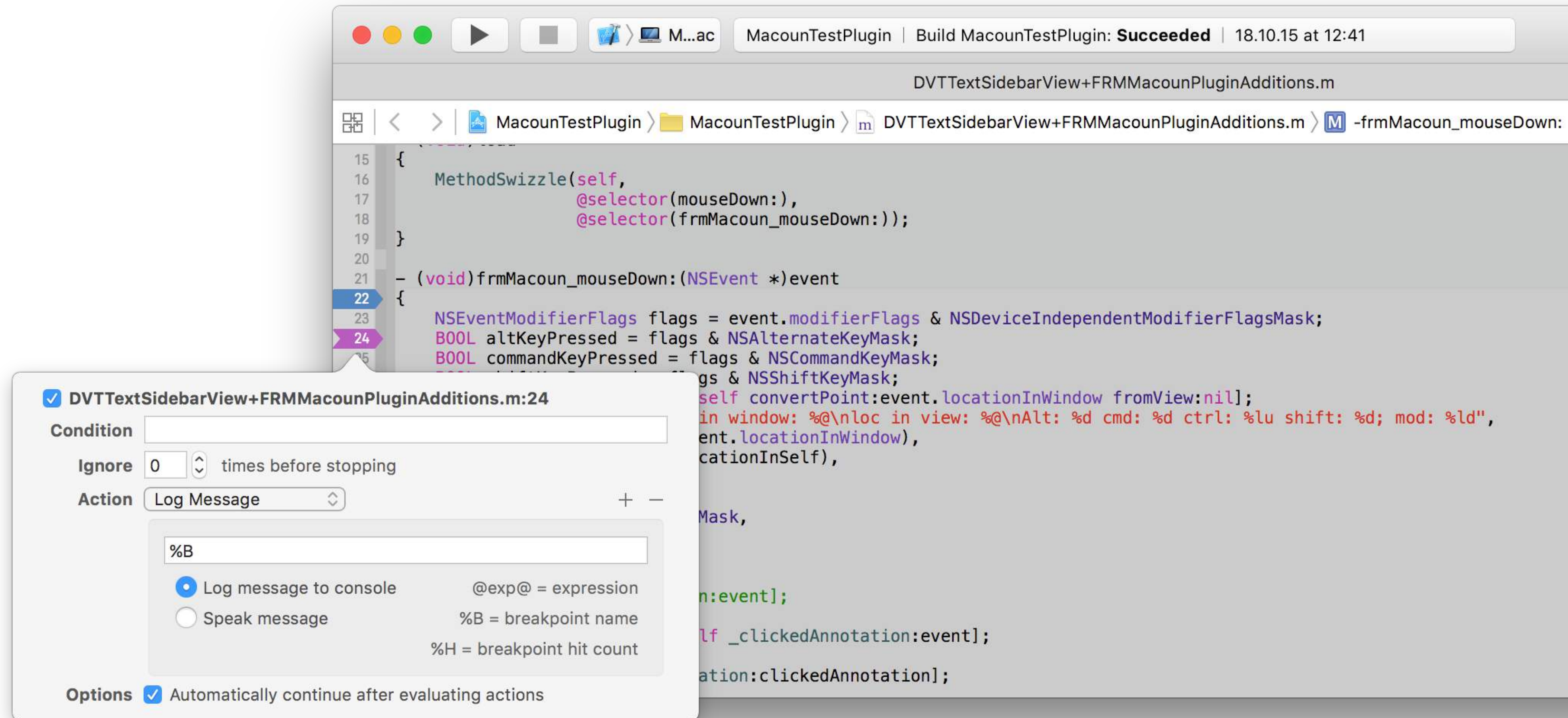


**Macoun**

# Xcode erweitern

Friedrich Markgraf

# Wir schreiben ein Plug-In



# Wir schreiben ein Plug-In

- Xcode von innen
- Plug-In-Aufbau
- Praxis
  - Debugger
  - Runtime
  - Disassembler
- Publizieren mit Alcatraz

# Teil I: Xcode

Demo

# Teil 2: Anfang

# Aufbau eines Plug-Ins

- Bundle mit Endung .xcplugin
- in ~/Library/Application Support/Developer/Shared/Xcode/Plug-ins
- Die erste geladene Klasse implementiert
  - + (void)pluginDidLoad:(NSBundle \*)plugin



# Info.plist

Schlüssel	Wert
XC4Compatible	YES
XCPluginHasUI	NO
DVTPlugInCompatibilityUUIDs	Array mit UUIDs

# Info.plist

Schlüssel	Wert
XC4Compatible	YES
XCPluginHasUI	NO
DVTPlugInCompatibilityUUIDs	Array mit UUIDs

```
defaults read /Applications/Xcode.app/Contents/Info DVTPlugInCompatibilityUUID
```

# Installieren und debuggen

- Build Settings:

Setting	Wert
Skip Install	NO
Installation Build Products Location	\$(HOME)
Installation Directory	/Library/Application Support/Developer/Shared/Xcode/Plug-ins

- Scheme Executable auf Xcode setzen
- RedXcode-Plugin ([github.com/orta/RedXcode](https://github.com/orta/RedXcode))

# Template

- [github.com/kattrali/Xcode-Plugin-Template](https://github.com/kattrali/Xcode-Plugin-Template)
- Oder Alcatraz → Templates → Xcode Plugin

Demo

# Teil 3: Class-Dump

Demo

# Class-Dump komplett

- Class-Dump erzeugen:

```
cd Applications/Xcode.app/Contents

find . \( -iname "*.framework" -or -iname *.ideplugin \)
! -ipath *.platform* ! -ipath *.app* -print0 |
xargs -0 -o -I % class-dump --arch x86_64 --sdk-mac -I -H
-o ~/Documents/XcodeClassDump/% %
```

- Dateiendungen entfernen:

```
cd ~/Documents/XcodeClassDump

find . \( -iname "*.framework" -or -iname *.ideplugin \) -print |
while read filename; do mv -v "$filename" "${filename%.*}"; done
```



# Class-Dump

- Auf Frameworks und Plug-Ins in Xcode.app anwenden
- Endungen entfernen
- Alles in ein Xcode-Projekt
- Download: [stevenygard.com/projects/class-dump/](http://stevenygard.com/projects/class-dump/)
- Script: [gist.github.com/fzwo/dc0533fe1ab6a124bd00](https://gist.github.com/fzwo/dc0533fe1ab6a124bd00)

# Teil 4: F-Script

# Demo

# F-Script

- Runtime-Klassen- und Methoden-Browser
- Smalltalk-ähnliche Workspaces
- Visuell Instanzen wählen, inspizieren und damit interagieren
- [www.fscript.org](http://www.fscript.org)

# F-Script Injection

- F-Script 10.10
- FScript.framework in /Library/Frameworks
- System Integrity Protection abschalten
- Per Debugger injecten

# Teil 5: Der erste Klick

Demo

# Class Categories

- Klassen um eigene Methoden erweitern
- Namenskonflikte vermeiden
- Symbols im Projekt selbst definieren (Class-Dump)
- Framework importieren



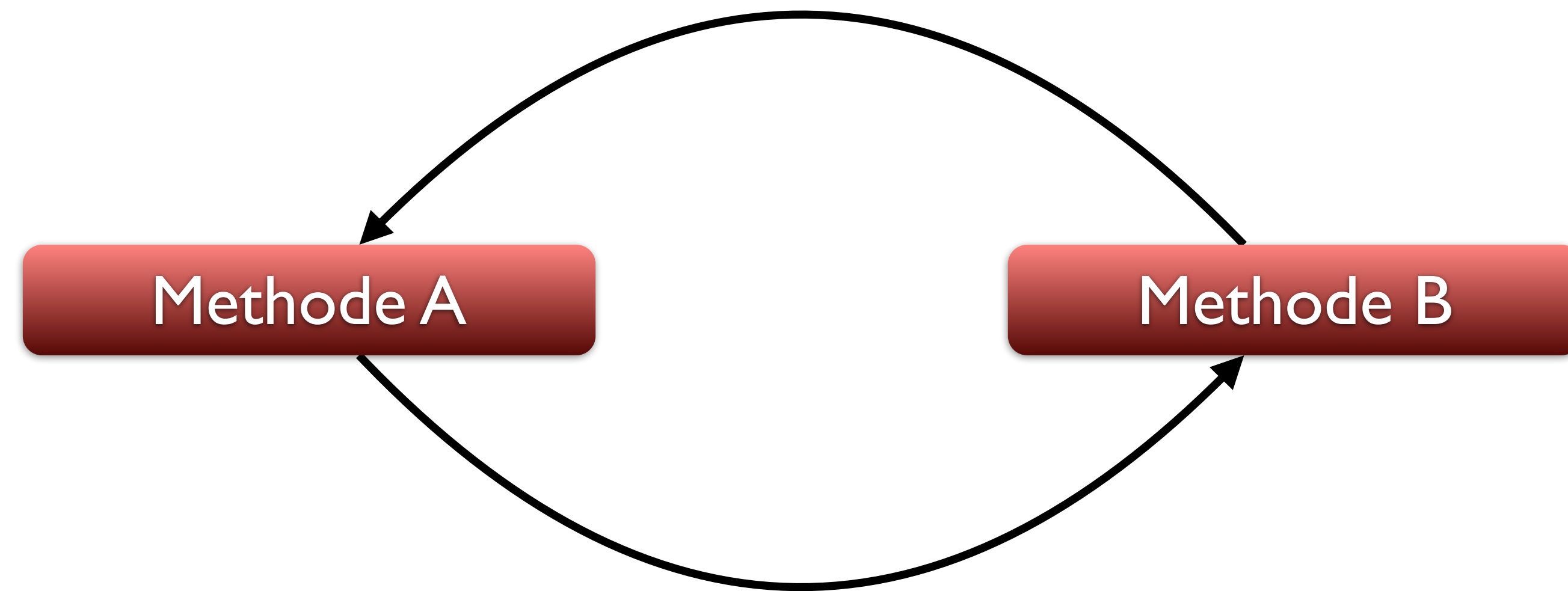
# Frameworks und Linker

- Frameworks aus Xcode.app/Resources/... ins Projekt ziehen
- Pfad „Relative to Developer Directory“
- Framework Search Path:

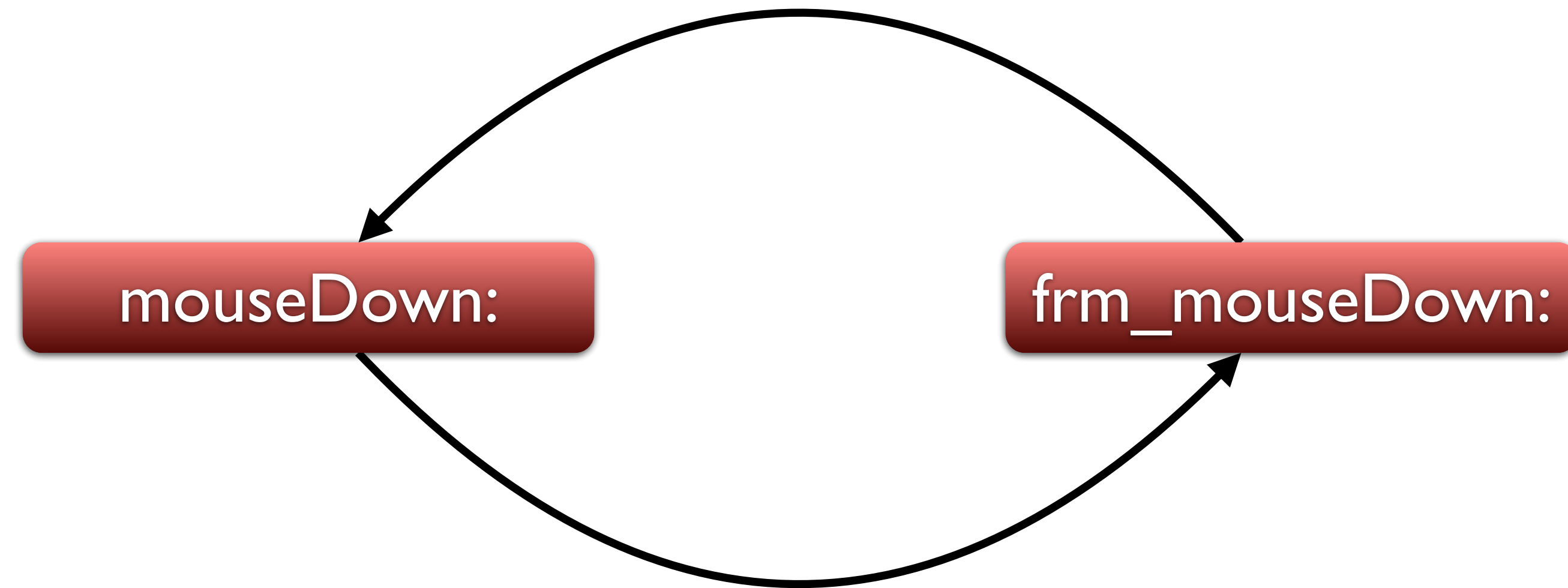
Wert
\$(inherited)
\$(SYSTEM_APPS_DIR)/Xcode.app/Contents/SharedFrameworks
\$(SYSTEM_APPS_DIR)/Xcode.app/Contents/Frameworks

# Teil 6: Method Swizzling

# Method Swizzling



# Method Swizzling



Demo

# Method Swizzling

- Methoden durch eigene aus Category ersetzen
- Swizzlen in `+ (void)load`
- Original-Methode bleibt unter neuem Namen erreichbar
- Original-Methode in eigener Methode aufrufen
- Namenskonflikte vermeiden

# Teil 7: Hopper Disassembler

Demo



# Disassembly mit Hopper

- Binaries mit Cmd+Shift+O laden
- Symbol in der linken Seite suchen
- Pseudocode mit Alt+Return
- [www.hopperapp.com](http://www.hopperapp.com) (89 €)
- Teure Alternative: IDA Pro ([hex-rays.com](http://hex-rays.com); 1019 €)

# Teil 8: Praxis

Demo

# Praxis

- Kombination aus Class-Dump-Headers, Hopper und Ildb
- Raten und in Xcode suchen
- Geduld

# Praxis

- Kombination aus Class-Dump-Headers, Hopper und Ildb
- Raten und in Xcode suchen
- Geduld

# Teil 9: Block

Demo

# Blöcke verstehen

- Blöcke sind Objekte
- `invoke`-Funktion mit self-Pointer
- Block Descriptor
- Signature
- [clang.llvm.org/docs/Block-ABI-Apple.html](http://clang.llvm.org/docs/Block-ABI-Apple.html)
- [developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ObjCRuntimeGuide/Articles/ocrtTypeEncodings.html](http://developer.apple.com/library/mac/documentation/Cocoa/Conceptual/ObjCRuntimeGuide/Articles/ocrtTypeEncodings.html)

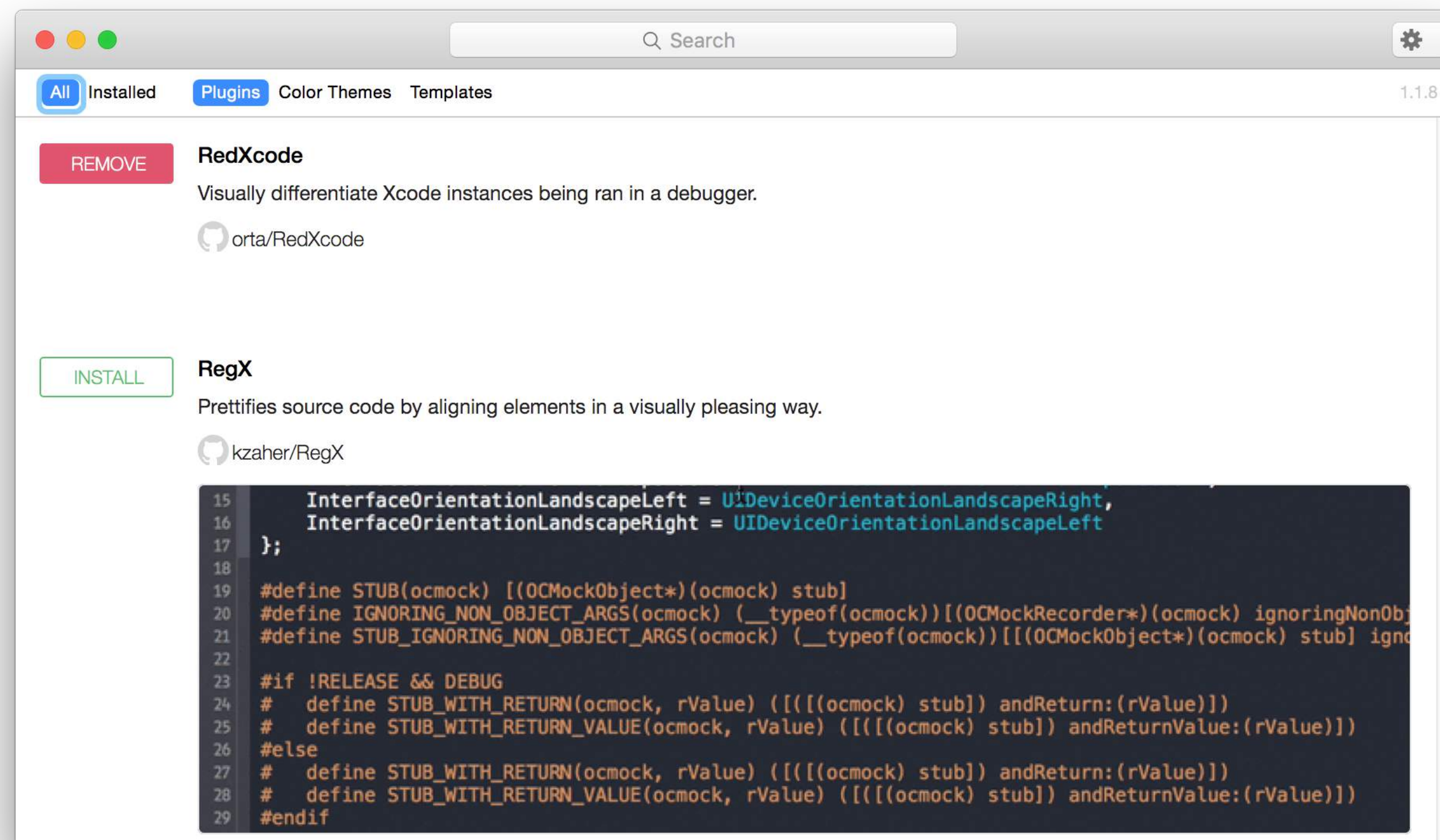


# Blöcke verstehen

- [ddeville.me/2013/02/block-debugging/](https://ddeville.me/2013/02/block-debugging/)
- [ddeville.me/2013/11/a-python-script-to-disassemble-a-block-in-lldb/](https://ddeville.me/2013/11/a-python-script-to-disassemble-a-block-in-lldb/)
- [github.com/ddeville/block-lldb-script](https://github.com/ddeville/block-lldb-script)

# Teil X: Publizieren

# Alcatraz



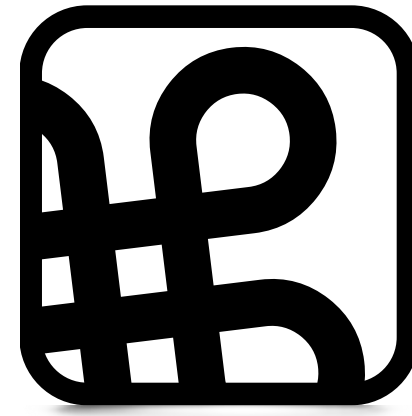
# Publizieren

- [github.com/fzwo/alcatraz-packages](https://github.com/fzwo/alcatraz-packages)
- Fork
- Zu packages.json hinzufügen
- Pull Request

[github.com/fzwo/FRMBreakFast](https://github.com/fzwo/FRMBreakFast)

Fragen?  
@fzwob

**Vielen Dank**



**Macoun**