

# Делаем свой микроскоп для тестирования iOS-приложения

Антон Власов

Виктория Кашлина

ПАО «Сбербанк»

```
var e,f=0,l=c.length,b)=f,l.length while  
tion(a){var b,e,g=a?a.ownerDocument||a||r  
a.className="i",la.getAttribute("className  
(a).id=u,ln.getElementsByName||ln.getElem  
(delete d.find.ID,d.filter.ID=function(  
of b.getElementsByTagName?b.getElementsB  
sName&&function(a,b){return"undefined"  
tion selected=""></option></select>"  
querySelectorAll(":checked").length|  
push("name"+L+"*[*^$|!~]?="),a.query  
{c.disconnectedMatch=s.call(a,"div"  
arentNode;return a===d||!(!d||1!==d  
entPosition;return d?d:(d=(a.owne  
d?-1:1}):function(a,b){if(a===b)r  
d);h(d));g[d]==v?-1:h(d)==v?1:  
match)|a.document&&11!==a.docume  
e(d)?e(a,b,l):void 0;return voi  
attributes.slice 0,sort(l),l)wh  
e else if 3  
0  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17  
18  
19  
20  
21  
22  
23  
24  
25  
26  
27  
28  
29  
30  
31  
32  
33  
34  
35  
36  
37  
38  
39  
40  
41  
42  
43  
44  
45  
46  
47  
48  
49  
50  
51  
52  
53  
54  
55  
56  
57  
58  
59  
60  
61  
62  
63  
64  
65  
66  
67  
68  
69  
70  
71  
72  
73  
74  
75  
76  
77  
78  
79  
80  
81  
82  
83  
84  
85  
86  
87  
88  
89  
90  
91  
92  
93  
94  
95  
96  
97  
98  
99  
100
```

# Для кого этот доклад





Тестировщики

Разработчики

Владельцы  
продукта



**Instruments**



**Xcode**

iOS, watchOS, tvOS, macOS, iPadOS

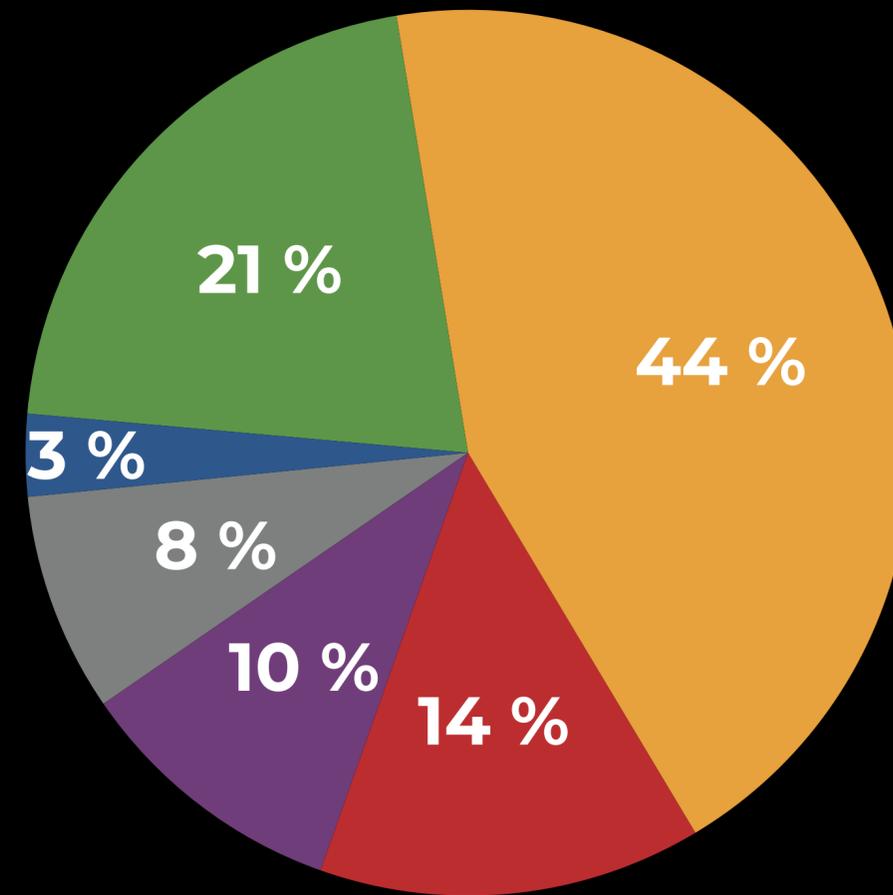
# Как часто вы используете Xcode Instruments?



\*опрошено

**146 разработов**

разного уровня



- Стараюсь профилировать каждую задачу
- 1-2 раза в месяц
- При починке багов
- Использовал на другом проекте, сейчас нет
- Ни разу не запускал
- Не знаю что это

# Зачем вам ЭТОТ ДОКЛАД

Что такое Performance согласно Apple



**Узнать как правильно  
и качественно мерить  
перфоманс приложения**

1

Что такое инструменты Xcode и как с ними работать



3

Делаем свой инструмент



2

Профилируем реальные кейсы



4

Автоматизируем инструменты на CI



A glowing blue and purple circular ring with a white text overlay. The ring is composed of multiple overlapping, slightly offset lines of light, creating a sense of motion and depth. The colors transition from a bright cyan at the top to a deep purple at the bottom. The text is centered within the ring.

# Level 1



## **Задача:**

Понять, что происходит  
в приложении  
с технической стороны

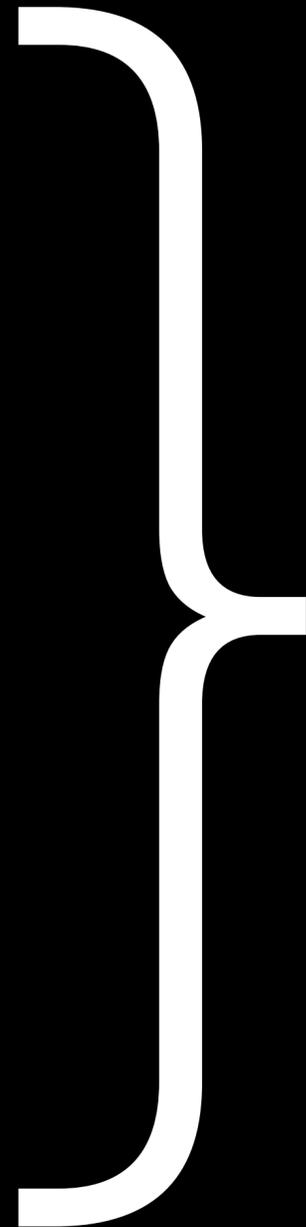
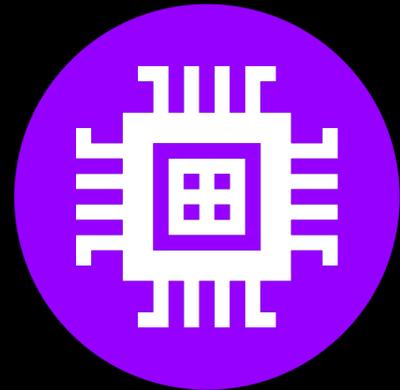
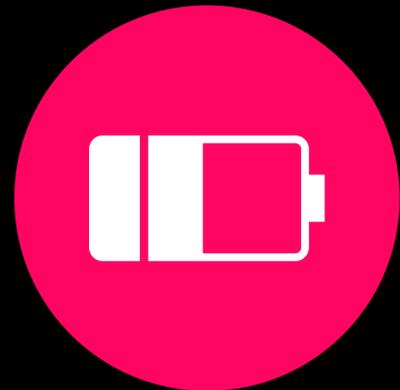
## **Узнаете:**

1. О возможностях с инструментами Xcode
2. О профилировании приложения без исходного кода

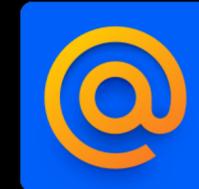
## **Когда применять:**

- ▶ Чем раньше, тем лучше
- ▶ Чем чаще, тем лучше

# Общие ресурсы



Сбербанк  
Онлайн



# Основные группы



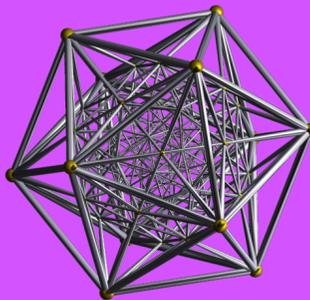
Memory



Perfomance



Energy



GPU

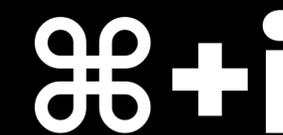


macOS  
perfomance



Core Data

# Выбираем инструмент



Standard Custom Recent Filter

 Blank	 Activity Monitor	 Allocations	 Core Animation	 Core Data	 Counters
 Energy Log	 File Activity	 Game Performance	 Leaks	 Metal System Trace	 Network
 SceneKit	 System Trace	 System Usage	 Time Profiler	 Zombies	

 **Zombies**  
Measures general memory usage while focusing on the detection of over-released "zombie" objects. Also provides statistics on object allocations by class as well as memory address histories for all active allocations.



DemoFox - Apple sample project

# Кейс: лаги в игре

iOS. Проанализировать  
выделение памяти

Estimate: 1

Priority: Minor 





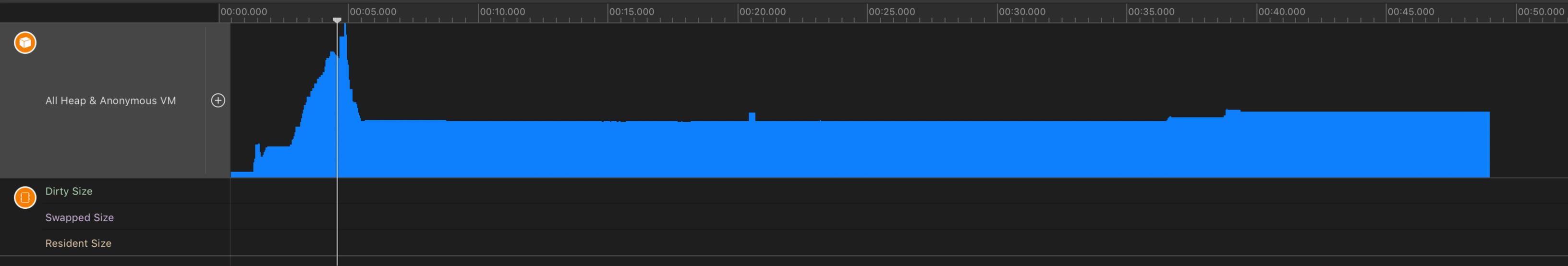
# Allocations

Отображает статистику  
по аллоцированию памяти  
во время работы приложения

**Видос**

Track Filter

All Instruments



Allocations > Statistics > Allocation Summary

Graph	Category	Persistent	# Persistent	# Transient	Total Bytes	# Total	Persistent/Total Bytes
<input checked="" type="checkbox"/>	All Heap & Anonymou...	125,28 MiB	103 399	1 702 233	1,28 GiB	1 805 632	+++
<input type="checkbox"/>	All Heap Allocations	101,31 MiB	103 337	1 701 260	972,85 MiB	1 804 597	
<input type="checkbox"/>	All Anonymous VM	23,96 MiB	62	973	336,62 MiB	1 035	
<input type="checkbox"/>	CGImageHandle	38,48 MiB	31	63	142,64 MiB	94	
<input type="checkbox"/>	VM: CoreServices	5,78 MiB	1	3	23,12 MiB	4	
<input type="checkbox"/>	VM: IOSurface	5,72 MiB	3	7	15,57 MiB	10	
<input type="checkbox"/>	VM: Stack	3,07 MiB	6	0	3,07 MiB	6	
<input type="checkbox"/>	Malloc 276,00 KiB	2,96 MiB	11	0	2,96 MiB	11	
<input type="checkbox"/>	Malloc 272,00 KiB	2,92 MiB	11	7	4,78 MiB	18	
<input type="checkbox"/>	VM: Image IO	2,74 MiB	1	10	8,80 MiB	11	
<input type="checkbox"/>	VM: ImageIO_PNG_Data	2,45 MiB	6	60	130,39 MiB	66	
<input type="checkbox"/>	Malloc 64,00 KiB	2,25 MiB	36	14	3,12 MiB	50	
<input type="checkbox"/>	Malloc 1,02 MiB	2,05 MiB	2	0	2,05 MiB	2	
<input type="checkbox"/>	Malloc 2,00 MiB	2,00 MiB	1	3	8,00 MiB	4	
<input type="checkbox"/>	Malloc 1,94 MiB	1,94 MiB	1	0	1,94 MiB	1	
<input type="checkbox"/>	Malloc 928,00 KiB	1,81 MiB	2	0	1,81 MiB	2	
<input type="checkbox"/>	Malloc 248,00 KiB	1,70 MiB	7	0	1,70 MiB	7	
<input type="checkbox"/>	Malloc 860,00 KiB	1,68 MiB	2	0	1,68 MiB	2	
<input type="checkbox"/>	Malloc 1,53 MiB	1,53 MiB	1	0	1,53 MiB	1	
<input type="checkbox"/>	Malloc 736,00 KiB	1,44 MiB	2	0	1,44 MiB	2	
<input type="checkbox"/>	Malloc 732,00 KiB	1,43 MiB	2	0	1,43 MiB	2	
<input type="checkbox"/>	Malloc 32,00 KiB	1,41 MiB	45	198	7,59 MiB	243	
<input type="checkbox"/>	Malloc 700,00 KiB	1,37 MiB	2	0	1,37 MiB	2	
<input type="checkbox"/>	Malloc 1,35 MiB	1,35 MiB	1	0	1,35 MiB	1	
<input type="checkbox"/>	Malloc 620,00 KiB	1,21 MiB	2	0	1,21 MiB	2	
<input type="checkbox"/>	C3DNode	1,17 MiB	4 037	487	1,31 MiB	4 524	
<input type="checkbox"/>	VM: CG backing stores	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 580,00 KiB	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 576,00 KiB	1,12 MiB	2	0	1,12 MiB	2	
<input type="checkbox"/>	Malloc 64 Bytes	1,07 MiB	17 560	49 886	4,12 MiB	67 446	
<input type="checkbox"/>	VM: CG image	1,05 MiB	7	74	136,34 MiB	81	
<input type="checkbox"/>	SCNNode	1 011,50 KiB	4 046	487	1,11 MiB	4 533	

No Detail

1 |v|

# Настройки

## для инструмента:

- Общие: время работы инструмента
- Специфичные: со своей спецификой для конкретного инструмента

MacBook Pro — Victoria > fox2.app

Options for: Allocations

All Allocations

- Discard unrecorded data upon stop
- Discard events for freed memory
- Only track VM allocations

Heap Allocations

- Record reference counts
- Identify virtual C++ objects
- Enable NSZombie detection

Recorded Types

Type String	Search	Action
<input checked="" type="checkbox"/> *	is Contain...	Record
<input type="checkbox"/> NS	is Prefix	Ignore
<input type="checkbox"/> CF	is Prefix	Ignore
<input type="checkbox"/> Malloc	is Prefix	Ignore

+ - Rules are evaluated top-to-bottom; later rules override earlier ones.

▼ Global Options:

Time limit: 12 hours

Recording Mode:  Immediate  Deferred  Last seconds

Close Record

# Итерации

Можно легко переключаться  
между замерами в верхней  
панели

Allocations ▾ Statistics Allocation Summary

Graph	Category	Persistent	# Persistent	# Transient	Total Bytes	# Total	Persistent/Total Bytes
<input checked="" type="checkbox"/>	All Heap & Anonymou...	125,28 MiB	103 399	1 702 233	1,28 GiB	1 805 632	
<input type="checkbox"/>	All Heap Allocations	101,31 MiB	103 337	1 701 260	972,85 MiB	1 804 597	
<input type="checkbox"/>	All Anonymous VM	23,96 MiB	62	973	336,62 MiB	1 035	
<input type="checkbox"/>	CGImageHandle	38,48 MiB	31	63	142,64 MiB	94	
<input type="checkbox"/>	VM: CoreServices	5,78 MiB	1	3	23,12 MiB	4	
<input type="checkbox"/>	VM: IOSurface	5,72 MiB	3	7	15,57 MiB	10	
<input type="checkbox"/>	VM: Stack	3,07 MiB	6	0	3,07 MiB	6	
<input type="checkbox"/>	Malloc 276,00 KiB	2,96 MiB	11	0	2,96 MiB	11	
<input type="checkbox"/>	Malloc 272,00 KiB	2,92 MiB	11	7	4,78 MiB	18	
<input type="checkbox"/>	VM: Image IO	2,74 MiB	1	10	8,80 MiB	11	
<input type="checkbox"/>	VM: ImageIO_PNG_Data	2,45 MiB	6	60	130,39 MiB	66	
<input type="checkbox"/>	Malloc 64,00 KiB	2,25 MiB	36	14	3,12 MiB	50	
<input type="checkbox"/>	Malloc 1,02 MiB	2,05 MiB	2	0	2,05 MiB	2	
<input type="checkbox"/>	Malloc 2,00 MiB	2,00 MiB	1	3	8,00 MiB	4	
<input type="checkbox"/>	Malloc 1,94 MiB	1,94 MiB	1	0	1,94 MiB	1	
<input type="checkbox"/>	Malloc 928,00 KiB	1,81 MiB	2	0	1,81 MiB	2	
<input type="checkbox"/>	Malloc 248,00 KiB	1,70 MiB	7	0	1,70 MiB	7	
<input type="checkbox"/>	Malloc 860,00 KiB	1,68 MiB	2	0	1,68 MiB	2	
<input type="checkbox"/>	Malloc 1,53 MiB	1,53 MiB	1	0	1,53 MiB	1	
<input type="checkbox"/>	Malloc 736,00 KiB	1,44 MiB	2	0	1,44 MiB	2	
<input type="checkbox"/>	Malloc 732,00 KiB	1,43 MiB	2	0	1,43 MiB	2	
<input type="checkbox"/>	Malloc 32,00 KiB	1,41 MiB	45	198	7,59 MiB	243	
<input type="checkbox"/>	Malloc 700,00 KiB	1,37 MiB	2	0	1,37 MiB	2	
<input type="checkbox"/>	Malloc 1,35 MiB	1,35 MiB	1	0	1,35 MiB	1	
<input type="checkbox"/>	Malloc 620,00 KiB	1,21 MiB	2	0	1,21 MiB	2	
<input type="checkbox"/>	C3DNode	1,17 MiB	4 037	487	1,31 MiB	4 524	
<input type="checkbox"/>	VM: CG backing stores	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 580,00 KiB	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 576,00 KiB	1,12 MiB	2	0	1,12 MiB	2	
<input type="checkbox"/>	Malloc 64 Bytes	1,07 MiB	17 560	49 886	4,12 MiB	67 446	
<input type="checkbox"/>	VM: CG image	1,05 MiB	7	74	136,34 MiB	81	
<input type="checkbox"/>	SCNNode	1 011,50 KiB	4 046	487	1,11 MiB	4 533	

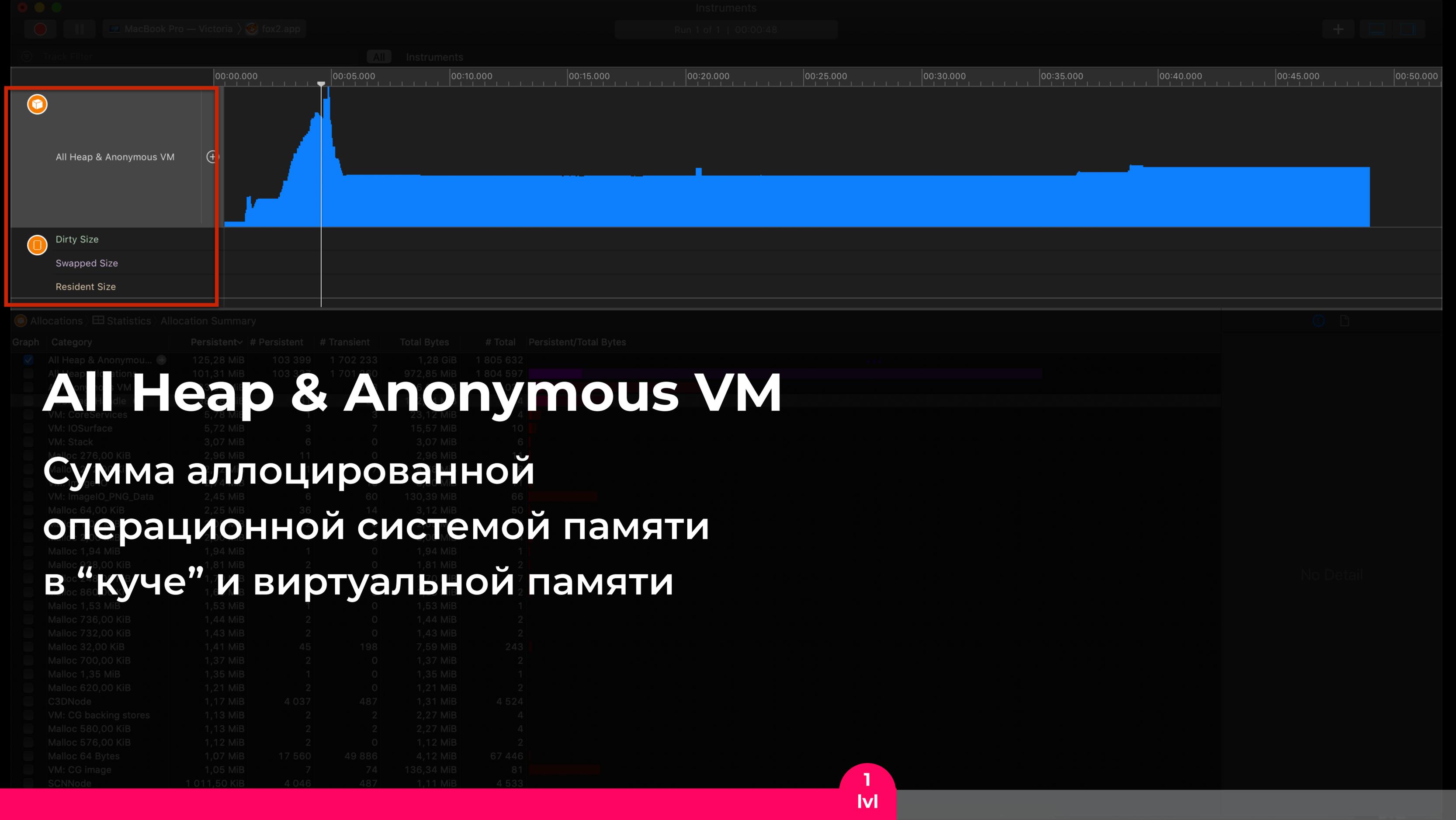
No Detail

# Добавление инструментов

- Возможность параллельного использования
- Исключение: те, для работы которых требуется реальный девайс

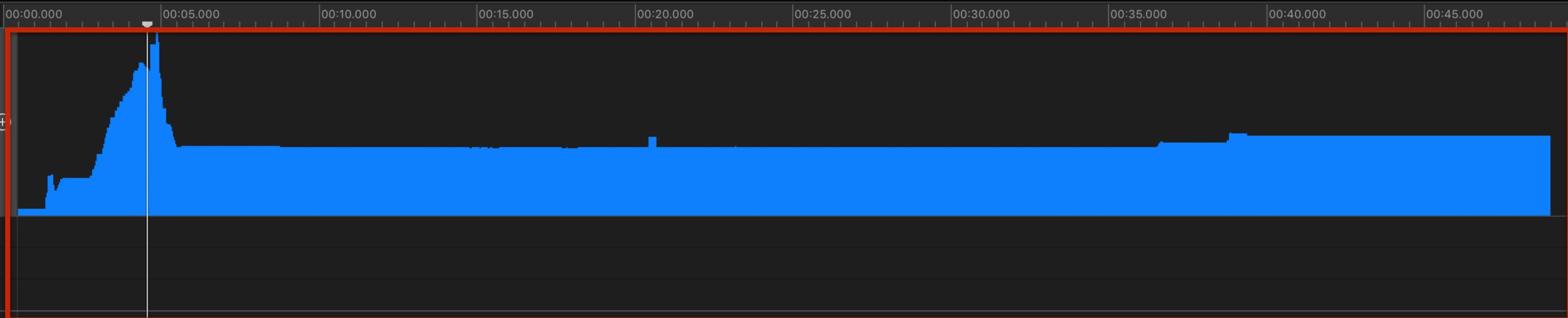
**+**

- Activity Monitor** - Measures system and process activity.
- Advanced Graphics Statistics** - Views graphics, driver-specific counters. These metrics are low level and OS/hardware dependent.
- Allocations** - Analyzes the memory life-cycle of process' allocated blocks; can record reference counting events.
- Bluetooth On/Off Log** - Tracks when the Bluetooth is enabled.
- Brightness Log** - Tracks the brightness level of the display.
- Carbon Events** - Monitors events returned from WaitNextEvent.
- Cocoa Events** - Monitors events sent to '[NSApp sendEvent:]'.
- Cocoa Layout** - Observe changes to NSLayoutConstraint objects.
- Core Animation FPS** - Graphs the estimated Core Animation frames per second.



# All Heap & Anonymous VM

Сумма аллоцированной операционной системой памяти в “куче” и виртуальной памяти



# Track

- Выбран только один инструмент
- График отображает аллоцирование памяти

# Allocation summary

Показания по всем данным,  
собранным инструментом за прогон  
процесса

Allocations > Statistics > Allocation Summary

Graph	Category	Persistent	# Persistent	# Transient	Total Bytes	# Total	Persistent/Total Bytes
<input checked="" type="checkbox"/>	All Heap & Anonymou...	125,28 MiB	103 399	1 702 233	1,28 GiB	1 805 632	+++
<input type="checkbox"/>	All Heap Allocations	101,31 MiB	103 337	1 701 260	972,85 MiB	1 804 597	
<input type="checkbox"/>	All Anonymous VM	23,96 MiB	62	973	336,62 MiB	1 035	
<input type="checkbox"/>	CGImageHandle	38,48 MiB	31	63	142,64 MiB	94	
<input type="checkbox"/>	VM: CoreServices	5,78 MiB	1	3	23,12 MiB	4	
<input type="checkbox"/>	VM: IOSurface	5,72 MiB	3	7	15,57 MiB	10	
<input type="checkbox"/>	VM: Stack	3,07 MiB	6	0	3,07 MiB	6	
<input type="checkbox"/>	Malloc 276,00 KiB	2,96 MiB	11	0	2,96 MiB	11	
<input type="checkbox"/>	Malloc 272,00 KiB	2,92 MiB	11	7	4,78 MiB	18	
<input type="checkbox"/>	VM: Image IO	2,74 MiB	1	10	8,80 MiB	11	
<input type="checkbox"/>	VM: ImageIO_PNG_Data	2,45 MiB	6	60	130,39 MiB	66	
<input type="checkbox"/>	Malloc 64,00 KiB	2,25 MiB	36	14	3,12 MiB	50	
<input type="checkbox"/>	Malloc 1,02 MiB	2,05 MiB	2	0	2,05 MiB	2	
<input type="checkbox"/>	Malloc 2,00 MiB	2,00 MiB	1	3	8,00 MiB	4	
<input type="checkbox"/>	Malloc 1,94 MiB	1,94 MiB	1	0	1,94 MiB	1	
<input type="checkbox"/>	Malloc 928,00 KiB	1,81 MiB	2	0	1,81 MiB	2	
<input type="checkbox"/>	Malloc 248,00 KiB	1,70 MiB	7	0	1,70 MiB	7	
<input type="checkbox"/>	Malloc 860,00 KiB	1,68 MiB	2	0	1,68 MiB	2	
<input type="checkbox"/>	Malloc 1,53 MiB	1,53 MiB	1	0	1,53 MiB	1	
<input type="checkbox"/>	Malloc 736,00 KiB	1,44 MiB	2	0	1,44 MiB	2	
<input type="checkbox"/>	Malloc 732,00 KiB	1,43 MiB	2	0	1,43 MiB	2	
<input type="checkbox"/>	Malloc 32,00 KiB	1,41 MiB	45	198	7,59 MiB	243	
<input type="checkbox"/>	Malloc 700,00 KiB	1,37 MiB	2	0	1,37 MiB	2	
<input type="checkbox"/>	Malloc 1,35 MiB	1,35 MiB	1	0	1,35 MiB	1	
<input type="checkbox"/>	Malloc 620,00 KiB	1,21 MiB	2	0	1,21 MiB	2	
<input type="checkbox"/>	C3DNode	1,17 MiB	4 037	487	1,31 MiB	4 524	
<input type="checkbox"/>	VM: CG backing stores	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 580,00 KiB	1,13 MiB	2	2	2,27 MiB	4	
<input type="checkbox"/>	Malloc 576,00 KiB	1,12 MiB	2	0	1,12 MiB	2	
<input type="checkbox"/>	Malloc 64 Bytes	1,07 MiB	17 560	49 886	4,12 MiB	67 446	
<input type="checkbox"/>	VM: CG image	1,05 MiB	7	74	136,34 MiB	81	
<input type="checkbox"/>	SCNNode	1 011,50 KiB	4 046	487	1,11 MiB	4 533	

No Detail

# Кейс: лаги в игре

iOS. Проанализировать работу CPU

Estimate: 1

Priority: Minor 

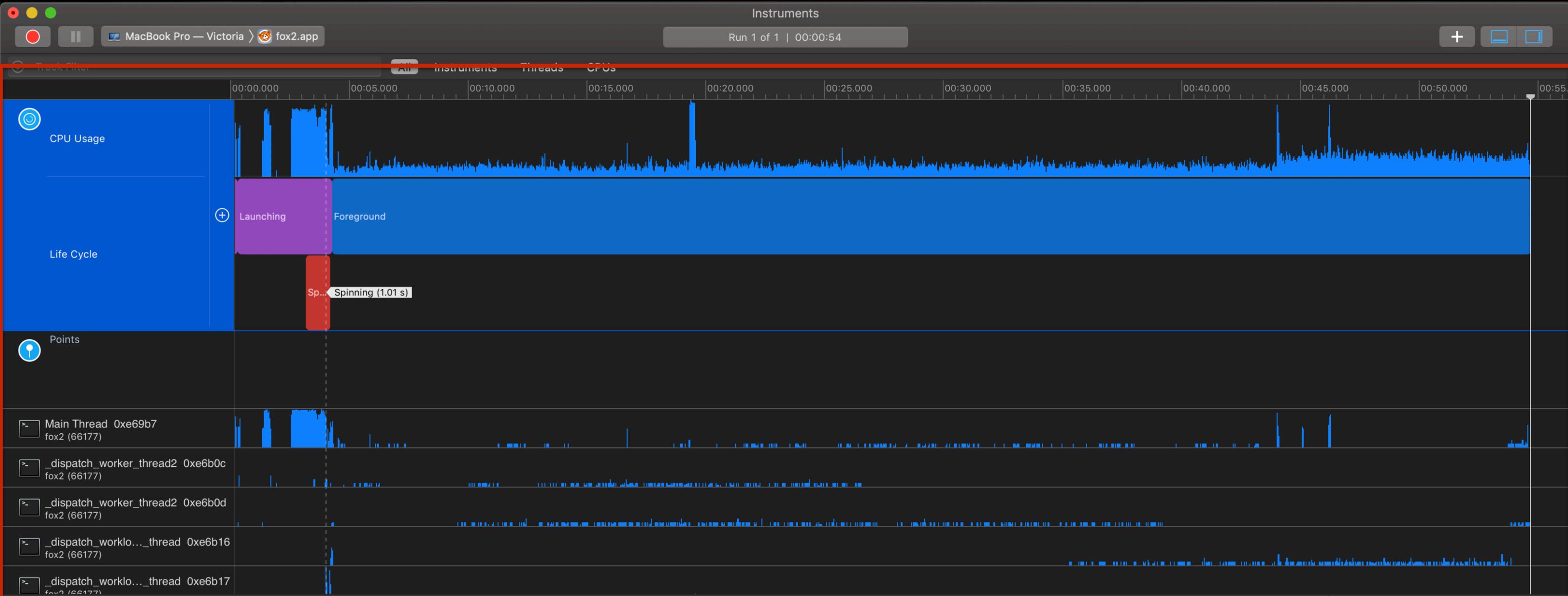




## **Time Profiler**

Отображает call tree, которое показывает количество времени, потраченное на выполнение методов в процессе работы приложения

**Видос**



# CPU Usage, Life Cycle, threads

Графическое отображение данных, собранных инструментом в разрезе разных аспектов

# Detail Panel

Показывает основную информацию  
о конкретном инструменте, который  
вы используете

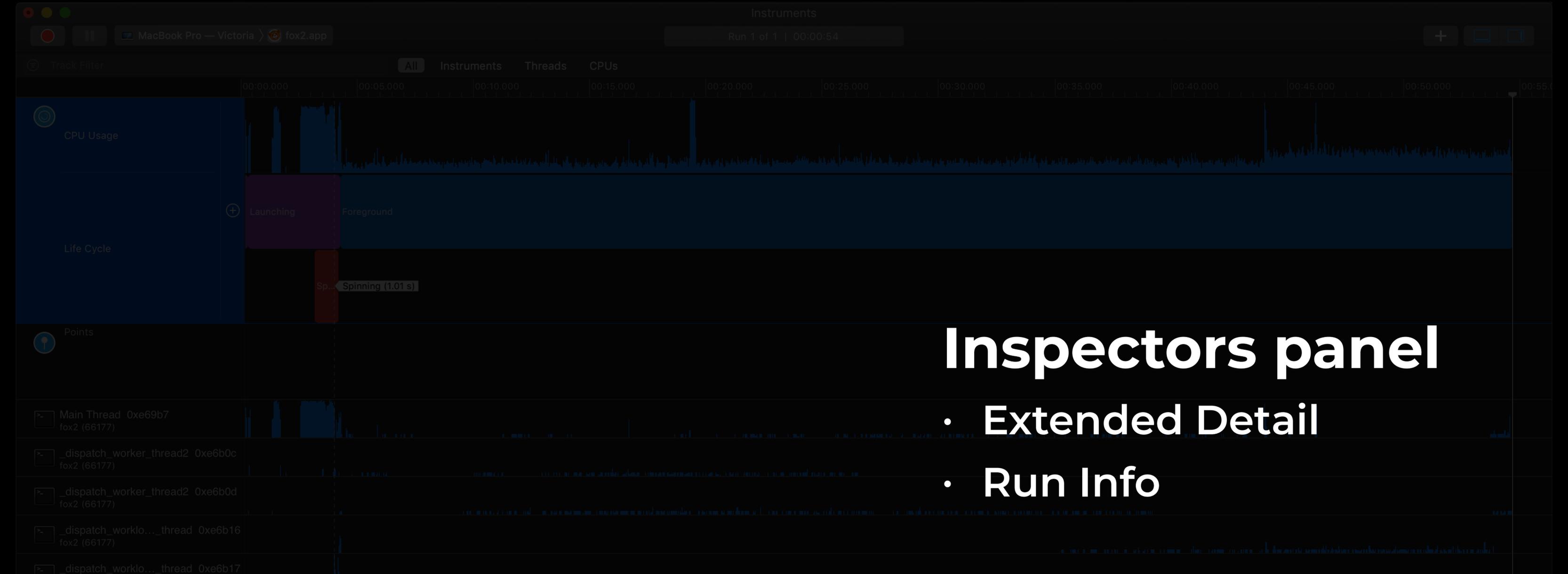
Time Profiler > Profile > Root

Weight	Self Weight	Symbol Name
11.41 s 100.0%	0 s	fox2 (66177)
5.93 s 51.9%	0 s	▶CVDisplayLink::runIOThread 0xe6b7c
2.23 s 19.5%	0 s	▶Main Thread 0xe69b7
752.00 ms 6.5%	0 s	▶_dispatch_workloop_worker_thread 0xe6b79
698.00 ms 6.1%	0 s	▶_dispatch_workloop_worker_thread 0xe6b17
488.00 ms 4.2%	0 s	▶HALC_ProxyIOContext::IOThreadEntry 0xe6b68
376.00 ms 3.2%	0 s	▶_dispatch_worker_thread2 0xe6b0d
358.00 ms 3.1%	0 s	▶_dispatch_workloop_worker_thread 0xe6b16
270.00 ms 2.3%	0 s	▶_dispatch_worker_thread2 0xe6b0c
153.00 ms 1.3%	0 s	▶_dispatch_worker_thread 0xe6b30
64.00 ms 0.5%	0 s	▶_dispatch_workloop_worker_thread 0xe6b7a
57.00 ms 0.4%	0 s	▶_NSEventThread 0xe6b7b
34.00 ms 0.2%	0 s	▶_dispatch_worker_thread2 0xe6b77
2.00 ms 0.0%	0 s	▶_dispatch_workloop_worker_thread 0xe6b78

Heaviest Stack Trace

```
11413 fox2 (66177)
5926 CVDisplayLink::runIOThread 0xe6b7c
5926 thread_start
```

```
.....
1 _platform_memmove$VARIANT$Has
```



# Inspectors panel

- Extended Detail
- Run Info

Time Profiler > Profile > Root

Weight	Self Weight	Symbol Name
11.41 s 100.0%	0 s	fox2 (66177)
5.93 s 51.9%	0 s	▶CVDisplayLink::runIOThread 0xe6b7c
2.23 s 19.5%	0 s	▶Main Thread 0xe69b7
752.00 ms 6.5%	0 s	▶_dispatch_workloop_worker_thread 0xe6b79
698.00 ms 6.1%	0 s	▶_dispatch_workloop_worker_thread 0xe6b17
488.00 ms 4.2%	0 s	▶HALC_ProxyIOContext::IOThreadEntry 0xe6b68
376.00 ms 3.2%	0 s	▶_dispatch_worker_thread2 0xe6b0d
358.00 ms 3.1%	0 s	▶_dispatch_workloop_worker_thread 0xe6b16
270.00 ms 2.3%	0 s	▶_dispatch_worker_thread2 0xe6b0c
153.00 ms 1.3%	0 s	▶_dispatch_worker_thread 0xe6b30
64.00 ms 0.5%	0 s	▶_dispatch_workloop_worker_thread 0xe6b7a
57.00 ms 0.4%	0 s	▶_NSEventThread 0xe6b7b
34.00 ms 0.2%	0 s	▶_dispatch_worker_thread2 0xe6b77
2.00 ms 0.0%	0 s	▶_dispatch_workloop_worker_thread 0xe6b78

Heaviest Stack Trace

```

11413 fox2 (66177)
5926 CVDisplayLink::runIOThread 0xe6b7c
5926 thread_start
.....
1 _platform_memmove$VARIANT$Has

```



## Плюсы первого уровня

- ✓ Предоставляют детализированные данные
- ✓ Видят “шире”, чем ручное тестирование
- ✓ Помогают найти проблему и сказать разработчику, где конкретно она возникла



## **Минусы первого уровня**

- **Данные не структурированы, хаотичны**
- **Много лишнего функционала**
- **Нужно уметь пользоваться**
- **Требуется практика**

# Что если нам этого недостаточно

- Готовых инструментов Apple бывает недостаточно
- Можно сделать инструменты более подробными и удобными





**Level  
2**



**Задача:**

Структурировать  
хаотичный поток данных

**Узнаете:** что такое signpost  
и как его использовать

**Получите:** более  
структурированные данные

**Сможете:** соотнести  
технические и бизнес-  
процессы между собой

**Когда применять:** хотим  
локализовать  
определенное событие

# Разметка событий signpost

## Тип события



### Одномоментное

(нажатие на кнопку, возникновение ошибки и др.)



### Интервальное

(сетевой запрос, загрузка документа и др.)

используем

os\_signpost с типом  
**.event**

os\_signpost с типом  
**.begin** и **.end**

# Интеграция в проект: Swift

импорт

```
import os.signpost
```

экземпляр

```
let pointsOfInterestLog = OSLog(subsystem: "com.example.your-app", category: .  
pointsOfInterest)  
let networkLog = OSLog(subsystem: "com.example.your-app", category:  
"NetworkOperations")
```

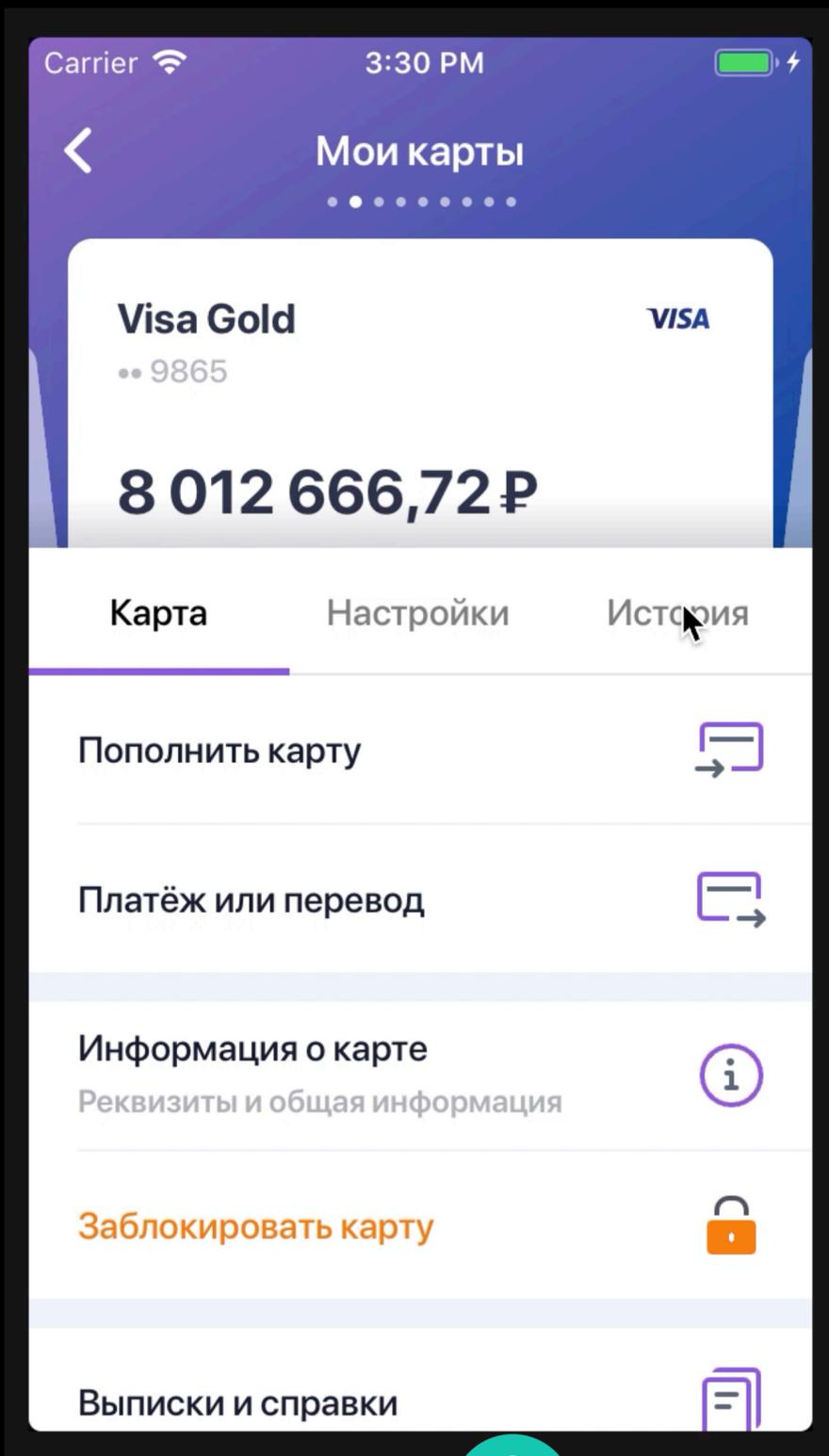
тип события  
тип категории

```
os_signpost(.event, log: pointsOfInterestLog, name: "Start work")  
os_signpost(.begin, log: networkLog, name: "Overall work")
```

тип события

```
for element in elements {  
    os_signpost(.begin, log: networkLog, name: "Element work")  
    makeWork(for: element)  
    os_signpost(.end, log: networkLog, name: "Element work")  
}  
  
os_sginpost(.end, log: networkLog, name: "Overall work")
```

# Бизнес-процесс



# Технический процесс

1

Получение данных  
по сети

2

Формирование  
документа

3

Отображение  
на экране



18:41

## Отчёт по истории

**СБЕРБАНК** ул. Вавилова, д. 19, г. Москва, 117997  
Контактный центр: ☎ 900 ☎, 8-800-555-55-50  
www.sberbank.ru

Справка По месту требования

**Об остатках на счетах**  
по состоянию на 10.10.2016, клиент:

**Довлатов Сергей Донатович**  
паспорт: серия 40 05 номер 738815  
выдан: Отделением Внутренних дел Орехово-Борисово Южное в городе Москва  
дата выдачи: 25.10.2002, код подразделения 772-005

ПАО Сбербанк сообщает\*, что у клиента имеются действующие счета:

Продукт	Валюта	Остаток в валюте счета / в рублях РФ**
Счет №4081784043812317366 «Сохраняй ОнЛ@йн» Дата открытия: 27.07.2015 <small>Открыт в доп. офисе №0002/01591 (г. Москва, ул. Раменки, 8, корпус 2) Московский банк Сбербанка России</small>	Доллар США	84,78 USD / 5 087,28 RUB Восемьдесят четыре доллара 78 центов / Пять тысяч восемьдесят семь рублей 28 копеек
Счет №4081784043812317366 «Управляй ОнЛ@йн» Дата открытия: 27.07.2015 <small>Открыт в доп. офисе №0002/01591 (г. Москва, ул. Раменки, 8, корпус 2) Московский банк Сбербанка России</small>	Доллар США	900,00 USD / 54 000,01 RUB Девятьсот долларов 00 центов / Пятьдесят четыре тысячи рублей 01 копейка
Счет №4081784043812317366 Standart MasterCard Дата открытия: 27.07.2015 <small>Открыт в доп. офисе №0002/01591 (г. Москва, ул. Раменки, 8, корпус 2) Московский банк Сбербанка России</small>	Рубль РФ	117 153,53 RUB Сто семнадцать тысяч сто пятьдесят три рубля 53 копейки
Счет №40802381061255887412 Номинальный счет Дата открытия: 20.06.2016 Владелец счета: Шареев Азамат Азаматович Бенефициар счета: Ромашкин Роман Романович <small>Открыт в доп. офисе №0002/01591 (г. Москва, ул. Раменки, 8, корпус 2) Московский банк ПАО Сбербанк</small>	Рубль РФ	120 100,50 RUB Сто двадцать тысяч сто рублей 50 копеек

Дата формирования: 10.10.2016

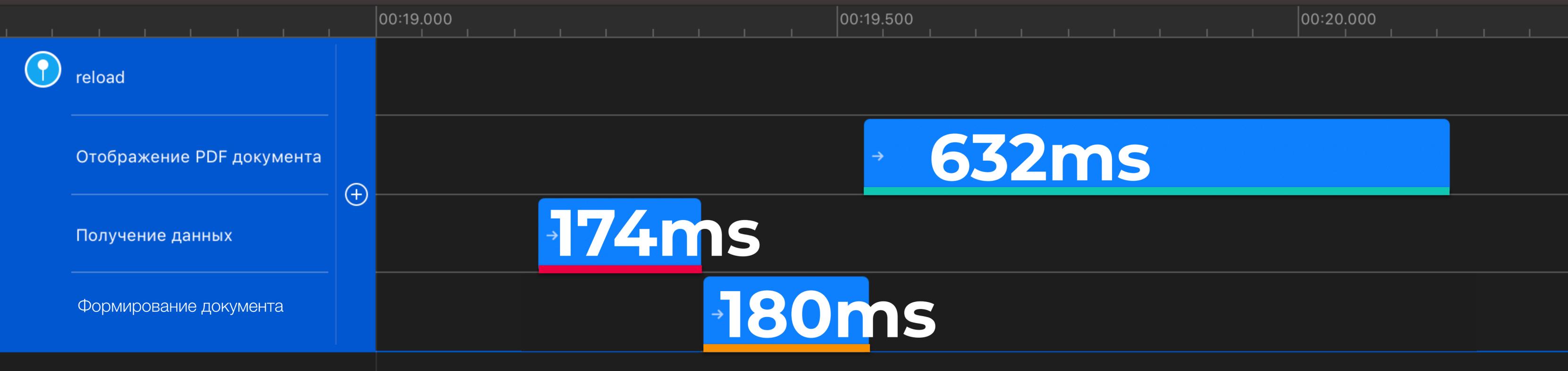
  
Сальникова Н.И.  
Управляющий директор Дивизиона «Забота о клиентах»

\* Обратите внимание, не все организации принимают документ без заверения его в отделении Банка. Рекомендуем уточнить требования к формату документа.

**СОХРАНИТЬ ИЛИ ОТПРАВИТЬ**

**ВЕРНУТЬСЯ НА ГЛАВНЫЙ**

# Слабое звено: отображение дольше, чем сетевой запрос



os\_signpost Summary: Intervals

Category / Name / Start Message / End Message	Count	Duration	Min Duration	Avg Duration	Std Dev Du...	Max Durati...
▼ * All *	4	1.79 s	75.89 µs	448.48 ms	353.42 ms	987.07 ms
▼ Statements	3	1.79 s	174.70 ms	597.94 ms	305.83 ms	987.07 ms
▶ Получение данных	1	174.70 ms	174.70 ms	174.70 ms	n/a	174.70 ms
▶ Создание PDF документа	1	987.07 ms	987.07 ms	987.07 ms	341.59 ms	987.07 ms
▶ Отображение PDF документа	1	632.06 ms	632.06 ms	632.06 ms	108.26 ms	632.06 ms

# Слабое звено: отображение дольше, чем сетевой запрос



**632ms**

00:20.000

Duration	Std Dev	Du...	Max Durati...
48 ms	353.42 ms	987.07 ms	
94 ms	305.83 ms	987.07 ms	
70 ms	n/a	174.70 ms	
07 ms	341.59 ms	987.07 ms	
06 ms	108.26 ms	632.06 ms	



**WKWebView**



**PDFKit**

**UIWebView**

**CoreGraphics**

iOS 2.0–12.0

Deprecated



# Кейс: отображение документа

Протестировать механизмы отображения документа (PDFKit vs WKWebView)

Estimate: 5-8

Priority: Major 



# Включаем signpost



Standard Custom Recent Filter

Blank Activity Monitor Allocations Core Animation Core Data Counters

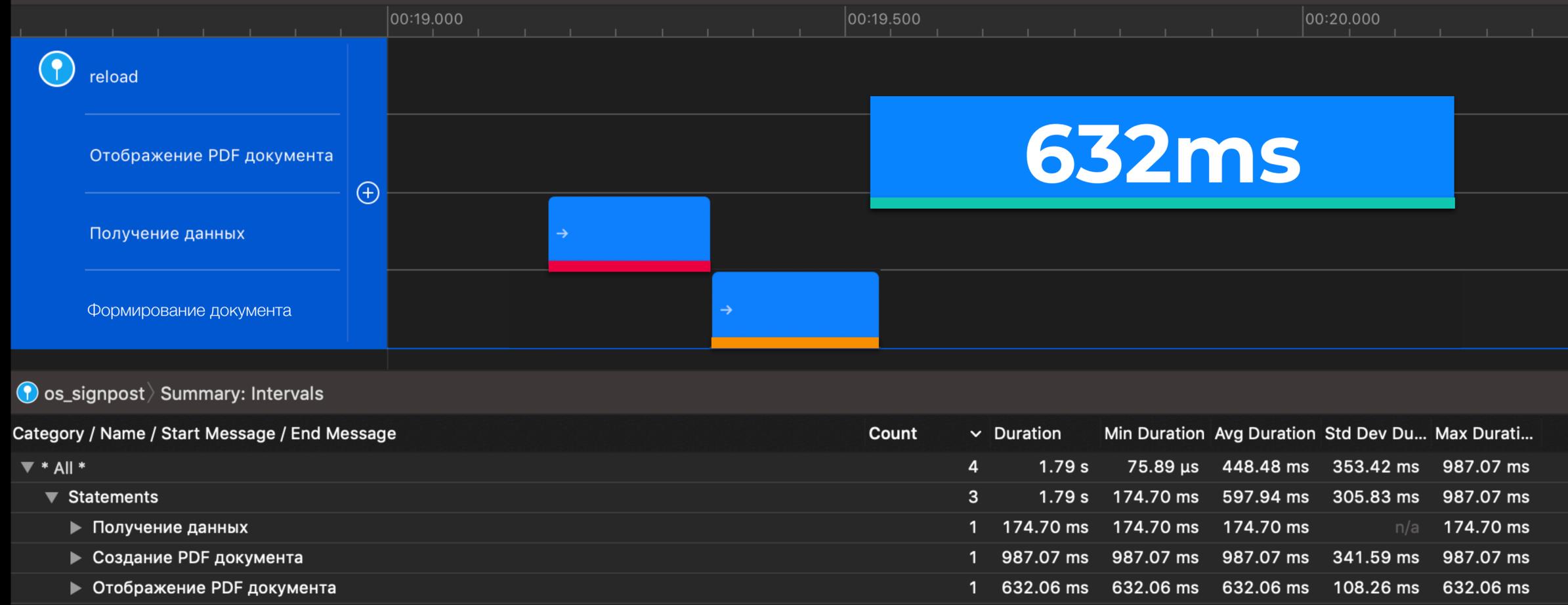
Energy Log File Activity Game Performance Leaks Metal System Trace MockIsAlive

 Blank  
A blank trace document that can be customized with instruments from the Library.

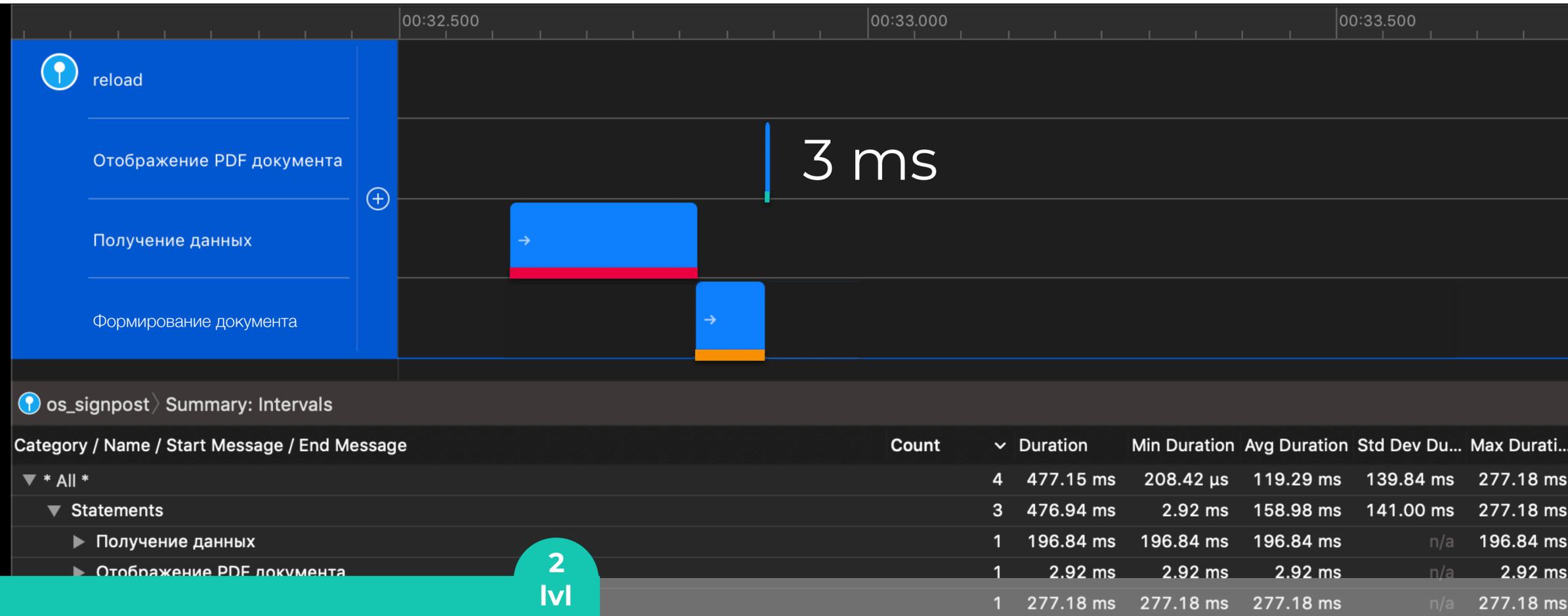
**Видос**



WKWebKit



PDFKit



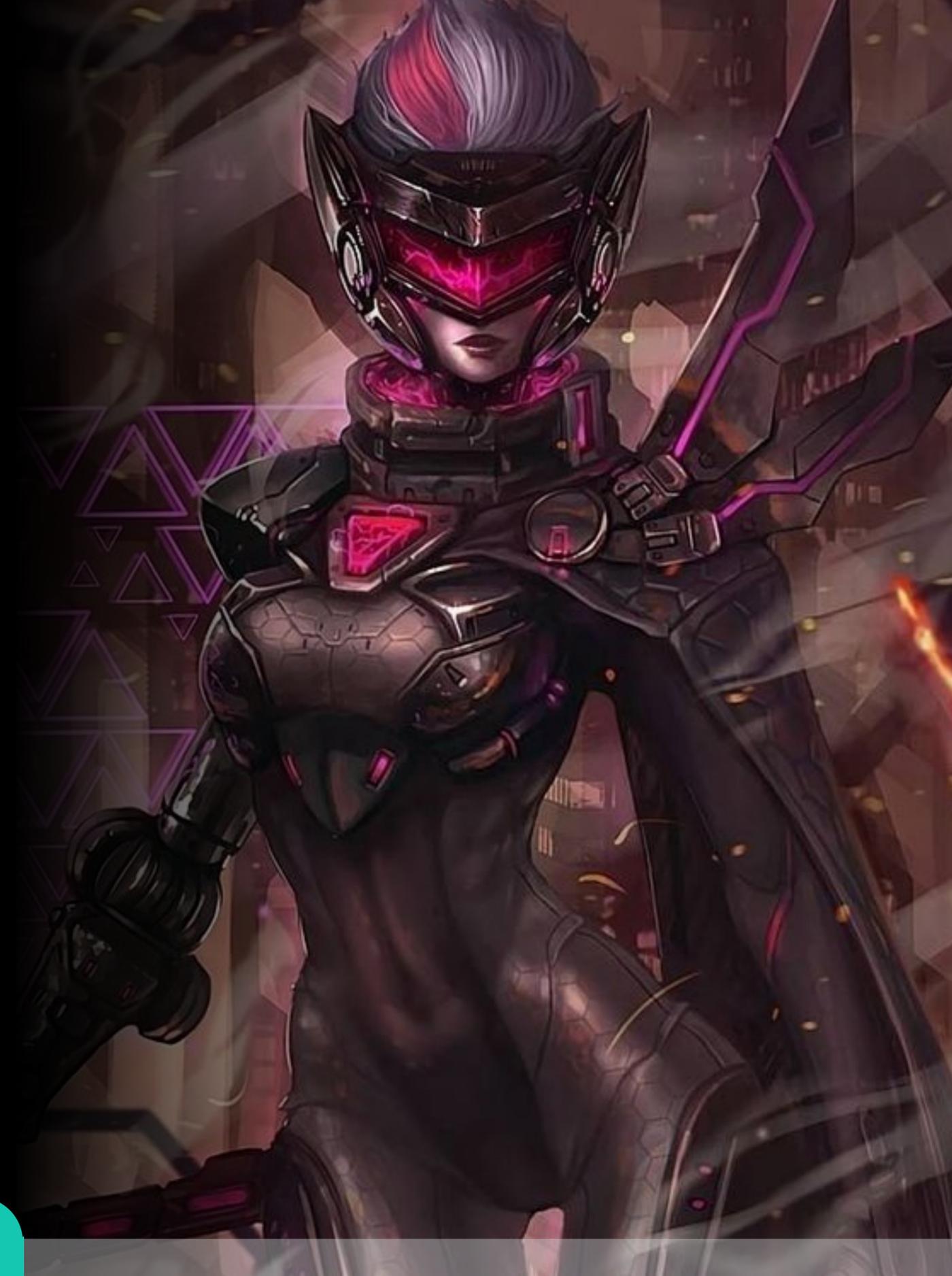
2  
lvl

# Кейс: формирование PDF-документа

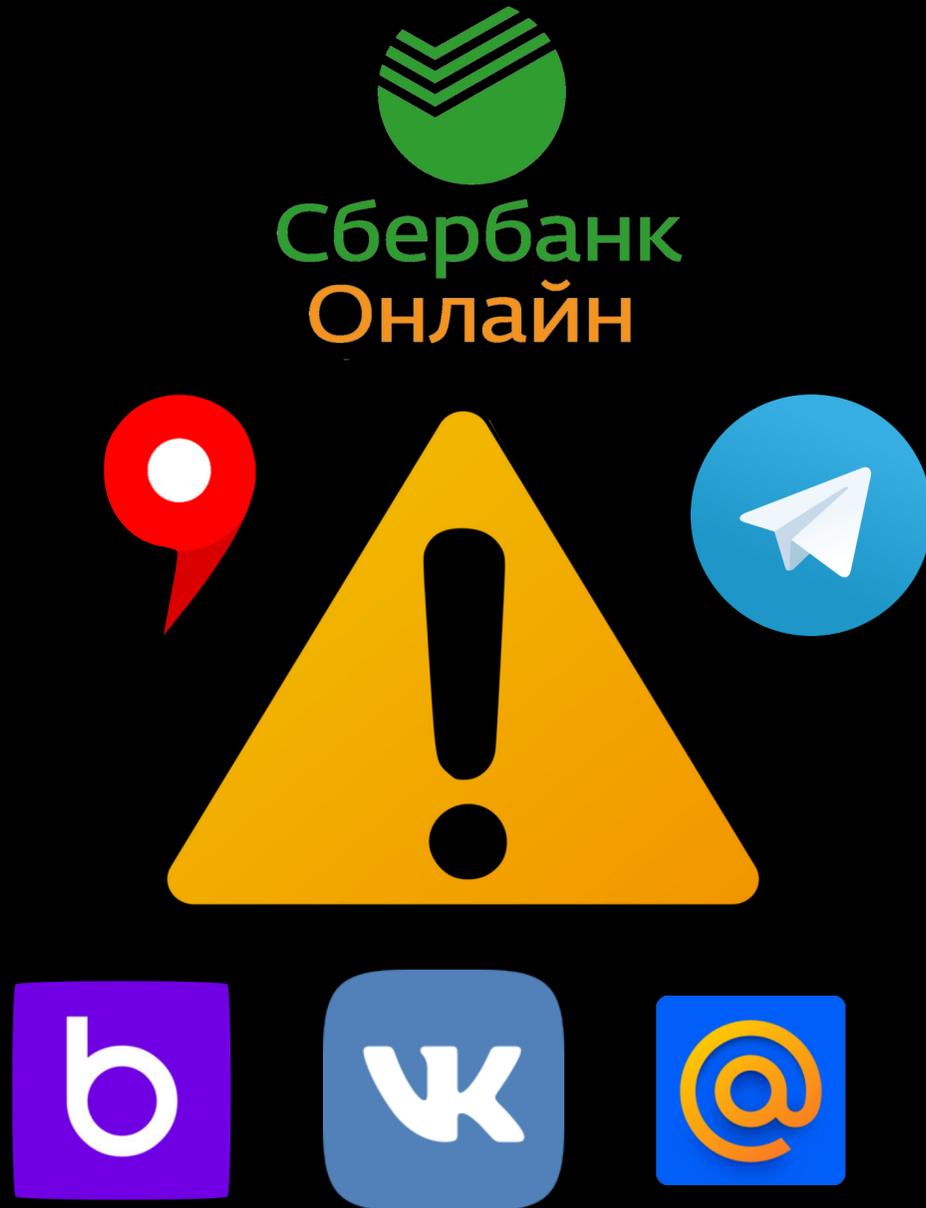
iOS. Проанализировать поведение приложения при memory warning

Estimate: 5-8

Priority: Major 



# Low Memory Warning



- © Если мало памяти, iOS шлёт `memory warning` в запущенное приложение
- © Система принудительно завершает процесс приложения, если память продолжит выделяться
- © Для пользователя — `crash`, для разработчика — не будет информации

# Технические детали



У нас есть очередь с операциями,  
где **текущая операция = формирование PDF-  
документа**

# Остановка очереди



# Самопроверка статуса

Текущая операция



операция должна  
проверить себя на отмену  
и остановиться

если не проверить —  
исполнение  
продолжится



Cancelled



Not Cancelled

# Игнорируем Low Memory Warning

Возникновение MemoryWarn... 

Отображение PDF документа

Получение данных →

Создание PDF документа → **Процесс продолжается** 

Формирование PDF докуме... →



All Heap & Anonymous VM

**Потенциальное падение приложения —  
система принудительно завершит процессы**

# Реагируем на Low Memory Warning

Возникновение MemoryWarn...



Получение данных



Создание PDF документа



Формирование PDF докуме...



Остановка  
процесса  
за 13ms



All Hear & Anonymous VM

## Ясный клиентский опыт



**Документ не загружен**

Воспользуйтесь услугой позднее или  
обратитесь в офис банка

ПОНЯТНО

# Частота проверки статуса

Текущая операция

как часто надо  
проверять на статус  
Cancelled?



Каждый грамм



Каждый килограмм





# Каждый грамм

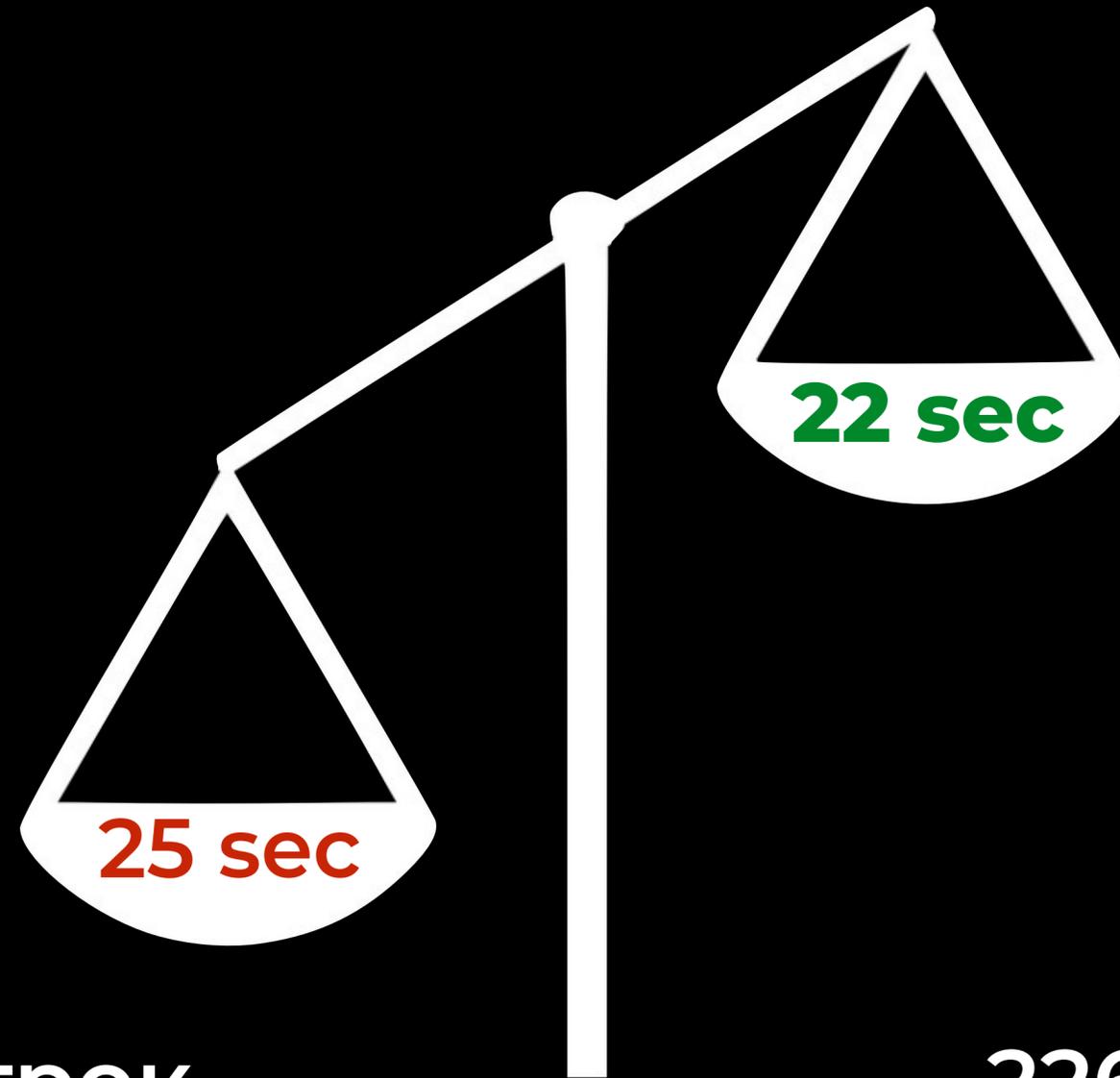
Построковая проверка статуса

# Каждый килограмм

Постраничная проверка статуса



# Время формирования PDF с проверкой статуса



10000 строк

226 страниц

Построковая  
проверка статуса

Постраничная  
проверка статуса

# Плюсы второго уровня



- ✓ **signpost** дополняет существующие решения
- ✓ С ним проще интерпретировать получаемые данные
- ✓ Самый простой вариант кастомизации инструментов



## Минусы второго уровня

- Не самое удобное использование
- Ручная разметка данных
- Потенциальные ошибки работы, как следствие человеческого фактора

# Что если нам этого недостаточно

- Нужно приложить много усилий чтобы завести signpost
- Хотим чтобы было по двойному клику и в космос 🚀





**Level**

**3**



**Задача:**

Сделать свой инструмент

**Узнаете:** как это можно сделать и когда это нужно делать

**Когда применять:** хотим автоматизировать логику - анализируем не сами, а за счет инструмента

**Показываем:** на примере unit-тестов проекта и сторонней библиотеки

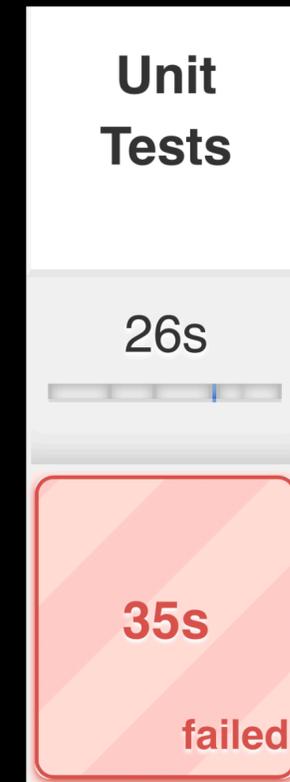
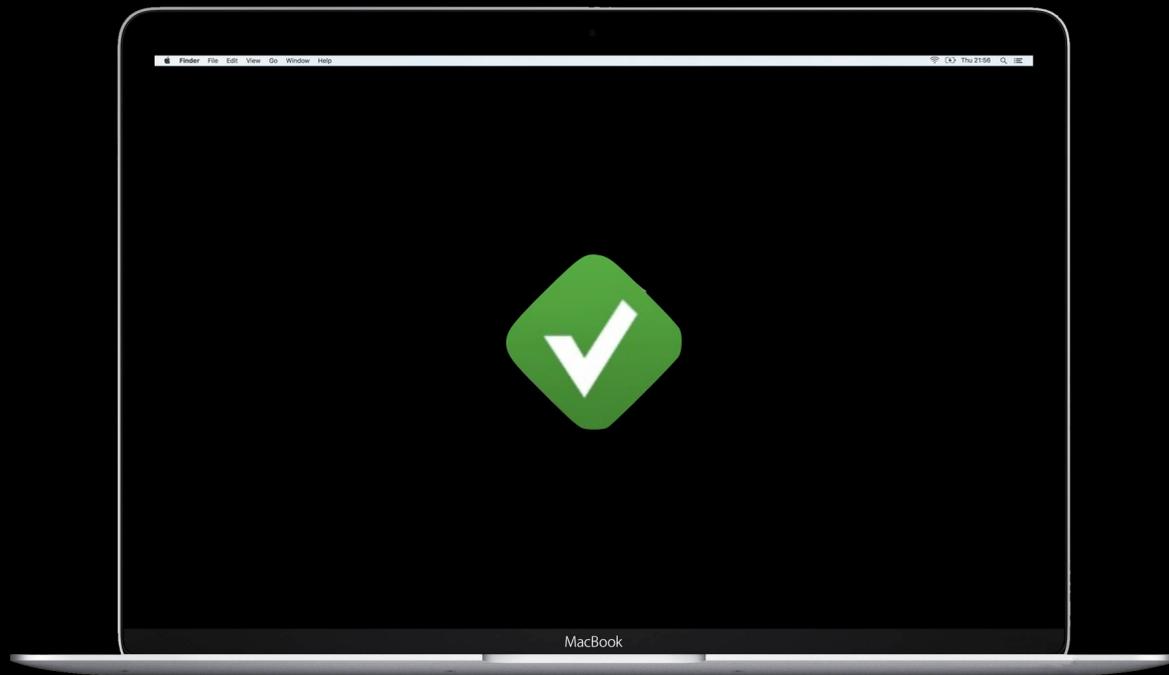
# Про unit-тесты в Сбербанк Онлайн

**41 000** unit-тестов  
в проекте

на **70** команд  
разработки iOS

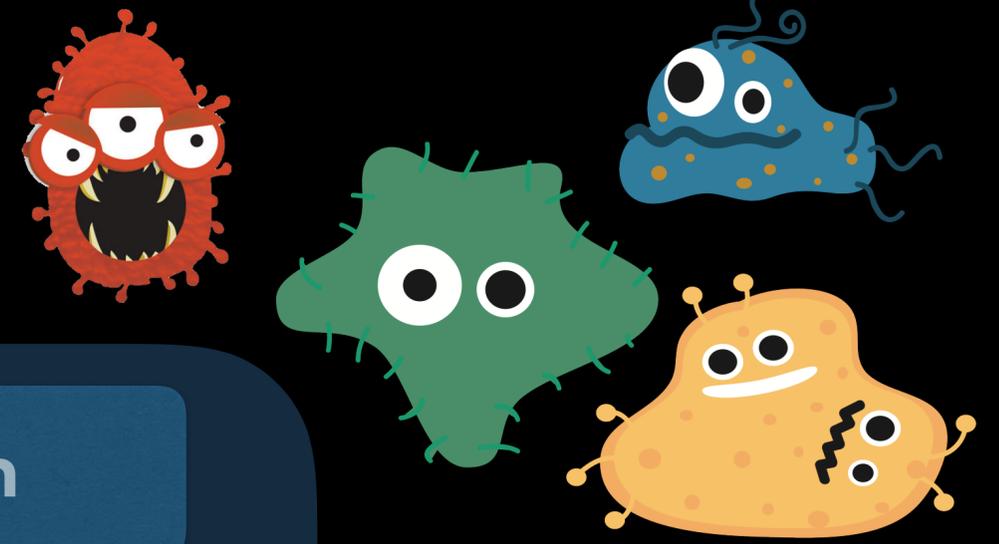
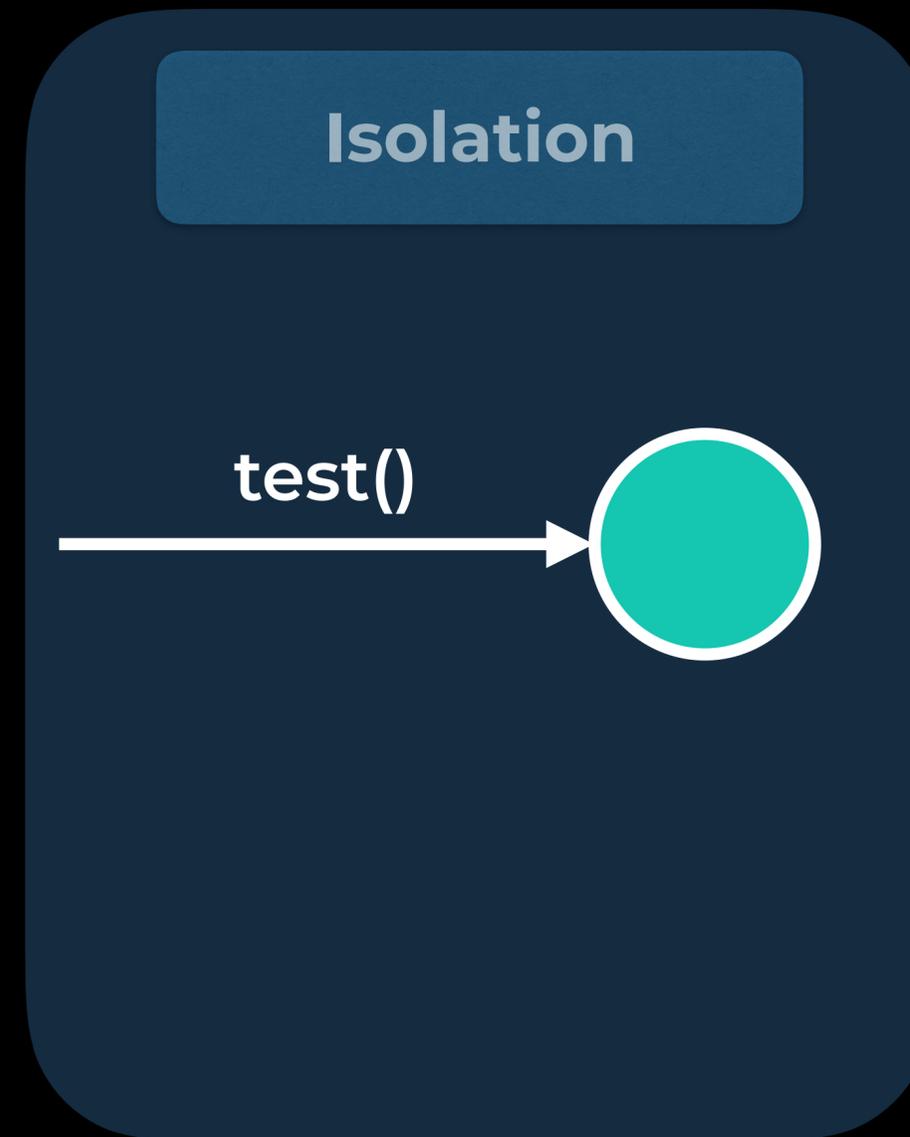
Когда у вас большая команда  
(у нас **~160 человек**)  
тесты очень важны

# Проблема на CI



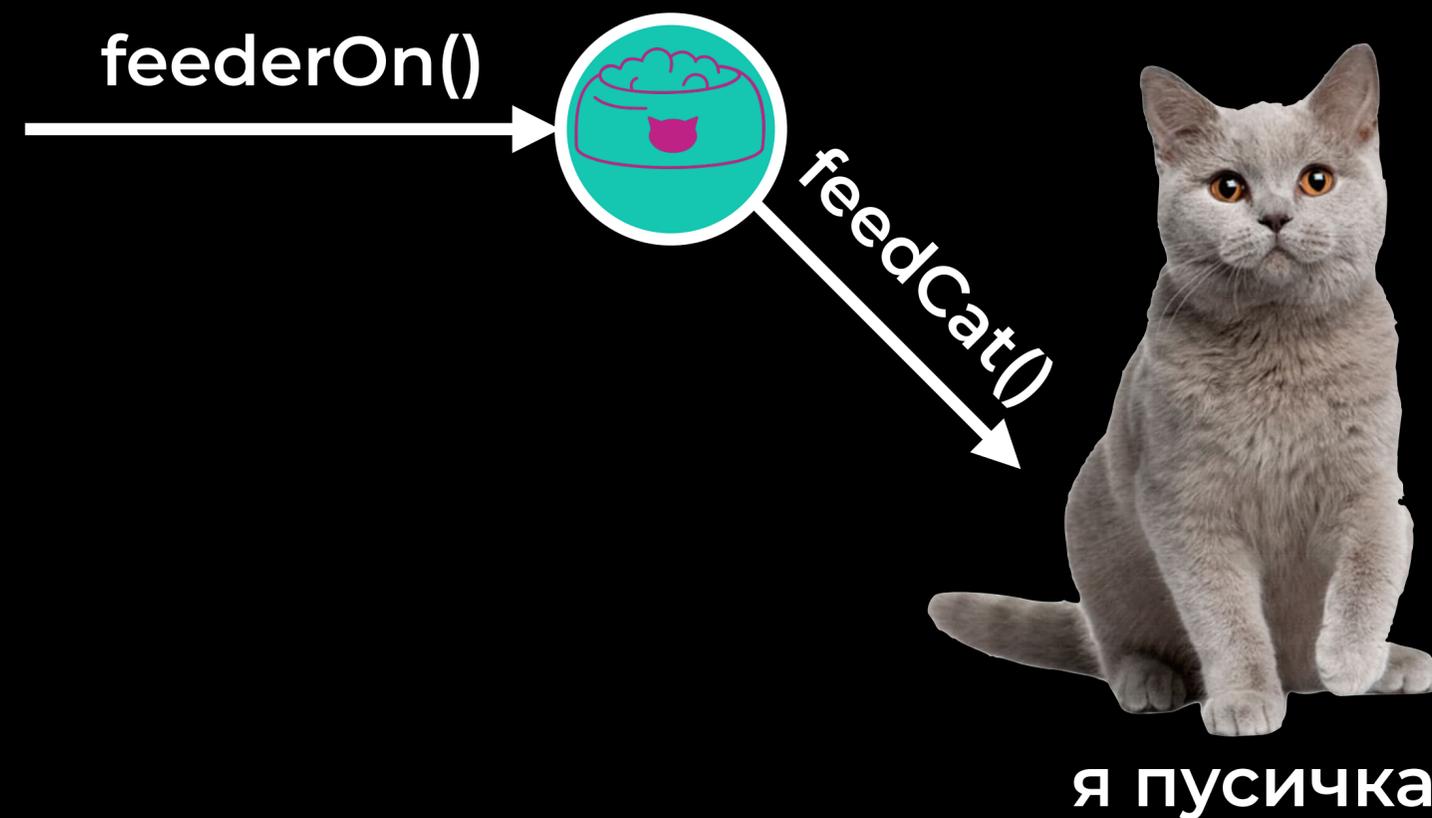
# Что такое unit-test?

Процесс, позволяющий проверить корректность исходного кода за счёт процедур использования и обработки



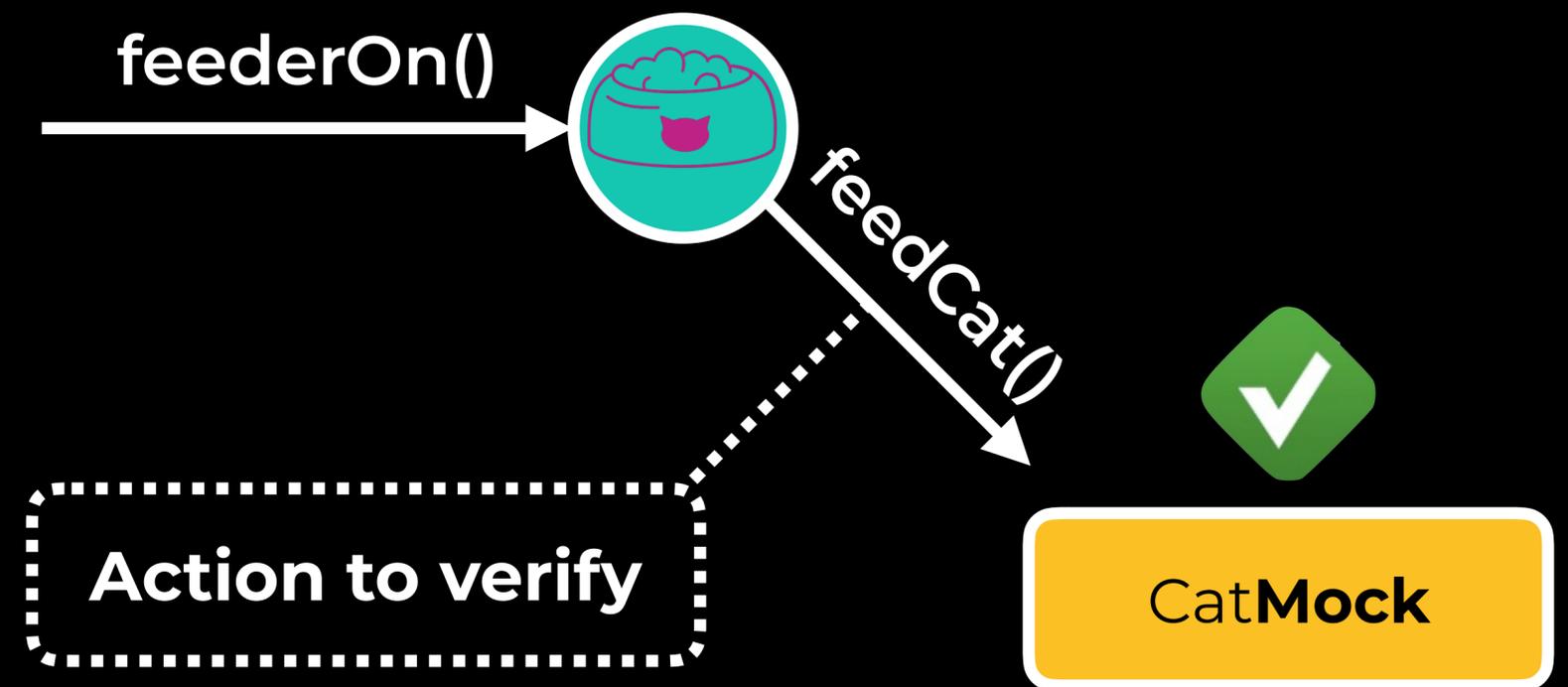
# Что такое Mock?

Моки «подсовываются» тестируемому объекту, чтобы можно было проверить, что тестируемый объект выполнил требуемые действия



# Что такое Mock?

Моки «подсовываются» тестируемому объекту, чтобы можно было проверить, что тестируемый объект выполнил требуемые действия



# OC Mock

## Mock objects for Objective-C

Stubs – return values for specific method invocations

Dynamic Mocks – verify interaction patterns

Partial Mocks – overwrite methods of existing objects

# Кейс: патчим стороннюю либу

iOS. Пофиксить OSMock и  
memory leaks

Estimate: >13

Priority: **Critical**



```
◇ @implementation ModuleTests
```

```
41 -  
42 - (void) setUp
```

```
43 {
```

```
44 >> [super setUp];
```

```
45 >> self.testModule = OCMPartialMock([SBOLModule new]);
```

```
46 }
```

```
47 -
```

```
48 - (void) tearDown
```

```
49 {
```

```
50 >> self.testModule = nil;
```

```
51 >> [super tearDown];
```

```
52 }
```

```
53 -
```

```
◇ - (void) testDemoMode
```

```
55 {
```

```
56 >> //arrange
```

```
57 >> SBFPParamsStorageSettings *paramsStorageMock = OCMClassMock([SBFPParamsStorageSettings class]);
```

```
58 >> OCMStub([self.sberAppMock paramsStorage]).andReturn(paramsStorageMock);
```

```
59 >> OCMStub(paramsStorageMock.demoMode).andReturn(@(YES));
```

```
60 -
```

```
61 >> OCMReject([self.transitionManagerMock resetState]);
```

```
62 >> OCMReject([self.appPermissionsServiceMock sendAnalyticsAfter:10]);
```

```
63 -
```

```
64 >> //act && assert
```

```
65 >> [self.testModule moduleAuthorizationStateDidChange:YES];
```

Создание мока

Уничтожение мока

Использование мока

# Получаем

unit-test

**Создание мока**  
- (void)setUp

**Завершение теста**  
- (void)tearDown

OSMock library

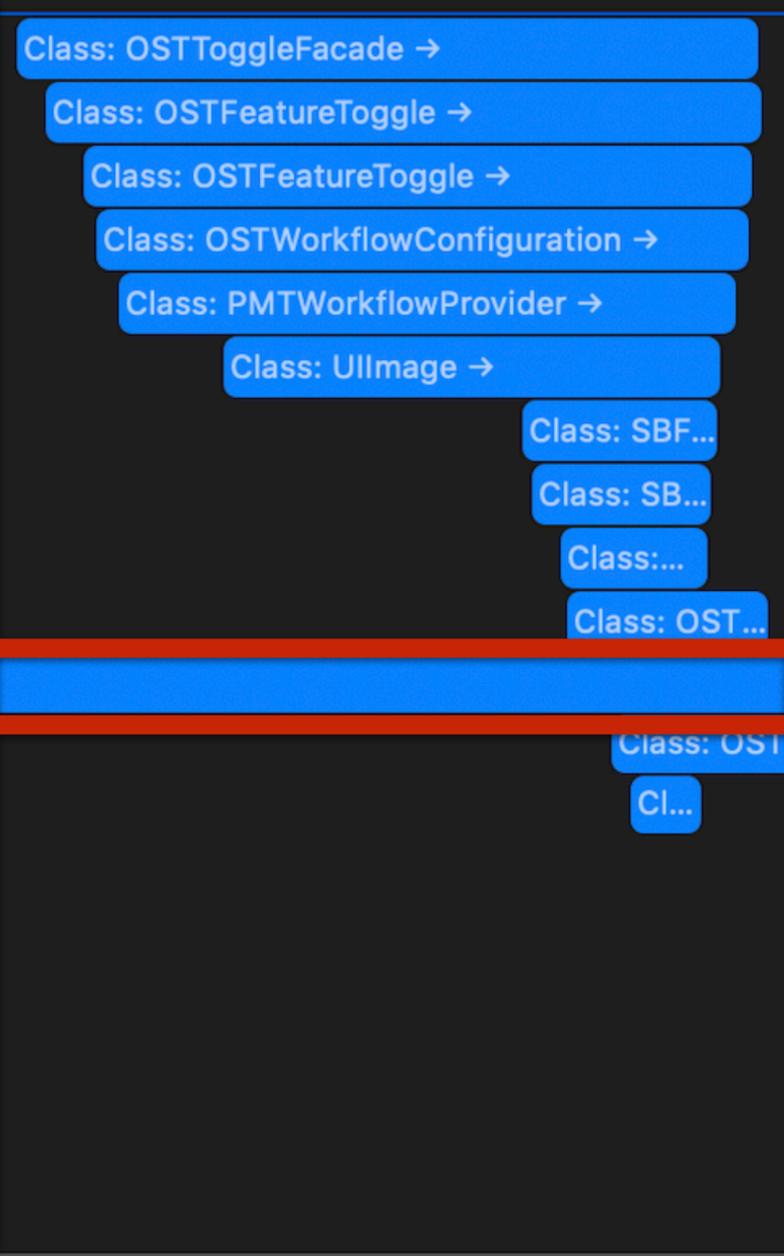
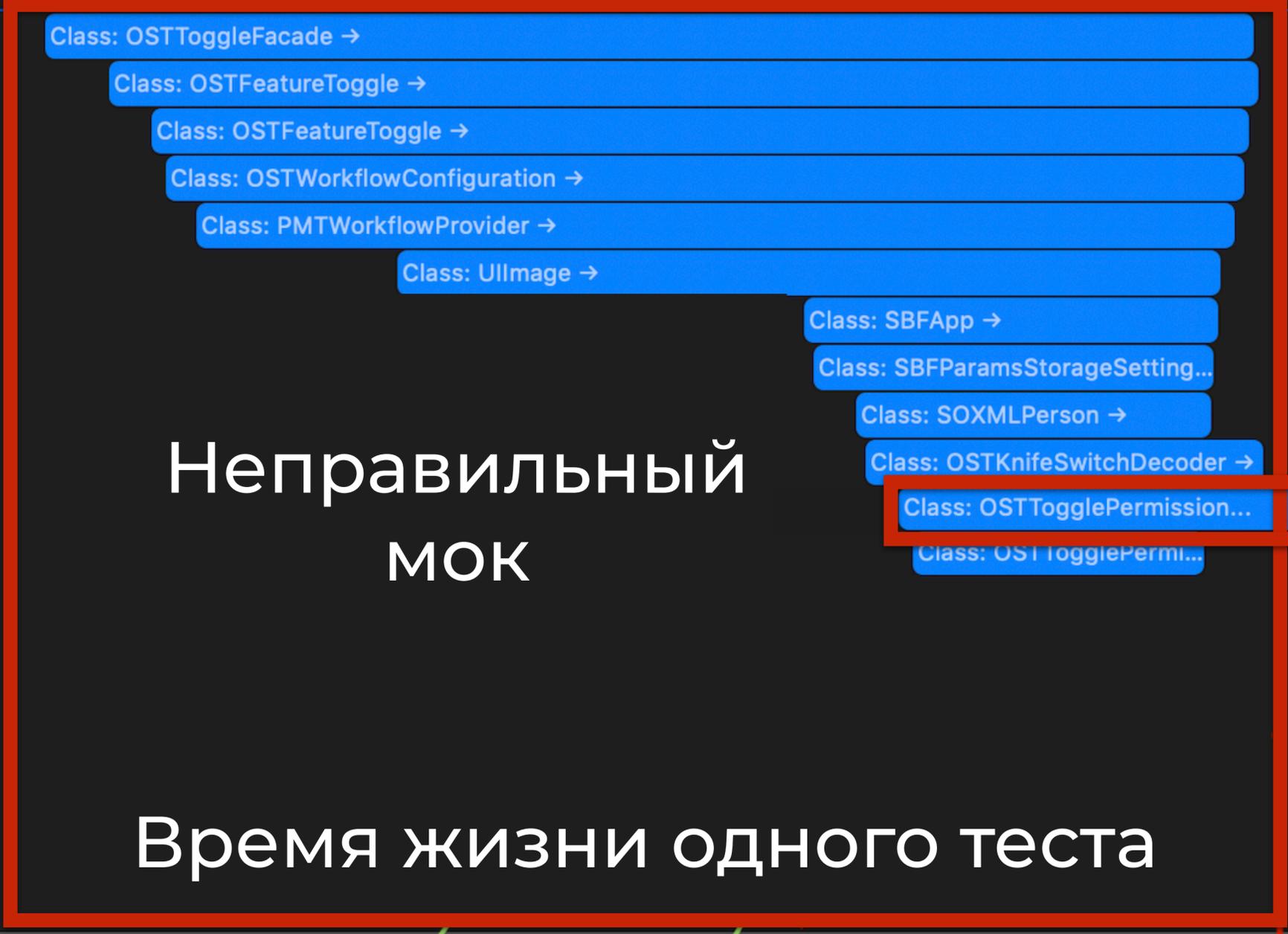
**Инициализируем моки**  
- (id)initWithClass:  
  (Class)aClass

**Уничтожаем моки**  
- (void)stopMocking

Ищем события, когда уничтожение  
происходит позже завершения теста

(No graphs) +

Mock



Test Name:-[OSTToggleFacadeTest testCardStatementEnabledV KnifeSwitchEnabled] → Name:-[OSTToggleFacadeTest testCardSt...

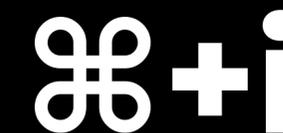
MockIsAlive > Результат

Время^ Описание результата

00:02.985.366 Мок 9241915778591403200 с типом OSTTogglePermission был создан в [OSTToggleFacadeTest testOperationsAndStatementsEnabledWithKnifeSwitchDisabled] и продолжает

00:02.985.366 Мок 9241915778591454848 с типом OSTTogglePermission был создан в [OSTToggleFacadeTest testCardStatementKnifeSwitchEnabledPermissionsDisabled] и продолжает ра

# Выбираем инструмент



Choose a profiling template for: iPhone SE (12.1) > Сбербанк

Standard

Custom

Recent

Filter



Blank



Activity Monitor



Allocations



Core Animation



Core Data



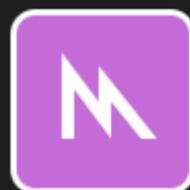
Counters



Energy Log



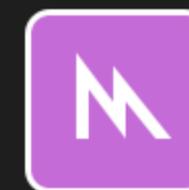
File Activity



Game Performance



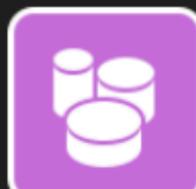
Leaks



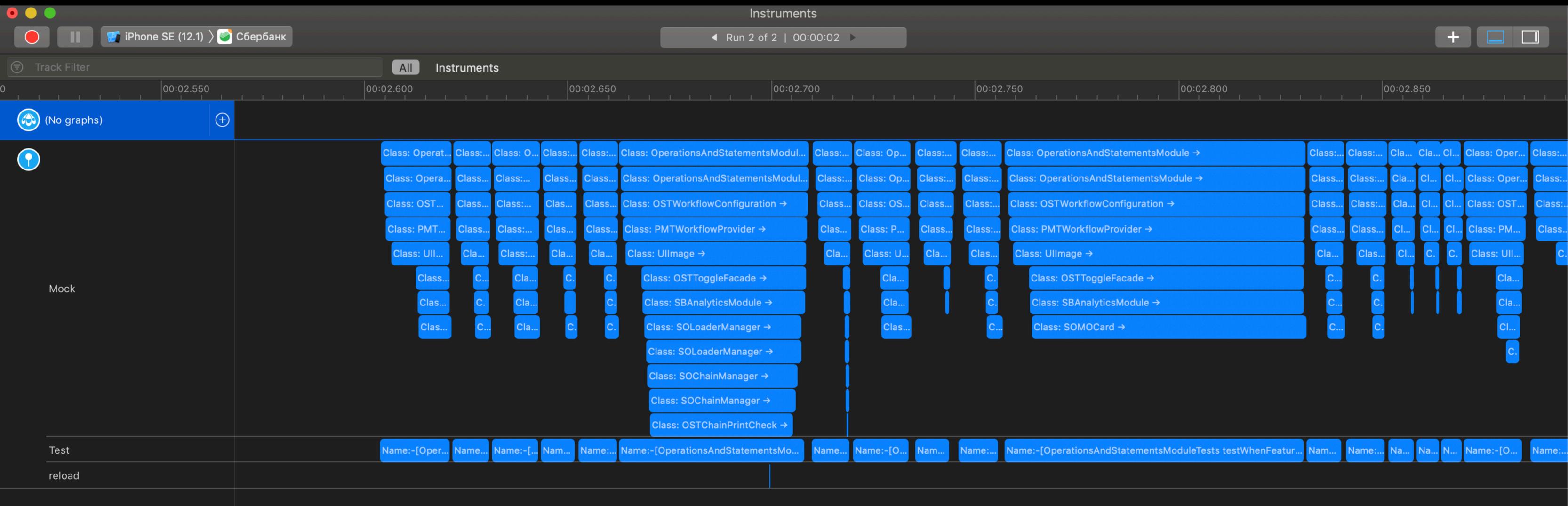
Metal System Trace



MockIsAlive



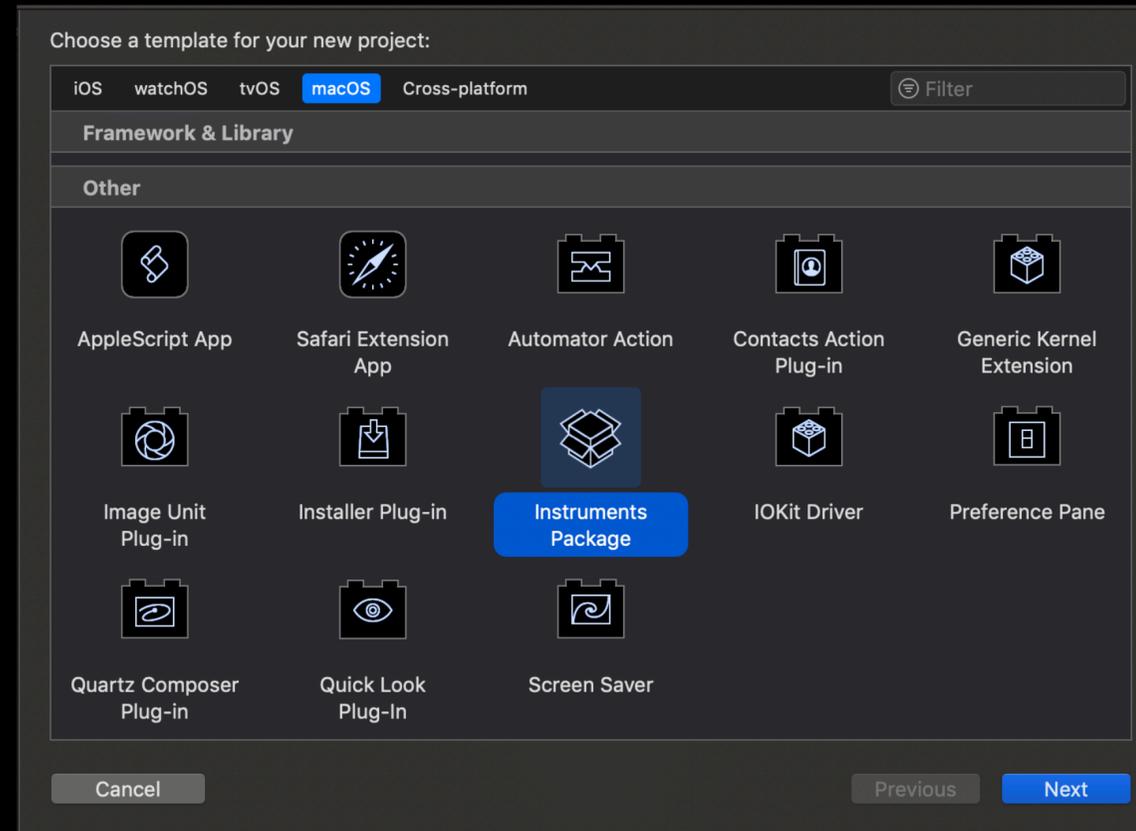
# Стало



No Data

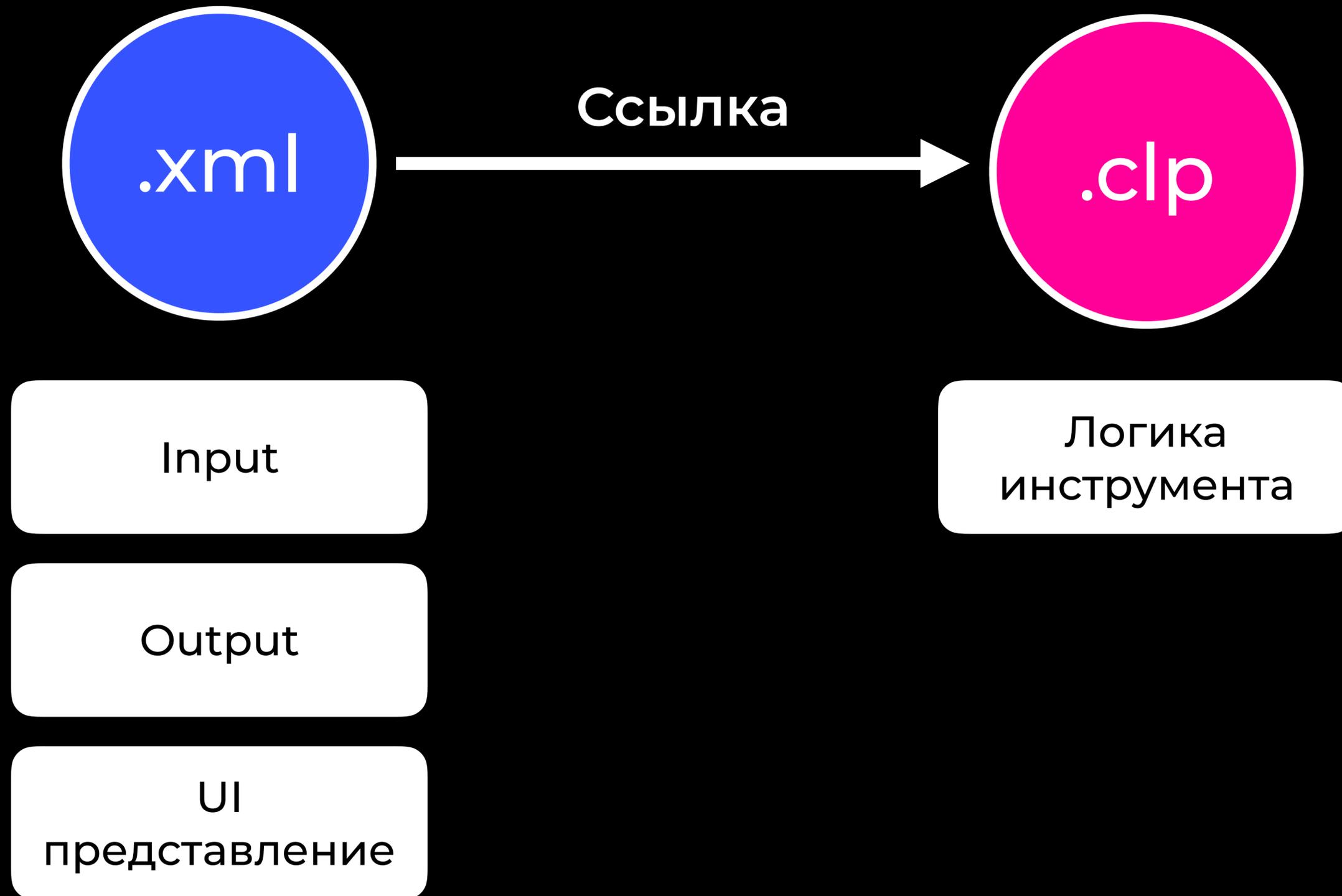
Поздравляем, моки не живут дольше одного теста!  
(Возможно инструмент настроен неправильно. Должны использоваться специальные версии библиотеки OCMock и класса XCTestCase, которые отправляют события signpost для инструмента)

# File -> New -> Project -> категория macOS -> Instruments package



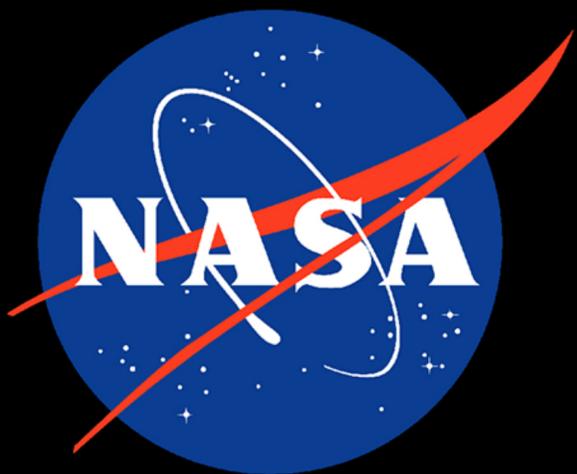
```
1 <?xml version="1.0" encoding="UTF-8" ?>
2 <package>
3   <id>none.HBInstrument</id>
4   <title>HBInstrument</title>
5   <owner>
6     <name>Victoria Kashlina</name>
7   </owner>
8
9   <!-- Instruments Developer Help: https://help.apple.com/instruments/developer/mac/current/ -->
10
11   <!-- MARK: Schema Definitions -->
12   <!-- Define point and interval schemas needed to represent the input and output tables your package will use. -->
13 </package>
```

# Компоненты инструмента



# С Language Integrated Production System

программная среда  
для разработки  
экспертных систем



Правило 1:

**ЕСЛИ**

(выполняются условия 1)

**ТОГДА**

(выполнить действия 1)

Правило 2:

...

# СУЩНОСТИ В CLIPS

(deftemplate **unitTest**

(**slot** time (**type** INTEGER))

(**slot** signpost id (**type** INTEGER))

(**slot** testName (**type** STRING))

)

(структура **unit-теста**

(**атрибут:** время старта

(**атрибут:** идентификатор

(**атрибут:** имя теста

)

Код инструмента

# Роль signpost

```
defrule MODELER:record-test-started
  (os-signpost (subsystem "com.example")
   (category "UnitTestLog")
   (name "Test")
   (event-type "Begin")
   (message$ "Name:"
    ?testName)
   (time ?time)
   (identifier ?identifier))
  =>
  (assert (unitTest
   (time ?time)
   (signpost-id ?identifier)
   (testName ?testName)))
)
```

задаём правило  
(когда ловим signpost)  
(с категорией)  
(именем "Test")  
(типом-ивента "Begin")  
(достаём данные:  
?имя Теста)  
(?время старта теста)  
(?идентификатор))  
=> то  
(записываем их (в unitTest  
(время ?time)  
(идентификатор ?identifier)  
(имя теста ?testName)))  
)

# Правило инструмента: “ЕСЛИ”

defrule **RECORDER::record-mock-started-before-unit-test-started**

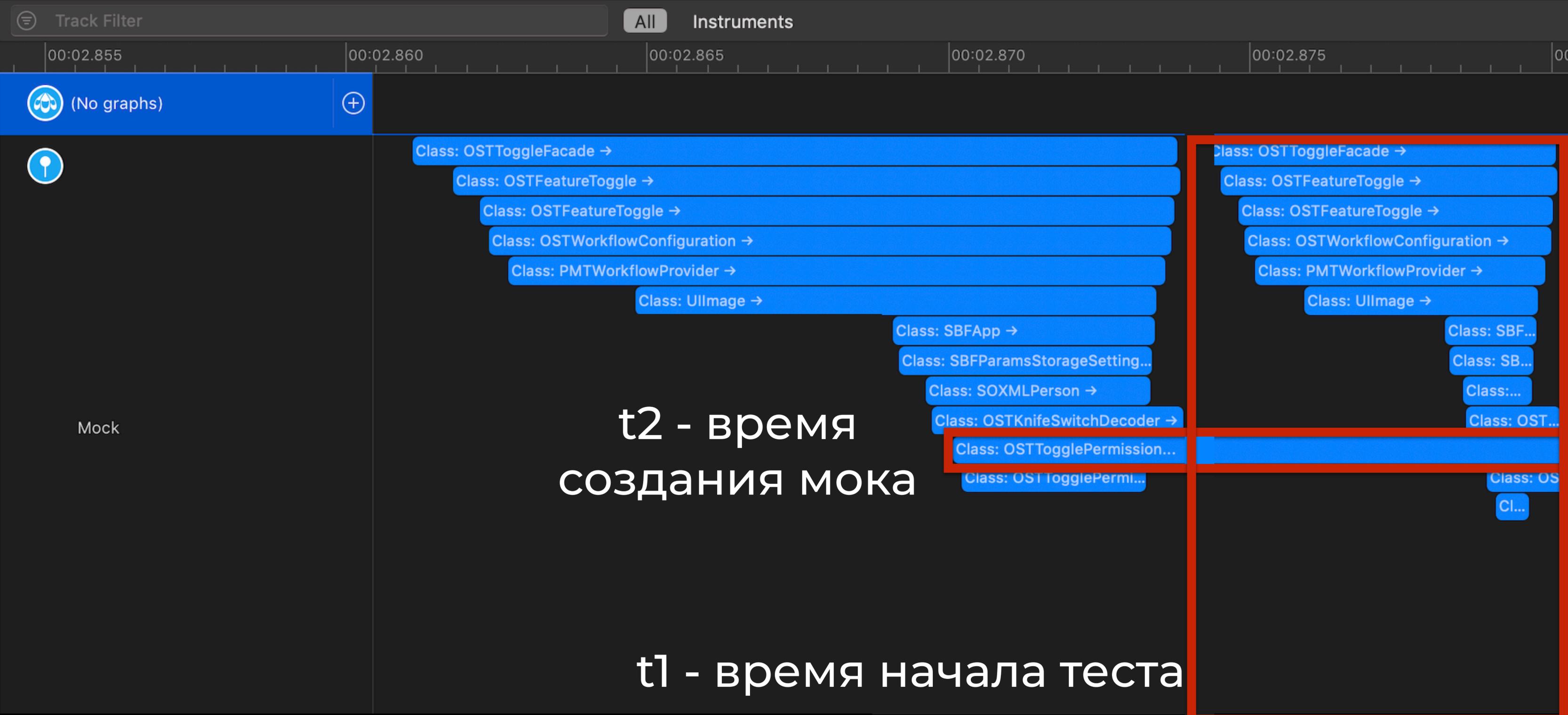
1 (unitTest (time ?t1) (testName ?leakedTestName))

(mock (time ?t2) (signpost-id ?signpost-id) (mockedClass ?  
mockedClass) (createdTestName ?createdTestName

2 (test (> ?t1 ?t2))

Если существует unitTest и mock

И, если при этом начало теста наступает позже существующего мока



На момент завершения теста есть “живой” мок-объект

# Таблица событий

3

(table (table-id ?output) (side append))

(table-attribute (table-id ?output) (has schema detected-mocks-narrative))

Если существует таблица для хранения результатов со схемой detected-mocks-narrative

# Правило инструмента: “ТО”

4 (create-new-row ?output)

5 (set-column time ?t1)

6 (set-column-narrative description "Мок %string% с типом %string% был создан в %string% и продолжает работать в %string%. Необходимо устранить причину удержания мока между тестами, либо явно вызвать stopMocking" ?signpost-id ?mockedClass ?createdTestName ?leakedTestName )

То создаём новую запись в таблице

То заполняем новую запись временем

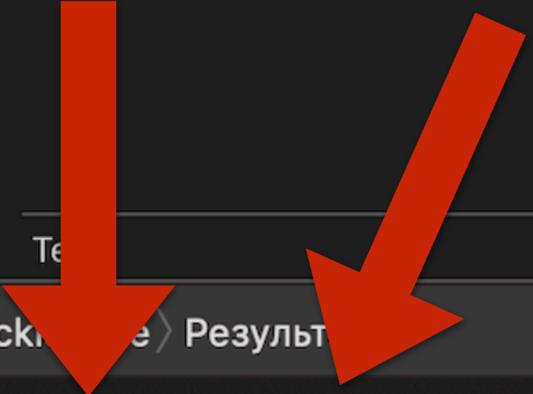
То дополняем новую запись описанием

(No graphs) +

Mock



# Время Описание

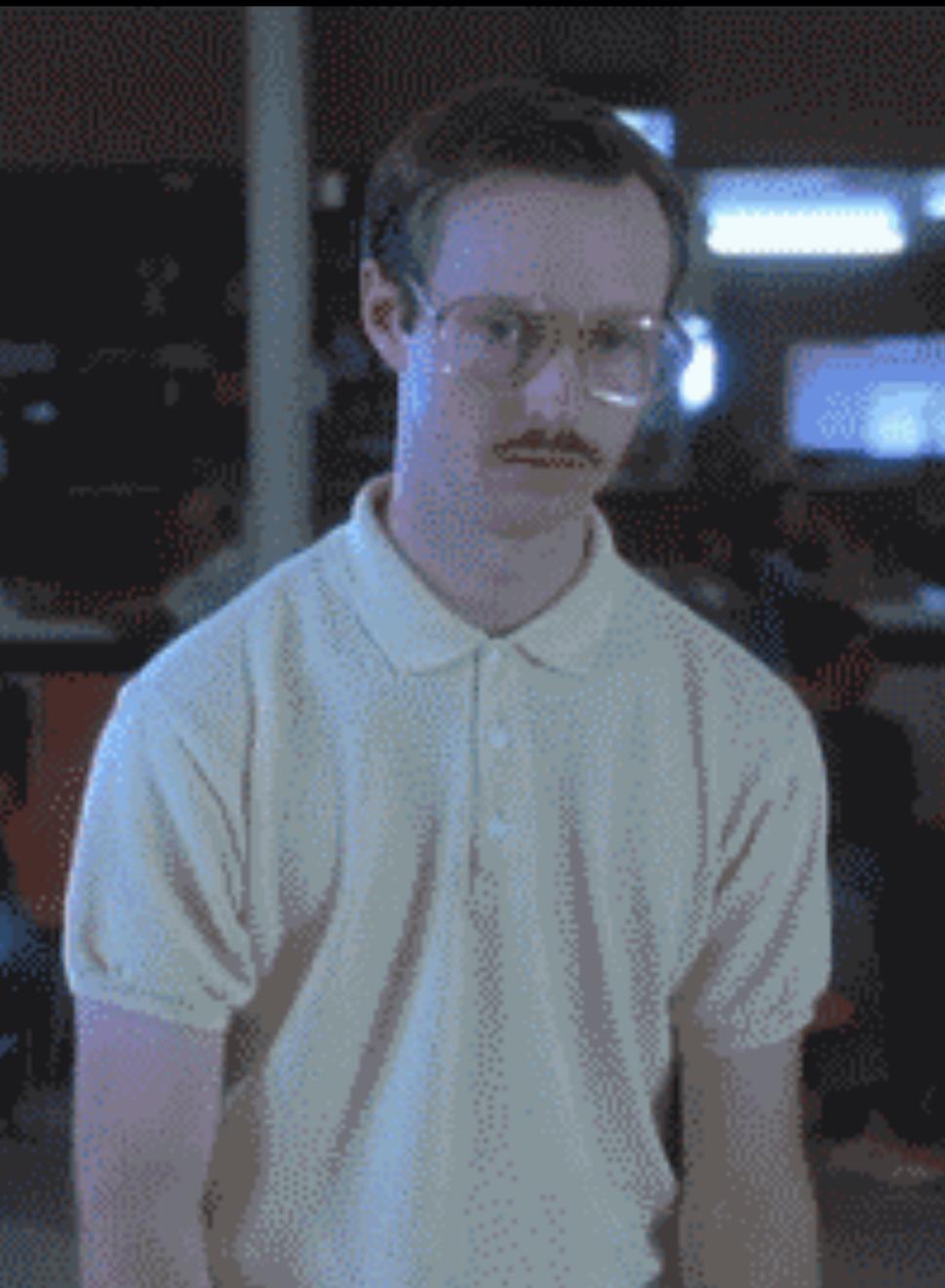


Mocking Result

Время^ Описание результата

00:02.985.366 Мок 9241915778591403200 с типом OSTTogglePermission был создан в [OSTToggleFacadeTest testOperationsAndStatementsEnabledWithKnifeSwitchDisabled] и продолжает

00:02.985.366 Мок 9241915778591454848 с типом OSTTogglePermission был создан в [OSTToggleFacadeTest testCardStatementKnifeSwitchEnabledPermissionsDisabled] и продолжает на



## Плюсы третьего уровня

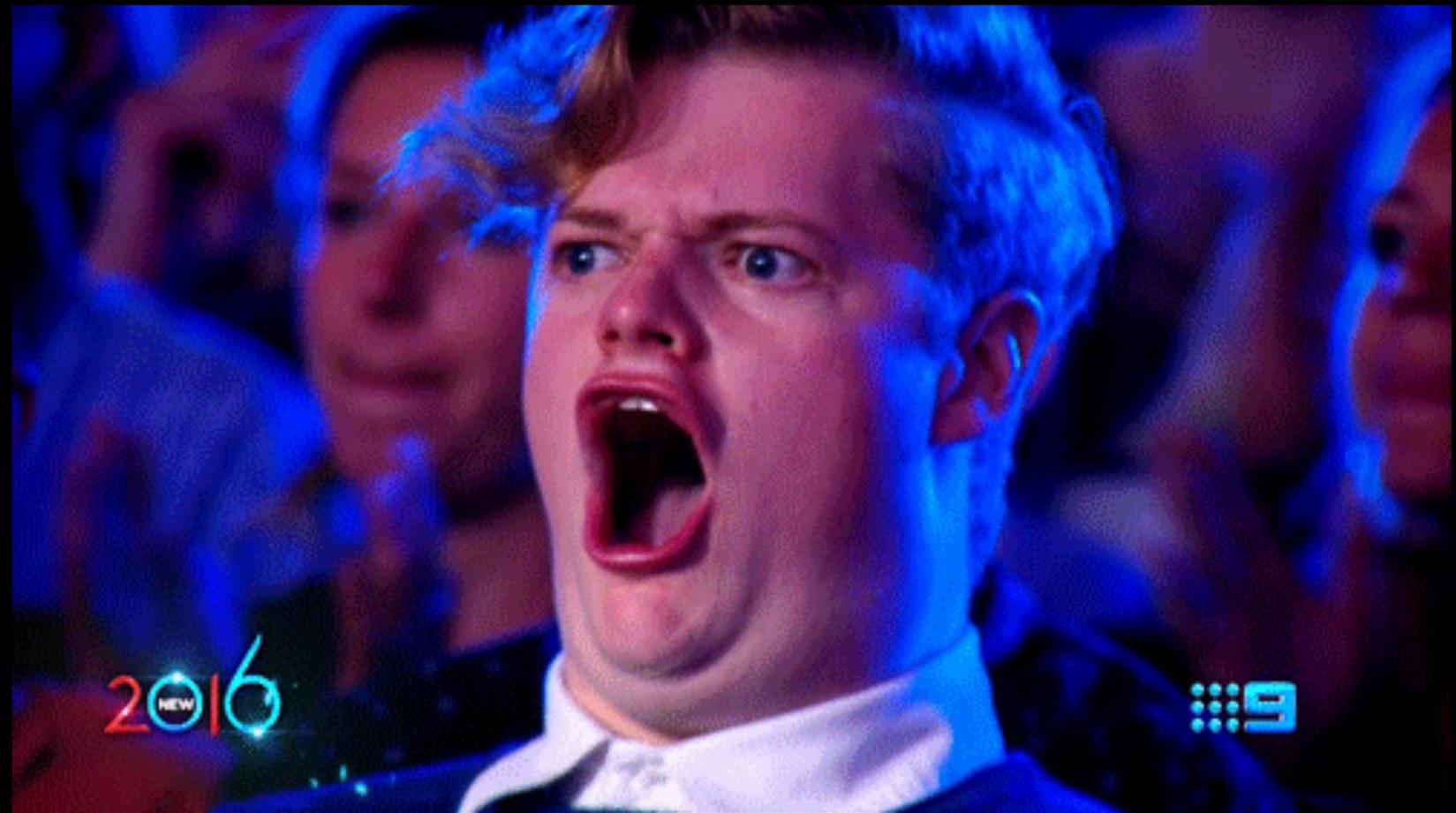
- ✓ Решение, которое работает из коробки
- ✓ Легко масштабировать на большую команду

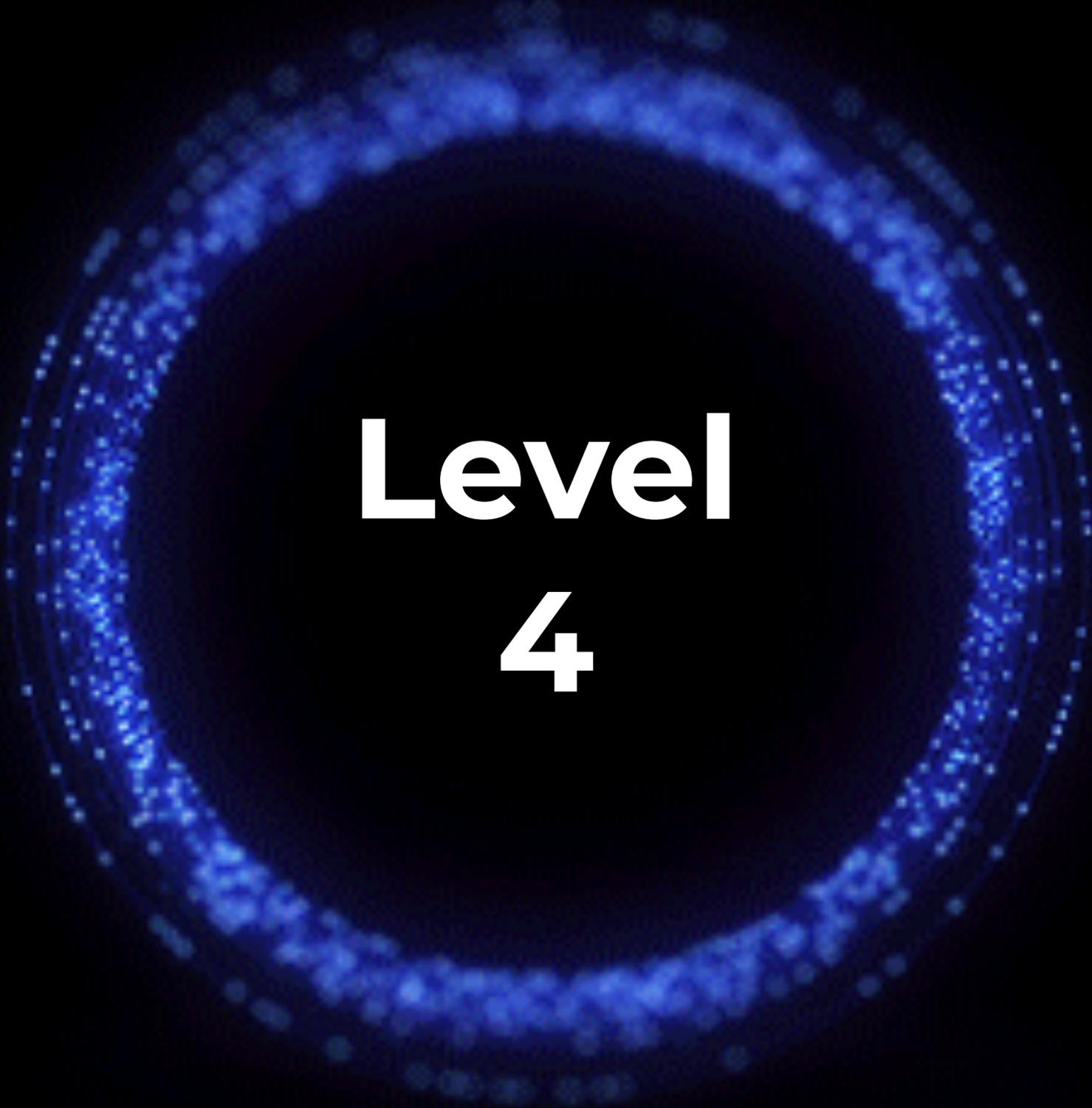


## **Минусы третьего уровня**

- Трудозатратное решение**
- Требуется поддержки по мере изменения проекта**
- Не факт, что инструмент использует каждый член команды**

Но что, если...  
нам хочется еще больше





**Level**

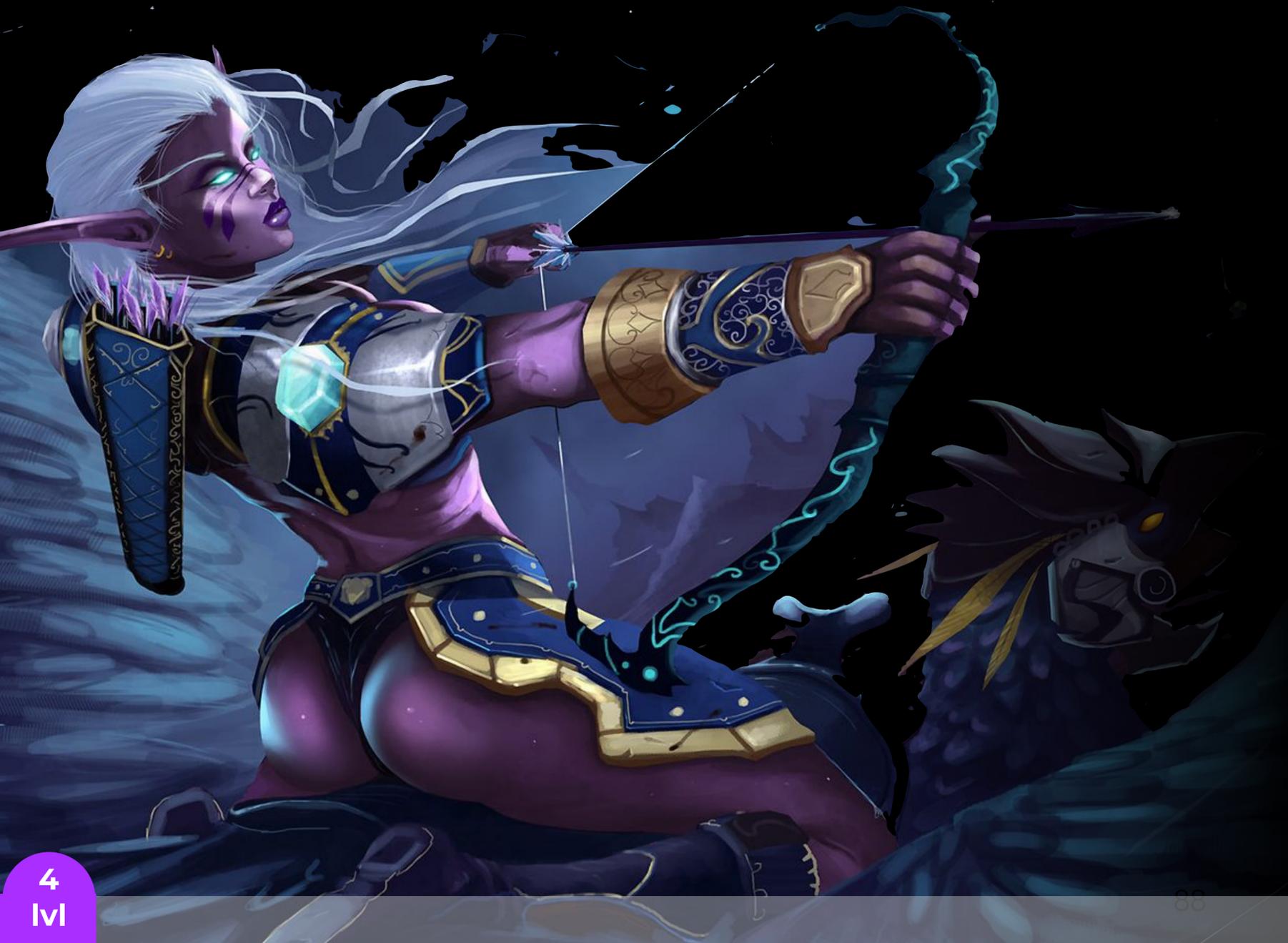
**4**

**Задача:**

Понять, как всё  
автоматизировать

**Узнаете:** как это всё  
можно автоматизировать  
на СІ

**Когда применять:** ХОТИМ  
регулярно проводить  
проверки без  
дополнительных усилий



# Кейс: автоматизация инструмента

iOS. Встроить инструмент в прогоны на CI

Estimate: 15

Priority: Minor 



# Запуск консоли

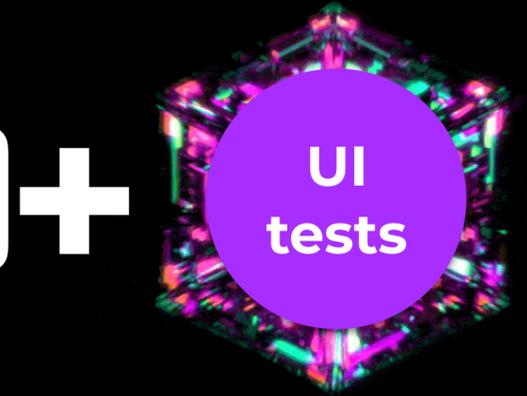
```
instruments -t Leaks -w 'iPhone 7' SberbankApp.app
```



Сбербанк  
Онлайн

# Запуск консоли

```
instruments -t Leaks -w 'iPhone 7' SberbankApp.app
```



XCTest



Сбербанк  
Онлайн



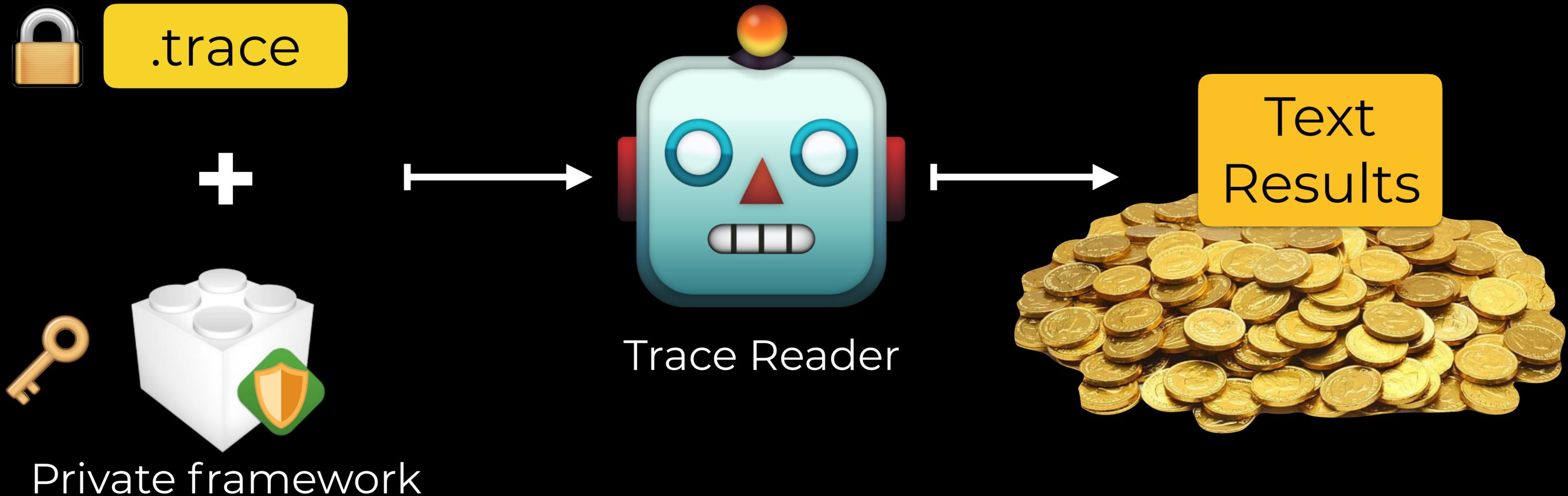
NSKeyedArchive



Trace  
Document



# Интерпретация результатов



# Отображение результатов

## Stage Logs (Instruments result) ✕

Shell Script -- ~/XcodeTraceReader -path /Users/cab-sa-ci000091/leaks.trace -binaryName Loyalty\_ (self time 2s)

```
+ /Users/cab-sa-ci000091/XcodeTraceReader -path /Users/cab-sa-ci000091/leaks.trace -binaryName Loyalty
```

```
CoreData: annotation: Failed to load optimized model at path '/Applications/Xcode.app/Contents/Applications/Instruments.app/Contents/Frameworks/InstrumentsPackaging.framework/Versions/A/Resources/XRPackageModel.momd/XRPackageModel 9.0.omo'
```

⚠️ В модуле 'Loyalty' есть 8 протечек памяти, смотри экземпляры с типом: ["LTYScreenRouter", "LTYUserInfoService", "LTYBalanceService", "LTYLevelsService", "LTYLocationService", "LTYLocationService", "LTYPartnersService", "LTYMainViewController"].

Instruments  
result

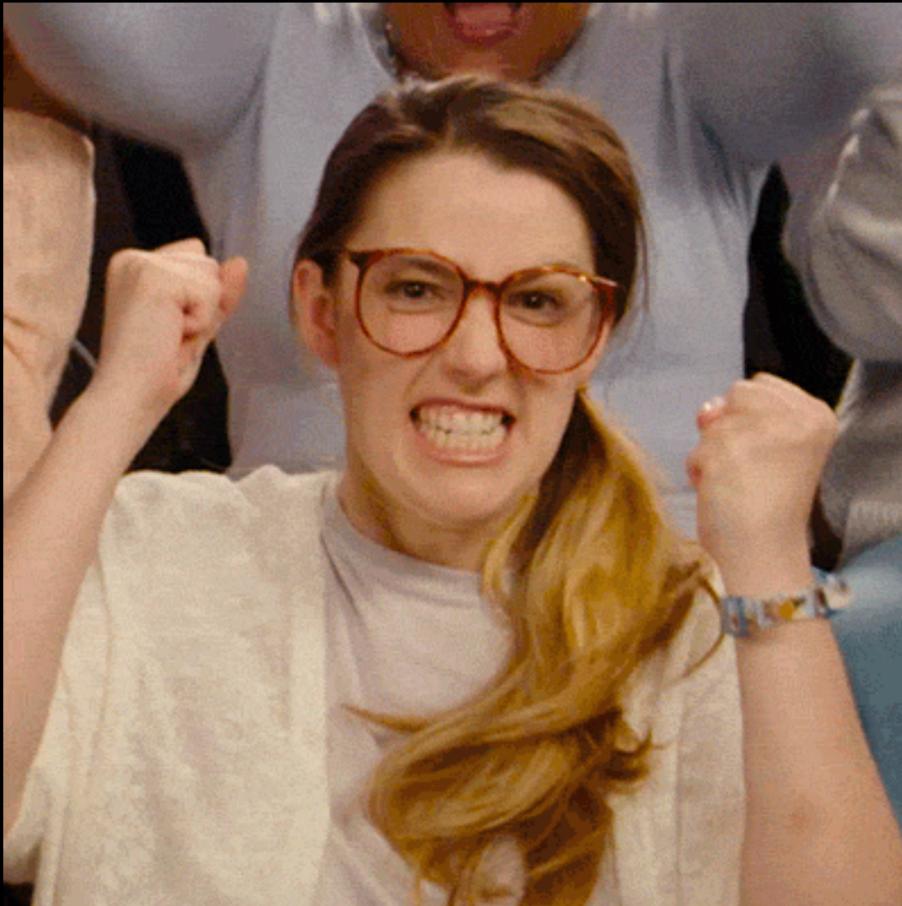
2s

2s

failed

2s

# Плюсы четвёртого уровня



- ✓ Будет автоматизировано и с периодическими прогонами
- ✓ Сделали один раз – получили долгосрочный профит на многих pull-request'ах
- ✓ Уменьшение ошибок в коде
- ✓ Выявление багов на ранних стадиях разработки



## **Минусы четвёртого уровня**

- **Нужны навыки DevOps'a**
- **Риски работы с приватными фреймворками**
- **Требуется много сил на последующую поддержку**

# Выводы

1. Ручного прогона может не хватать
2. Есть дополнительные нюансы, которые нельзя учесть глазами
3. Много пользователей — больше уникальных кейсов, которые не выявить ручным прогоном
4. Использование инструментов доступно всем
5. Есть возможность автоматизировать проверку

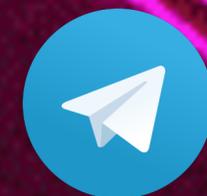
# Немного ссылок



- 1 <https://help.apple.com/instruments/developer/mac/current/#/devcd5016d31>  
(Instruments Developer Help)
- 2 <https://habr.com/ru/company/sberbank/blog/443414/>  
(Signpost: когда брейкпоинтов недостаточно, Антон Власов, Кашлина Виктория)
- 2 <https://developer.apple.com/videos/play/wwdc2018/405/>  
(WWDC, Measuring Performance Using Logging)
- 2 <https://www.swiftbysundell.com/daily-wwdc/getting-started-with-signposts>  
(Getting started with signposts, John Sundell)
- 3 <https://developer.apple.com/videos/play/wwdc2018/410/>  
(WWDC, Creating Custom Instruments)
- 3 <https://habr.com/ru/company/sberbank/blog/447594/>  
(Custom instruments: когда signpost недостаточно, Антон Власов, Кашлина Виктория)
- 4 <http://jamie-wong.com/post/reverse-engineering-instruments-file-format/>  
(Reverse Engineering Instruments' File Format, Jamie Wong)

**EXTRA LEVEL**

**Спасибо  
за внимание!**



**@kindredqb  
@VlasovAnt**