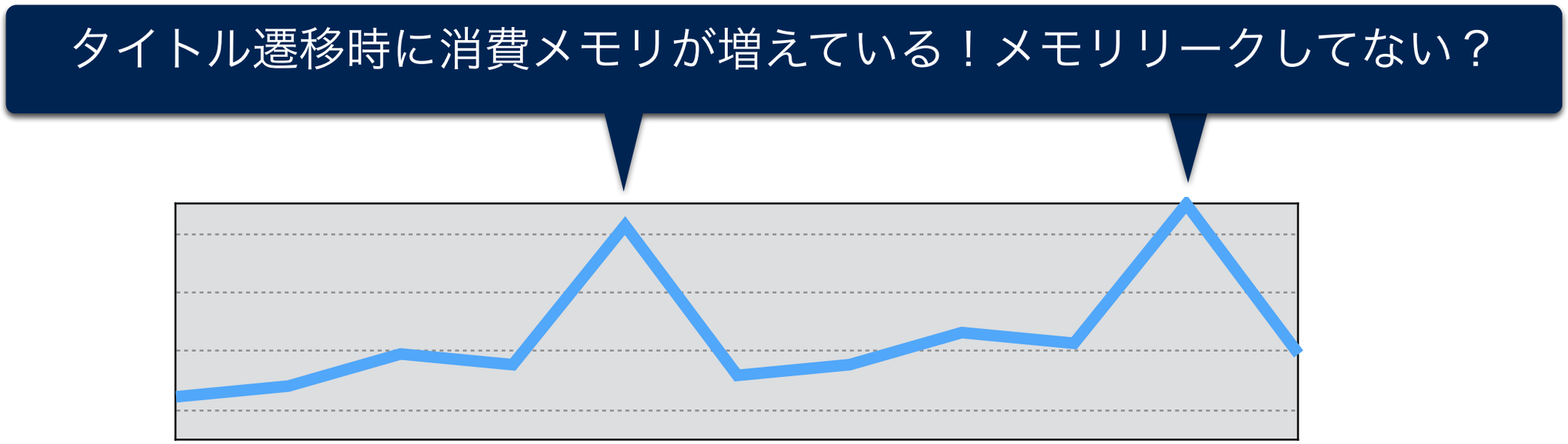
通信でデータを取得するアプリでは
必ずしも適切なデータがやってくるとは限らない

- サーバ更新時のオペレーションミス
- 想定外の値を受け取ってしまった

継続不可能な場合はアサート・例外エラーとして扱う
QA時はあえてCrashさせてレポートを収集することも

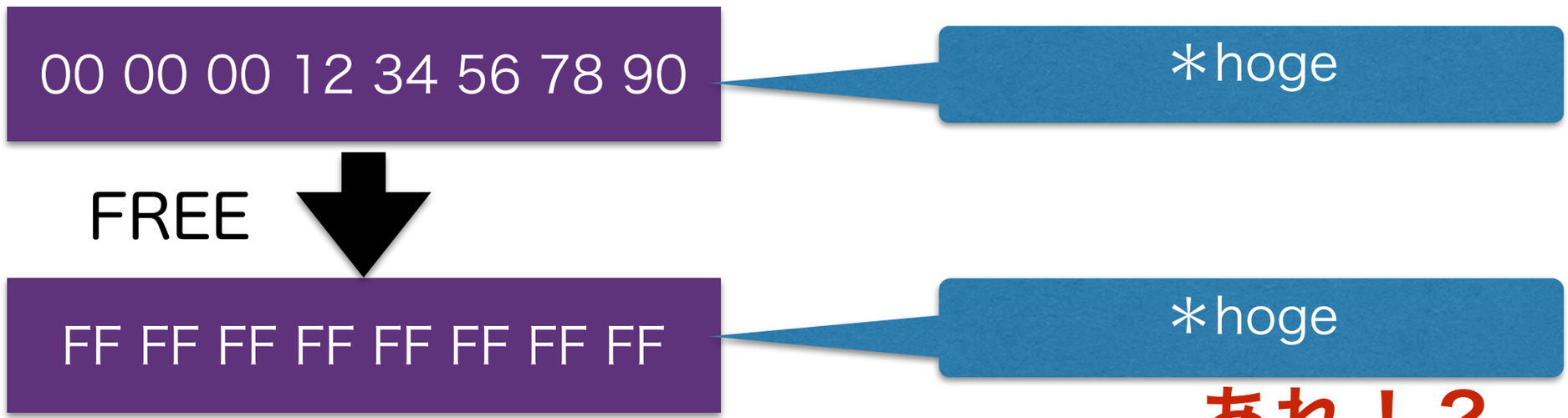
メモリグラフを表示

メモリ推移を画面上にグラフ表示する
- メモリリークの発見
- メモリの大量消費タイミングを把握



ゾンビーオブジェクト

解放したメモリを参照してしまうバグを発見する。
・メモリアロケーターを独自に実装
・QAではメモリ解放時に対象メモリ領域を0xFFでFILL



クラッシュレポートの収集

クラッシュレポートは情報の宝庫。
積極的に利用してバグを特定！

iOS 8からはCrashReportへアプリ独自のログを出力することも可能に

```
Incident Identifier: D0F7FEF7-3522-4C1E-BD41-35A254E3B43B
CrashReporter Key:   10f5b1b53d4983c8965dc4c342750f35069491ea
Hardware Model:      iPhone3,1
Process:             VanguardSM [509]
Path:                /var/mobile/Applications/C234DFD9-0FC2-4D4D-BC88-CCF16BA3BDB9/VanguardSM.app/VanguardSM
Identifier:          com.suare-enix.VanguardSM
Version:             1.0 (1.0)
Code Type:           ARM (Native)
Parent Process:      launchd [1]

Date/Time:           2014-06-19 16:17:16.539 +0900
OS Version:          iOS 7.1 (11D169)
Report Version:      104

Exception Type:  EXC_BAD_ACCESS (SIGSEGV)
Exception Subtype: KERN_INVALID_ADDRESS at 0xffffffff
Triggered by Thread:  0

Thread 0 Crashed:
0   libobjc.A.dylib                 0x3a1f2622 objc_msgSend + 2
1   VanguardSM                      0x000c56b8 -[AppMain(PauseMenu) isPushedPauseIcon:] (PauseMenu.m:191)
2   VanguardSM                      0x000aa224 -[AppMain(SectionTitleInternal) doTitleCommandAction:] (SectionTitle.m:348)
3   VanguardSM                      0x000a9900 -[AppMain(SectionTitle) doTitleMenu:] (SectionTitle.m:164)
4   VanguardSM                      0x000a7efa -[AppMain(SectionMain) doMainSection:] (SectionMain.m:84)
5   VanguardSM                      0x0009e0ec -[AppMain doMain:] (AppMain.m:181)
6   VanguardSM                      0x000dca44 -[MSFView(InternalMethods) runMain:] (MSFView.m:297)
7   Foundation                      0x30425b00 __NSFireTimer + 60
8   CoreFoundation                  0x2fa071b4 __CFRUNLOOP_IS_CALLING_OUT_TO_A_TIMER_CALLBACK_FUNCTION__ + 12
9   CoreFoundation                  0x2fa06fca __CFRunLoopDoTimer + 778
10  CoreFoundation                  0x2fa05166 __CFRunLoopRun + 1206
11  CoreFoundation                  0x2f96ff4a CFRunLoopRunSpecific + 518
12  CoreFoundation                  0x2f96fd2e CFRunLoopRunInMode + 102
13  GraphicsServices                0x3487465e GSEventRunModal + 134
14  UIKit                           0x322bb168 UIApplicationMain + 1132
15  VanguardSM                      0x000997da main (main.m:16)
16  libdyld.dylib                   0x3a6f5ab4 start + 0

Thread 1:
0   libsystem_kernel.dylib          0x3a798808 kevent64 + 24
1   libdispatch.dylib               0x3a6e4de8 _dispatch_mgr_invoke + 228
2   libdispatch.dylib               0x3a6d3f6e _dispatch_mgr_thread$VARIANT$up + 34

Thread 2:
0   libsystem_kernel.dylib          0x3a7abc70 __workq_kernreturn + 8
1   libsystem_pthread.dylib         0x3a812bda _pthread_wqthread + 306
2   libsystem_pthread.dylib         0x3a812a94 start_wqthread + 4

Thread 3:
0   libsystem_kernel.dylib          0x3a798a58 mach_msg_trap + 20
1   libsystem_kernel.dylib          0x3a798854 mach_msg + 44
2   AudioToolbox                    0x2f3eb716 AURemoteIO::IOThread::Run() + 102
3   AudioToolbox                    0x2f3ee8dc AURemoteIO::IOThread::Entry(void*) + 4
4   AudioToolbox                    0x2f32bfc0 CAPThread::Entry(CAPThread*) + 208
5   libsystem_pthread.dylib         0x3a814916 _pthread_body + 138
6   libsystem_pthread.dylib         0x3a8148b6 _pthread_start + 98
7   libsystem_pthread.dylib         0x3a812aa0 thread_start + 4
```

シンボル情報を復元して解析！

- Android™: objdump
- iOS: symbolicatecrash

SQUARE ENIX

第三章
～バグを追うために～

おおエンジニアよ
バグをみのがすとはなさけない

バグを追う為に

どれほど手厚いデバッグを行ったとしても、ユーザーの手に
アプリが届いた後にバグが発見される可能性は否定できない。

どのようなバグが、どれだけのユーザーに影響を与えているかを
リアルタイムで把握できる仕組みを準備しておく必要がある。

・FROSK株式会社が運営するクラッシュ解析ツール。
http://smrtbeat.com/

・cocos2d-x、iOS(Objective-C/C++)、
Android™(java/C++)、Unity(C#/js)に対応

・主な機能として
クラッシュレポートの自動シンボル復元
アプリバージョン別の品質スコアリングなど。

SmartBeatのしくみ

SmartBeatサーバ

アプリ起動時にレポート送信

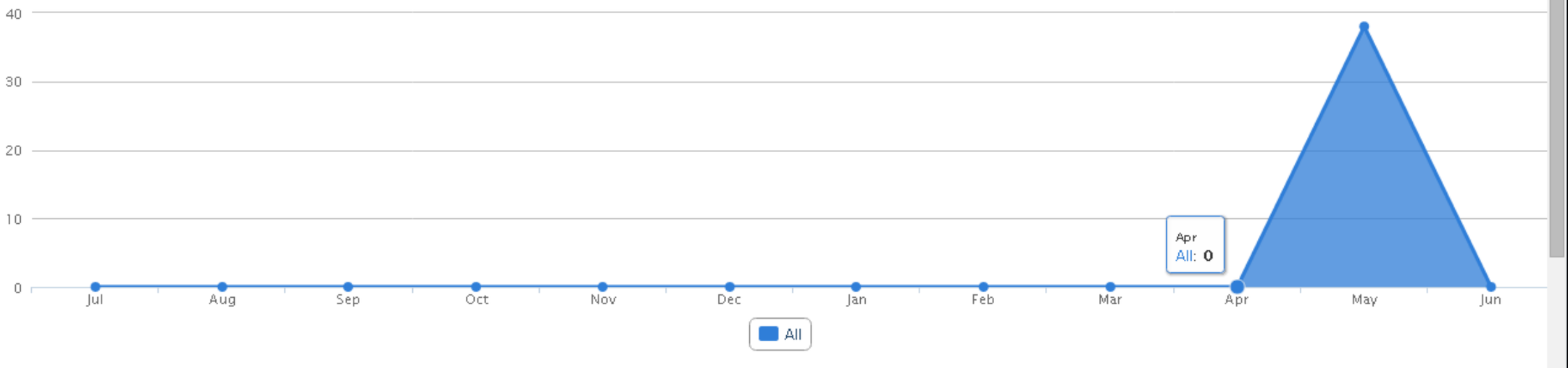
アプリ

- 端末情報
- 例外エラー
- Log
- スタック情報
- スクリーンショット
- アプリバージョン

アプリバージョン、エラー別にグループ化

SmartBeat | エラー管理 | 分析 | 設定 | 日本語 | hisanao ログアウト

UnityForAndroid



エラー種別
- 全て
- クラッシュ
- 例外

期間
1年

アプリバージョン
- 1.0

OSバージョン
- 4.3

すべて選択　エラー検索　並び替え: 発生回数 | 影響ユーザ数 | 初回発生日時 | 最終発生日時

6
Unity: DllNotFoundException: cppplugintest1
[Open]
ID: 3663　影響ユーザ数: 3
発生箇所: ShooterController.NullTest ()
タグ: 追加
最終発生: 2014/05/28 18:10:07

6
SIGSEGV
[Open]
ID: 3657　影響ユーザ数: 3
発生箇所: unknown
タグ: 追加
最終発生: 2014/05/27 19:14:26

SIGSEGV

SQUARE ENIX

クラッシュログへシンボル情報を割り当て

クラス:
unknown

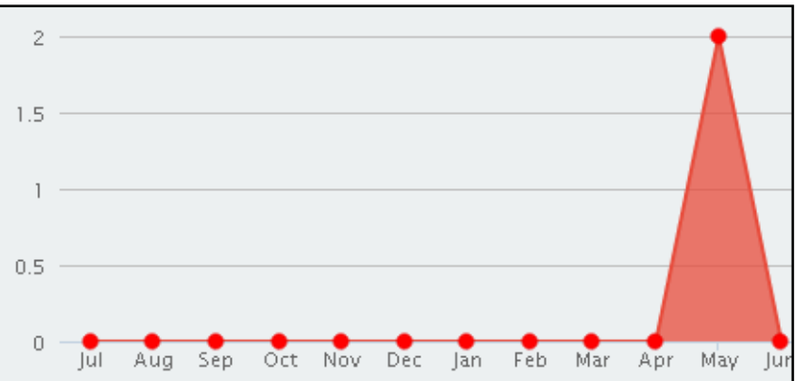
発生箇所:
unknown

ステータス:
Open 変更

タグ: 追加

最終発生日時:
2014/05/21 20:32:27

初回発生日時:
2014/05/21 20:24:00



2014/05/21 | 20:32:27 | 20:24:00

(2件中 2件)

スタックトレース | 端末情報 | LogCat | 画面履歴

```
0  libc.so + 0x1c746
1  dalvik-heap (deleted) + 0x12dc4c6
2  dalvik-heap (deleted) + 0x12dc4c6
3  dalvik-heap (deleted) + 0x12dc51e
4  libc.so + 0x1c12b
5  libc.so + 0x1c23f
6  libFileTest.so!Java_com_example_filetest_MainActivity_loadFromJNI [FileTest.cpp : 86 + 0x9]
7  libdvm.so + 0x1f372
8  dalvik-heap (deleted) + 0xd18c1e
9  data@app@com.example.filetest-2.apk@classes.dex + 0x72e00
10  libdvm.so + 0x4e0bb
11  data@app@com.example.filetest-2.apk@classes.dex + 0x72dfe
12  libFileTest.so!Java_com_example_filetest_MainActivity_saveFromJNI [FileTest.cpp : 69 + 0xf]
13  libc.so + 0x15ab7
14  libc.so + 0x172c3
15  libdvm.so + 0x50059
16  dalvik-mark-stack (deleted) + 0x1b5276c
```

メトリックス

WiFi	Mobile Network
0.0%	0.0%
GPS	Memory
0.0%	20.9MB

発生アプリバージョンリスト

	占有率	エラー発生回数
1.0	100%	2

発生OSバージョンリスト

	占有率	エラー発生回数
4.1.2	100%	2

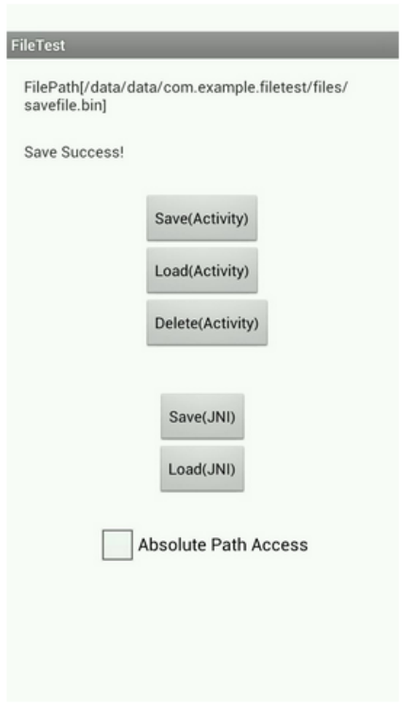
SQUARE ENIX

クラッシュ直前のスクリーンショット

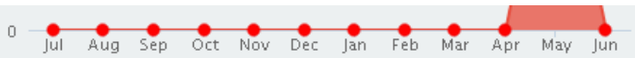
2014/05/21 | 20:32:27 | 20:24:00

(2件中 2件)

スタックトレース | 端末情報 | LogCat | 画面履歴



1 2 3



メトリックス

WiFi **0.0%**

Mobile Network **0.0%**

GPS **0.0%**

Memory **20.9MB**

発生アプリバージョンリスト

	占有率	エラー発生回数
1.0	100%	2

発生OSバージョンリスト

	占有率	エラー発生回数
4.1.2	100%	2

発生デバイスリスト

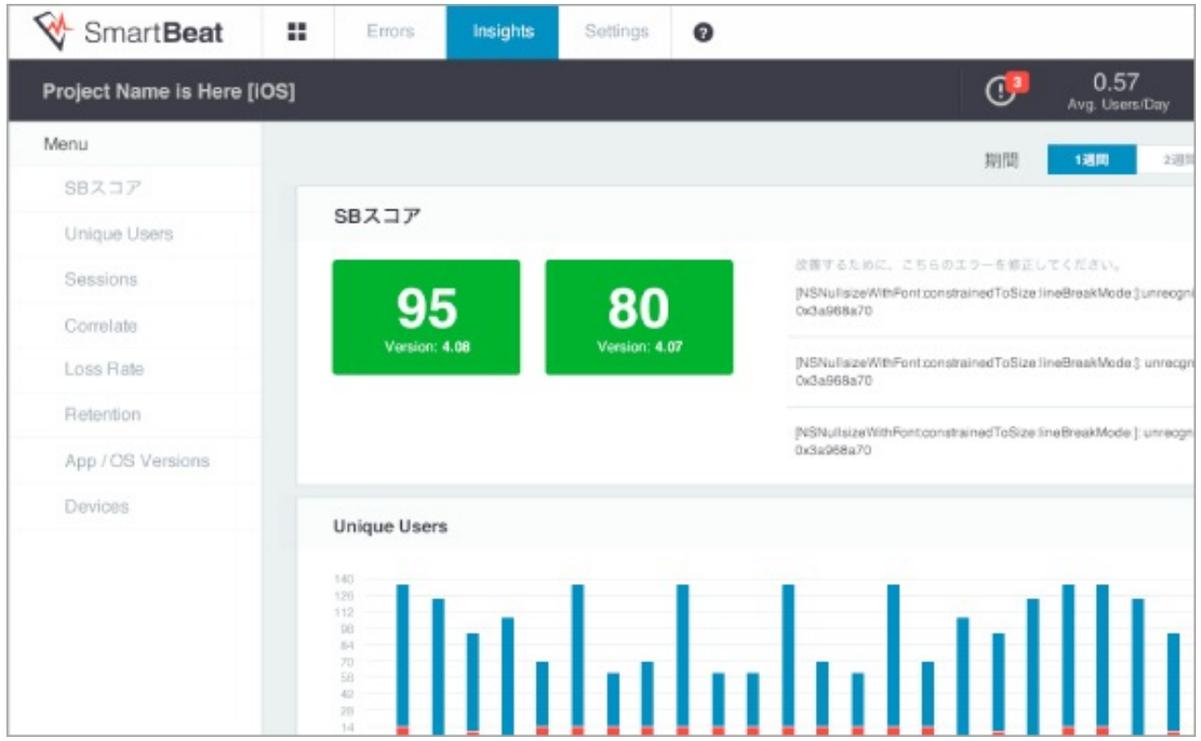
	占有率	エラー発生回数
SCL21	100%	2

発生ロケール

	占有率	エラー発生回数
Japanese	100%	2

SQUARE ENIX

バージョン別にスコア付け



アクセスユーザーのエラー被害率

…などなど便利な機能が満載！　詳しくはWEBで

http://smrtbeat.com/

SQUARE ENIX

バグとの戦いはつづく・・・

ソフトウェアを開発する限り
バグは常に戦わなければならない相手となります

終わりなき戦いを続けるとしても
バグに立ち向かう為の装備を忘れてはなりません

あなたの開発の道が無事でありますように・・・

ご清聴ありがとうございました

おつかれさまでした
このままでんげんを　おきりください

SQUARE ENIX