
Vấn đề bộ nhớ trong lập trình iOS

Phát hiện vấn đề

- Một app tốt không chỉ chạy đúng, còn phải chạy nhanh, tiêu tốn ít tài nguyên
- Trong quá trình làm dự án, có nhiều vấn đề trở thành nguy cơ tiềm ẩn
- 2 vấn đề cần quan tâm:
 - **Abandoned Memory**
 - **Leaks Memory**

Abandoned Memory

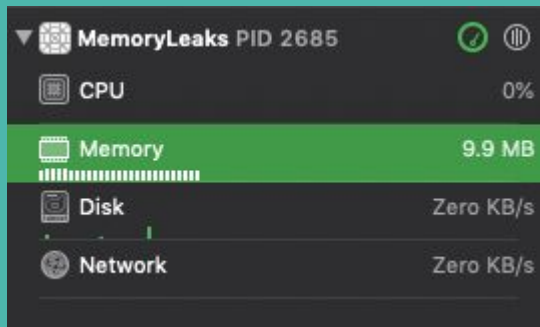
Abandoned Memory là gì?

- Abandoned là vùng nhớ được cấp phát để lưu thông tin, sau đó vùng nhớ này không còn được sử dụng nữa, tuy nhiên vùng nhớ vẫn tồn tại, và vẫn truy cập bình thường
- Abandoned memory xảy ra khi bộ nhớ không ngừng tăng lên khi thực hiện lặp đi lặp lại một chuỗi các thao tác

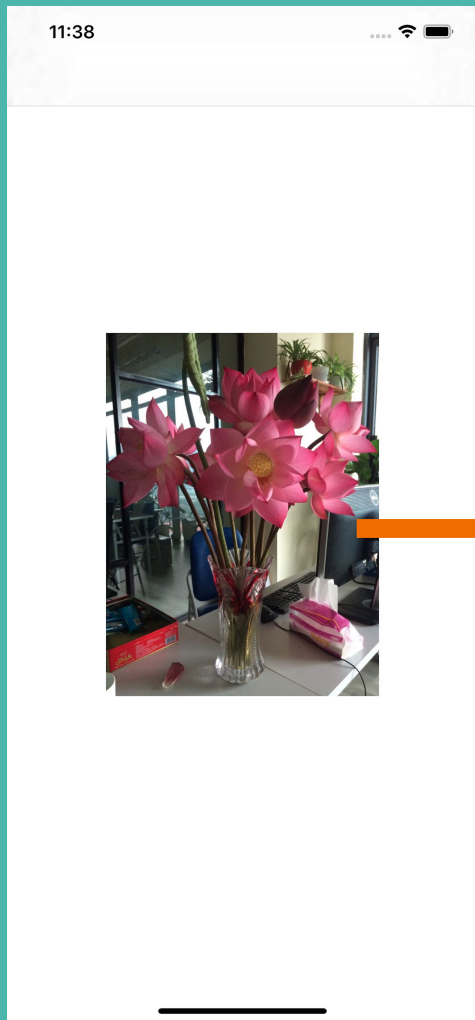
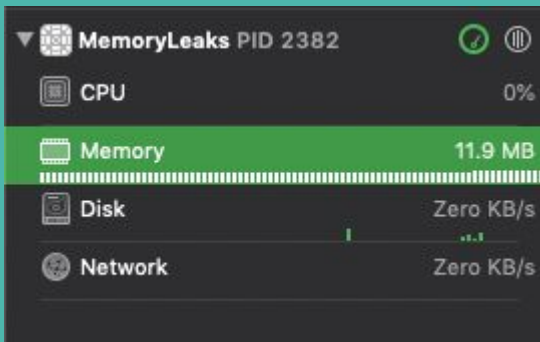
Ví dụ:

- Khi chuyển từ **ViewController A** -> **ViewController B**. Một số object được khởi tạo và cache lại. Khi back lại **ViewController A** thì các object không còn cần thiết nữa thì cần clear cache đi. Tuy nhiên, nếu không thực hiện clear cache, khi thực hiện lặp đi lặp lại các thao tác: **ViewController A** -> **ViewController B** -> **ViewController A** thì bộ nhớ sẽ liên tục tăng.

Trước:



Sau:



Công cụ: Allocation Instrument

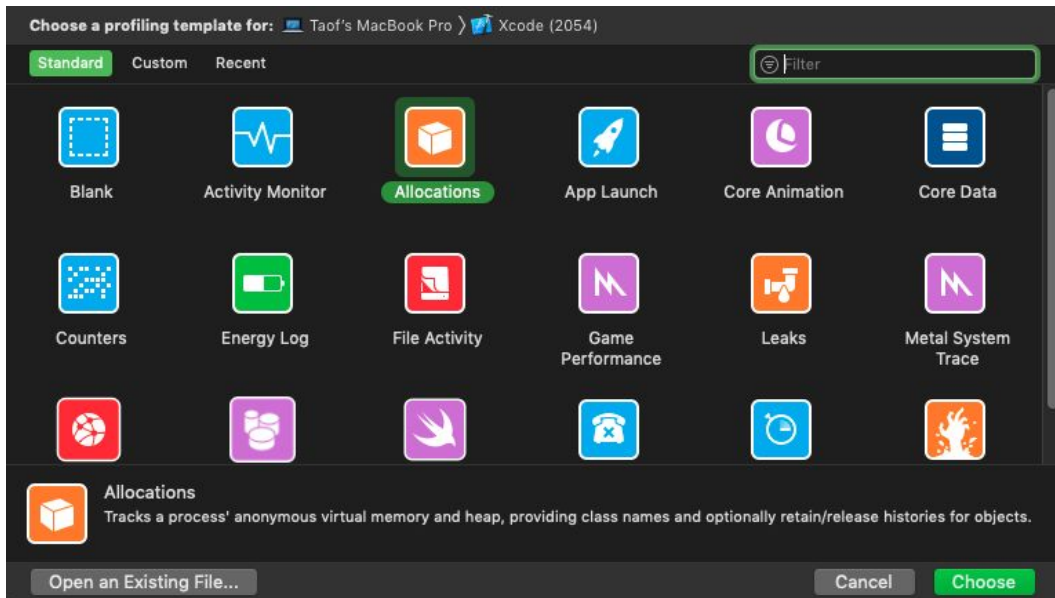
Allocation template quét thông tin vùng heap memory và track lại các thông tin allocations.

Flow chung:

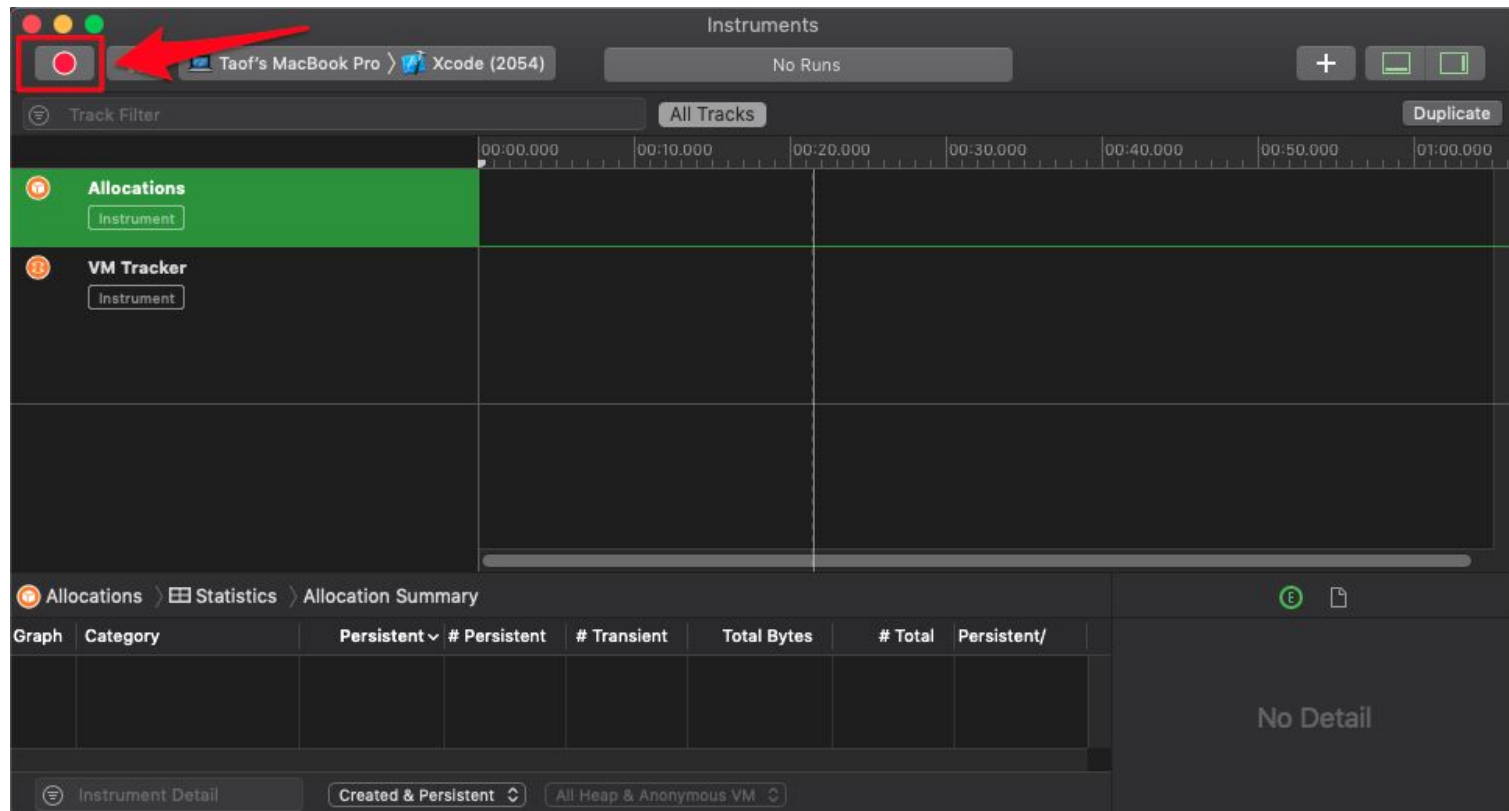
1. Từ trạng thái ban đầu track lại thông tin allocations
2. Thực hiện một số thao tác
3. Quay trở về trạng thái ban đầu và track lại thông tin allocations

Sử dụng Allocation Template

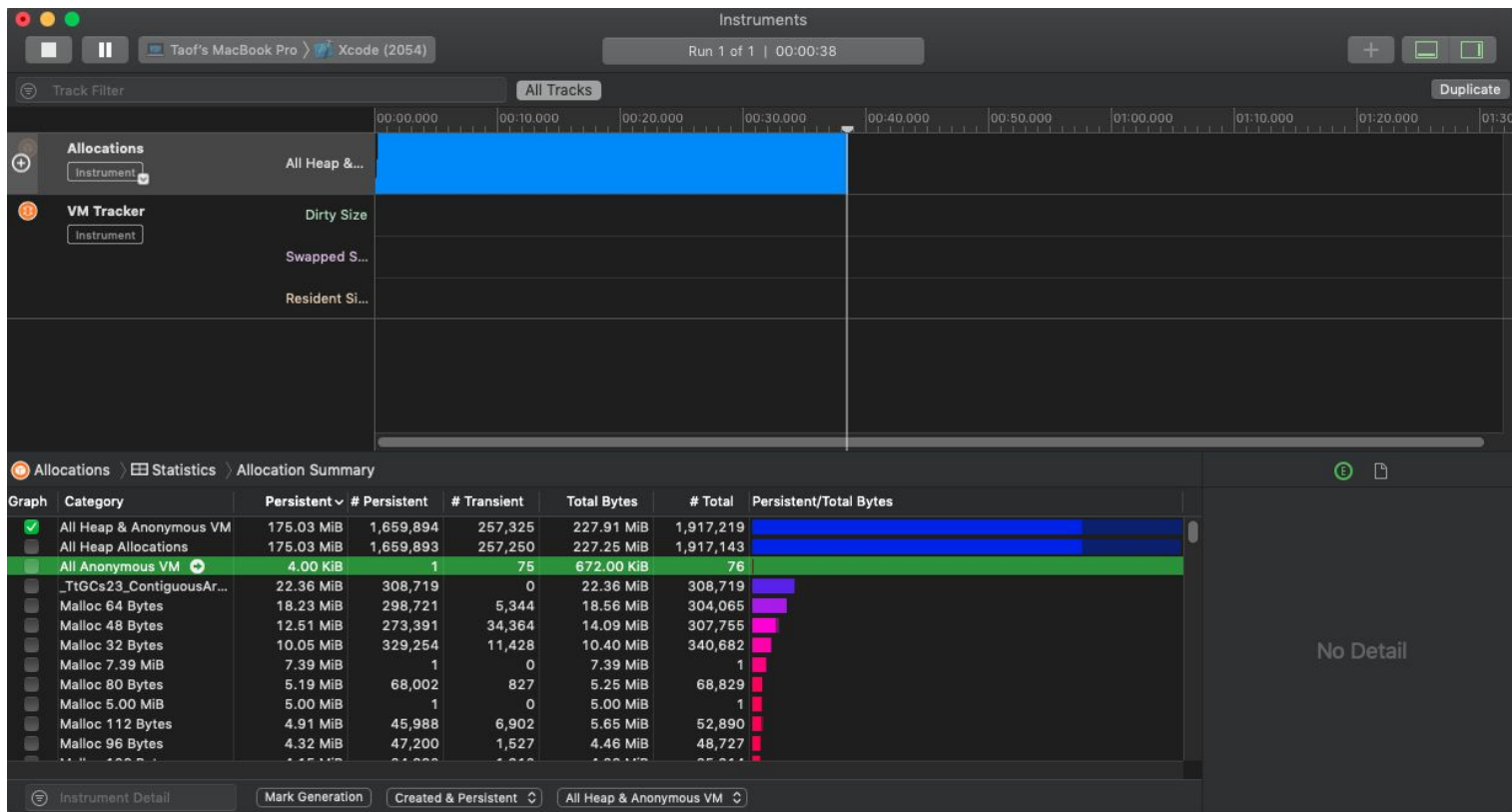
1. Xcode -> Open Developers Tool -> Instrument
2. Chọn Allocations



3. Chọn nút record màu đỏ



4. Thực hiện lặp đi lặp các thao tác phần Flow chung để theo dõi



Leaks Memory

Leak Memory là gì?

- Một **memory leak** là một phần bộ nhớ bị chiếm vĩnh viễn và không thể sử dụng lại được nữa. Chúng là **rác**, chúng chiếm lĩnh các khoảng trống và gây ra nhiều vấn đề

Leaks đến từ đâu

- Leaks có thể đến từ SDK hoặc framework của bên thứ 3, hoặc từ chính Apple như CALayer, UILabel. Trong trường hợp như vậy thì phải chờ bản cập nhật hoặc huỷ SDK đó đi
- Tuy nhiên, phần lớn leaks đến từ source code của dev

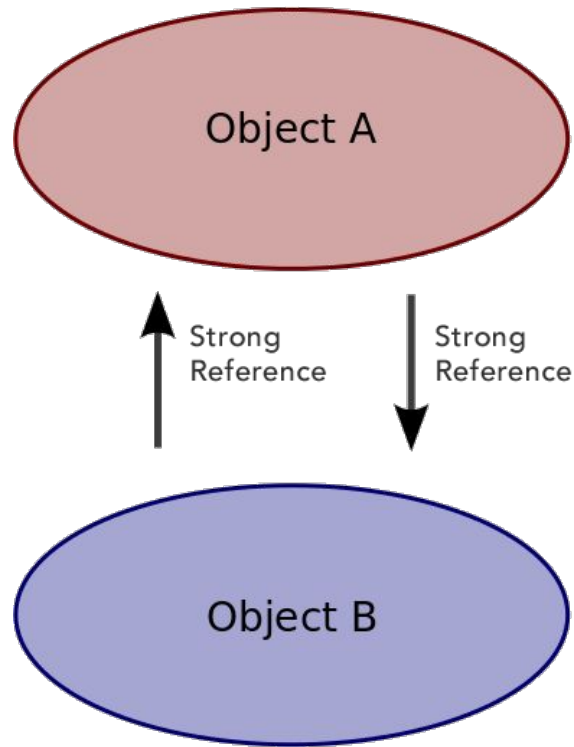
Leaks trong source code dev

Vấn đề leaks memory đến từ ngay trong source code của dev:

- Khởi tạo quá nhiều các biến, hằng, object và không sử dụng đến
- Trước kia để tránh memory leak, dev phải khởi tạo một object bằng **alloc**, và sử dụng **release** để huỷ đi. Từ khi ARC (**Automatic Reference Counting**) ra đời, alloc, release được compiler tự động thêm vào. Tuy nhiên leak memory vẫn có thể xảy ra do **retain cycle**

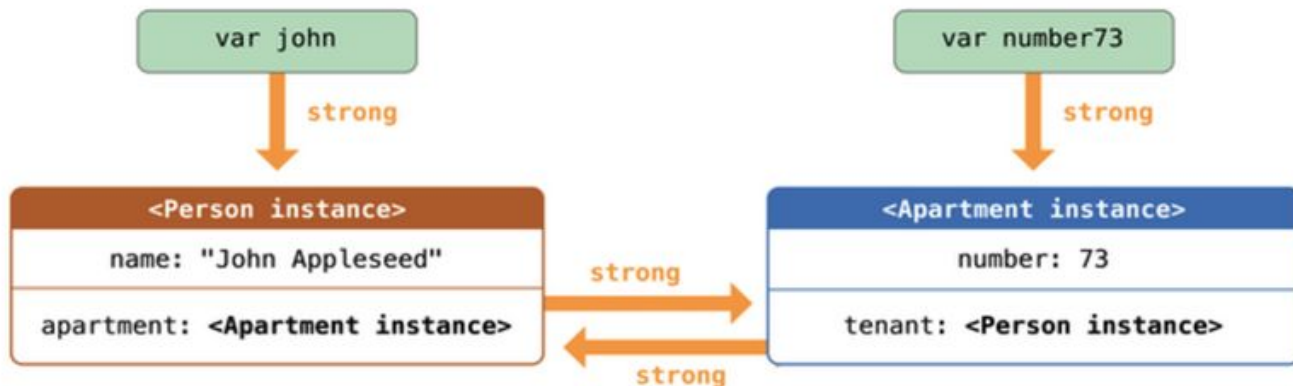
Retain Cycle

- Trong Swift, khi một đối tượng có một liên kết mạnh mẽ đến đối tượng khác, nó sẽ giữ lại nó (**retain**). Nghĩa là các đối tượng đó có sự tham chiếu đến nhau
- Cần phải phát hiện và xử lý retain cycle



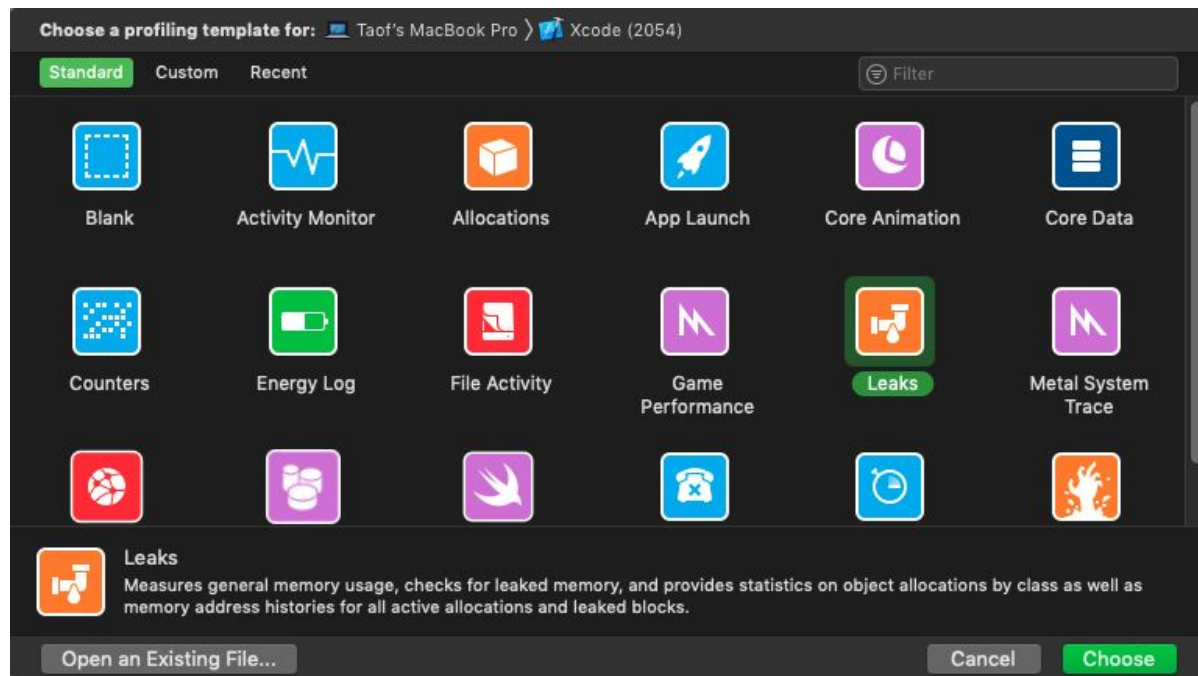
Ví dụ:

- Một đối tượng **Person** có thuộc tính **name** và **apartment** người đó ở
- Một đối tượng **Apartment** có thuộc tính **name** và **person** ở trong apartment đó

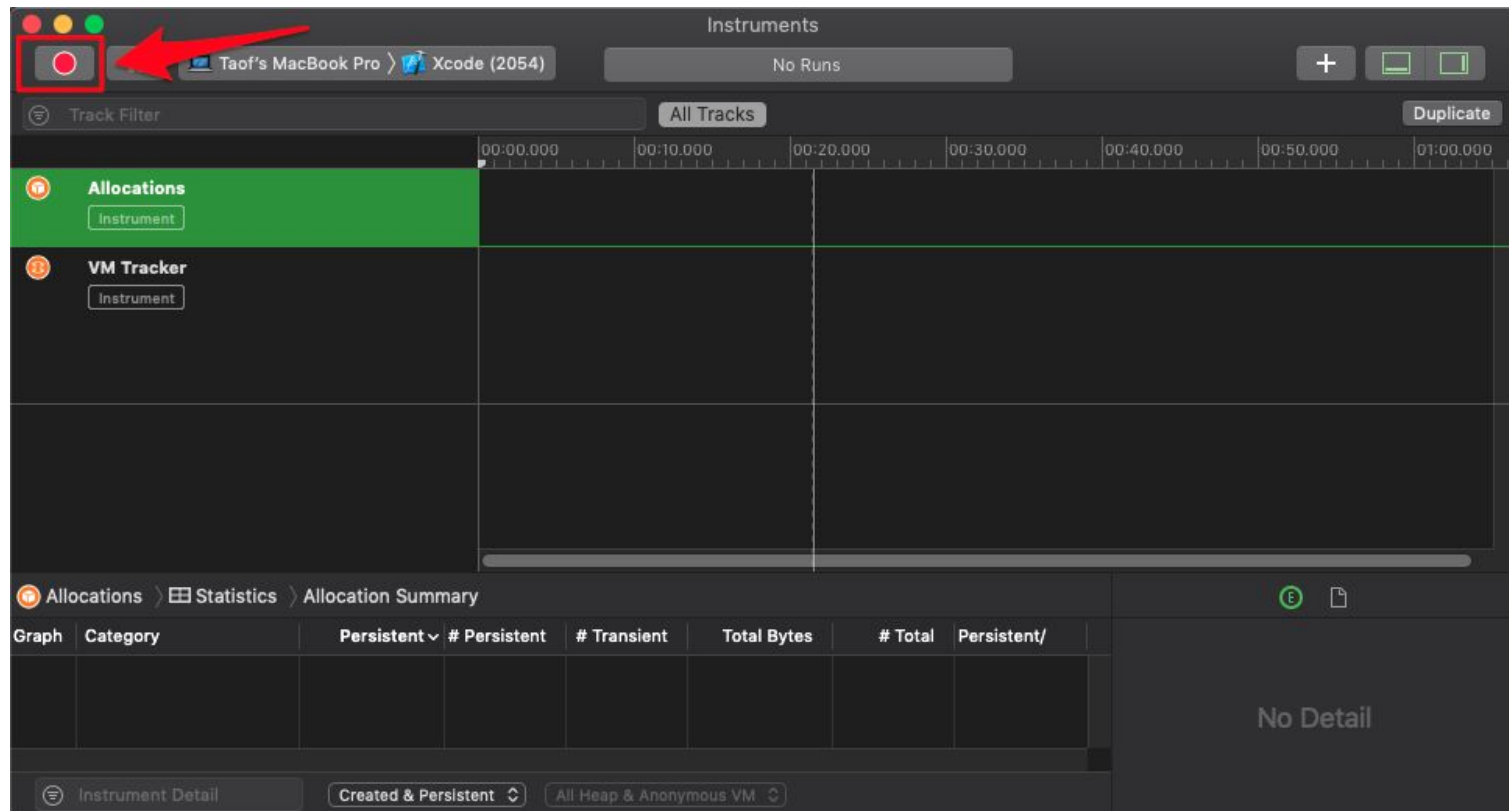


Công cụ: Leaks Instrument

1. Xcode -> Open Developers Tool -> Instrument
2. Chọn Leaks



3. Chọn nút **record** màu đỏ



4. Thao tác trên app, khi có đường màu đỏ xuất hiện, nghĩa là có leaks, click **stop** để dừng

Instruments5

iPhone 6 (8.3 Simul...) TestLeak (5139) Run 1 of 1 00:00:20

Leak

Leaked Object	#	Address	Size	Responsible Library	Responsible Frame
Department	2 < multiple >		64 Bytes	TestLeak	-[ViewController viewWillAppear:]
Person	2 < multiple >		64 Bytes	TestLeak	-[ViewController viewWillAppear:]
Person	1 0x7ffaa0d73210		32 Bytes		
Department	1 0x7ffaa0db0390		32 Bytes		

Snapshots

☒ Automatic Snapshotting

Snapshot Interval (sec) 10.0

Status: Paused

Snapshot Now

Leaks Configuration

☐ Gather Leaked Memory Contents

Call Tree

☐ Separate by Thread

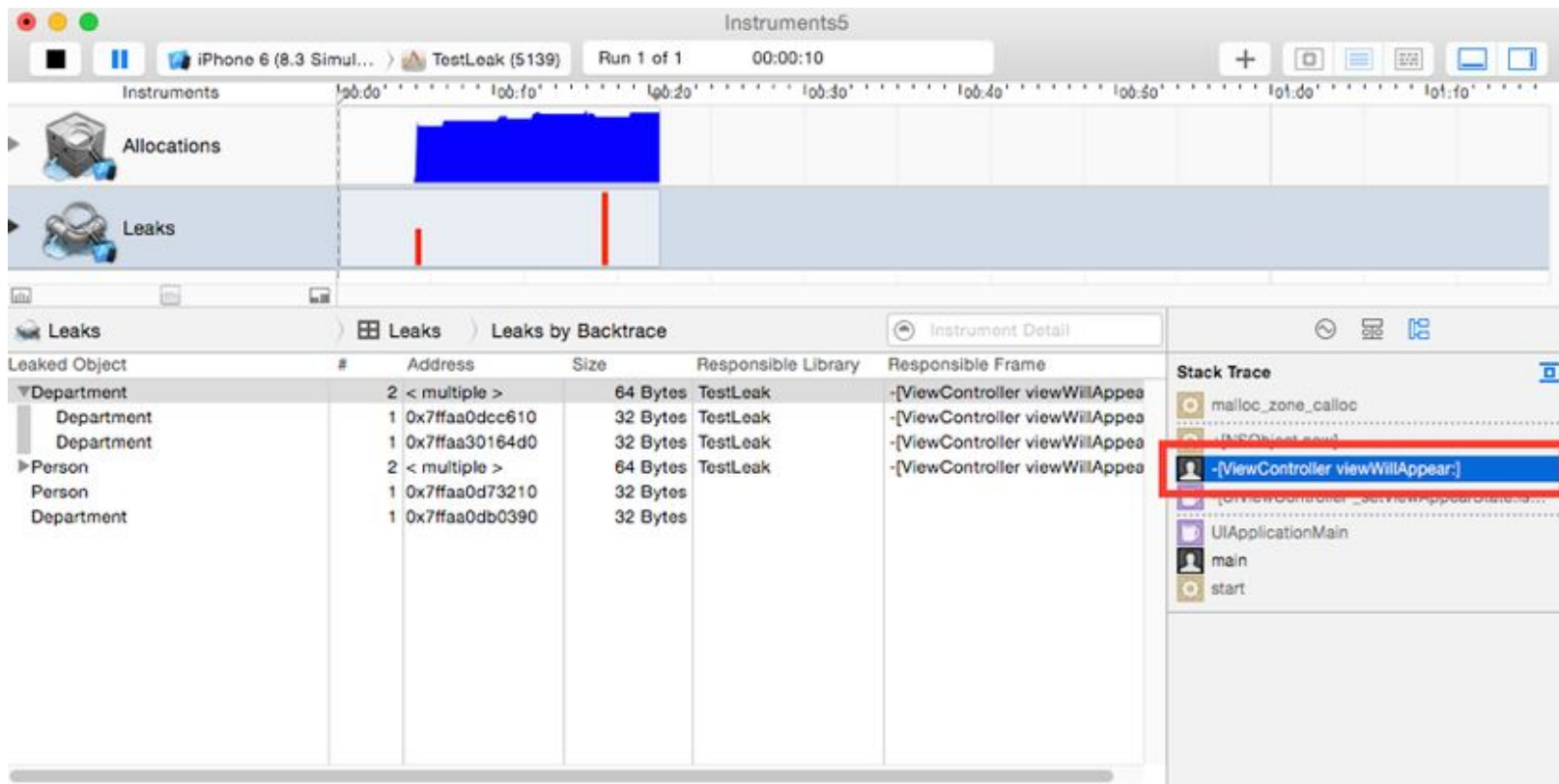
☐ Invert Call Tree

☐ Hide System Libraries

☐ Flatten Recursion

Call Tree Constraints

5. Chọn đối tượng bị leaks, ở Extended Detail inspector, click vào **strace instruction** để xem code gây ra leaks



6. Click vào hình mũi tên sau mỗi instance để xem quá trình tăng giảm '**retain count**'

Leaks						Instrument Detail
Leaks						
Leaks by Backtrace						
Leaked Object	#	Address	Size	Responsible Library	Responsible Frame	
▼ Department	2	< multiple >	64 Bytes	TestLeak	-[ViewController viewWillAppea	
Department	1	0x7ffaa0dcc...	32 Bytes	TestLeak	-[ViewController viewWillAppea	
Department	1	0x7ffaa30164d0	32 Bytes	TestLeak	-[ViewController viewWillAppea	
► Person	2	< multiple >	64 Bytes	TestLeak	-[ViewController viewWillAppea	
Person	1	0x7ffaa0d73210	32 Bytes			
Department	1	0x7ffaa0db0390	32 Bytes			

7. Dựa vào quá trình tăng giảm **retain count** để tìm ra nguyên nhân gây leaks memory

Leaks							
Leak by Backtrace 0x7ffaa0dcc610 History Instrument Detail							
Show: All Unpaired By Group By Time							
#	Event Type	Δ RefCt	RefCt	Timestamp	Responsible L...	Responsible Caller	
0	Malloc	+1	1	00:10.420.801	TestLeak	-[ViewController viewWillAppear:]	
1	Retain	+1	2	00:10.420.805	TestLeak	-[Person setDepartment:]	
2	Release	-1	1	00:10.420.809	TestLeak	-[ViewController viewWillAppear:]	

Tài liệu:

- Sử dụng Developer Tool Instrument

<https://help.apple.com/instruments/mac/current/>

- Tham khảo cuốn sách: **The Pragmatic Programmer**