#WWDC18

iOS Memory Deep Dive

Session 416

Kyle Howarth, Software Engineer
James Snee, Software Engineer
Kris Markel, Software Engineer

Why reduce memory

Memory footprint

Tools for profiling footprint

Images

Optimizing when in background

Demo

Why Reduce Memory

Users have a better experience.

Memory Footprint

Not all memory is created equal

Memory Footprint

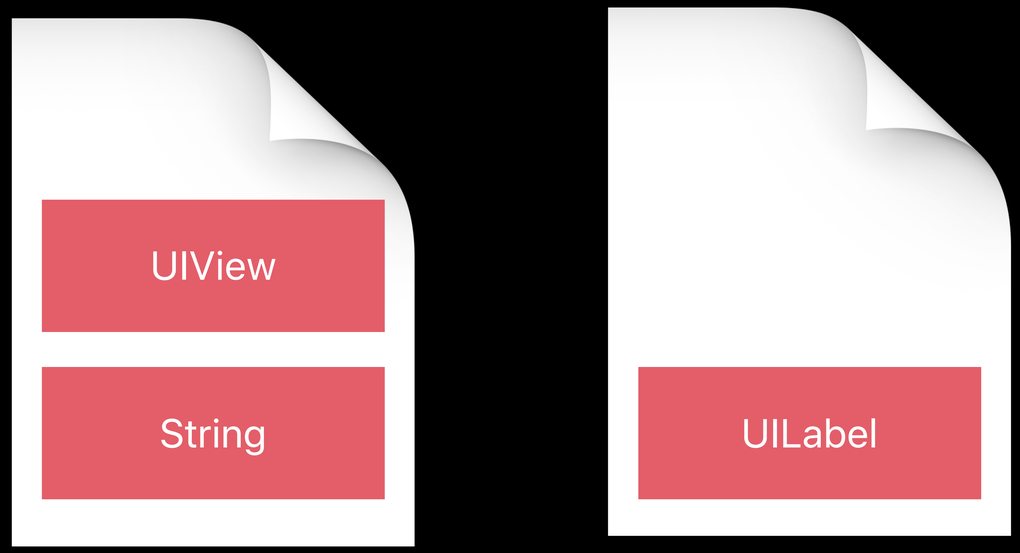
Pages

Memory Footprint

Pages

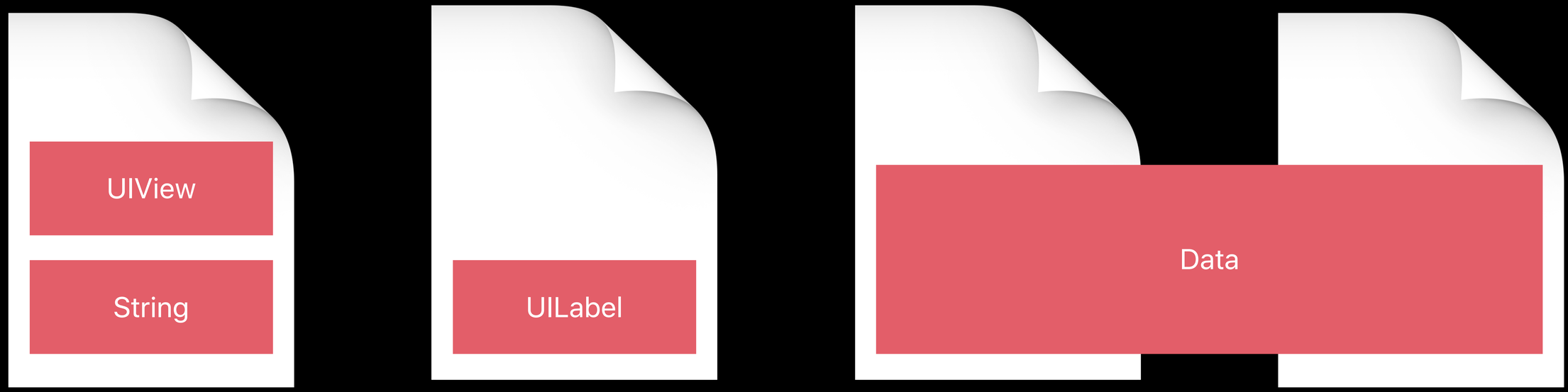
Memory Footprint

Pages



Memory Footprint

Pages



Memory Footprint

Pages

Typically 16KB

Page types

- Clean
- Dirty

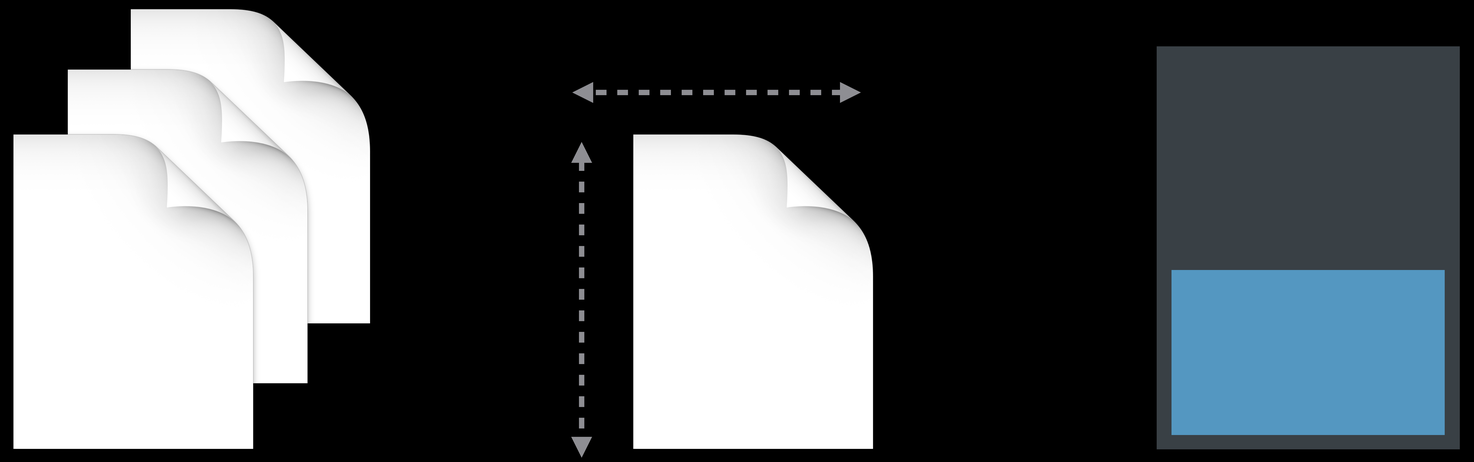
Memory Footprint

Pages

Typically 16KB

Page types

- Clean
- Dirty



Number of pages x **Page size** = **Memory in use**

Memory Footprint

Clean and dirty pages

```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```

Memory Footprint

Clean and dirty pages

```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```

Memory Footprint

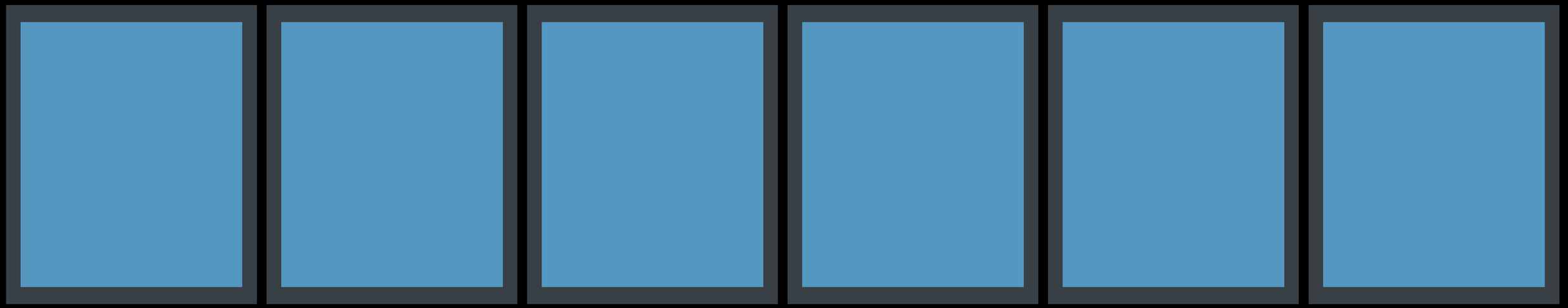
Clean and dirty pages

```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```

Memory Footprint

Clean and dirty pages

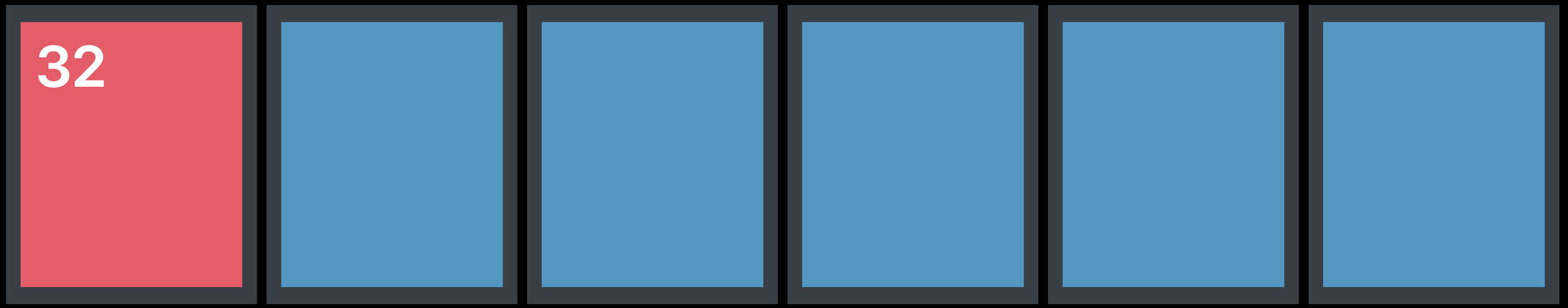
```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```



Memory Footprint

Clean and dirty pages

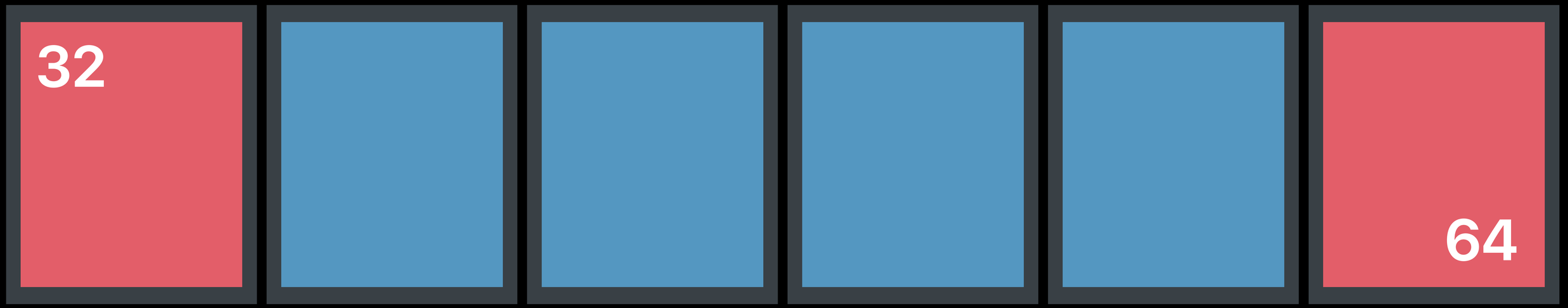
```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```



Memory Footprint

Clean and dirty pages

```
int *array = malloc(20000 * sizeof(int));
array[0] = 32
array[19999] = 64
```



Memory Footprint

Memory mapped files

File-backed memory

Read-only files are clean

Kernel manages when file is in RAM

Memory Footprint

Memory mapped files

50KB

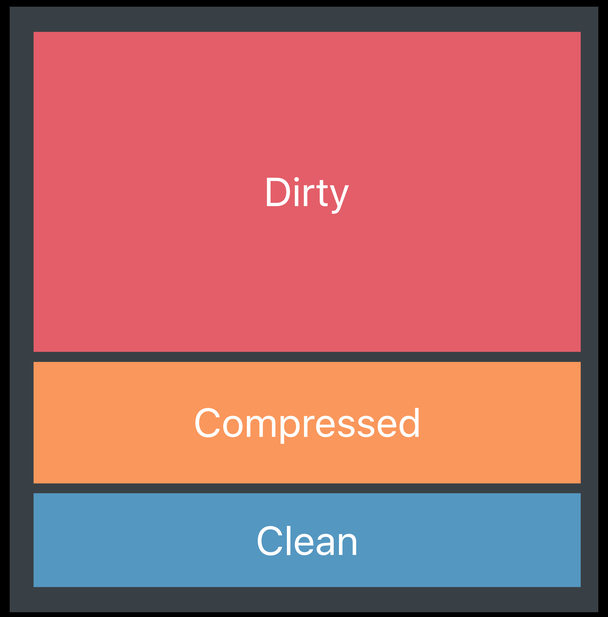
Memory Footprint

Memory mapped files

50KB

Memory Footprint

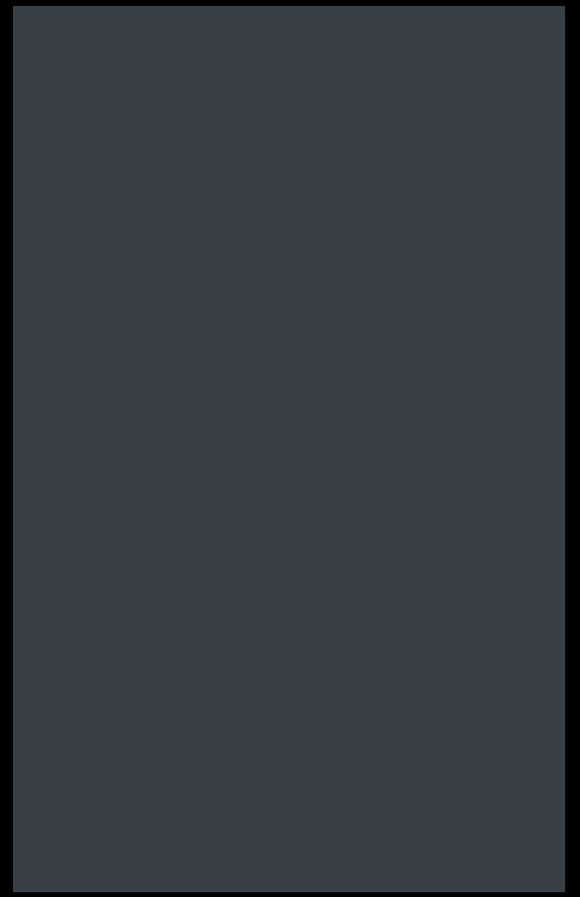
Typical app memory profile



Memory Footprint

Clean memory

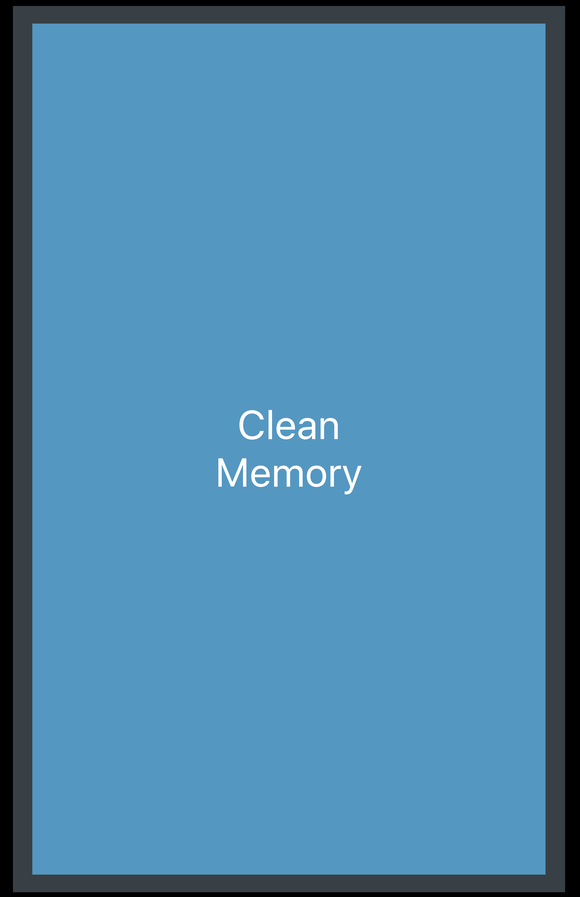
Data that can be paged out of memory



Memory Footprint

Clean memory

Data that can be paged out of memory

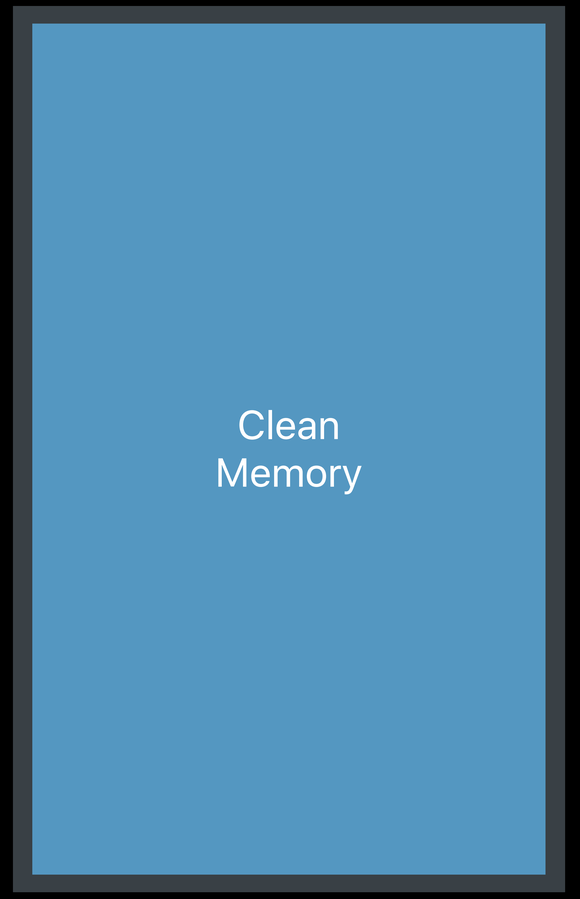


Memory Footprint

Clean memory

Data that can be paged out of memory

Memory mapped files

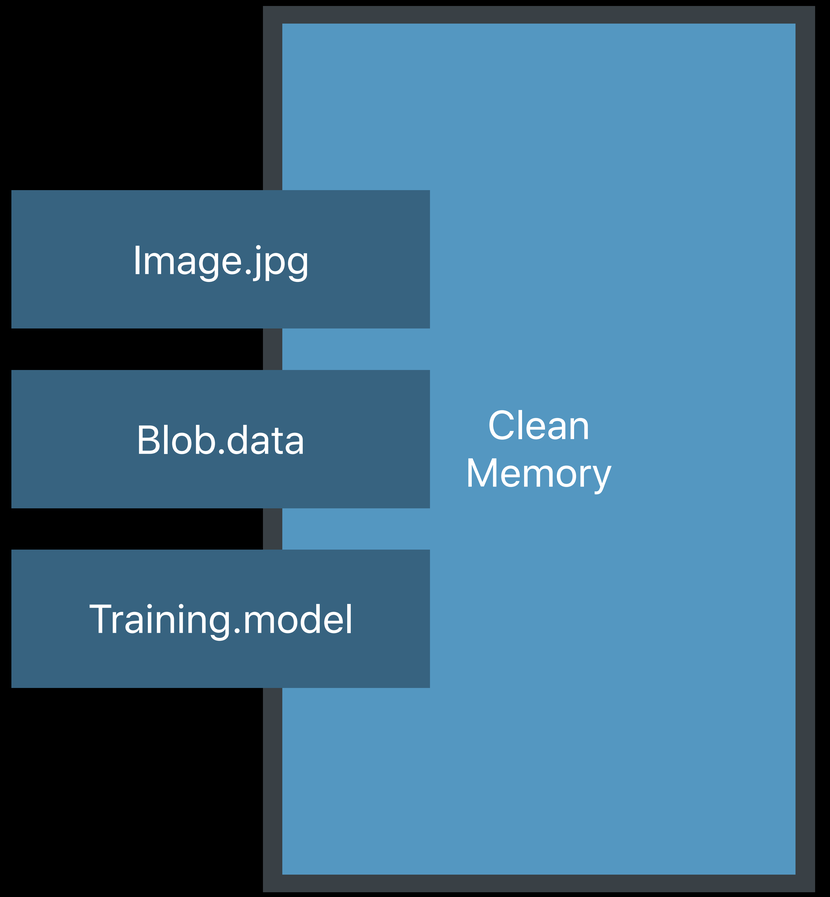


Memory Footprint

Clean memory

Data that can be paged out of memory

Memory mapped files



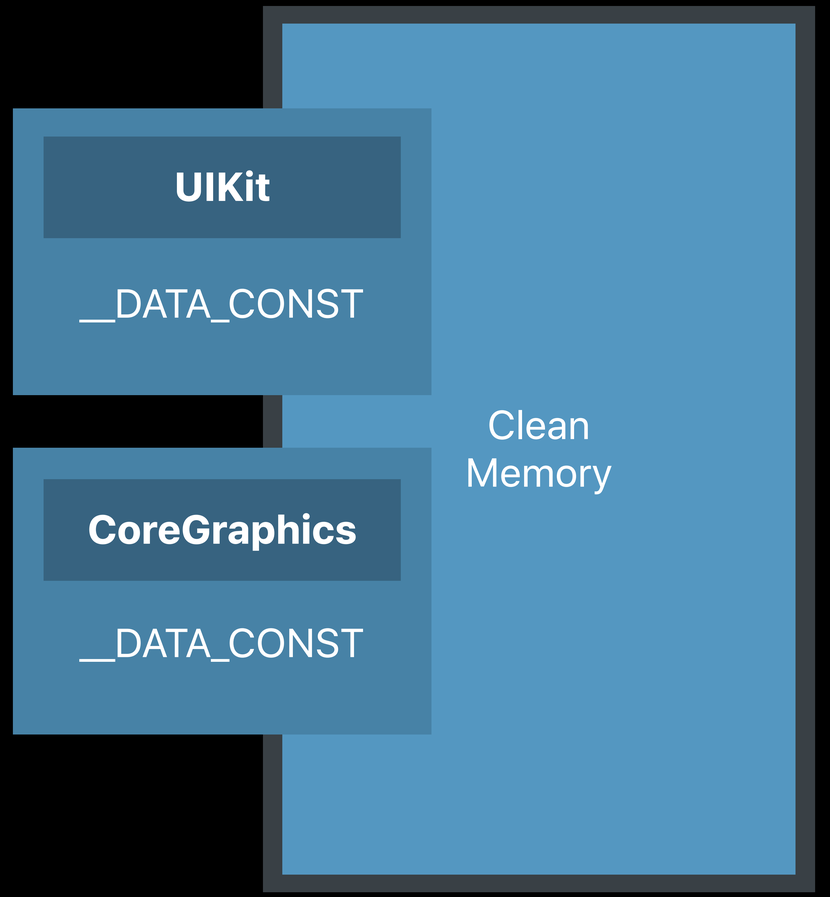
Memory Footprint

Clean memory

Data that can be paged out of memory

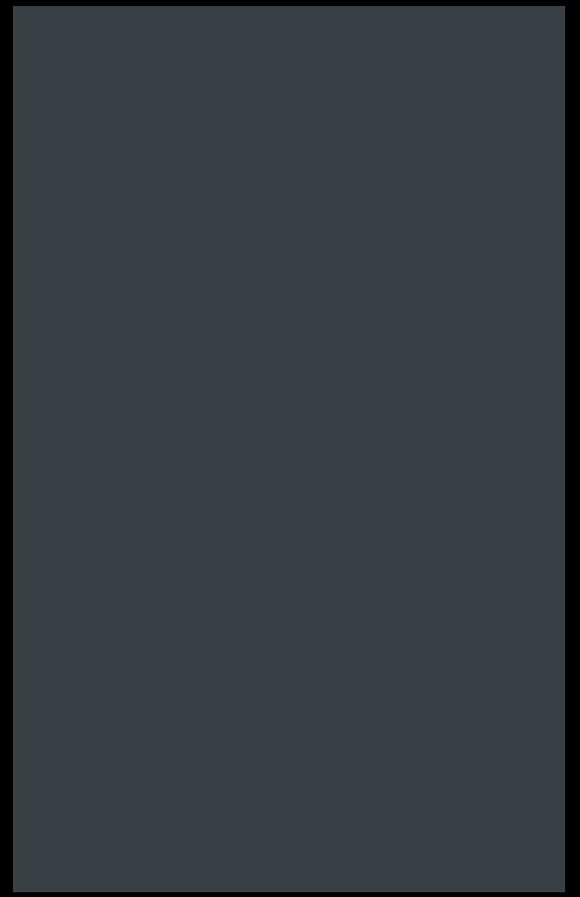
Memory mapped files

Frameworks*



Memory Footprint

Dirty memory



Memory Footprint

Dirty memory

Memory Footprint

Dirty memory

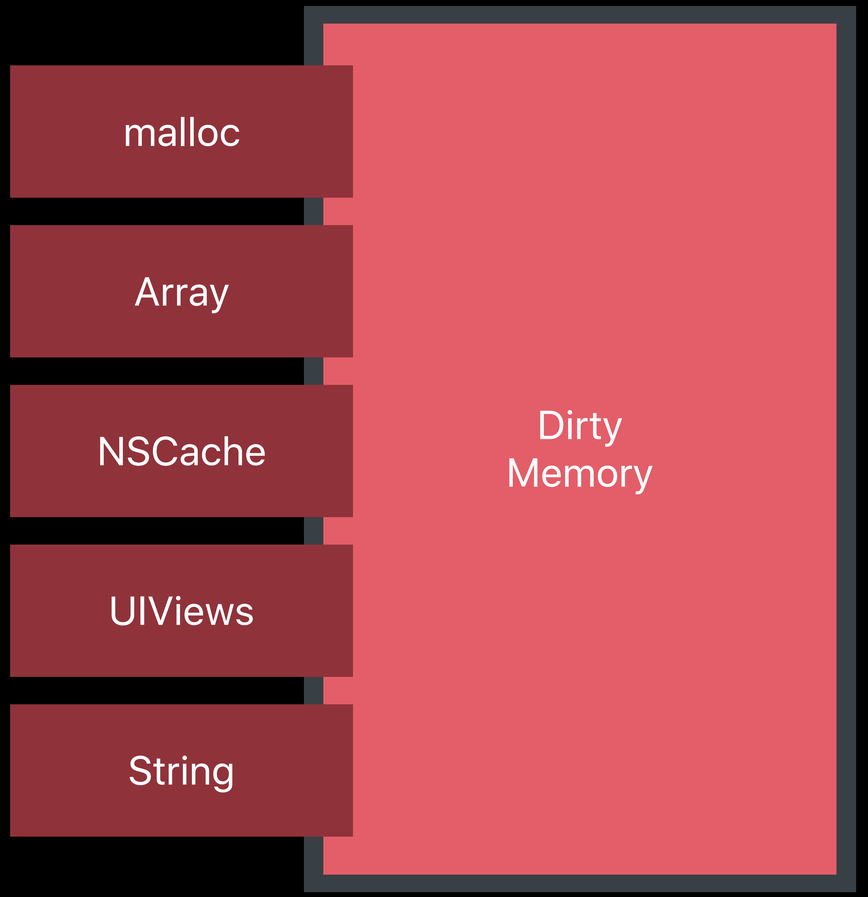
Memory written by an app

Memory Footprint

Dirty memory

Memory written by an app

All heap allocations



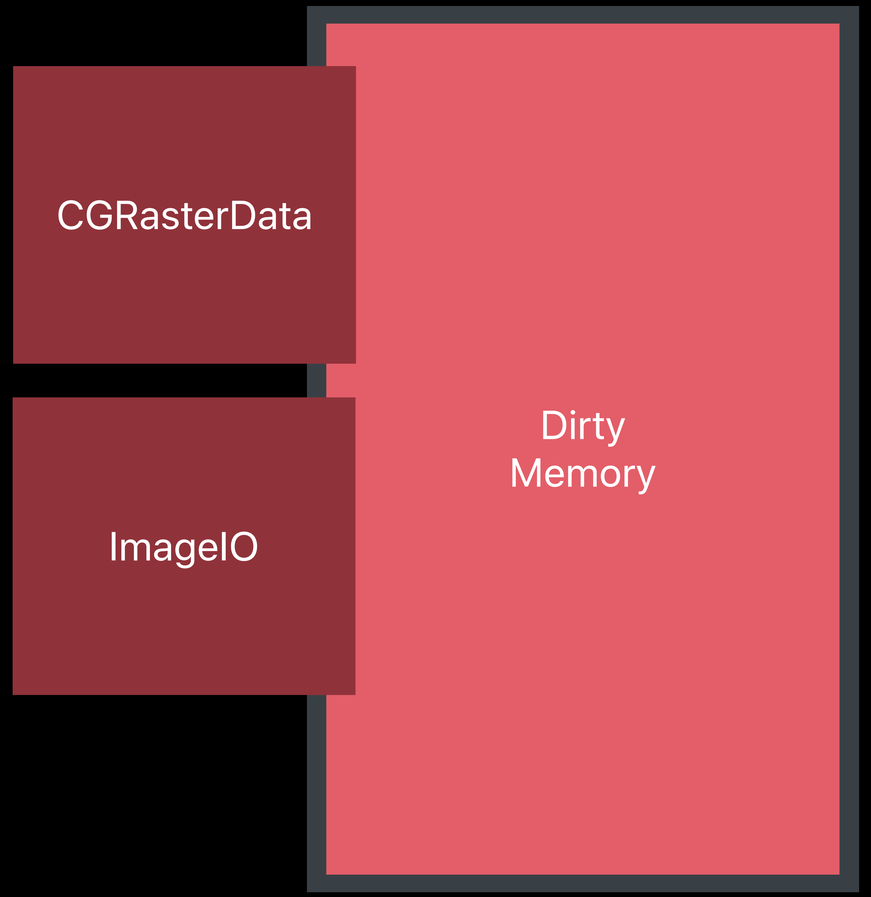
Memory Footprint

Dirty memory

Memory written by an app

All heap allocations

Decoded image buffers



Memory Footprint

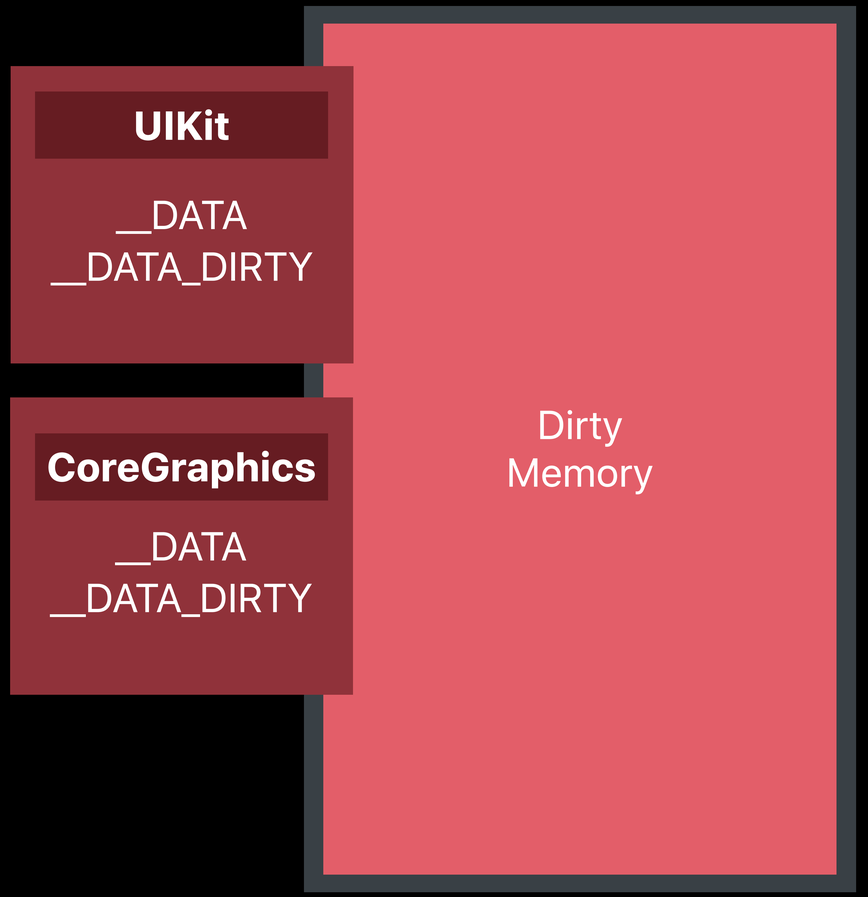
Dirty memory

Memory written by an app

All heap allocations

Decoded image buffers

Frameworks



Memory Footprint

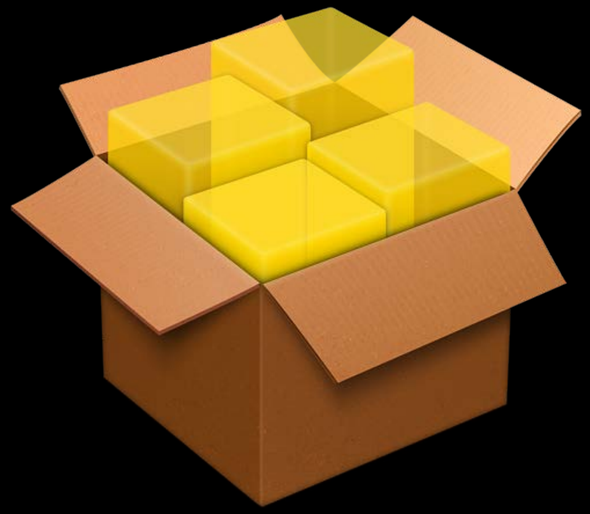
Frameworks

Frameworks contribute to dirty memory

Singletons

Global initializers

```
+(void)load;                        // Objective-C
+(void)initialize;                  // Objective-C
__attribute__((constructor))        // Objective-C

initialize();                       // Swift
```



Memory Footprint

Compressed memory

No traditional disk swap

Memory compressor

- Compresses unaccessed pages
- Decompresses pages upon access

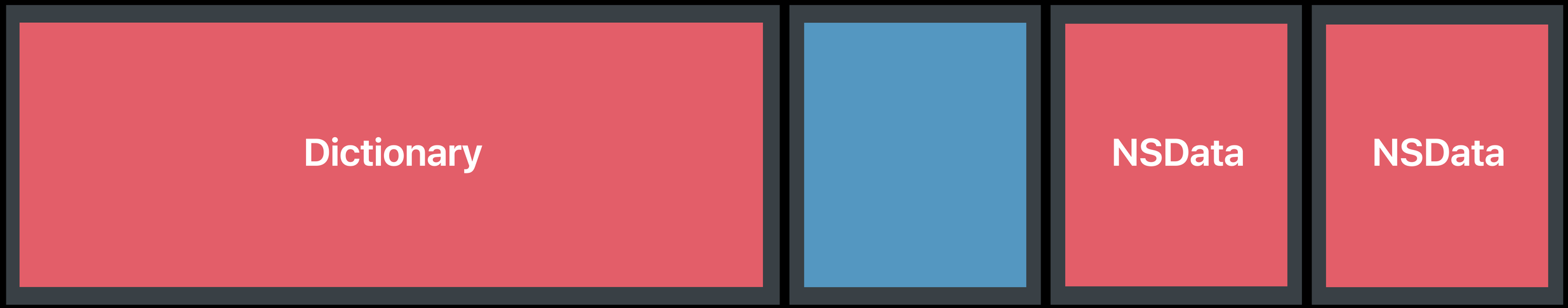
Memory Footprint

Compressed memory

No traditional disk swap

Memory compressor

- Compresses unaccessed pages
- Decompresses pages upon access



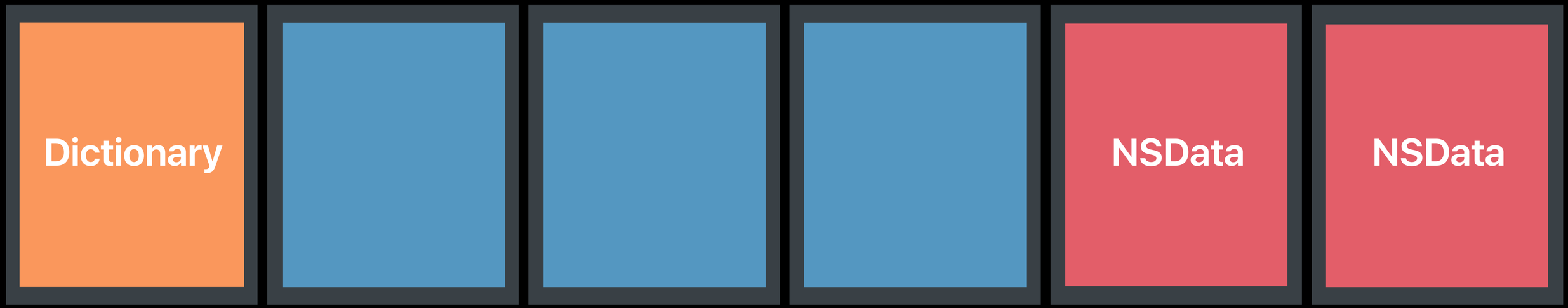
Memory Footprint

Compressed memory

No traditional disk swap

Memory compressor

- Compresses stale pages
- Decompresses pages upon access



Memory Footprint

Memory warnings

App is not always the cause

Compressor complicates freeing memory

Prefer policy changes over cache purging

Memory Footprint

Memory warnings

```
override func didReceiveMemoryWarning() {
    cache.removeAllObjects()
    super.didReceiveMemoryWarning()
}
```

Memory Footprint

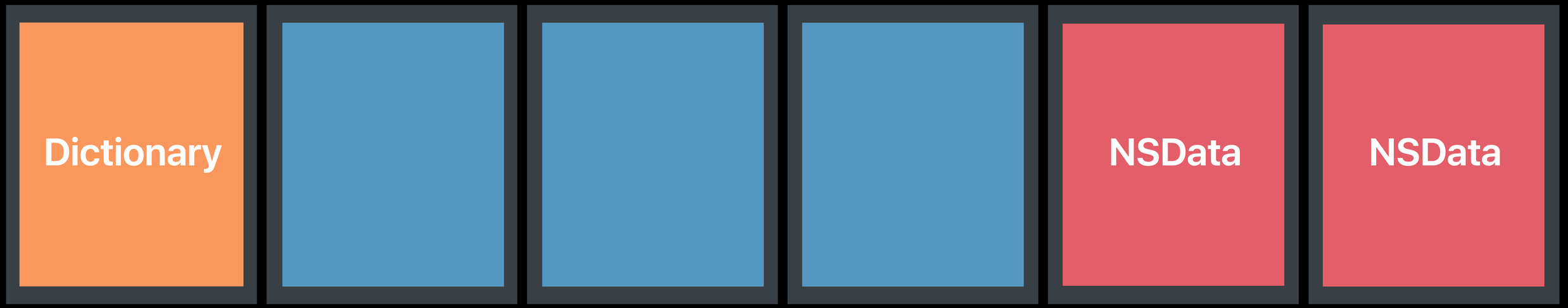
Memory warnings

```
override func didReceiveMemoryWarning() {
    cache.removeAllObjects()
    super.didReceiveMemoryWarning()
}
```

Memory Footprint

Memory warnings

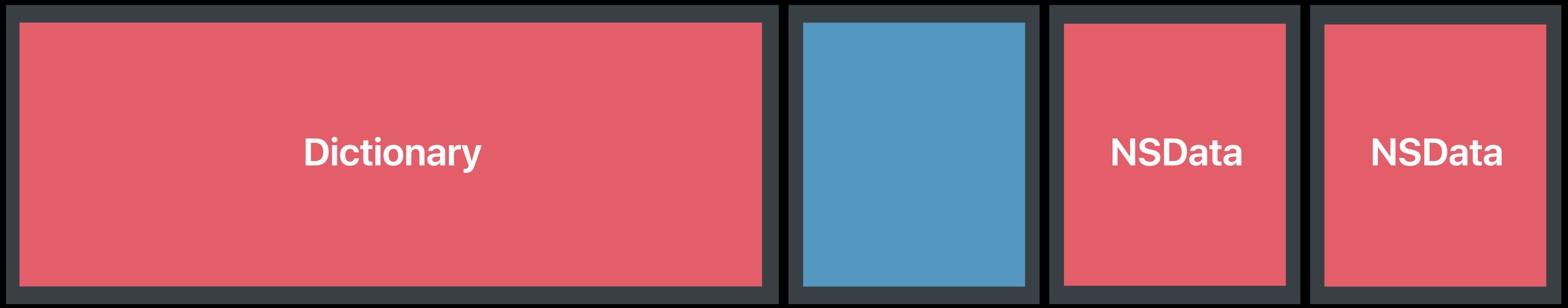
```
override func didReceiveMemoryWarning() {
    cache.removeAllObjects()
    super.didReceiveMemoryWarning()
}
```



Memory Footprint

Memory warnings

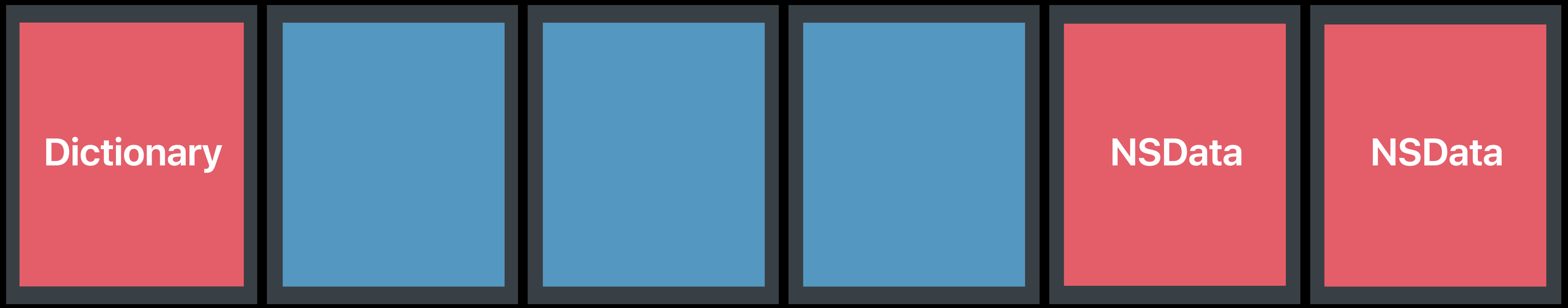
```
override func didReceiveMemoryWarning() {
    cache.removeAllObjects()
    super.didReceiveMemoryWarning()
}
```



Memory Footprint

Memory warnings

```
override func didReceiveMemoryWarning() {
    cache.removeAllObjects()
    super.didReceiveMemoryWarning()
}
```



Memory Footprint

Caching

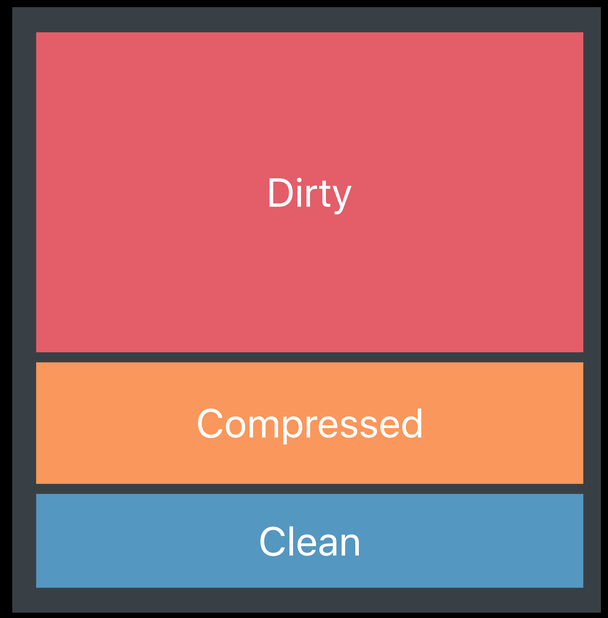
Trade-offs between CPU and memory

Remember the compressor

Prefer NSCache over dictionary

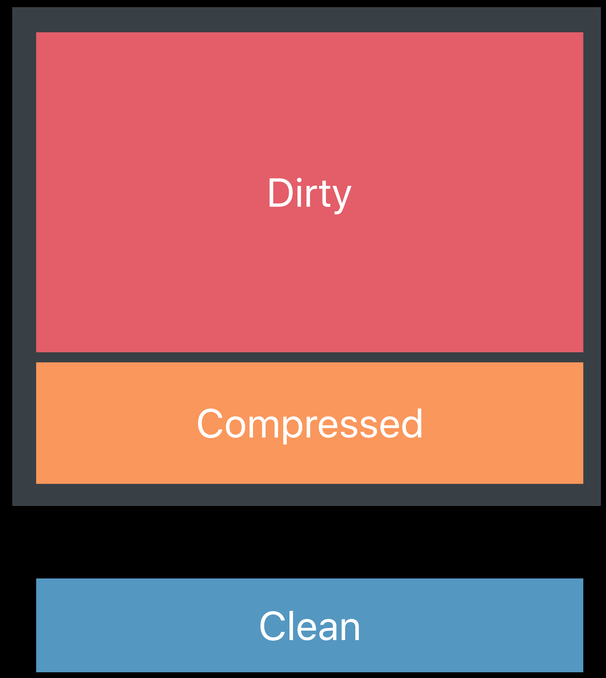
Memory Footprint

Typical app memory profile



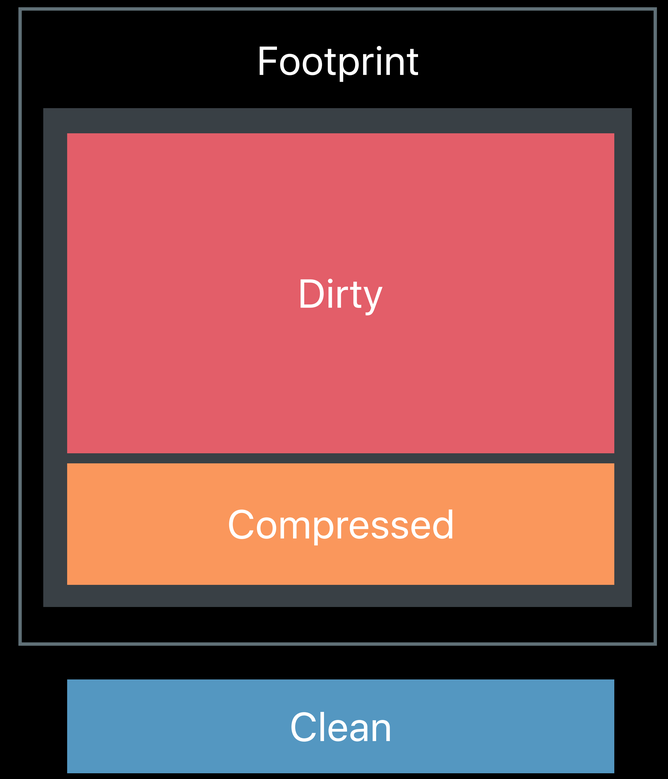
Memory Footprint

Typical app memory profile



Memory Footprint

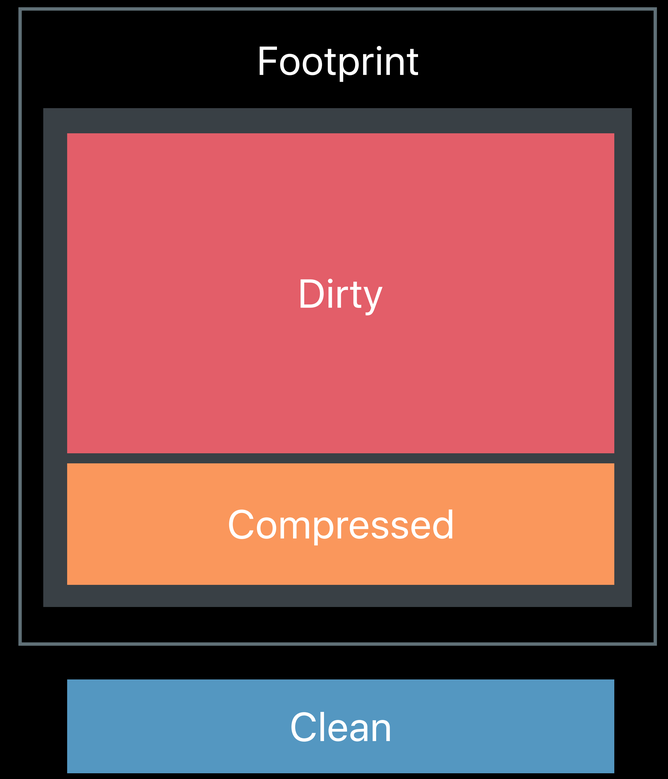
Typical app memory profile



Memory Footprint

Typical app memory profile

Footprint limits

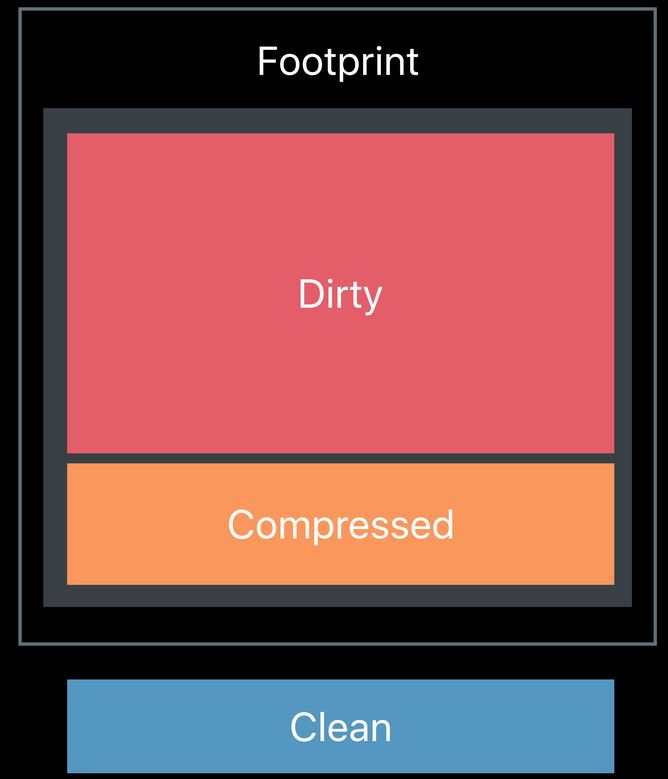


Memory Footprint

Typical app memory profile

Footprint limits

- Limits vary by device

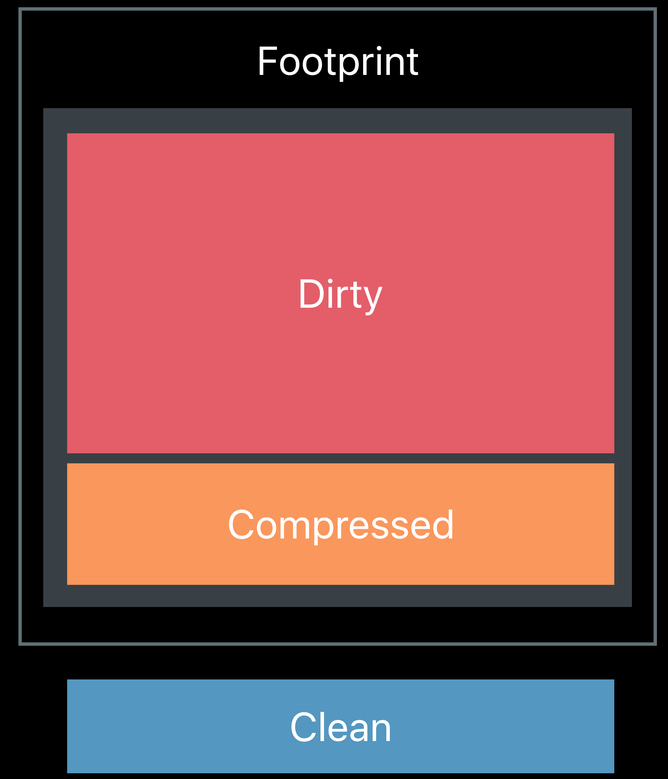


Memory Footprint

Typical app memory profile

Footprint limits

- Limits vary by device
- Apps have a fairly high footprint limit

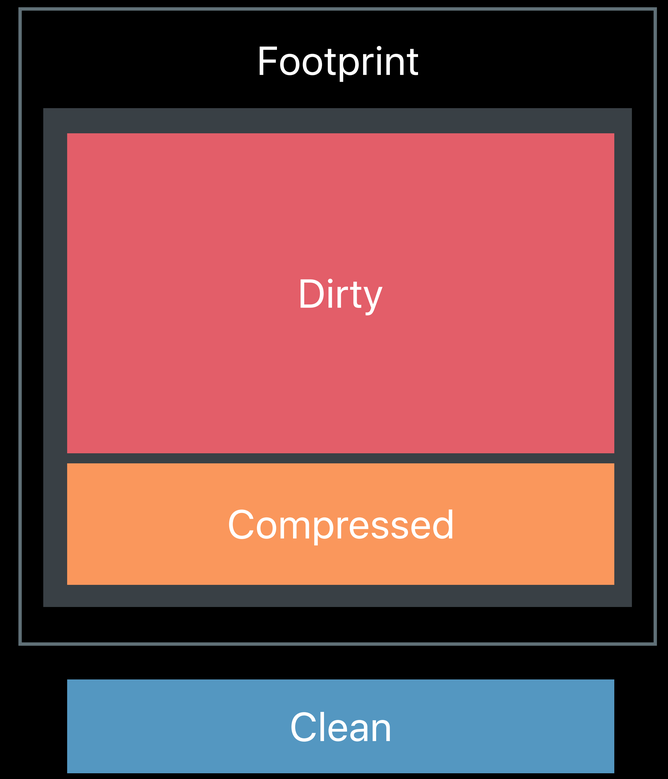


Memory Footprint

Typical app memory profile

Footprint limits

- Limits vary by device
- Apps have a fairly high footprint limit
- Extensions have a much lower limit



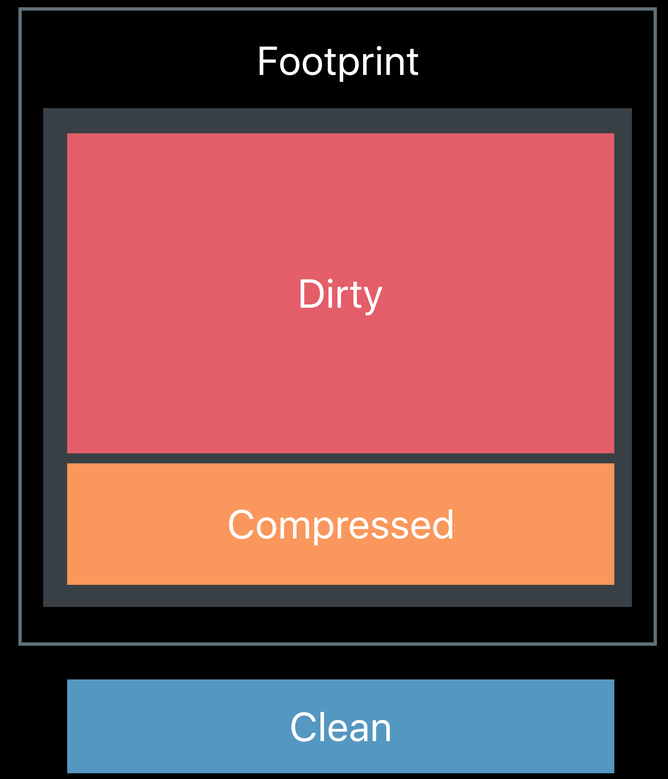
Memory Footprint

Typical app memory profile

Footprint limits

- Limits vary by device
- Apps have a fairly high footprint limit
- Extensions have a much lower limit

Exception upon exceeding limit



Memory Footprint

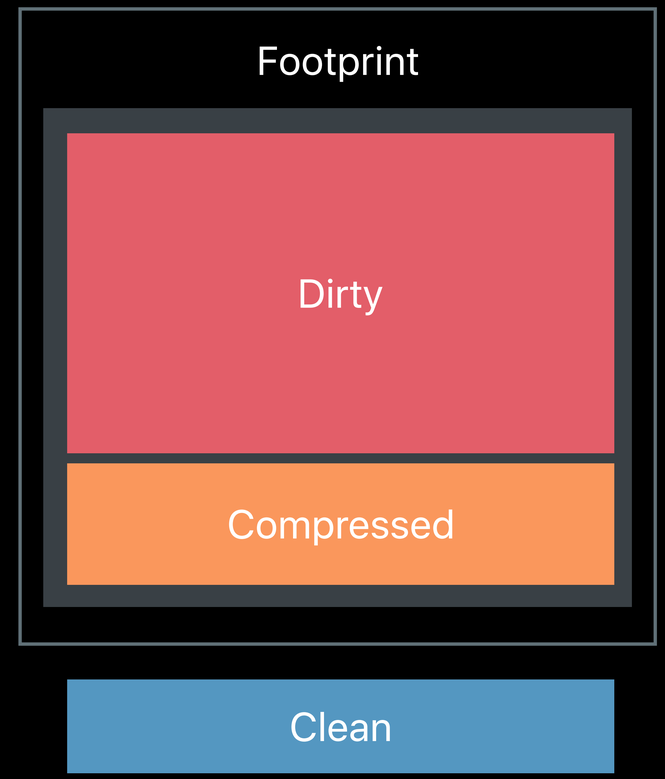
Typical app memory profile

Footprint limits

- Limits vary by device
- Apps have a fairly high footprint limit
- Extensions have a much lower limit

Exception upon exceeding limit

- `EXC_RESOURCE_EXCEPTION`

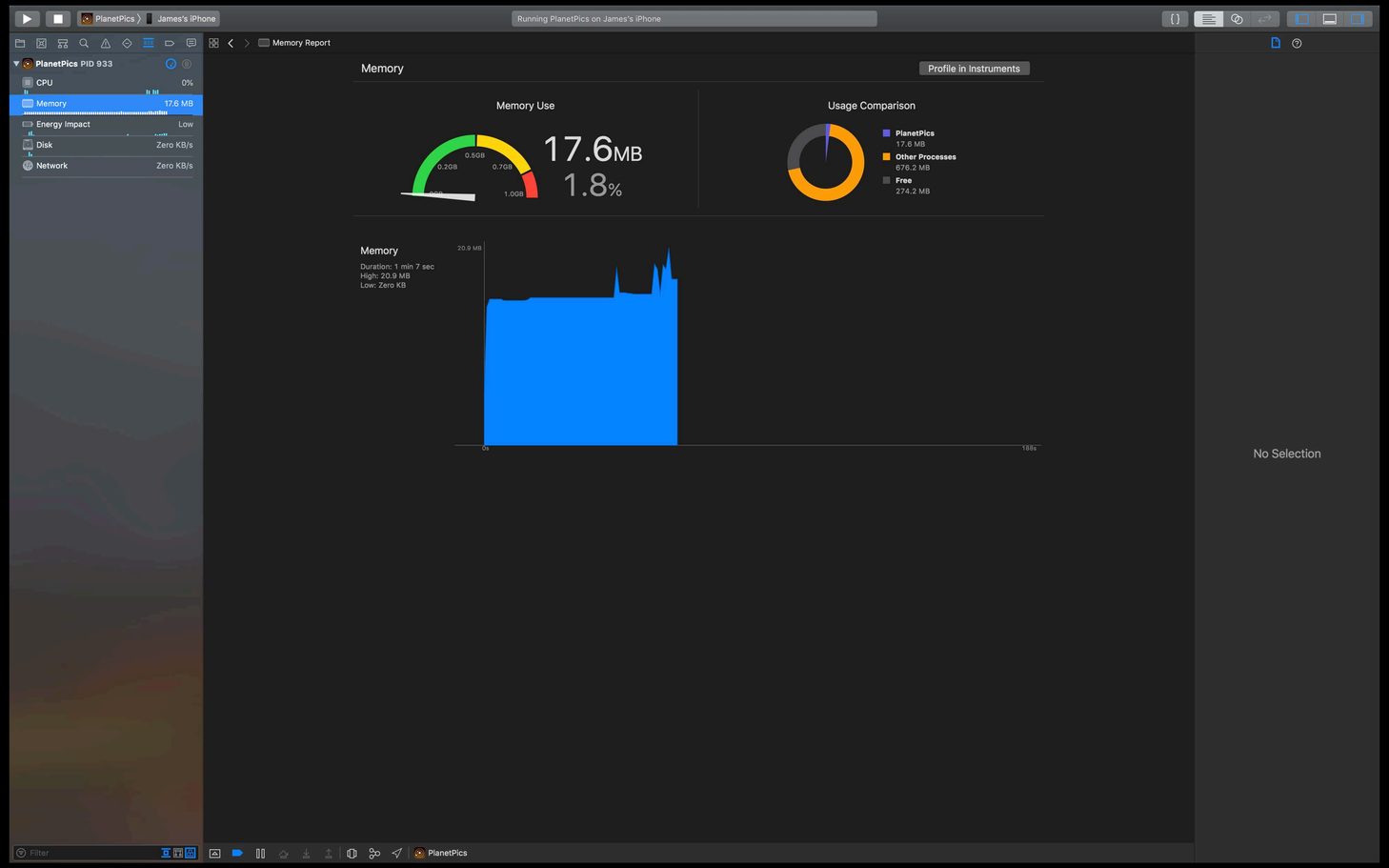


Tools for Profiling Footprint

James Snee, Apple/Software Engineer

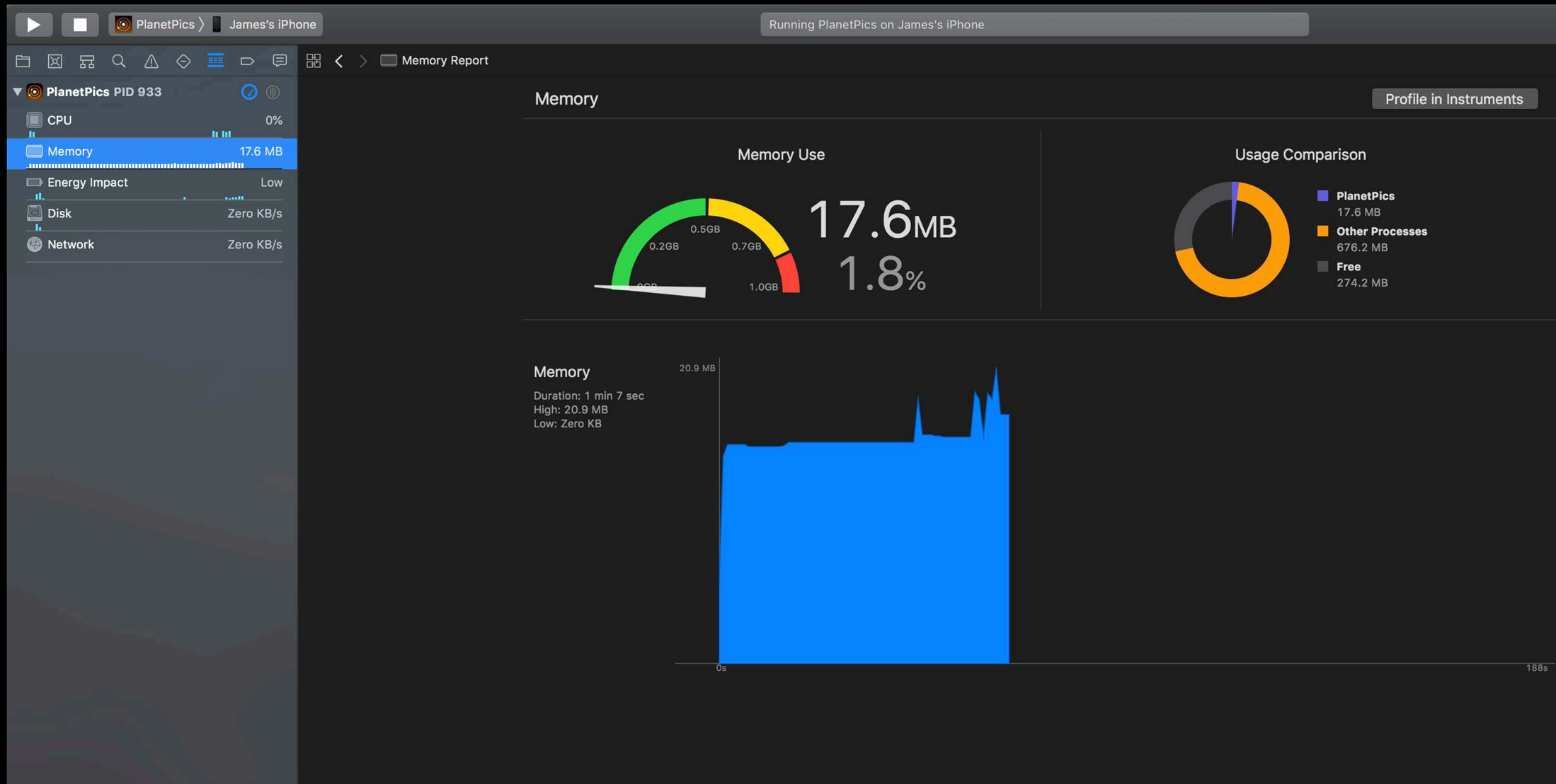
Tools for Profiling Footprint

Xcode memory gauge



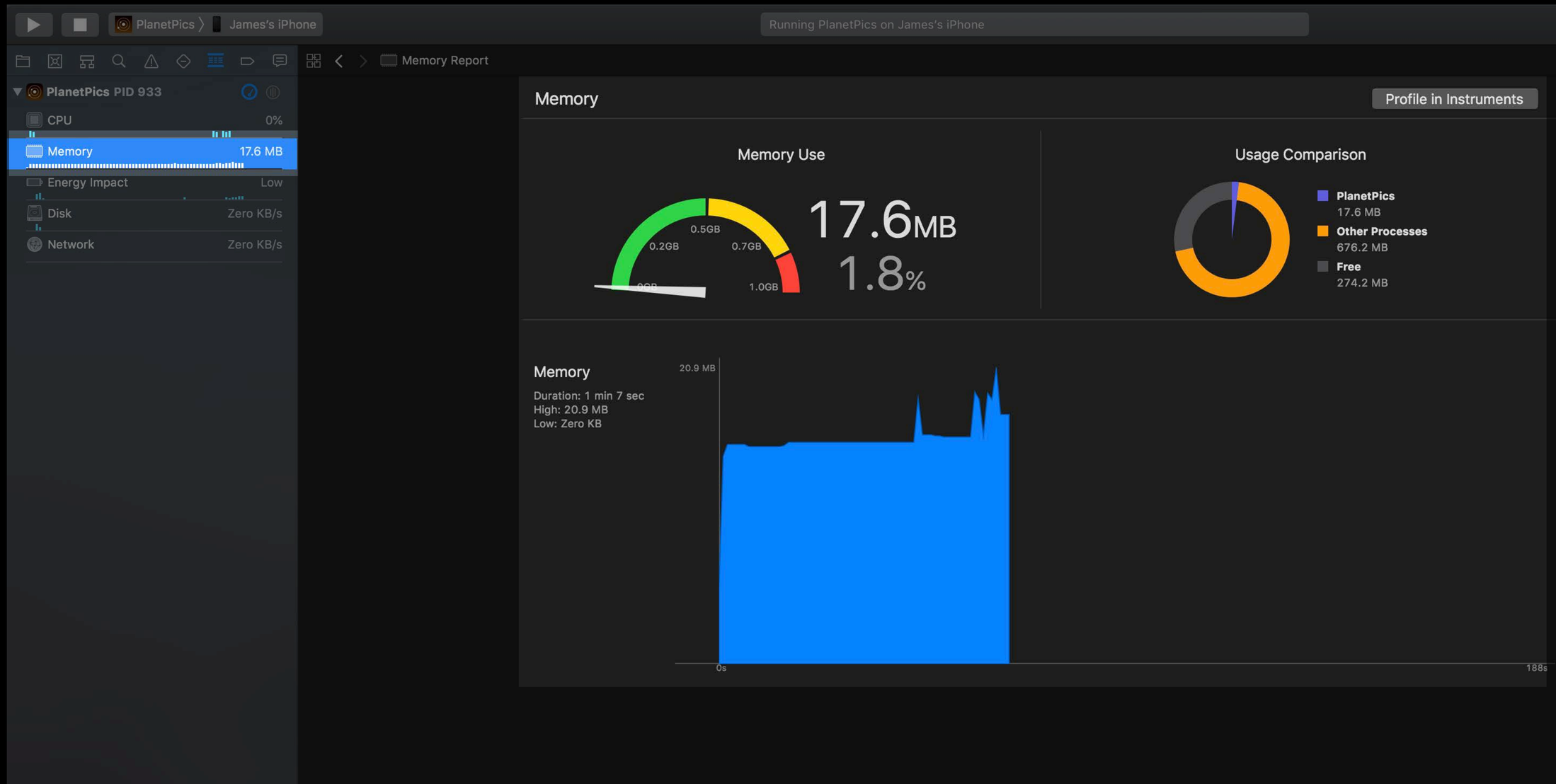
Tools for Profiling Footprint

Xcode memory gauge



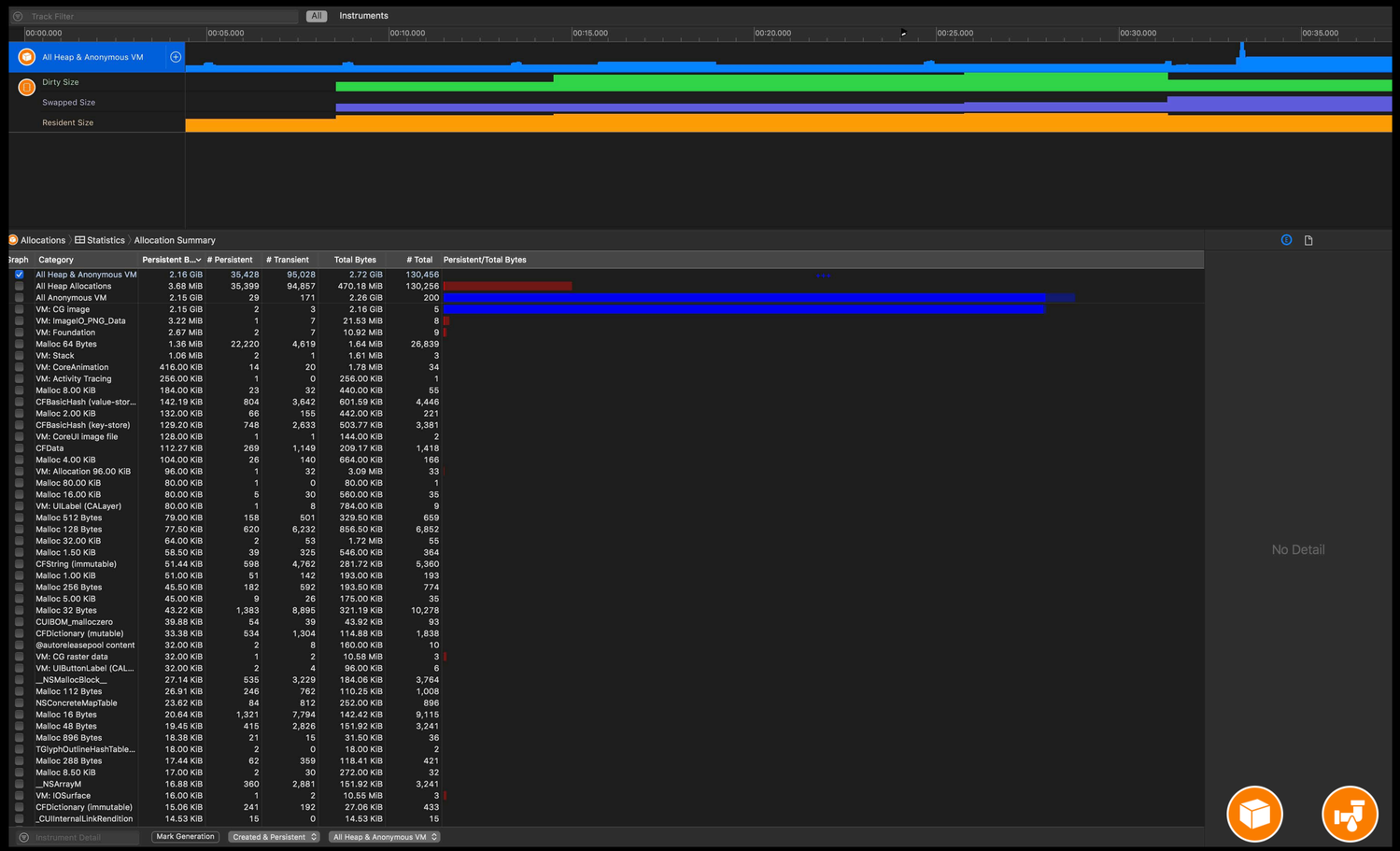
Tools for Profiling Footprint

Xcode memory gauge



Tools for Profiling Footprint

Instruments



Tools for Profiling Footprint

Instruments

Allocations 

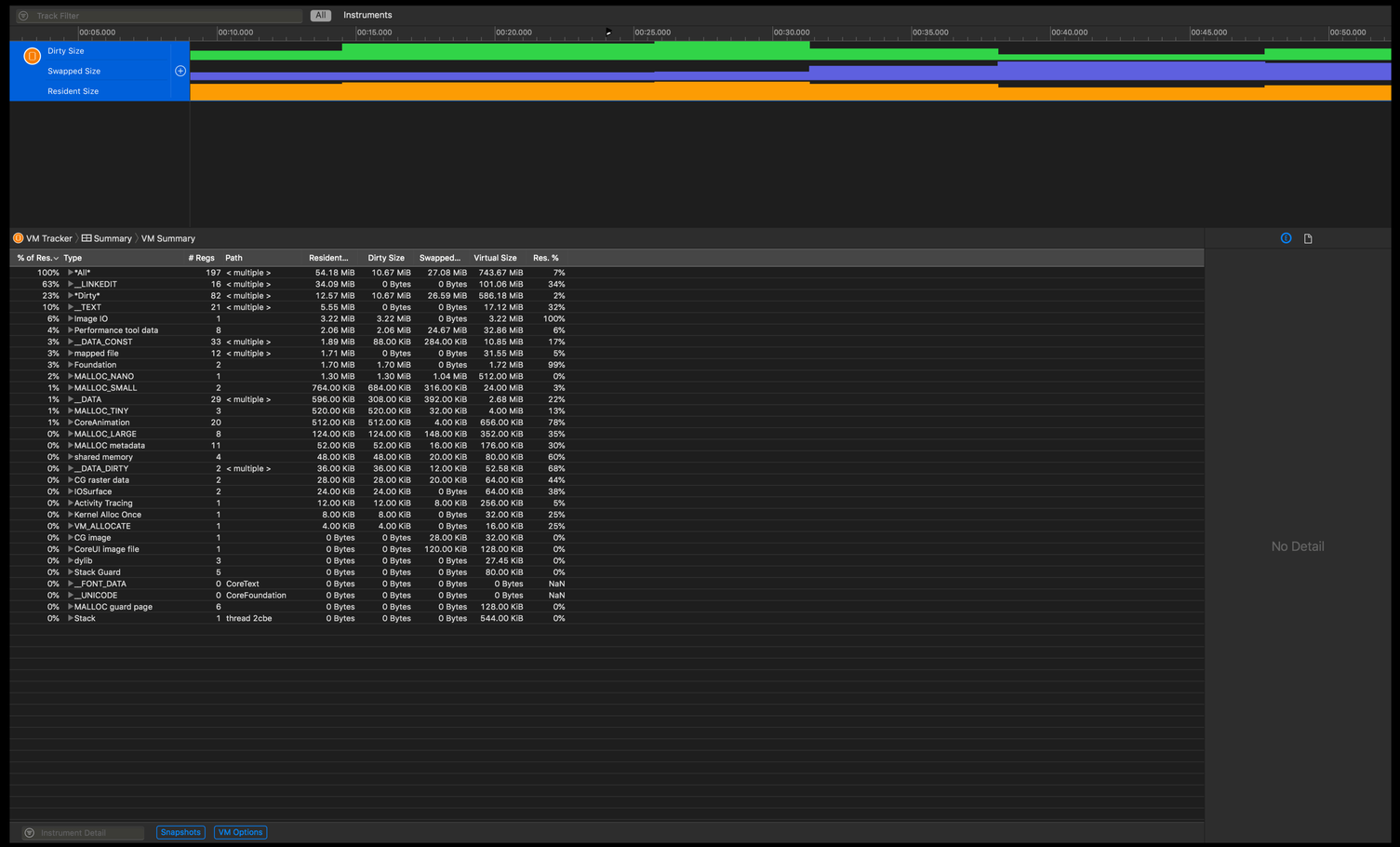
Leaks 

VM Tracker 

Virtual memory trace 

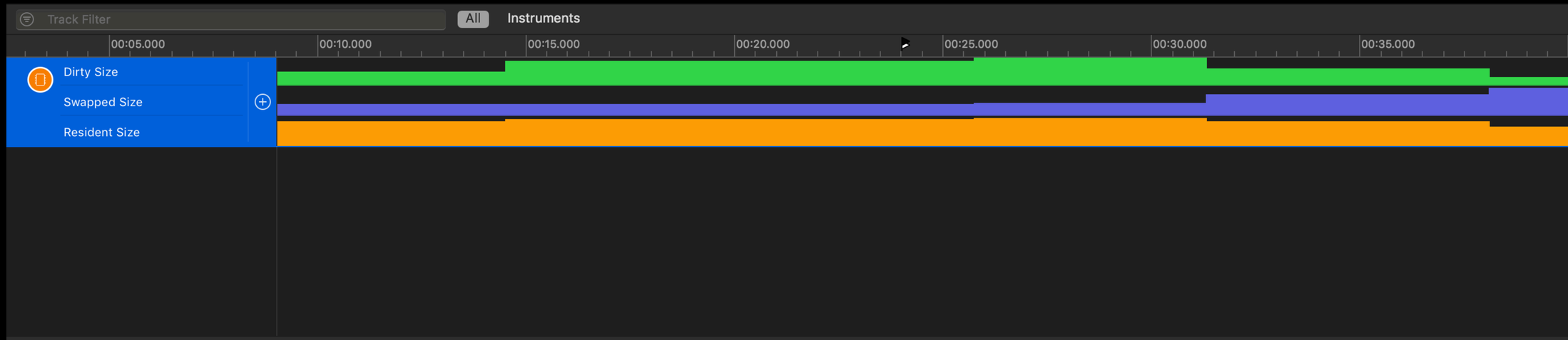
Tools for Profiling Footprint

Instruments — VM Tracker



Tools for Profiling Footprint

Instruments — VM Tracker

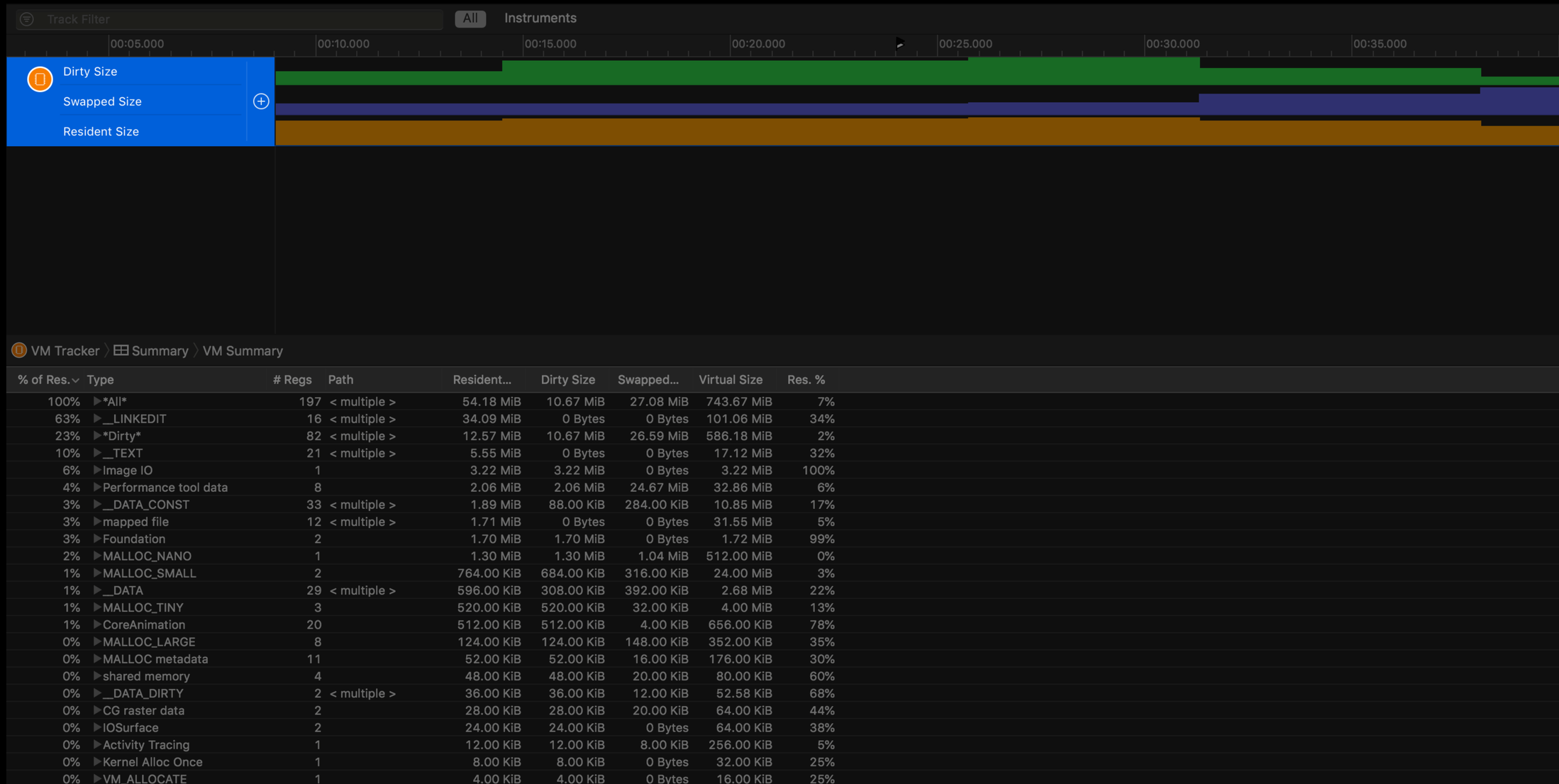


VM Tracker 〉 Summary 〉 VM Summary

% of Res.	Type	# Regs	Path	Resident...	Dirty Size	Swapped...	Virtual Size	Res. %
100%	*All*	197	< multiple >	54.18 MiB	10.67 MiB	27.08 MiB	743.67 MiB	7%
63%	__LINKEDIT	16	< multiple >	34.09 MiB	0 Bytes	0 Bytes	101.06 MiB	34%
23%	*Dirty*	82	< multiple >	12.57 MiB	10.67 MiB	26.59 MiB	586.18 MiB	2%
10%	__TEXT	21	< multiple >	5.55 MiB	0 Bytes	0 Bytes	17.12 MiB	32%
6%	Image IO	1		3.22 MiB	3.22 MiB	0 Bytes	3.22 MiB	100%
4%	Performance tool data	8		2.06 MiB	2.06 MiB	24.67 MiB	32.86 MiB	6%
3%	__DATA_CONST	33	< multiple >	1.89 MiB	88.00 KiB	284.00 KiB	10.85 MiB	17%
3%	mapped file	12	< multiple >	1.71 MiB	0 Bytes	0 Bytes	31.55 MiB	5%
3%	Foundation	2		1.70 MiB	1.70 MiB	0 Bytes	1.72 MiB	99%
2%	MALLOC_NANO	1		1.30 MiB	1.30 MiB	1.04 MiB	512.00 MiB	0%
1%	MALLOC_SMALL	2		764.00 KiB	684.00 KiB	316.00 KiB	24.00 MiB	3%
1%	__DATA	29	< multiple >	596.00 KiB	308.00 KiB	392.00 KiB	2.68 MiB	22%
1%	MALLOC_TINY	3		520.00 KiB	520.00 KiB	32.00 KiB	4.00 MiB	13%
1%	CoreAnimation	20		512.00 KiB	512.00 KiB	4.00 KiB	656.00 KiB	78%
0%	MALLOC_LARGE	8		124.00 KiB	124.00 KiB	148.00 KiB	352.00 KiB	35%
0%	MALLOC metadata	11		52.00 KiB	52.00 KiB	16.00 KiB	176.00 KiB	30%
0%	shared memory	4		48.00 KiB	48.00 KiB	20.00 KiB	80.00 KiB	60%
0%	__DATA_DIRTY	2	< multiple >	36.00 KiB	36.00 KiB	12.00 KiB	52.58 KiB	68%
0%	CG raster data	2		28.00 KiB	28.00 KiB	20.00 KiB	64.00 KiB	44%
0%	IOSurface	2		24.00 KiB	24.00 KiB	0 Bytes	64.00 KiB	38%
0%	Activity Tracing	1		12.00 KiB	12.00 KiB	8.00 KiB	256.00 KiB	5%
0%	Kernel Alloc Once	1		8.00 KiB	8.00 KiB	0 Bytes	32.00 KiB	25%
0%	VM_ALLOCATE	1		4.00 KiB	4.00 KiB	0 Bytes	16.00 KiB	25%

Tools for Profiling Footprint

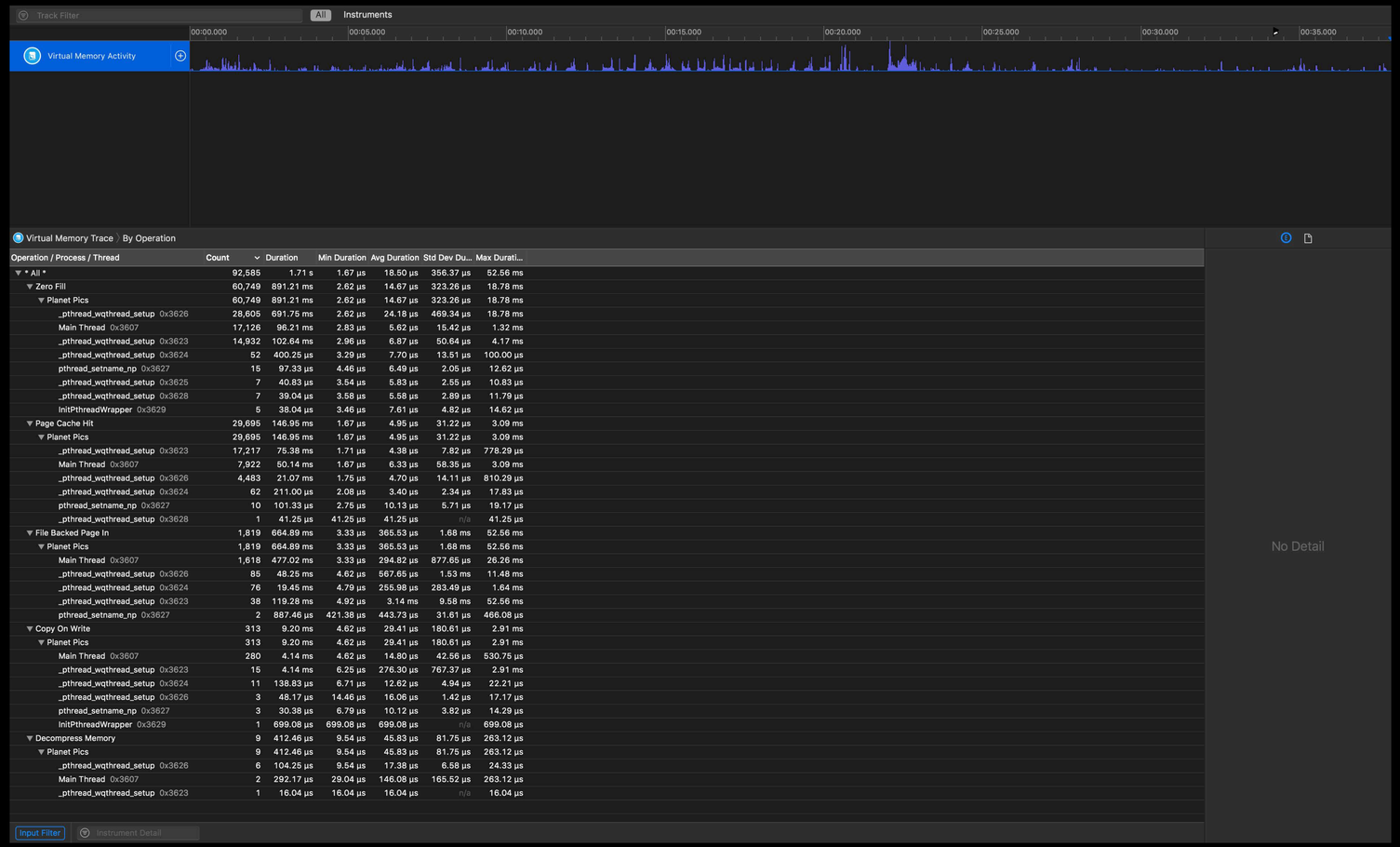
Instruments — VM Tracker



% of Res.	Type	# Regs	Path	Resident...	Dirty Size	Swapped...	Virtual Size	Res. %
100%	*All*	197	< multiple >	54.18 MiB	10.67 MiB	27.08 MiB	743.67 MiB	7%
63%	__LINKEDIT	16	< multiple >	34.09 MiB	0 Bytes	0 Bytes	101.06 MiB	34%
23%	*Dirty*	82	< multiple >	12.57 MiB	10.67 MiB	26.59 MiB	586.18 MiB	2%
10%	__TEXT	21	< multiple >	5.55 MiB	0 Bytes	0 Bytes	17.12 MiB	32%
6%	Image IO	1		3.22 MiB	3.22 MiB	0 Bytes	3.22 MiB	100%
4%	Performance tool data	8		2.06 MiB	2.06 MiB	24.67 MiB	32.86 MiB	6%
3%	__DATA_CONST	33	< multiple >	1.89 MiB	88.00 KiB	284.00 KiB	10.85 MiB	17%
3%	mapped file	12	< multiple >	1.71 MiB	0 Bytes	0 Bytes	31.55 MiB	5%
3%	Foundation	2		1.70 MiB	1.70 MiB	0 Bytes	1.72 MiB	99%
2%	MALLOC_NANO	1		1.30 MiB	1.30 MiB	1.04 MiB	512.00 MiB	0%
1%	MALLOC_SMALL	2		764.00 KiB	684.00 KiB	316.00 KiB	24.00 MiB	3%
1%	__DATA	29	< multiple >	596.00 KiB	308.00 KiB	392.00 KiB	2.68 MiB	22%
1%	MALLOC_TINY	3		520.00 KiB	520.00 KiB	32.00 KiB	4.00 MiB	13%
1%	CoreAnimation	20		512.00 KiB	512.00 KiB	4.00 KiB	656.00 KiB	78%
0%	MALLOC_LARGE	8		124.00 KiB	124.00 KiB	148.00 KiB	352.00 KiB	35%
0%	MALLOC metadata	11		52.00 KiB	52.00 KiB	16.00 KiB	176.00 KiB	30%
0%	shared memory	4		48.00 KiB	48.00 KiB	20.00 KiB	80.00 KiB	60%
0%	__DATA_DIRTY	2	< multiple >	36.00 KiB	36.00 KiB	12.00 KiB	52.58 KiB	68%
0%	CG raster data	2		28.00 KiB	28.00 KiB	20.00 KiB	64.00 KiB	44%
0%	IOSurface	2		24.00 KiB	24.00 KiB	0 Bytes	64.00 KiB	38%
0%	Activity Tracing	1		12.00 KiB	12.00 KiB	8.00 KiB	256.00 KiB	5%
0%	Kernel Alloc Once	1		8.00 KiB	8.00 KiB	0 Bytes	32.00 KiB	25%
0%	VM_ALLOCATE	1		4.00 KiB	4.00 KiB	0 Bytes	16.00 KiB	25%

Tools for Profiling Footprint

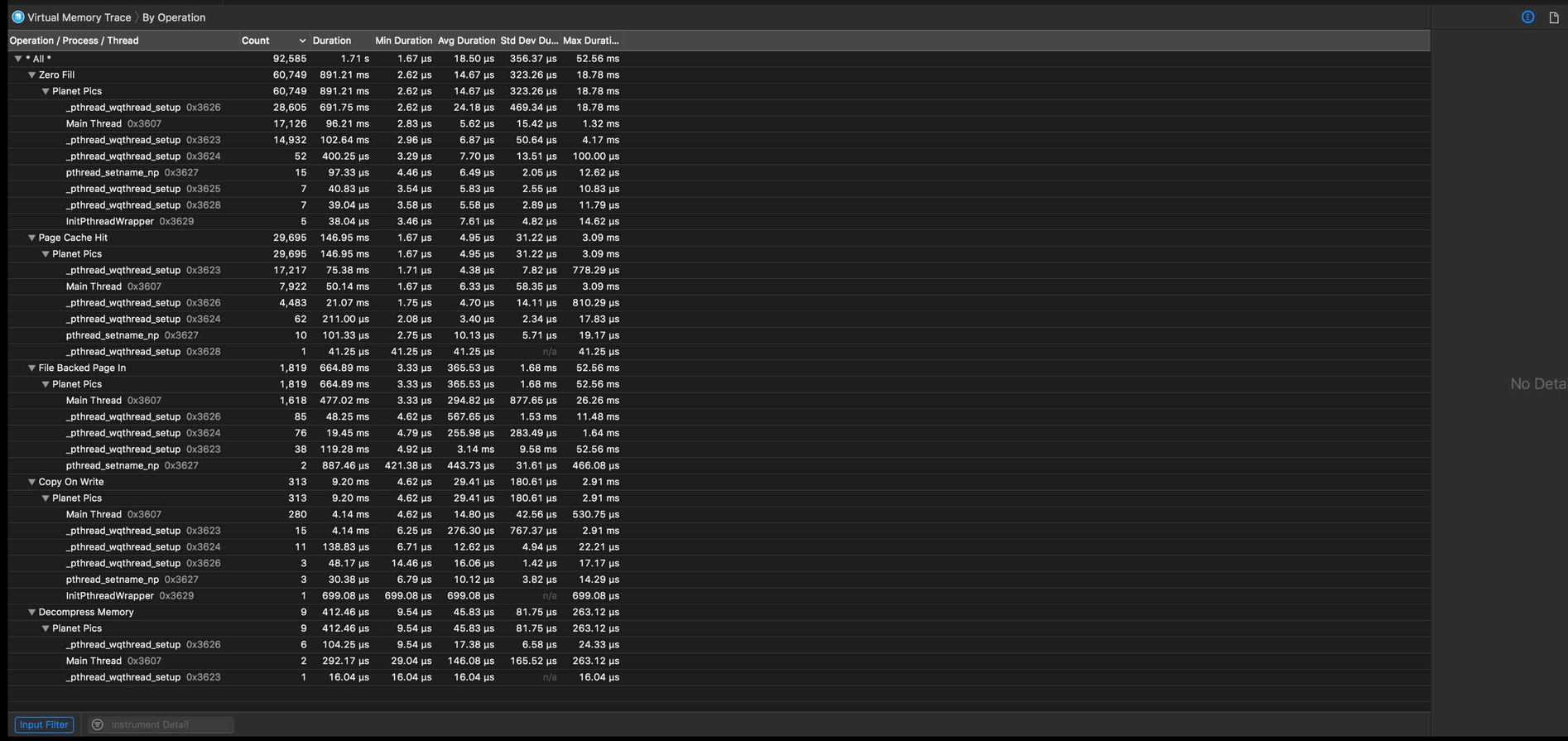
Instruments — Virtual Memory Trace



Operation / Process / Thread	Count	Duration	Min Duration	Avg Duration	Std Dev Du...	Max Durati...
▼ * All *	92,585	1.71 s	1.67 µs	18.50 µs	356.37 µs	52.56 ms
▼ Zero Fill	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
▼ Planet Pics	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
_pthread_wqthread_setup 0x3626	28,605	691.75 ms	2.62 µs	24.18 µs	469.34 µs	18.78 ms
Main Thread 0x3607	17,126	96.21 ms	2.83 µs	5.62 µs	15.42 µs	1.32 ms
_pthread_wqthread_setup 0x3623	14,932	102.64 ms	2.96 µs	6.87 µs	50.64 µs	4.17 ms
_pthread_wqthread_setup 0x3624	52	400.25 µs	3.29 µs	7.70 µs	13.51 µs	100.00 µs
pthread_setname_np 0x3627	15	97.33 µs	4.46 µs	6.49 µs	2.05 µs	12.62 µs
_pthread_wqthread_setup 0x3625	7	40.83 µs	3.54 µs	5.83 µs	2.55 µs	10.83 µs
_pthread_wqthread_setup 0x3628	7	39.04 µs	3.58 µs	5.58 µs	2.89 µs	11.79 µs
InitPthreadWrapper 0x3629	5	38.04 µs	3.46 µs	7.61 µs	4.82 µs	14.62 µs
▼ Page Cache Hit	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
▼ Planet Pics	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
_pthread_wqthread_setup 0x3623	17,217	75.38 ms	1.71 µs	4.38 µs	7.82 µs	778.29 µs
Main Thread 0x3607	7,922	50.14 ms	1.67 µs	6.33 µs	58.35 µs	3.09 ms
_pthread_wqthread_setup 0x3626	4,483	21.07 ms	1.75 µs	4.70 µs	14.11 µs	810.29 µs
_pthread_wqthread_setup 0x3624	62	211.00 µs	2.08 µs	3.40 µs	2.34 µs	17.83 µs
pthread_setname_np 0x3627	10	101.33 µs	2.75 µs	10.13 µs	5.71 µs	19.17 µs
_pthread_wqthread_setup 0x3628	1	41.25 µs	41.25 µs	41.25 µs	n/a	41.25 µs
▼ File Backed Page In	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
▼ Planet Pics	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
Main Thread 0x3607	1,618	477.02 ms	3.33 µs	294.82 µs	877.65 µs	26.26 ms
_pthread_wqthread_setup 0x3626	85	48.25 ms	4.62 µs	567.65 µs	1.53 ms	11.48 ms
_pthread_wqthread_setup 0x3624	76	19.45 ms	4.79 µs	255.98 µs	283.49 µs	1.64 ms
_pthread_wqthread_setup 0x3623	38	119.28 ms	4.92 µs	3.14 ms	9.58 ms	52.56 ms
pthread_setname_np 0x3627	2	887.46 µs	421.38 µs	443.73 µs	31.61 µs	466.08 µs
▼ Copy On Write	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
▼ Planet Pics	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
Main Thread 0x3607	280	4.14 ms	4.62 µs	14.80 µs	42.56 µs	530.75 µs
_pthread_wqthread_setup 0x3623	15	4.14 ms	6.25 µs	276.30 µs	767.37 µs	2.91 ms
_pthread_wqthread_setup 0x3624	11	138.83 µs	6.71 µs	12.62 µs	4.94 µs	22.21 µs
_pthread_wqthread_setup 0x3626	3	48.17 µs	14.46 µs	16.06 µs	1.42 µs	17.17 µs
pthread_setname_np 0x3627	3	30.38 µs	6.79 µs	10.12 µs	3.82 µs	14.29 µs
InitPthreadWrapper 0x3629	1	699.08 µs	699.08 µs	699.08 µs	n/a	699.08 µs
▼ Decompress Memory	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
▼ Planet Pics	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
_pthread_wqthread_setup 0x3626	6	104.25 µs	9.54 µs	17.38 µs	6.58 µs	24.33 µs
Main Thread 0x3607	2	292.17 µs	29.04 µs	146.08 µs	165.52 µs	263.12 µs
_pthread_wqthread_setup 0x3623	1	16.04 µs	16.04 µs	16.04 µs	n/a	16.04 µs

Tools for Profiling Footprint

Instruments — Virtual Memory Trace



Virtual Memory Trace 〉 By Operation

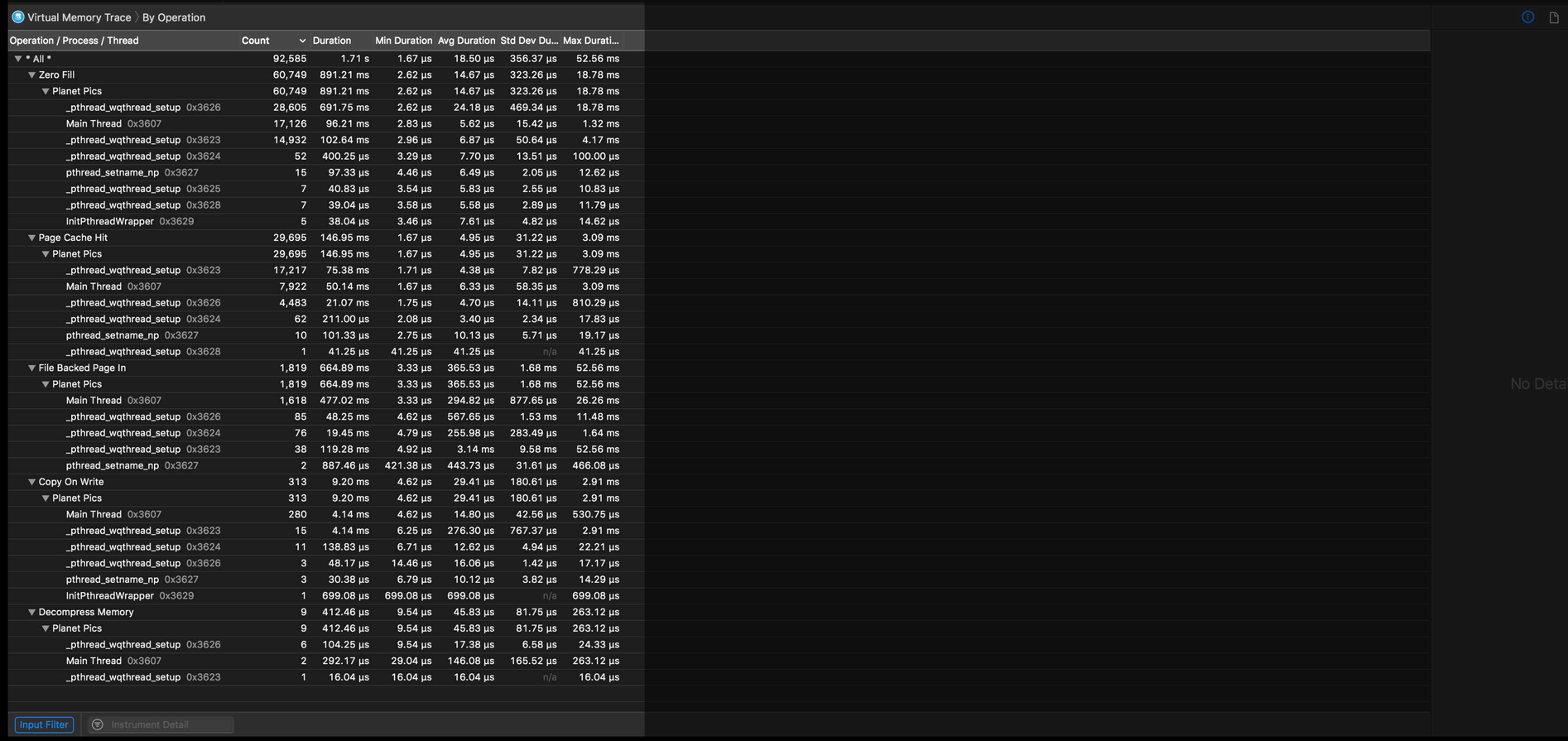
Operation / Process / Thread	Count	Duration	Min Duration	Avg Duration	Std Dev Du...	Max Durati...
▼ * All *	92,585	1.71 s	1.67 µs	18.50 µs	356.37 µs	52.56 ms
▼ Zero Fill	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
▼ Planet Pics	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
_pthread_wqthread_setup 0x3626	28,605	691.75 ms	2.62 µs	24.18 µs	469.34 µs	18.78 ms
Main Thread 0x3607	17,126	96.21 ms	2.83 µs	5.62 µs	15.42 µs	1.32 ms
_pthread_wqthread_setup 0x3623	14,932	102.64 ms	2.96 µs	6.87 µs	50.64 µs	4.17 ms
_pthread_wqthread_setup 0x3624	52	400.25 µs	3.29 µs	7.70 µs	13.51 µs	100.00 µs
pthread_setname_np 0x3627	15	97.33 µs	4.46 µs	6.49 µs	2.05 µs	12.62 µs
_pthread_wqthread_setup 0x3625	7	40.83 µs	3.54 µs	5.83 µs	2.55 µs	10.83 µs
_pthread_wqthread_setup 0x3628	7	39.04 µs	3.58 µs	5.58 µs	2.89 µs	11.79 µs
InitPthreadWrapper 0x3629	5	38.04 µs	3.46 µs	7.61 µs	4.82 µs	14.62 µs
▼ Page Cache Hit	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
▼ Planet Pics	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
_pthread_wqthread_setup 0x3623	17,217	75.38 ms	1.71 µs	4.38 µs	7.82 µs	778.29 µs
Main Thread 0x3607	7,922	50.14 ms	1.67 µs	6.33 µs	58.35 µs	3.09 ms
_pthread_wqthread_setup 0x3626	4,483	21.07 ms	1.75 µs	4.70 µs	14.11 µs	810.29 µs
_pthread_wqthread_setup 0x3624	62	211.00 µs	2.08 µs	3.40 µs	2.34 µs	17.83 µs
pthread_setname_np 0x3627	10	101.33 µs	2.75 µs	10.13 µs	5.71 µs	19.17 µs
_pthread_wqthread_setup 0x3628	1	41.25 µs	41.25 µs	41.25 µs	n/a	41.25 µs
▼ File Backed Page In	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
▼ Planet Pics	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
Main Thread 0x3607	1,618	477.02 ms	3.33 µs	294.82 µs	877.65 µs	26.26 ms
_pthread_wqthread_setup 0x3626	85	48.25 ms	4.62 µs	567.65 µs	1.53 ms	11.48 ms
_pthread_wqthread_setup 0x3624	76	19.45 ms	4.79 µs	255.98 µs	283.49 µs	1.64 ms
_pthread_wqthread_setup 0x3623	38	119.28 ms	4.92 µs	3.14 ms	9.58 ms	52.56 ms
pthread_setname_np 0x3627	2	887.46 µs	421.38 µs	443.73 µs	31.61 µs	466.08 µs
▼ Copy On Write	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
▼ Planet Pics	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
Main Thread 0x3607	280	4.14 ms	4.62 µs	14.80 µs	42.56 µs	530.75 µs
_pthread_wqthread_setup 0x3623	15	4.14 ms	6.25 µs	276.30 µs	767.37 µs	2.91 ms
_pthread_wqthread_setup 0x3624	11	138.83 µs	6.71 µs	12.62 µs	4.94 µs	22.21 µs
_pthread_wqthread_setup 0x3626	3	48.17 µs	14.46 µs	16.06 µs	1.42 µs	17.17 µs
pthread_setname_np 0x3627	3	30.38 µs	6.79 µs	10.12 µs	3.82 µs	14.29 µs
InitPthreadWrapper 0x3629	1	699.08 µs	699.08 µs	699.08 µs	n/a	699.08 µs
▼ Decompress Memory	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
▼ Planet Pics	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
_pthread_wqthread_setup 0x3626	6	104.25 µs	9.54 µs	17.38 µs	6.58 µs	24.33 µs
Main Thread 0x3607	2	292.17 µs	29.04 µs	146.08 µs	165.52 µs	263.12 µs
_pthread_wqthread_setup 0x3623	1	16.04 µs	16.04 µs	16.04 µs	n/a	16.04 µs

Input Filter | Instrument Detail

No Deta

Tools for Profiling Footprint

Instruments — Virtual Memory Trace



Virtual Memory Trace 〉 By Operation

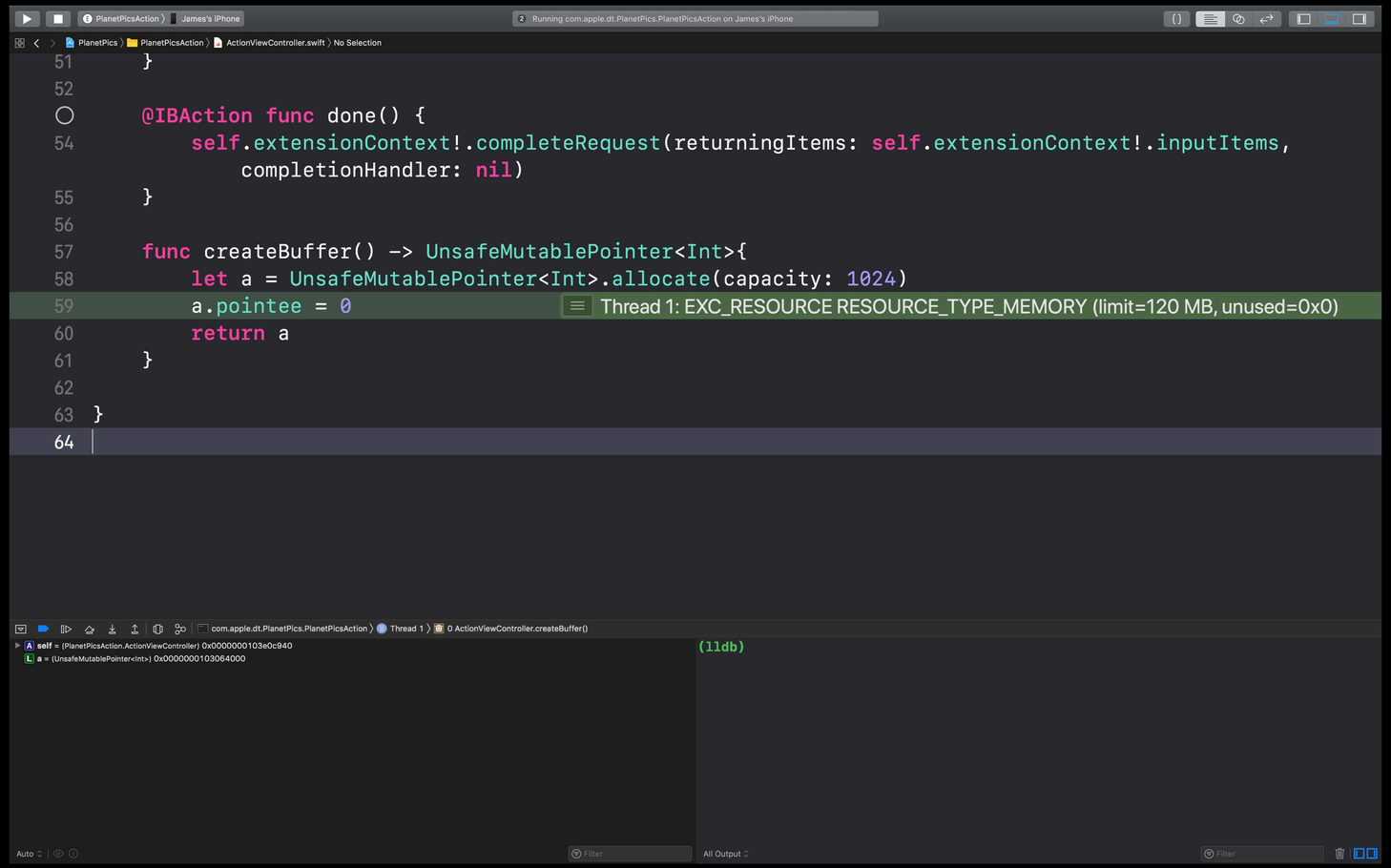
Operation / Process / Thread	Count	Duration	Min Duration	Avg Duration	Std Dev Du...	Max Durati...
▼ * All *	92,585	1.71 s	1.67 µs	18.50 µs	356.37 µs	52.56 ms
▼ Zero Fill	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
▼ Planet Pics	60,749	891.21 ms	2.62 µs	14.67 µs	323.26 µs	18.78 ms
_pthread_wqthread_setup 0x3626	28,605	691.75 ms	2.62 µs	24.18 µs	469.34 µs	18.78 ms
Main Thread 0x3607	17,126	96.21 ms	2.83 µs	5.62 µs	15.42 µs	1.32 ms
_pthread_wqthread_setup 0x3623	14,932	102.64 ms	2.96 µs	6.87 µs	50.64 µs	4.17 ms
_pthread_wqthread_setup 0x3624	52	400.25 µs	3.29 µs	7.70 µs	13.51 µs	100.00 µs
pthread_setname_np 0x3627	15	97.33 µs	4.46 µs	6.49 µs	2.05 µs	12.62 µs
_pthread_wqthread_setup 0x3625	7	40.83 µs	3.54 µs	5.83 µs	2.55 µs	10.83 µs
_pthread_wqthread_setup 0x3628	7	39.04 µs	3.58 µs	5.58 µs	2.89 µs	11.79 µs
InitPthreadWrapper 0x3629	5	38.04 µs	3.46 µs	7.61 µs	4.82 µs	14.62 µs
▼ Page Cache Hit	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
▼ Planet Pics	29,695	146.95 ms	1.67 µs	4.95 µs	31.22 µs	3.09 ms
_pthread_wqthread_setup 0x3623	17,217	75.38 ms	1.71 µs	4.38 µs	7.82 µs	778.29 µs
Main Thread 0x3607	7,922	50.14 ms	1.67 µs	6.33 µs	58.35 µs	3.09 ms
_pthread_wqthread_setup 0x3626	4,483	21.07 ms	1.75 µs	4.70 µs	14.11 µs	810.29 µs
_pthread_wqthread_setup 0x3624	62	211.00 µs	2.08 µs	3.40 µs	2.34 µs	17.83 µs
pthread_setname_np 0x3627	10	101.33 µs	2.75 µs	10.13 µs	5.71 µs	19.17 µs
_pthread_wqthread_setup 0x3628	1	41.25 µs	41.25 µs	41.25 µs	n/a	41.25 µs
▼ File Backed Page In	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
▼ Planet Pics	1,819	664.89 ms	3.33 µs	365.53 µs	1.68 ms	52.56 ms
Main Thread 0x3607	1,618	477.02 ms	3.33 µs	294.82 µs	877.65 µs	26.26 ms
_pthread_wqthread_setup 0x3626	85	48.25 ms	4.62 µs	567.65 µs	1.53 ms	11.48 ms
_pthread_wqthread_setup 0x3624	76	19.45 ms	4.79 µs	255.98 µs	283.49 µs	1.64 ms
_pthread_wqthread_setup 0x3623	38	119.28 ms	4.92 µs	3.14 ms	9.58 ms	52.56 ms
pthread_setname_np 0x3627	2	887.46 µs	421.38 µs	443.73 µs	31.61 µs	466.08 µs
▼ Copy On Write	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
▼ Planet Pics	313	9.20 ms	4.62 µs	29.41 µs	180.61 µs	2.91 ms
Main Thread 0x3607	280	4.14 ms	4.62 µs	14.80 µs	42.56 µs	530.75 µs
_pthread_wqthread_setup 0x3623	15	4.14 ms	6.25 µs	276.30 µs	767.37 µs	2.91 ms
_pthread_wqthread_setup 0x3624	11	138.83 µs	6.71 µs	12.62 µs	4.94 µs	22.21 µs
_pthread_wqthread_setup 0x3626	3	48.17 µs	14.46 µs	16.06 µs	1.42 µs	17.17 µs
pthread_setname_np 0x3627	3	30.38 µs	6.79 µs	10.12 µs	3.82 µs	14.29 µs
InitPthreadWrapper 0x3629	1	699.08 µs	699.08 µs	699.08 µs	n/a	699.08 µs
▼ Decompress Memory	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
▼ Planet Pics	9	412.46 µs	9.54 µs	45.83 µs	81.75 µs	263.12 µs
_pthread_wqthread_setup 0x3626	6	104.25 µs	9.54 µs	17.38 µs	6.58 µs	24.33 µs
Main Thread 0x3607	2	292.17 µs	29.04 µs	146.08 µs	165.52 µs	263.12 µs
_pthread_wqthread_setup 0x3623	1	16.04 µs	16.04 µs	16.04 µs	n/a	16.04 µs

Input Filter | Instrument Detail

Tools for Profiling Footprint

Xcode Debugger — memory resource exceptions

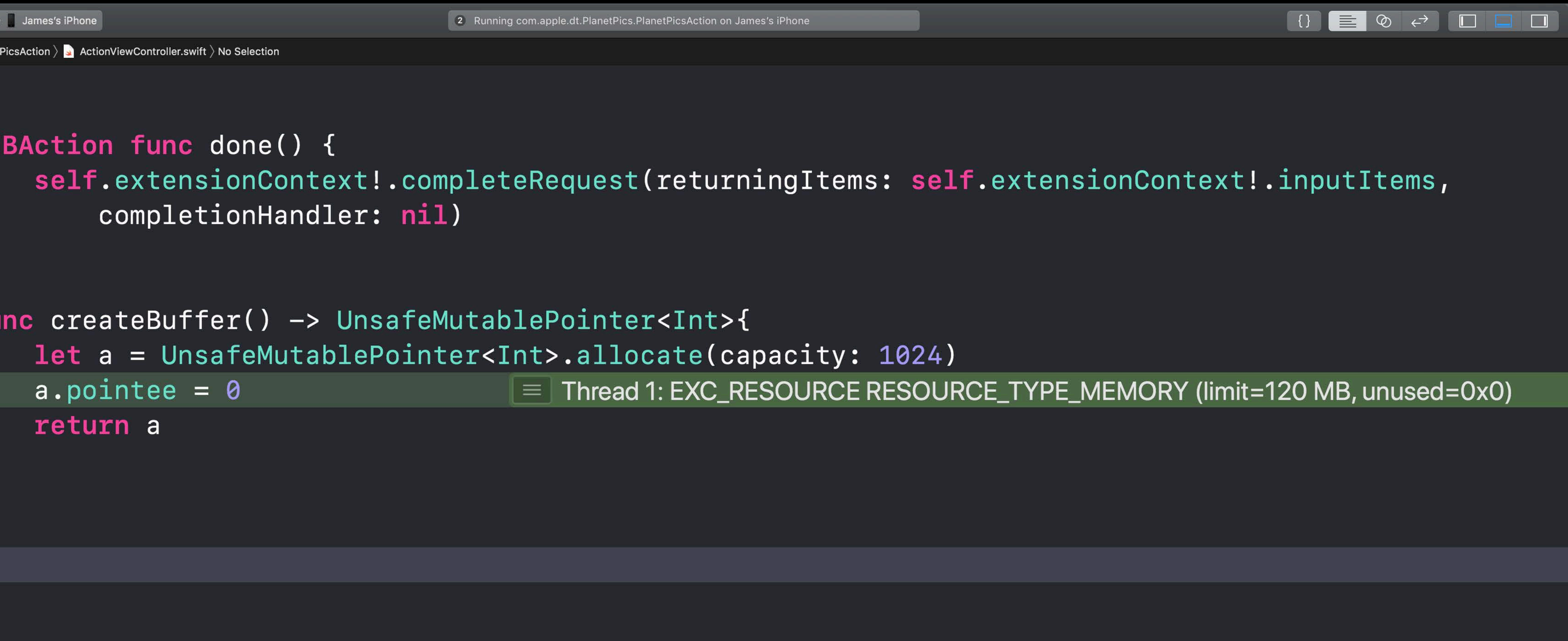
NEW



Tools for Profiling Footprint

Xcode Debugger — memory resource exceptions

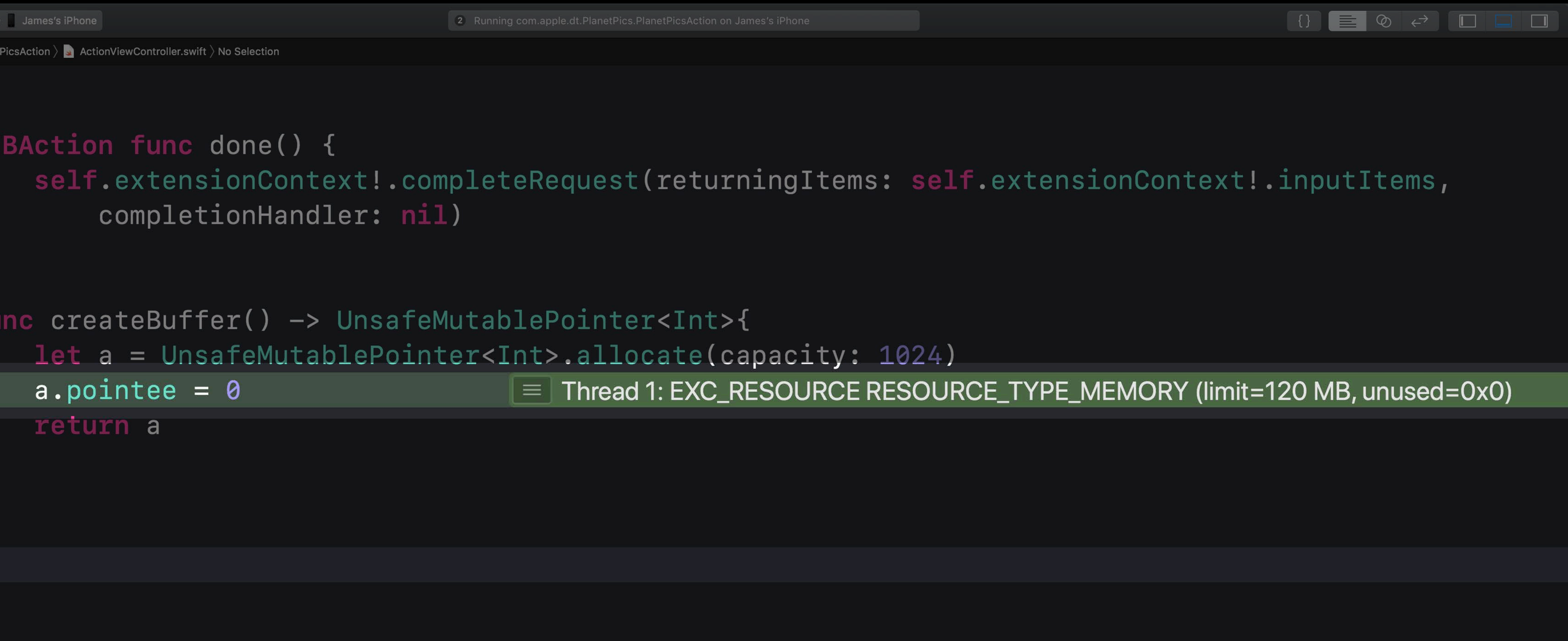
NEW



Tools for Profiling Footprint

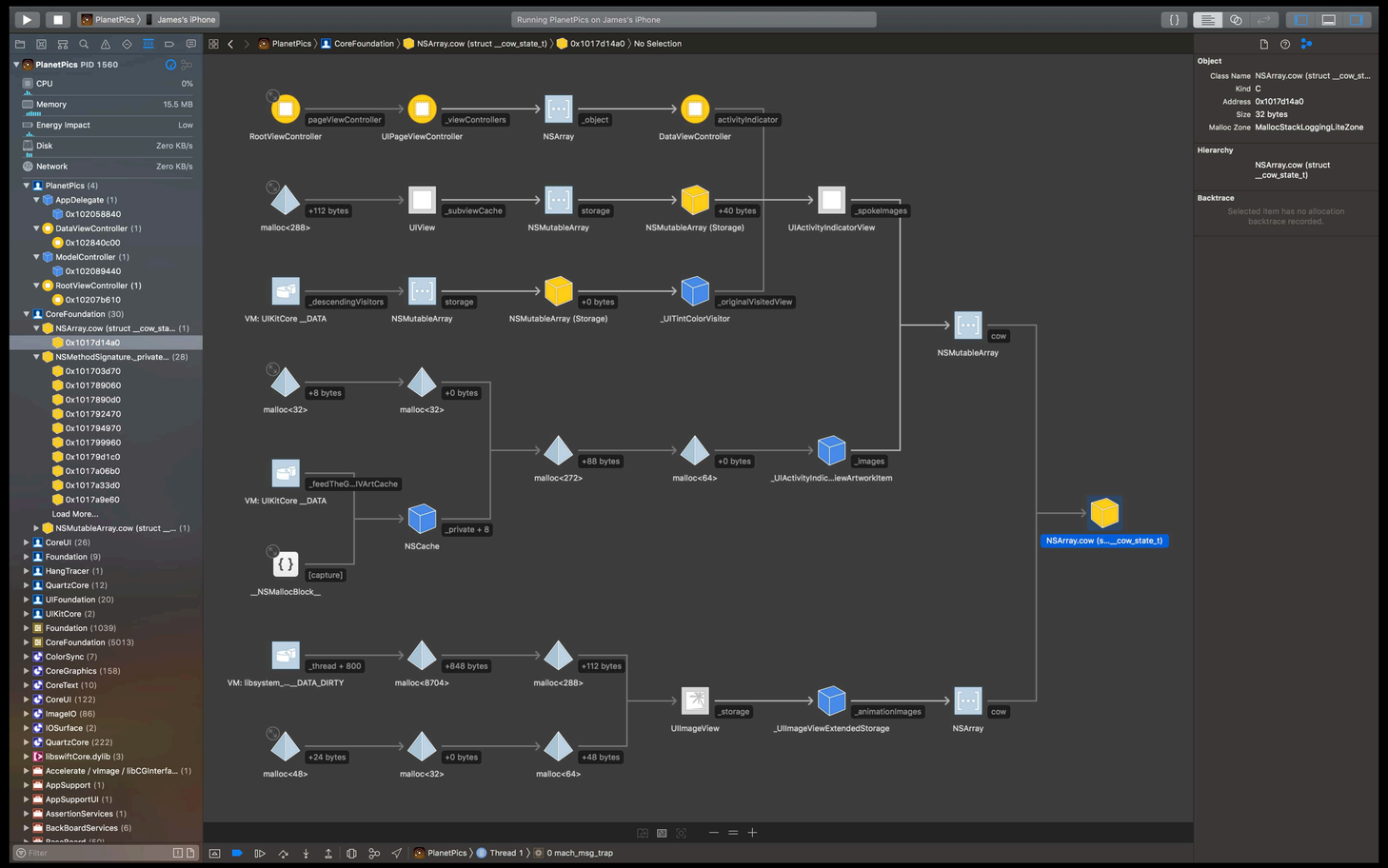
Xcode Debugger — memory resource exceptions

NEW



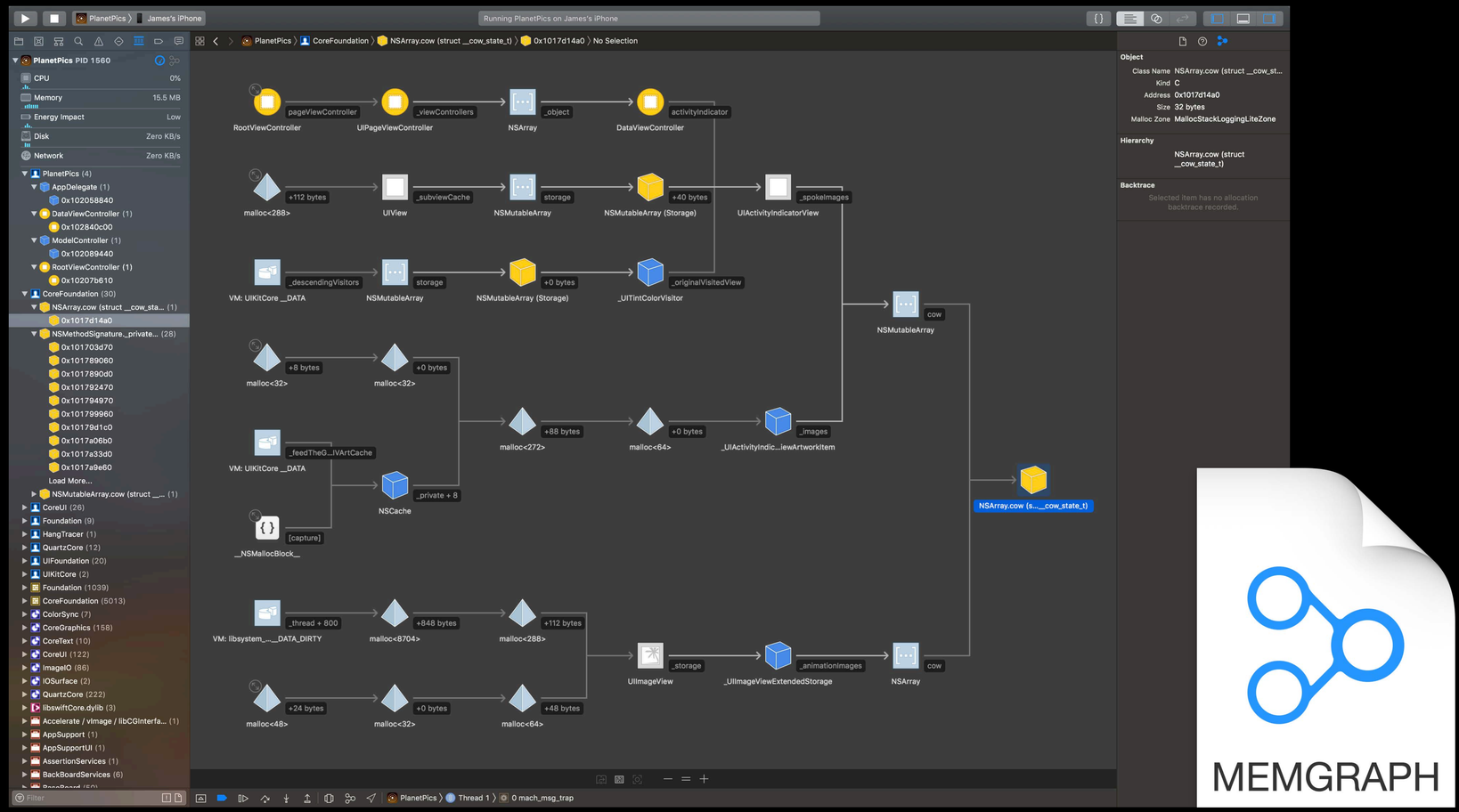
Tools for Profiling Footprint

Xcode Memory Debugger



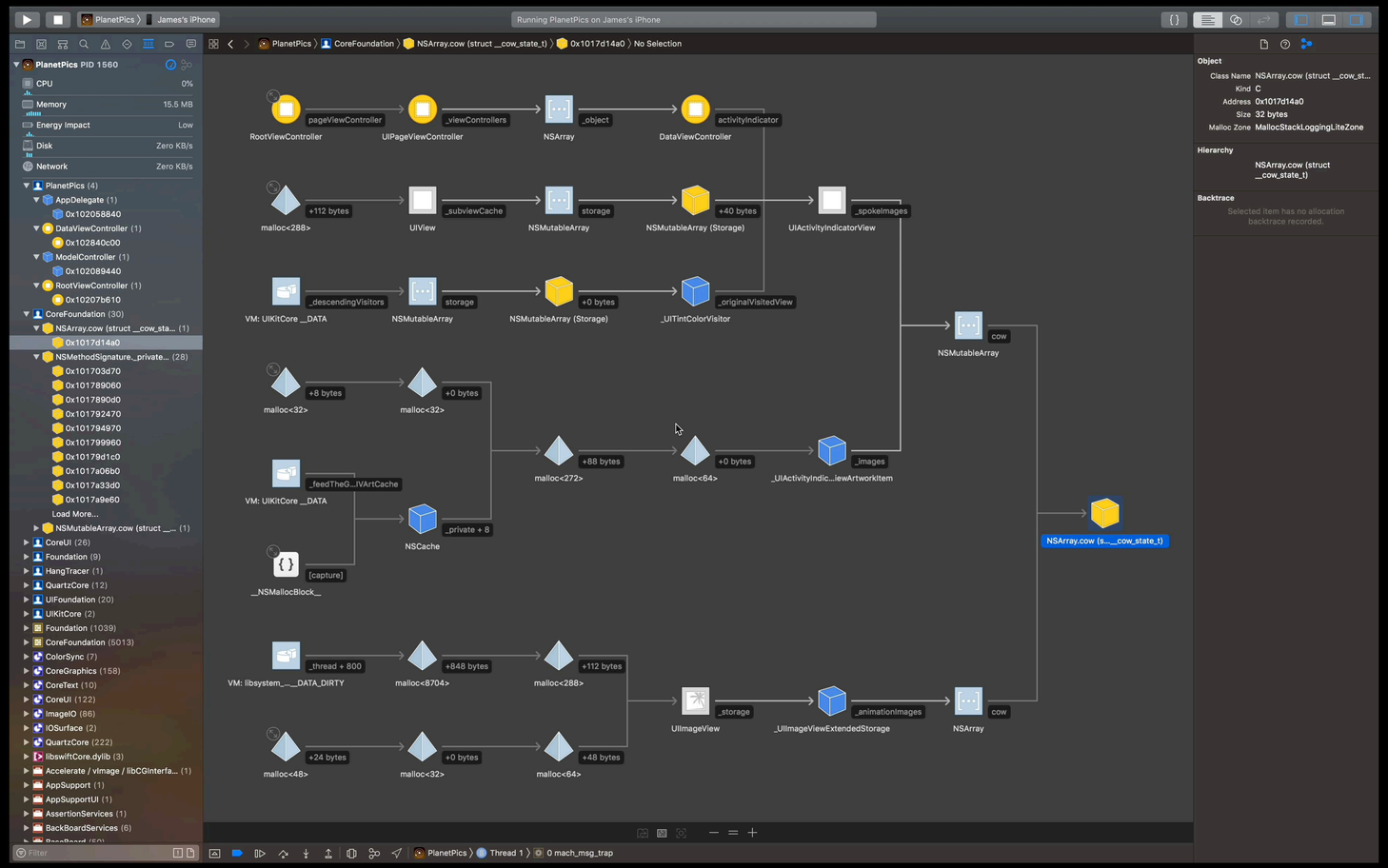
Tools for Profiling Footprint

Xcode Memory Debugger



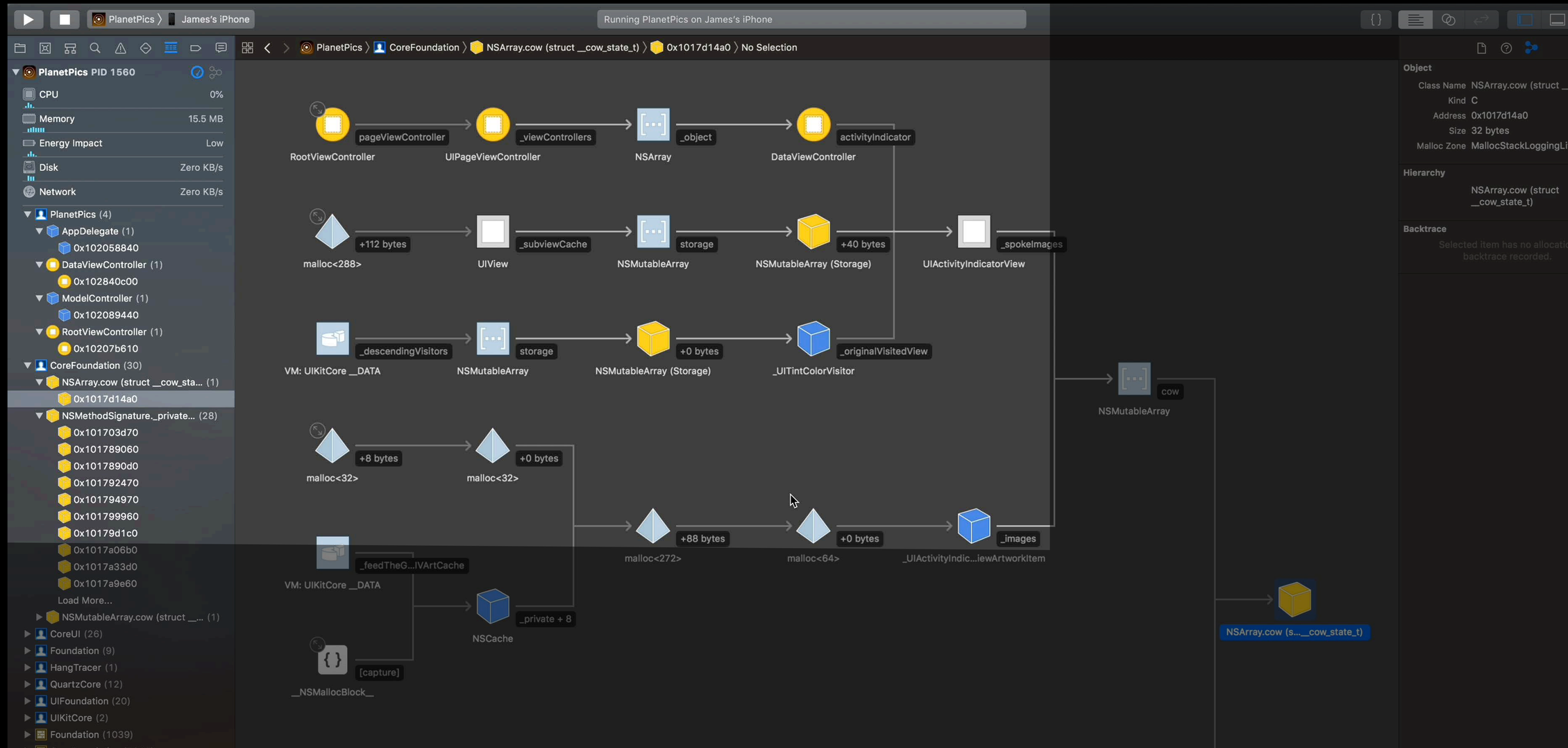
Tools for Profiling Footprint

Xcode Memory Debugger



Tools for Profiling Footprint

Xcode Memory Debugger



Tools for Profiling Footprint

vmmap

NEW

Shows virtual memory regions allocated in a process

```
vmmap App.memgraph
vmmap --summary App.memgraph
```

Tools for Profiling Footprint

vmmap --summary App.memgraph

planetstagram — -bash — 159×45

```
$ vmmap --summary PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

ReadOnly portion of Libraries: Total=373.4M resident=102.7M(27%) swapped_out_or_unallocated=270.7M(73%)
Writable regions: Total=2.0G written=1.2G(59%) resident=804.6M(39%) swapped_out=689.7M(33%) unallocated=591.8M(28%)

                                VIRTUAL RESIDENT    DIRTY  SWAPPED VOLATILE   NONVOL    EMPTY   REGION
REGION TYPE                        SIZE     SIZE     SIZE     SIZE     SIZE     SIZE     SIZE   COUNT (non-coalesced)
===========                     ======= ========    =====  ======= ========   ======    =====  =======
Activity Tracing                   256K       32K      32K       0K       0K      32K       0K        2
CG image                         976.6M    326.0M   326.0M   650.6M       0K       0K       0K        3
CoreAnimation                      448K       432K     432K       0K       0K     192K      16K       16
CoreImage                         64.0M     64.0M    64.0M       0K       0K       0K       0K        2
CoreUI image data                 1008K     1008K    1008K       0K       0K       0K       0K       10
CoreUI image file                  128K      128K     128K       0K       0K       0K       0K        2
Foundation                         976K      976K     976K       0K       0K       0K       0K        4
IOKit                             3168K      688K     688K       0K       0K     640K       0K       13
IOSurface                       192.2M    192.2M   192.2M       0K   160.1M    32.0M       0K       10
Image IO                         50.5M     41.7M    41.7M       0K       0K    39.2M       0K        3
Kernel Alloc Once                  32K       16K      16K       0K       0K       0K       0K        2
MALLOC guard page                 192K        0K       0K       0K       0K       0K       0K       10
MALLOC metadata                   240K      240K     240K       0K       0K       0K       0K       16
MALLOC_LARGE                    244.1M    170.2M   170.2M    39.1M       0K       0K       0K       24         see MALLOC ZONE table below
MALLOC_LARGE metadata             16K       16K      16K       0K       0K       0K       0K        2         see MALLOC ZONE table below
MALLOC_NANO                     512.0M       64K      64K       0K       0K       0K       0K        2         see MALLOC ZONE table below
MALLOC_SMALL                     24.0M     1312K    1296K       0K       0K       0K       0K        4         see MALLOC ZONE table below
MALLOC_TINY                      7168K     2864K    2864K       0K       0K       0K       0K        6         see MALLOC ZONE table below
Performance tool data            4496K     4496K    4496K       0K       0K       0K       0K        5         not counted in TOTAL below
STACK GUARD                       144K        0K       0K       0K       0K       0K       0K       10
```

Tools for Profiling Footprint

vmmap --summary App.memgraph

planetstagram — -bash — 159×45

```
$ vmmap --summary PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

ReadOnly portion of Libraries: Total=373.4M resident=102.7M(27%) swapped_out_or_unallocated=270.7M(73%)
Writable regions: Total=2.0G written=1.2G(59%) resident=804.6M(39%) swapped_out=689.7M(33%) unallocated=591.8M(28%)

                                VIRTUAL  RESIDENT    DIRTY  SWAPPED VOLATILE   NONVOL    EMPTY   REGION
REGION TYPE                        SIZE      SIZE     SIZE     SIZE     SIZE     SIZE     SIZE    COUNT (non-coalesced)
===========                     =======  ========  =======  =======  ========  ======  =======  =======
Activity Tracing                   256K       32K       32K       0K       0K      32K       0K        2
CG image                         976.6M    326.0M    326.0M   650.6M       0K       0K       0K        3
CoreAnimation                      448K      432K      432K       0K       0K     192K      16K       16
CoreImage                         64.0M     64.0M     64.0M       0K       0K       0K       0K        2
CoreUI image data                 1008K     1008K     1008K       0K       0K       0K       0K       10
CoreUI image file                  128K      128K      128K       0K       0K       0K       0K        2
Foundation                         976K      976K      976K       0K       0K       0K       0K        4
IOKit                             3168K      688K      688K       0K       0K     640K       0K       13
IOSurface                       192.2M    192.2M    192.2M       0K   160.1M    32.0M       0K       10
Image IO                         50.5M     41.7M     41.7M       0K       0K    39.2M       0K        3
Kernel Alloc Once                  32K       16K       16K       0K       0K       0K       0K        2
MALLOC guard page                 192K        0K        0K       0K       0K       0K       0K       10
```

Tools for Profiling Footprint

vmmap --summary App.memgraph

planetstagram — -bash — 159×45

```
$ vmmap --summary PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

ReadOnly portion of Libraries: Total=373.4M resident=102.7M(27%) swapped_out_or_unallocated=270.7M(73%)
Writable regions: Total=2.0G written=1.2G(59%) resident=804.6M(39%) swapped_out=689.7M(33%) unallocated=591.8M(28%)

                                VIRTUAL RESIDENT    DIRTY  SWAPPED VOLATILE   NONVOL    EMPTY   REGION
REGION TYPE                        SIZE     SIZE     SIZE     SIZE     SIZE     SIZE     SIZE    COUNT (non-coalesced)
===========                     ======= ========    =====  ======= ========   ======    =====  =======
Activity Tracing                   256K      32K     32K       0K       0K      32K       0K        2
CG image                         976.6M   326.0M  326.0M   650.6M       0K       0K       0K        3
CoreAnimation                      448K     432K    432K       0K       0K     192K      16K       16
CoreImage                         64.0M    64.0M   64.0M       0K       0K       0K       0K        2
CoreUI image data                 1008K    1008K   1008K       0K       0K       0K       0K       10
CoreUI image file                  128K     128K    128K       0K       0K       0K       0K        2
Foundation                         976K     976K    976K       0K       0K       0K       0K        4
IOKit                             3168K     688K    688K       0K       0K     640K       0K       13
IOSurface                        192.2M   192.2M  192.2M       0K   160.1M    32.0M       0K       10
Image IO                          50.5M    41.7M   41.7M       0K       0K    39.2M       0K        3
Kernel Alloc Once                  32K      16K     16K       0K       0K       0K       0K        2
MALLOC guard page                 192K       0K      0K       0K       0K       0K       0K       10
```

Tools for Profiling Footprint

vmmap --summary App.memgraph

planetstagram — -bash — 159×45

```
$ vmmap --summary PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

ReadOnly portion of Libraries: Total=373.4M resident=102.7M(27%) swapped_out_or_unallocated=270.7M(73%)
Writable regions: Total=2.0G written=1.2G(59%) resident=804.6M(39%) swapped_out=689.7M(33%) unallocated=591.8M(28%)

                                VIRTUAL RESIDENT    DIRTY   SWAPPED VOLATILE   NONVOL    EMPTY   REGION
REGION TYPE                        SIZE     SIZE     SIZE      SIZE     SIZE     SIZE     SIZE    COUNT (non-coalesced)
===========                     ======= ========    =====   ======= ========   ======    =====  =======
Activity Tracing                  256K      32K      32K        0K       0K      32K       0K        2
CG image                          976.6M   326.0M   326.0M   650.6M       0K       0K       0K        3
CoreAnimation                     448K     432K     432K       0K       0K     192K      16K       16
CoreImage                         64.0M    64.0M    64.0M      0K       0K       0K       0K        2
CoreUI image data                 1008K    1008K    1008K      0K       0K       0K       0K       10
CoreUI image file                 128K     128K     128K       0K       0K       0K       0K        2
Foundation                        976K     976K     976K       0K       0K       0K       0K        4
IOKit                             3168K    688K     688K       0K       0K     640K       0K       13
IOSurface                         192.2M   192.2M   192.2M     0K   160.1M    32.0M       0K       10
Image IO                          50.5M    41.7M    41.7M      0K       0K    39.2M       0K        3
Kernel Alloc Once                 32K      16K      16K        0K       0K       0K       0K        2
MALLOC guard page                 192K       0K       0K       0K       0K       0K       0K       10
```

Tools for Profiling Footprint

vmmap App.memgraph

```
planetstagram — -bash — 159×45
$ vmmap PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Virtual Memory Map of process 3079 (PlanetPics)
Output report format:  2.4  -- 64-bit process
VM page size:  16384 bytes

==== Non-writable regions for process 3079
REGION TYPE                    START - END             [ VSIZE  RSDNT  DIRTY   SWAP] PRT/MAX SHRMOD PURGE    REGION DETAIL
__TEXT                 0000000100000000-000000010001c000 [  112K   112K     0K     0K] r-x/r-x SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 000000010001c000-0000000100020000 [   16K    16K    16K     0K] r-x/rwx SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 0000000100020000-0000000100024000 [   16K    16K     0K     0K] r-x/r-x SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__LINKEDIT             000000010002c000-0000000100064000 [  224K   224K     0K     0K] r--/r-- SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 0000000100064000-0000000100084000 [  128K   112K     0K     0K] r-x/r-x SM=COW          /usr/lib/dyld
__TEXT                 0000000100084000-0000000100088000 [   16K    16K    16K     0K] r-x/rwx SM=COW          /usr/lib/dyld
__TEXT                 0000000100088000-00000001000c0000 [  224K   224K     0K     0K] r-x/r-x SM=COW          /usr/lib/dyld
__LINKEDIT             00000001000f8000-0000000100128000 [  192K    16K     0K     0K] r--/r-- SM=COW          /usr/lib/dyld
__LINKEDIT             0000000100128000-000000010012c000 [   16K     0K     0K     0K] r--/r-- SM=NUL          /usr/lib/dyld
shared memory          000000010012c000-0000000100130000 [   16K    16K    16K     0K] r--/r-- SM=SHM
__TEXT                 0000000100130000-000000010013c000 [   48K    48K     0K     0K] r-x/rwx SM=COW          /Developer/usr/lib/libBacktraceRecording.dylib
__LINKEDIT             0000000100140000-0000000100144000 [   16K    16K     0K     0K] r--/rw- SM=COW          /Developer/usr/lib/libBacktraceRecording.dylib
__TEXT                 0000000100148000-000000010018c000 [  272K   112K    16K     0K] r-x/rwx SM=COW          /Developer/usr/lib/libMainThreadChecker.dylib
__LINKEDIT             0000000100350000-0000000100354000 [   16K    16K     0K     0K] r--/rw- SM=COW          /Developer/usr/lib/libMainThreadChecker.dylib
__TEXT                 0000000100358000-0000000100390000 [  224K    64K     0K     0K] r-x/rwx SM=COW          ...upport.framework/libViewDebuggerSupport.dylib
__LINKEDIT             00000001003ac000-00000001003b8000 [   48K    48K     0K     0K] r--/rw- SM=COW          ...upport.framework/libViewDebuggerSupport.dylib
__TEXT                 00000001003b8000-000000010072c000 [ 3536K  1520K    16K     0K] r-x/rwx SM=COW          .../PlanetPics.app/Frameworks/libswiftCore.dylib
__LINKEDIT             000000010077c000-0000000100a24000 [ 2720K  2640K     0K     0K] r--/rw- SM=COW          .../PlanetPics.app/Frameworks/libswiftCore.dylib
__TEXT                 0000000100a6c000-0000000100a74000 [   32K    16K     0K     0K] r-x/rwx SM=COW          ...s.app/Frameworks/libswiftCoreFoundation.dylib
__LINKEDIT             0000000100a78000-0000000100a7c000 [   16K    16K     0K     0K] r--/rw- SM=COW          ...s.app/Frameworks/libswiftCoreFoundation.dylib
```

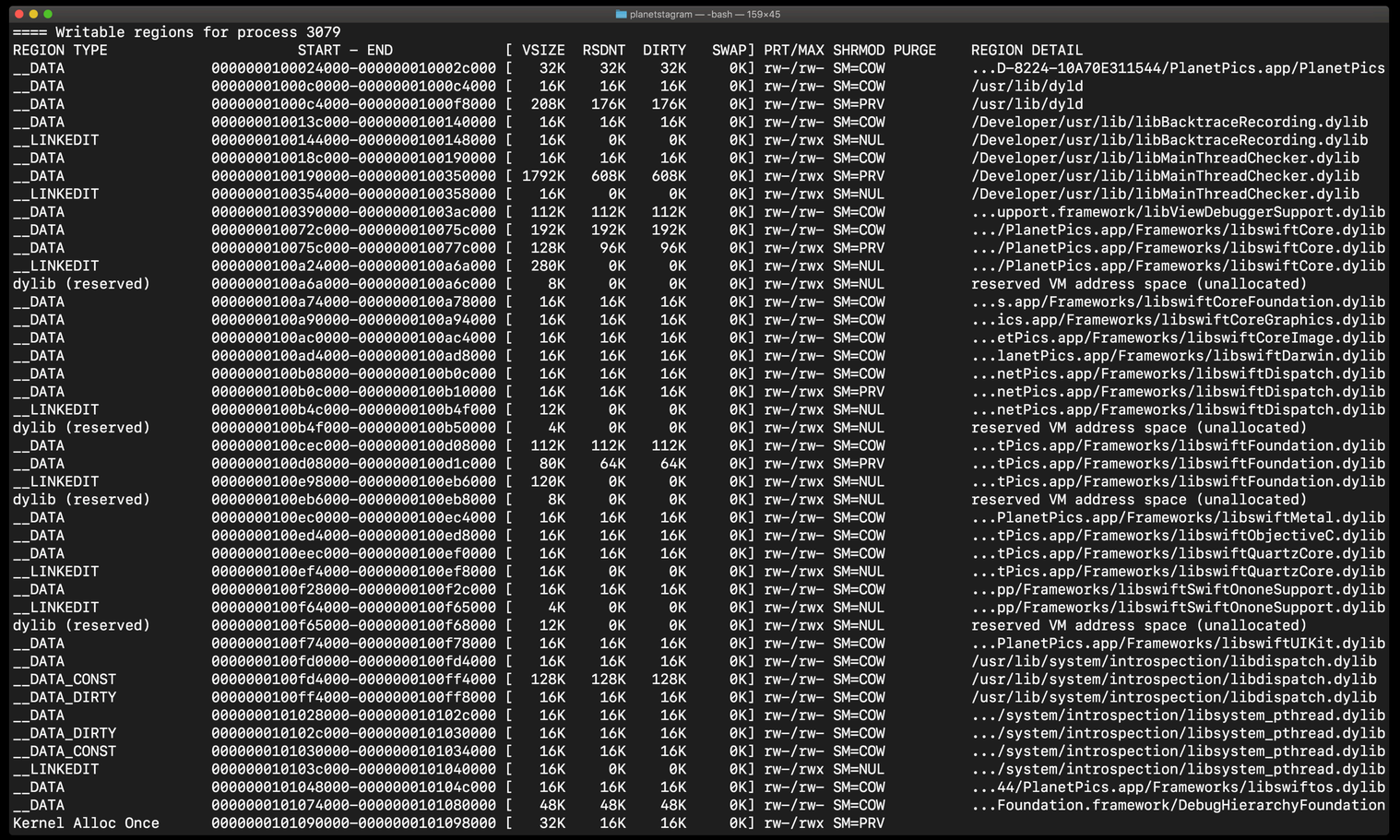
Tools for Profiling Footprint

vmmap App.memgraph

```
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Virtual Memory Map of process 3079 (PlanetPics)
Output report format:  2.4  -- 64-bit process
VM page size:  16384 bytes

==== Non-writable regions for process 3079
REGION TYPE                      START - END             [ VSIZE  RSDNT  DIRTY   SWAP] PRT/MAX SHRMOD PURGE    REGION DETAIL
__TEXT                 0000000100000000-000000010001c000 [  112K   112K     0K     0K] r-x/r-x SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 000000010001c000-0000000100020000 [   16K    16K    16K     0K] r-x/rwx SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 0000000100020000-0000000100024000 [   16K    16K     0K     0K] r-x/r-x SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__LINKEDIT             000000010002c000-0000000100064000 [  224K   224K     0K     0K] r--/r-- SM=COW          ...D-8224-10A70E311544/PlanetPics.app/PlanetPics
__TEXT                 0000000100064000-0000000100084000 [  128K   112K     0K     0K] r-x/r-x SM=COW          /usr/lib/dyld
__TEXT                 0000000100084000-0000000100088000 [   16K    16K    16K     0K] r-x/rwx SM=COW          /usr/lib/dyld
__TEXT                 0000000100088000-00000001000c0000 [  224K   224K     0K     0K] r-x/r-x SM=COW          /usr/lib/dyld
__LINKEDIT             00000001000f8000-0000000100128000 [  192K    16K     0K     0K] r--/r-- SM=COW          /usr/lib/dyld
__LINKEDIT             0000000100128000-000000010012c000 [   16K     0K     0K     0K] r--/r-- SM=NUL          /usr/lib/dyld
shared memory          000000010012c000-0000000100130000 [   16K    16K    16K     0K] r--/r-- SM=SHM
__TEXT                 0000000100130000-000000010013c000 [   48K    48K     0K     0K] r-x/rwx SM=COW          /Developer/usr/lib/libBacktraceRecording.dylib
__LINKEDIT             0000000100140000-0000000100144000 [   16K    16K     0K     0K] r--/rw- SM=COW          /Developer/usr/lib/libBacktraceRecording.dylib
__TEXT                 0000000100148000-000000010018c000 [  272K   112K    16K     0K] r-x/rwx SM=COW          /Developer/usr/lib/libMainThreadChecker.dylib
__LINKEDIT             0000000100350000-0000000100354000 [   16K    16K     0K     0K] r--/rw- SM=COW          /Developer/usr/lib/libMainThreadChecker.dylib
__TEXT                 0000000100358000-0000000100390000 [  224K    64K     0K     0K] r-x/rwx SM=COW          ...upport.framework/libViewDebuggerSupport.dylib
__LINKEDIT             00000001003ac000-00000001003b8000 [   48K    48K     0K     0K] r--/rw- SM=COW          ...upport.framework/libViewDebuggerSupport.dylib
__TEXT                 00000001003b8000-000000010072c000 [ 3536K  1520K    16K     0K] r-x/rwx SM=COW          .../PlanetPics.app/Frameworks/libswiftCore.dylib
__LINKEDIT             000000010077c000-0000000100a24000 [ 2720K  2640K     0K     0K] r--/rw- SM=COW          .../PlanetPics.app/Frameworks/libswiftCore.dylib
__TEXT                 0000000100a6c000-0000000100a74000 [   32K    16K     0K     0K] r-x/rwx SM=COW          ...s.app/Frameworks/libswiftCoreFoundation.dylib
__LINKEDIT             0000000100a78000-0000000100a7c000 [   16K    16K     0K     0K] r--/rw- SM=COW          ...s.app/Frameworks/libswiftCoreFoundation.dylib
```

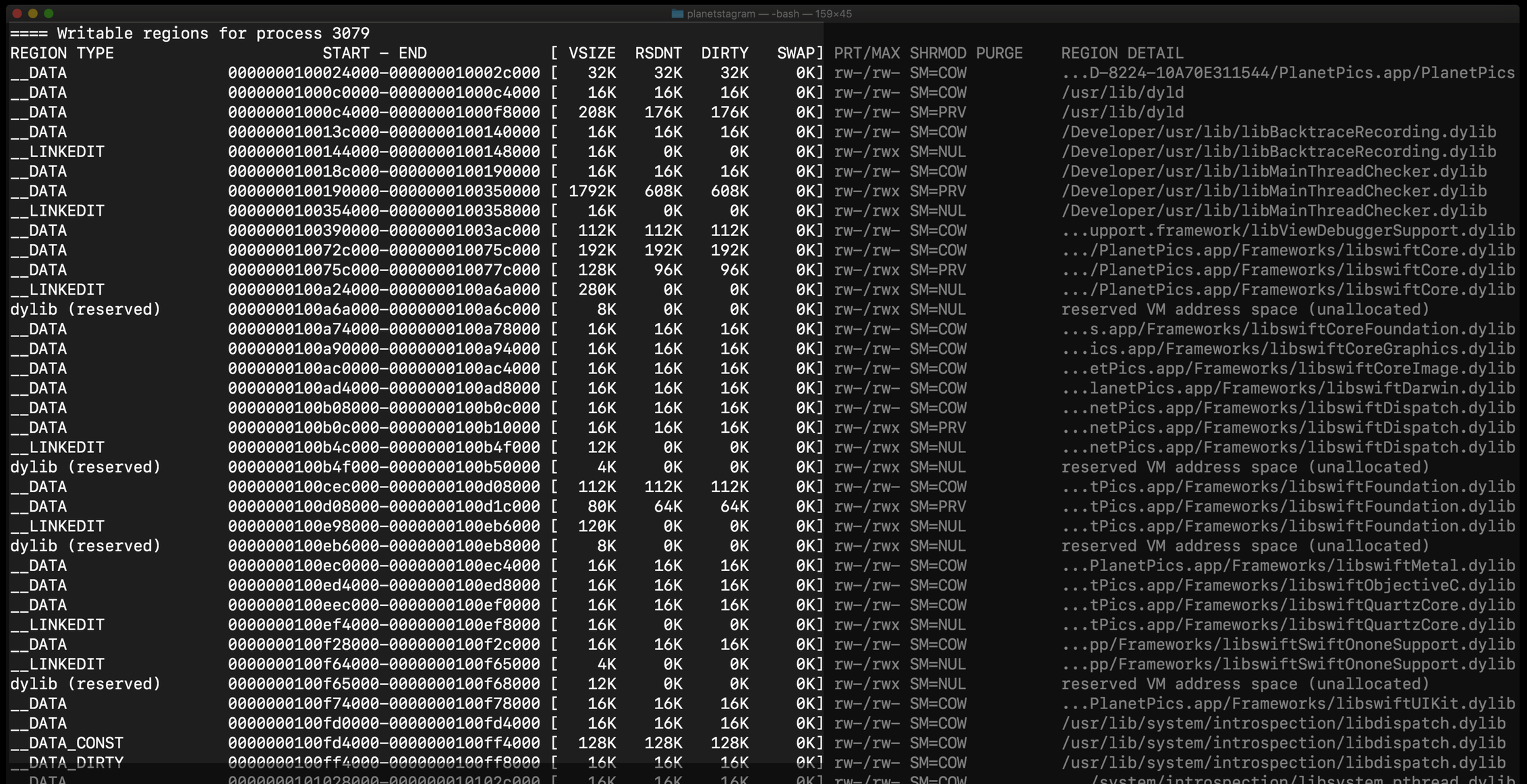
Tools for Profiling Footprint

vmmap App.memgraph



Tools for Profiling Footprint

vmmap App.memgraph

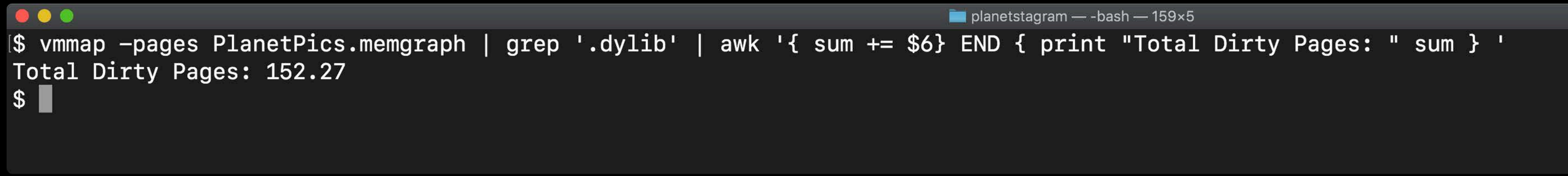


Tools for Profiling Footprint

vmmap and AWK

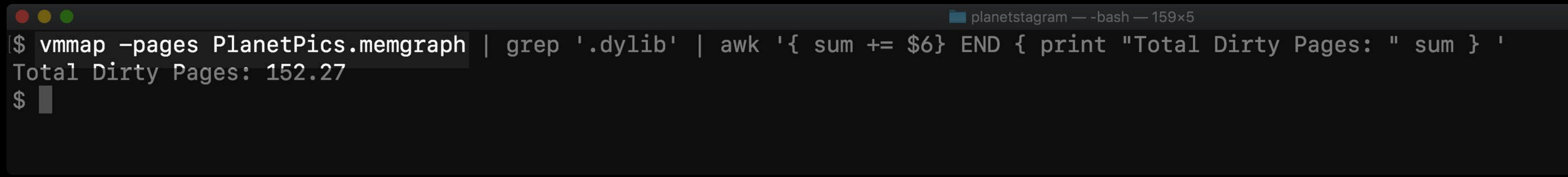
Tools for Profiling Footprint

vmmap and AWK



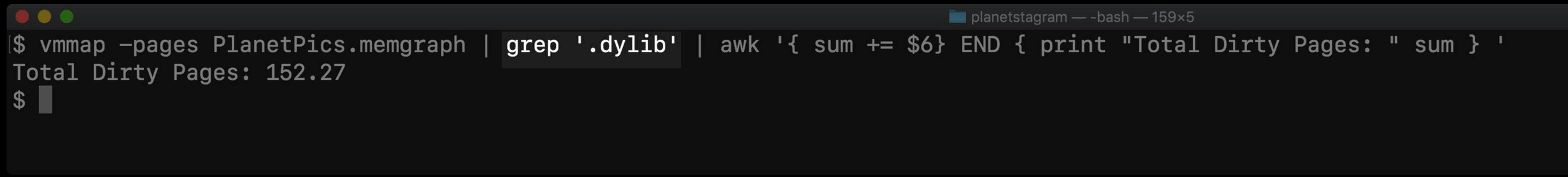
Tools for Profiling Footprint

vmmap and AWK



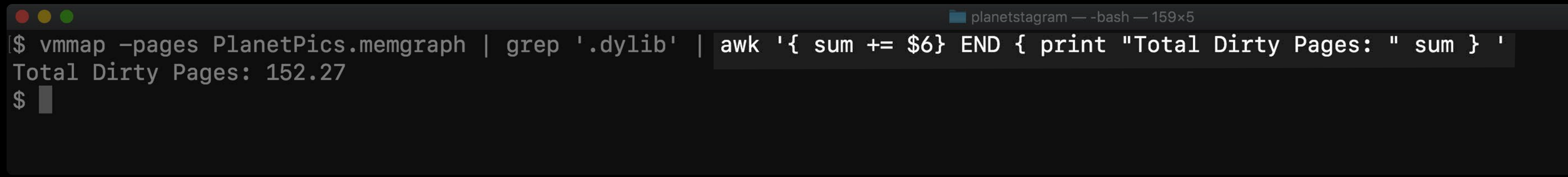
Tools for Profiling Footprint

vmmap and AWK



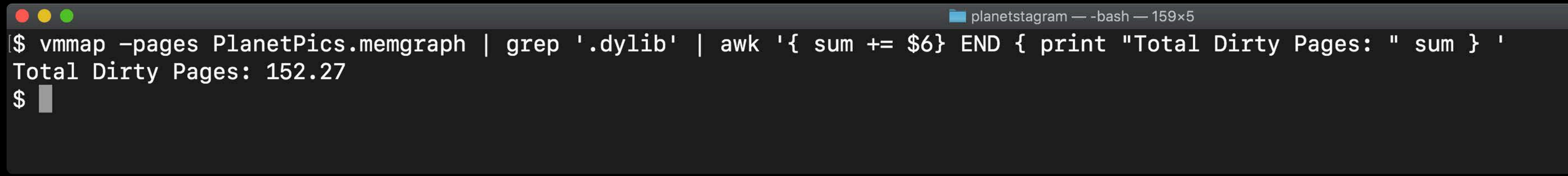
Tools for Profiling Footprint

vmmap and AWK



Tools for Profiling Footprint

vmmap and AWK



Tools for Profiling Footprint

leaks

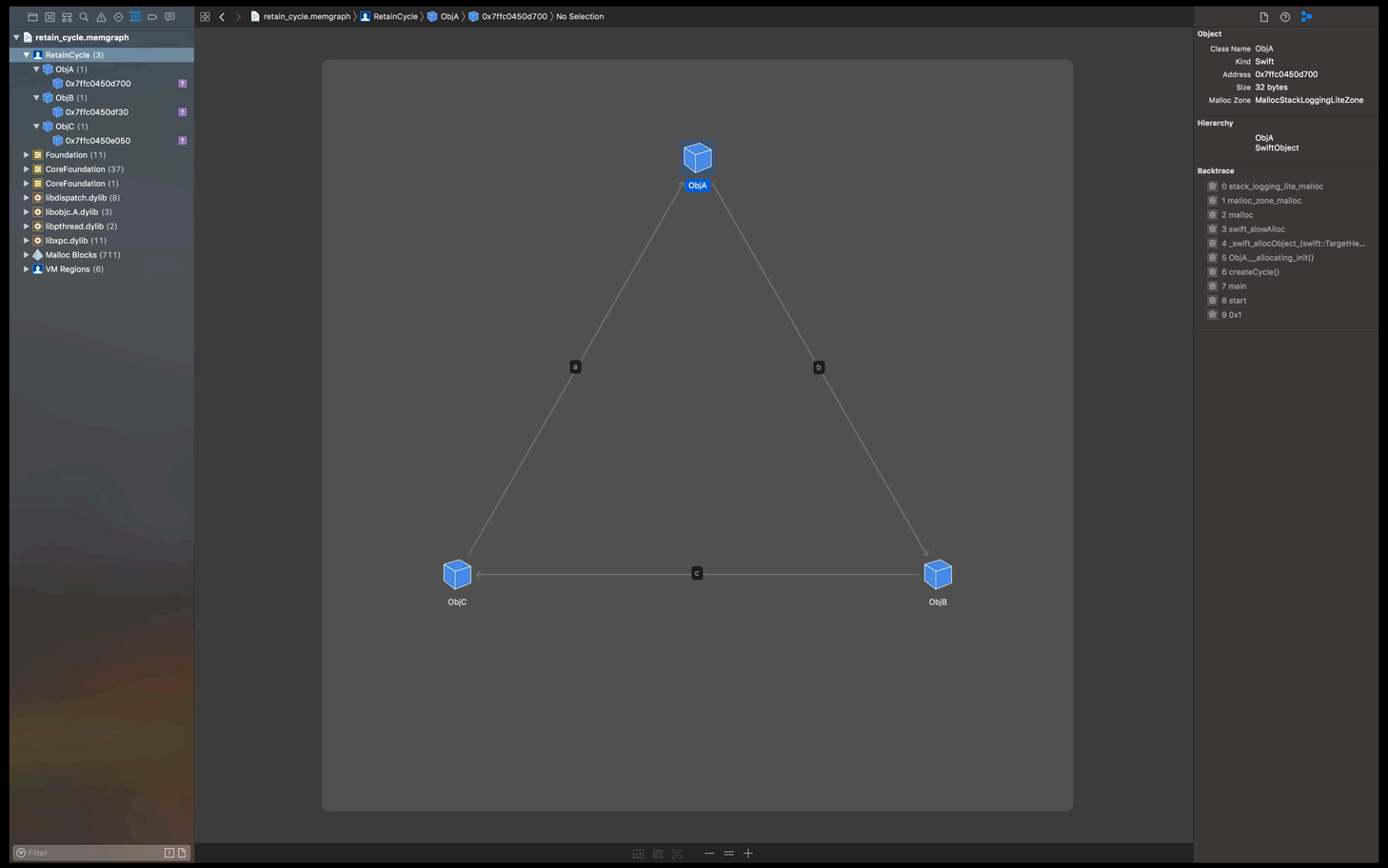
NEW

Shows objects that are allocated, but no longer referenced

```
leaks App.memgraph
```

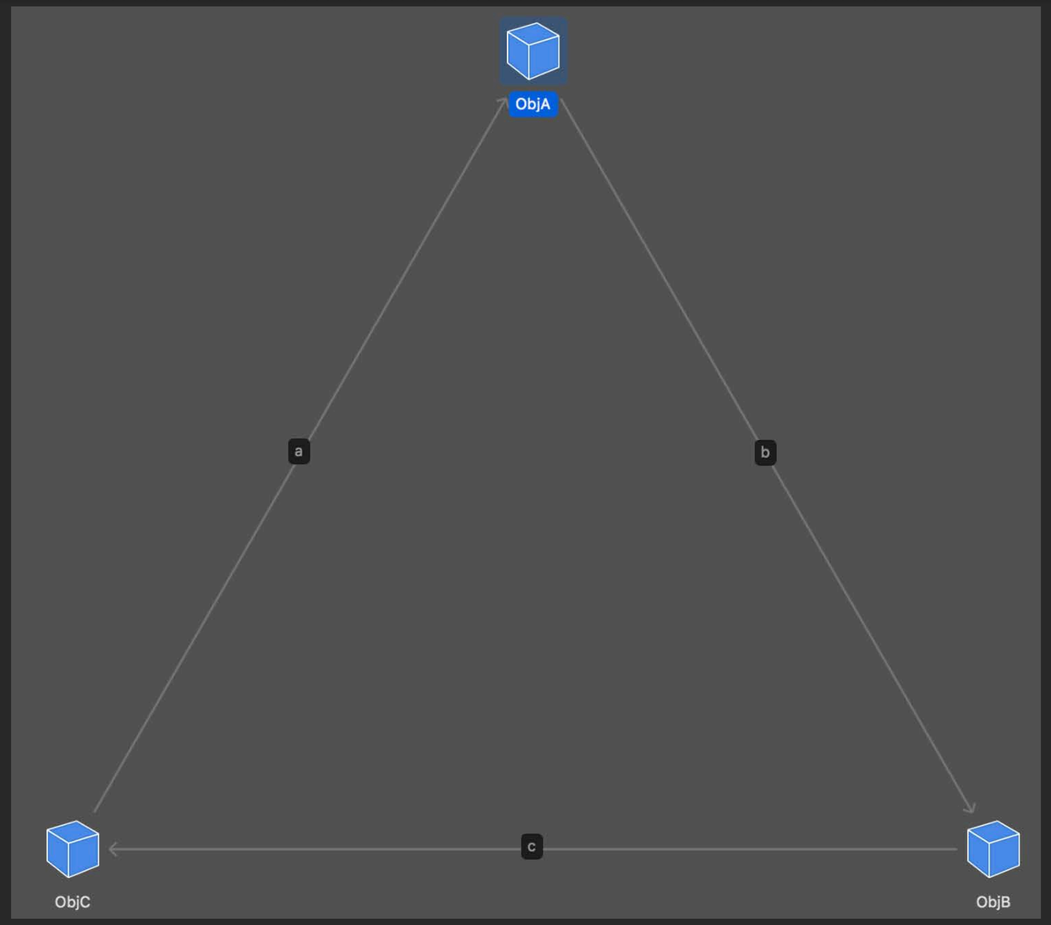
Tools for Profiling Footprint

leaks



Tools for Profiling Footprint

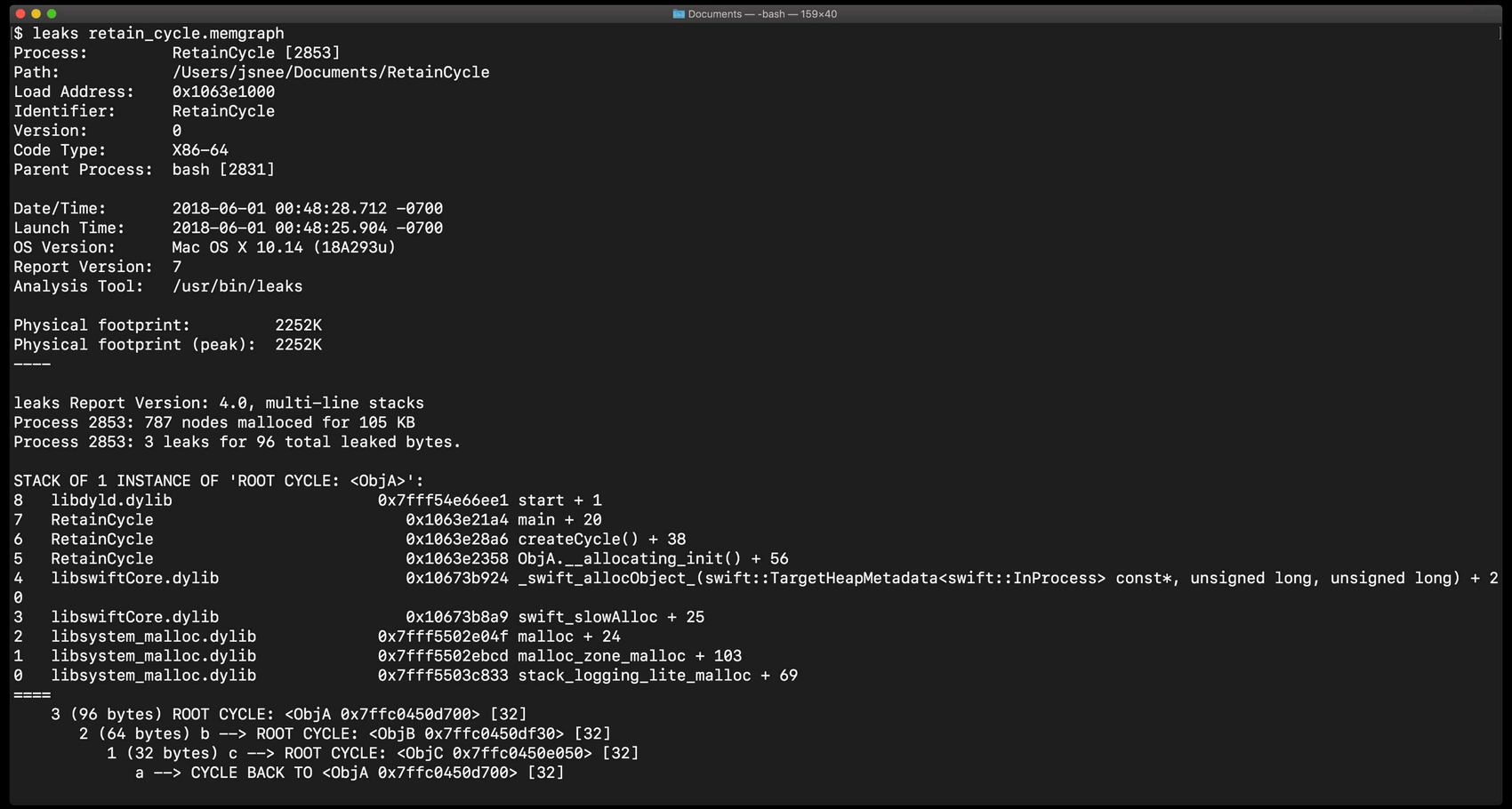
leaks



Tools for Profiling Footprint

leaks MyApp.memgraph

NEW



Tools for Profiling Footprint

leaks MyApp.memgraph

NEW

```
Version:         0
Code Type:       X86-64
Parent Process:  bash [2831]

Date/Time:       2018-06-01 00:48:28.712 -0700
Launch Time:     2018-06-01 00:48:25.904 -0700
OS Version:      Mac OS X 10.14 (18A293u)
Report Version:  7
Analysis Tool:   /usr/bin/leaks

Physical footprint:         2252K
Physical footprint (peak):  2252K
----

leaks Report Version: 4.0, multi-line stacks
Process 2853: 787 nodes malloced for 105 KB
Process 2853: 3 leaks for 96 total leaked bytes.

STACK OF 1 INSTANCE OF 'ROOT CYCLE: <ObjA>':
8   libdyld.dylib                   0x7fff54e66ee1 start + 1
7   RetainCycle                        0x1063e21a4 main + 20
6   RetainCycle                        0x1063e28a6 createCycle() + 38
5   RetainCycle                        0x1063e2358 ObjA.__allocating_init() + 56
4   libswiftCore.dylib                 0x10673b924 _swift_allocObject_(swift::TargetHeapMetadata<swift::InProcess> const*, unsigned long, unsigned long) + 2
0
3   libswiftCore.dylib                 0x10673b8a9 swift_slowAlloc + 25
2   libsystem_malloc.dylib          0x7fff5502e04f malloc + 24
1   libsystem_malloc.dylib          0x7fff5502ebcd malloc_zone_malloc + 103
0   libsystem_malloc.dylib          0x7fff5503c833 stack_logging_lite_malloc + 69
====
    3 (96 bytes) ROOT CYCLE: <ObjA 0x7ffc0450d700> [32]
       2 (64 bytes) b --> ROOT CYCLE: <ObjB 0x7ffc0450df30> [32]
          1 (32 bytes) c --> ROOT CYCLE: <ObjC 0x7ffc0450e050> [32]
             a --> CYCLE BACK TO <ObjA 0x7ffc0450d700> [32]
```

Tools for Profiling Footprint

leaks MyApp.memgraph

NEW

```
Version:         0
Code Type:       X86-64
Parent Process:  bash [2831]

Date/Time:       2018-06-01 00:48:28.712 -0700
Launch Time:     2018-06-01 00:48:25.904 -0700
OS Version:      Mac OS X 10.14 (18A293u)
Report Version:  7
Analysis Tool:   /usr/bin/leaks

Physical footprint:         2252K
Physical footprint (peak):  2252K
----

leaks Report Version: 4.0, multi-line stacks
Process 2853: 787 nodes malloced for 105 KB
Process 2853: 3 leaks for 96 total leaked bytes.

STACK OF 1 INSTANCE OF 'ROOT CYCLE: <ObjA>':
8   libdyld.dylib                   0x7fff54e66ee1 start + 1
7   RetainCycle                        0x1063e21a4 main + 20
6   RetainCycle                        0x1063e28a6 createCycle() + 38
5   RetainCycle                        0x1063e2358 ObjA.__allocating_init() + 56
4   libswiftCore.dylib                 0x10673b924 _swift_allocObject_(swift::TargetHeapMetadata<swift::InProcess> const*, unsigned long, unsigned long) + 2
0
3   libswiftCore.dylib                 0x10673b8a9 swift_slowAlloc + 25
2   libsystem_malloc.dylib          0x7fff5502e04f malloc + 24
1   libsystem_malloc.dylib          0x7fff5502ebcd malloc_zone_malloc + 103
0   libsystem_malloc.dylib          0x7fff5503c833 stack_logging_lite_malloc + 69
====
    3 (96 bytes) ROOT CYCLE: <ObjA 0x7ffc0450d700> [32]
       2 (64 bytes) b --> ROOT CYCLE: <ObjB 0x7ffc0450df30> [32]
          1 (32 bytes) c --> ROOT CYCLE: <ObjC 0x7ffc0450e050> [32]
             a --> CYCLE BACK TO <ObjA 0x7ffc0450d700> [32]
```

Tools for Profiling Footprint

leaks MyApp.memgraph

NEW

```
Version:         0
Code Type:       X86-64
Parent Process:  bash [2831]

Date/Time:       2018-06-01 00:48:28.712 -0700
Launch Time:     2018-06-01 00:48:25.904 -0700
OS Version:      Mac OS X 10.14 (18A293u)
Report Version:  7
Analysis Tool:   /usr/bin/leaks

Physical footprint:         2252K
Physical footprint (peak):  2252K
----

leaks Report Version: 4.0, multi-line stacks
Process 2853: 787 nodes malloced for 105 KB
Process 2853: 3 leaks for 96 total leaked bytes.

STACK OF 1 INSTANCE OF 'ROOT CYCLE: <ObjA>':
8   libdyld.dylib                   0x7fff54e66ee1 start + 1
7   RetainCycle                        0x1063e21a4 main + 20
6   RetainCycle                        0x1063e28a6 createCycle() + 38
5   RetainCycle                        0x1063e2358 ObjA.__allocating_init() + 56
4   libswiftCore.dylib                 0x10673b924 _swift_allocObject_(swift::TargetHeapMetadata<swift::InProcess> const*, unsigned long, unsigned long) + 2
0
3   libswiftCore.dylib                 0x10673b8a9 swift_slowAlloc + 25
2   libsystem_malloc.dylib          0x7fff5502e04f malloc + 24
1   libsystem_malloc.dylib          0x7fff5502ebcd malloc_zone_malloc + 103
0   libsystem_malloc.dylib          0x7fff5503c833 stack_logging_lite_malloc + 69
====
    3 (96 bytes) ROOT CYCLE: <ObjA 0x7ffc0450d700> [32]
       2 (64 bytes) b --> ROOT CYCLE: <ObjB 0x7ffc0450df30> [32]
          1 (32 bytes) c --> ROOT CYCLE: <ObjC 0x7ffc0450e050> [32]
             a --> CYCLE BACK TO <ObjA 0x7ffc0450d700> [32]
```

Tools for Profiling Footprint

heap

NEW

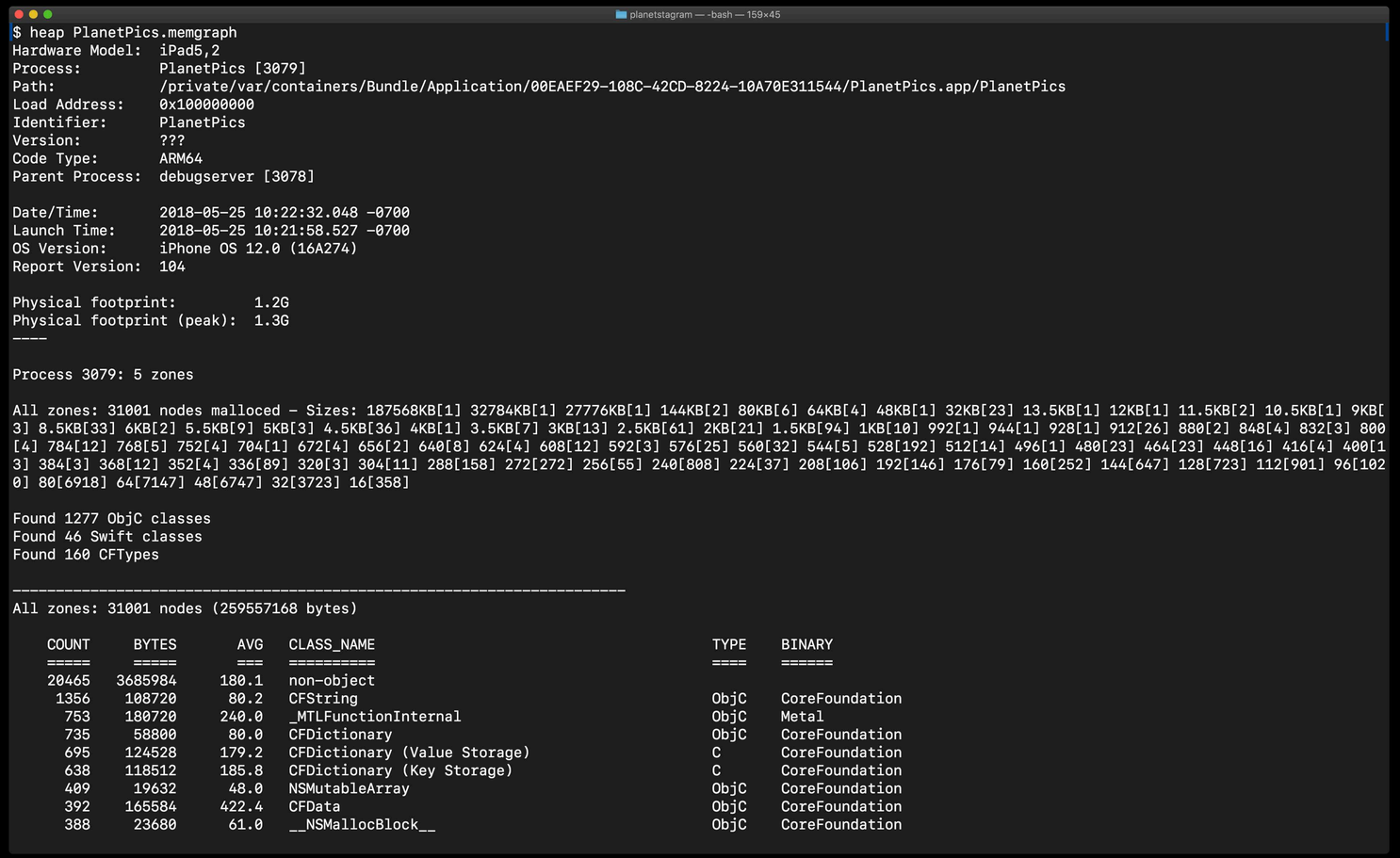
Shows objects allocated on the heap

Useful for identifying large objects in memory and what allocated it

```
heap App.memgraph
heap App.memgraph -sortBySize
heap App.memgraph -addresses all | <classes-pattern>
```

Tools for Profiling Footprint

heap App.memgraph



Tools for Profiling Footprint

heap App.memgraph

```
Date/Time:      2018-05-25 10:22:12.045 -0700
Launch Time:    2018-05-25 10:21:58.527 -0700
OS Version:     iPhone OS 12.0 (16A274)
Report Version: 104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Process 3079: 5 zones

All zones: 31001 nodes malloced - Sizes: 187568KB[1] 32784KB[1] 27776KB[1] 144KB[2] 80KB[6] 64KB[4] 48KB[1] 32KB[23] 13.5KB[1] 12KB[1] 11.5KB[2] 10.5KB[1] 9KB[3] 8.5KB[33] 6KB[2] 5.5KB[9] 5KB[3] 4.5KB[36] 4KB[1] 3.5KB[7] 3KB[13] 2.5KB[61] 2KB[21] 1.5KB[94] 1KB[10] 992[1] 944[1] 928[1] 912[26] 880[2] 848[4] 832[3] 800[4] 784[12] 768[5] 752[4] 704[1] 672[4] 656[2] 640[8] 624[4] 608[12] 592[3] 576[25] 560[32] 544[5] 528[192] 512[14] 496[1] 480[23] 464[23] 448[16] 416[4] 400[13] 384[3] 368[12] 352[4] 336[89] 320[3] 304[11] 288[158] 272[272] 256[55] 240[808] 224[37] 208[106] 192[146] 176[79] 160[252] 144[647] 128[723] 112[901] 96[1020] 80[6918] 64[7147] 48[6747] 32[3723] 16[358]

Found 1277 ObjC classes
Found 46 Swift classes
Found 160 CFTypes

------------------------------------------------------------------------
All zones: 31001 nodes (259557168 bytes)

    COUNT      BYTES       AVG   CLASS_NAME                                TYPE    BINARY
    =====      =====       ===   ==========                                ====    ======
    20465    3685984     180.1   non-object
     1356     108720      80.2   CFString                                  ObjC    CoreFoundation
      753     180720     240.0   _MTLFunctionInternal                      ObjC    Metal
      735      58800      80.0   CFDictionary                              ObjC    CoreFoundation
      695     124528     179.2   CFDictionary (Value Storage)              C       CoreFoundation
      638     118512     185.8   CFDictionary (Key Storage)                C       CoreFoundation
      409      19632      48.0   NSMutableArray                            ObjC    CoreFoundation
      392     165584     422.4   CFData                                    ObjC    CoreFoundation
      388      23680      61.0   __NSMallocBlock__                         ObjC    CoreFoundation
```

Tools for Profiling Footprint

heap App.memgraph

```
Date/Time:      2018-05-25 10:22:12.045 -0700
Launch Time:    2018-05-25 10:21:58.527 -0700
OS Version:     iPhone OS 12.0 (16A274)
Report Version: 104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Process 3079: 5 zones

All zones: 31001 nodes malloced - Sizes: 187568KB[1] 32784KB[1] 27776KB[1] 144KB[2] 80KB[6] 64KB[4] 48KB[1] 32KB[23] 13.5KB[1] 12KB[1] 11.5KB[2] 10.5KB[1] 9KB[3] 8.5KB[33] 6KB[2] 5.5KB[9] 5KB[3] 4.5KB[36] 4KB[1] 3.5KB[7] 3KB[13] 2.5KB[61] 2KB[21] 1.5KB[94] 1KB[10] 992[1] 944[1] 928[1] 912[26] 880[2] 848[4] 832[3] 800[4] 784[12] 768[5] 752[4] 704[1] 672[4] 656[2] 640[8] 624[4] 608[12] 592[3] 576[25] 560[32] 544[5] 528[192] 512[14] 496[1] 480[23] 464[23] 448[16] 416[4] 400[13] 384[3] 368[12] 352[4] 336[89] 320[3] 304[11] 288[158] 272[272] 256[55] 240[808] 224[37] 208[106] 192[146] 176[79] 160[252] 144[647] 128[723] 112[901] 96[1020] 80[6918] 64[7147] 48[6747] 32[3723] 16[358]

Found 1277 ObjC classes
Found 46 Swift classes
Found 160 CFTypes

------------------------------------------------------------------------
All zones: 31001 nodes (259557168 bytes)

  COUNT      BYTES       AVG   CLASS_NAME                                TYPE    BINARY
  =====      =====       ===   ==========                                ====    ======
  20465    3685984     180.1   non-object
   1356     108720      80.2   CFString                                  ObjC    CoreFoundation
    753     180720     240.0   _MTLFunctionInternal                      ObjC    Metal
    735      58800      80.0   CFDictionary                              ObjC    CoreFoundation
    695     124528     179.2   CFDictionary (Value Storage)              C       CoreFoundation
    638     118512     185.8   CFDictionary (Key Storage)                C       CoreFoundation
    409      19632      48.0   NSMutableArray                            ObjC    CoreFoundation
    392     165584     422.4   CFData                                    ObjC    CoreFoundation
    388      23680      61.0   __NSMallocBlock__                         ObjC    CoreFoundation
```

Tools for Profiling Footprint

heap App.memgraph

```
Date/Time:      2018-05-25 10:22:12.045 -0700
Launch Time:    2018-05-25 10:21:58.527 -0700
OS Version:     iPhone OS 12.0 (16A274)
Report Version: 104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Process 3079: 5 zones

All zones: 31001 nodes malloced - Sizes: 187568KB[1] 32784KB[1] 27776KB[1] 144KB[2] 80KB[6] 64KB[4] 48KB[1] 32KB[23] 13.5KB[1] 12KB[1] 11.5KB[2] 10.5KB[1] 9KB[3] 8.5KB[33] 6KB[2] 5.5KB[9] 5KB[3] 4.5KB[36] 4KB[1] 3.5KB[7] 3KB[13] 2.5KB[61] 2KB[21] 1.5KB[94] 1KB[10] 992[1] 944[1] 928[1] 912[26] 880[2] 848[4] 832[3] 800[4] 784[12] 768[5] 752[4] 704[1] 672[4] 656[2] 640[8] 624[4] 608[12] 592[3] 576[25] 560[32] 544[5] 528[192] 512[14] 496[1] 480[23] 464[23] 448[16] 416[4] 400[13] 384[3] 368[12] 352[4] 336[89] 320[3] 304[11] 288[158] 272[272] 256[55] 240[808] 224[37] 208[106] 192[146] 176[79] 160[252] 144[647] 128[723] 112[901] 96[1020] 80[6918] 64[7147] 48[6747] 32[3723] 16[358]

Found 1277 ObjC classes
Found 46 Swift classes
Found 160 CFTypes

------------------------------------------------------------------------
All zones: 31001 nodes (259557168 bytes)

    COUNT      BYTES        AVG   CLASS_NAME                                TYPE    BINARY
    =====      =====        ===   ==========                                ====    ======
    20465    3685984      180.1   non-object
     1356     108720       80.2   CFString                                  ObjC    CoreFoundation
      753     180720      240.0   _MTLFunctionInternal                      ObjC    Metal
      735      58800       80.0   CFDictionary                              ObjC    CoreFoundation
      695     124528      179.2   CFDictionary (Value Storage)              C       CoreFoundation
      638     118512      185.8   CFDictionary (Key Storage)                C       CoreFoundation
      409      19632       48.0   NSMutableArray                            ObjC    CoreFoundation
      392     165584      422.4   CFData                                    ObjC    CoreFoundation
      388      23680       61.0   __NSMallocBlock__                         ObjC    CoreFoundation
```

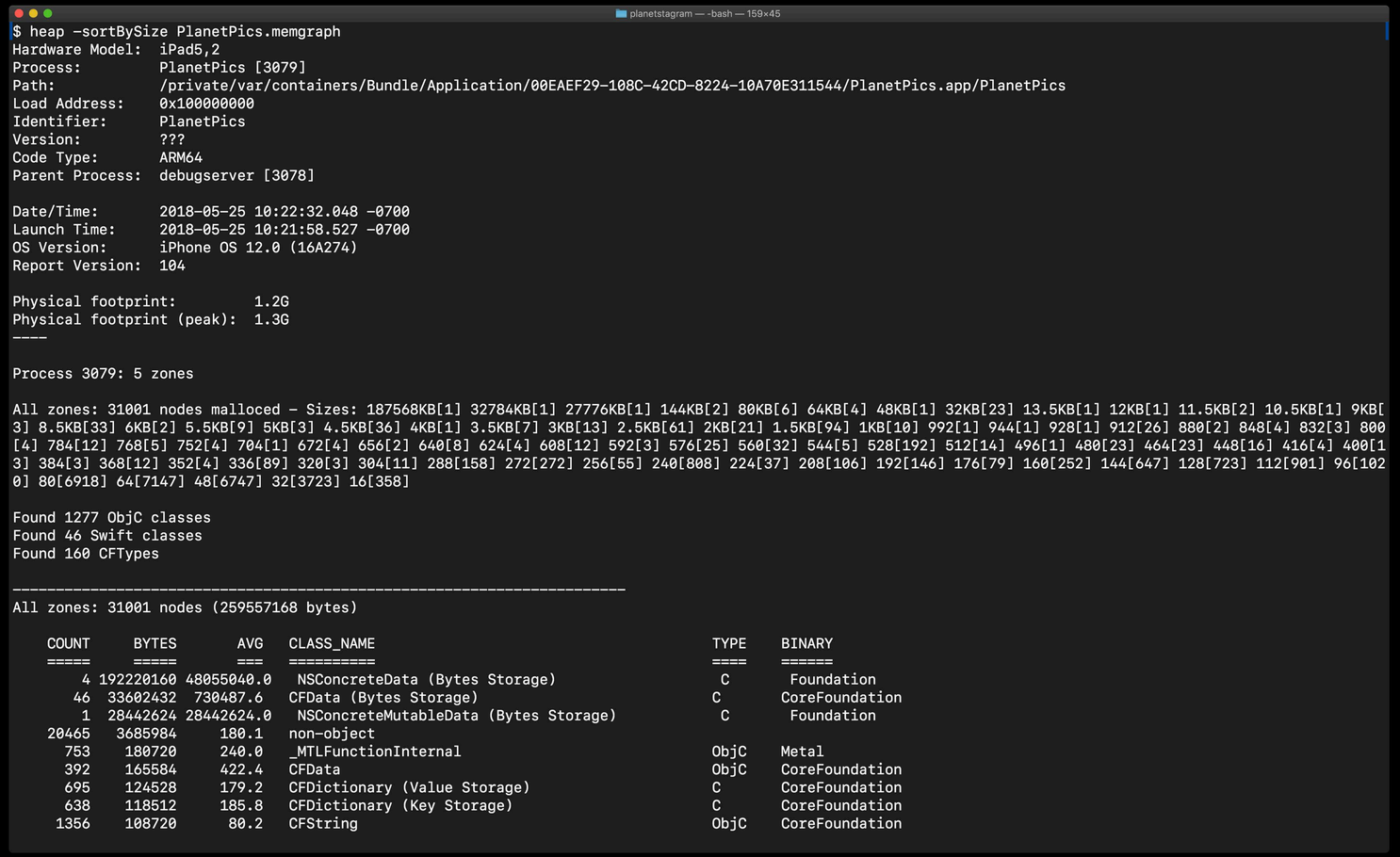
Tools for Profiling Footprint

heap App.memgraph

```
Date/Time:       2018-05-25 10:22:02.043 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Process 3079: 5 zones

All zones: 31001 nodes malloced - Sizes: 187568KB[1] 32784KB[1] 27776KB[1] 144KB[2] 80KB[6] 64KB[4] 48KB[1] 32KB[23] 13.5KB[1] 12KB[1] 11.5KB[2] 10.5KB[1] 9KB[3] 8.5KB[33] 6KB[2] 5.5KB[9] 5KB[3] 4.5KB[36] 4KB[1] 3.5KB[7] 3KB[13] 2.5KB[61] 2KB[21] 1.5KB[94] 1KB[10] 992[1] 944[1] 928[1] 912[26] 880[2] 848[4] 832[3] 800[4] 784[12] 768[5] 752[4] 704[1] 672[4] 656[2] 640[8] 624[4] 608[12] 592[3] 576[25] 560[32] 544[5] 528[192] 512[14] 496[1] 480[23] 464[23] 448[16] 416[4] 400[13] 384[3] 368[12] 352[4] 336[89] 320[3] 304[11] 288[158] 272[272] 256[55] 240[808] 224[37] 208[106] 192[146] 176[79] 160[252] 144[647] 128[723] 112[901] 96[1020] 80[6918] 64[7147] 48[6747] 32[3723] 16[358]

Found 1277 ObjC classes
Found 46 Swift classes
Found 160 CFTypes

------------------------------------------------------------------------
All zones: 31001 nodes (259557168 bytes)

    COUNT      BYTES       AVG   CLASS_NAME                                TYPE    BINARY
    =====      =====       ===   ==========                                ====    ======
    20465    3685984     180.1   non-object
     1356     108720      80.2   CFString                                  ObjC    CoreFoundation
      753     180720     240.0   _MTLFunctionInternal                      ObjC    Metal
      735      58800      80.0   CFDictionary                              ObjC    CoreFoundation
      695     124528     179.2   CFDictionary (Value Storage)              C       CoreFoundation
      638     118512     185.8   CFDictionary (Key Storage)                C       CoreFoundation
      409      19632      48.0   NSMutableArray                            ObjC    CoreFoundation
      392     165584     422.4   CFData                                    ObjC    CoreFoundation
      388      23680      61.0   __NSMallocBlock__                         ObjC    CoreFoundation
```

Tools for Profiling Footprint

heap --sortBySize App.memgraph



Tools for Profiling Footprint

heap --sortBySize App.memgraph

```
Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Process 3079: 5 zones

All zones: 31001 nodes malloced - Sizes: 187568KB[1] 32784KB[1] 27776KB[1] 144KB[2] 80KB[6] 64KB[4] 48KB[1] 32KB[23] 13.5KB[1] 12KB[1] 11.5KB[2] 10.5KB[1] 9KB[3] 8.5KB[33] 6KB[2] 5.5KB[9] 5KB[3] 4.5KB[36] 4KB[1] 3.5KB[7] 3KB[13] 2.5KB[61] 2KB[21] 1.5KB[94] 1KB[10] 992[1] 944[1] 928[1] 912[26] 880[2] 848[4] 832[3] 800[4] 784[12] 768[5] 752[4] 704[1] 672[4] 656[2] 640[8] 624[4] 608[12] 592[3] 576[25] 560[32] 544[5] 528[192] 512[14] 496[1] 480[23] 464[23] 448[16] 416[4] 400[13] 384[3] 368[12] 352[4] 336[89] 320[3] 304[11] 288[158] 272[272] 256[55] 240[808] 224[37] 208[106] 192[146] 176[79] 160[252] 144[647] 128[723] 112[901] 96[1020] 80[6918] 64[7147] 48[6747] 32[3723] 16[358]

Found 1277 ObjC classes
Found 46 Swift classes
Found 160 CFTypes

------------------------------------------------------------------------
All zones: 31001 nodes (259557168 bytes)

    COUNT      BYTES       AVG   CLASS_NAME                                TYPE    BINARY
    =====      =====       ===   ==========                                ====    ======
        4  192220160  48055040.0   NSConcreteData (Bytes Storage)          C       Foundation
       46   33602432    730487.6   CFData (Bytes Storage)                  C       CoreFoundation
        1   28442624  28442624.0   NSConcreteMutableData (Bytes Storage)   C       Foundation
    20465    3685984       180.1   non-object
      753     180720       240.0   _MTLFunctionInternal                    ObjC    Metal
      392     165584       422.4   CFData                                  ObjC    CoreFoundation
      695     124528       179.2   CFDictionary (Value Storage)            C       CoreFoundation
      638     118512       185.8   CFDictionary (Key Storage)              C       CoreFoundation
     1356     108720        80.2   CFString                                ObjC    CoreFoundation
```

Tools for Profiling Footprint

heap App.memgraph -addresses all | <classes-pattern>

planetstagram — -bash — 159×28

```
$ heap -addresses NSConcreteData PlanetPics.memgraph
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Active blocks in Zone (null) that match pattern 'NSConcreteData':
0x10169bfd0: NSConcreteData (48 bytes)
0x10169ca80: NSConcreteData (48 bytes)
0x1016e6ff0: NSConcreteData (48 bytes)
0x1017cb340: NSConcreteData (48 bytes)
0x1017f52e0: NSConcreteData (48 bytes)
0x105534100: NSConcreteData (48 bytes)
0x105552a60: NSConcreteData (48 bytes)
0x1055664e0: NSConcreteData (48 bytes)
```

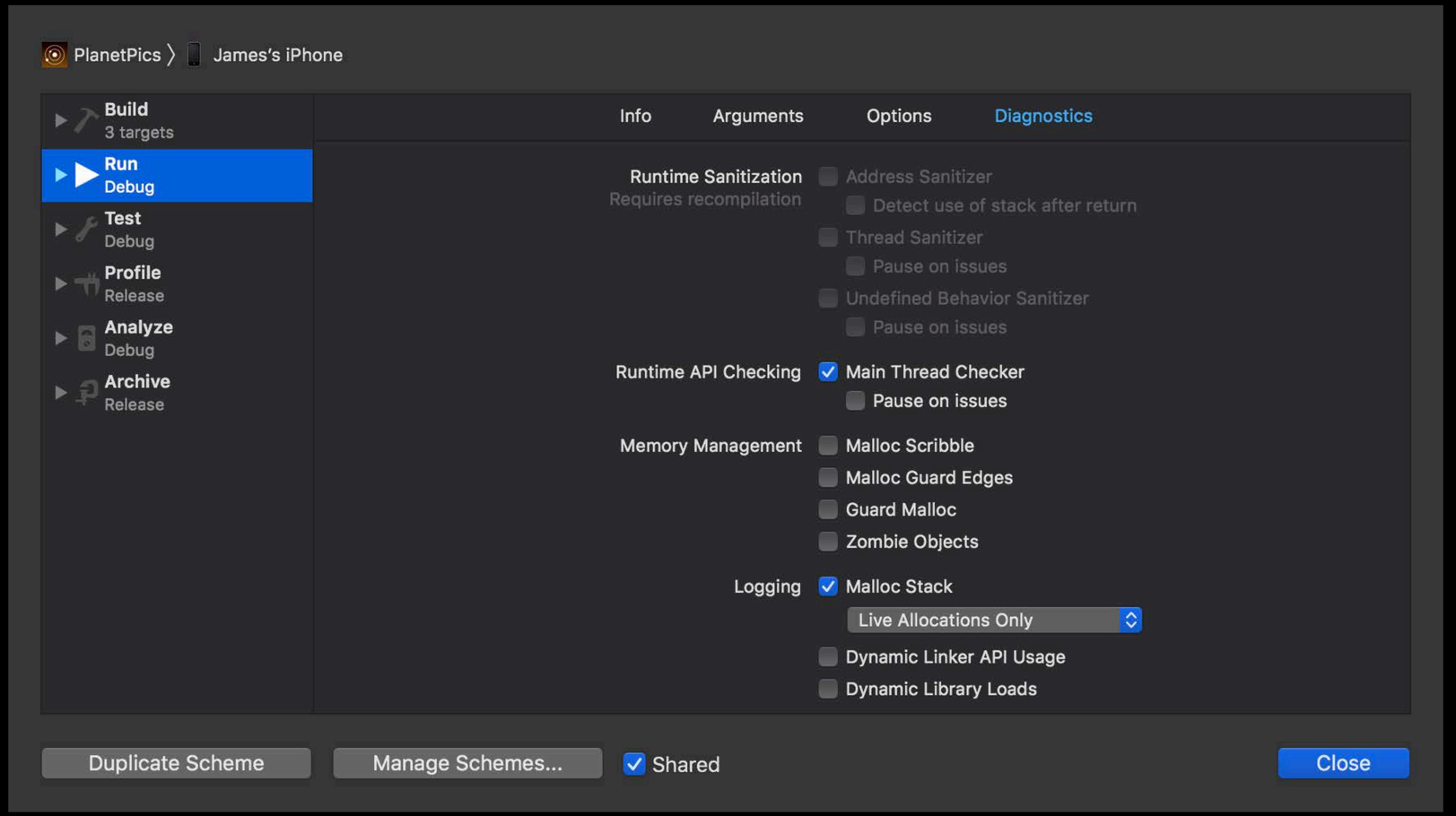
Tools for Profiling Footprint

heap App.memgraph -addresses all | <classes-pattern>

```
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/Pl
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Active blocks in Zone (null) that match pattern 'NSConcreteData':
0x10169bfd0: NSConcreteData (48 bytes)
0x10169ca80: NSConcreteData (48 bytes)
0x1016e6ff0: NSConcreteData (48 bytes)
0x1017cb340: NSConcreteData (48 bytes)
0x1017f52e0: NSConcreteData (48 bytes)
0x105534100: NSConcreteData (48 bytes)
0x105552a60: NSConcreteData (48 bytes)
0x1055664e0: NSConcreteData (48 bytes)
```

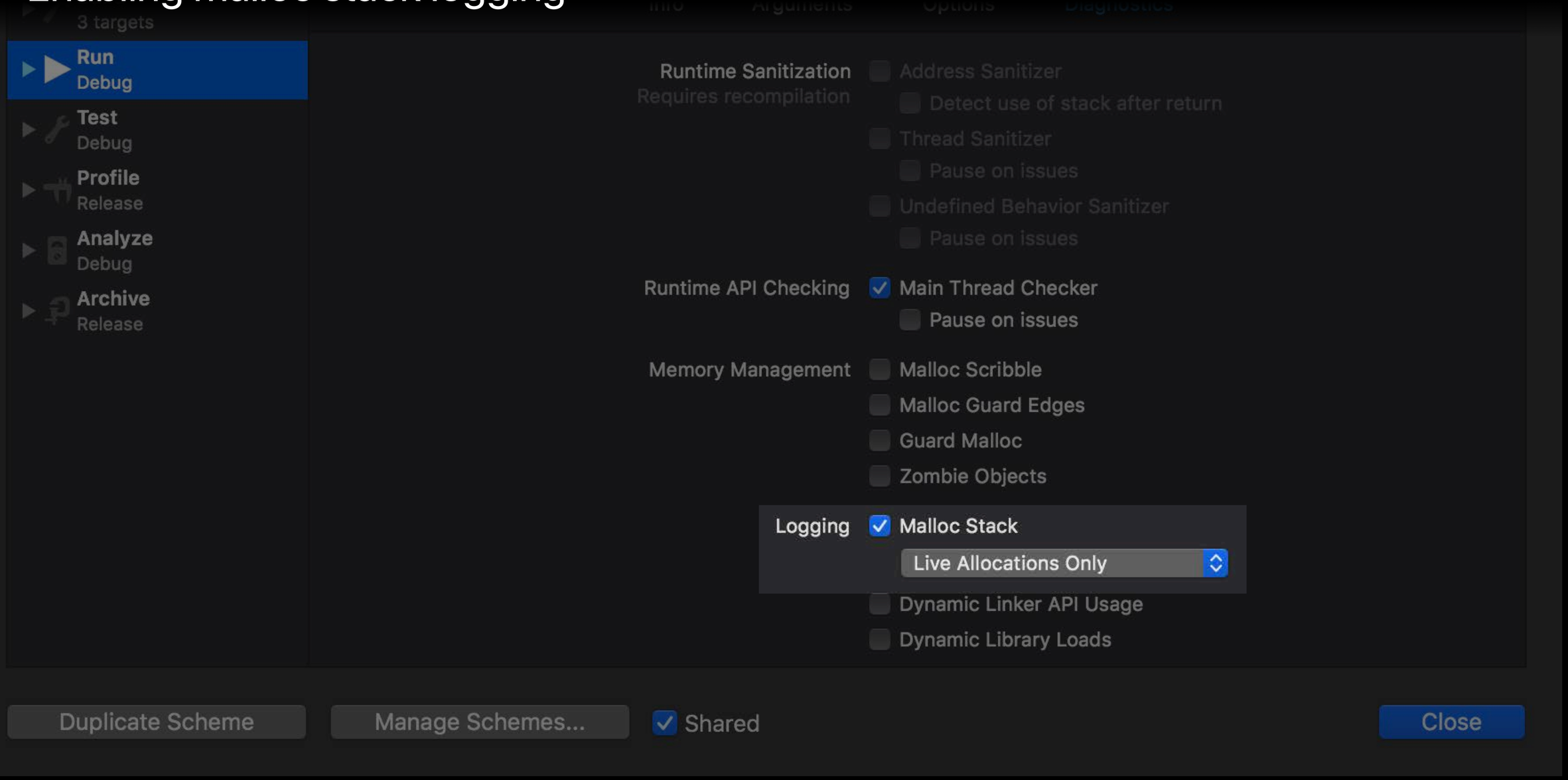
Tools for Profiling Footprint

Enabling malloc stack logging



Tools for Profiling Footprint

Enabling malloc stack logging



Tools for Profiling Footprint

malloc_history

NEW

Shows backtraces for malloc and anonymous VM region allocations

```
malloc_history App.memgraph [address]
```

Tools for Profiling Footprint

malloc_history <memgraph> <address>

```
planetstagram — -bash — 159×37
$ malloc_history -callTree PlanetPics.memgraph 0x1016e6ff0
malloc_history Report Version:  2.0
Hardware Model:  iPad5,2
Process:         PlanetPics [3079]
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Call graph:
    1 (48 bytes) start_wqthread  (in libsystem_pthread.dylib) + 4  [0x101019218]
      1 (48 bytes) _pthread_wqthread  (in libsystem_pthread.dylib) + 472  [0x1010193fc]
        1 (48 bytes) _dispatch_worker_thread2  (in libdispatch.dylib) + 144  [0x100fa5678]
          1 (48 bytes) _dispatch_root_queue_drain  (in libdispatch.dylib) + 724  [0x100fa4dfc]
            1 (48 bytes) _dispatch_client_callout  (in libdispatch.dylib) + 16  [0x100f8d3a4]
              1 (48 bytes) _dispatch_call_block_and_release  (in libdispatch.dylib) + 24  [0x100f8d3e4]
                1 (48 bytes) thunk for @escaping @callee_guaranteed () -> ()  (in PlanetPics) + 52  [0x10001597c]
                  1 (48 bytes) closure #1 in DataViewController.applyFilter(filter:)  (in PlanetPics) + 2156  [0x100015280]
                    1 (48 bytes) protocol witness for Filter.apply(image:) in conformance NoirFilter  (in PlanetPics) + 28  [0x100007508]
                      1 (48 bytes) NoirFilter.apply(image:)  (in PlanetPics) + 620  [0x100006784]
                        1 (48 bytes) @nonobjc CIContext.__allocating_init(cgContext:options:)  (in PlanetPics) + 200  [0x100006fc8]
                          1 (48 bytes) +[CIContext contextWithCGContext:options:]  (in CoreImage) + 60  [0x19debbd08]
                            1 (48 bytes) -[CIContext initWithCGContext:options:]  (in CoreImage) + 324  [0x19debbe60]
                              1 (48 bytes) -[CIContext initWithOptions:]  (in CoreImage) + 596  [0x19debbbb0]
                                1 (48 bytes) +[CIContext(Internal) internalContextWithMTLDevice:options:]  (in CoreImage) + 876  [0x19dec0554]
                                  1 (48 bytes) CI::MetalContext::MetalContext(void const*, void const*, char const*, CGColorSpace*, CGColorSpace*, CI::PixelFor
```

Tools for Profiling Footprint

malloc_history <memgraph> <address>

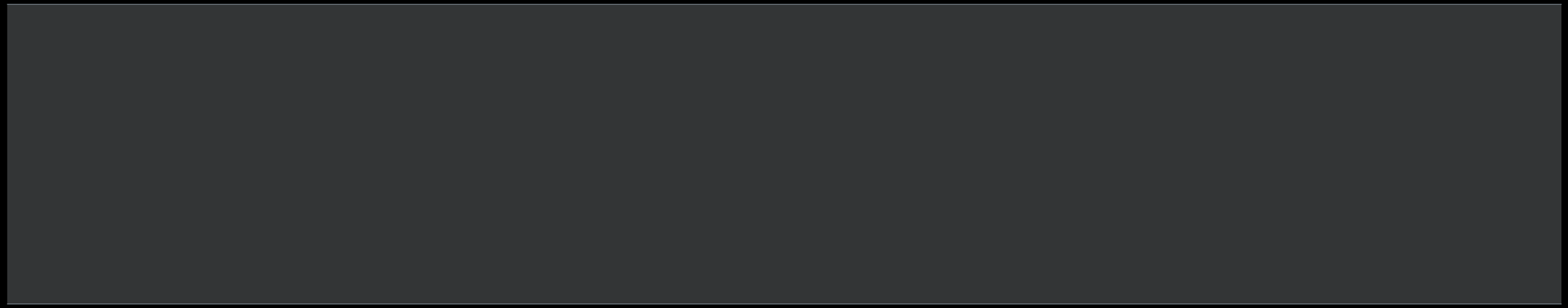
```
Path:            /private/var/containers/Bundle/Application/00EAEF29-108C-42CD-8224-10A70E311544/PlanetPics.app/PlanetPics
Load Address:    0x100000000
Identifier:      PlanetPics
Version:         ???
Code Type:       ARM64
Parent Process:  debugserver [3078]

Date/Time:       2018-05-25 10:22:32.048 -0700
Launch Time:     2018-05-25 10:21:58.527 -0700
OS Version:      iPhone OS 12.0 (16A274)
Report Version:  104

Physical footprint:         1.2G
Physical footprint (peak):  1.3G
----

Call graph:
    1 (48 bytes) start_wqthread  (in libsystem_pthread.dylib) + 4  [0x101019218]
      1 (48 bytes) _pthread_wqthread  (in libsystem_pthread.dylib) + 472  [0x1010193fc]
        1 (48 bytes) _dispatch_worker_thread2  (in libdispatch.dylib) + 144  [0x100fa5678]
          1 (48 bytes) _dispatch_root_queue_drain  (in libdispatch.dylib) + 724  [0x100fa4dfc]
            1 (48 bytes) _dispatch_client_callout  (in libdispatch.dylib) + 16  [0x100f8d3a4]
              1 (48 bytes) _dispatch_call_block_and_release  (in libdispatch.dylib) + 24  [0x100f8d3e4]
                1 (48 bytes) thunk for @escaping @callee_guaranteed () -> ()  (in PlanetPics) + 52  [0x10001597c]
                  1 (48 bytes) closure #1 in DataViewController.applyFilter(filter:)  (in PlanetPics) + 2156  [0x100015280]
                    1 (48 bytes) protocol witness for Filter.apply(image:) in conformance NoirFilter  (in PlanetPics) + 28  [0x100007508]
                      1 (48 bytes) NoirFilter.apply(image:)  (in PlanetPics) + 620  [0x100006784]
                        1 (48 bytes) @nonobjc CIContext.__allocating_init(cgContext:options:)  (in PlanetPics) + 200  [0x100006fc8]
                          1 (48 bytes) +[CIContext contextWithCGContext:options:]  (in CoreImage) + 60  [0x19debbd08]
                            1 (48 bytes) -[CIContext initWithCGContext:options:]  (in CoreImage) + 324  [0x19debbe60]
                              1 (48 bytes) -[CIContext initWithOptions:]  (in CoreImage) + 596  [0x19debbbb0]
                                1 (48 bytes) +[CIContext(Internal) internalContextWithMTLDevice:options:]  (in CoreImage) + 876  [0x19dec0554]
                                  1 (48 bytes) CI::MetalContext::MetalContext(void const*, void const*, char const*, CGColorSpace*, CGColorSpace*, CI::
```

Tools for Profiling Footprint

Which tool to pick?

Tools for Profiling Footprint

Which tool to pick?

Creation	Reference	Size
		

Tools for Profiling Footprint

Which tool to pick?

Creation	Reference	Size
malloc_history		

Tools for Profiling Footprint

Which tool to pick?

Creation	Reference	Size
malloc_history	leaks	

Tools for Profiling Footprint

Which tool to pick?

Creation	Reference	Size
malloc_history	leaks	vmmap, heap

Images

Kyle Howarth, Apple/Software Engineer

Memory use is related to the dimensions of the image, not the file size.

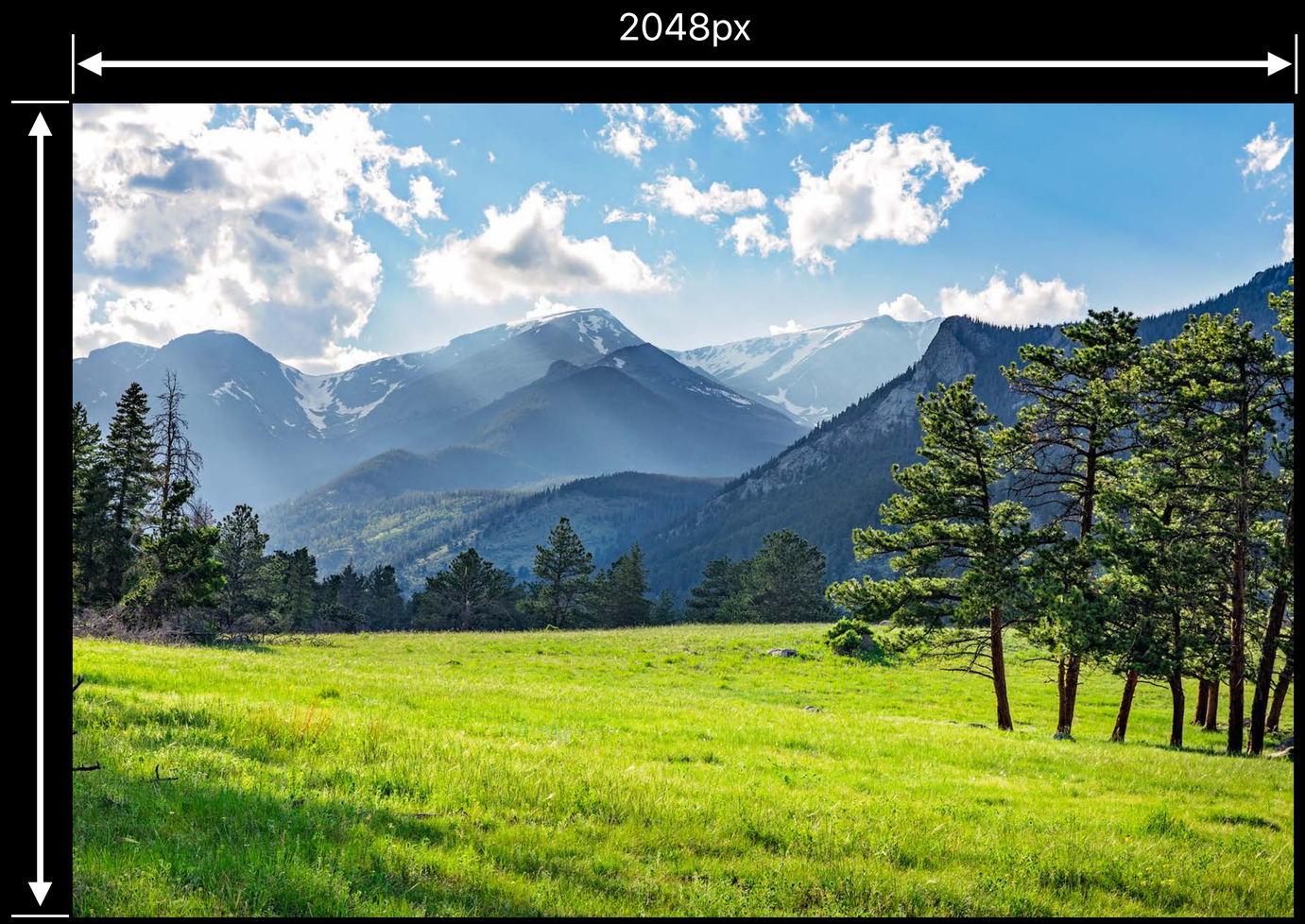
Images

590Kb file size

Images

590KB file size

Images



590KB file size

10MB

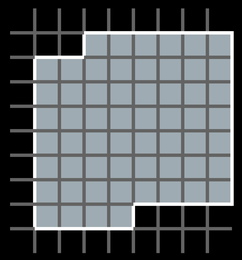
2048 pixels x 1536 pixels x 4 bytes per pixel

Images

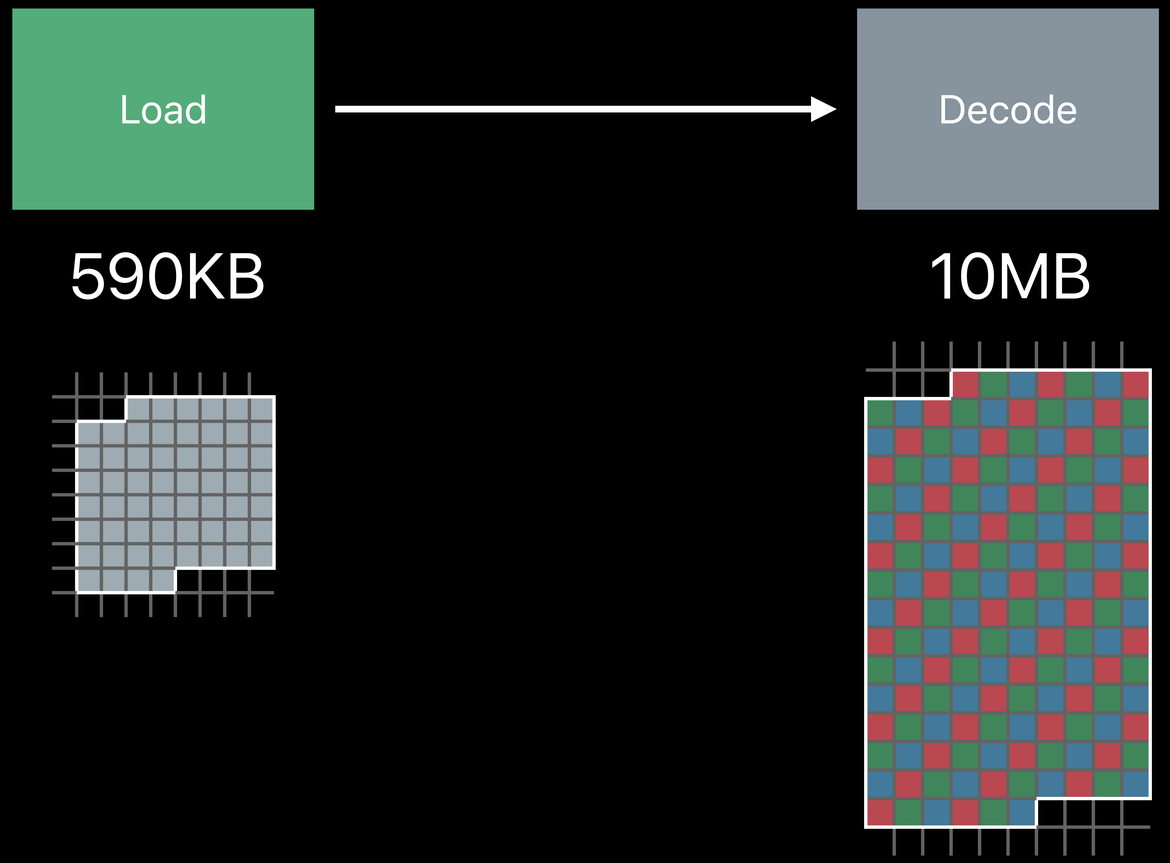
Images

Load

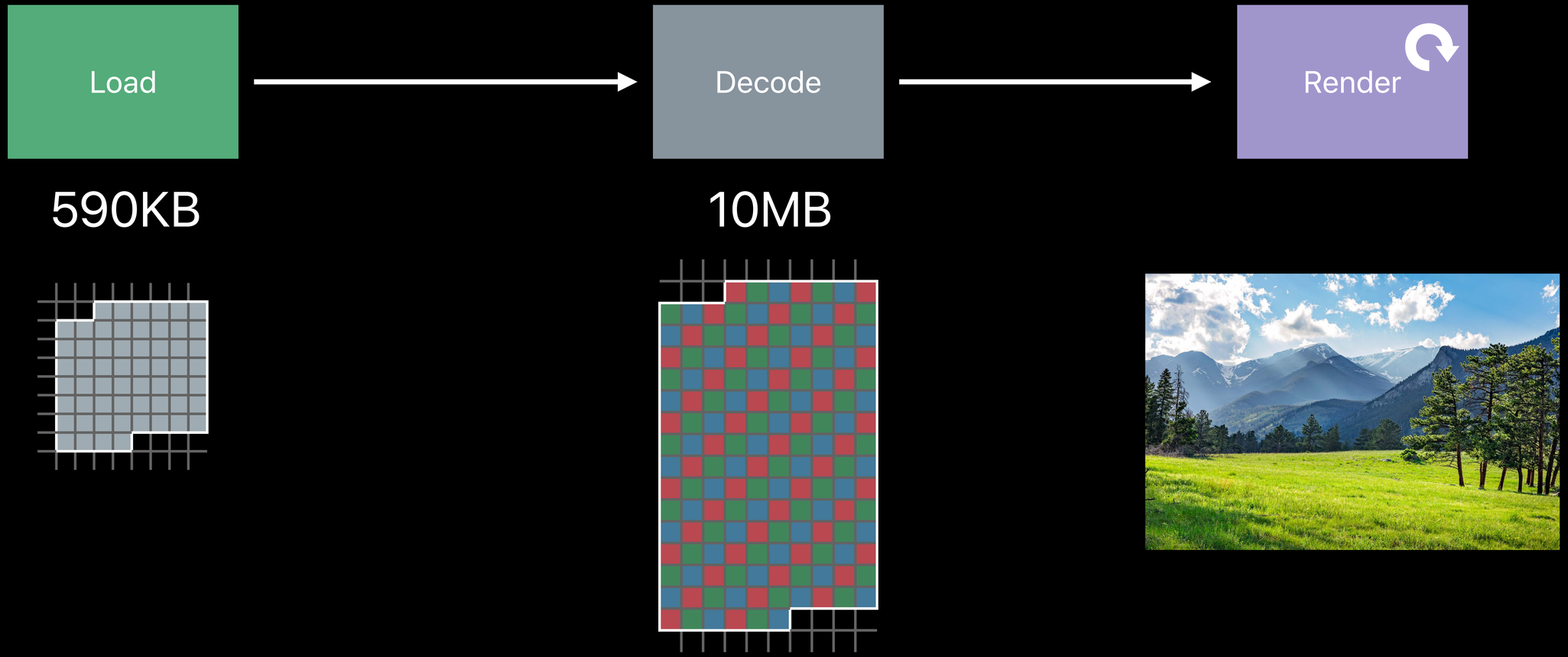
590Kb



Images



Images

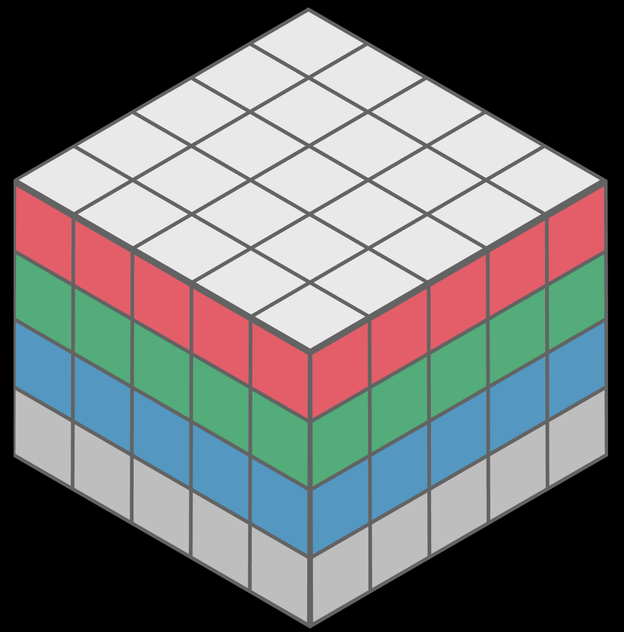


Images

Image-rendering formats

SRGB Format

- Four bytes per pixel
- Full color images

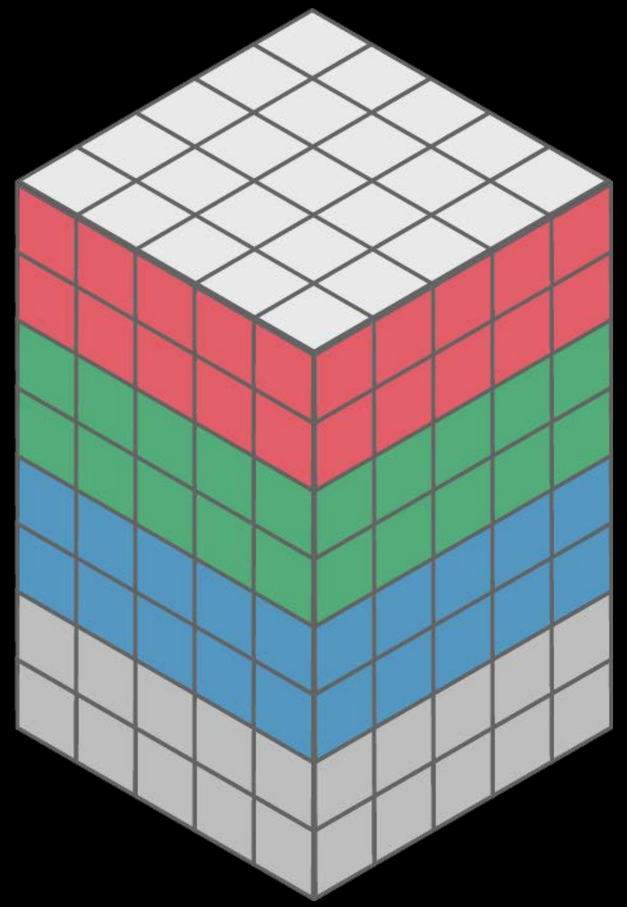


Images

Image-rendering formats

Wide format

- Eight bytes per pixel
- Super accurate colors
- Only useful with wide color displays
- Wide color capture cameras
  - iPhone 7, iPhone 8, iPhone X, iPad Pro 10.5”, iPad Pro 13” (2nd generation)

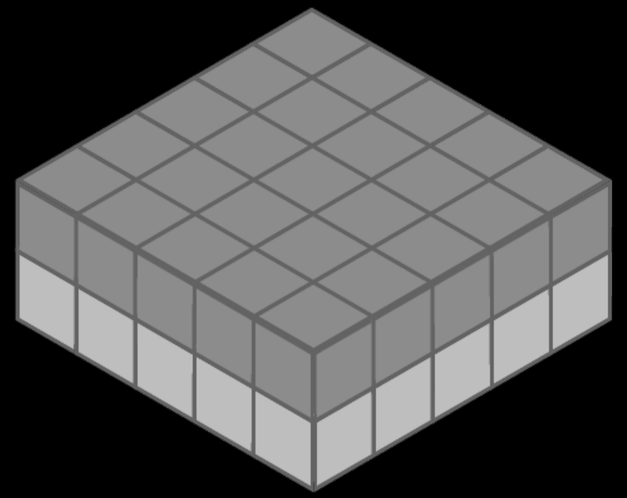


Images

Image rendering formats

Luminance and alpha 8 format

- Two bytes per pixel
- Single-color images and alpha
- Metal shaders

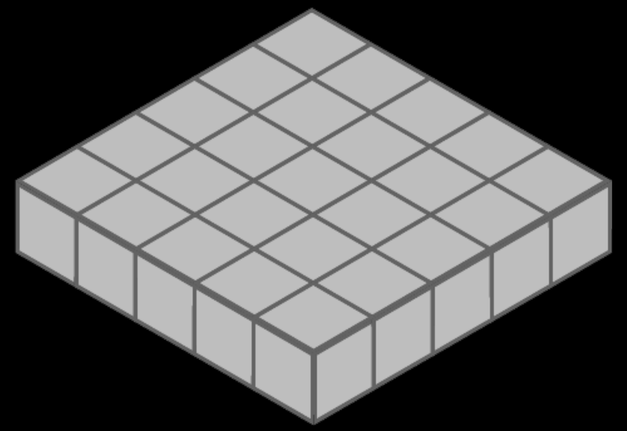


Images

Image rendering formats

Alpha 8 Format

- One byte per pixel
- Useful for monochrome images
  - Masks
  - Emoji-free text
- 75 percent smaller than SRGB



Images

Image rendering formats

Optimized formats reduce memory use

- One byte: Alpha 8
- Two bytes: luminance and alpha 8
- Four bytes: SRGB
- Eight bytes: wide format

How do I pick the right format?

Images

Picking the right format

Don’t pick the format, let the format pick you

Stop using `UIGraphicsBeginImageContextWithOptions`

- Four bytes per pixel format

Start or keep using `UIGraphicsImageRenderer`

- Introduced in iOS 10
- Automatically picks best graphics format in iOS 12

```swift
// Circle via UIGraphicsImageContext

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
UIGraphicsBeginImageContextWithOptions(bounds.size, false, 0)

// Drawing Code
UIColor.black.setFill()
let path = UIBezierPath(roundedRect: bounds,
                        byRoundingCorners: UIRectCorner.allCorners,
                              cornerRadii: CGSize(width: 20, height: 20))
path.addClip()
UIRectFill(bounds)

let image = UIGraphicsGetImageFromCurrentImageContext()
UIGraphicsEndImageContext()
```

```
// Circle via UIGraphicsImageContext

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
UIGraphicsBeginImageContextWithOptions(bounds.size, false, 0)

// Drawing Code
UIColor.black.setFill()
let path = UIBezierPath(roundedRect: bounds,
                        byRoundingCorners: UIRectCorner.allCorners,
                              cornerRadii: CGSize(width: 20, height: 20))
path.addClip()
UIRectFill(bounds)

let image = UIGraphicsGetImageFromCurrentImageContext()
UIGraphicsEndImageContext()
```

```
// Circle via UIGraphicsImageRenderer

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
let renderer = UIGraphicsImageRenderer(size: bounds.size)
let image = renderer.image { context in
    // Drawing Code
    UIColor.black.setFill()
    let path = UIBezierPath(roundedRect: bounds,
                            byRoundingCorners: UIRectCorner.allCorners,
                                  cornerRadii: CGSize(width: 20, height: 20))
    path.addClip()
    UIRectFill(bounds)
}
```

```
// Circle via UIGraphicsImageRenderer

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
let renderer = UIGraphicsImageRenderer(size: bounds.size)
let image = renderer.image { context in
    // Drawing Code
    UIColor.black.setFill()
    let path = UIBezierPath(roundedRect: bounds,
                            byRoundingCorners: UIRectCorner.allCorners,
                                  cornerRadii: CGSize(width: 20, height: 20))
    path.addClip()
    UIRectFill(bounds)
}
```

```swift
// Circle via UIGraphicsImageRenderer

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
let renderer = UIGraphicsImageRenderer(size: bounds.size)
let image = renderer.image { context in
    // Drawing Code
    UIColor.black.setFill()
    let path = UIBezierPath(roundedRect: bounds,
                            byRoundingCorners: UIRectCorner.allCorners,
                                  cornerRadii: CGSize(width: 20, height: 20))
    path.addClip()
    UIRectFill(bounds)
}

// Make circle render blue, but stay at 1 byte-per-pixel image
let imageView = UIImageView(image: image)
imageView.tintColor = .blue
```

```swift
// Circle via UIGraphicsImageRenderer

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
let renderer = UIGraphicsImageRenderer(size: bounds.size)
let image = renderer.image { context in
    // Drawing Code
    UIColor.black.setFill()
    let path = UIBezierPath(roundedRect: bounds,
                            byRoundingCorners: UIRectCorner.allCorners,
                                  cornerRadii: CGSize(width: 20, height: 20))
    path.addClip()
    UIRectFill(bounds)
}

// Make circle render blue, but stay at 1 byte-per-pixel image
let imageView = UIImageView(image: image)
imageView.tintColor = .blue
```

```swift
// Circle via UIGraphicsImageRenderer

let bounds = CGRect(x: 0, y: 0, width:300, height: 100)
let renderer = UIGraphicsImageRenderer(size: bounds.size)
let image = renderer.image { context in
    // Drawing Code
    UIColor.black.setFill()
    let path = UIBezierPath(roundedRect: bounds,
                            byRoundingCorners: UIRectCorner.allCorners,
                                  cornerRadii: CGSize(width: 20, height: 20))
    path.addClip()
    UIRectFill(bounds)
}

// Make circle render blue, but stay at 1 byte-per-pixel image
let imageView = UIImageView(image: image)
imageView.tintColor = .blue
```

Images

Downsampling

`UIImage` is expensive for sizing and to resizing

- Will decompress original image into memory
- Internal coordinate space transforms are expensive

`ImageIO` can read image sizes and metadata information without dirtying memory

`ImageIO` can resize images at cost of resized image only

```
// Image size with UIImage

import UIKit

// Getting image size
let filePath = "/path/to/image.jpg"
let image = UIImage(contentsOfFile: filePath)
let imageSize = image.size

// Resizing image
let scale = 0.2
let size = CGSize(image.size.width * scale, image.size.height * scale)
let renderer = UIGraphicsImageRenderer(size: size)
let resizedImage = renderer.image { context in
    image.draw(in: CGRect(x: 0, y: 0, width: size.width, height: size.height))
}
```

```
// Image size with UIImage

import UIKit

// Getting image size
let filePath = "/path/to/image.jpg"
let image = UIImage(contentsOfFile: filePath)
let imageSize = image.size

// Resizing image
let scale = 0.2
let size = CGSize(image.size.width * scale, image.size.height * scale)
let renderer = UIGraphicsImageRenderer(size: size)
let resizedImage = renderer.image { context in
    image.draw(in: CGRect(x: 0, y: 0, width: size.width, height: size.height))
}
```

```swift
// Image size with UIImage

import UIKit

// Getting image size
let filePath = "/path/to/image.jpg"
let image = UIImage(contentsOfFile: filePath)
let imageSize = image.size

// Resizing image
let scale = 0.2
let size = CGSize(image.size.width * scale, image.size.height * scale)
let renderer = UIGraphicsImageRenderer(size: size)
let resizedImage = renderer.image { context in
    image.draw(in: CGRect(x: 0, y: 0, width: size.width, height: size.height))
}
```

```
// Image size with ImageIO

import ImageIO

let filePath = "/path/to/image.jpg"
let url = NSURL(fileURLWithPath: path)

let imageSource = CGImageSourceCreateWithURL(url, nil)
let properties = CGImageSourceCopyPropertiesAtIndex(imageSource, 0, nil)
let options: [NSString: Any] = [
    kCGImageSourceThumbnailMaxPixelSize: 100,
    kCGImageSourceCreateThumbnailFromImageAlways: true
]

let scaledImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0, options)
```

```swift
// Image size with ImageIO

import ImageIO

let filePath = "/path/to/image.jpg"
let url = NSURL(fileURLWithPath: path)

let imageSource = CGImageSourceCreateWithURL(url, nil)
let properties = CGImageSourceCopyPropertiesAtIndex(imageSource, 0, nil)
let options: [NSString: Any] = [
    kCGImageSourceThumbnailMaxPixelSize: 100,
    kCGImageSourceCreateThumbnailFromImageAlways: true
]

let scaledImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0, options)
```

```swift
// Image size with ImageIO

import ImageIO

let filePath = "/path/to/image.jpg"
let url = NSURL(fileURLWithPath: path)

let imageSource = CGImageSourceCreateWithURL(url, nil)
let properties = CGImageSourceCopyPropertiesAtIndex(imageSource, 0, nil)
let options: [NSString: Any] = [
    kCGImageSourceThumbnailMaxPixelSize: 100,
    kCGImageSourceCreateThumbnailFromImageAlways: true
]

let scaledImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0, options)
```

```
// Image size with ImageIO

import ImageIO

let filePath = "/path/to/image.jpg"
let url = NSURL(fileURLWithPath: path)

let imageSource = CGImageSourceCreateWithURL(url, nil)
let properties = CGImageSourceCopyPropertiesAtIndex(imageSource, 0, nil)
let options: [NSString: Any] = [
    kCGImageSourceThumbnailMaxPixelSize: 100,
    kCGImageSourceCreateThumbnailFromImageAlways: true
]

let scaledImage = CGImageSourceCreateThumbnailAtIndex(imageSource, 0, options)
```

Optimizing when in the background


9:41
Venus
Credit: NASA/JPL/USGS
Filters


9:41
FaceTime
Monday
4
Calendar
Photos
Camera
Mail
Clock
Maps
280
Weather
Notes
Reminders
News
Stocks
TV
iTunes Store
App Store
Books
Health
Home
Wallet
Settings

Unload large resources you cannot see.

Optimizing When in the Background

Unloading when off-screen

App lifecycle

- Listen for `UIApplicationWillEnterForeground` and `UIApplicationDidEnterBackground` notifications
- Applies to on-screen views

`UIViewController` appearance cycle

- Leverage `viewWillAppear` and `viewDidDisappear`
- Off-screen view controllers in `UITabBarController` and `UINavigationController`

```
// Unloading resources on foreground/background

NotificationCenter.default.addObserver(forName: .UIApplicationDidEnterBackground,
                                       object: nil,
                                        queue: .main) { [weak self] (note) in
                                            // Unload large resources when off-screen
                                            self.unloadImages()
}


NotificationCenter.default.addObserver(forName: .UIApplicationWillEnterForeground,
                                       object: nil,
                                        queue:.main) { [weak self] (note) in
                                            // Load large resources when off-screen
                                            self.loadImages()
}
```

```
// Unloading resources on foreground/background

NotificationCenter.default.addObserver(forName: .UIApplicationDidEnterBackground,
                                       object: nil,
                                        queue: .main) { [weak self] (note) in
                                            // Unload large resources when off-screen
                                            self.unloadImages()
}


NotificationCenter.default.addObserver(forName: .UIApplicationWillEnterForeground,
                                       object: nil,
                                        queue:.main) { [weak self] (note) in
                                            // Load large resources when off-screen
                                            self.loadImages()
}
```

```
// Unloading resources on foreground/background

NotificationCenter.default.addObserver(forName: .UIApplicationDidEnterBackground,
                                       object: nil,
                                        queue: .main) { [weak self] (note) in
                                            // Unload large resources when off-screen
                                            self.unloadImages()
}
```

```
NotificationCenter.default.addObserver(forName: .UIApplicationWillEnterForeground,
                                       object: nil,
                                        queue:.main) { [weak self] (note) in
                                            // Load large resources when off-screen
                                            self.loadImages()
}
```

```swift
// Unloading resources on foreground/background


// Unload large resource when off-screen
override func viewDidDisappear(_ animated: Bool) {
    unloadImages()
    super.viewDidDisappear(animated)
}

// Load large resources when on-screen
override func viewWillAppear(_ animated: Bool) {
    loadImages()
    super.viewWillAppear(animated)
}
```

```
// Unloading resources on foreground/background


// Unload large resource when off-screen
override func viewDidDisappear(_ animated: Bool) {
    unloadImages()
    super.viewDidDisappear(animated)
}

// Load large resources when on-screen
override func viewWillAppear(_ animated: Bool) {
    loadImages()
    super.viewWillAppear(animated)
}
```

```
// Unloading resources on foreground/background


// Unload large resource when off-screen
override func viewDidDisappear(_ animated: Bool) {
    unloadImages()
    super.viewDidDisappear(animated)
}

// Load large resources when on-screen
override func viewWillAppear(_ animated: Bool) {
    loadImages()
    super.viewWillAppear(animated)
}
```

Demo

Kris Markel, Apple/Software Engineer

Summary

Summary

Memory is a finite and shared resource

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Let iOS pick your image formats

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Let iOS pick your image formats

Use ImageIO for downsampling images

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Let iOS pick your image formats

Use ImageIO for downsampling images

Unload large resources that are off-screen

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Let iOS pick your image formats

Use ImageIO for downsampling images

Unload large resources that are off-screen

Use memory graphs to further understand and reduce memory footprint

Summary

Memory is a finite and shared resource

Monitor memory use when running from Xcode

Let iOS pick your image formats

Use ImageIO for downsampling images

Unload large resources that are off-screen

Use memory graphs to further understand and reduce memory footprint

More Information

https://developer.apple.com/wwdc18/416


WWDC18