avito.tech Москва 2026

Как разобрать macOS на части

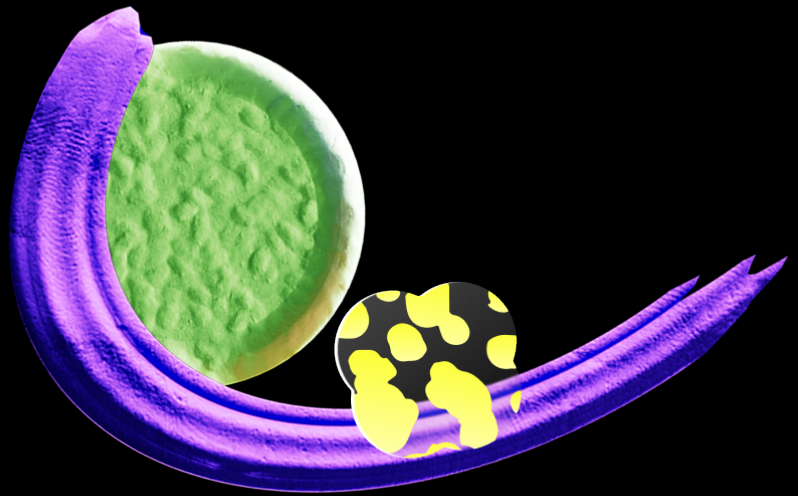
От исследования процессов к изменению логики

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

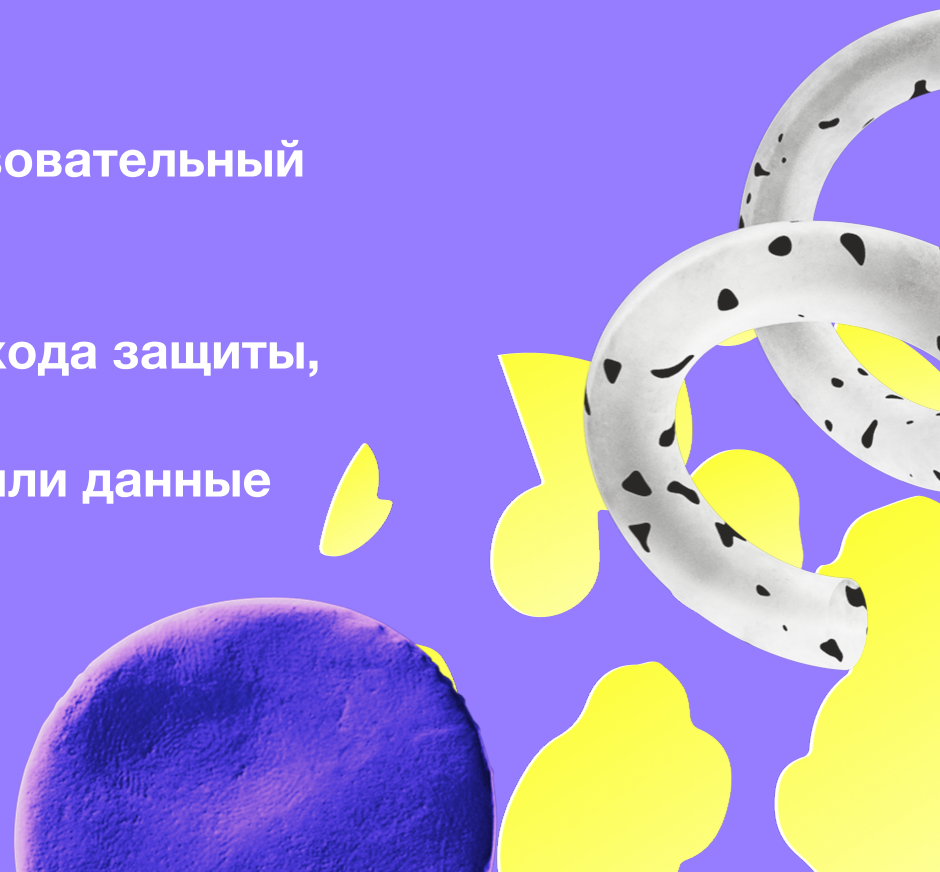
Содержание

Содержание

WARNING!

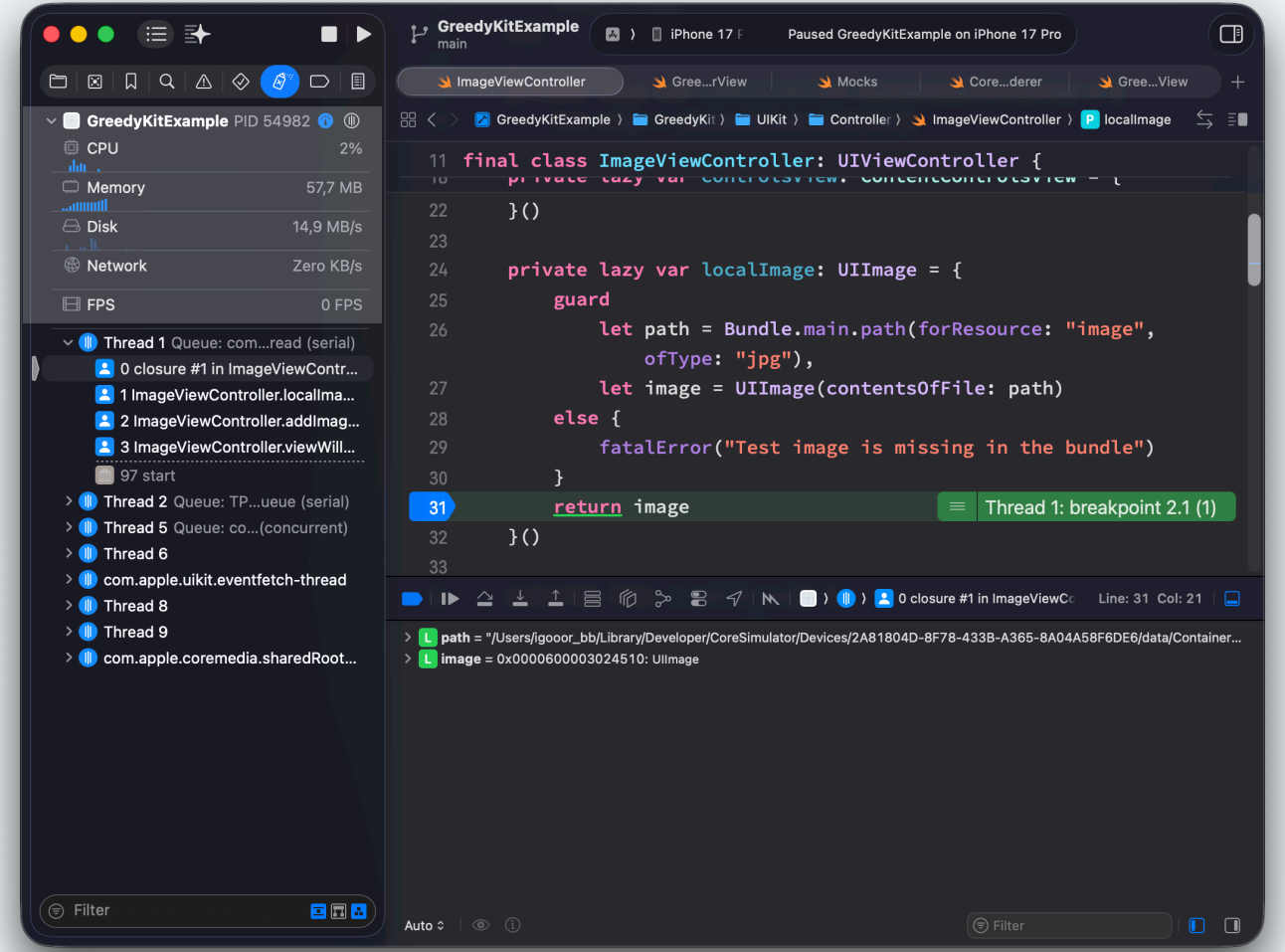
Доклад носит исключительно образовательный характер

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



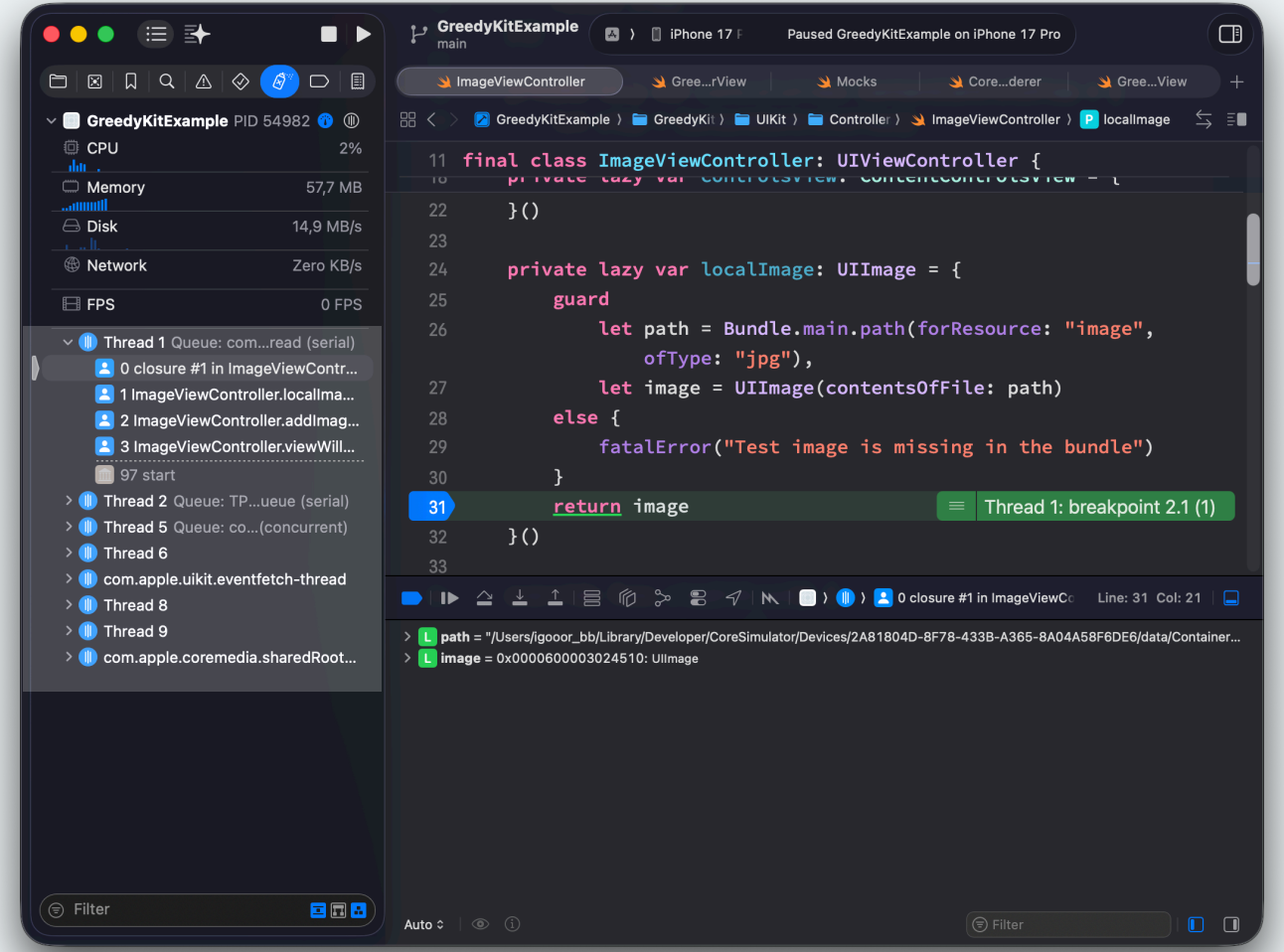
Точка входа – LLDB

1 Подключиться к живому приложению



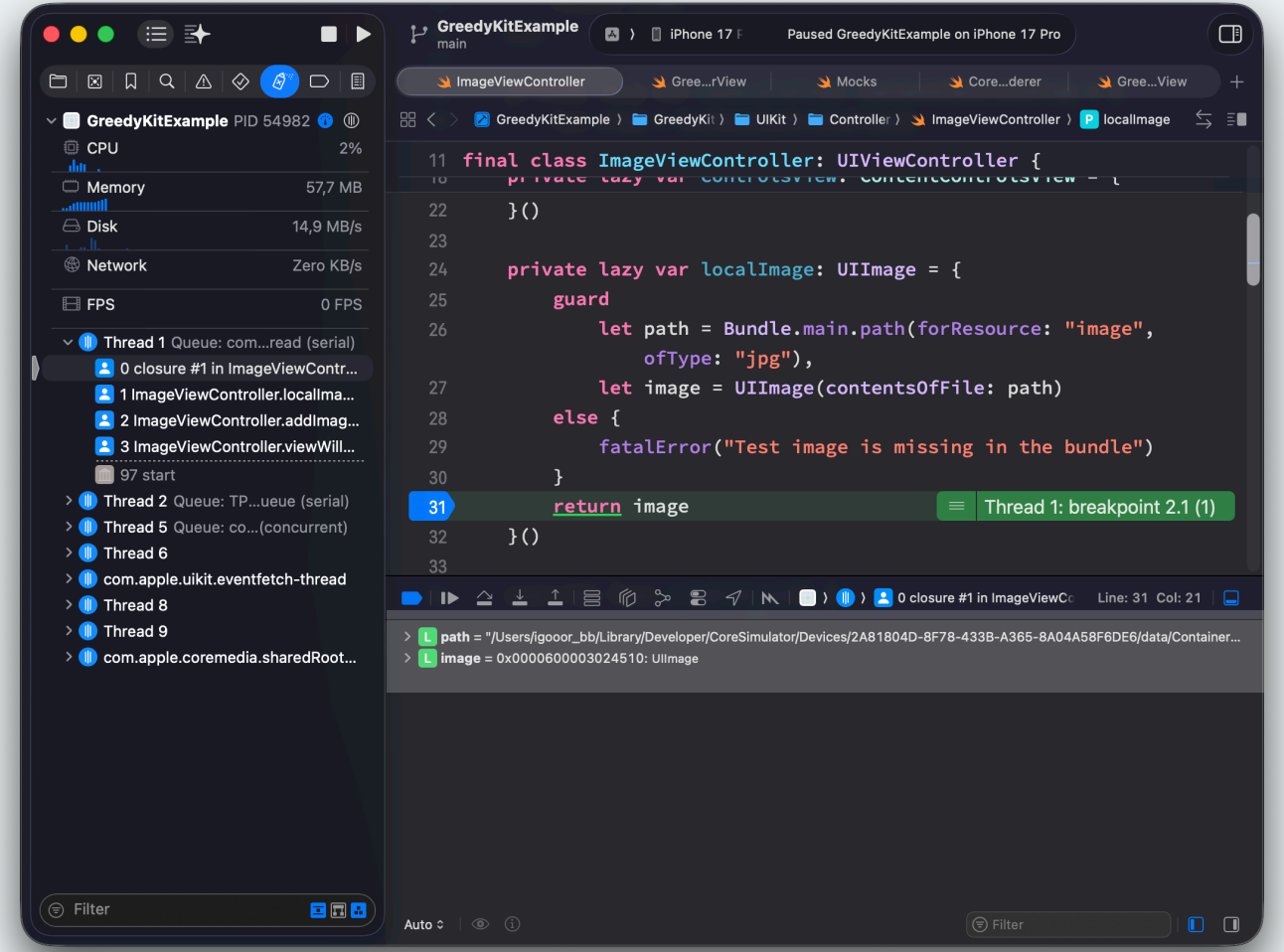
Точка входа – LLDB

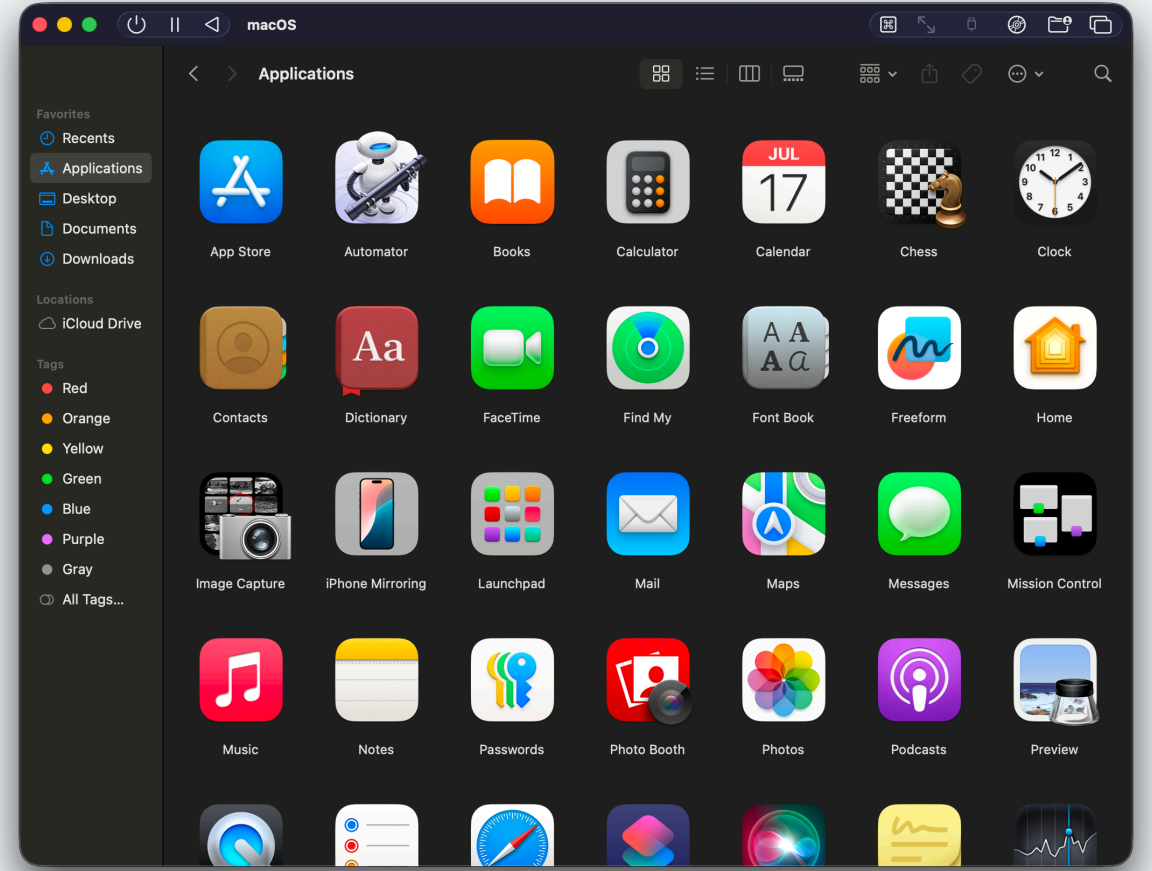
1. Подключиться к живому приложению
2. Остановиться в нужной точке

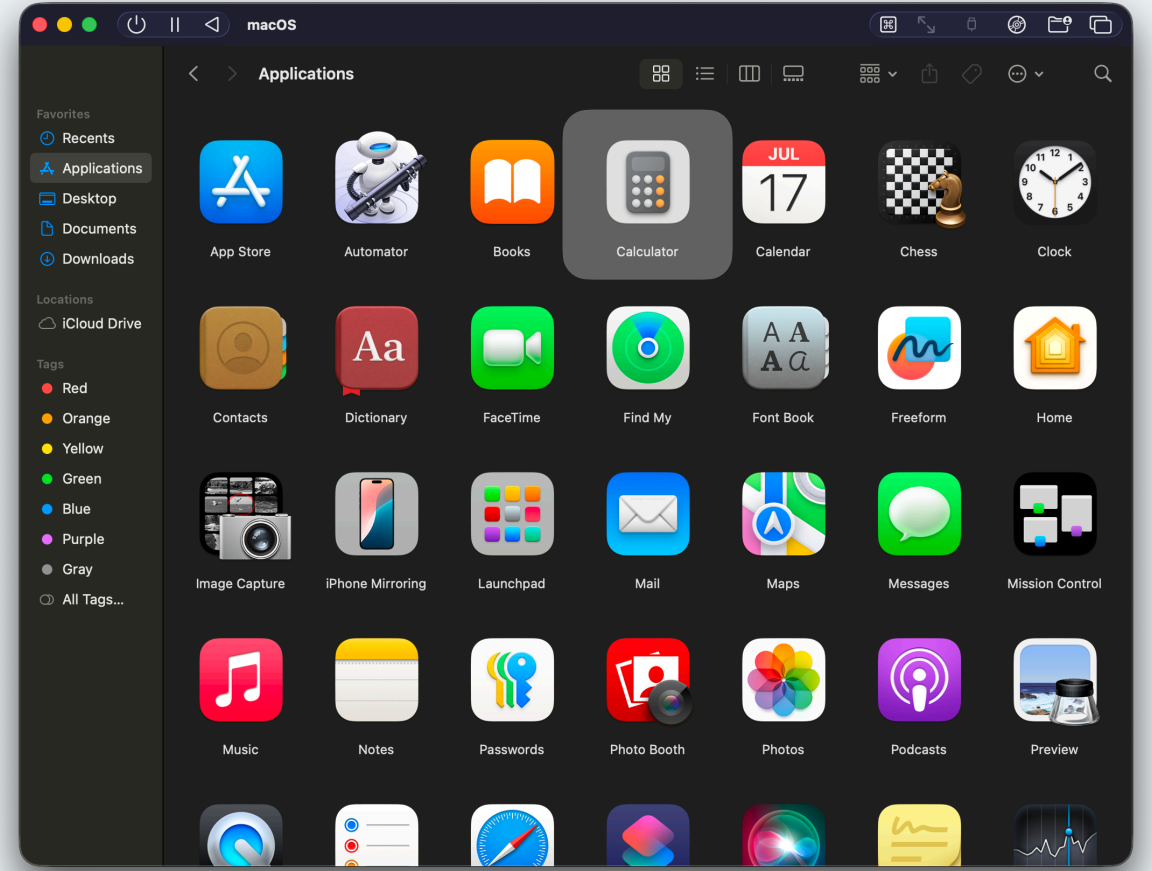


Точка входа — LLDB

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

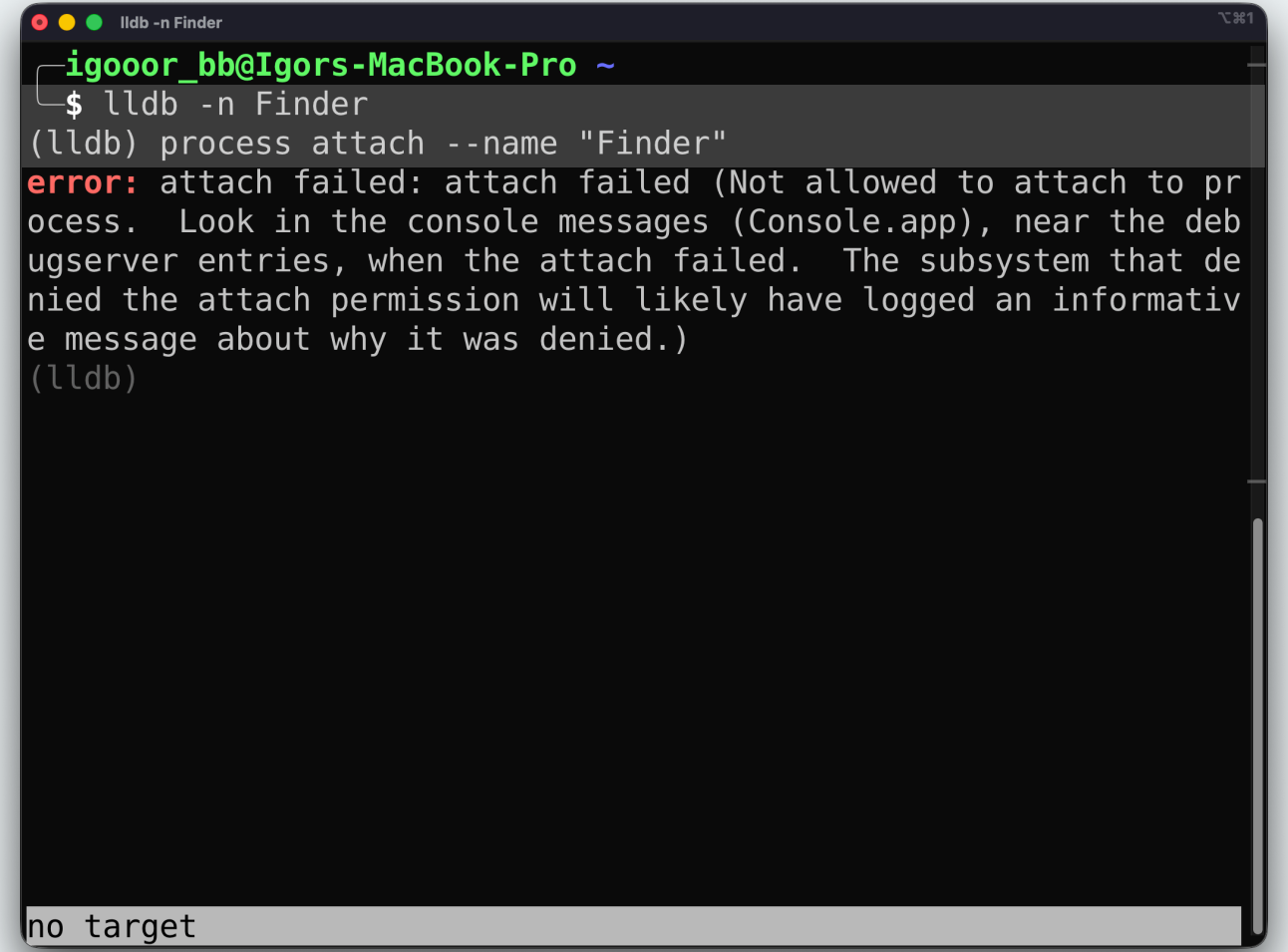



macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17


macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17

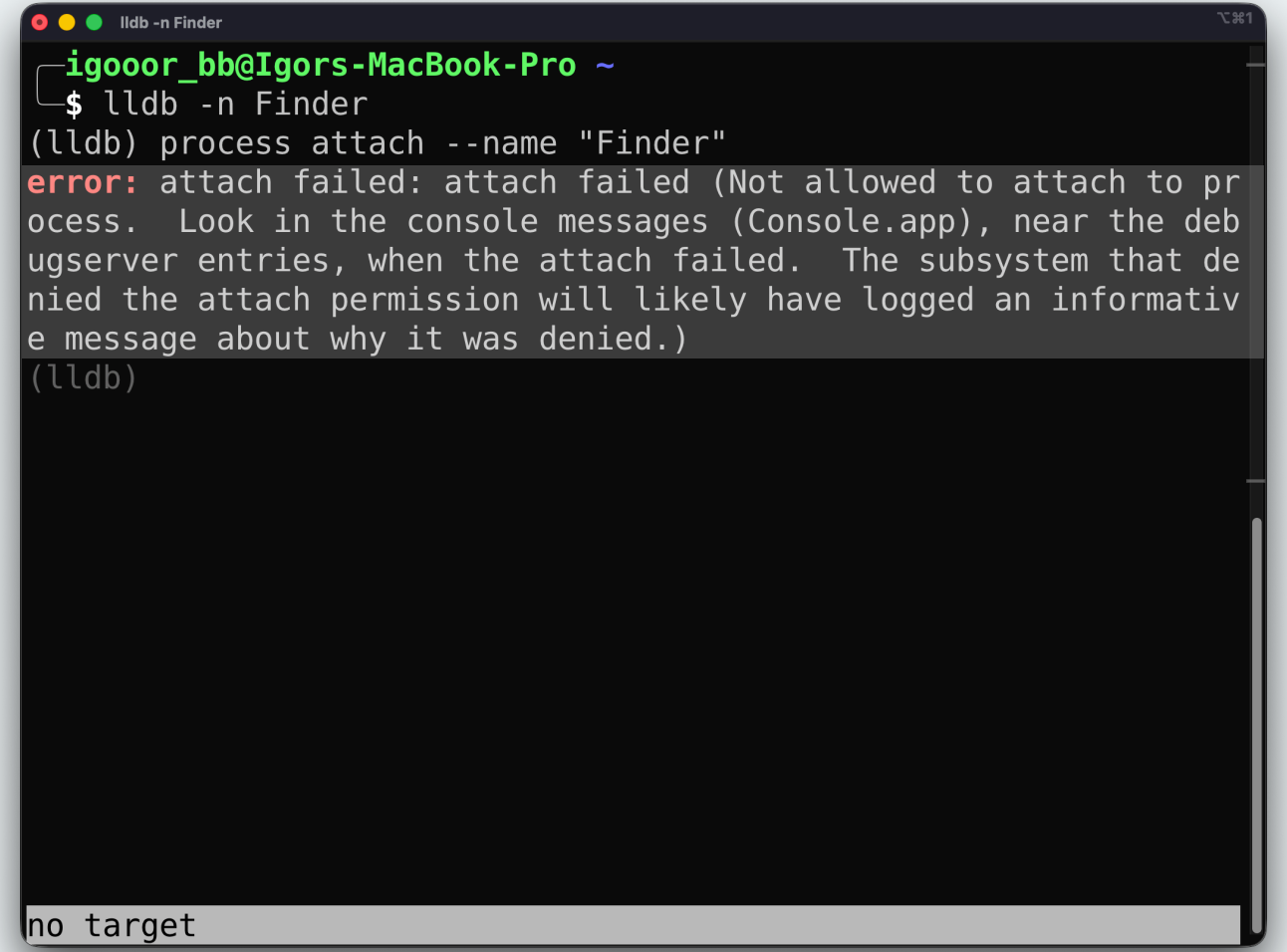
OS не обязана нас пускать

1 Системные процессы защищены



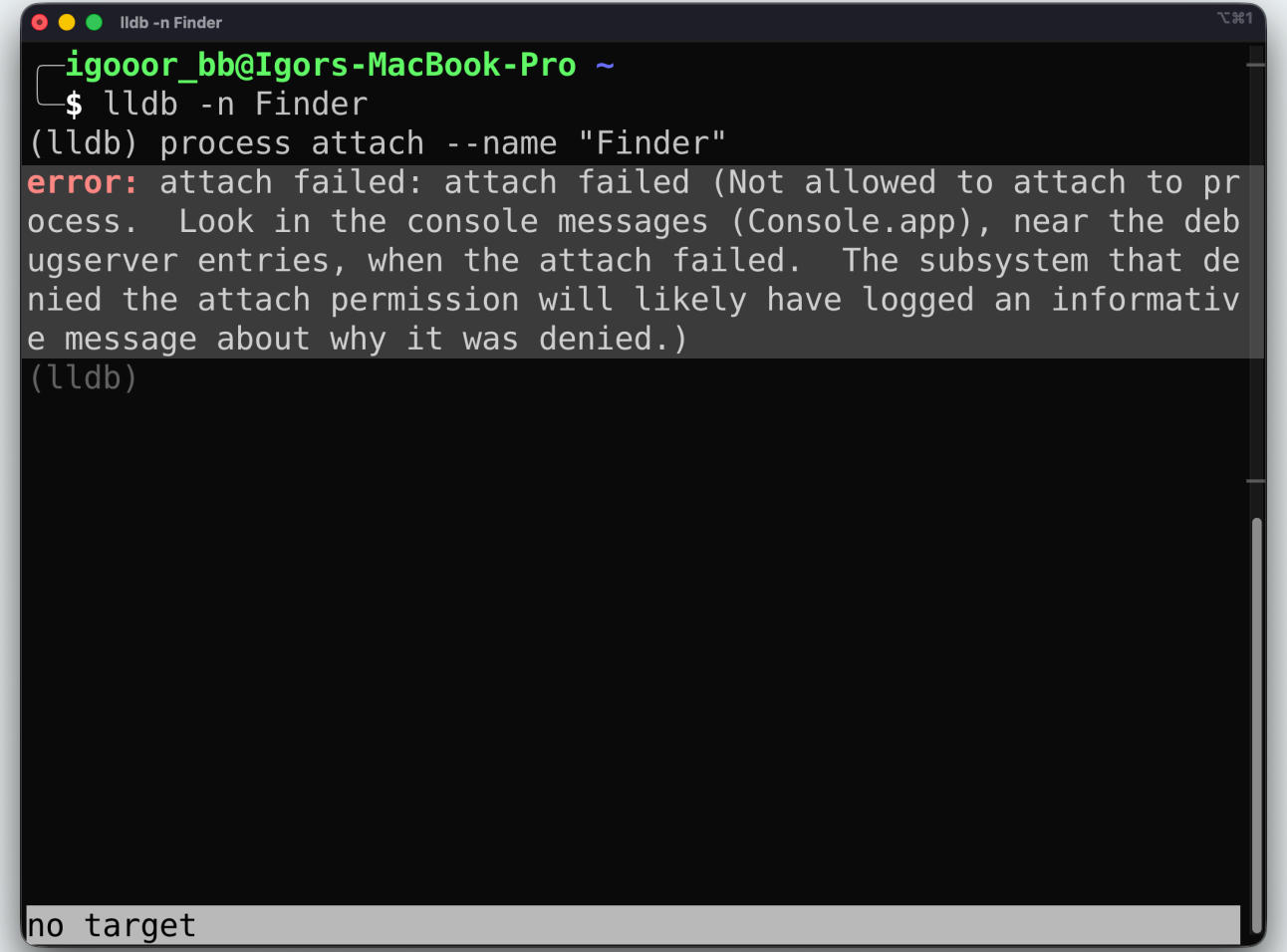
OS не обязана нас пускать

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



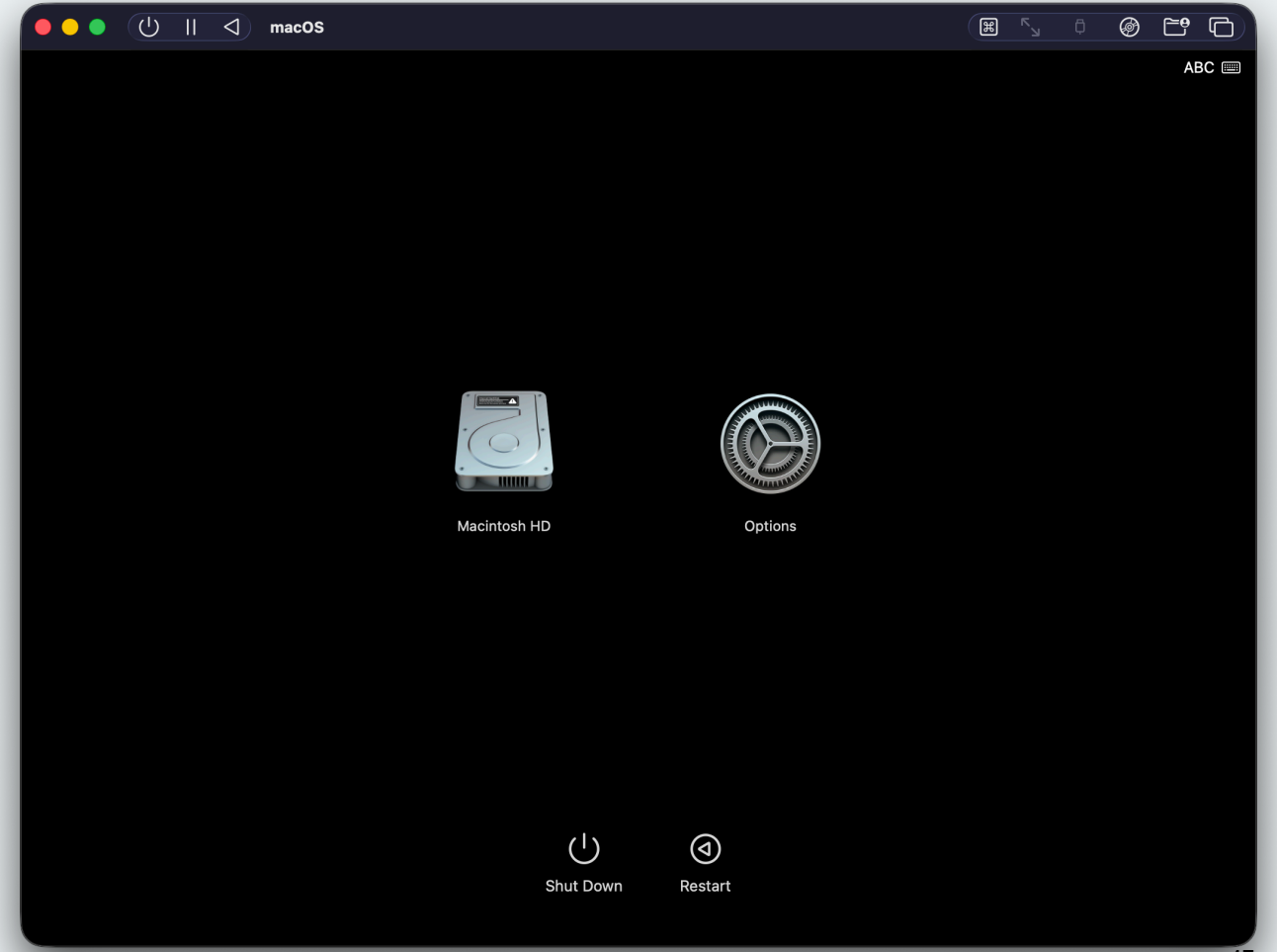
OS не обязана нас пускать

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



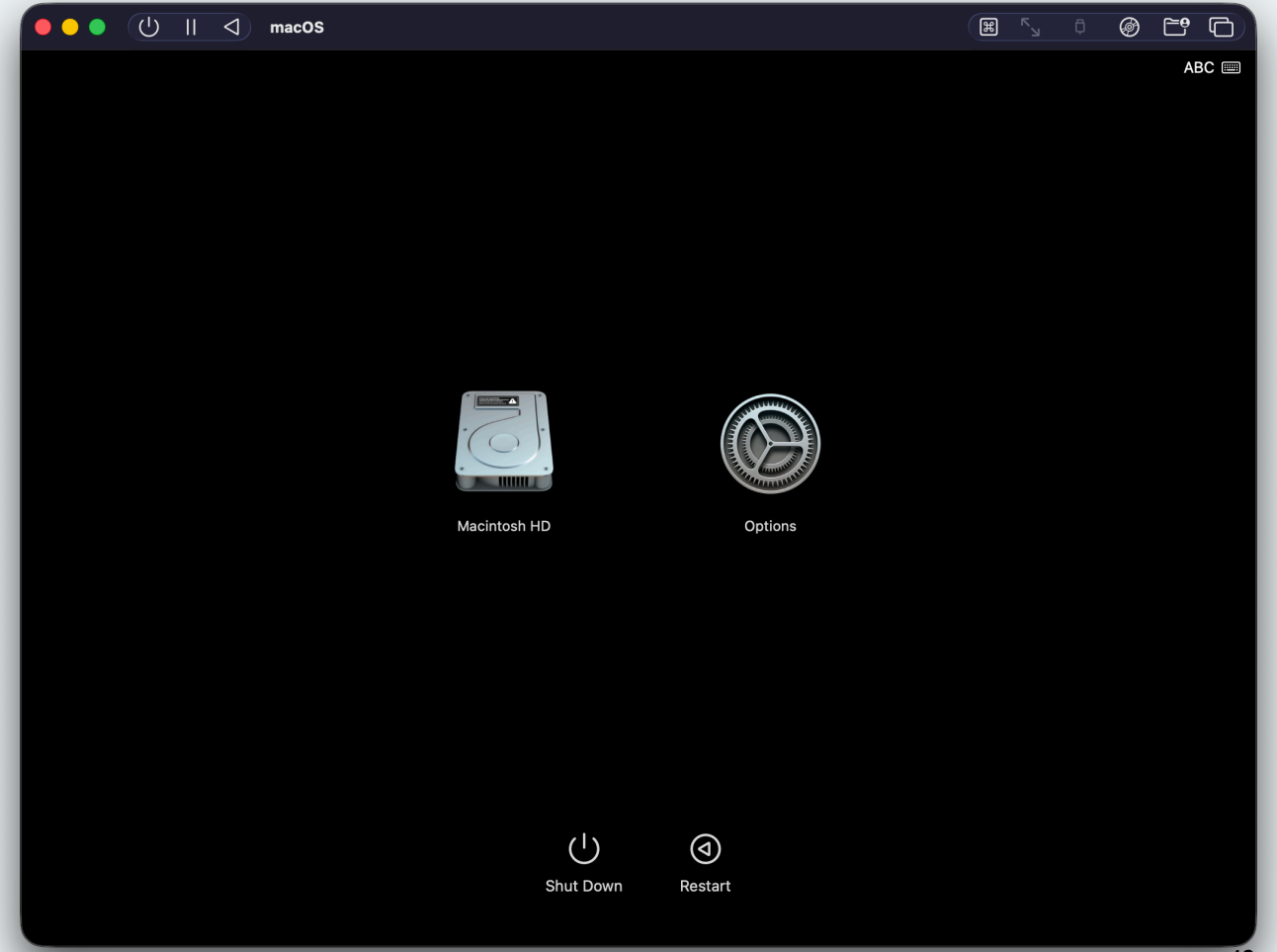
System Integrity Protection (SIP)

1 Механизм защиты системы



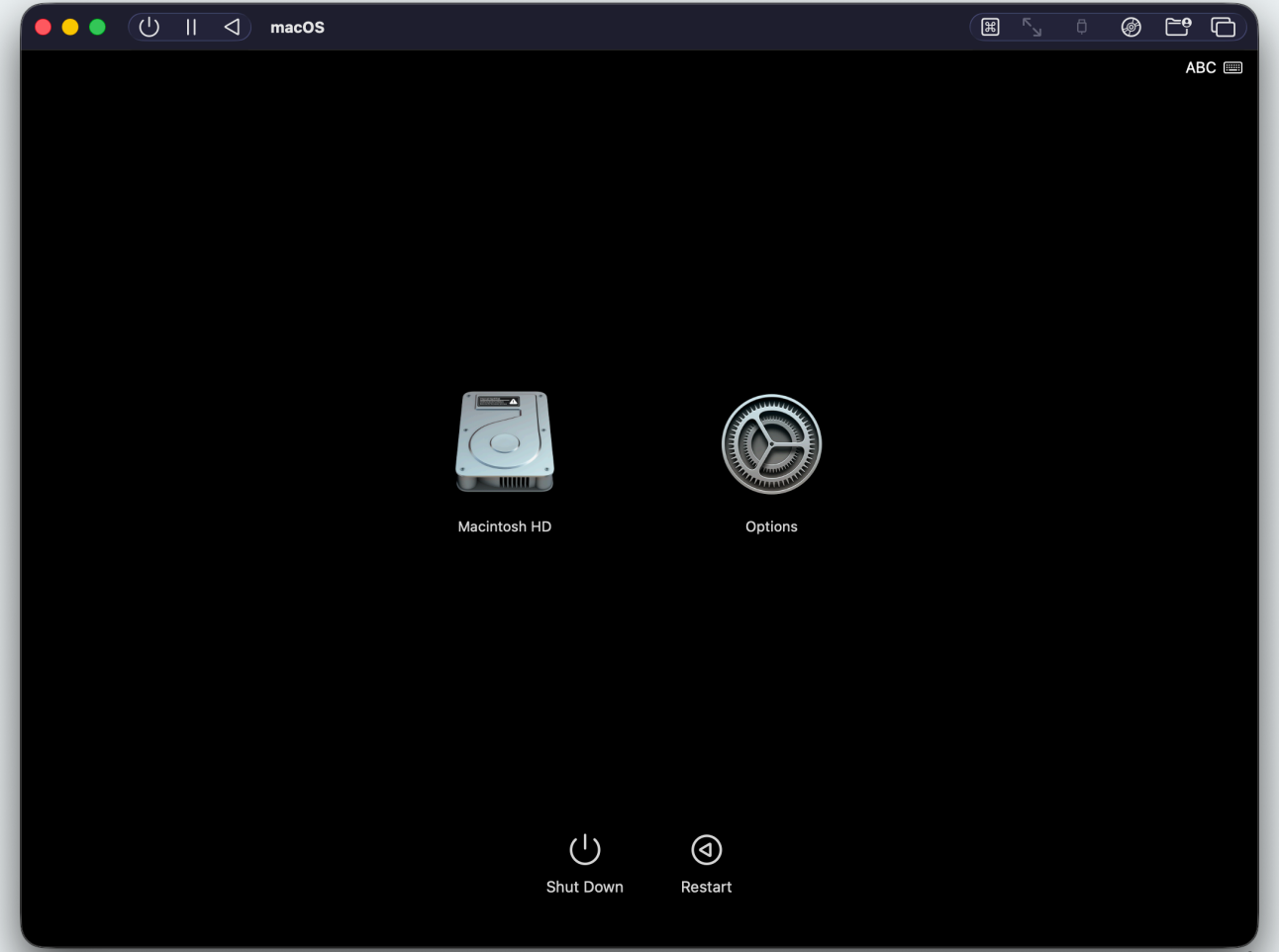
System Integrity Protection (SIP)

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



System Integrity Protection (SIP)

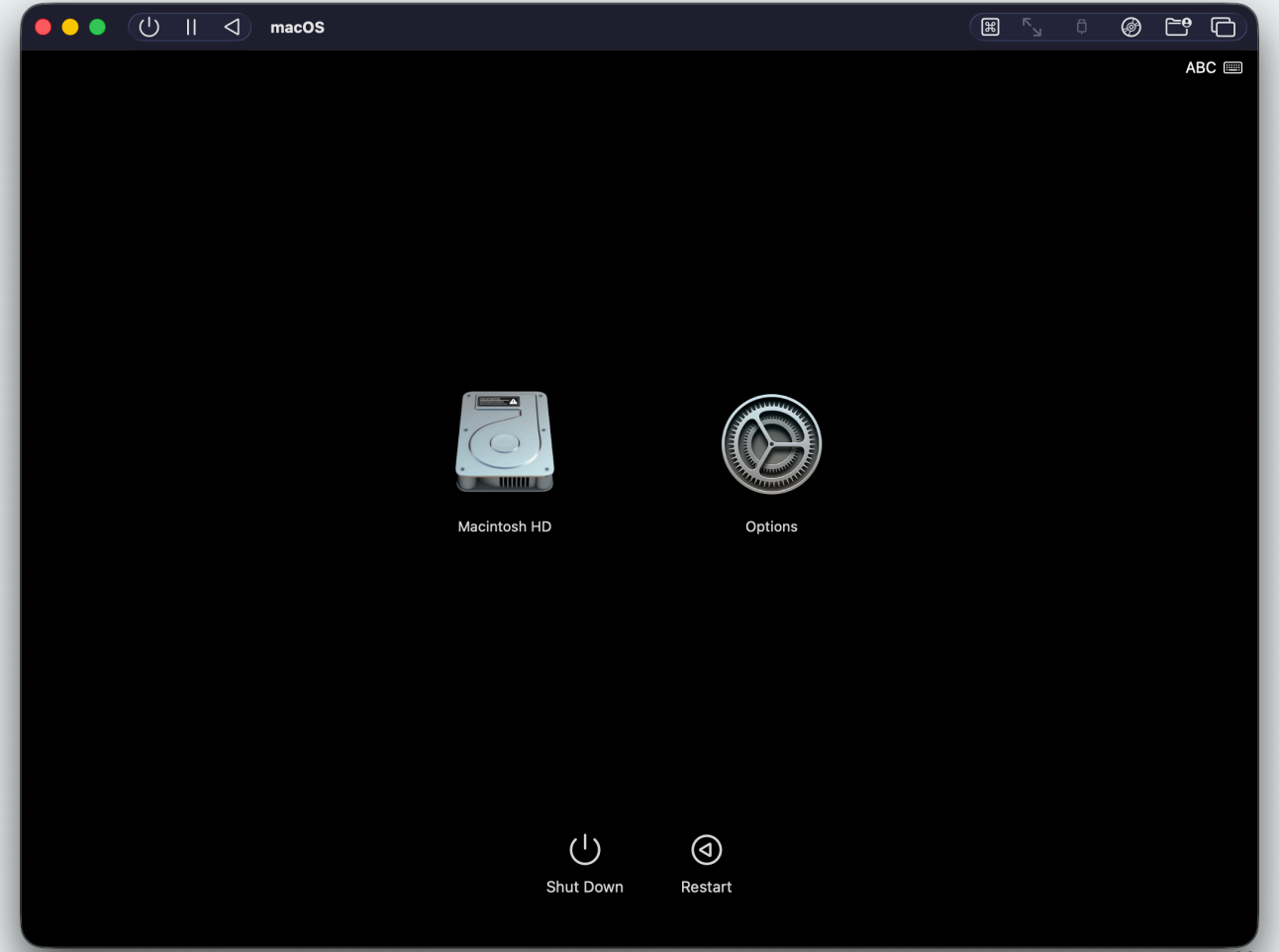
1. Механизм защиты системы
2. Ограничивает доступ к важным компонентам
3. Предотвращает отладку и инъекцию в чужие процессы



System Integrity Protection (SIP)

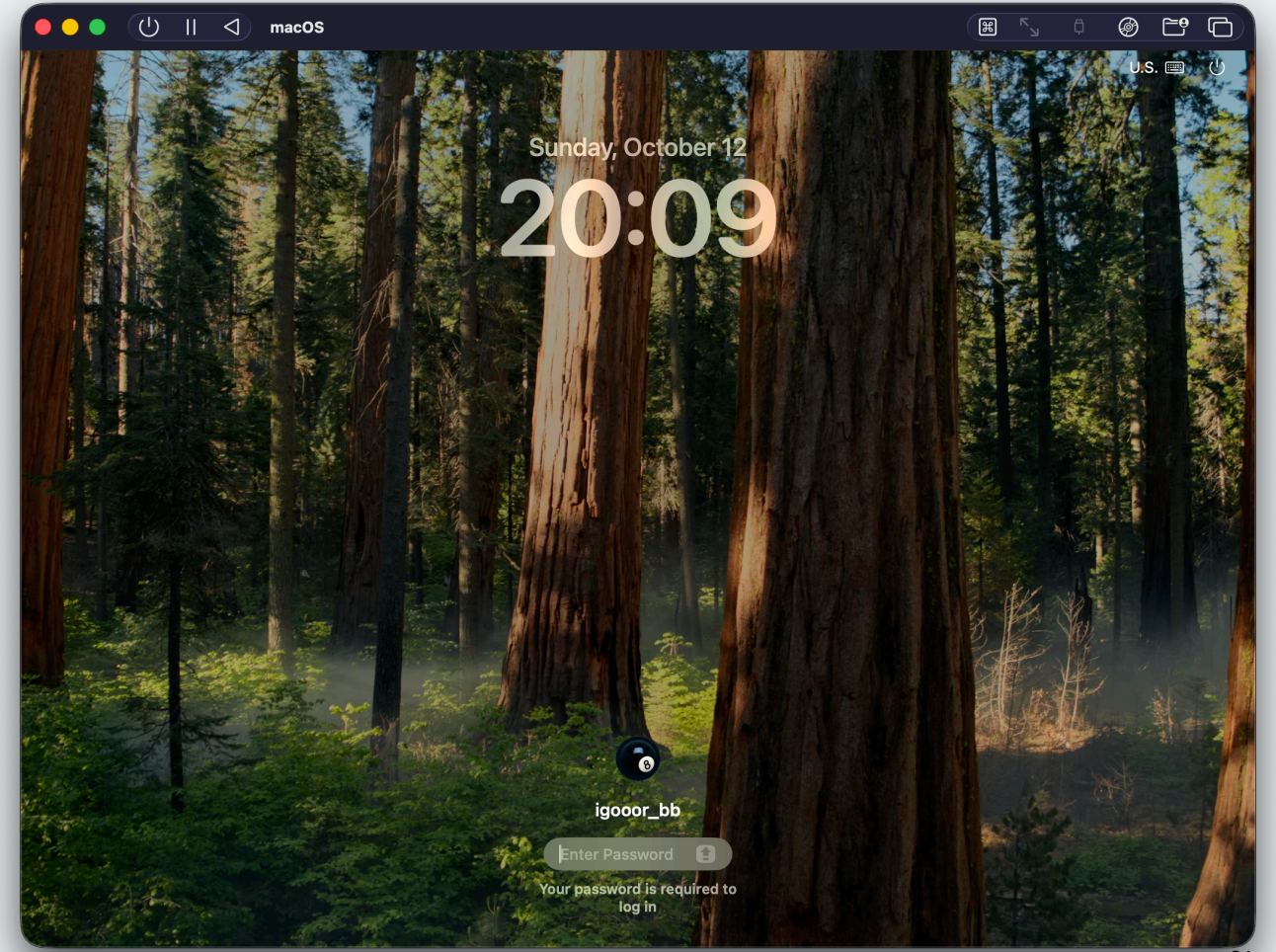
1. Механизм защиты системы
2. Ограничивает доступ к важным компонентам
3. Предотвращает отладку и инъекцию в чужие процессы

Отключается в Recovery Mode с помощью команды csrutil disable



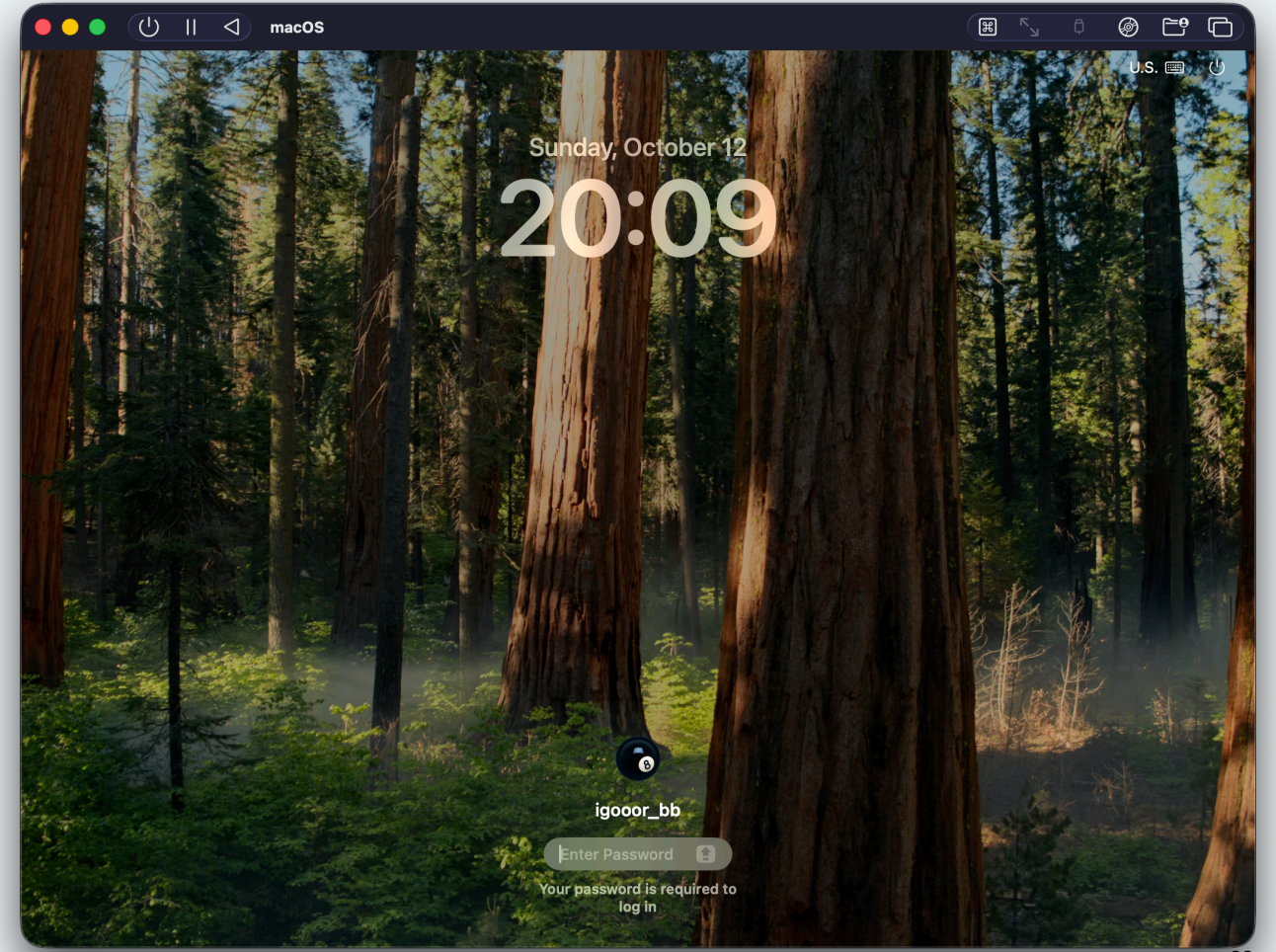
Виртуализация

1 Отдельная среда для исследований



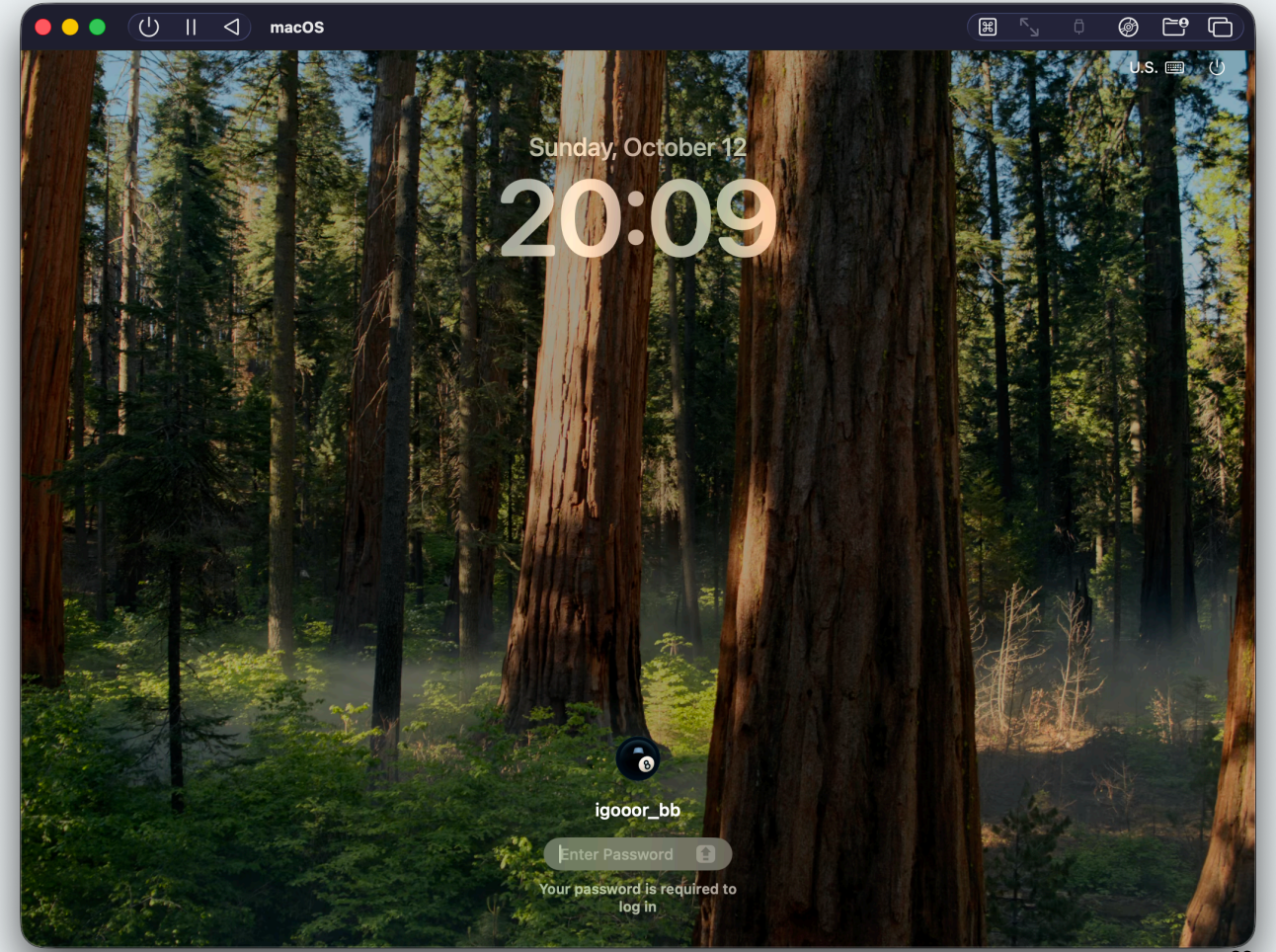
Виртуализация

1. Отдельная среда для исследований
2. Безопасные эксперименты



Виртуализация

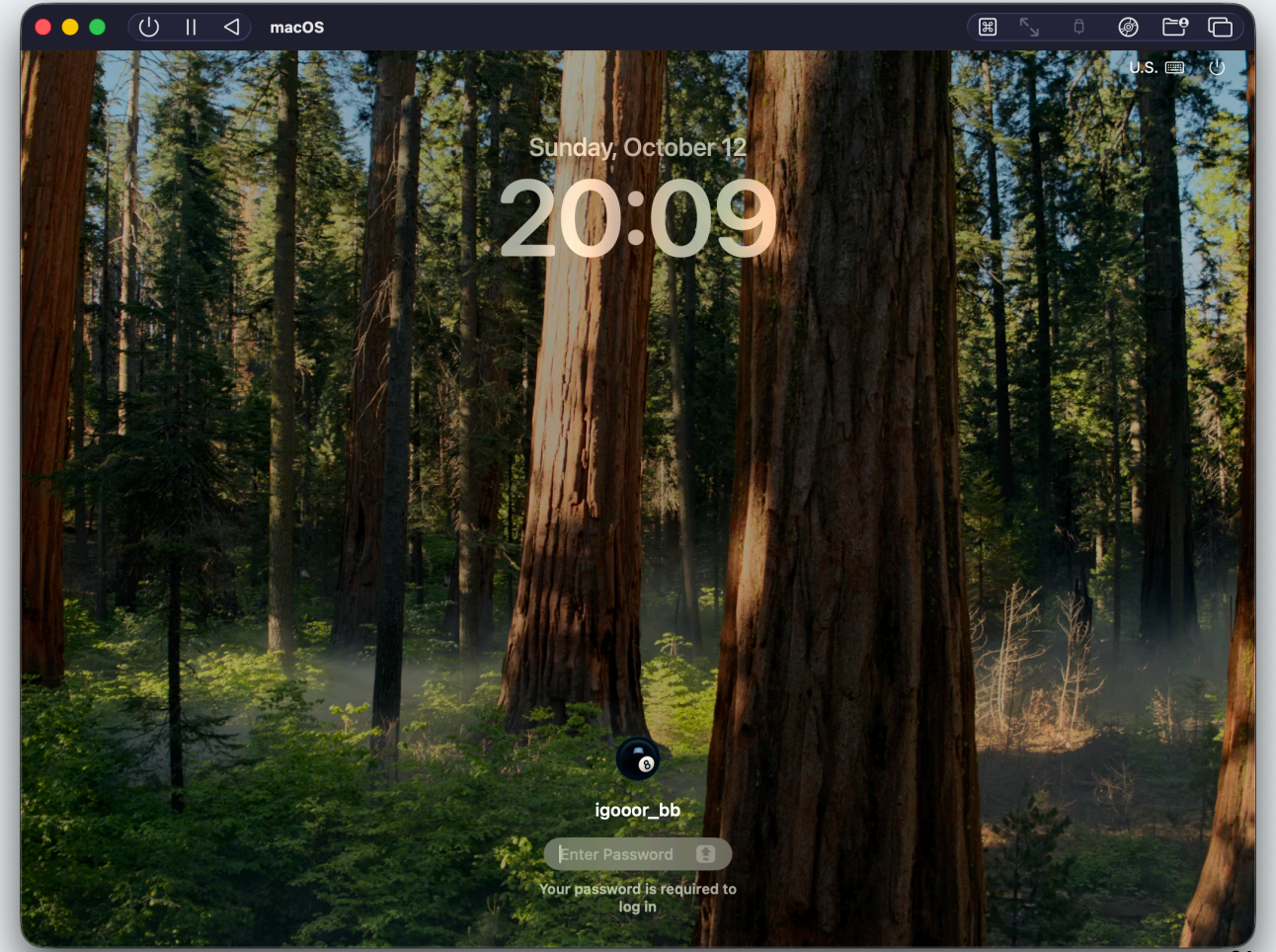
1. Отдельная среда для исследований
2. Безопасные эксперименты
3. Воспроизводимость окружения

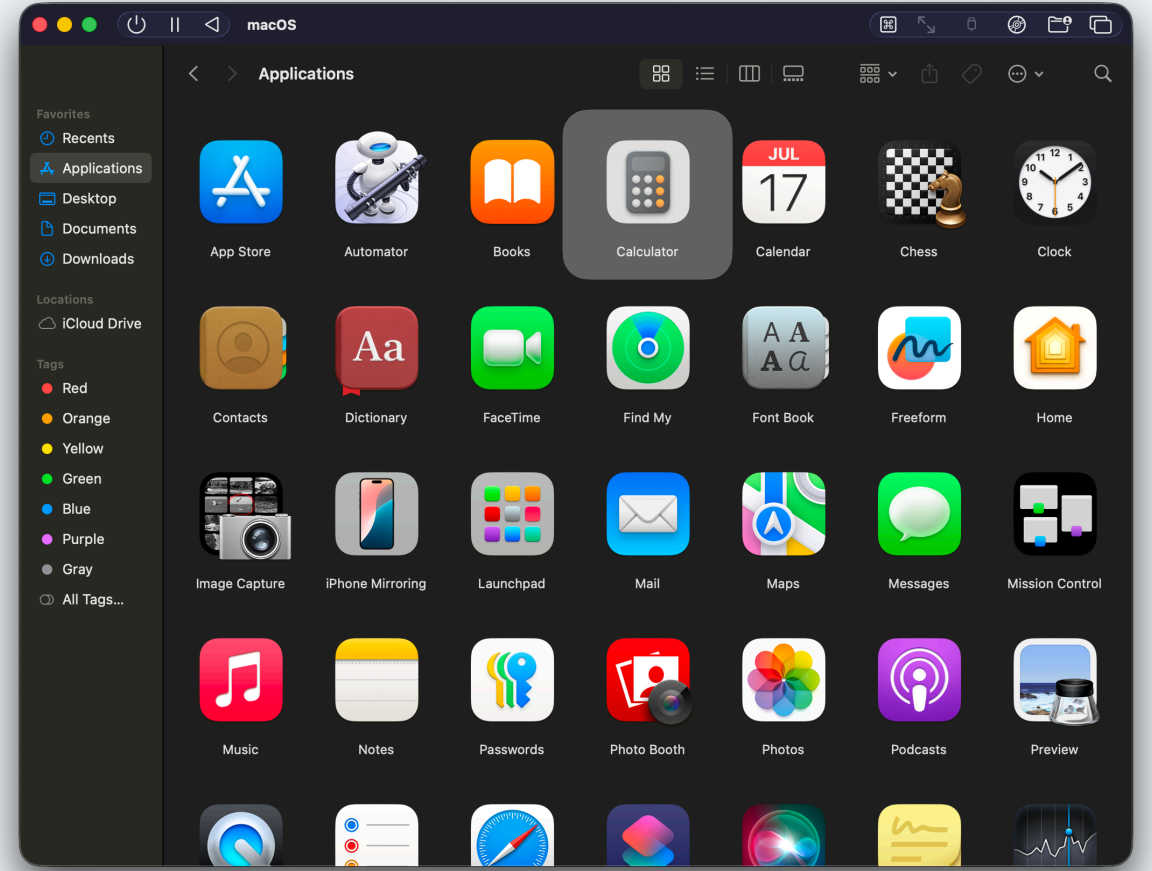


Виртуализация

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

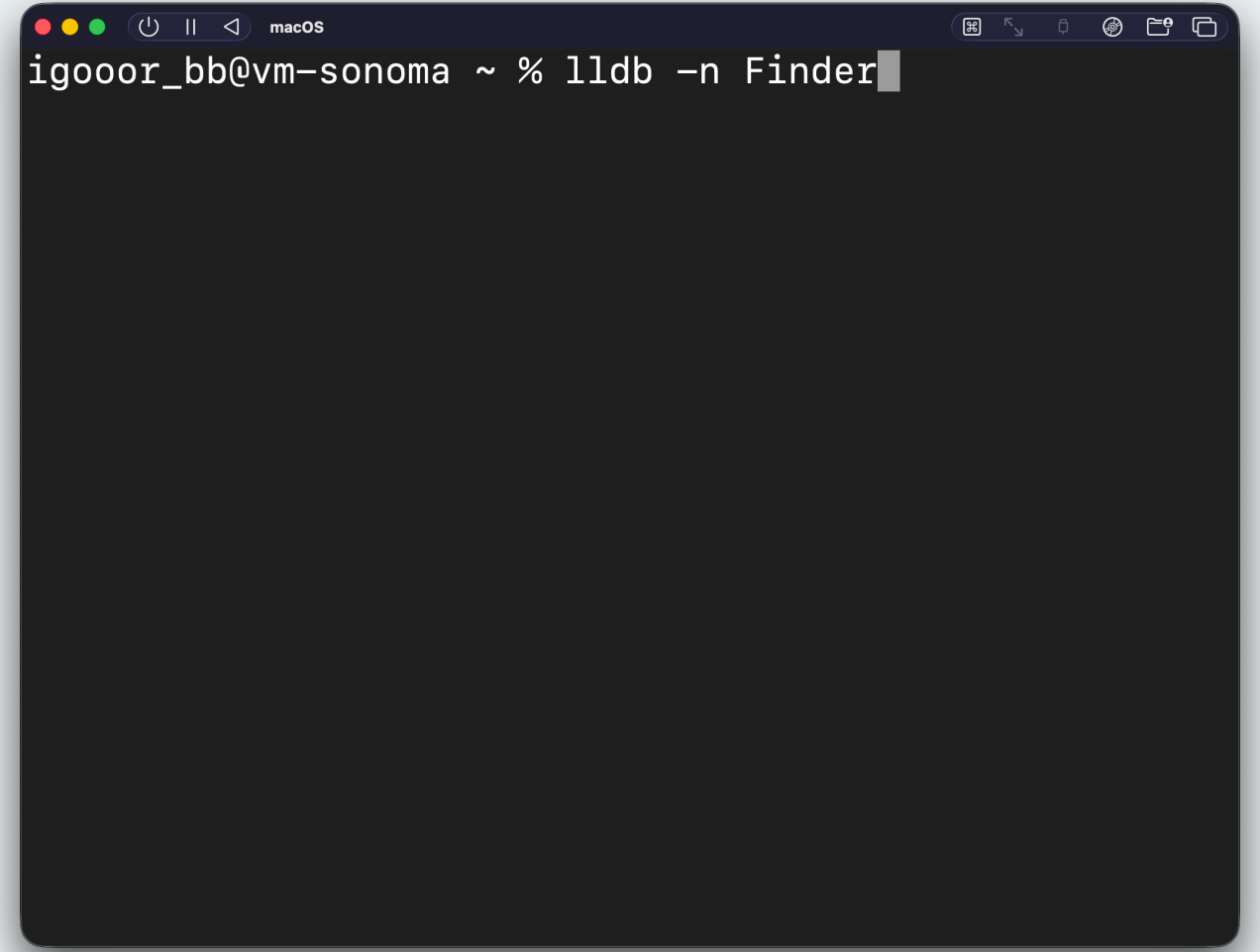
Далее все действия будут происходить внутри виртуальной машины




macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17

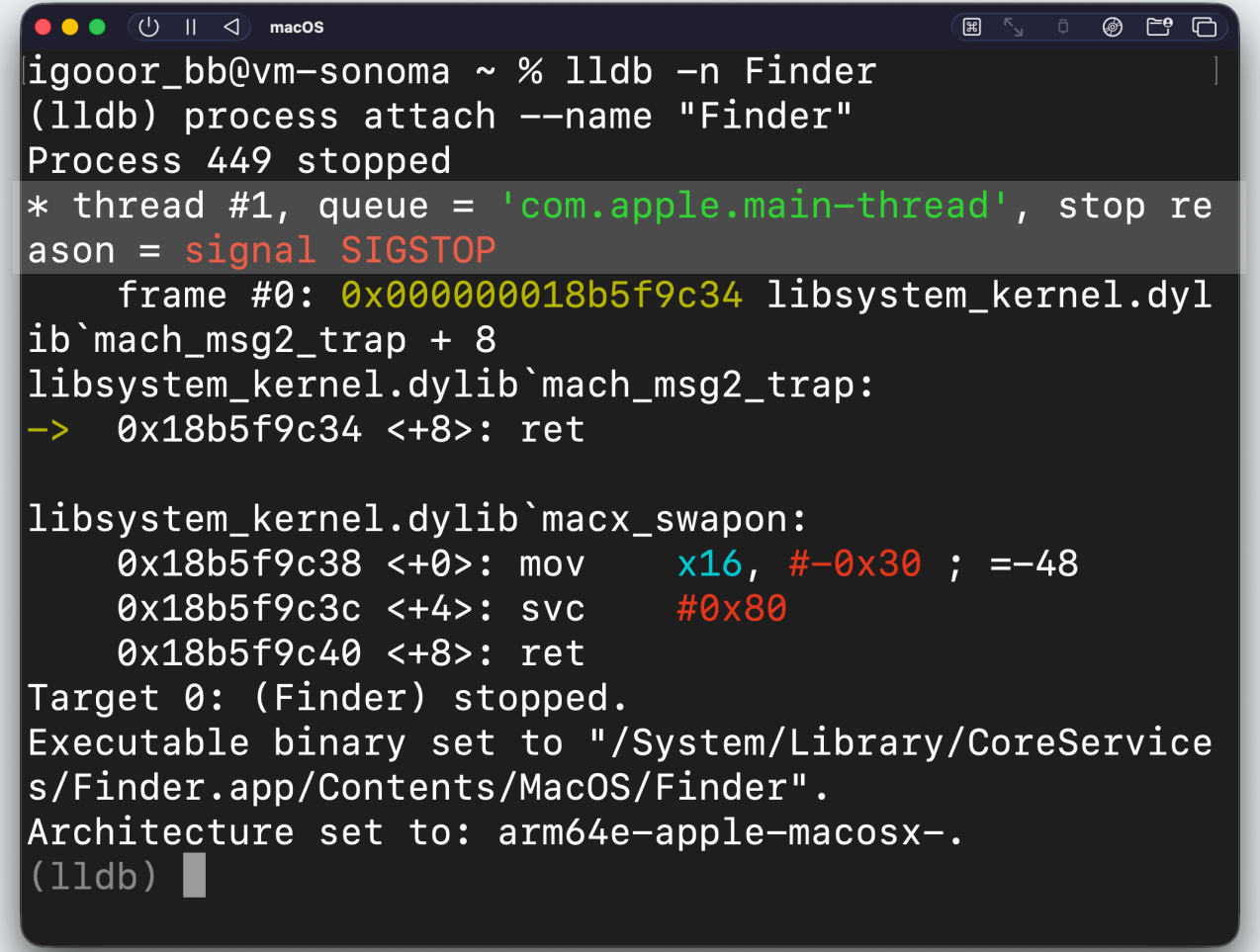
Исследование

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



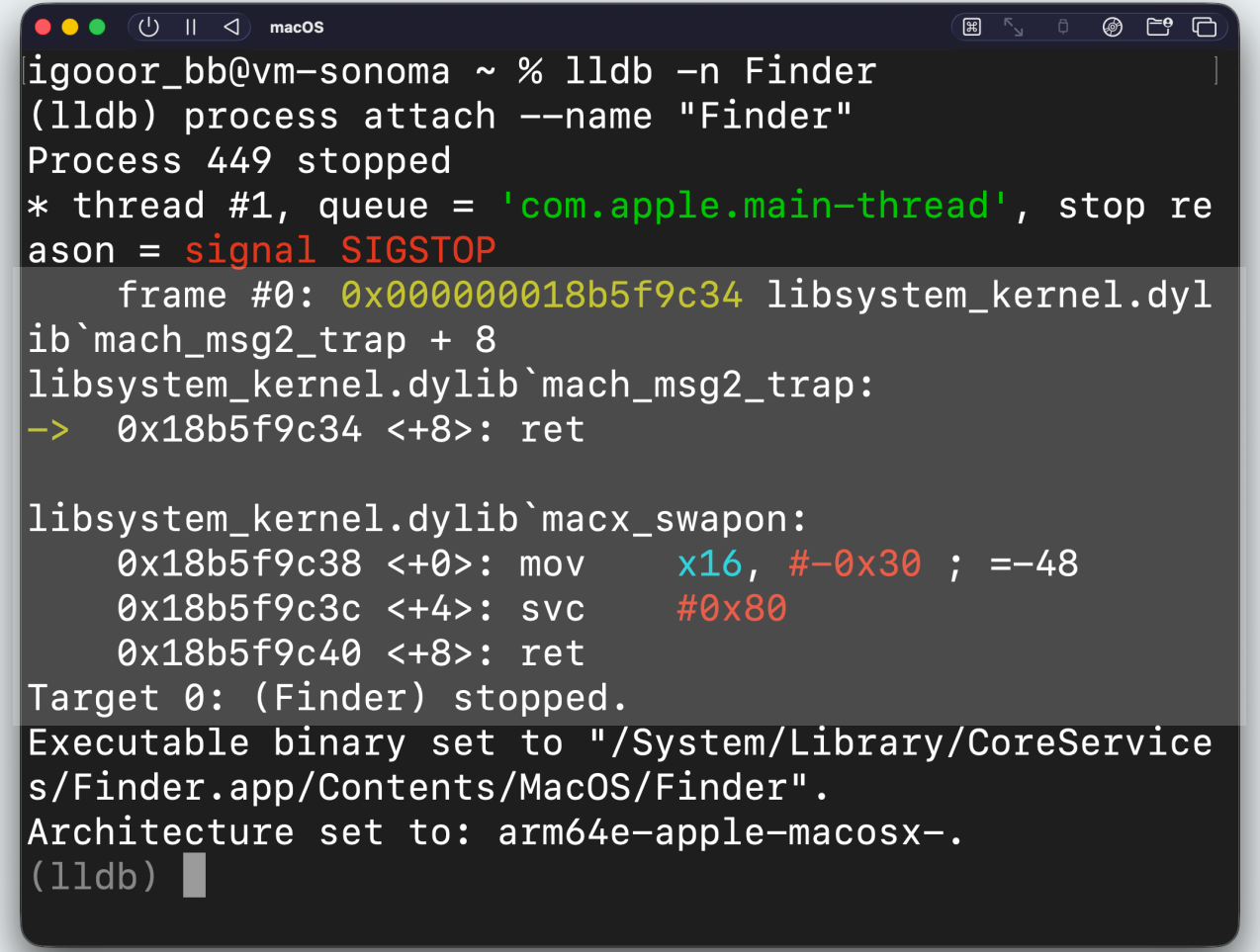
Исследование

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



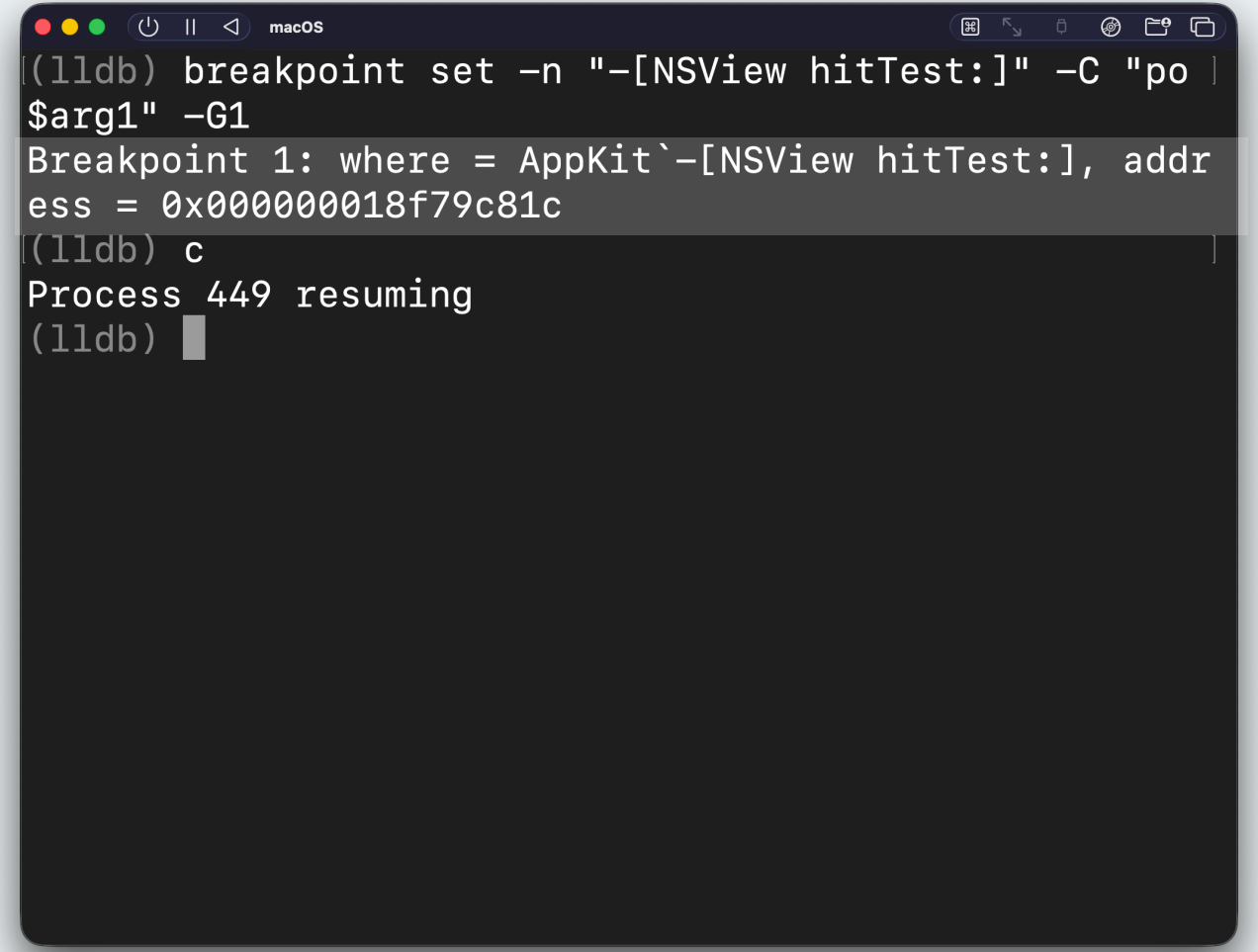
Исследование

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

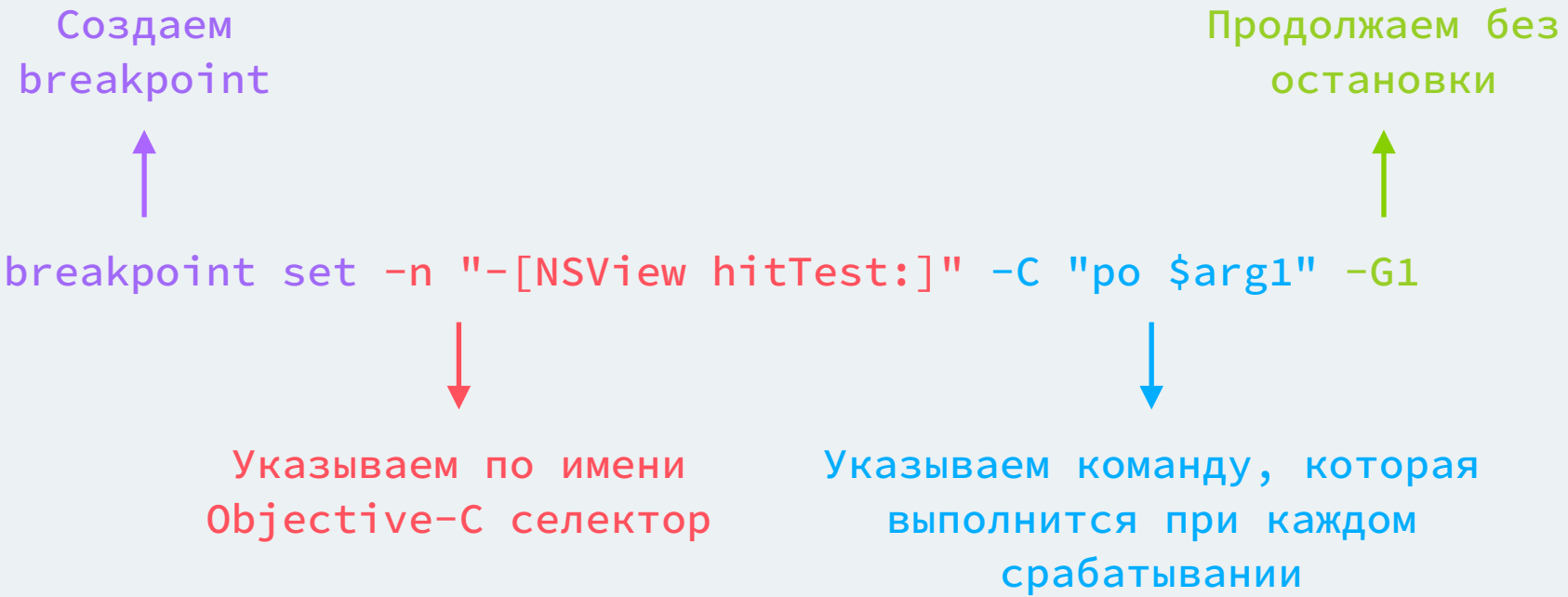


Исследование

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

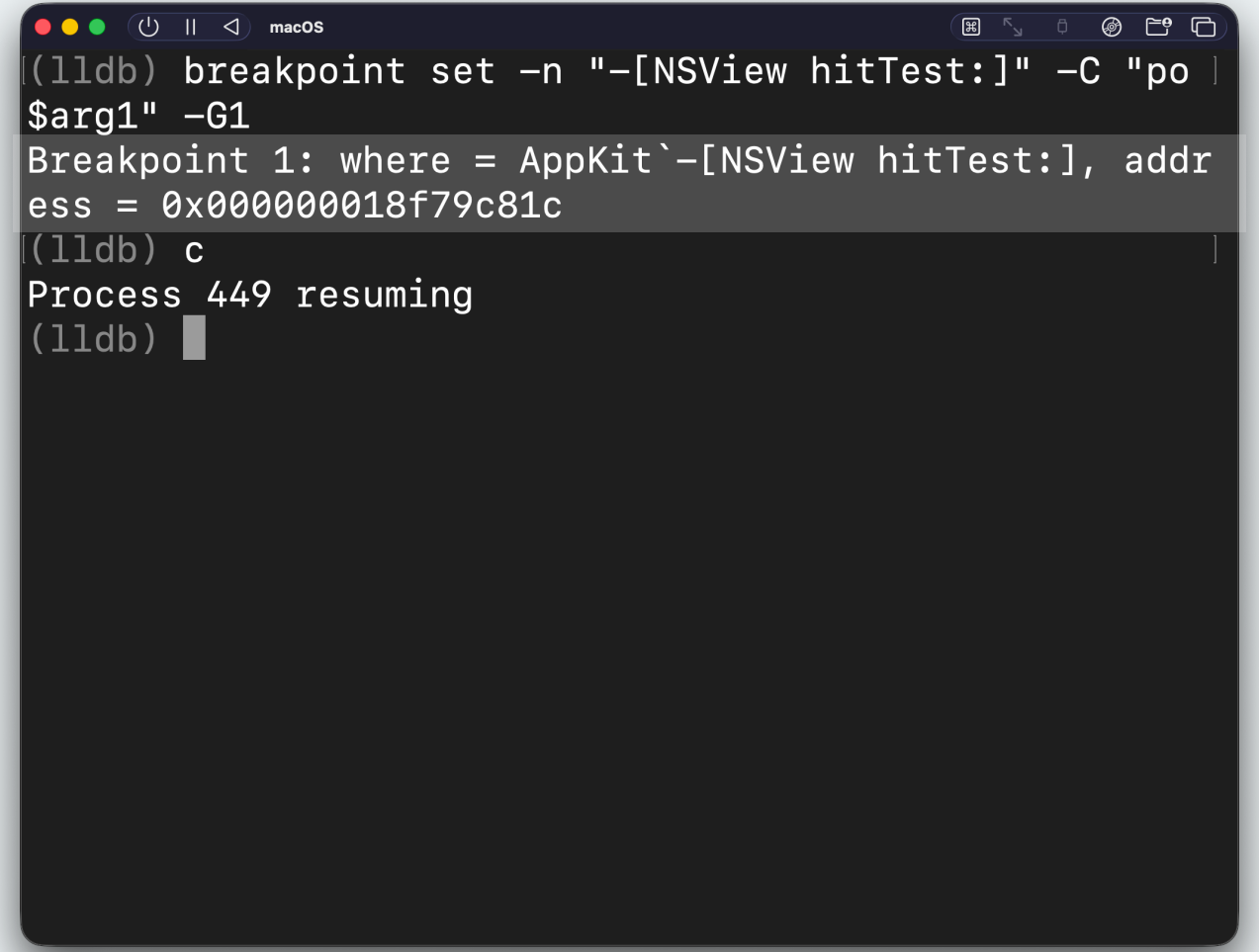


```
breakpoint set -n "-[NSView hitTest:]" -C "po $arg1" -G1
```


Создаем breakpoint
Продолжаем без остановки
breakpoint set -n "-[NSView hitTest:]" -C "po $arg1" -G1
Указываем по имени Objective-C селектор
Указываем команду, которая выполнится при каждом срабатывании

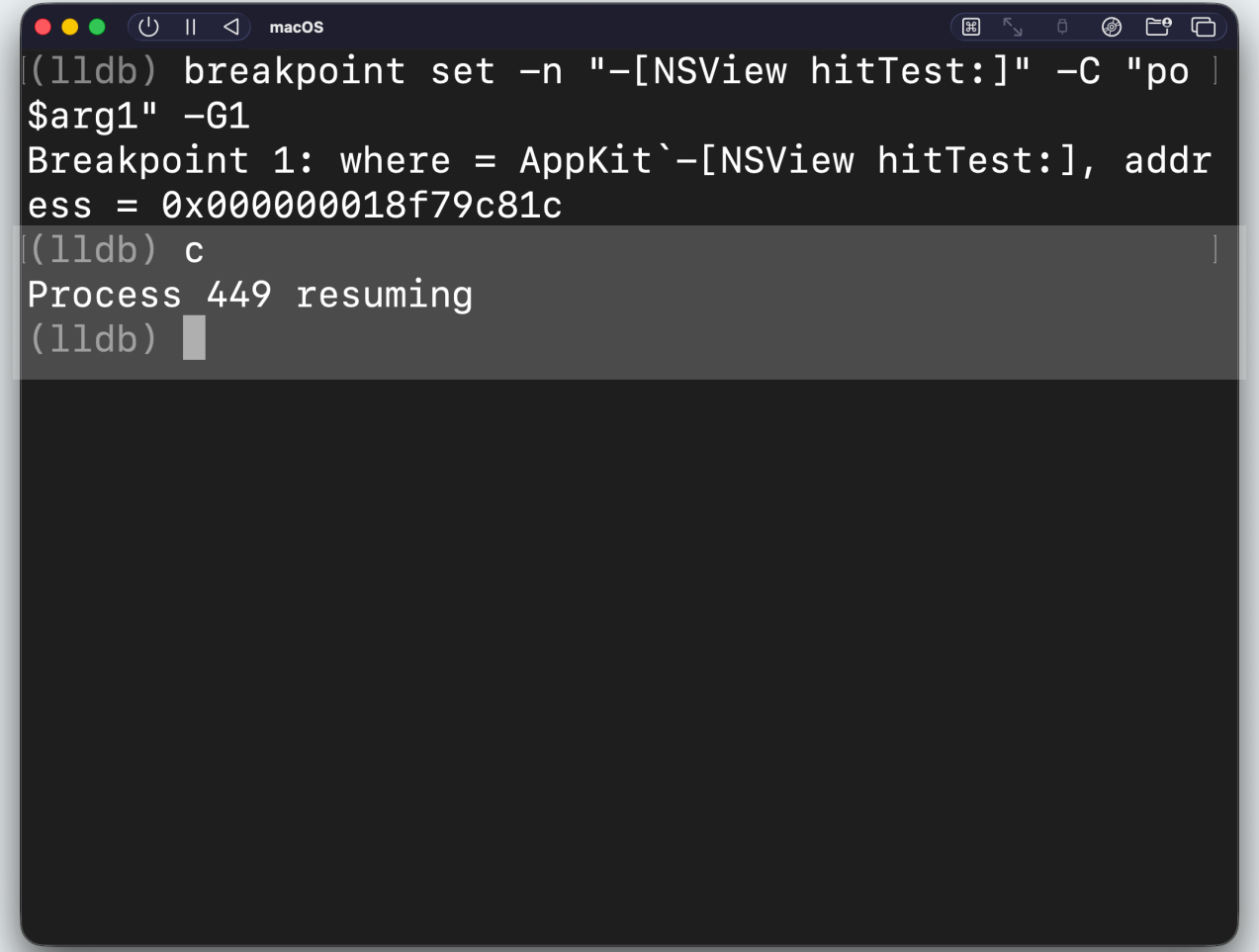
Исследование

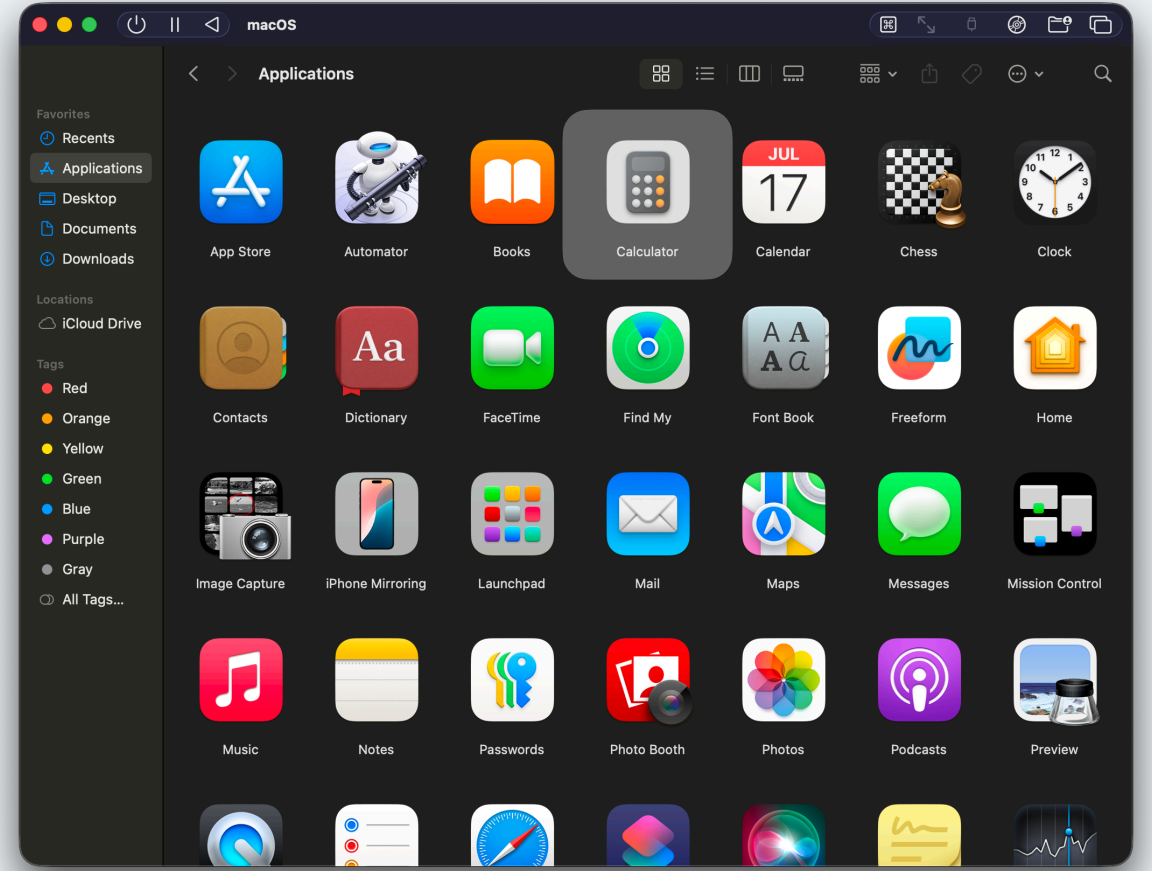
1. Можем делать точки остановки
2. Можем использовать **hitTest**, чтобы найти нужный объект



Исследование

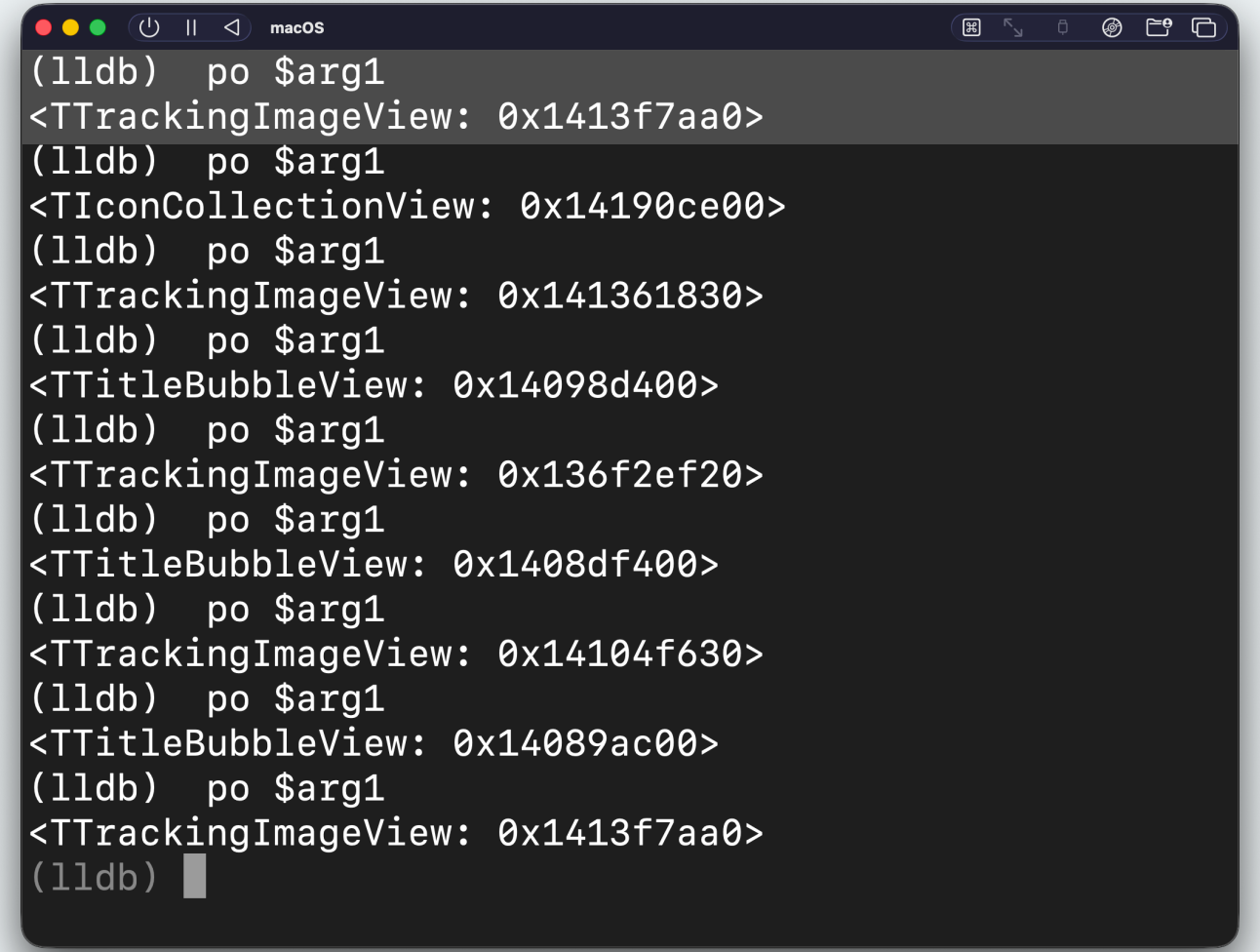
1. Можем делать точки остановки
2. Можем использовать **hitTest**, чтобы найти нужный объект




macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17

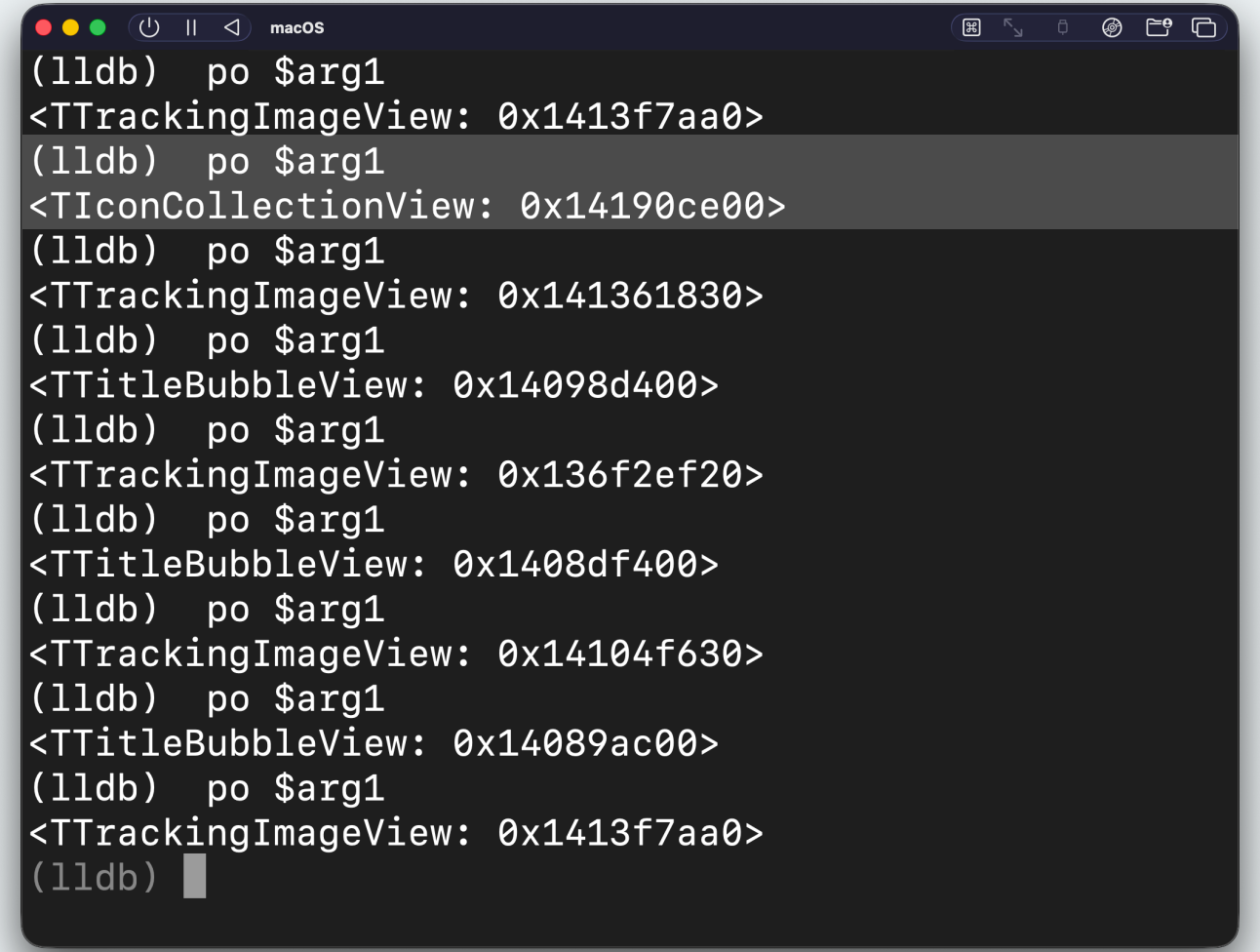
Исследование

1 Обход выполняется по принципу **Responder Chain**



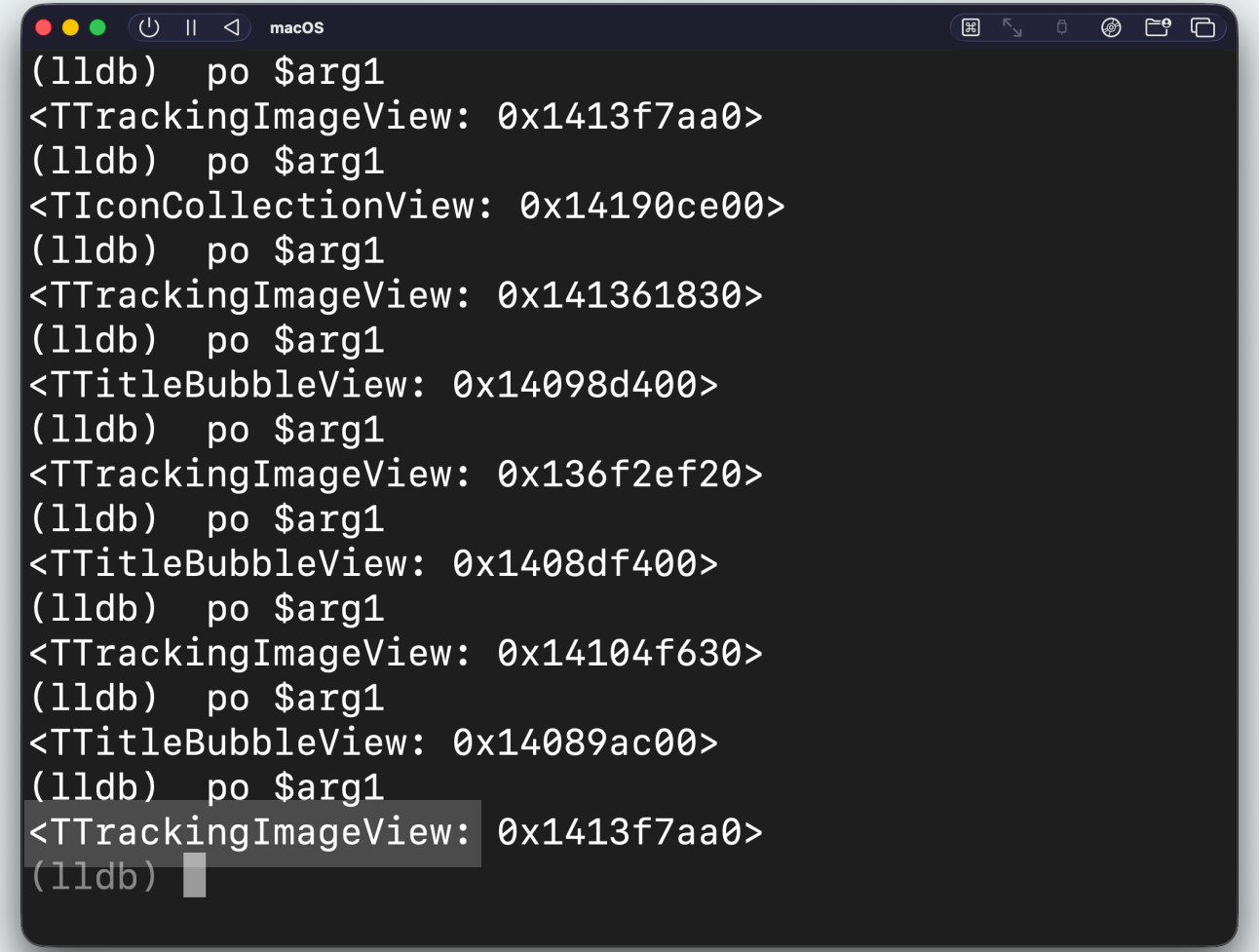
Исследование

1 Обход выполняется по принципу **Responder Chain**



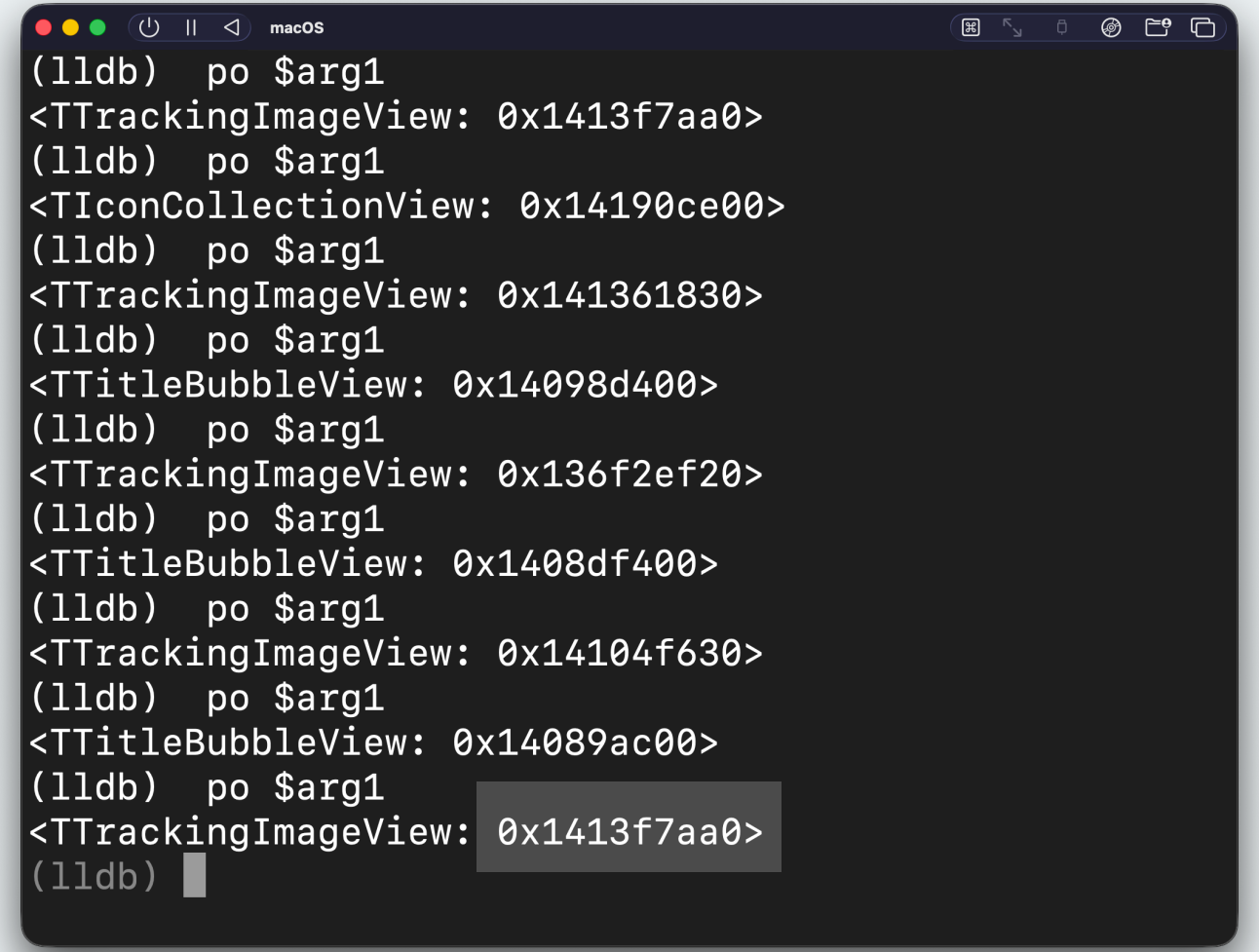
Исследование

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



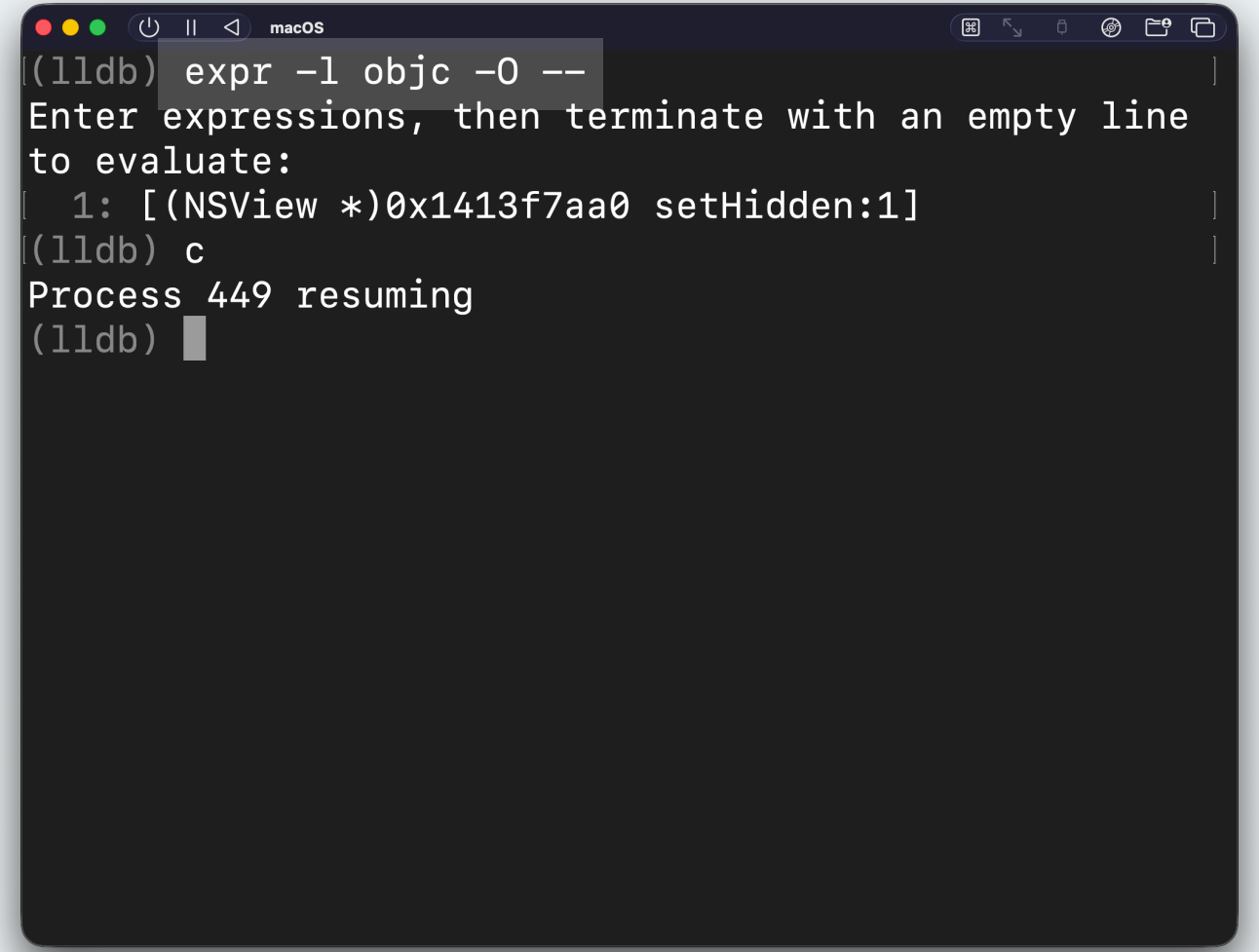
Исследование

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



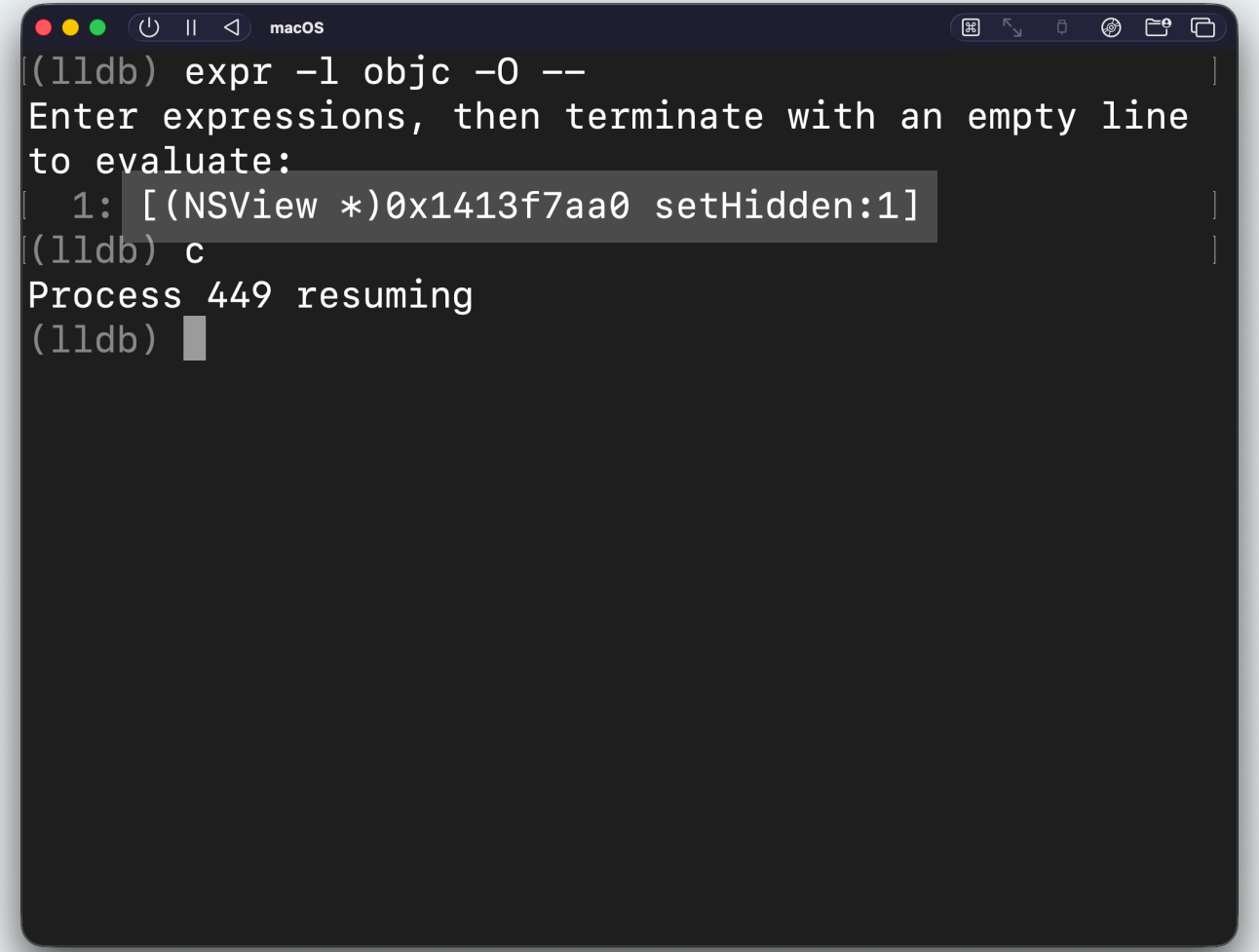
Выражения

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



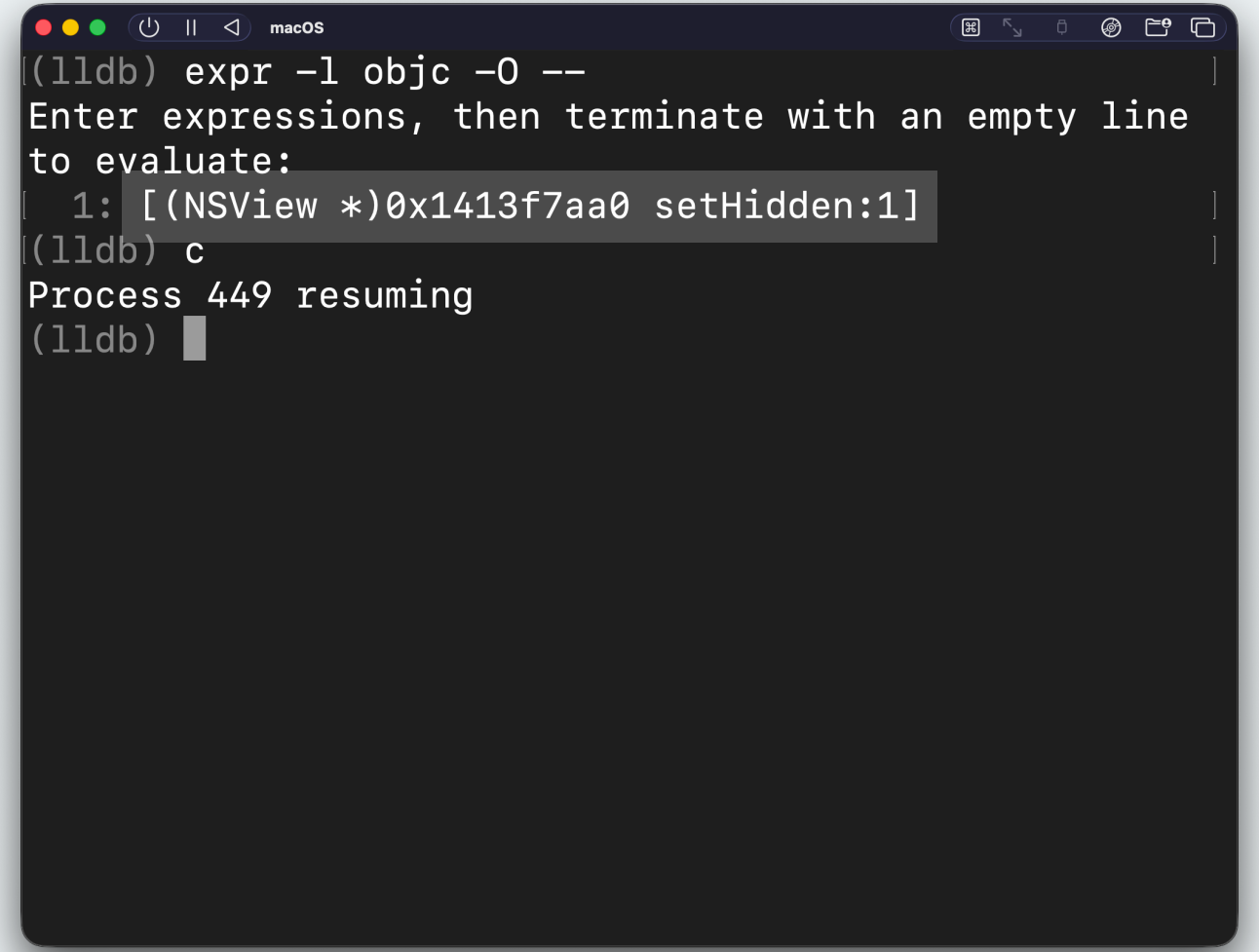
Выражения

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



Выражения

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



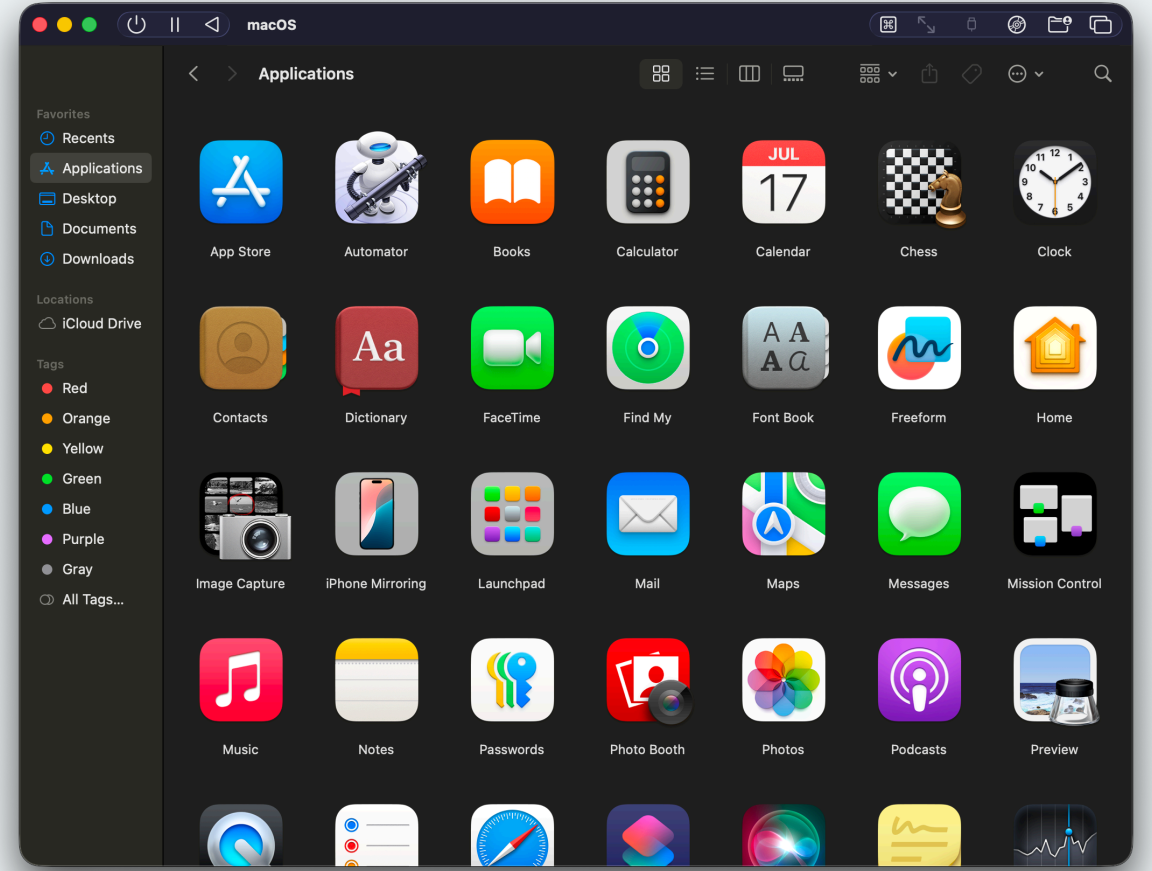
```
expr -l objc -O -- [(NSView *)0x1413f7aa0 setHidden:1]
```

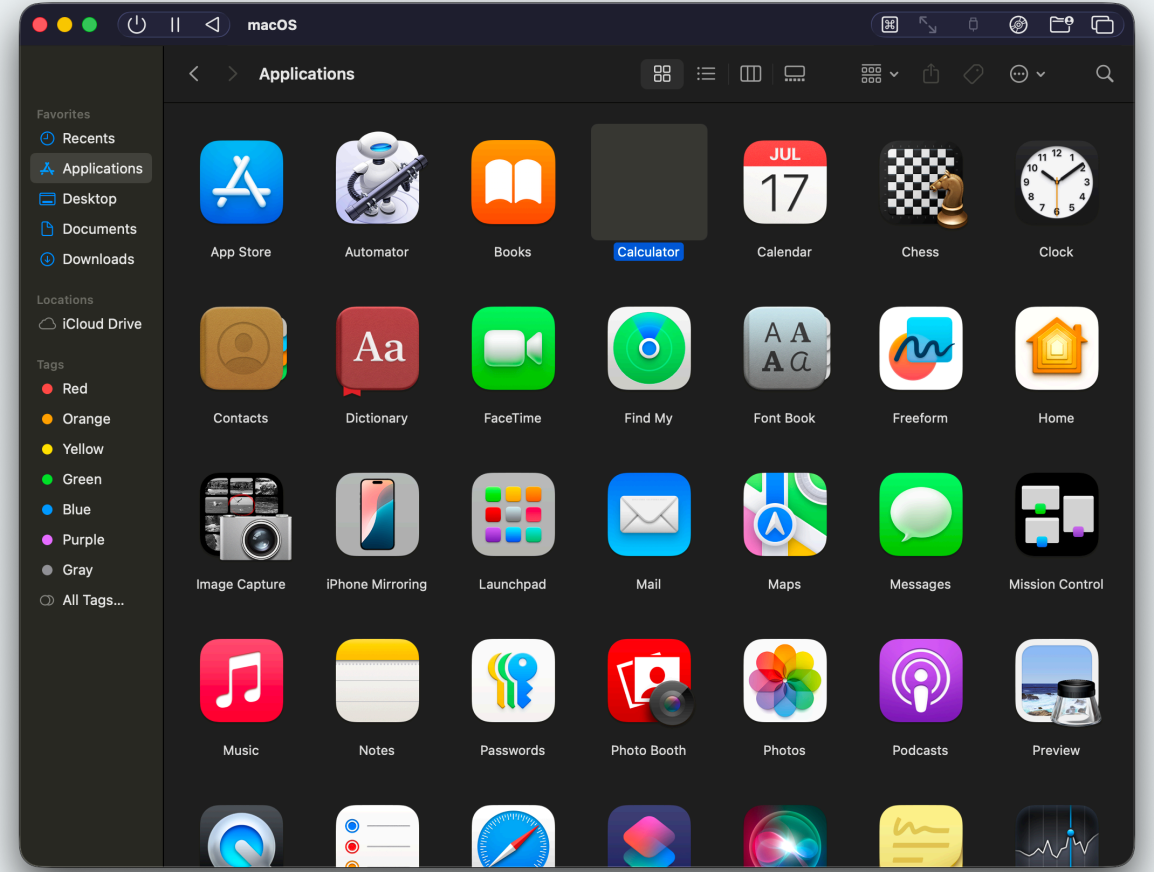
Выполняем Objective-C выражение

Обращаемся к объекту как к NSView и меняем его свойство

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

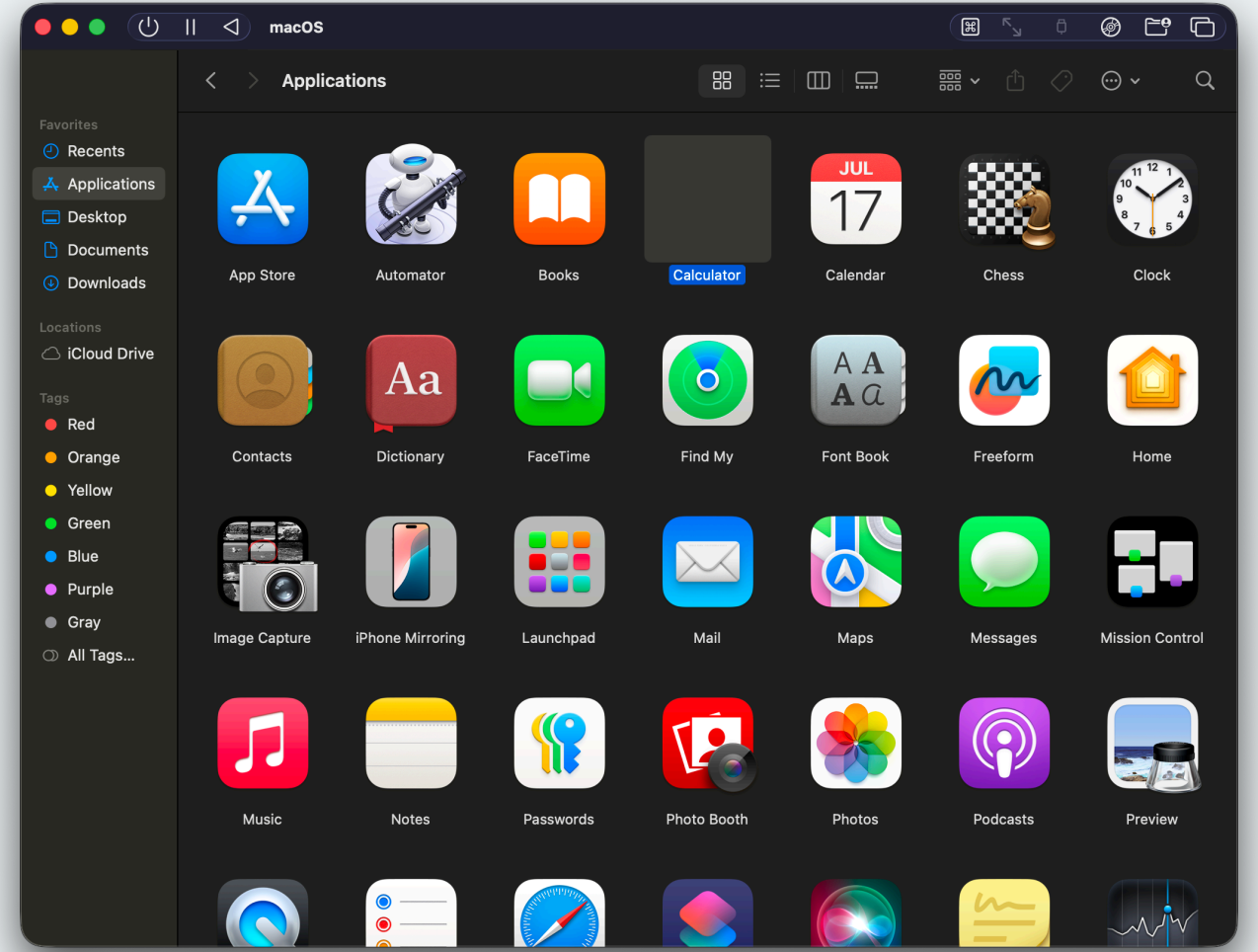
Печатаем описание объекта на выходе (удобство)


macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17


macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview

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

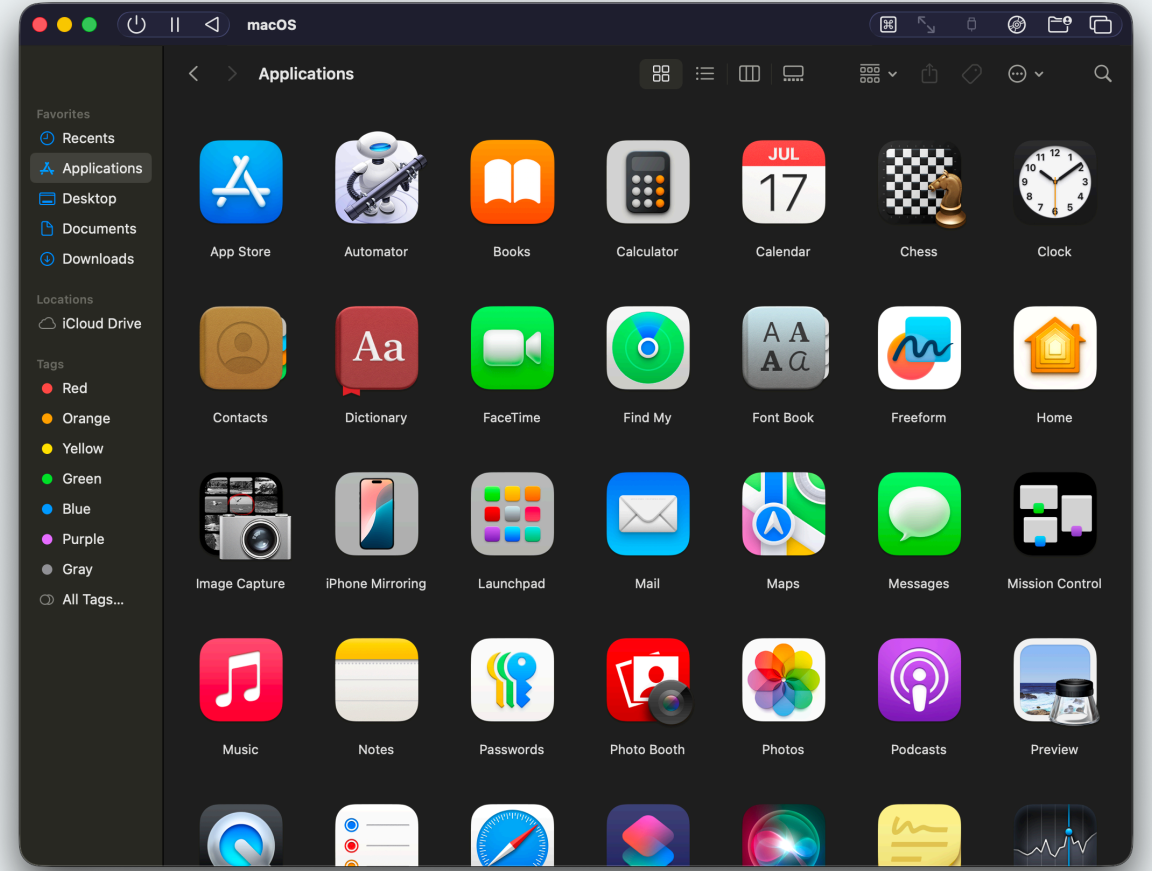
1. Можем писать более сложные выражения
2. Иногда нужно добавлять импорты для стандартных типов

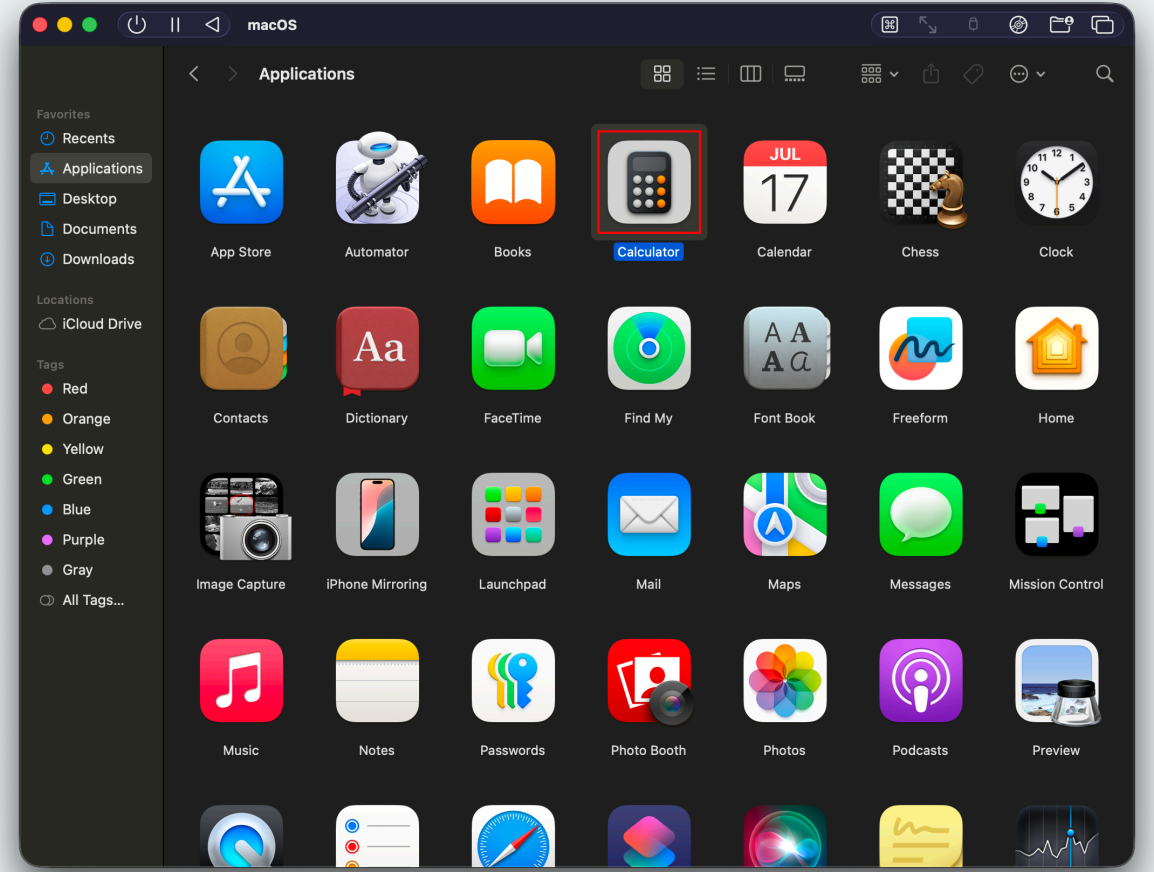
```
@import AppKit;

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];
```


macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17


macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17

```
@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

```
# ~/.lldbinit

command alias pv expr -l objc -O -- [self view]
```

Настройки объявляются в файле ~/.lldbinit

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 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

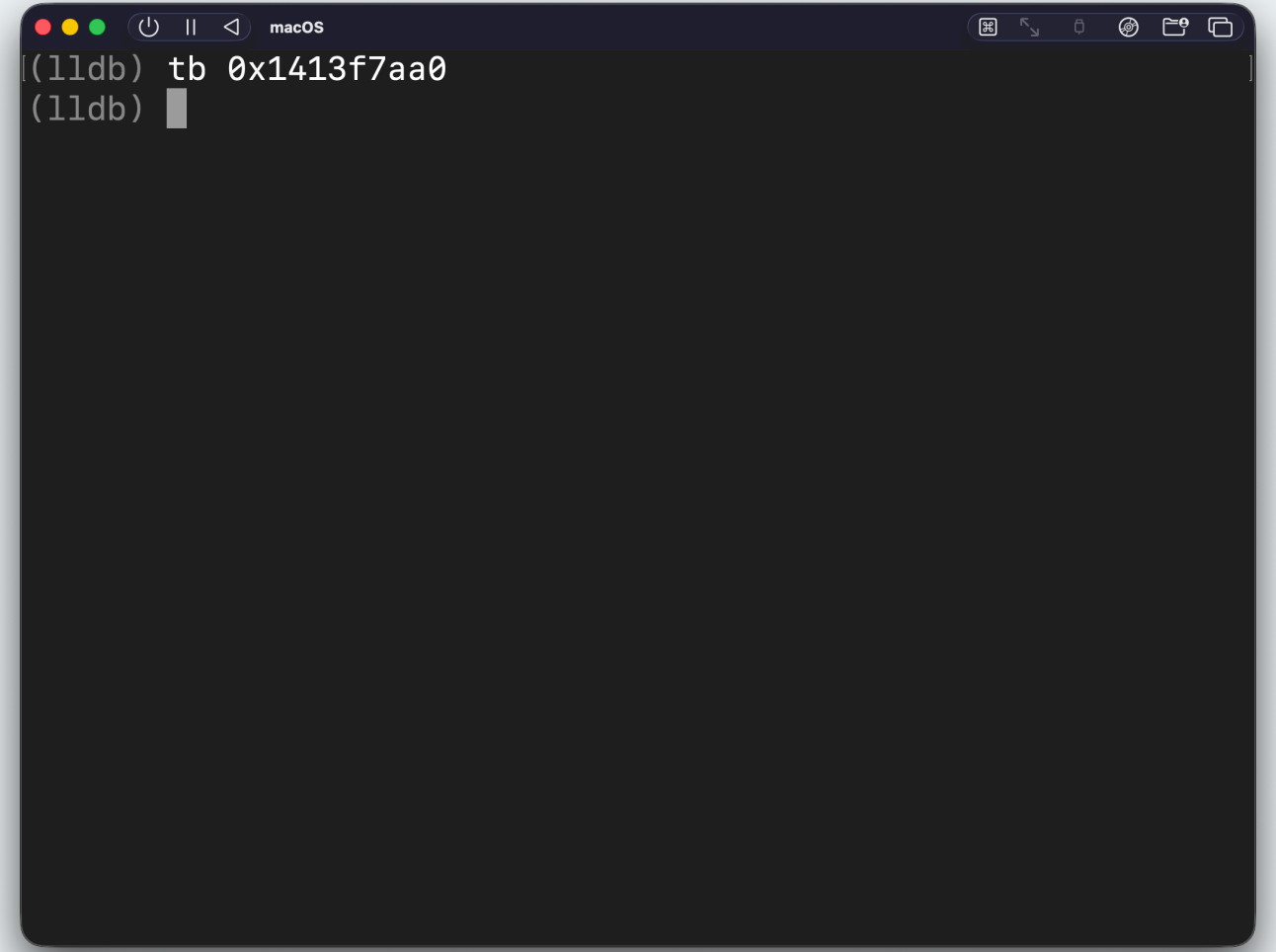
Настройки объявляются в файле ~/.lldbinit

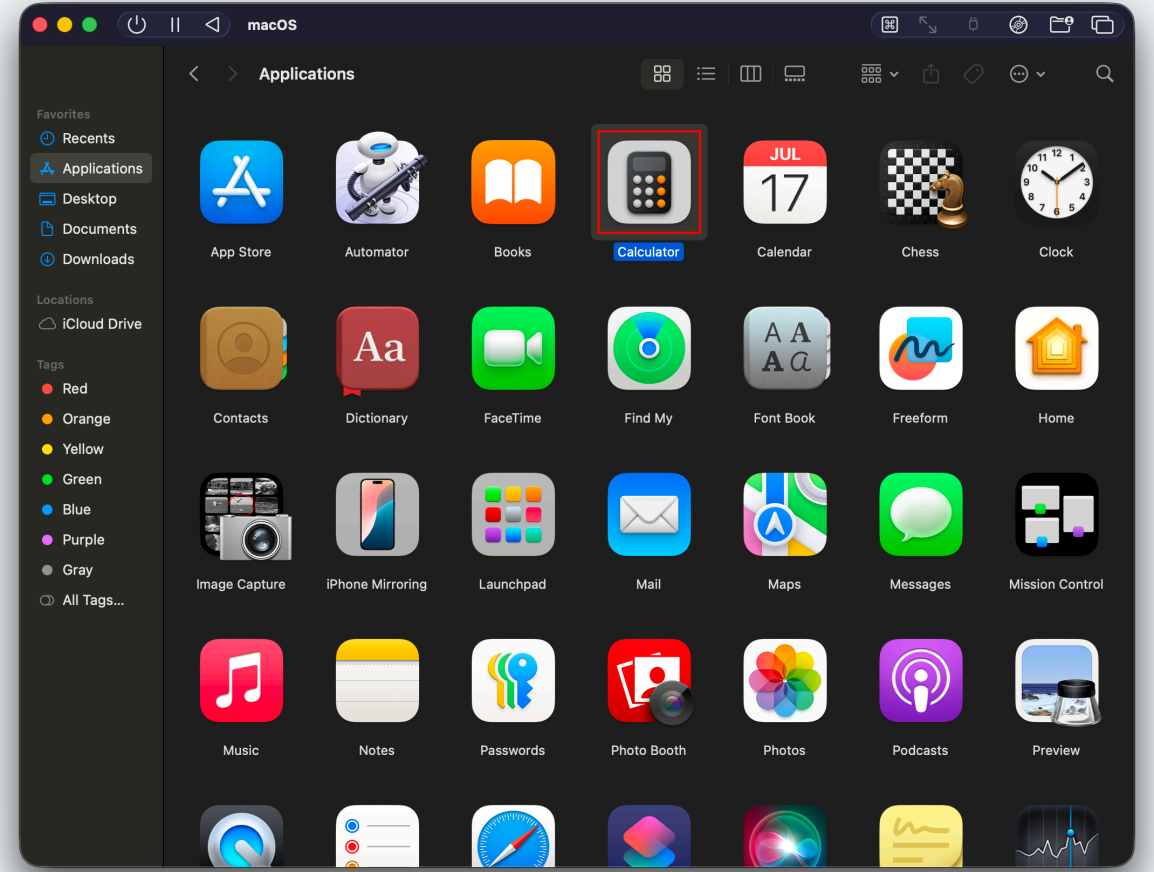
```
# ~/.lldbinit

command regex tb 's/(.+)/expr -l objc -O --
@import 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




macOS
Applications
Favorites
Recents
Applications
Desktop
Documents
Downloads
Locations
iCloud Drive
Tags
Red
Orange
Yellow
Green
Blue
Purple
Gray
All Tags...
App Store
Automator
Books
Calculator
Calendar
Chess
Clock
Contacts
Dictionary
FaceTime
Find My
Font Book
Freeform
Home
Image Capture
iPhone Mirroring
Launchpad
Mail
Maps
Messages
Mission Control
Music
Notes
Passwords
Photo Booth
Photos
Podcasts
Preview
JUL
17

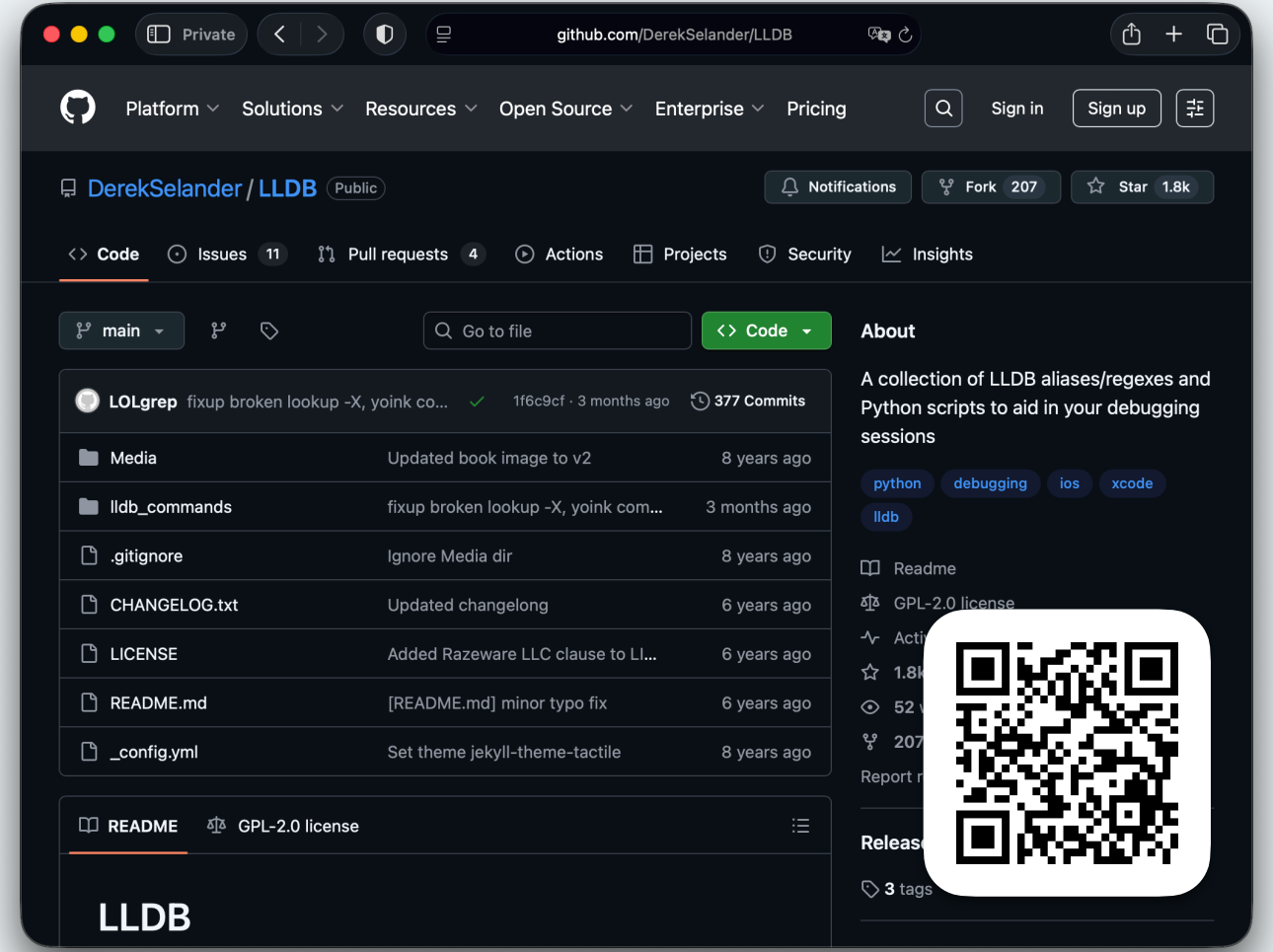
Команды

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



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, embeded `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
```

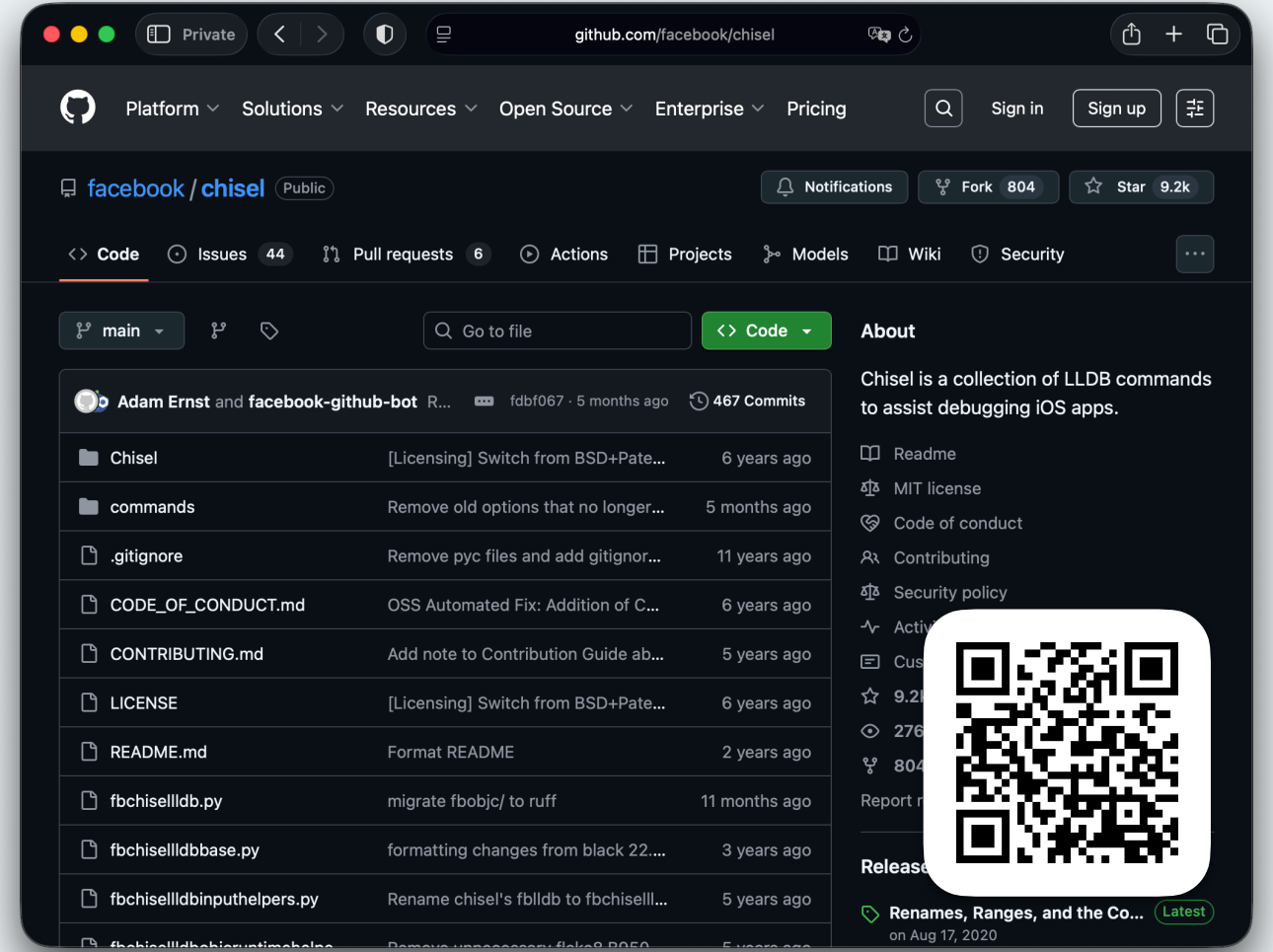
Источник: https://github.com/derekselander/lldb

Команды

Chisel

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

* Компания Meta признана экстремистской организацией и запрещена на территории РФ



Команды

Chisel

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

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

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

* Компания Meta признана экстремистской организацией и запрещена на территории РФ

Умеем исследовать любой процесс

Умеем изменять поведение

Не хотим каждый раз вызывать LLDB

```c
#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**

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

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

```c
#define _DARWIN_C_SOURCE

#include <stdlib.h>
#include <dlfcn.h>
#include <stdatomic.h>

typedef int (*rand_fn_t)(void);
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);
    }
}
```

```c
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);
    }
}
```

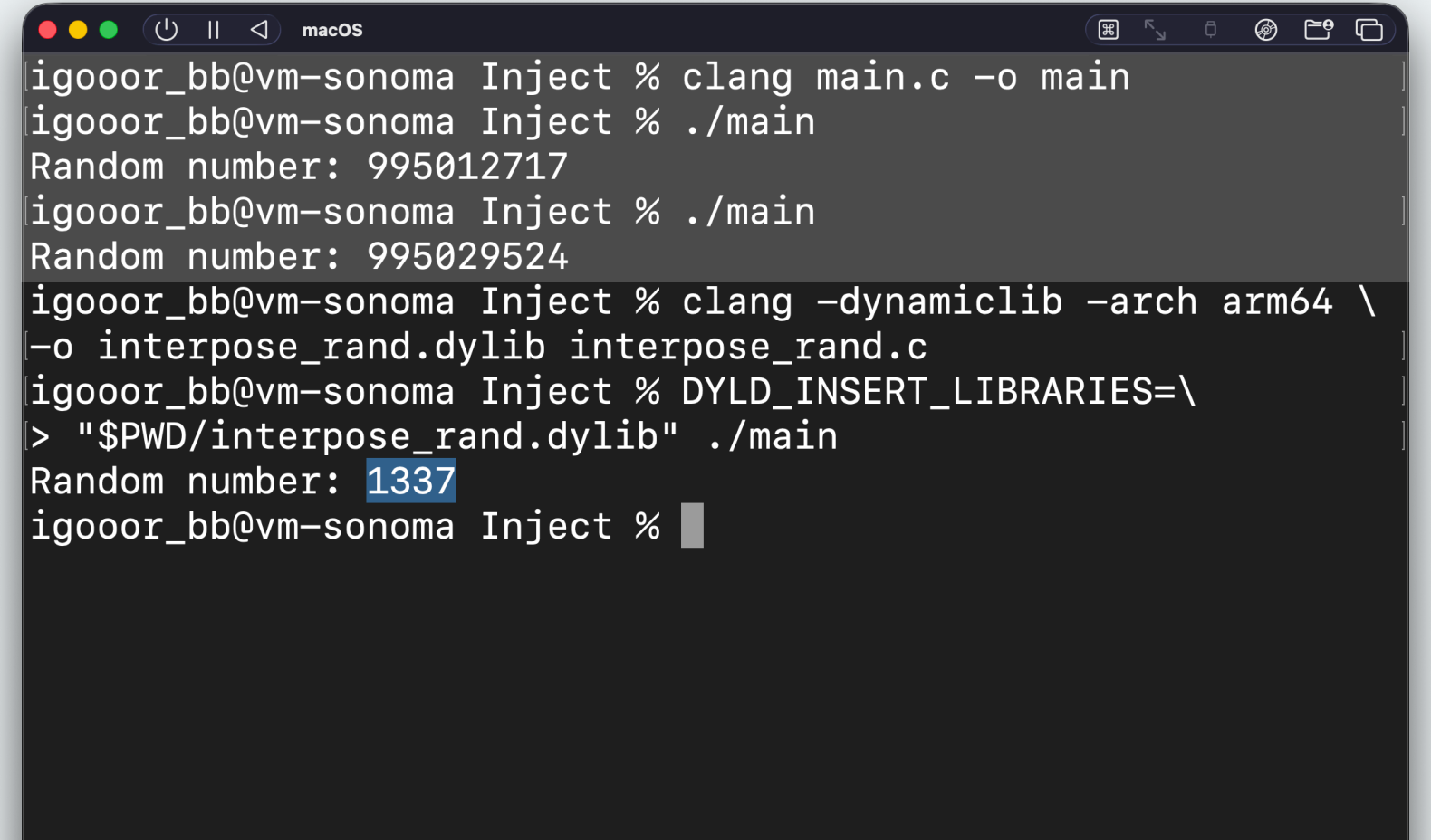
```c
static int custom_rand(void) {
    rand_fn_t orig_pt = atomic_load_explicit(&orig_rand_atomic,
                                             memory_order_acquire);
    if (!orig_pt) {
        orig_pt = (rand_fn_t)dlsym(RTLD_NEXT, "rand");
        atomic_store_explicit(&orig_rand_atomic, orig_pt,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

    return 1337;
}
```

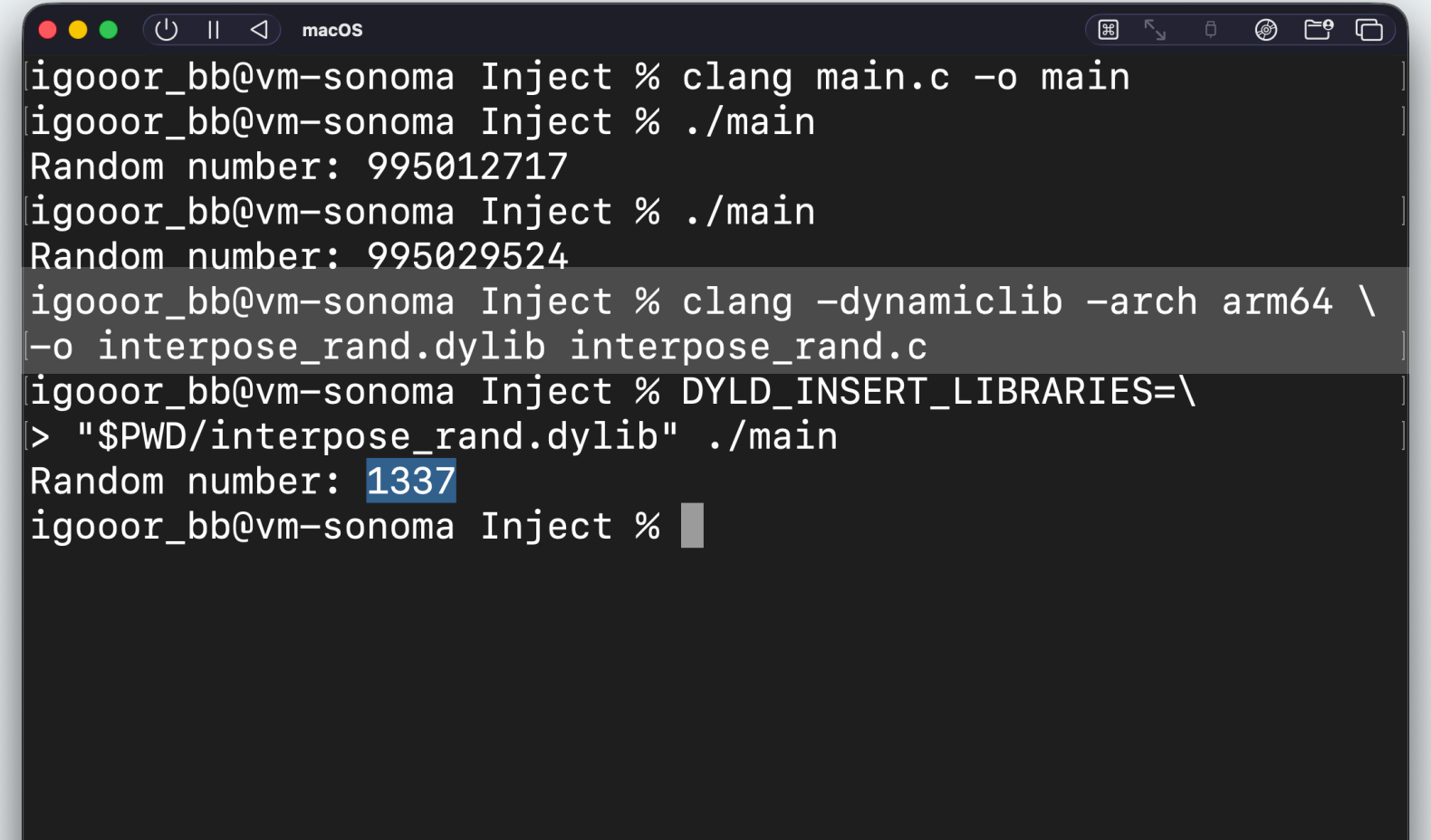
```c
static _Atomic(rand_fn_t) orig_rand_atomic = NULL;

__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;
}
```

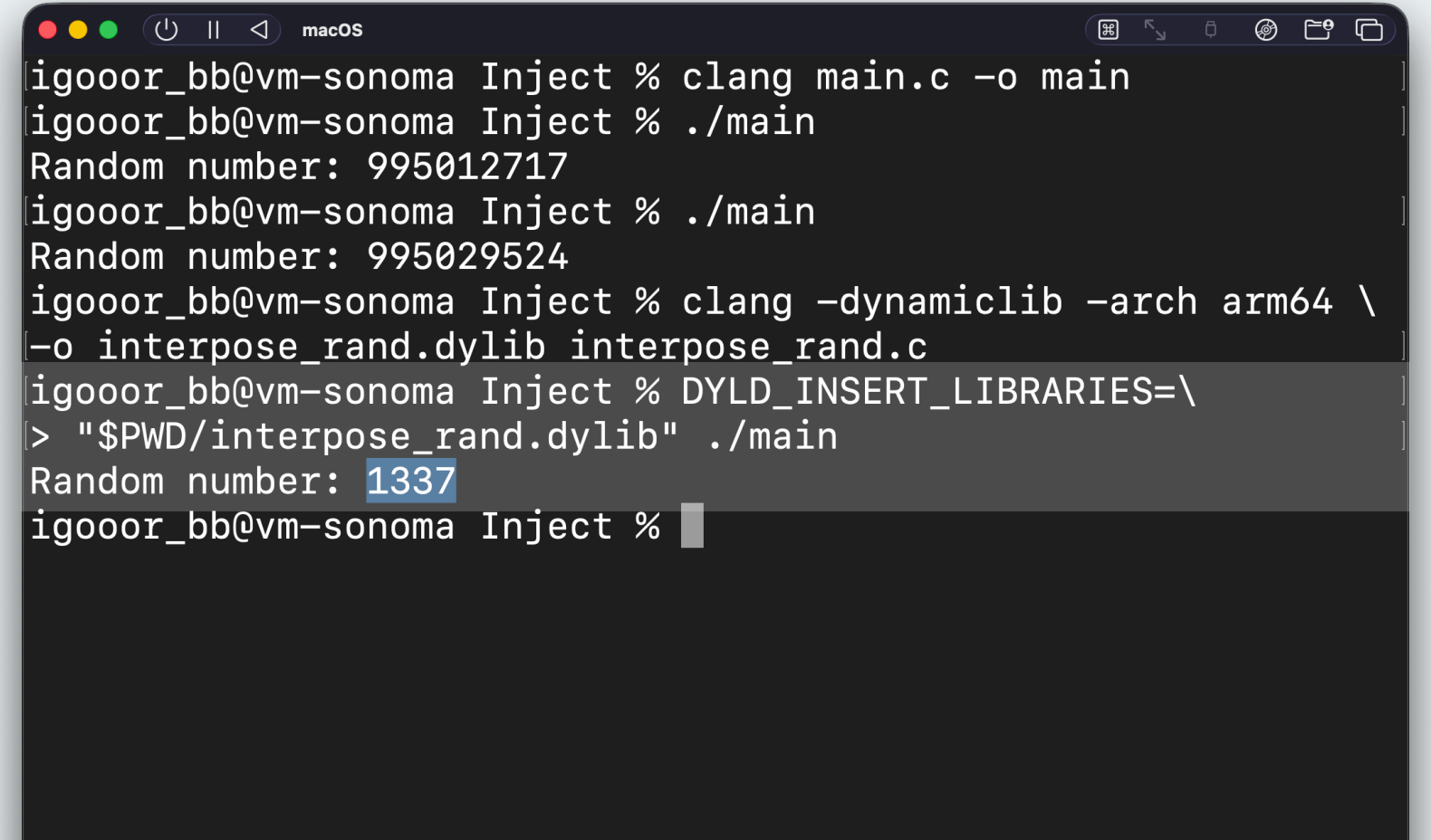
```
}

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


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


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


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

Где здесь iOS разработка?

```objc
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;
```

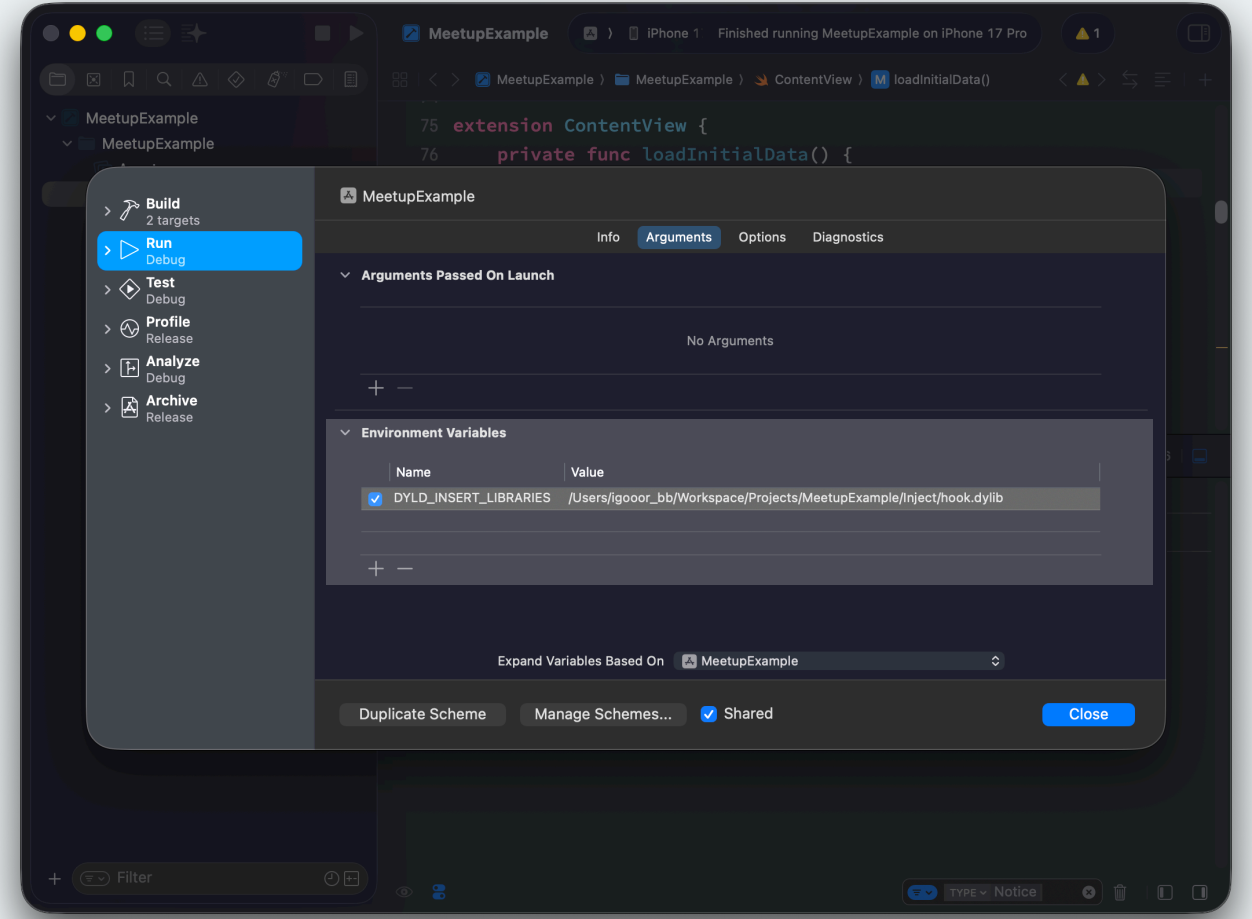
```
    NSURL *u = request.URL;
    if (u) NSLog(@"[URLSwizzle] %@ %@", HTTPMethodOrGET(request), u.absoluteString);

    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

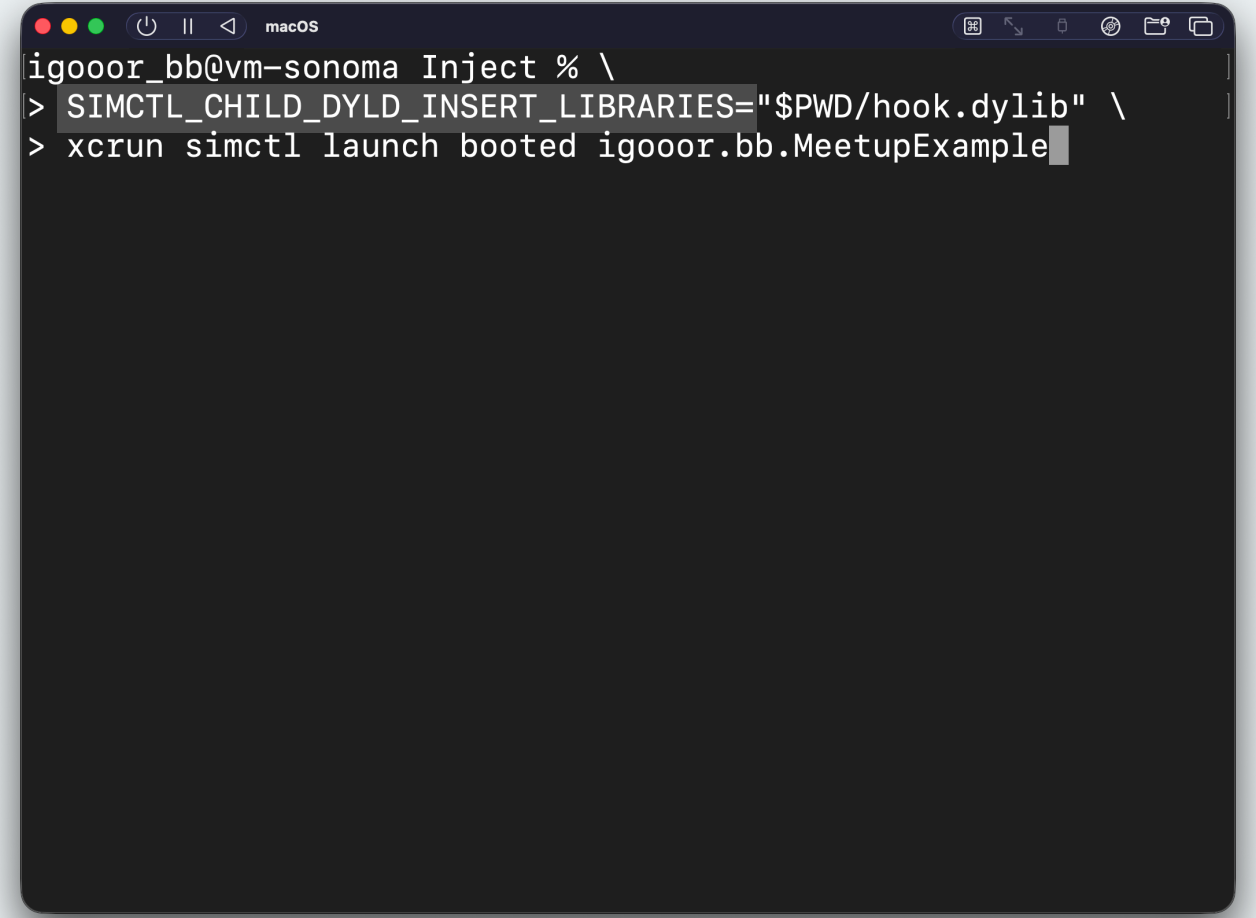
1. Возвращаемся к **DYLD_INSERT_LIBRARIES**
2. Можем передать путь к динамической библиотеке через **аргументы схемы**



Внедрение

simctl launch

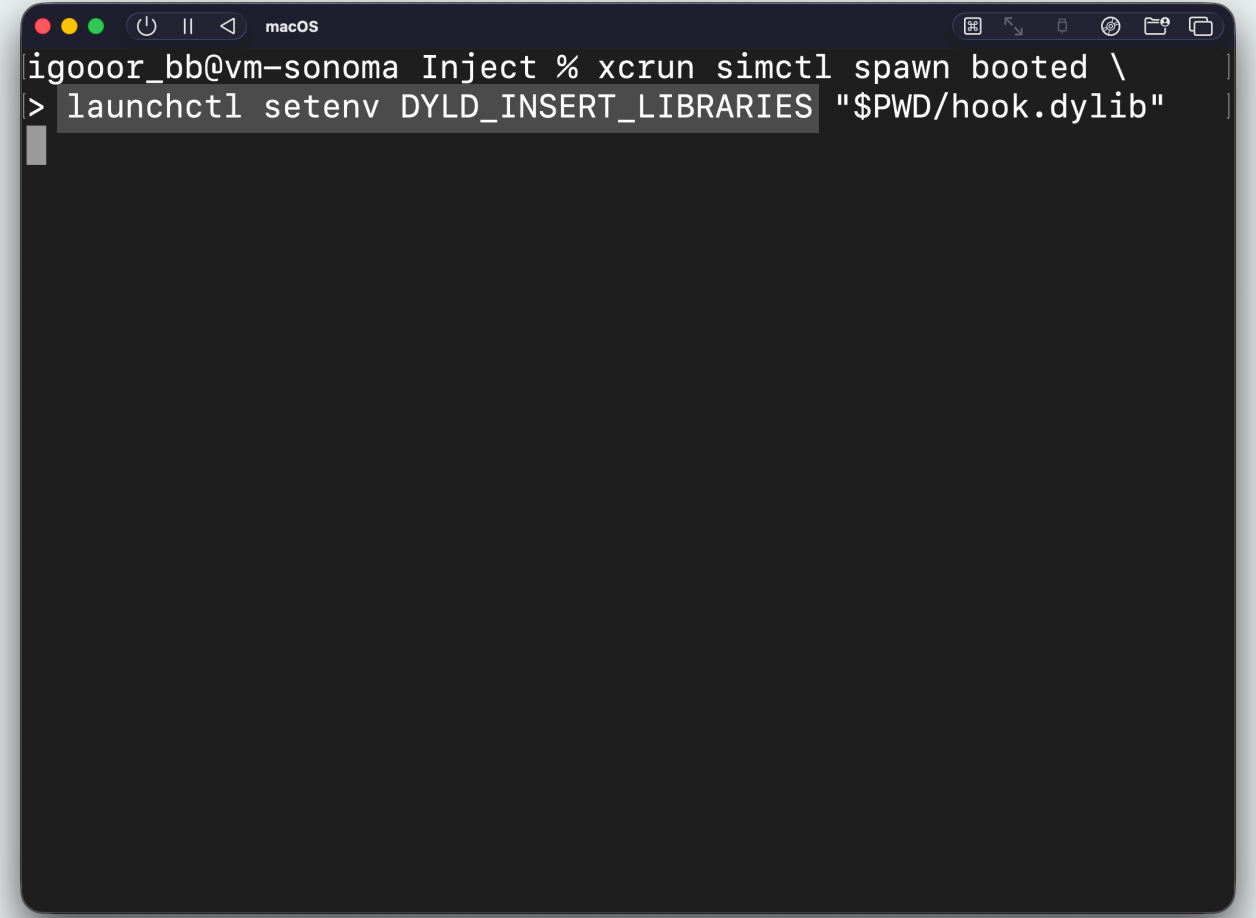
1 Нужно указать префикс **SIMCTL_CHILD_**

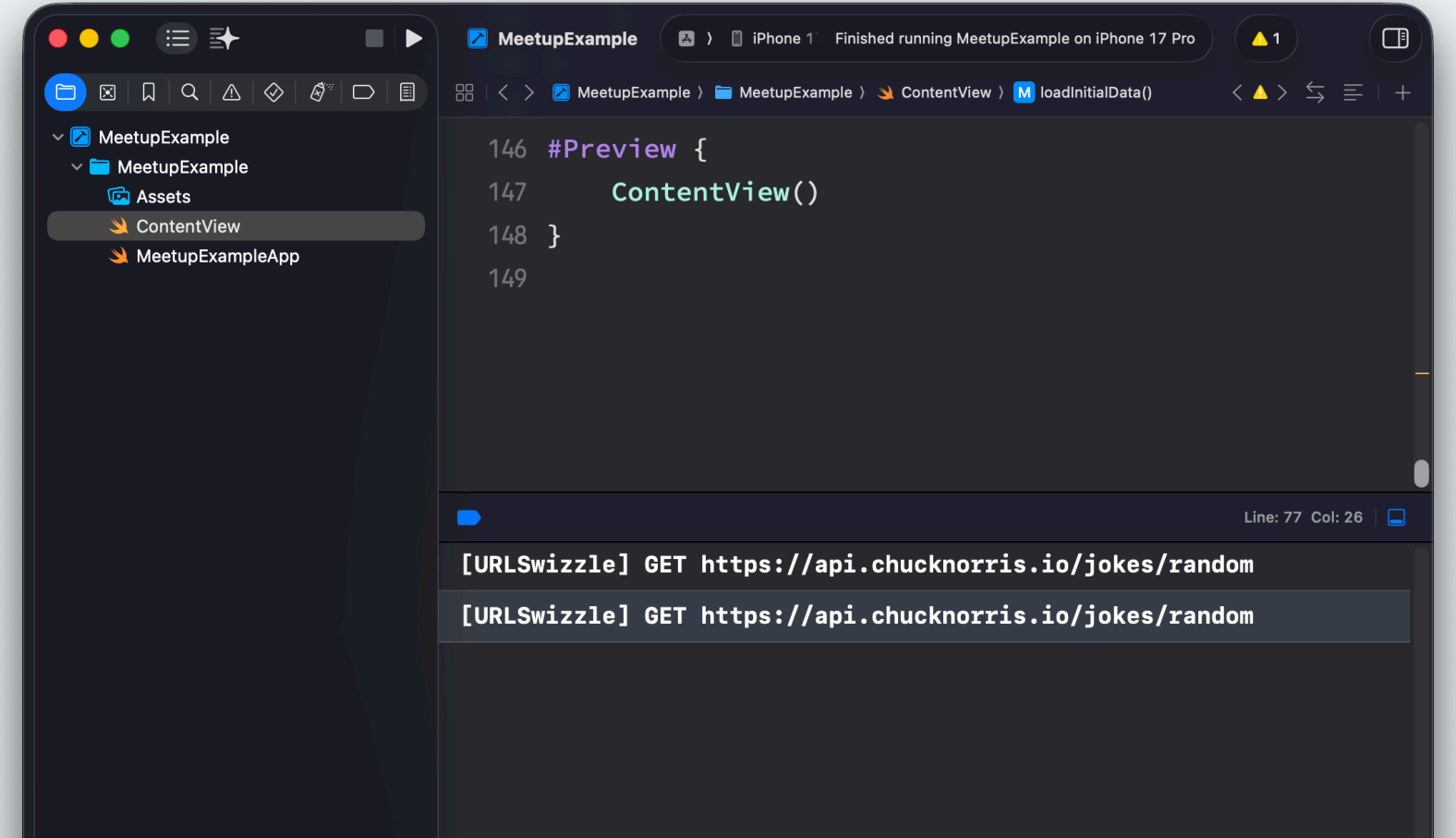


Внедрение

launchctl

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




MeetupExample
iPhone 17 Pro
Finished running MeetupExample on iPhone 17 Pro
1
MeetupExample
MeetupExample
Assets
ContentView
MeetupExampleApp
MeetupExample
MeetupExample
ContentView
loadInitialData()
146 #Preview {
147 ContentView()
148 }
149
Line: 77 Col: 26
[URLSwizzle] GET https://api.chucknorris.io/jokes/random
[URLSwizzle] GET https://api.chucknorris.io/jokes/random

Применение

Simulator Status Magic

1. Популярная утилита для настройки внешнего вида статус-бара симулятора
2. Используется для подготовки к снапшот-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 Kodeco

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