DEBUGGING FAST TRACK

» Inspirado na apresentação

Debugging Distilled with Xcode 6 & friends

» Kendall Helmstetter Gelner (@kendalldevdiary)

TALES PINHEIRO DE ANDRADE
@TALESP

» Mestre em computação pelo IME-USP

» Desenvolvedor C desde 2002

» Objective-C desde 2007 (coisas basicas antes do iPhone!)

» iOS desde 2010

“WE CAN NOT SOLVE OUR PROBLEMS WITH THE SAME LEVEL OF THINKING THAT CREATED THEM”

Albert Einstein/Carl Jung

“The first moral of the story is that program testing can be used very effectively to show the presence of bugs but never to show their absence.”
Edsger W. Dijkstra

On the reliability of programs.

TÓPICOS

» Depuração visual

» Depurando fluxo

» Depurando estado

DEPURAÇÃO VISUAL

DEPURAÇÃO VISUAL

- Simulador ajuda com aspectos visuais
  - Animações lentas
  - Mesclagem ou desalinhamento grafico

CORE ANIMATION

» Performance de UI mensuravel

» Inspeção visual de atributos aocultos

» Exame limitado da estrutura de aplicações de terceiros

XCODE 6: VIEW DEBUGGING

- Dispositivos iOS 8+
- Ativado via ```Debug -> View Debugging -> Capture View
  - Paralisa a aplicação e monta hierarquia de views

REVEAL

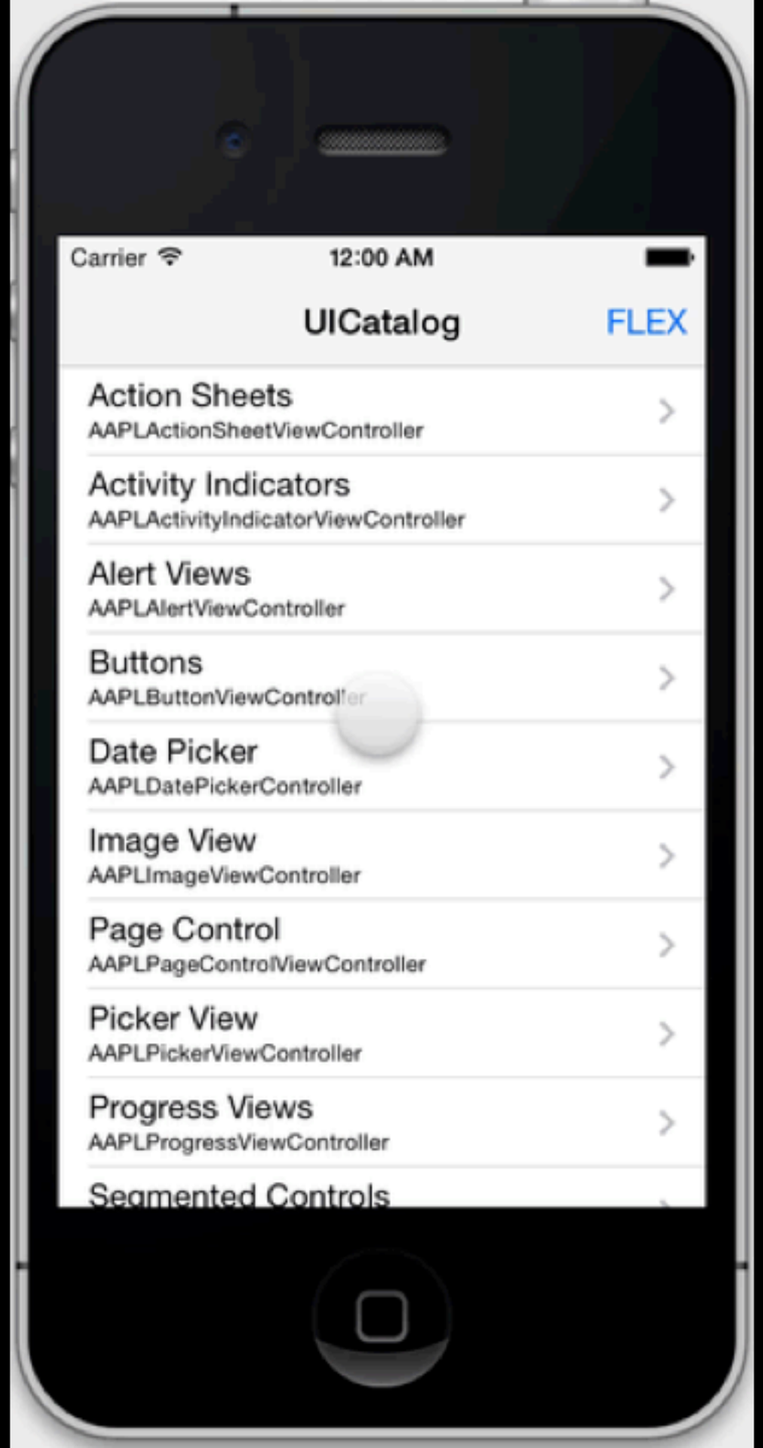
- revealapp.com - trial disponivel, $89
  - Mostra hierarquia de forma 2D ou 3D, permite manipulação das propriedades
  - Integração simples
  - Facil de usar/integrar
    - static library
    - Dynamic Library
    - cocoapod

FLEX

- FLEX: Flipboard EXplorer
- "Debug" visual in-app criado pelo Flipboard
- cocoapods
  - inspeção de elementos
  - navegação na arvore de subviews
  - manipulação de atributos

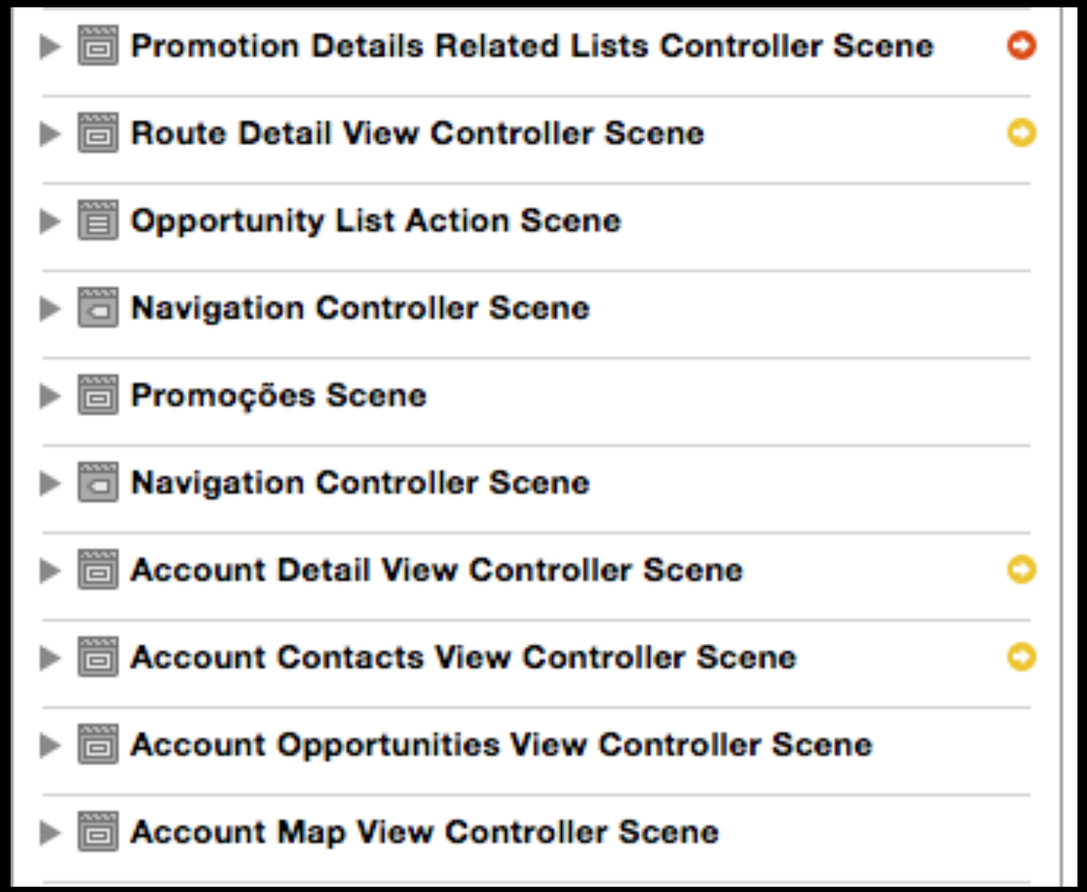
Para exibir

```
[[FLEXManager sharedManager] showExplorer];
```



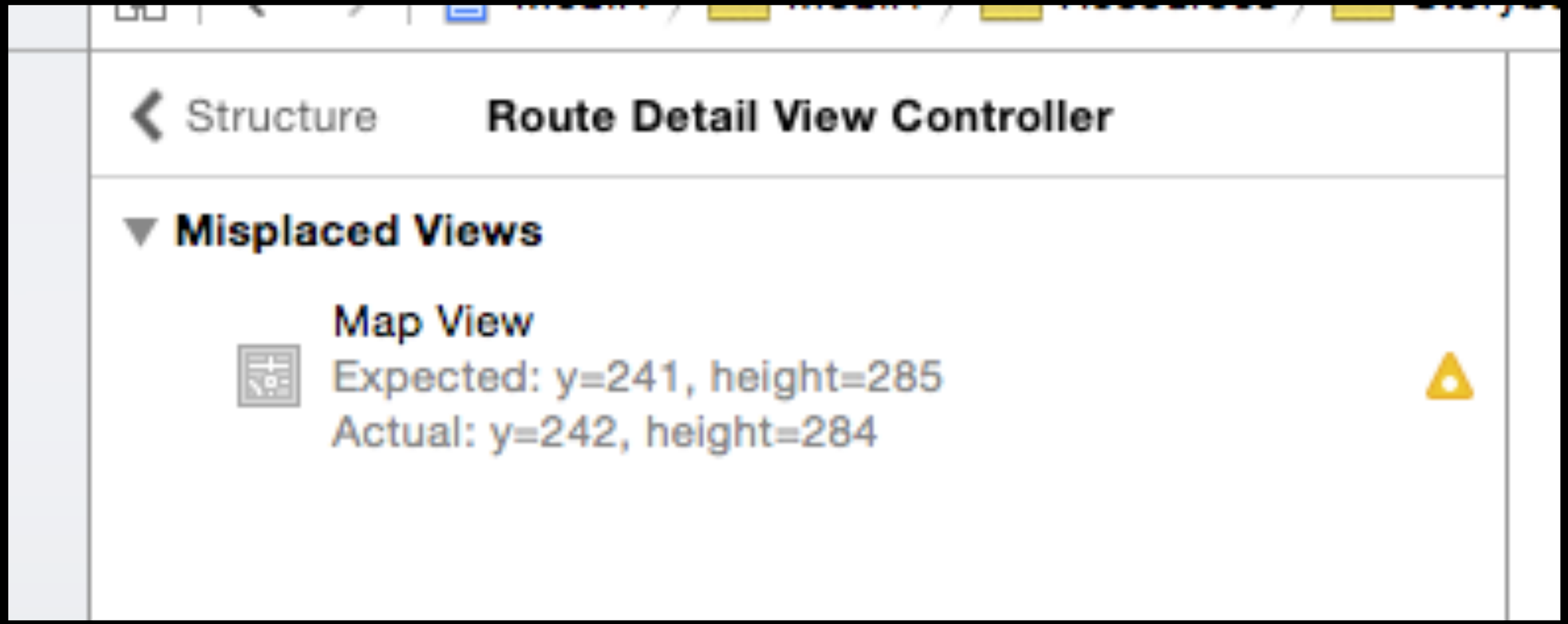
XCODE 6: CONSTRAINT DEBUGGING

» warnings do Interface Builder



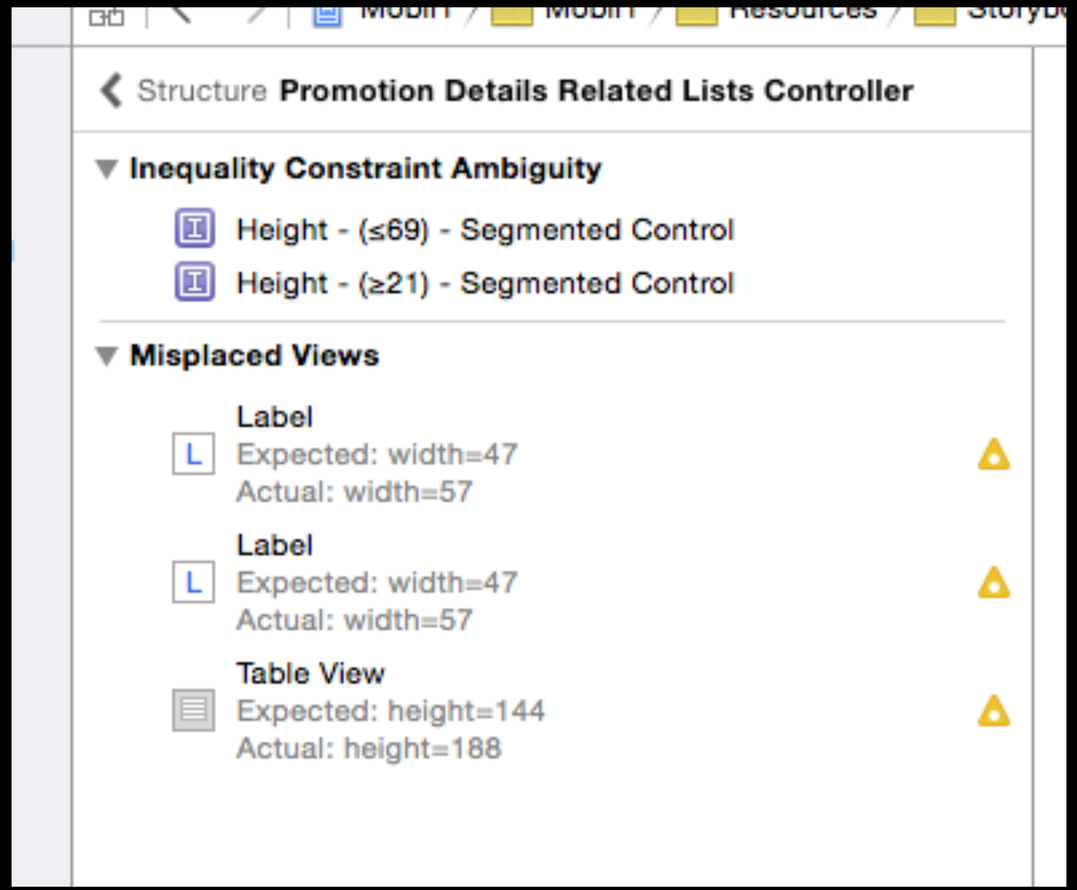
XCODE 6: CONSTRAINT DEBUGGING

» warnings do Interface Builder



XCODE 6: CONSTRAINT DEBUGGING

» warnings do Interface Builder



XCODE 6: CONSTRAINT DEBUGGING

» warnings do Interface Builder

  » View Debugging exibe constraints

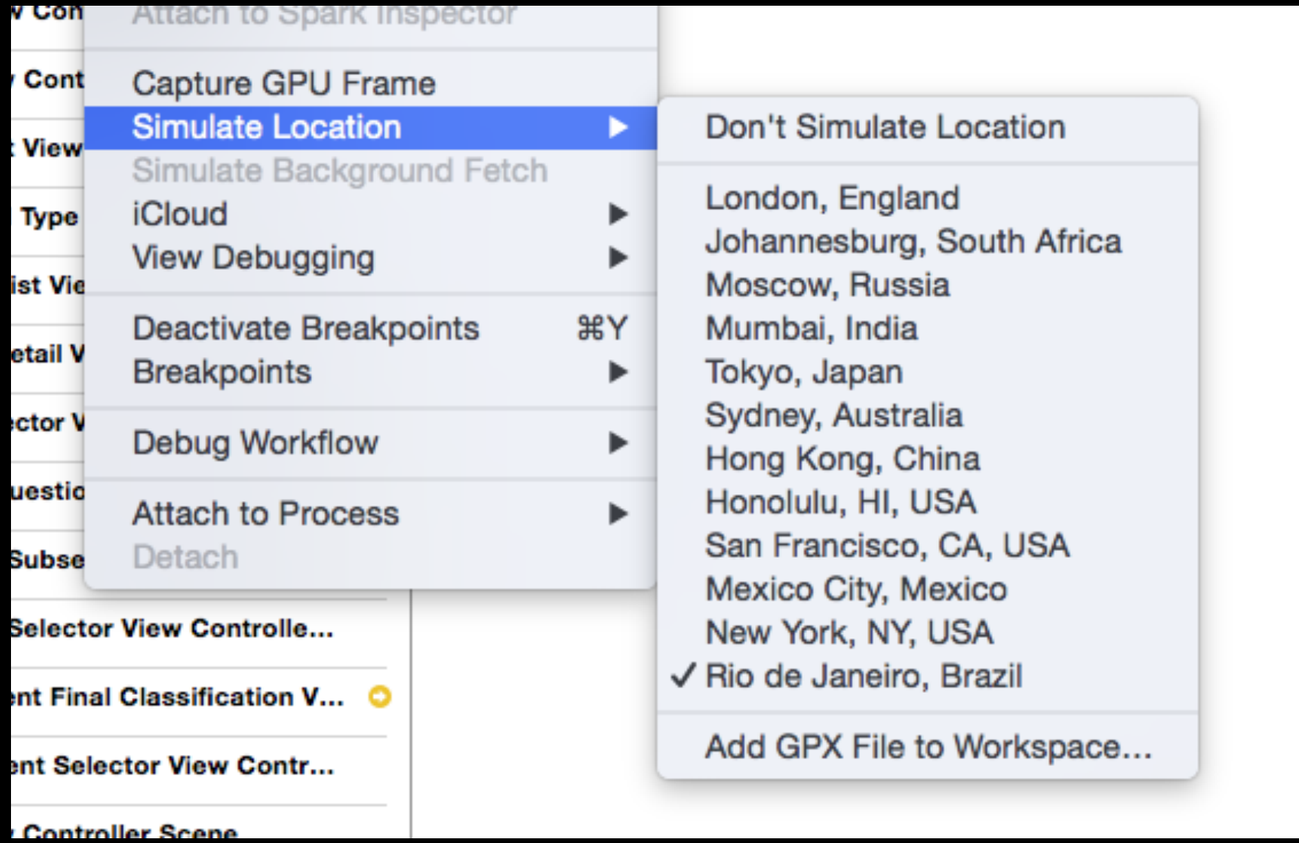
XCODE 6: CONSTRAINT DEBUGGING

» Pense sobre restrições como largura / altura mais a informação de onde algo deve ir.

» Um único erro pode cascatear para muitos, assim que encontrar a raiz do que a restrição problema está vinculado.

SIMULANDO A LOCALIZAÇÃO

- É possível configurar manualmente a localização no simulador;
- Usar arquivo GPX (simulando um caminho)
- Automator permite simulação mais precisa (direção e velocidade)
  - Apenas no simulador, não no dispositivo



BREAKPOINTS

» mais util quando ele não para

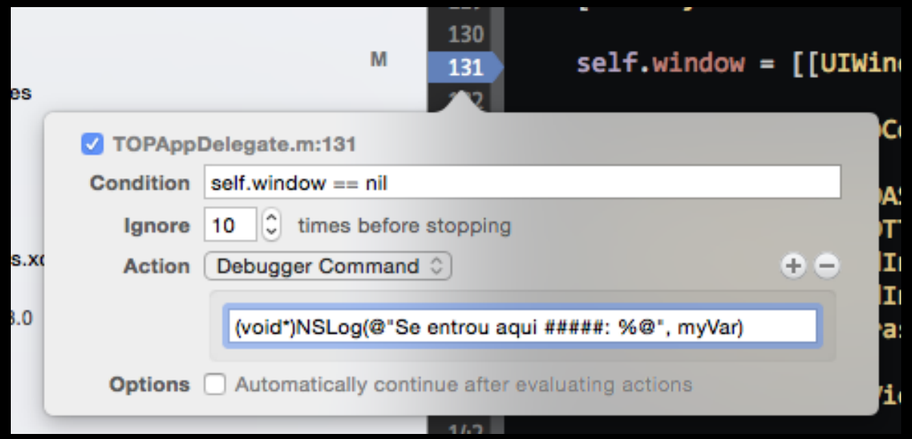
» uso de expressões como condição de parada

» Expressão deve ser usada na linguagem onde o código está

BREAKPOINTS
CONDIÇÕES E EXPRESSÕES

» lldb deve poder interpretar o breakpoint

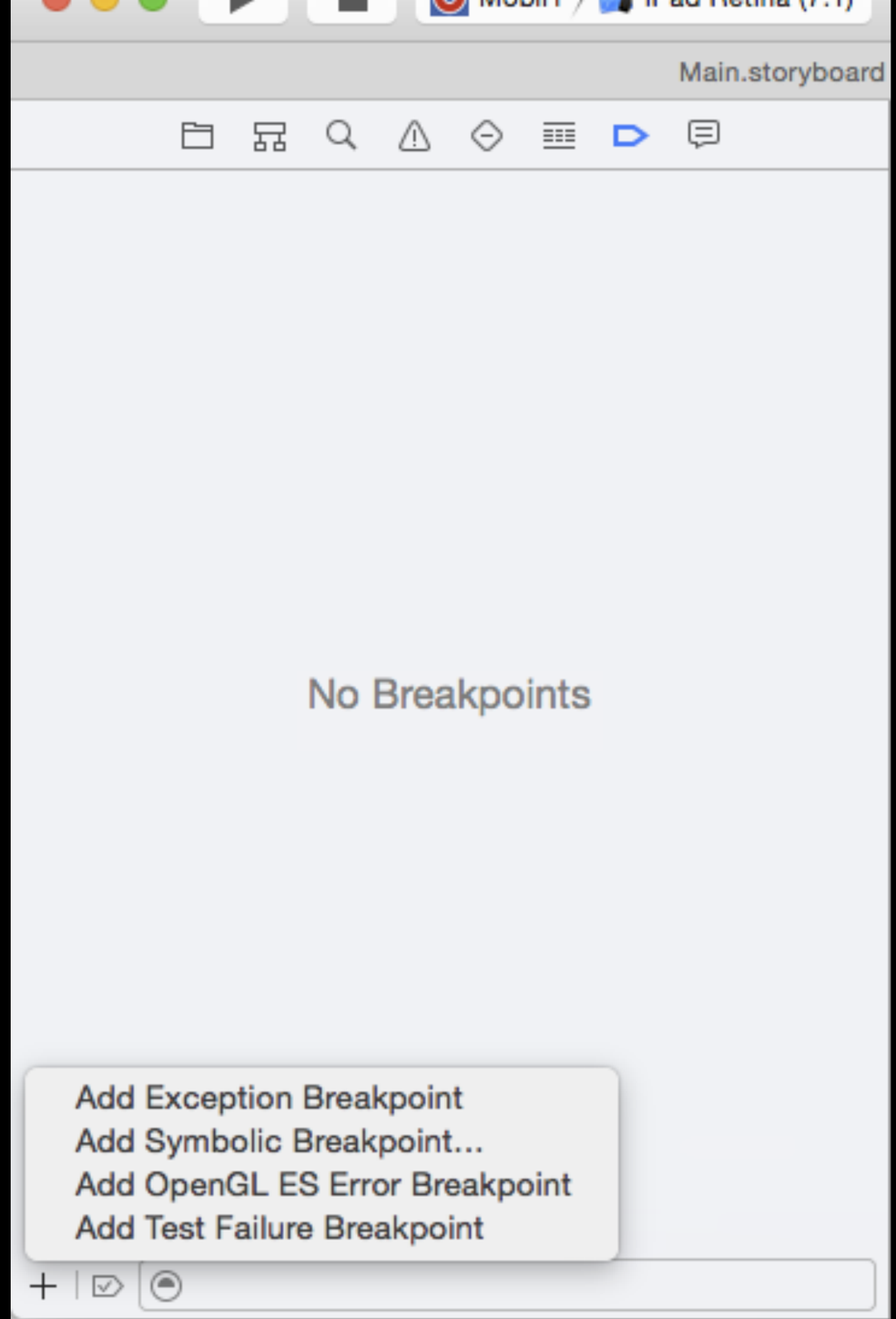
» com breakpoint "auto-continue", é possível aidionar pequenos blocos de código para adicionar logica on the fly



BREAKPOINTS

» Exception breakpoints

» Symbolic Breakpoints

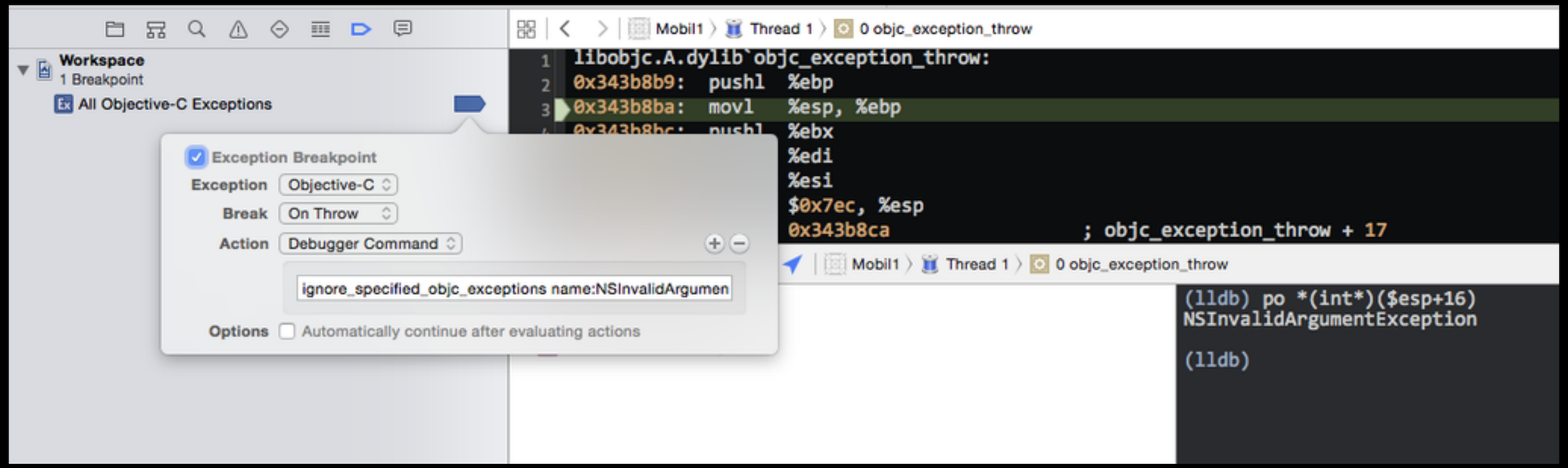


EXCEPTION BREAKPOINT

Selectively ignoring Objective-C exceptions in Xcode

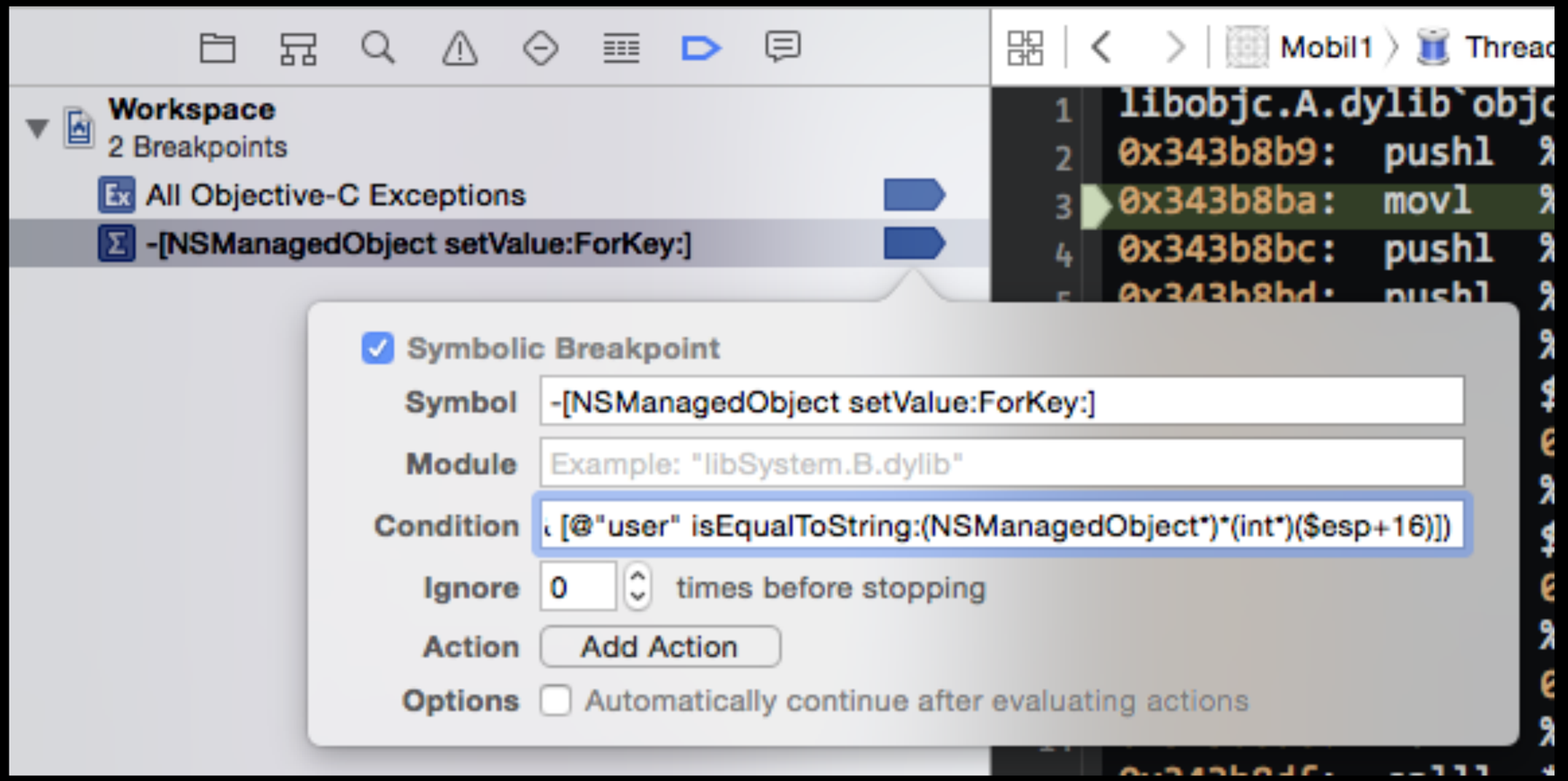
Adicione a linha abaixo no arquivo ~/.lldbinit

```
command script import ~/Library/lldb/ignore_specified_objc_exceptions.py
```



SYMBOLIC BREAKPOINT

Permite parar na chamada de um método



SYMBOLIC BREAKPOINT

Permite parar na chamada de um método

```
([@"User" isEqualToString:[(NSEntityDescription*)[(NSManagedObject*)*(int*)($esp+4) entity] name]] &&
    [@"contact" isEqualToString:(NSManagedObject*)*(int*)($esp+16)]) ||
([@"Contact" isEqualToString:[(NSEntityDescription*)[(NSManagedObject*)*(int*)($esp+4) entity] name]] &&
    [@"user" isEqualToString:(NSManagedObject*)*(int*)($esp+16)])
```

LOG MELHORADO

» Função/método + número da linha:

```
#define ALog(format, ...) DLog((@"%s [L%d] " format), __PRETTY_FUNCTION__, __LINE__, ##__VA_ARGS__)
#define DLog(format, ...) ALog(format, ##__VA_ARGS__)
```

» Swift não tem #DEFINE, use bibliotecas externas

» XCGLogger

LOG MELHORADO

» Opção melhor para Objective-C: CocoaLumberjack

  » niveis de debugs

  » cores no console via plugin XcodeColors (Alcatraz)

  » loggers para serviços de terceiros

```
[DDLog addLogger:[DDASLLogger sharedInstance]];
[DDLog addLogger:[DDTTYLogger sharedInstance]];
[[DDTTYLogger sharedInstance] setColorsEnabled:YES];
[[DDTTYLogger sharedInstance] setForegroundColor:[UIColor blueColor]
                                 backgroundColor:nil
                                         forFlag:LOG_FLAG_INFO];
[DDLog addLogger:[CrashlyticsLogger sharedInstance]];
```

LOG MELHORADO

» Logs são pesados, use com cuidado

» Use blocks para logs pesados

» Background logging com Notificações locais (neanderthal debugging - Michael Oliver) (APENAS EM DESENVOLVIMENTO!!!1!!1!1!!)

INSTRUMENT: TIME PROFILER

» Forma ótima de verificar onde o app está lento

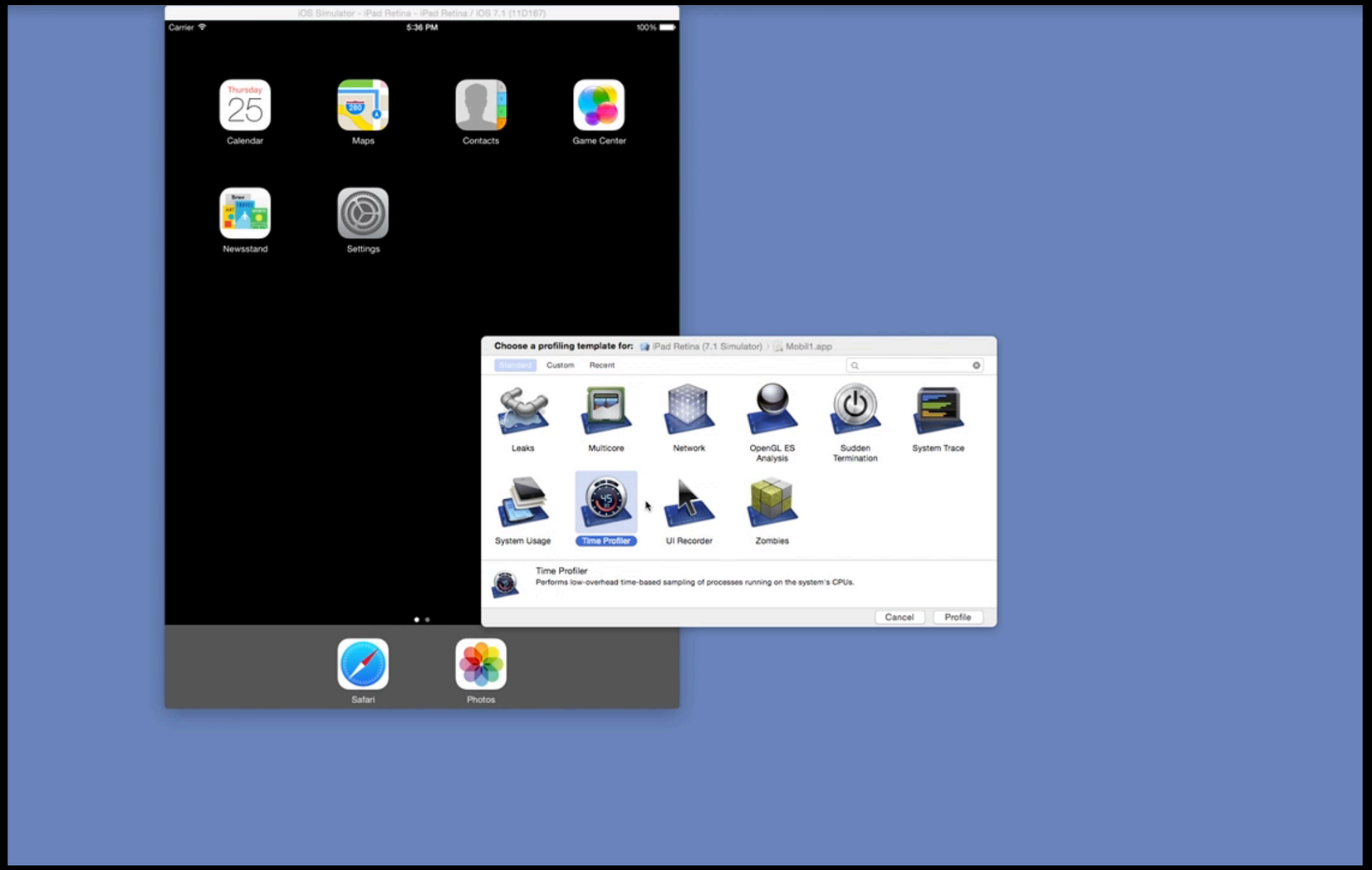
» Ajuda a focar primeiro onde tem mais problema de performance

» Permite visualizar por nucleo de processador

» Xcode Profiler serve como guide line

» Record Waiting threads ajuda

INSTRUMENT: TIME PROFILER



LLDB - EXPRESSIONS

» acessada via expr no console do lldb

» Permite rodar código real em run-time

» permite definir variáveis globais com prefixo (use id para tipos complexos)

» patcht em real-time!

LLDB - COMANDOS UTEIS

- po recursiveDesccription: dump da hierarquia de views
- crash na chamada de objc_msgsend?
  - p (char *)$ecx (simulator)
  - p (char *)$r1 (device)
- Watchpoints
  - watchpoint set variable self->_myProp
  - watchpoint delete <number>

OUTRAS FERRAMENTAS

- Network link conditioner
  - Parte do "Hardware IO Tools"
  - Afeta todo o sistema
- Charles Proxy
- Chisel
- CoreDataPro
- Core Data Editor