

Как разобрать macOS на части

От исследования процессов
к изменению логики



Игорь Белов
iOS Engineer

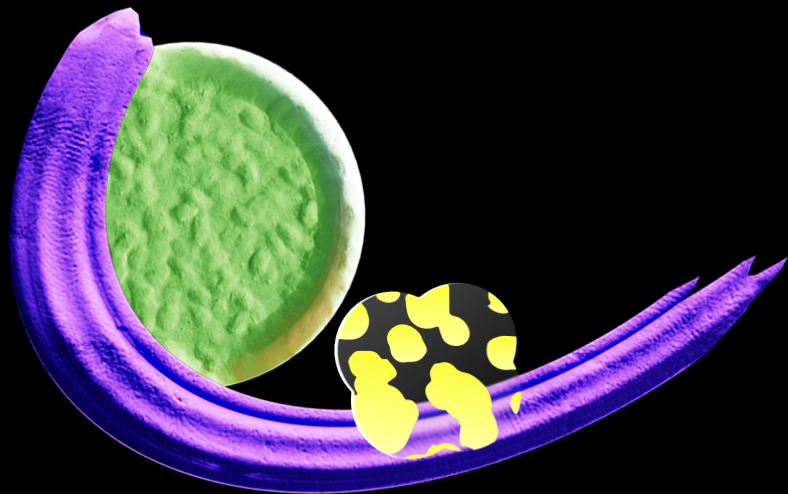
О себе



Игорь Белов iOS Engineer

Инженер в команде
Mobile Architecture Авито
и энтузиаст
в Computer Science

Больше шести лет занимаюсь iOS разработкой, люблю разбираться в фундаментальных темах Computer Science и изучать как вещи работают «под капотом»



Программа ведет себя неправильно

Программа ведет себя неправильно

Наше приложение

- ⊕ Есть исходники
- ⊕ Можем открыть в Xcode
- ⊕ Можем поставить breakpoints
- ⊕ Можем сделать print
- ⊕ Можем пересобрать

Чужой черный ящик

- ⊖ Есть только процесс или бинарь
- ⊖ Наблюдаем только поведение

Содержание

01 Исследование

Про LLDB и исследование
запущенных процессов

02 Внедрение

Про инъекцию своего кода
для изменения поведения

Содержание

01 Теория и подготовка

Про инструменты,
ограничения и окружение

03 Внедрение

Про инъекцию своего кода
для изменения поведения

02 Исследование

Про LLDB и исследование
запущенных процессов

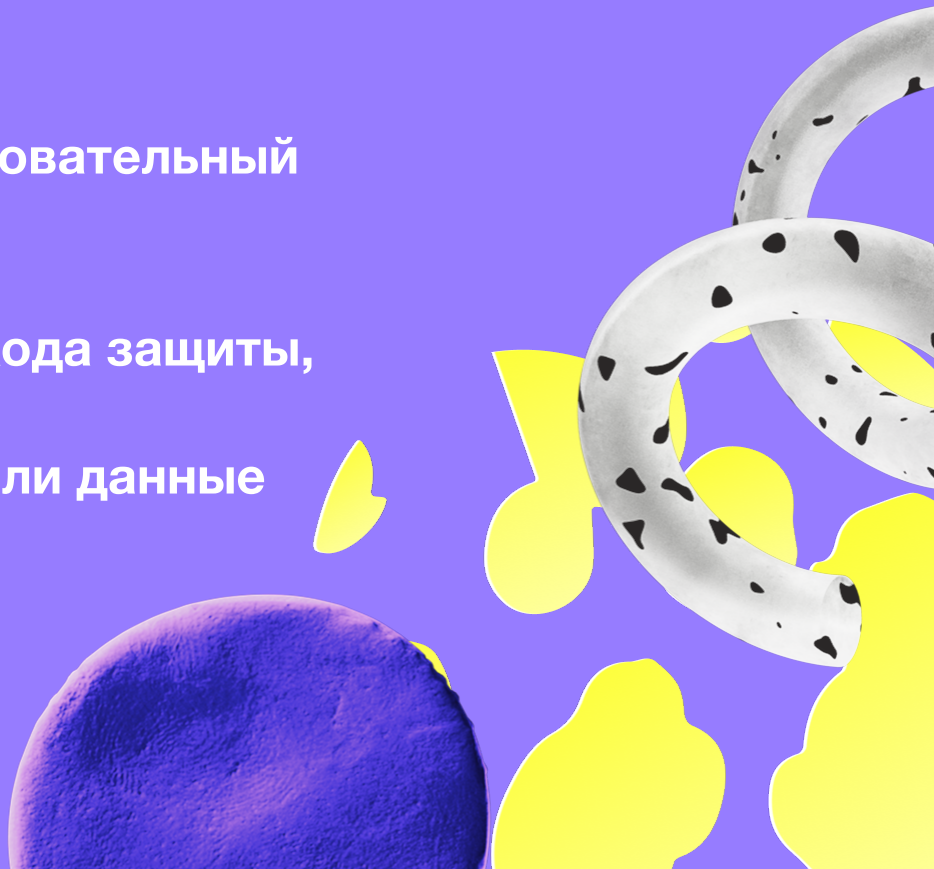
04 Дальнейшие шаги

Про литературу,
рекомендации и выводы

WARNING!

Доклад носит исключительно образовательный характер

Примеры не предназначены для обхода защиты, несанкционированного доступа и вмешательства в чужие сервисы или данные



Точка входа – LLDB

1 Подключиться к живому приложению

The screenshot displays the Xcode IDE interface for debugging a live application. On the left, the 'Process' pane shows 'GreedyKitExample' with system metrics: CPU (2%), Memory (57.7 MB), Disk (14.9 MB/s), Network (Zero KB/s), and FPS (0 FPS). Below this, the 'Thread' pane lists several threads, with 'Thread 1' selected, showing a stack of frames including '0 closure #1 in UIImageViewContr...', '1 UIImageViewController.localIma...', '2 UIImageViewController.addImag...', and '3 UIImageViewController.viewWill...'. The main editor shows the source code for 'UIImageViewController' with a breakpoint set at line 31, which is the 'return image' statement. The bottom console shows the current state of variables: 'path' is a file path and 'image' is a 'UIImage' object.

```
final class UIImageViewController: UIViewController {
    private lazy var contentView: ContentControlsView = {
    }()

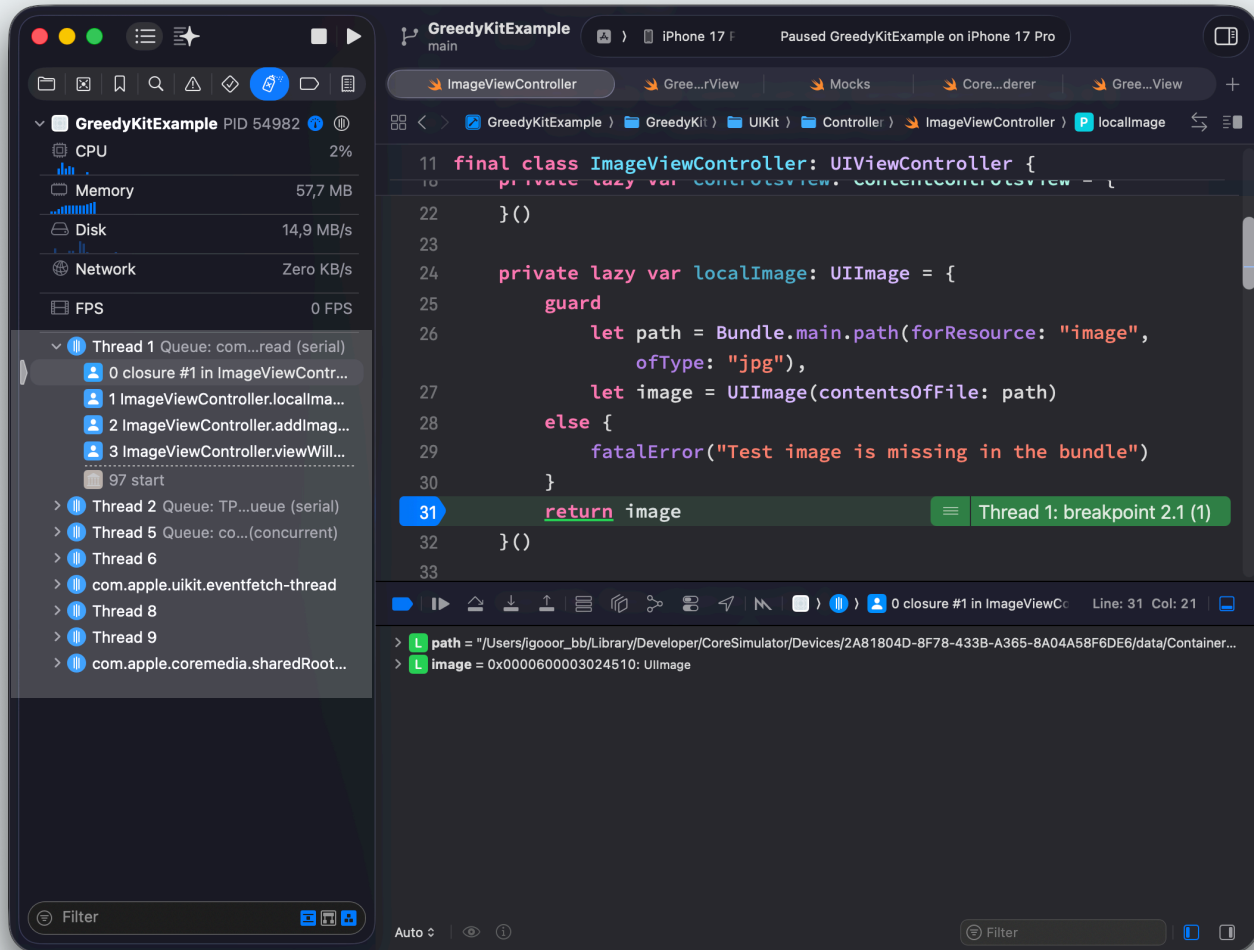
    private lazy var localImage: UIImage = {
        guard
            let path = Bundle.main.path(forResource: "image",
                                         ofType: "jpg"),
            let image = UIImage(contentsOfFile: path)
        else {
            fatalError("Test image is missing in the bundle")
        }
    }()

    return image
}()
```

path = "/Users/goor_bb/Library/Developer/CoreSimulator/Devices/2A81804D-8F78-433B-A365-8A04A58F6DE6/data/Container..."
image = 0x00006000003024510: UIImage

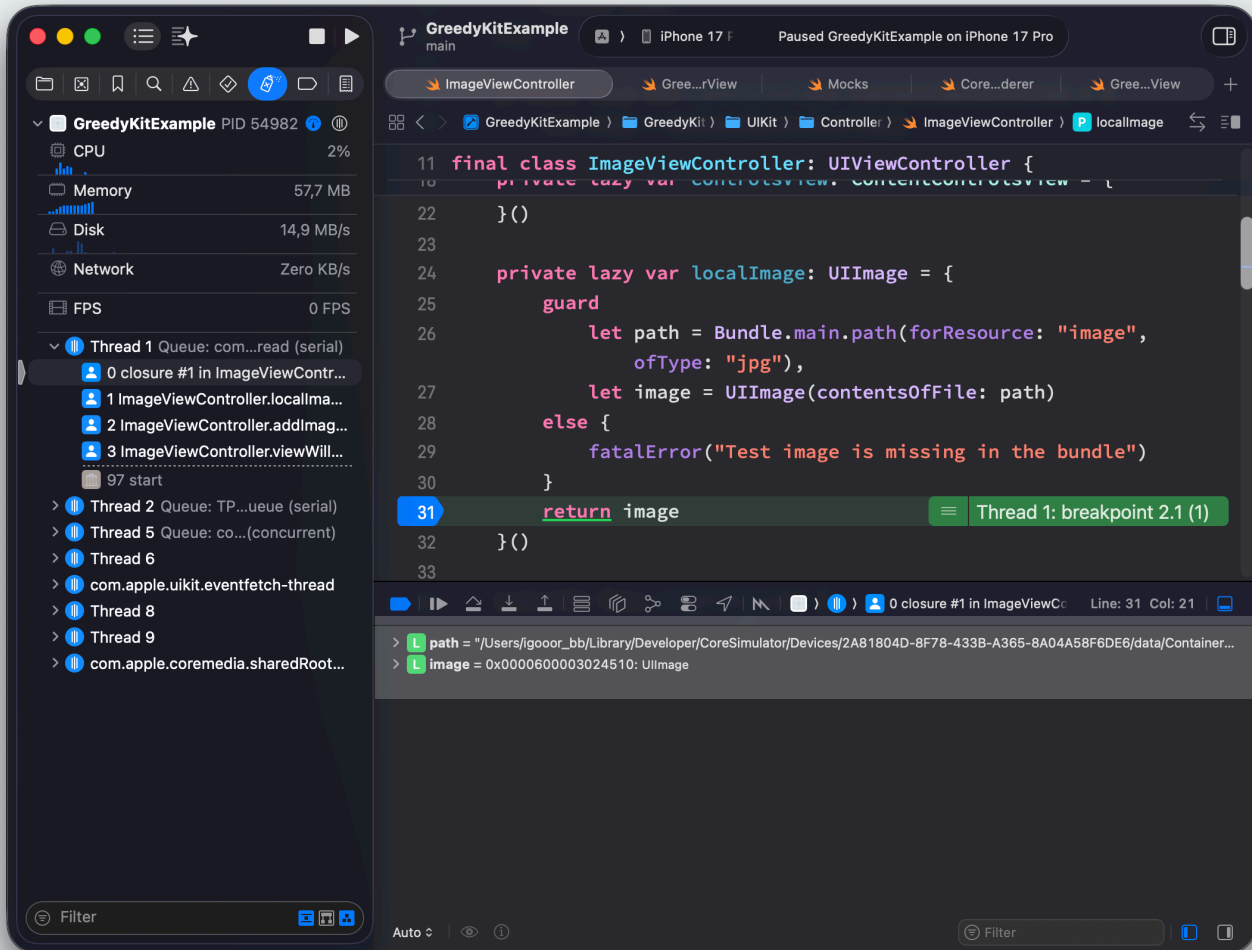
Точка входа – LLDB

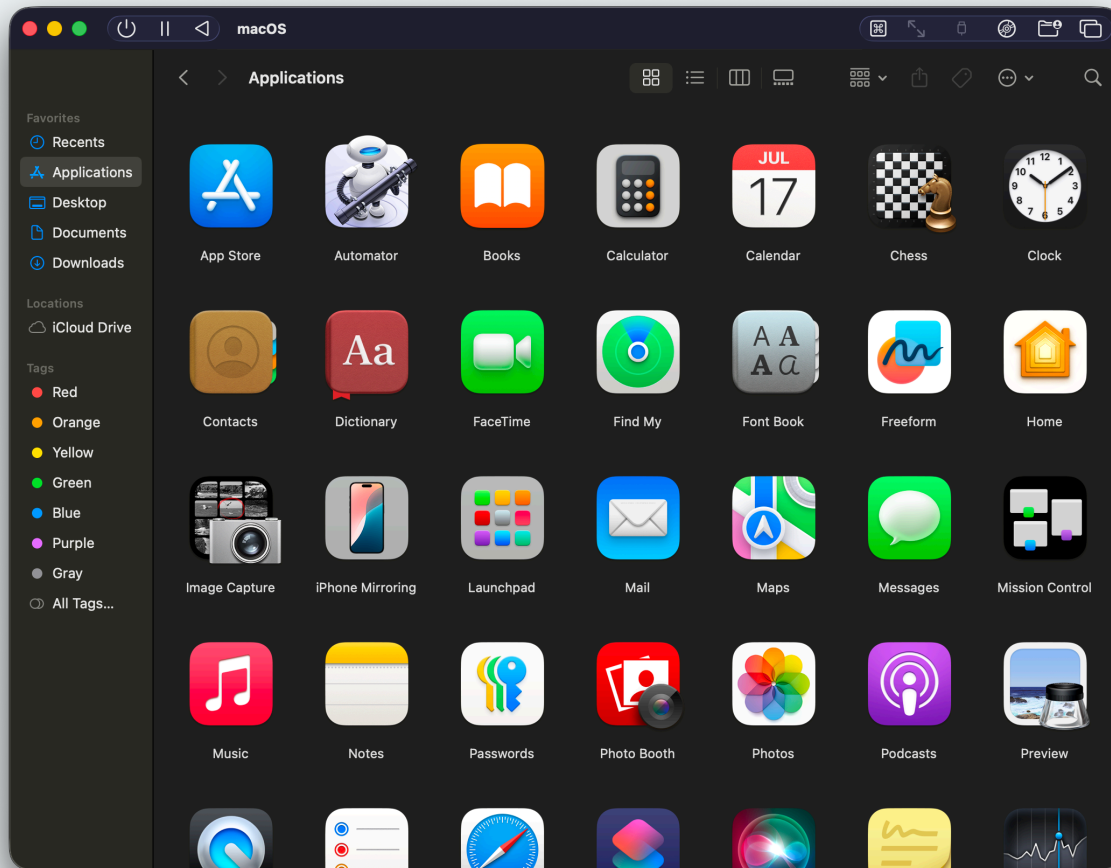
- 1 Подключиться к живому приложению
- 2 Остановиться в нужной точке

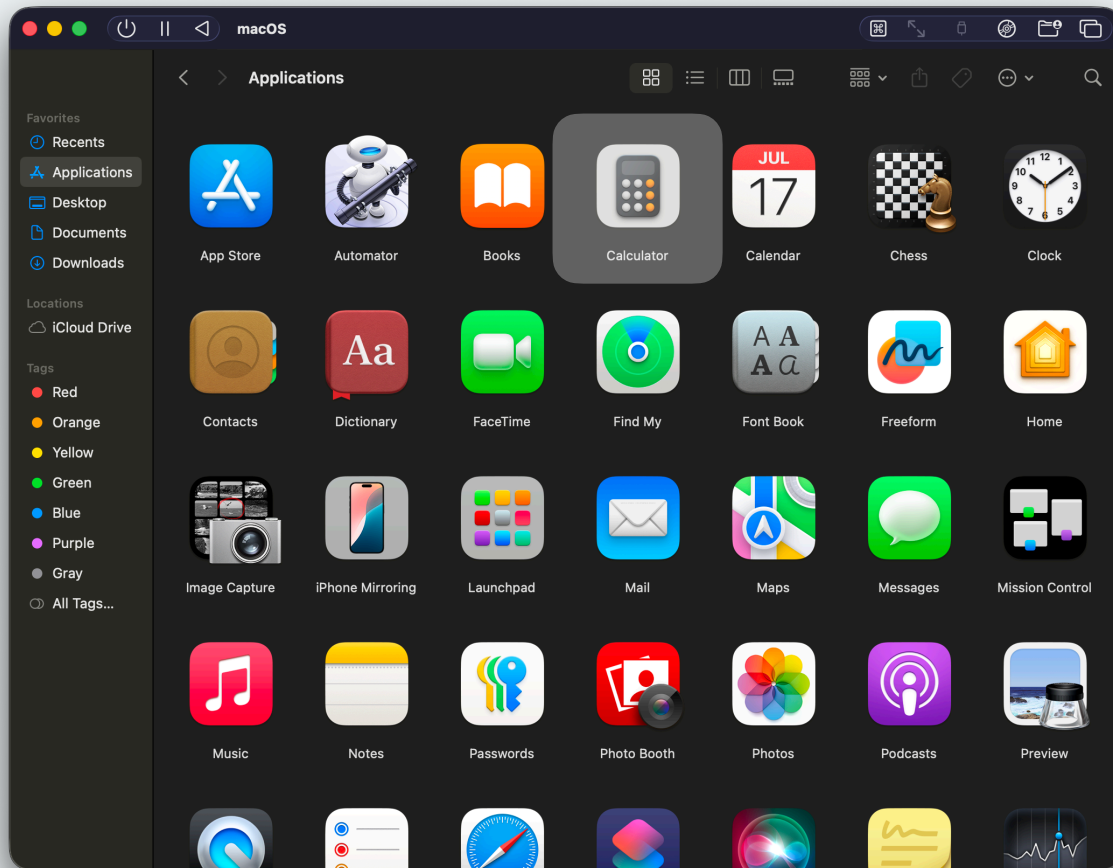


Точка входа – LLDB

- 1 Подключиться к живому приложению
- 2 Остановиться в нужной точке
- 3 Проверить, что процесс делает на самом деле

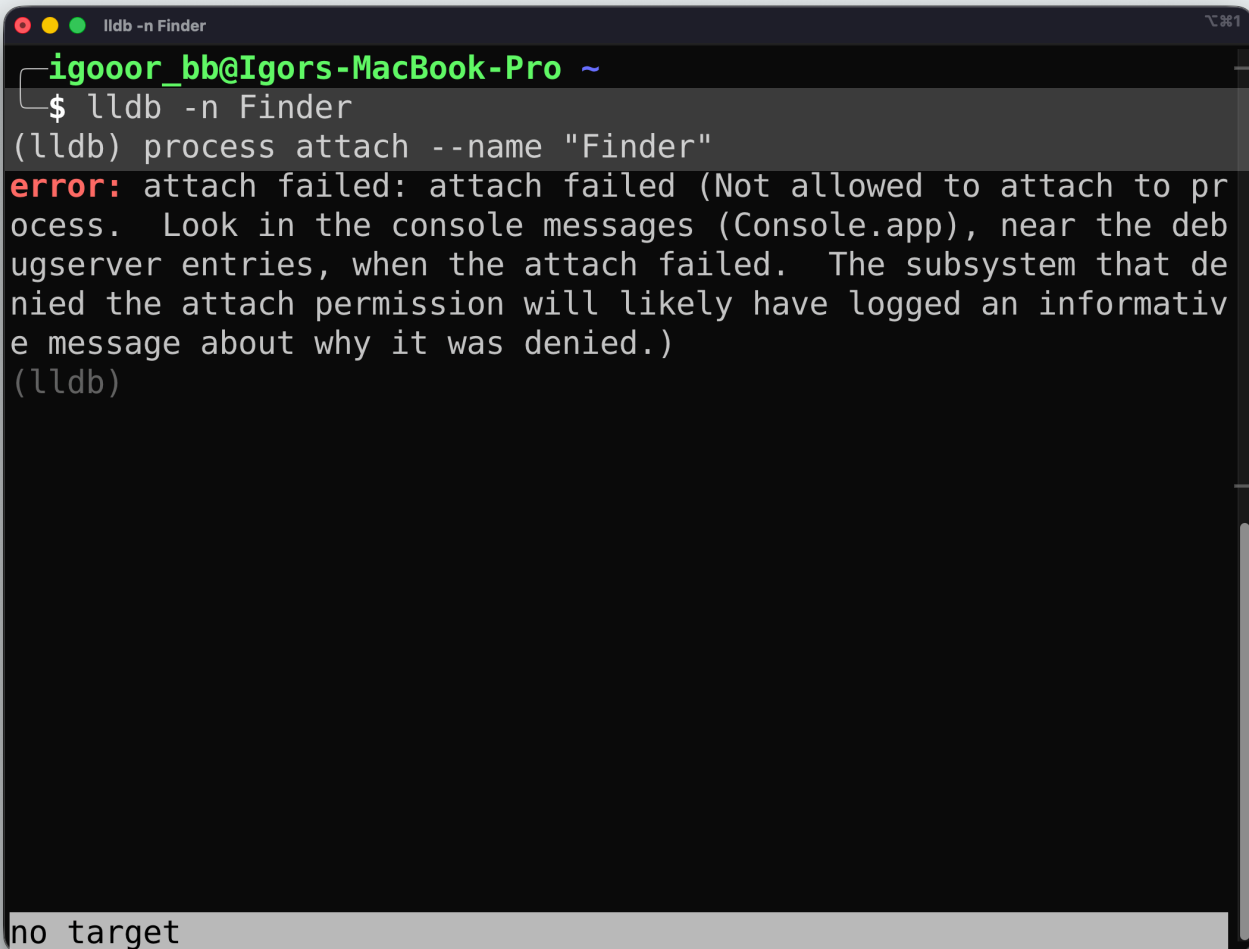






OS не обязана нас пускать

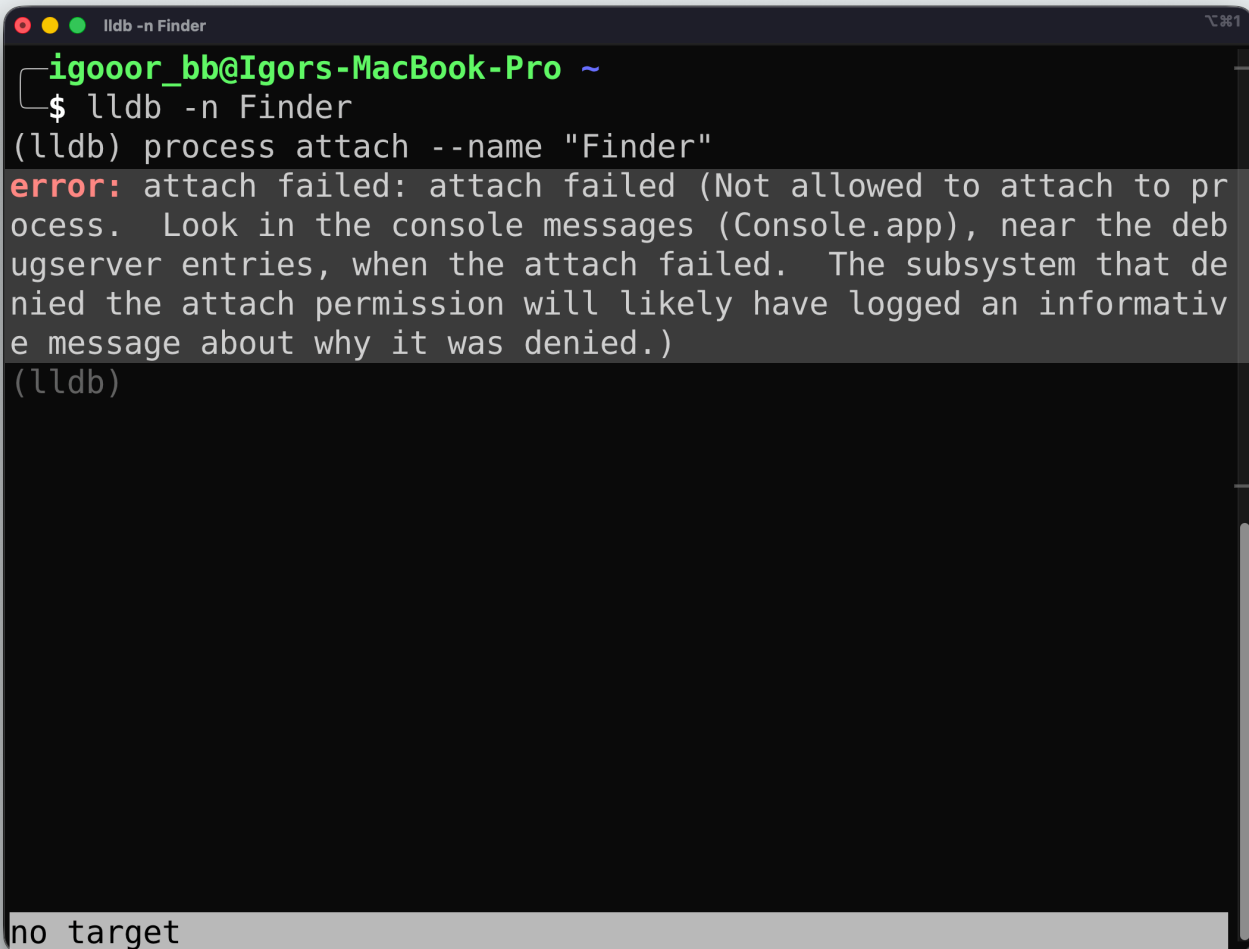
- 1 Системные процессы защищены



```
lldb -n Finder
igoor_bb@Igor's-MacBook-Pro ~
$ lldb -n Finder
(lldb) process attach --name "Finder"
error: attach failed: attach failed (Not allowed to attach to process. Look in the console messages (Console.app), near the debugger entries, when the attach failed. The subsystem that denied the attach permission will likely have logged an informative message about why it was denied.)
(lldb)
no target
```

OS не обязана нас пускать

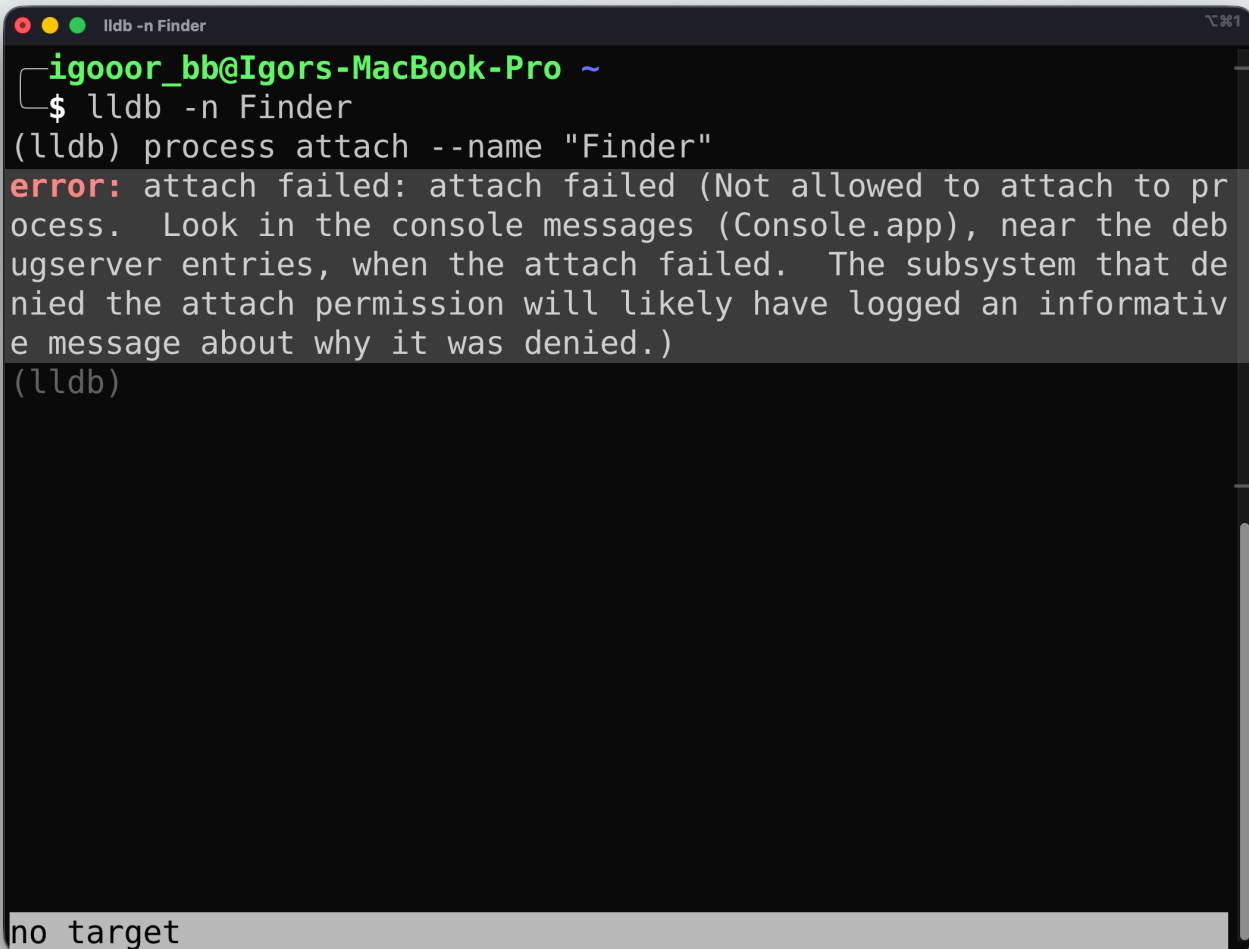
- 1 Системные процессы защищены
- 2 Подключение к ним не всегда возможно



```
lldb -n Finder
igoor_bb@Igor's-MacBook-Pro ~
$ lldb -n Finder
(lldb) process attach --name "Finder"
error: attach failed: attach failed (Not allowed to attach to process. Look in the console messages (Console.app), near the debugger entries, when the attach failed. The subsystem that denied the attach permission will likely have logged an informative message about why it was denied.)
(lldb)
no target
```

OS не обязана нас пускать

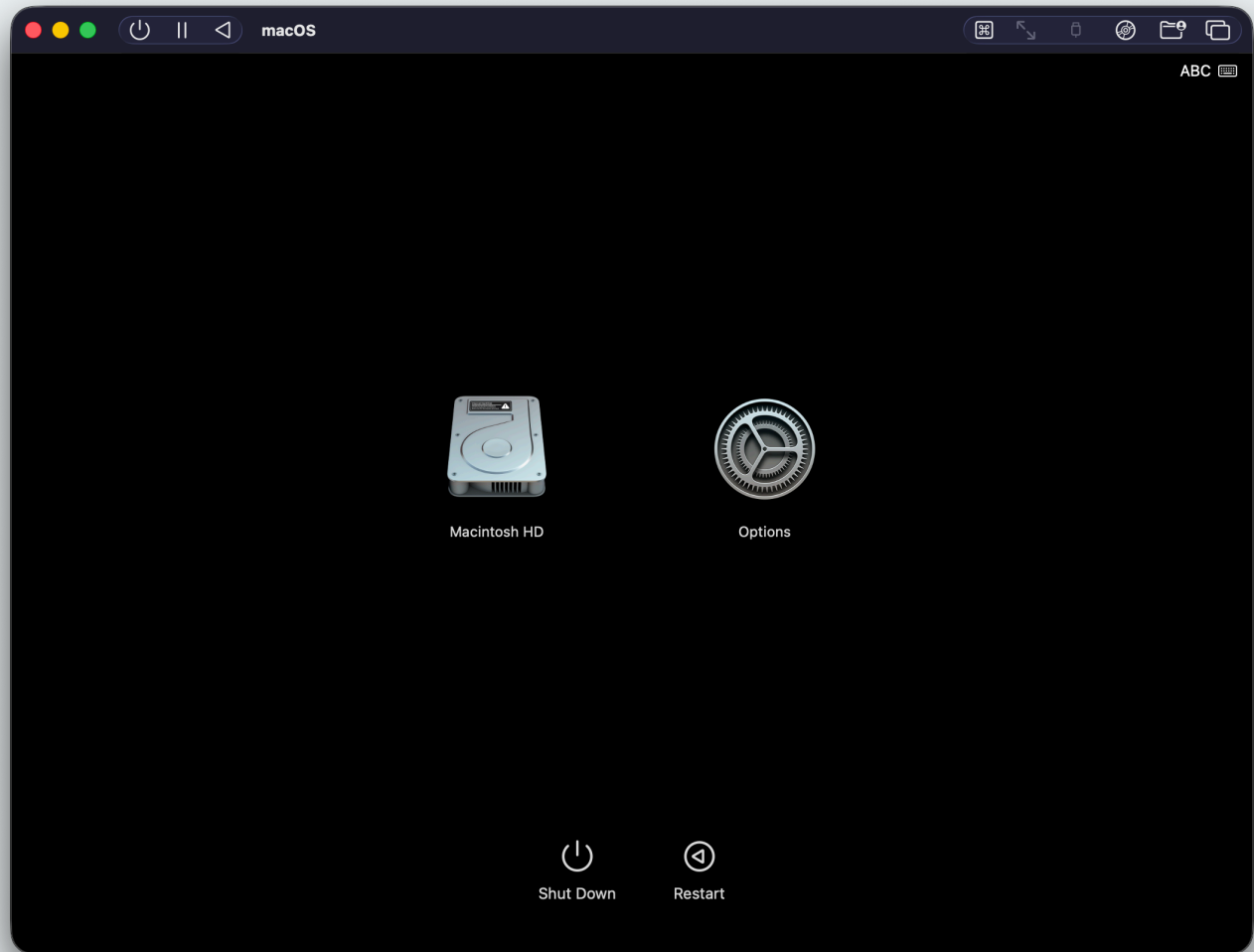
- 1 Системные процессы защищены
- 2 Подключение к ним не всегда возможно
- 3 И это совершенно нормально



```
igoor_bb@Igor's-MacBook-Pro ~  
$ lldb -n Finder  
(lldb) process attach --name "Finder"  
error: attach failed: attach failed (Not allowed to attach to process. Look in the console messages (Console.app), near the debugger entries, when the attach failed. The subsystem that denied the attach permission will likely have logged an informative message about why it was denied.)  
(lldb)  
  
no target
```

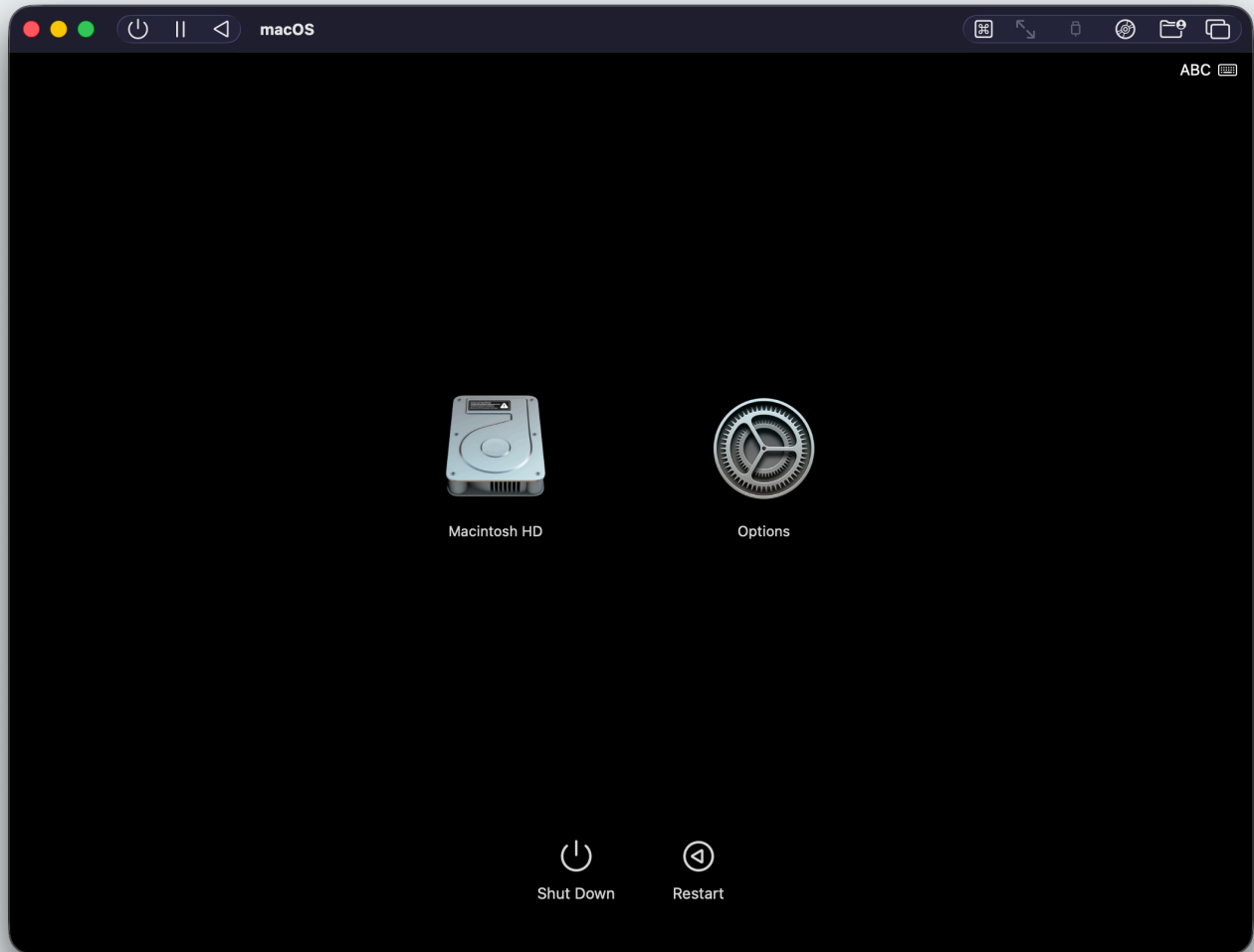
System Integrity Protection (SIP)

1 Механизм защиты системы



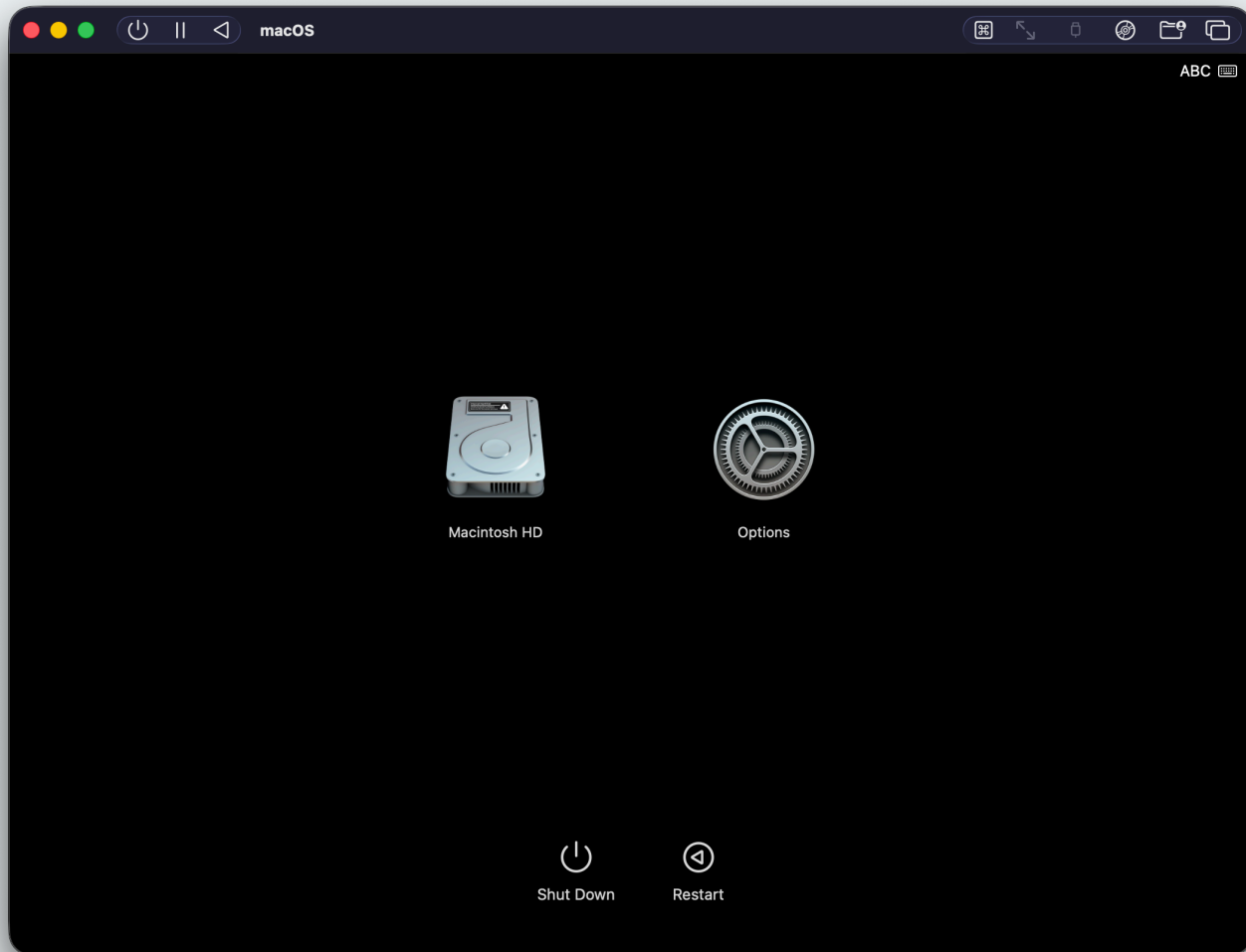
System Integrity Protection (SIP)

- 1 Механизм защиты системы
- 2 Ограничивает доступ к важным компонентам



System Integrity Protection (SIP)

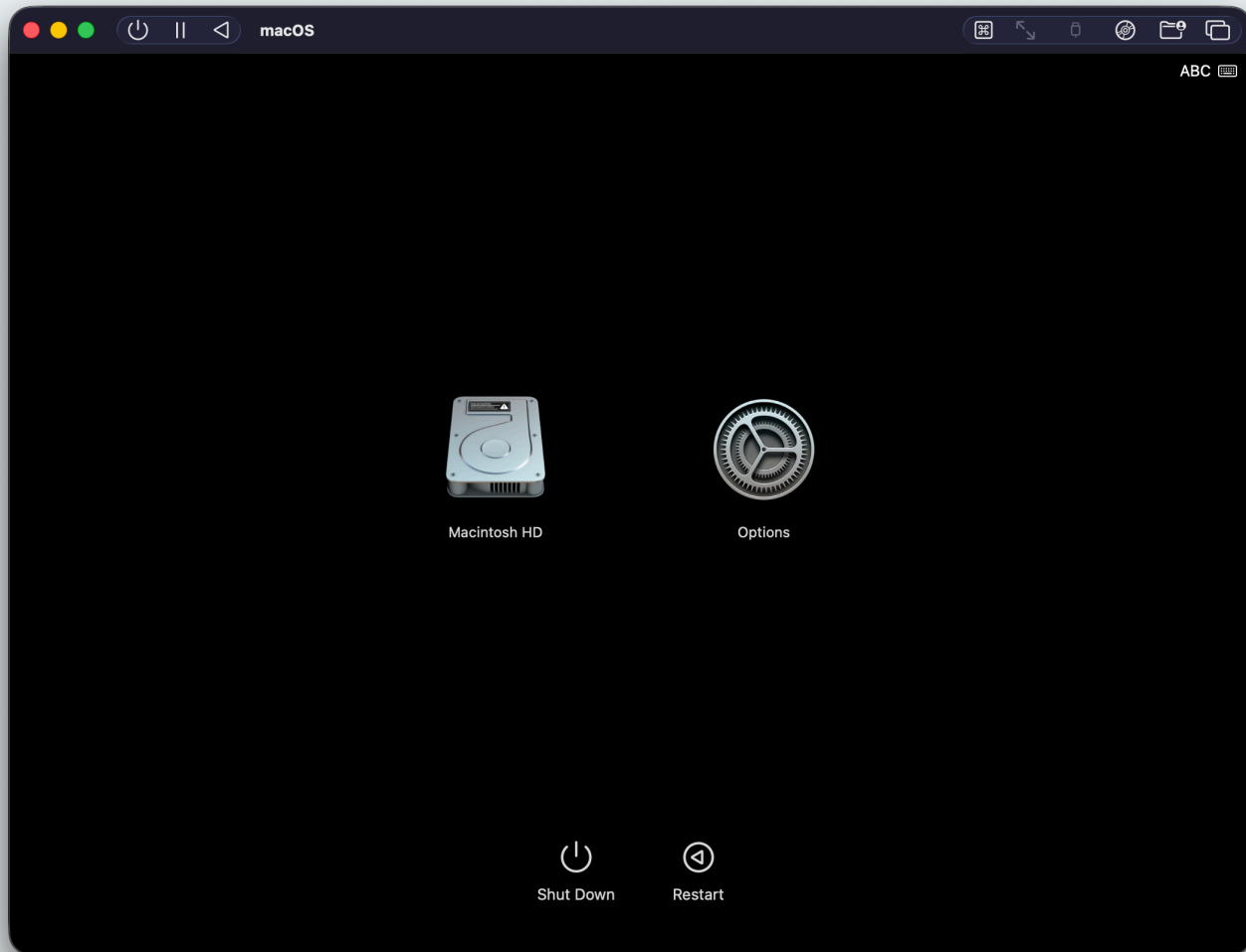
- 1 Механизм защиты системы
- 2 Ограничивает доступ к важным компонентам
- 3 Предотвращает отладку и инъекцию в чужие процессы



System Integrity Protection (SIP)

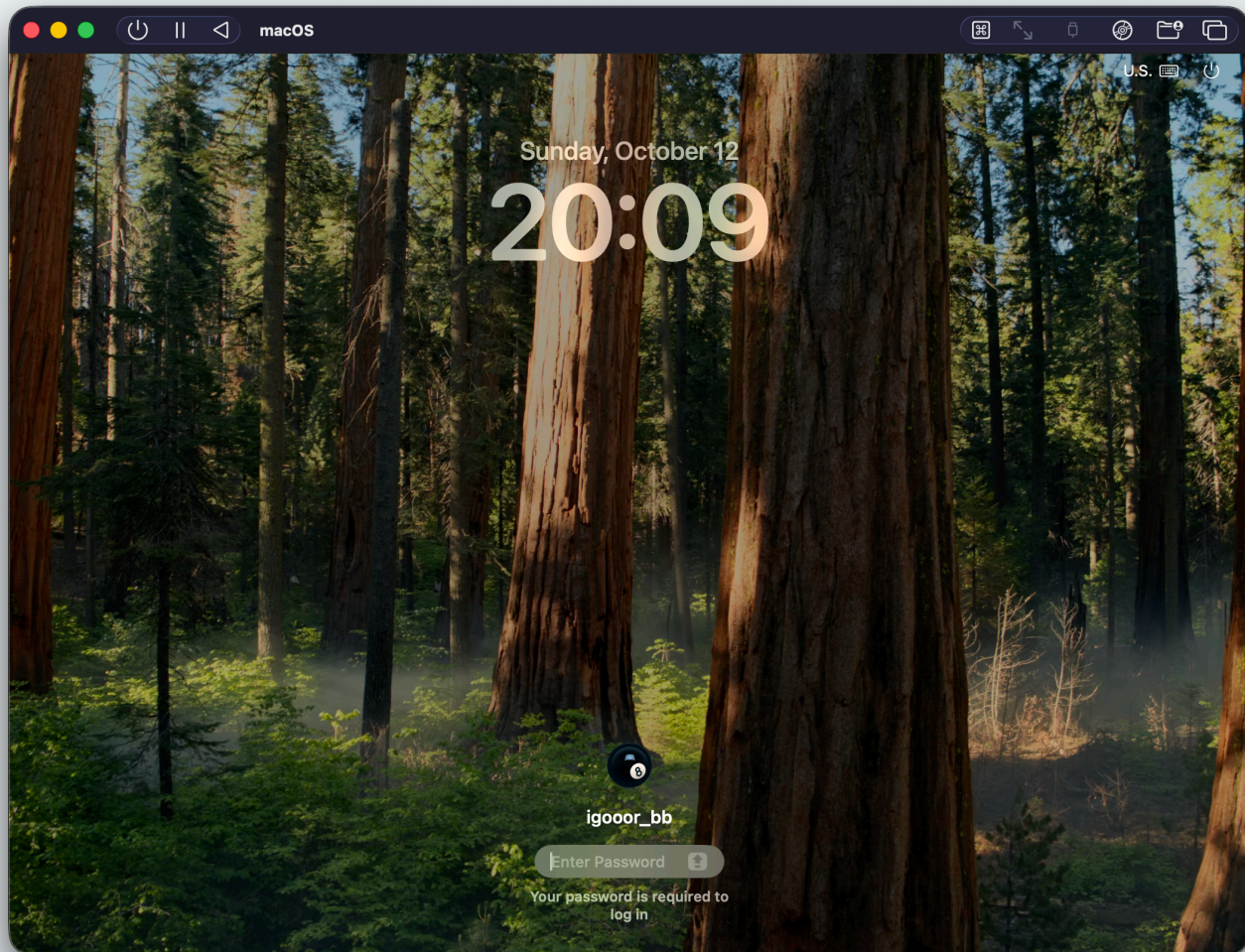
- 1 Механизм защиты системы
- 2 Ограничивает доступ к важным компонентам
- 3 Предотвращает отладку и инъекцию в чужие процессы

Отключается в Recovery Mode с помощью команды `csrutil disable`



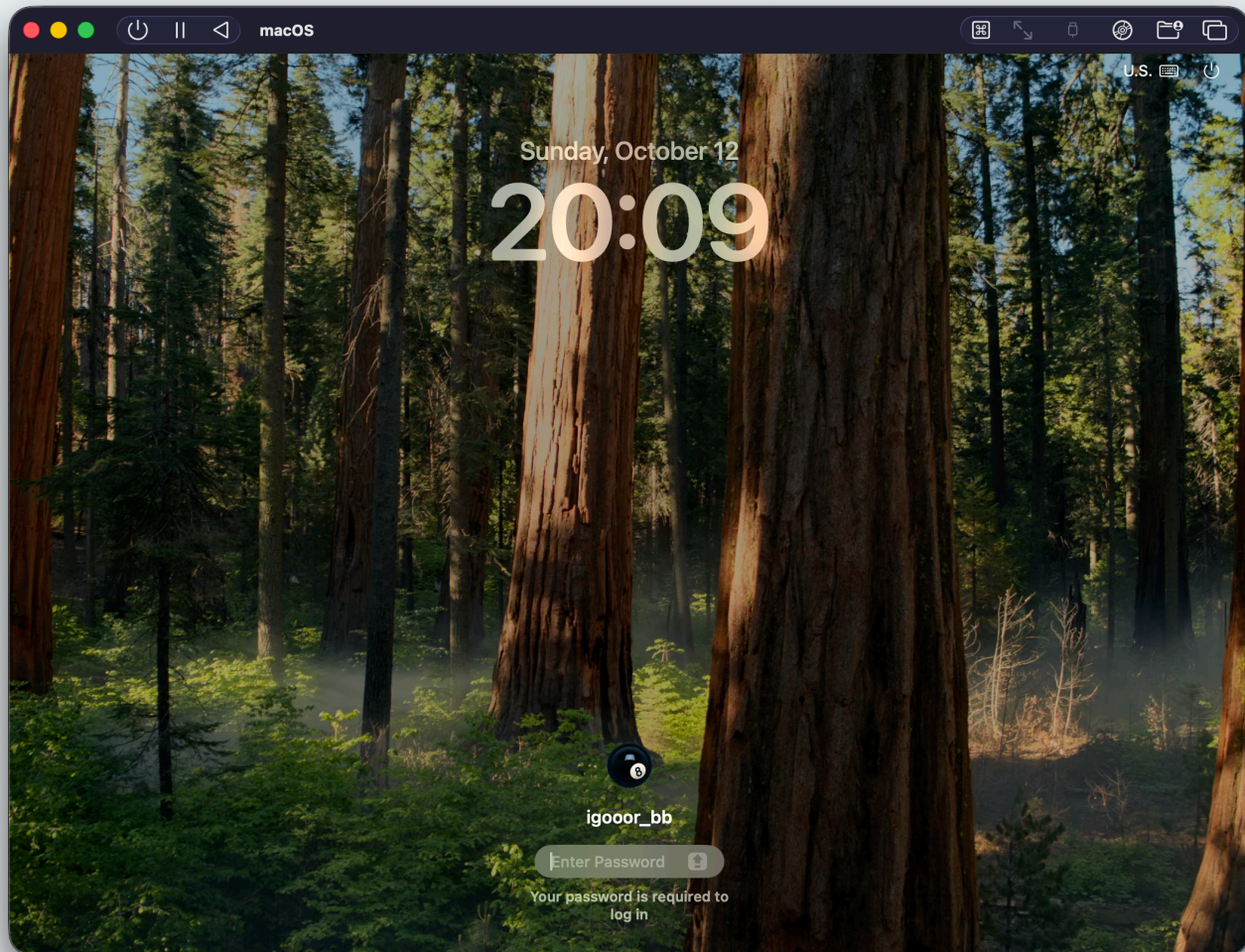
Виртуализация

- 1 Отдельная среда для исследований



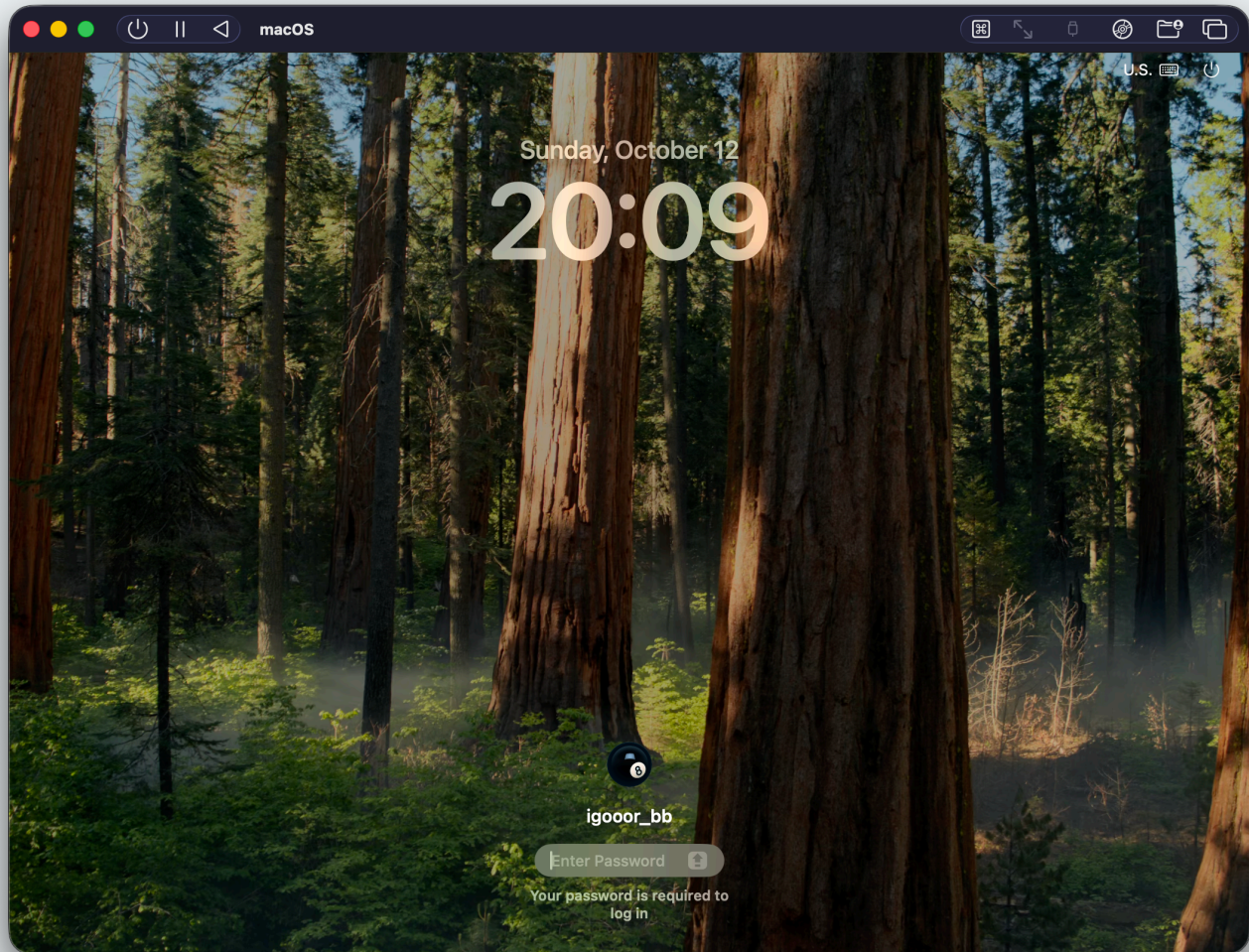
Виртуализация

- 1 Отдельная среда для исследований
- 2 Безопасные эксперименты



Виртуализация

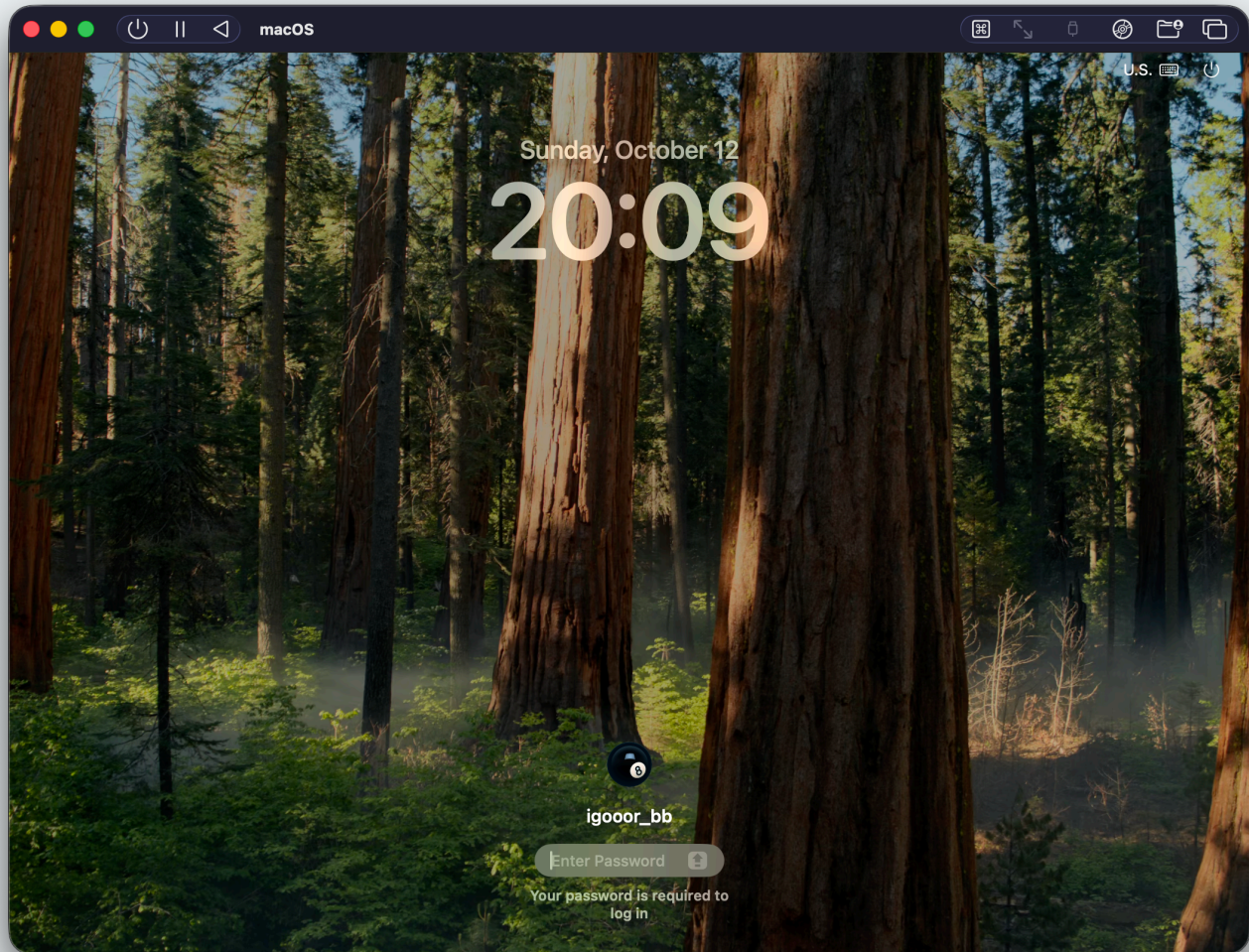
- 1 Отдельная среда для исследований
- 2 Безопасные эксперименты
- 3 Воспроизводимость окружения

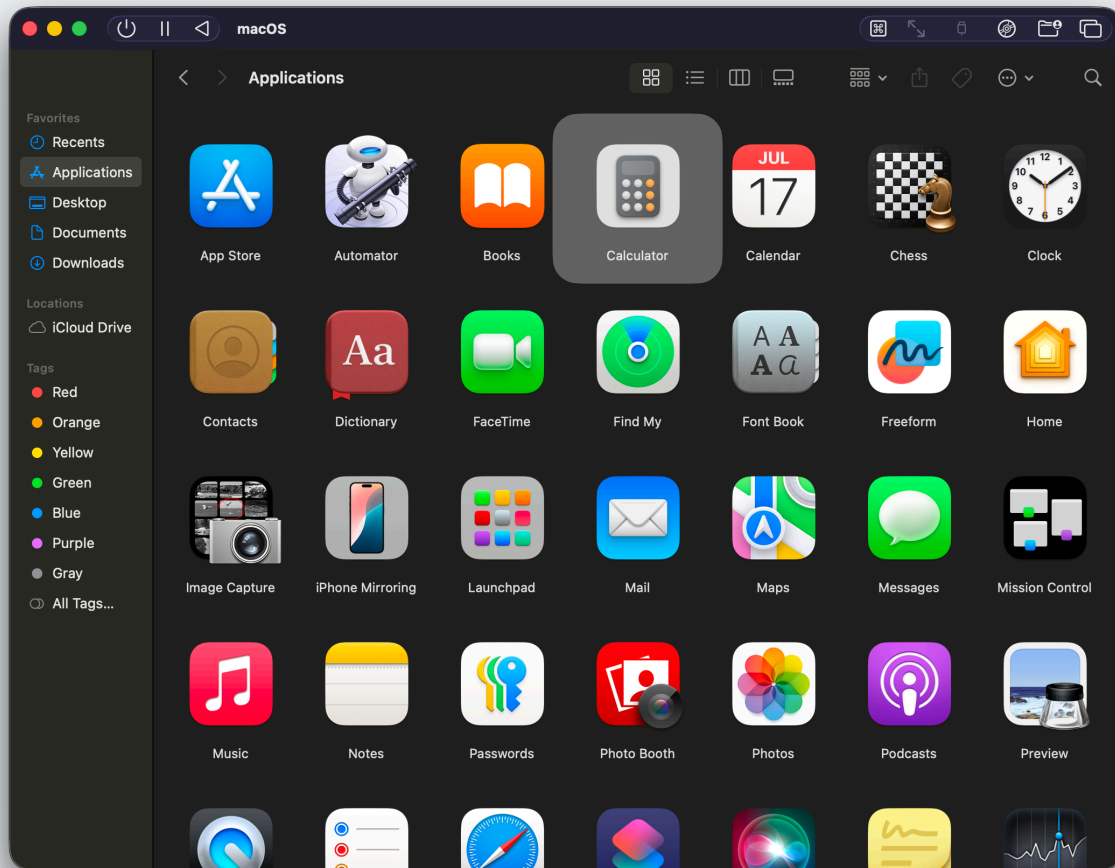


Виртуализация

- 1 Отдельная среда для исследований
- 2 Безопасные эксперименты
- 3 Воспроизводимость окружения

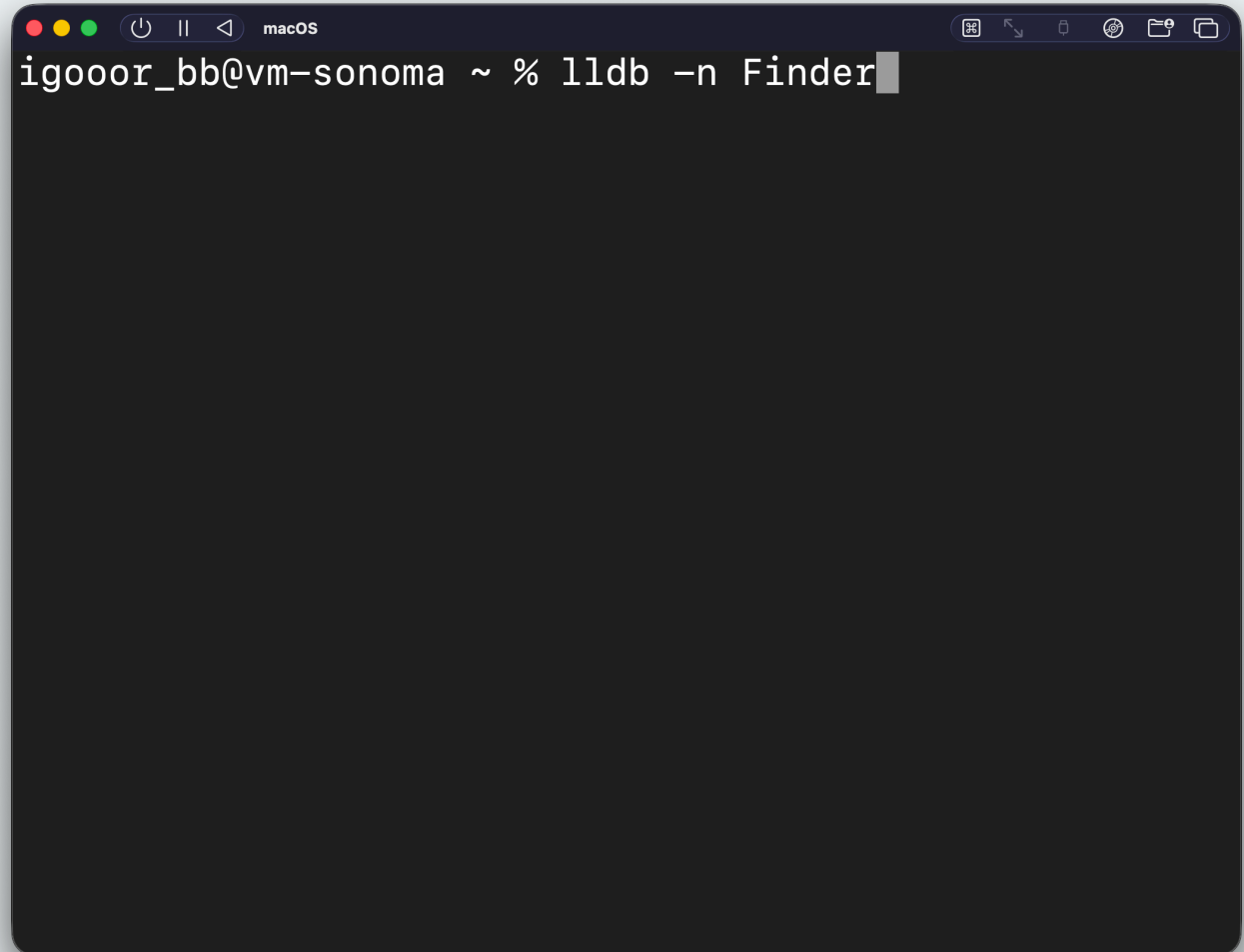
Далее все действия будут происходить внутри виртуальной машины





Исследование

- 1 К процессу можно подключиться по его **PID** или **имени**

A terminal window with a dark background and light text. The window title bar shows standard macOS window controls (red, yellow, green buttons, power, pause, back) and the text 'macOS'. The terminal prompt is 'igoorr_bb@vm-sonoma ~ %' followed by the command 'lldb -n Finder' and a cursor. The rest of the terminal is empty.

```
igoorr_bb@vm-sonoma ~ % lldb -n Finder
```

Исследование

- 1 К процессу можно подключиться по его **PID** или **имени**
- 2 При остановке отправляется **SIGSTOP**

```
igoor_bb@vm-sonoma ~ % lldb -n Finder
(lldb) process attach --name "Finder"
Process 449 stopped
* thread #1, queue = 'com.apple.main-thread', stop reason = signal SIGSTOP
    frame #0: 0x000000018b5f9c34 libsystem_kernel.dylib`mach_msg2_trap + 8
    libsystem_kernel.dylib`mach_msg2_trap:
-> 0x18b5f9c34 <+8>: ret

libsystem_kernel.dylib`macx_swapon:
    0x18b5f9c38 <+0>: mov     x16, #-0x30 ; =-48
    0x18b5f9c3c <+4>: svc     #0x80
    0x18b5f9c40 <+8>: ret

Target 0: (Finder) stopped.
Executable binary set to "/System/Library/CoreServices/Finder.app/Contents/MacOS/Finder".
Architecture set to: arm64e-apple-macosx-.
(lldb) █
```

Исследование

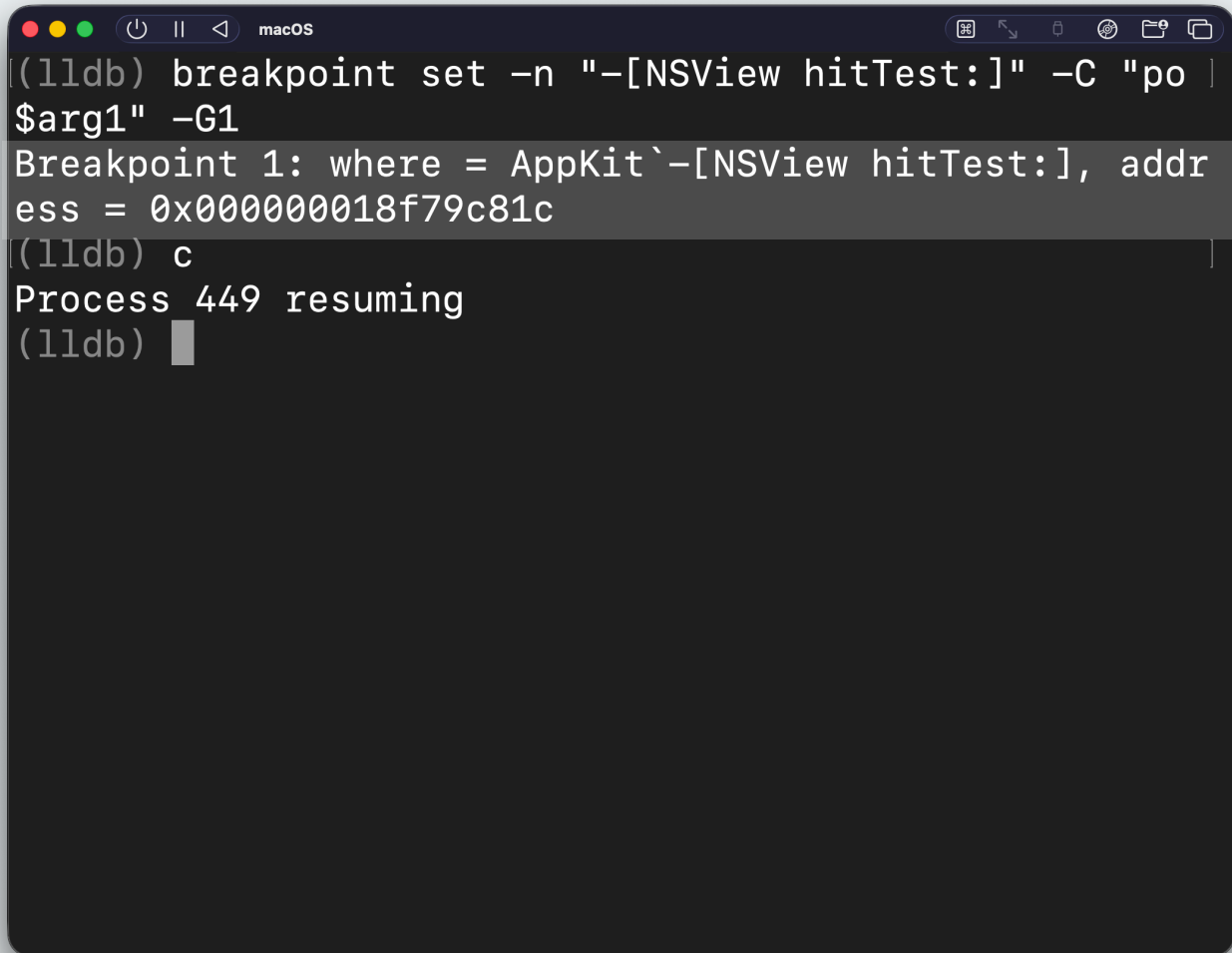
- 1 К процессу можно подключиться по его **PID** или **имени**
- 2 При остановке отправляется **SIGSTOP**

```
igoor_bb@vm-sonoma ~ % lldb -n Finder
(lldb) process attach --name "Finder"
Process 449 stopped
* thread #1, queue = 'com.apple.main-thread', stop reason = signal SIGSTOP
    frame #0: 0x000000018b5f9c34 libsystem_kernel.dylib`mach_msg2_trap + 8
libsystem_kernel.dylib`mach_msg2_trap:
-> 0x18b5f9c34 <+8>: ret

libsystem_kernel.dylib`macx_swapon:
    0x18b5f9c38 <+0>: mov     x16, #-0x30 ; =-48
    0x18b5f9c3c <+4>: svc     #0x80
    0x18b5f9c40 <+8>: ret
Target 0: (Finder) stopped.
Executable binary set to "/System/Library/CoreServices/Finder.app/Contents/MacOS/Finder".
Architecture set to: arm64e-apple-macosx-.
(lldb) █
```

Исследование

- 1 Можем делать точки остановки
- 2 Можем использовать **hitTest**, чтобы найти нужный объект



```
(lldb) breakpoint set -n "-[UIView hitTest:]" -C "po |
$args1" -G1
Breakpoint 1: where = AppKit`-[UIView hitTest:], addr
ess = 0x000000018f79c81c
(lldb) c
Process 449 resuming
(lldb) █
```

```
breakpoint set -n "-[UIView hitTest:]" -C "po $arg1" -G1
```

Создаем
breakpoint



```
breakpoint set -n "-[UIView hitTest:]" -C "po $arg1" -G1
```



Указываем по имени
Objective-C селектор



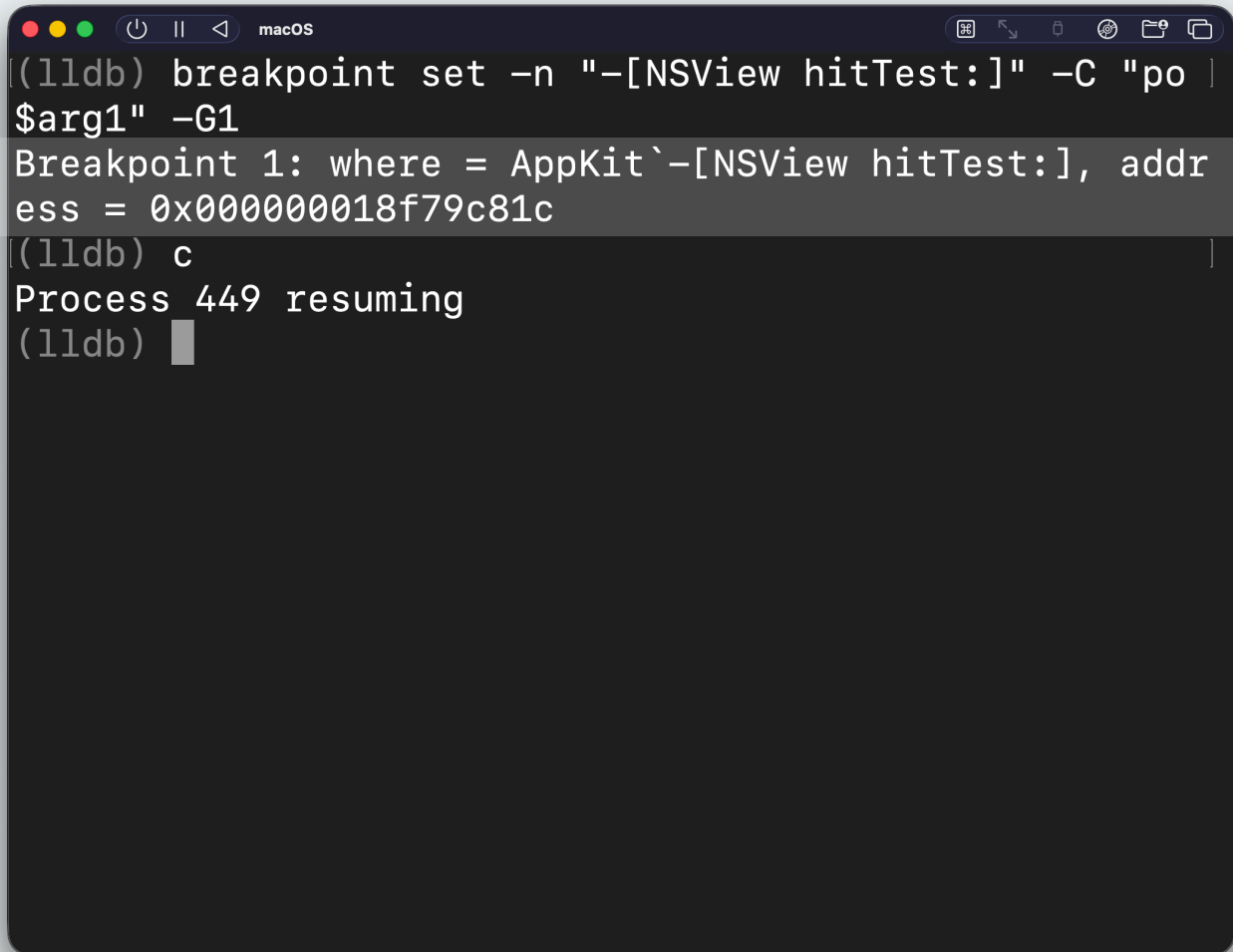
Указываем команду, которая
выполнится при каждом
срабатывании

Продолжаем без
остановки



Исследование

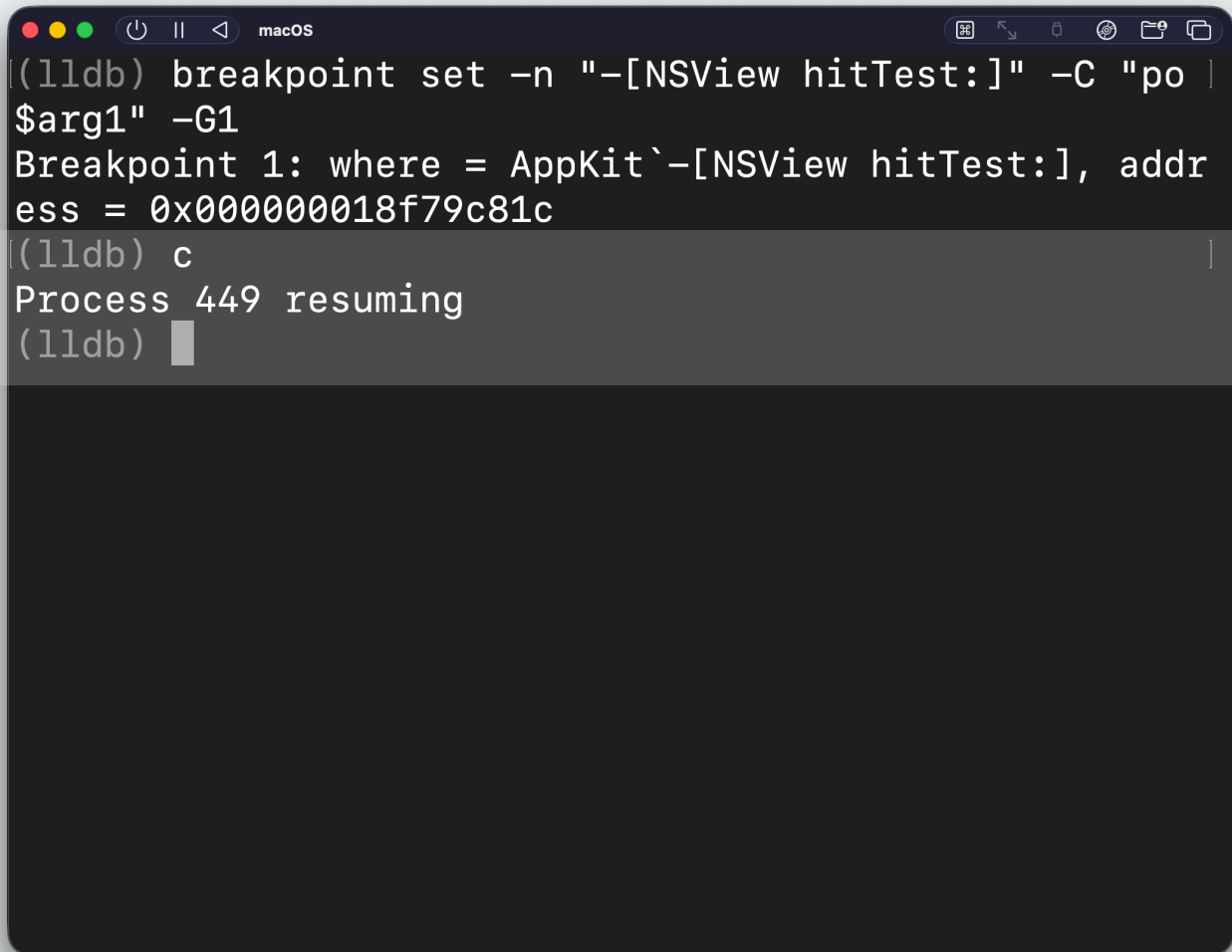
- 1 Можем делать точки остановки
- 2 Можем использовать **hitTest**, чтобы найти нужный объект



```
(lldb) breakpoint set -n "-[UIView hitTest:]" -C "po |
$args1" -G1
Breakpoint 1: where = AppKit`-[UIView hitTest:], addr
ess = 0x000000018f79c81c
(lldb) c
Process 449 resuming
(lldb) █
```

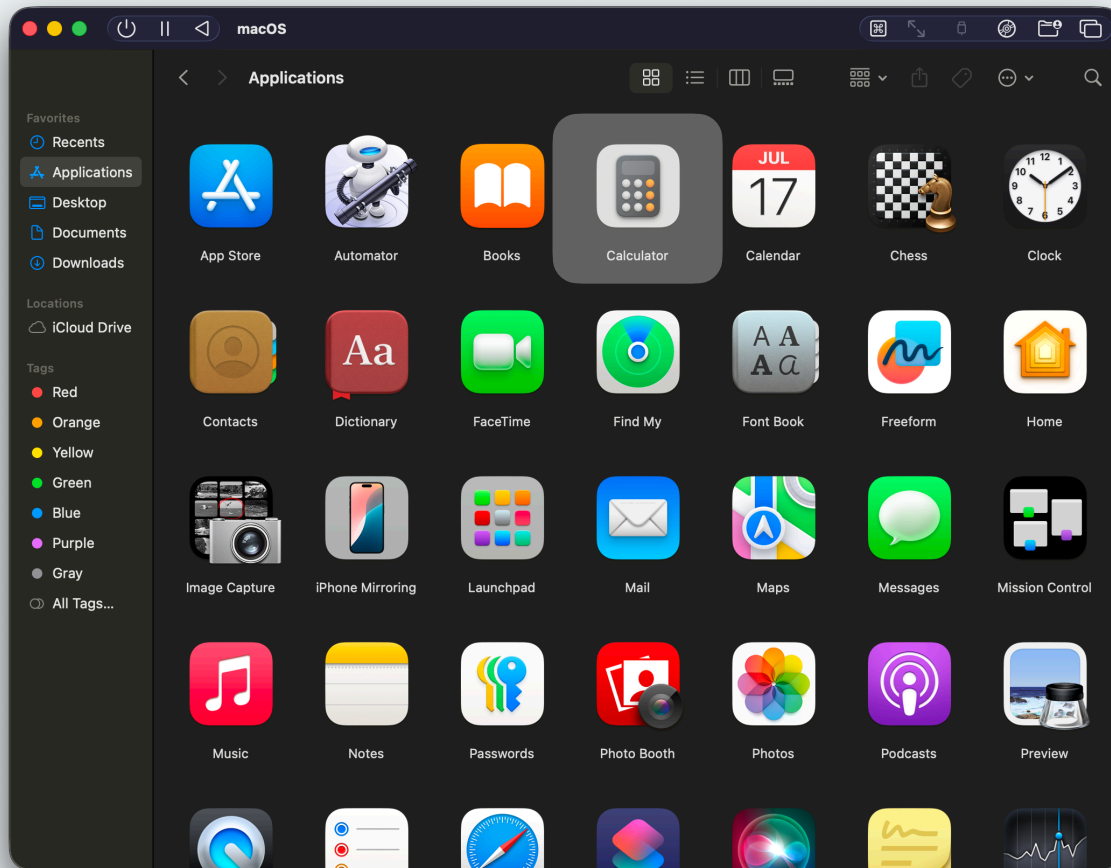
Исследование

- 1 Можем делать точки остановки
- 2 Можем использовать **hitTest**, чтобы найти нужный объект



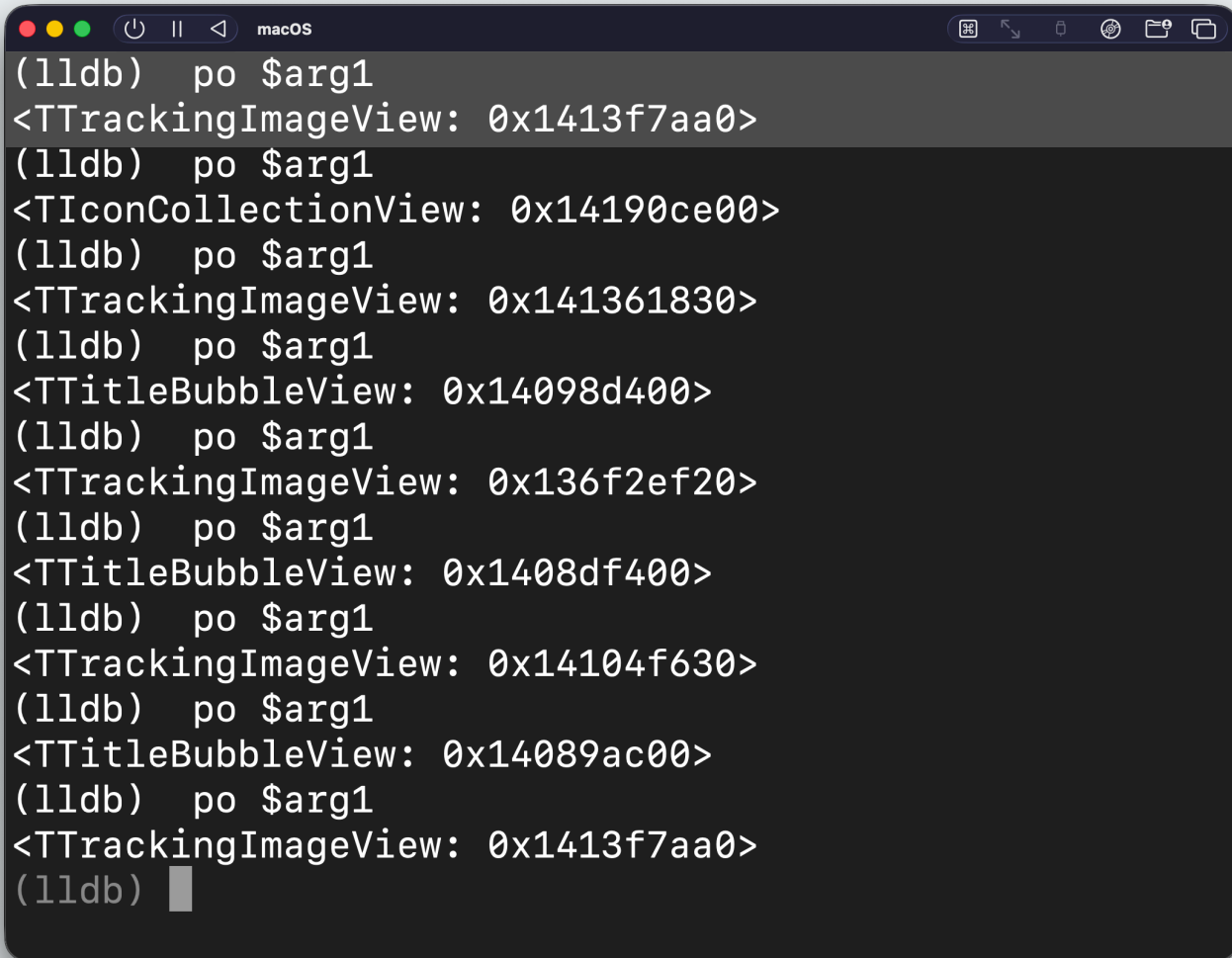
```
(lldb) breakpoint set -n "-[UIView hitTest:]" -C "po $arg1" -G1
Breakpoint 1: where = AppKit`-[UIView hitTest:], address = 0x000000018f79c81c

(lldb) c
Process 449 resuming
(lldb) █
```



Исследование

- 1 Обход выполняется по принципу **Responder Chain**



```
(lldb) po $arg1
<TTTrackingImageView: 0x1413f7aa0>
(lldb) po $arg1
<TIconCollectionView: 0x14190ce00>
(lldb) po $arg1
<TTTrackingImageView: 0x141361830>
(lldb) po $arg1
<TTitleBubbleView: 0x14098d400>
(lldb) po $arg1
<TTTrackingImageView: 0x136f2ef20>
(lldb) po $arg1
<TTitleBubbleView: 0x1408df400>
(lldb) po $arg1
<TTTrackingImageView: 0x14104f630>
(lldb) po $arg1
<TTitleBubbleView: 0x14089ac00>
(lldb) po $arg1
<TTTrackingImageView: 0x1413f7aa0>
(lldb) █
```

Исследование

- 1 Обход выполняется по принципу **Responder Chain**

```
macOS
(lldb) po $arg1
<TTTrackingImageView: 0x1413f7aa0>
(lldb) po $arg1
<TIconCollectionView: 0x14190ce00>
(lldb) po $arg1
<TTTrackingImageView: 0x141361830>
(lldb) po $arg1
<TTitleBubbleView: 0x14098d400>
(lldb) po $arg1
<TTTrackingImageView: 0x136f2ef20>
(lldb) po $arg1
<TTitleBubbleView: 0x1408df400>
(lldb) po $arg1
<TTTrackingImageView: 0x14104f630>
(lldb) po $arg1
<TTitleBubbleView: 0x14089ac00>
(lldb) po $arg1
<TTTrackingImageView: 0x1413f7aa0>
(lldb) █
```

Исследование

- 1 Обход выполняется по принципу **Responder Chain**
- 2 Нужным объектом окажется **последний в списке**

```
macOS
(lldb) po $arg1
<TTrackingImageView: 0x1413f7aa0>
(lldb) po $arg1
<TIconCollectionView: 0x14190ce00>
(lldb) po $arg1
<TTrackingImageView: 0x141361830>
(lldb) po $arg1
<TTitleBubbleView: 0x14098d400>
(lldb) po $arg1
<TTrackingImageView: 0x136f2ef20>
(lldb) po $arg1
<TTitleBubbleView: 0x1408df400>
(lldb) po $arg1
<TTrackingImageView: 0x14104f630>
(lldb) po $arg1
<TTitleBubbleView: 0x14089ac00>
(lldb) po $arg1
<TTrackingImageView: 0x1413f7aa0>
(lldb)
```

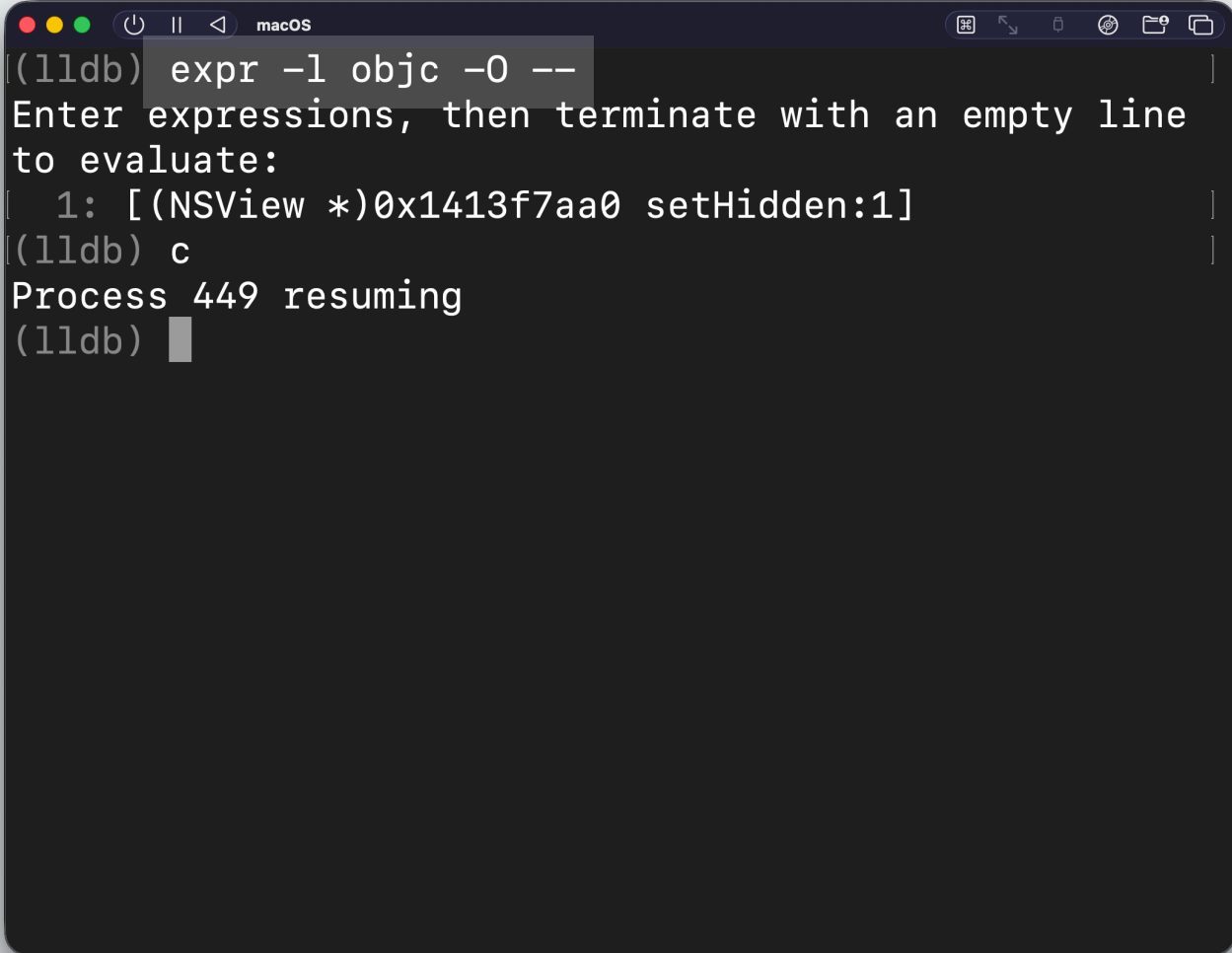
Исследование

- 1 Обход выполняется по принципу **Responder Chain**
- 2 Нужным объектом окажется **последний в списке**

```
macOS
(lldb) po $arg1
<TTrackingImageView: 0x1413f7aa0>
(lldb) po $arg1
<TIconCollectionView: 0x14190ce00>
(lldb) po $arg1
<TTrackingImageView: 0x141361830>
(lldb) po $arg1
<TTitleBubbleView: 0x14098d400>
(lldb) po $arg1
<TTrackingImageView: 0x136f2ef20>
(lldb) po $arg1
<TTitleBubbleView: 0x1408df400>
(lldb) po $arg1
<TTrackingImageView: 0x14104f630>
(lldb) po $arg1
<TTitleBubbleView: 0x14089ac00>
(lldb) po $arg1
<TTrackingImageView: 0x1413f7aa0>
(lldb) █
```

Выражения

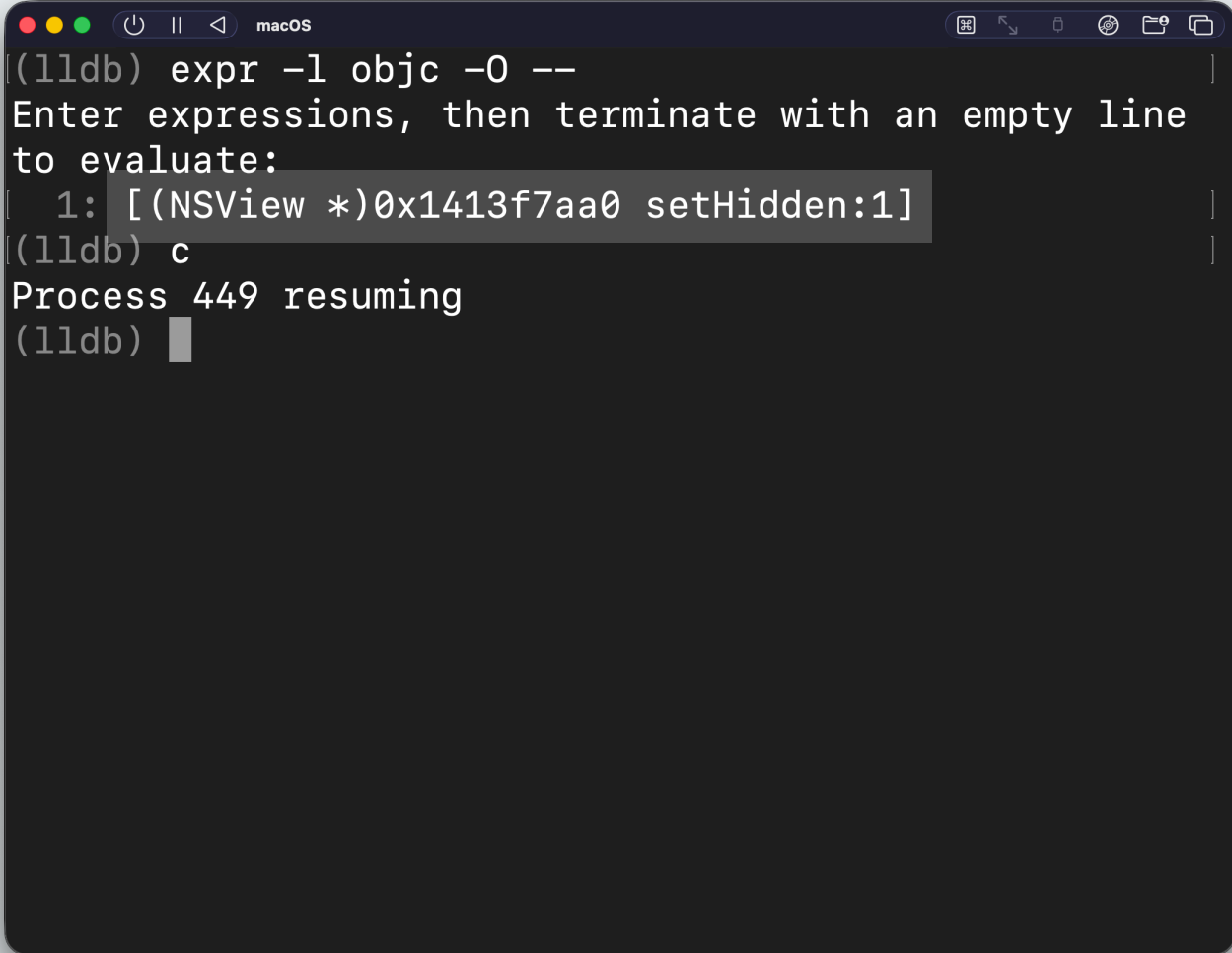
- 1 Имеем **адрес** объекта, который нас интересует



```
(lldb) expr -l objc -O --
Enter expressions, then terminate with an empty line
to evaluate:
  1: [(NSView *)0x1413f7aa0 setHidden:1]
(lldb) c
Process 449 resuming
(lldb) █
```

Выражения

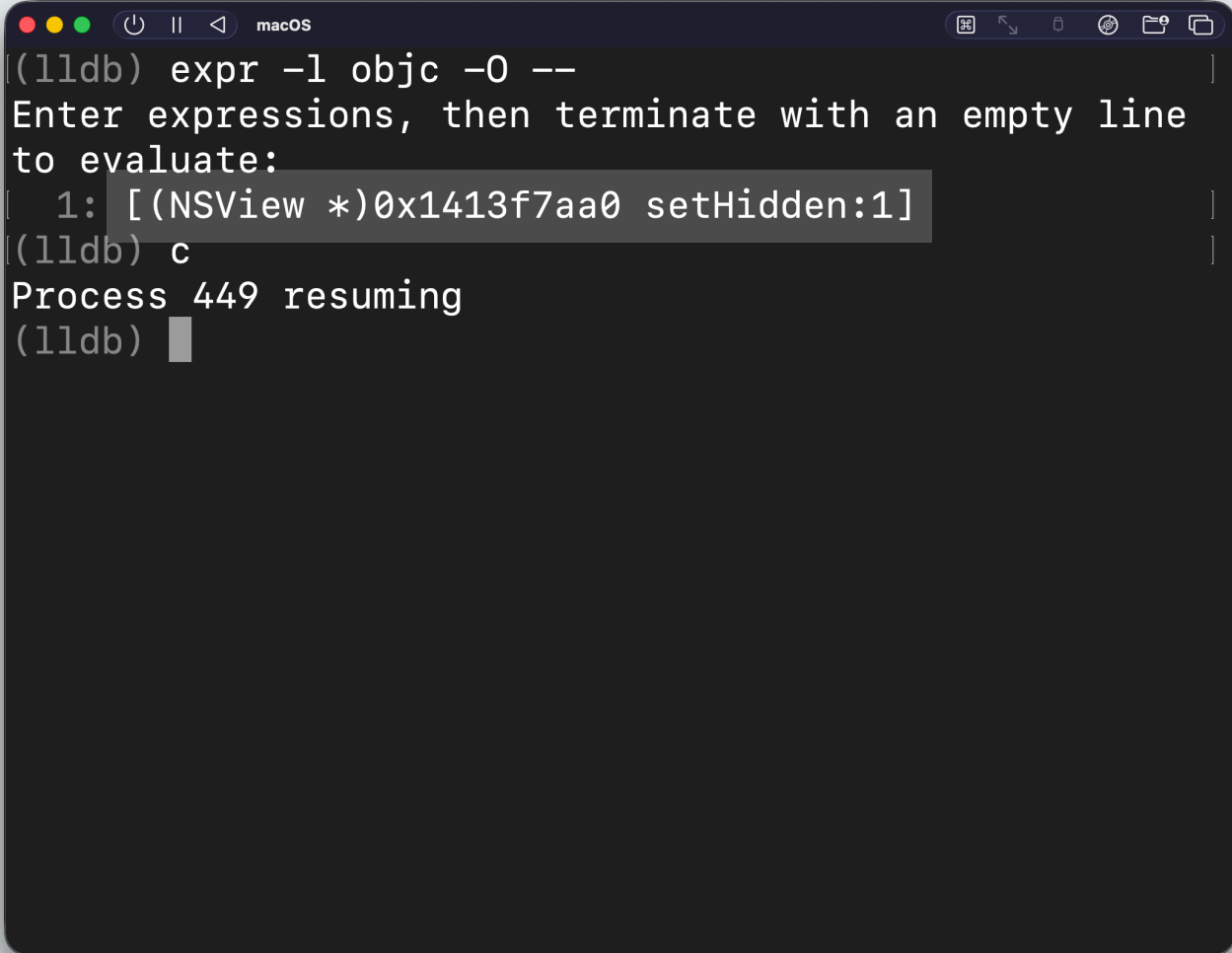
- 1 Имеем **адрес** объекта, который нас интересует
- 2 Можем использовать его в выражениях



```
(lldb) expr -l objc -O --
Enter expressions, then terminate with an empty line
to evaluate:
  1: [(NSView *)0x1413f7aa0 setHidden:1]
(lldb) c
Process 449 resuming
(lldb) █
```

Выражения

- 1 Имеем **адрес** объекта, который нас интересует
- 2 Можем использовать его в выражениях
- 3 Выражения можно писать на Swift и **Objective-C**



```
(lldb) expr -l objc -0 --
Enter expressions, then terminate with an empty line
to evaluate:
  1: [(NSView *)0x1413f7aa0 setHidden:1]
(lldb) c
Process 449 resuming
(lldb)
```

```
expr -l objc -0 -- [(NSView *)0x1413f7aa0 setHidden:1]
```

Выполняем Objective-C
выражение



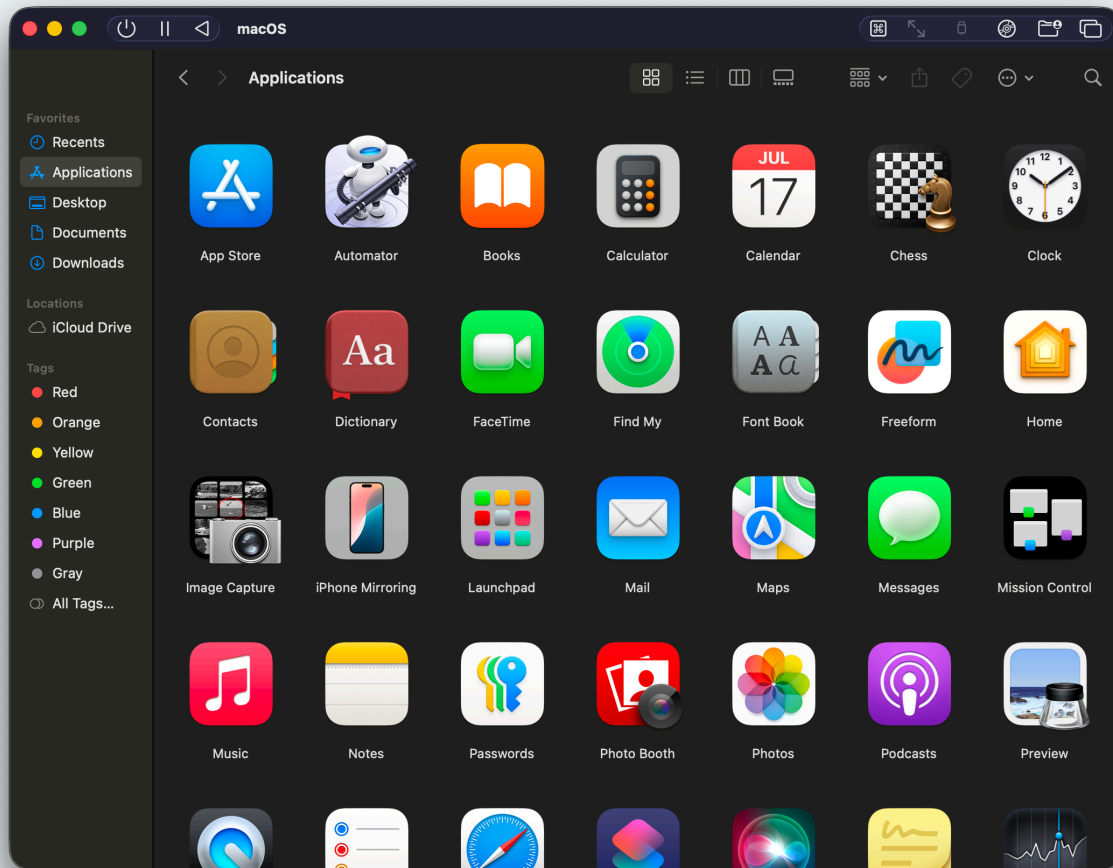
```
expr -l objc -O -- [(NSView *)0x1413f7aa0 setHidden:1]
```

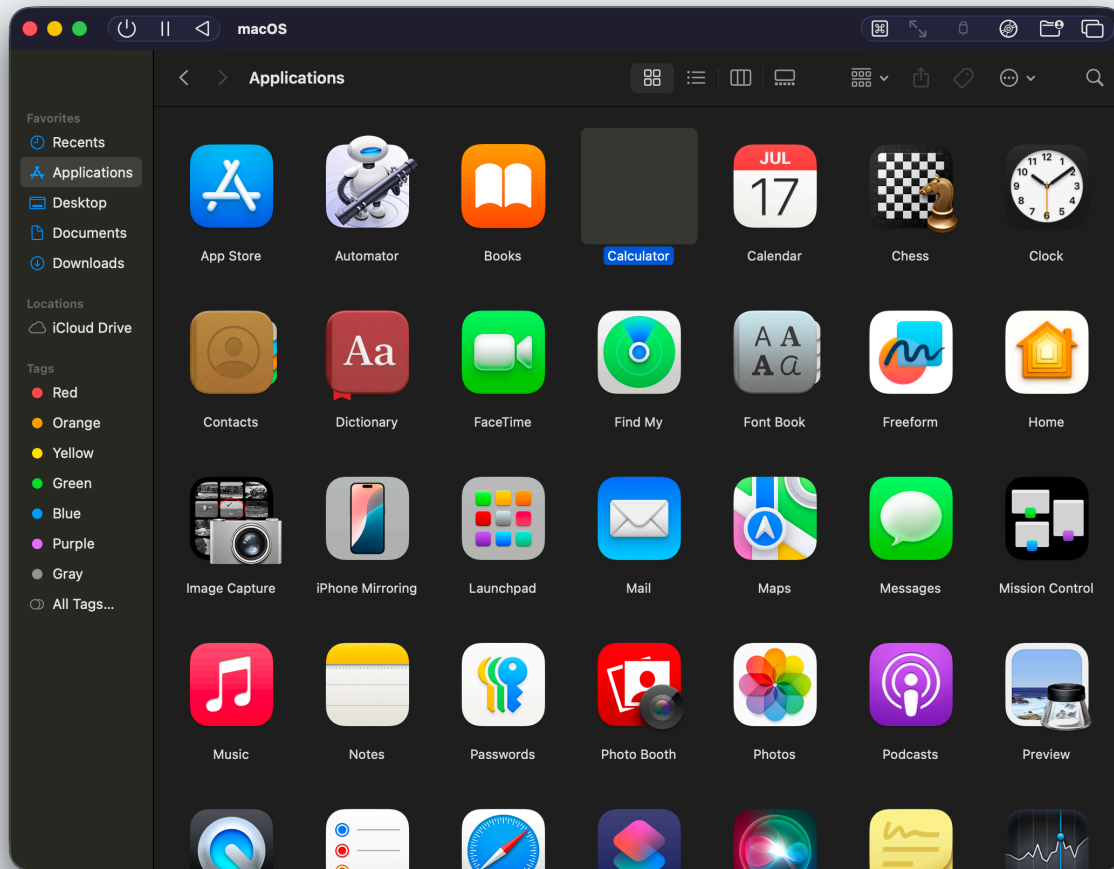


Печатаем описание
объекта на выходе
(удобство)

Обращаемся к объекту как к
NSView
и меняем его свойство

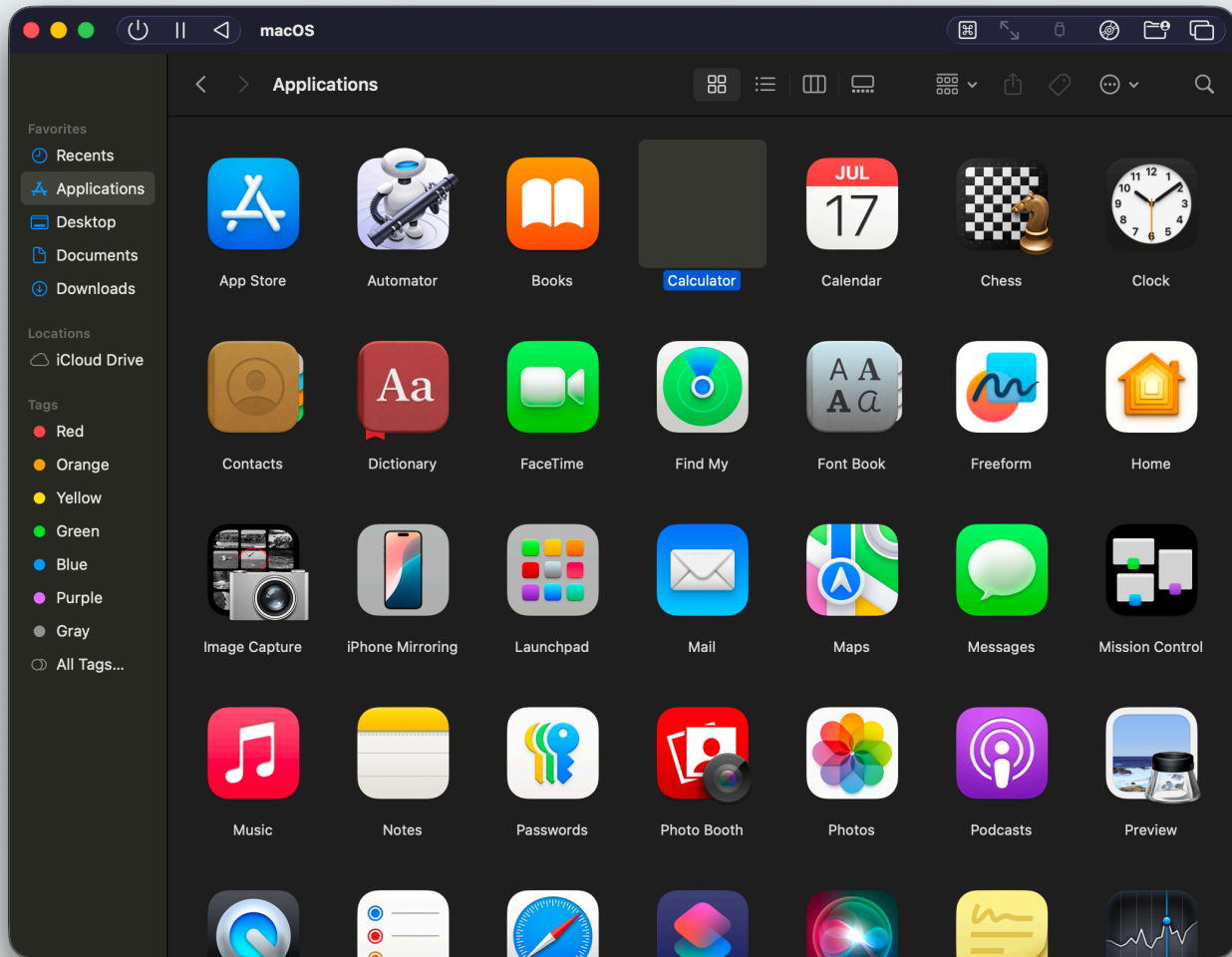






Результат

- 1 Иногда необходимо дождаться следующей итерации **Render Loop**
- 2 Можно форсировать с помощью селектора **[NSView display]**



Выражения

- 1 Можем писать более сложные выражения

```
@import AppKit;

NSView *view = (NSView *)0x1413f7aa0;
[view setWantsLayer:1];

CALayer *layer = (CALayer *)[view layer];
[layer setBorderWidth:2.0];
[layer setBorderColor:[NSColor redColor] CGColor];

[view display];
```

Выражения

- 1 Можем писать более сложные выражения
- 2 Иногда нужно добавлять импорты для стандартных типов

```
@import AppKit;
```

```
NSView *view = (NSView *)0x1413f7aa0;  
[view setWantsLayer:1];
```

```
CALayer *layer = (CALayer *)[view layer];  
[layer setBorderWidth:2.0];  
[layer setBorderColor:[[NSColor redColor] CGColor]];
```

```
[view display];
```

Выражения

- 1 Можем писать более сложные выражения
- 2 Иногда нужно добавлять импорты для стандартных типов

```
@import AppKit;
```

```
NSView *view = (NSView *)0x1413f7aa0;  
[view setWantsLayer:1];
```

```
CALayer *layer = (CALayer *)[view layer];  
[layer setBorderWidth:2.0];  
[layer setBorderColor:[NSColor redColor] CGColor];
```

```
[view display];
```

Выражения

- 1 Можем писать более сложные выражения
- 2 Иногда нужно добавлять импорты для стандартных типов

```
@import AppKit;
```

```
NSView *view = (NSView *)0x1413f7aa0;  
[view setWantsLayer:1];
```

```
CALayer *layer = (CALayer *)[view layer];  
[layer setBorderWidth:2.0];  
[layer setBorderColor:[[NSColor redColor] CGColor]];
```

```
[view display];
```

Выражения

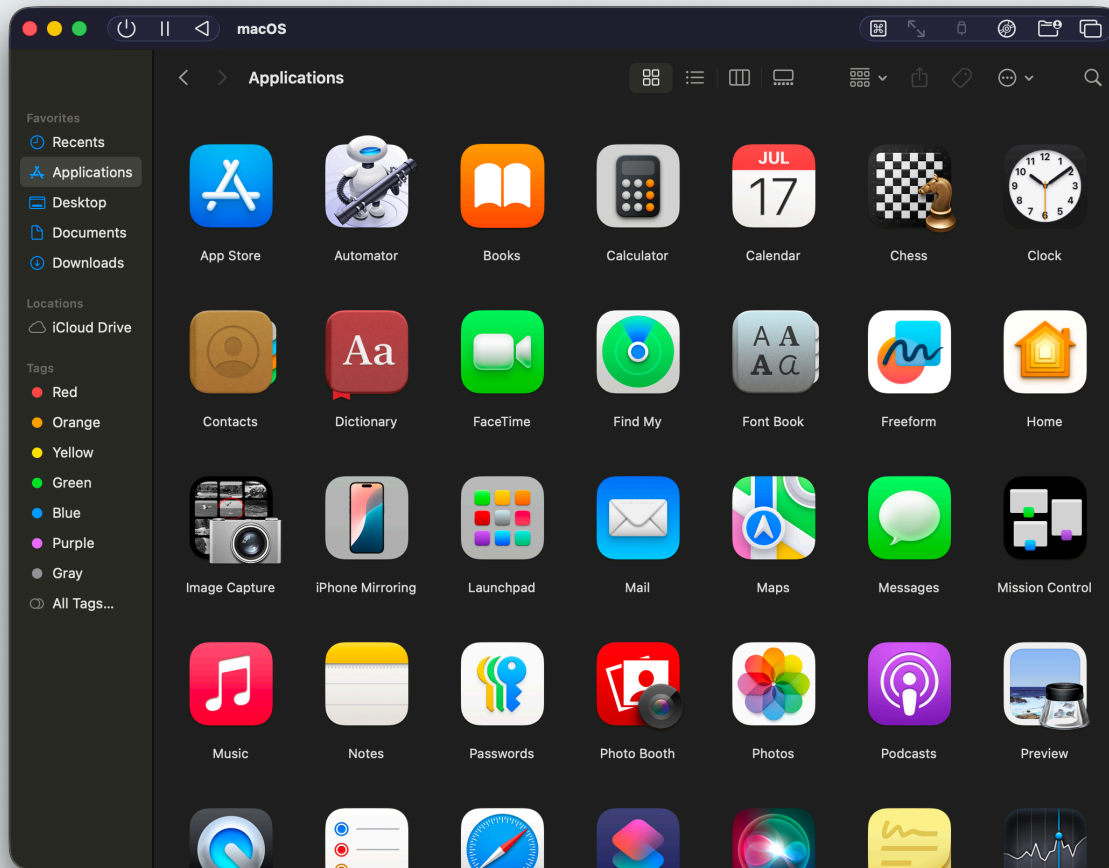
- 1 Можем писать более сложные выражения
- 2 Иногда нужно добавлять импорты для стандартных типов

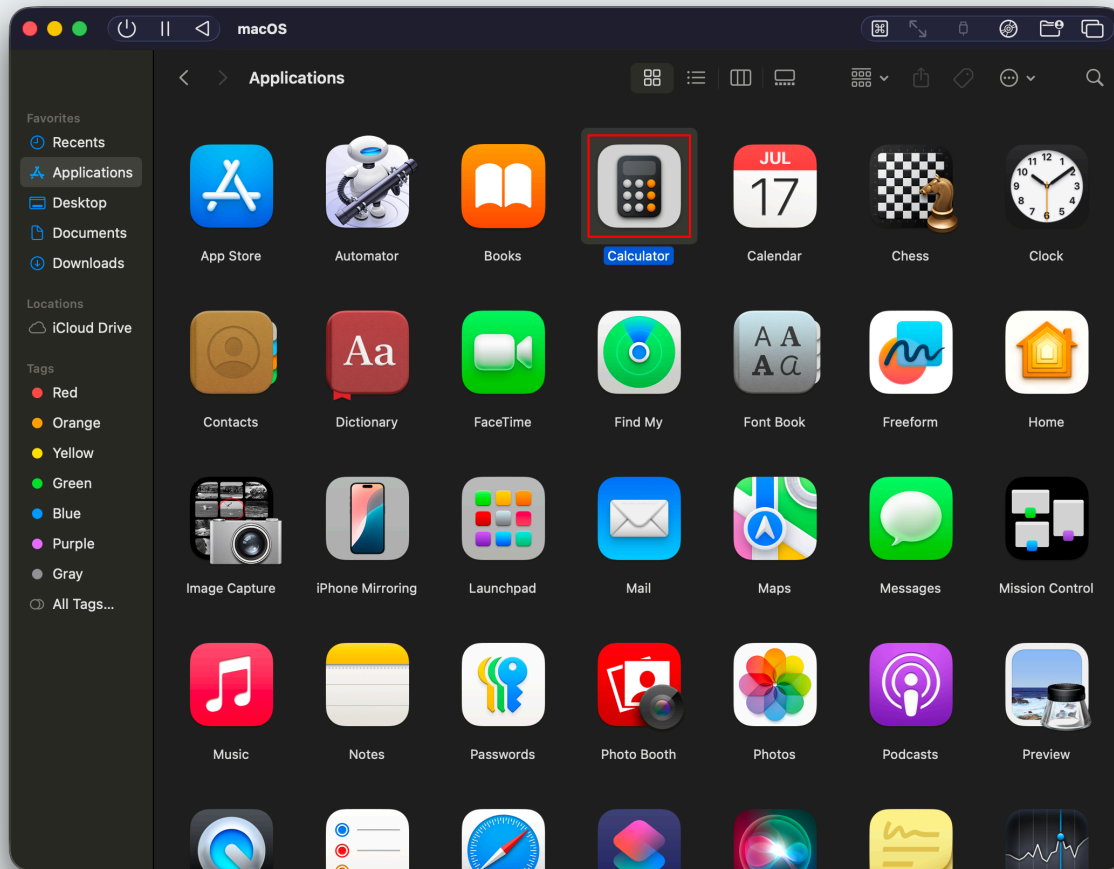
```
@import AppKit;

NSView *view = (NSView *)0x1413f7aa0;
[view setWantsLayer:1];

CALayer *layer = (CALayer *)[view layer];
[layer setBorderWidth:2.0];
[layer setBorderColor:[[NSColor redColor] CGColor]];

[view display];
```





```
@import AppKit
```

```
NSView *view = (NSView *)0x115064bb0;  
[view setWantsLayer:1];
```

```
CALayer *layer = (CALayer *)[view layer];  
BOOL isRemoved = false;
```

```
for (s in [layer subviews]) {  
    if ([[NSString*][s name] isEqualToString:@"lldb_highlight"]) {  
        [s removeFromSuperlayer];  
        isRemoved = true;  
        [view display];  
        break;  
    }  
}
```

```
if (!isRemoved) {  
    id highlight = [CALayer layer];  
    [highlight setName:@"lldb_highlight"];
```



```
@import AppKit

NSView *view = (NSView *)0x115064bb0;
[view setWantsLayer:1];

CALayer *layer = (CALayer *)[view layer];
BOOL isRemoved = false;

for (s in [layer subviews]) {
    if ([[NSString*s] name] isEqualToString:@"lldb_highlight"]) {
        [s removeFromSuperlayer];
        isRemoved = true;
        [view display];
        break;
    }
}

if (!isRemoved) {
    id highlight = [CALayer layer];
    [highlight setName:@"lldb_highlight"];
}
```

```
BOOL isRemoved = false;
```

```
for (s in [layer subviews]) {  
    if ([[NSString]*][s name] isEqualToString:@"lldb_highlight"]) {  
        [s removeFromSuperlayer];  
        isRemoved = true;  
        [view display];  
        break;  
    }  
}
```

```
if (!isRemoved) {  
    id highlight = [CALayer layer];  
    [highlight setName:@"lldb_highlight"];  
    [highlight setBorderWidth:2.0];  
    [highlight setBorderColor:[NSColor redColor] CGColor];  
    [highlight setMasksToBounds:0];  
    [highlight setZPosition:9999.0];  
  
    [highlight setValue:[view valueForKey:@"bounds"] forKey:@"frame"];  
    [layer addSublayer:highlight];  
    [view display];  
}
```

```
BOOL isRemoved = false;
```

```
for (s in [layer subviews]) {  
    if ([[NSString]*][s name] isEqualToString:@"lldb_highlight"]) {  
        [s removeFromSuperlayer];  
        isRemoved = true;  
        [view display];  
        break;  
    }  
}
```

```
if (!isRemoved) {  
    id highlight = [CALayer layer];  
    [highlight setName:@"lldb_highlight"];  
    [highlight setBorderWidth:2.0];  
    [highlight setBorderColor:[NSColor redColor] CGColor];  
    [highlight setMasksToBounds:0];  
    [highlight setZPosition:9999.0];  
  
    [highlight setValue:[view valueForKey:@"bounds"] forKey:@"frame"];  
    [layer addSublayer:highlight];  
    [view display];  
}
```

Кастомизация

- 1 Можем создать алиас для команды
- 2 Это просто **псевдоним** для последовательности текста
- 3 Можем передать параметр только в конец

Настройки объявляются в файле ~/.lldbinit

```
# ~/.lldbinit  
  
command alias pv expr -l objc -0 -- [self view]
```

Кастомизация

- 1 Можем создать regex-алиас и делать подставному параметра в нужное место

Настройки объявляются в файле ~/.lldbinit

```
# ~/.lldbinit
```

```
command regex tb 's/(.+)/expr -l objc -0 --  
@import AppKit; { NSView *view = (NSView *)%1;  
view.wantsLayer = true; CALayer *layer =  
(CALayer*)[view layer]; BOOL removed = 0; for (id  
s in [layer sublayers]) { if ([[NSString *)s  
name] isEqualToString:@"LLDBHightlight"]) { [s  
removeFromSuperlayer]; removed = 1; [view  
display]; break; } } if (!removed) { id hl =  
[CALayer layer]; [hl setName:@"LLDBHightlight"];  
[hl setBorderWidth:2.0]; [hl setBorderColor:  
[[NSColor redColor] CGColor]]; [hl  
setMasksToBounds:0]; [hl setZPosition:99999.0];  
[hl setValue:[view valueForKey:@"bounds"]  
forKey:@"frame"]; [layer addSublayer:hl]; [view  
display]; } }/'
```

Кастомизация

- 1 Можем создать regex-алиас и делать подставному параметра в нужное место

Настройки объявляются в файле ~/.lldbinit

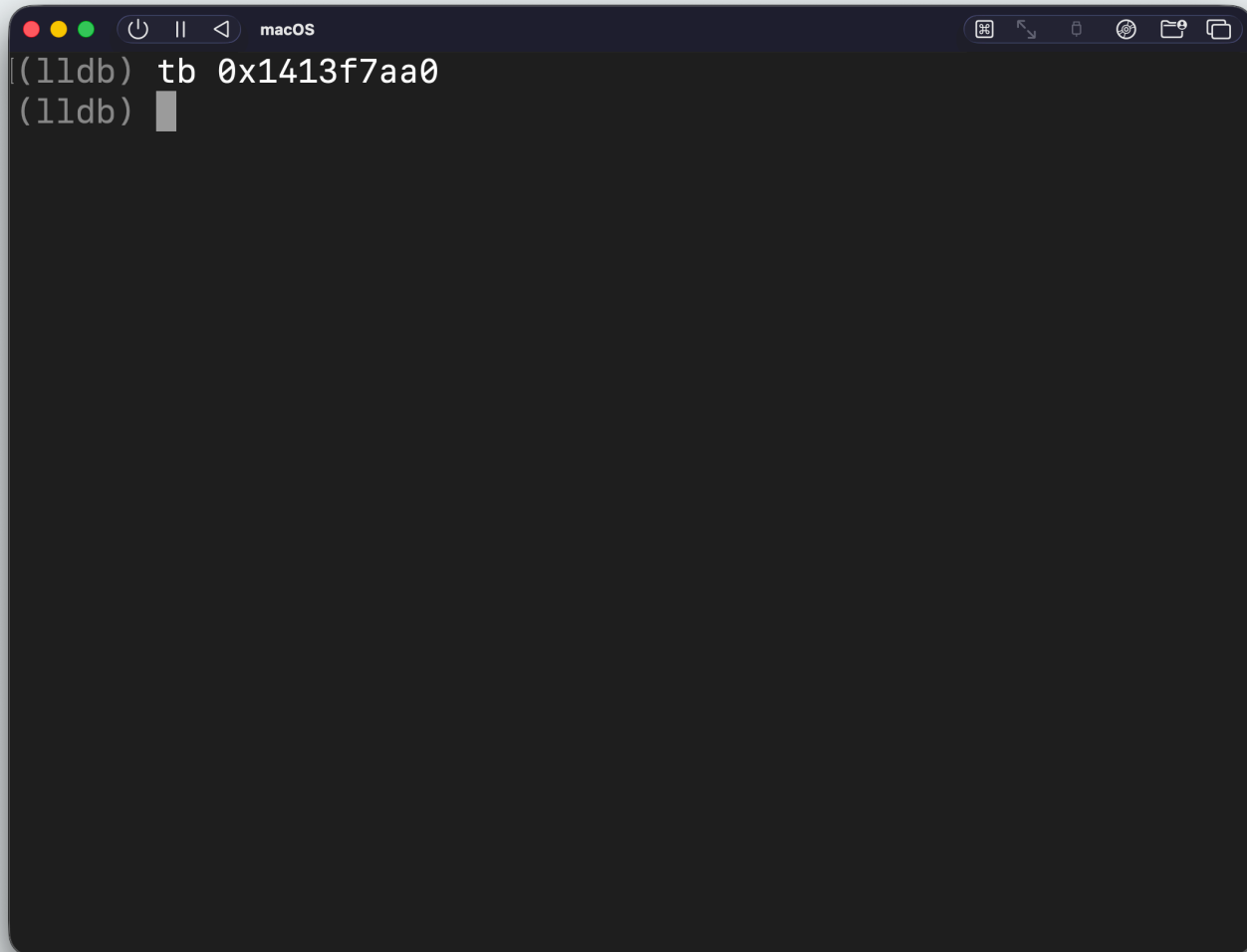
```
# ~/.lldbinit
```

```
command regex tb 's/(.+)/expr -l objc -o --  
@import UIKit ; { NSView *view = (NSView *)%1;  
view.wantsLayer = true; CALayer *layer =  
(CALayer*)[view layer]; BOOL removed = 0; r (id  
s in [layer sublayers]) { if ([[NSString *)s  
name]) isEqualToString:@"LLDBHighlight"]) { [s  
removeFromSuperlayer]; removed = 1; [view  
display]; break; } } if (!removed) { id hl =  
[CALayer layer]; [hl setName:@"LLDBHighlight"];  
[hl setBorderWidth:2.0]; [hl setBorderColor:  
[[NSColor redColor] CGColor]]; [hl  
setMasksToBounds:0]; [hl setZPosition:99999.0];  
[hl setValue:[view valueForKey:@"bounds"]  
forKey:@"frame"]; [layer addSublayer:hl]; [view  
display]; } }/'
```

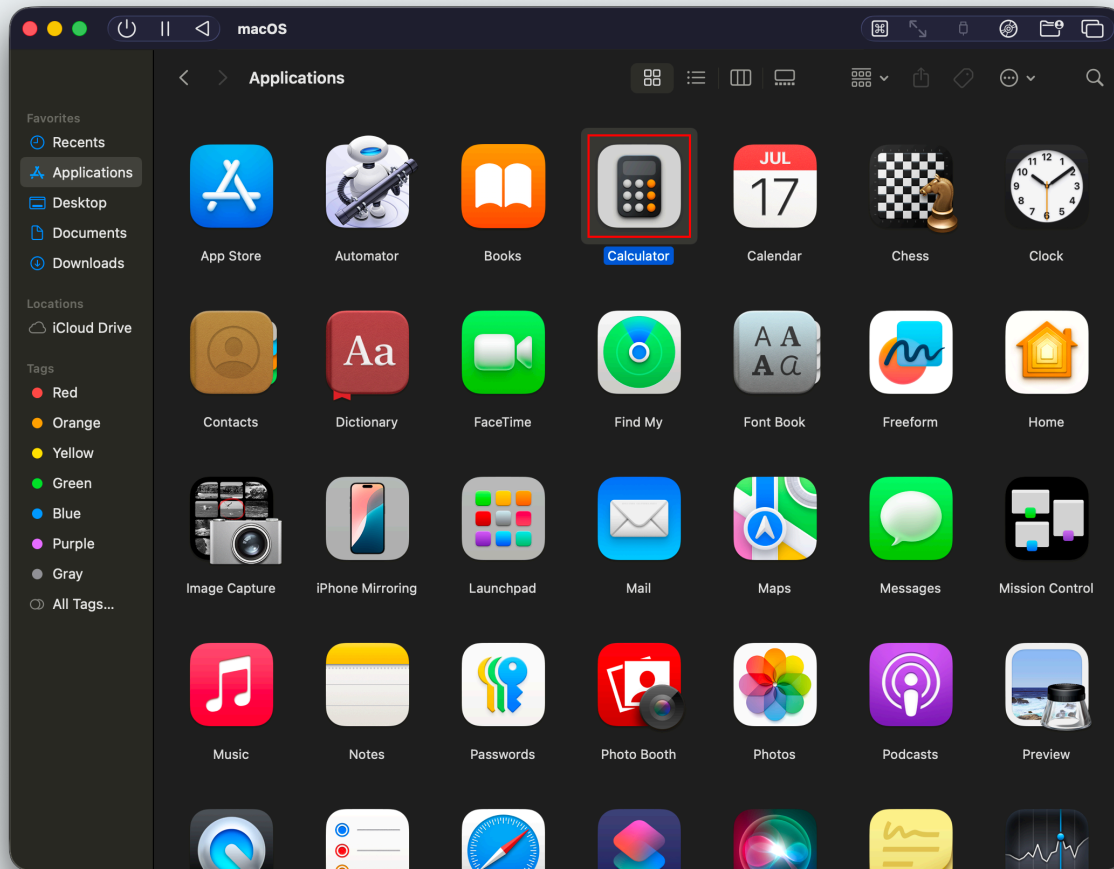
Кастомизация

- 1 Можем создать regex-алиас и делать подставному параметра в нужное место

Настройки объявляются в файле ~/.lldbinit

A screenshot of a macOS terminal window with a dark background. The window title bar shows standard macOS window controls (red, yellow, green buttons, power, pause, back) and the text 'macOS'. The terminal content shows two lines of lldb commands: '(lldb) tb 0x1413f7aa0' followed by '(lldb) █' where the █ represents a cursor.

```
(lldb) tb 0x1413f7aa0
(lldb) █
```



Команды DerekSelander

- 1 В OpenSource можно встретить коллекции ГОТОВЫХ КОМАНД
- 2 **Дерек Селандер** — автор книги *Advanced Apple Debugging and Reverse Engineering*
- 2 Фокус на исследованиях и реверсе

github.com/DerekSelander/LLDB

Platform Solutions Resources Open Source Enterprise Pricing

Search Sign in Sign up

DerekSelander / LLDB Public

Notifications Fork 207 Star 1.8k

Code Issues 11 Pull requests 4 Actions Projects Security Insights

main Go to file Code

LOLgrep	fixup broken lookup -X, yoink co...	1f6c9cf · 3 months ago	377 Commits
Media	Updated book image to v2	8 years ago	
lldb_commands	fixup broken lookup -X, yoink com...	3 months ago	
.gitignore	Ignore Media dir	8 years ago	
CHANGELOG.txt	Updated changelong	6 years ago	
LICENSE	Added Razeware LLC clause to LI...	6 years ago	
README.md	[README.md] minor typo fix	6 years ago	
_config.yml	Set theme jekyll-theme-tactile	8 years ago	

README GPL-2.0 license

LLDB

About

A collection of LLDB aliases/regexes and Python scripts to aid in your debugging sessions

python debugging ios xcode

lldb

Readme

GPL-2.0 license

1.8k

52 v

207

Report r

Releas

3 tags

Команды

DerekSelander

- 1 В OpenSource можно встретить коллекции ГОТОВЫХ КОМАНД
- 2 **Дерек Селандер** — автор книги *Advanced Apple Debugging and Reverse Engineering*
- 2 Фокус на исследованиях и реверсе

lookup

Perform a regular expression search for stuff in an executable

Example:

```
# Find all methods that contain the phrase viewDidLoad
(lldb) lookup viewDidLoad

# Find a summary of all the modules that have a (known) function containing the phrase viewDi
(lldb) lookup viewDidLoad -s

# Search for Objective-C code in a stripped module (i.e. in SpringBoard)
(lldb) loo -x StocksFramework .

# Search for Objective-C code containing the case insensitive phrase init inside a stripped m
(lldb) lookup -X (?i)init

# Search for all hardcoded, embedded `char *` inside an executable containing the phrase *http
(lldb) lookup -S http -m UIKit

# Dump all the md5'd base64 keys in libMobileGestalt along w/ the address in memory
(lldb) loo -S ^[a-zA-Z0-9\+]{22,22}$ -m libMobileGestalt.dylib -l

# Dump all the global bss code referenced by DWARF. Ideal for accessing `static` variables wh
(lldb) lookup . -g HonoluluArt -l

# Look for phrase "nominal" (Swift's nominal type descriptors) in module "SwiftTest" and get
(lldb) lookup -G SwiftTest nominal -l
```

Источник: <https://github.com/derekselander/lldb>

Команды

Chisel

- 1 Коллекция команд от Facebook *
- 2 Фокус на повседневной разработке

* Компания Meta признана экстремистской организацией и запрещена на территории РФ

github.com/facebook/chisel

Platform Solutions Resources Open Source Enterprise Pricing Sign in Sign up

facebook / chisel Public Notifications Fork 804 Star 9.2k

Code Issues 44 Pull requests 6 Actions Projects Models Wiki Security

main Go to file Code

Adam Ernst and facebook-github-bot R... fdbf067 · 5 months ago 467 Commits

Chisel	[Licensing] Switch from BSD+Pate...	6 years ago
commands	Remove old options that no longer...	5 months ago
.gitignore	Remove pyc files and add gitignor...	11 years ago
CODE_OF_CONDUCT.md	OSS Automated Fix: Addition of C...	6 years ago
CONTRIBUTING.md	Add note to Contribution Guide ab...	5 years ago
LICENSE	[Licensing] Switch from BSD+Pate...	6 years ago
README.md	Format README	2 years ago
fbchiselldb.py	migrate fbobjc/ to ruff	11 months ago
fbchiselldbbase.py	formatting changes from black 22...	3 years ago
fbchiselldbinputhelpers.py	Rename chisel's fbldb to fbchiselll...	5 years ago
fbchiselldbinputhelpers.py	Remove unnecessary flake8_P050	5 years ago

About

Chisel is a collection of LLDB commands to assist debugging iOS apps.

- Readme
- MIT license
- Code of conduct
- Contributing
- Security policy
- Active issues
- Custom issues

9.2k Stars

276 Open issues

804 Forks

Report problem

Releases

Renames, Ranges, and the Co... Latest on Aug 17, 2020

Команды

Chisel

- 1 Коллекция команд от Facebook *
- 2 Фокус на повседневной разработке

* Компания Meta признана экстремистской организацией и запрещена на территории РФ

Command	Description
<code>pviews</code>	Print the recursive view description for the key window.
<code>pvc</code>	Print the recursive view controller description for the key window.
<code>visualize</code>	Open a <code>UIImage</code> , <code>CGImageRef</code> , <code>UIView</code> , <code>CALayer</code> , <code>NSData</code> (of an image), <code>UIColor</code> , <code>CIColor</code> , <code>CIImage</code> , <code>CGColorRef</code> or <code>CVPixelBuffer</code> in Preview.app on your Mac.
<code>fv</code>	Find a view in the hierarchy whose class name matches the provided regex.
<code>fvc</code>	Find a view controller in the hierarchy whose class name matches the provided regex.
<code>show/hide</code>	Show or hide the given view or layer. You don't even have to continue the process to see the changes!
<code>mask/unmask</code>	Overlay a view or layer with a transparent rectangle to visualize where it is.
<code>border/unborder</code>	Add a border to a view or layer to visualize where it is.
<code>caflush</code>	Flush the render server (equivalent to a "repaint" if no animations are in-flight).

Источник: <https://github.com/facebook/chisel>

Умеем исследовать любой процесс

Умеем изменять поведение

Не хотим каждый раз вызывать LLDB

```

#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main() {
    srand(time(NULL));
    printf("Random number: %d\n", rand());
    return 0;
}
```

Внедрение

- 1 Хотим подменить случайное значение
- 2 Можем воспользоваться техникой **Interposing**

```
#include <stdio.h>
#include <stdlib.h>
#include <time.h>

int main() {
    srand(time(NULL));
    printf("Random number: %d\n", rand());
    return 0;
}
```



```
static _Atomic(rand_fn_t) orig_rand_atomic = NULL;
```

```
__attribute__((constructor))
```

```
static void init_orig_rand(void) {
```

```
    rand_fn_t pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");
```

```
    if (pt) {
```

```
        atomic_store_explicit(&orig_rand_atomic, pt,  
                               memory_order_release);
```

```
    }
```

```
}
```

```
static int custom_rand(void) {
```

```
    rand_fn_t orig_pt = atomic_load_explicit(&orig_rand_atomic,  
                                              memory_order_acquire);
```

```
    if (!orig_pt) {
```

```
        orig_pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");  
        atomic_store_explicit(&orig_rand_atomic, orig_pt,  
                               memory_order_release);
```

```
    }
```

```
    return 1337;
```

```
}
```

```
static _Atomic(rand_fn_t) orig_rand_atomic = NULL;

__attribute__((constructor))
static void init_orig_rand(void) {
    rand_fn_t pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");

    if (pt) {
        atomic_store_explicit(&orig_rand_atomic, pt,
                               memory_order_release);
    }
}

static int custom_rand(void) {
    rand_fn_t orig_pt = atomic_load_explicit(&orig_rand_atomic,
                                              memory_order_acquire);

    if (!orig_pt) {
        orig_pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");
        atomic_store_explicit(&orig_rand_atomic, orig_pt,
                               memory_order_release);
    }

    return 1337;
}
```

```
}  
  
static int custom_rand(void) {  
    rand_fn_t orig_pt = atomic_load_explicit(&orig_rand_atomic,  
                                             memory_order_acquire);  
  
    if (!orig_pt) {  
        orig_pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");  
        atomic_store_explicit(&orig_rand_atomic, orig_pt,  
                              memory_order_release);  
    }  
  
    return 1337;  
}
```

```
__attribute__((used))  
static struct {  
    const void *replacement;  
    const void *original;  
} interposers[]  
__attribute__((section("__DATA,__interpose"))) = {  
    { (const void *)custom_rand, (const void *)rand }  
};
```

DYLD_INSERT_LIBRARIES

```
igoor_bb@vm-sonoma Inject % clang main.c -o main
igoor_bb@vm-sonoma Inject % ./main
Random number: 995012717
igoor_bb@vm-sonoma Inject % ./main
Random number: 995029524
igoor_bb@vm-sonoma Inject % clang -dynamiclib -arch arm64 \
-o interpose_rand.dylib interpose_rand.c
igoor_bb@vm-sonoma Inject % DYLD_INSERT_LIBRARIES=\
> "$PWD/interpose_rand.dylib" ./main
Random number: 1337
igoor_bb@vm-sonoma Inject % █
```

```
igoor_bb@vm-sonoma Inject % clang main.c -o main
igoor_bb@vm-sonoma Inject % ./main
Random number: 995012717
igoor_bb@vm-sonoma Inject % ./main
Random number: 995029524
igoor_bb@vm-sonoma Inject % clang -dynamiclib -arch arm64 \
-o interpose_rand.dylib interpose_rand.c
igoor_bb@vm-sonoma Inject % DYLD_INSERT_LIBRARIES=\
> "$PWD/interpose_rand.dylib" ./main
Random number: 1337
igoor_bb@vm-sonoma Inject % █
```

```
igoor_bb@vm-sonoma Inject % clang main.c -o main
igoor_bb@vm-sonoma Inject % ./main
Random number: 995012717
igoor_bb@vm-sonoma Inject % ./main
Random number: 995029524
igoor_bb@vm-sonoma Inject % clang -dynamiclib -arch arm64 \
-o interpose_rand.dylib interpose_rand.c
igoor_bb@vm-sonoma Inject % DYLD_INSERT_LIBRARIES=\
> "$PWD/interpose_rand.dylib" ./main
Random number: 1337
igoor_bb@vm-sonoma Inject %
```

Где здесь iOS разработка?

```
typedef void (^URLCompletion)(NSData *, NSURLResponse *, NSError *);
```

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

```
#import <objc/runtime.h>
```

```
static inline void Swizzle(Class c, SEL o, SEL r) {  
    method_exchangeImplementations(  
        class_getInstanceMethod(c, o),  
        class_getInstanceMethod(c, r)  
    );  
}
```

```
static inline NSString *HTTPMethodOrGET(NSURLRequest *r) {  
    return r.HTTPMethod ?: @"GET";  
}
```

```
@implementation NSURLSession (URLLoggerSwizzle)
```

```
+ (void)load {  
    static dispatch_once_t once;  
    dispatch_once(&once, ^{  
        Class c = self;
```

```
        Swizzle(c
```

@implementation NSURLSession (URLLoggerSwizzle)

```

+ (void)load {
    static dispatch_once_t once;
    dispatch_once(&once, ^{
        Class c = self;

        Swizzle(c,
                @selector(dataTaskWithRequest:completionHandler:),
                @selector(urllogger_dataTaskWithRequest:completionHandler:));

        Swizzle(c,
                @selector(dataTaskWithURL:completionHandler:),
                @selector(urllogger_dataTaskWithURL:completionHandler:));

        Swizzle(c,
                @selector(dataTaskWithRequest:),
                @selector(urllogger_dataTaskWithRequest:));

        Swizzle(c,
                @selector(dataTaskWithURL:),
                @selector(urllogger_dataTaskWithURL:));
    });
}

- (NSURLSessionDataTask *)urllogger_dataTaskWithRequest:(NSURLRequest *)request
    completionHandler:(URLCompletion)completionHandler
{
    NSURL *u = request.URL;

```

```
@implementation NSURLSession (URLLoggerSwizzle)
```

```
+ (void)load {
    static dispatch_once_t once;
    dispatch_once(&once, ^{
        Class c = self;
```

```
        Swizzle(c,
                @selector(dataTaskWithRequest:completionHandler:),
                @selector(urllogger_dataTaskWithRequest:completionHandler:));
```

```
        Swizzle(c,
                @selector(dataTaskWithURL:completionHandler:),
                @selector(urllogger_dataTaskWithURL:completionHandler:));
```

```
        Swizzle(c,
                @selector(dataTaskWithRequest:),
                @selector(urllogger_dataTaskWithRequest:));
```

```
        Swizzle(c,
                @selector(dataTaskWithURL:),
                @selector(urllogger_dataTaskWithURL:));
```

```
    });
```

```
}
```

```
- (NSURLSessionDataTask *)urllogger_dataTaskWithRequest:(NSURLRequest *)request
    completionHandler:(URLCompletion)completionHandler
```

```
{
```

```
    NSURL *u = request.URL;
```

```
    NSURL *u = request.URL;
    if (u) NSLog(@"[URLSwizzle] %@ %@", HTTPMethodOrGET(request), u.absoluteString);

    return [self urllogger_dataTaskWithRequest:request
            completionHandler:completionHandler];
}
```

```
- (NSURLSessionDataTask *)urllogger_dataTaskWithURL:(NSURL *)url
    completionHandler:(URLCompletion)completionHandler
{
    if (url) NSLog(@"[URLSwizzle] GET %@", url.absoluteString);

    return [self urllogger_dataTaskWithURL:url
            completionHandler:completionHandler];
}
```

```
- (NSURLSessionDataTask *)urllogger_dataTaskWithRequest:(NSURLRequest *)request {
    NSURL *u = request.URL;
    if (u) NSLog(@"[URLSwizzle] %@ %@", HTTPMethodOrGET(request), u.absoluteString);
    return [self urllogger_dataTaskWithRequest:request];
}
```

```
- (NSURLSessionDataTask *)urllogger_dataTaskWithURL:(NSURL *)url {
    if (url) NSLog(@"[URLSwizzle] GET %@", url.absoluteString);
    return [self urllogger_dataTaskWithURL:url];
}
```

@end

```
        NSURL *u = request.URL;
        if (u) NSLog(@"[URLSwizzle] %@ %@", HTTPMethodOrGET(request), u.absoluteString);

        return [self urllogger_dataTaskWithRequest:request
                completionHandler:completionHandler];
    }

- (NSURLSessionDataTask *)urllogger_dataTaskWithURL:(NSURL *)url
    completionHandler:(URLCompletion)completionHandler
{
    if (url) NSLog(@"[URLSwizzle] GET %@", url.absoluteString);

    return [self urllogger_dataTaskWithURL:url
            completionHandler:completionHandler];
}

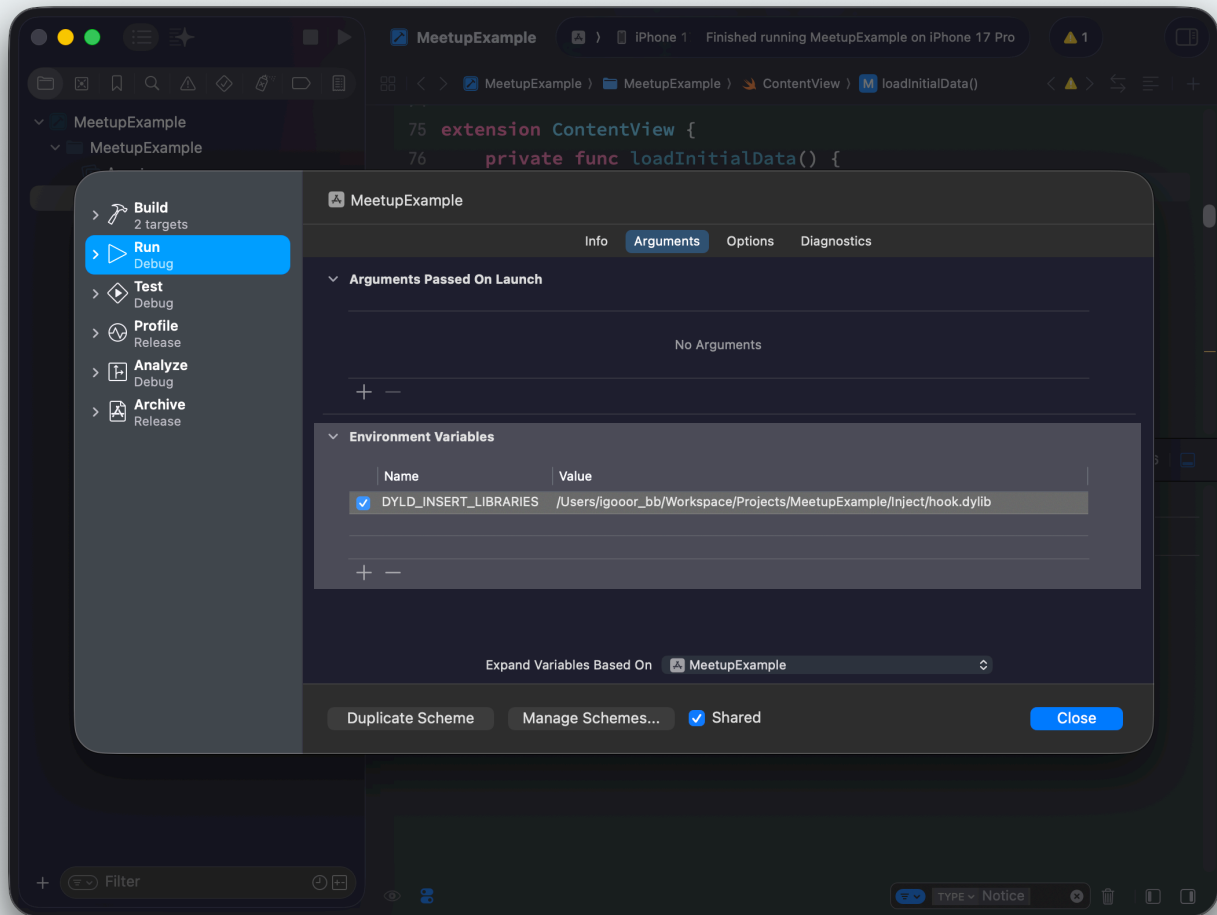
- (NSURLSessionDataTask *)urllogger_dataTaskWithRequest:(NSURLRequest *)request {
    NSURL *u = request.URL;
    if (u) NSLog(@"[URLSwizzle] %@ %@", HTTPMethodOrGET(request), u.absoluteString);
    return [self urllogger_dataTaskWithRequest:request];
}

- (NSURLSessionDataTask *)urllogger_dataTaskWithURL:(NSURL *)url {
    if (url) NSLog(@"[URLSwizzle] GET %@", url.absoluteString);
    return [self urllogger_dataTaskWithURL:url];
}

@end
```

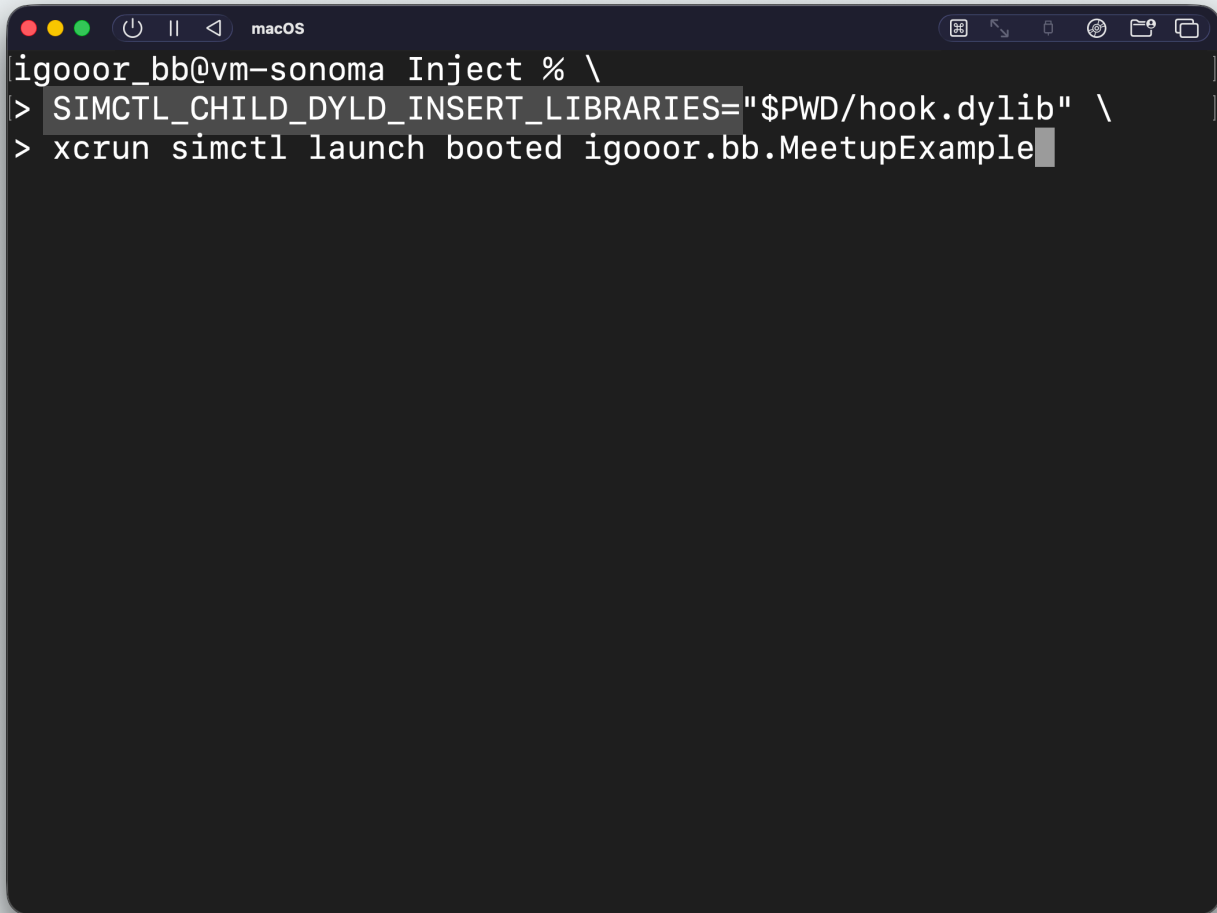
Внедрение Xcode

- 1 Возвращаемся к **DYLD_INSERT_LIBRARIES**
- 2 Можем передать путь к динамической библиотеке через **аргументы схемы**



Внедрение simctl launch

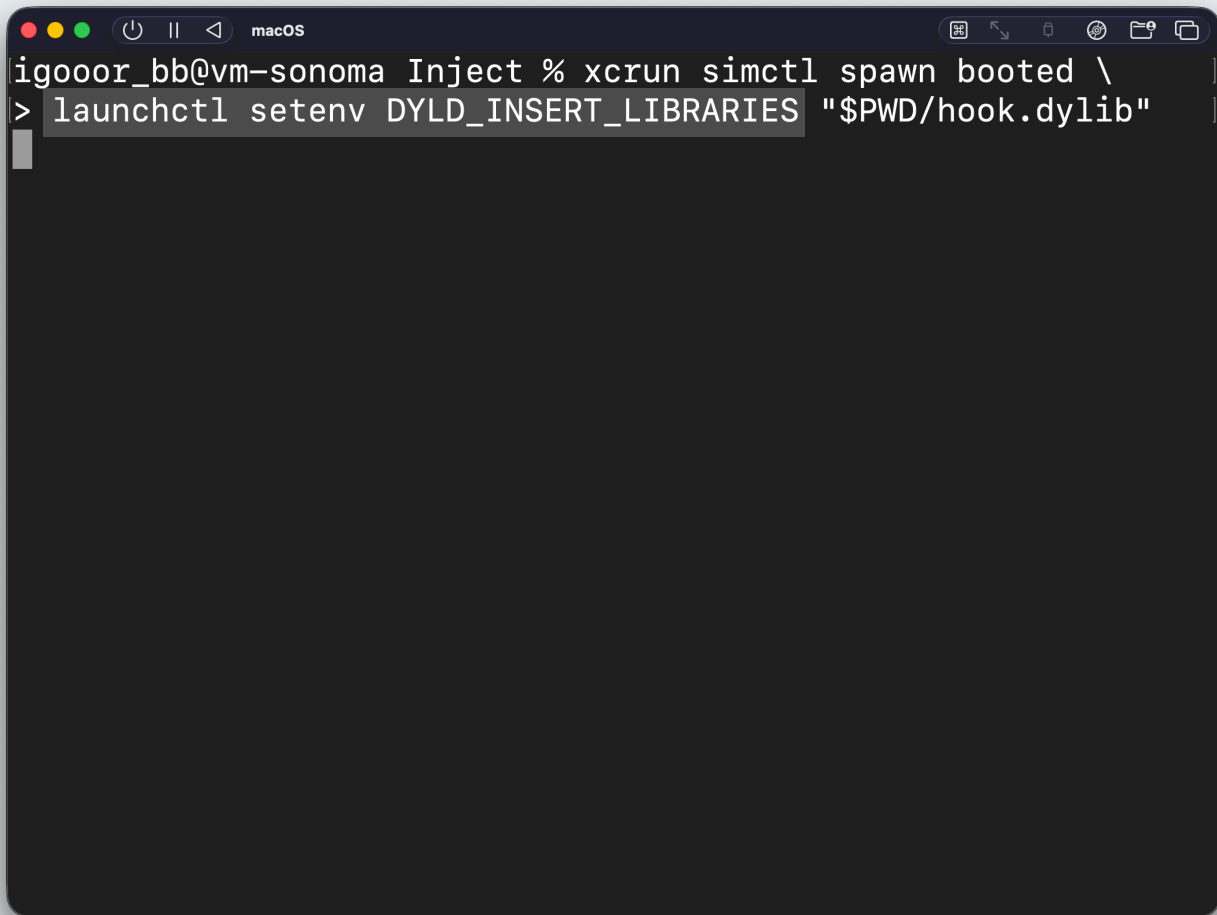
- 1 Нужно указать префикс **SIMCTL_CHILD_**



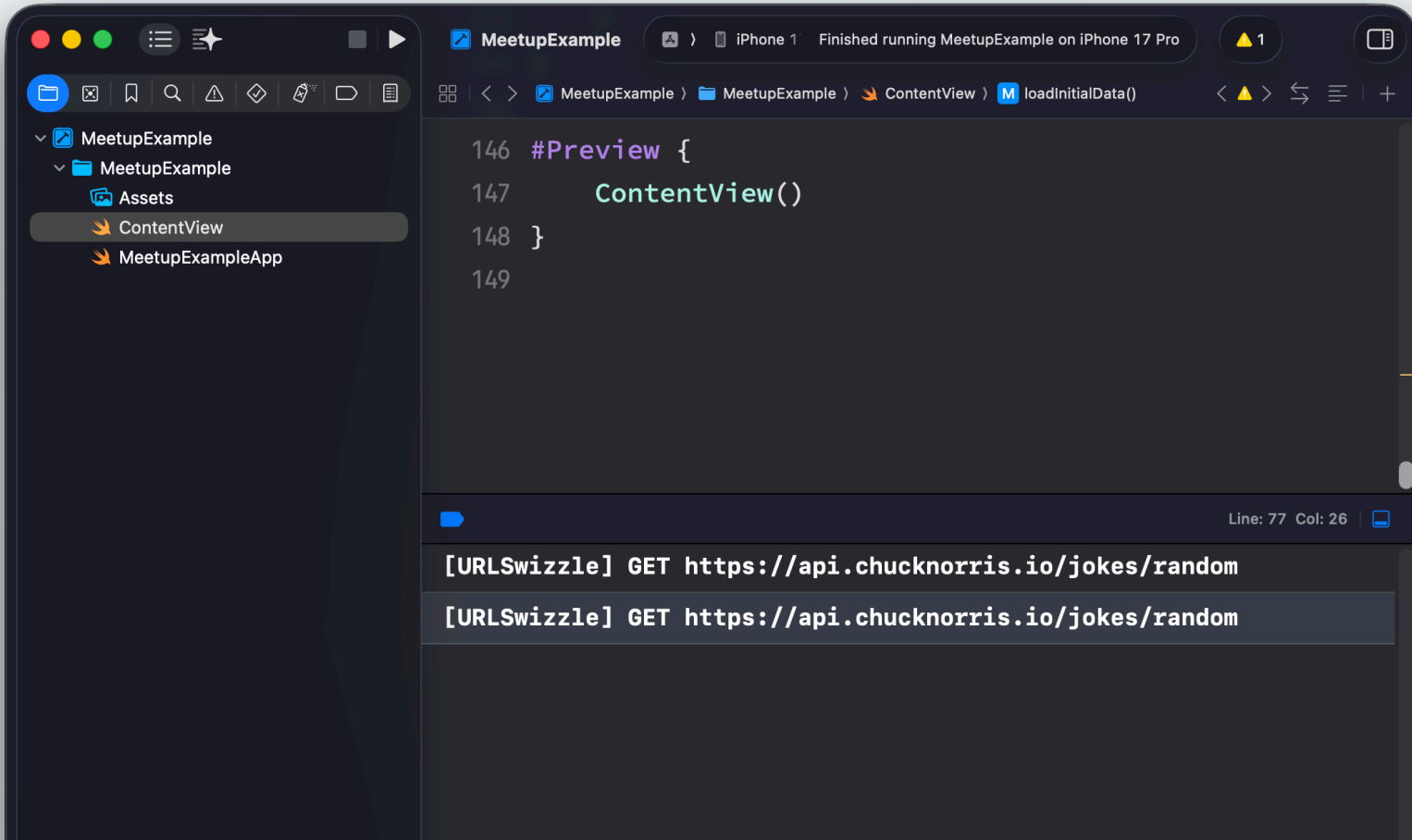
```
igoor_bb@vm-sonoma Inject % \  
> SIMCTL_CHILD_DYLD_INSERT_LIBRARIES="$PWD/hook.dylib" \  
> xcrun simctl launch booted igoor.bb.MeetupExample
```

Внедрение launchctl

- 1 Любой запускаемый процесс унаследует эти переменные
- 2 Можно менять сам симулятор (**SpringBoard**)



```
igoor_bb@vm-sonoma Inject % xcrun simctl spawn booted \  
> launchctl setenv DYLD_INSERT_LIBRARIES "$PWD/hook.dylib"
```



Применение Simulator Status Magic

- 1 Популярная утилита для настройки внешнего вида статус-бара симулятора
- 2 Используется для подготовки к снимкам-тестам



Применение

Что мы ещё можем сделать

- ✓ Настраиваемый Status Bar
- ✓ Полноценная имитация камеры
- ✓ Детерминированное время
- ✓ Детерминированный рандом
- ✓ И многое другое...



Заключение

Литература

01 Advanced Apple Debugging and Reverse Engineering

Книга **Дерека Селандера**
под издательством Kodesco

02 Reverse Engineering the iOS Simulator's SpringBoard

Выступление **Дерека Селандера**,
где он делает интересные трюки с LLDB

03 Swift Secrets

Книга **Джона Холдсворта**, создателя
Injection, SwiftTrace и Fortify

Заключение

Выводы

- ✔ LLDB — больше чем просто отладчик
- ✔ Исследовать можно легально
- ✔ Понимание платформы делает нас сильнее



Get in touch

Ссылки для связи
и дальнейшего
обсуждения



Игорь Белов
iOS Engineer

📍 @igooor_bb