Speed 4: Overclocked

PERFORMANCE TUNING FOR IOS

LunarLincoln – Jonathan Wiley (@JonathanWiley)

4.25.2018

INTRODUCING...

Meteorite Mania

Meteorite Mania Requirements

- Must load all 45k meteorites from a CSV into a local database each time the app is launched.
- Must show a list of all data.
- Must show a map with all data plotted at once.
- Must show graphs of the data.
- Must give credit for resources used to create app.

Meteorite Mania Requirements

BEST PRACTICE #1

Don't prematurely optimize

Let’s see what happens out of the box…

TO THE CODE!

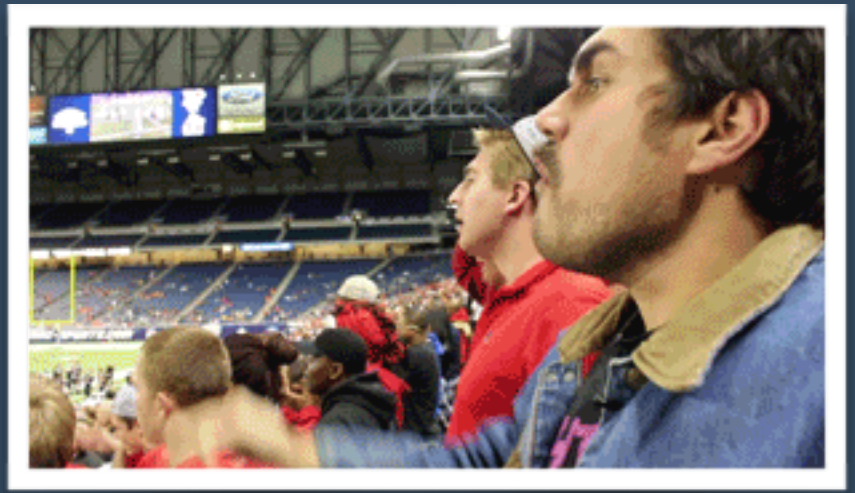
But how will you make your app faster?

- Panic?
- Think the requirements are stupid?
- Question the entire app's purpose?
- Wonder why phones are so terribly slow?

Take a deep breath, realize you're an engineer, and optimize your solution

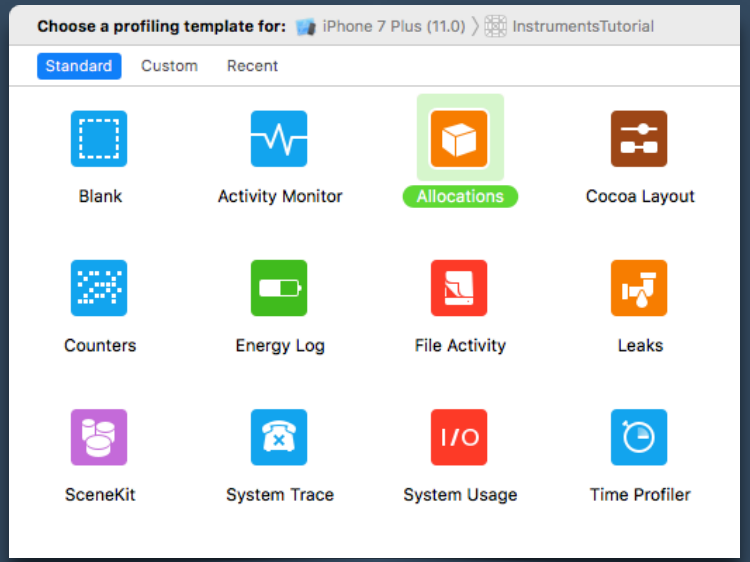
BEST PRACTICE #2

Don’t guess. Measure and plan.



Introducing: Instruments!

- A tool suite bundled with Xcode
- Analyze the performance of your app
- Find code blocking your main thread
- Analyze memory usage and find leaks
- Find zombies (deallocated memory access)



Let’s use Instruments to optimize our code…

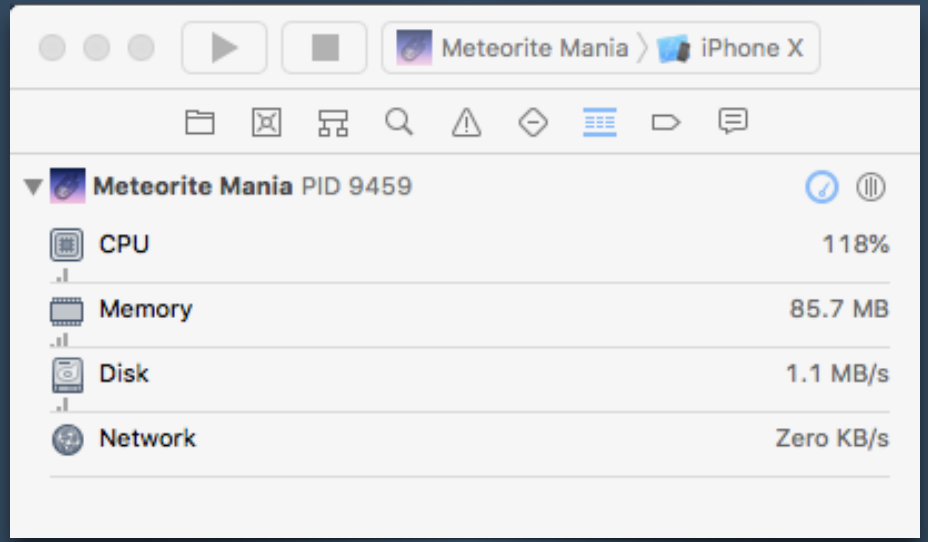
TO THE CODE!

WE DID IT! WHAT DID WE LEARN?

Let's Recap

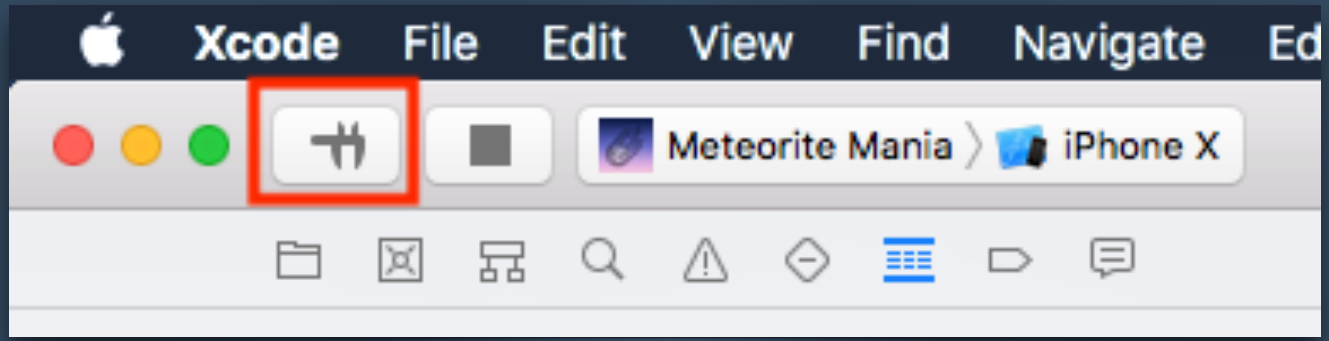
Debugging Gauges

Keep an eye on Xcode's debugging gauges to spot performance problems early



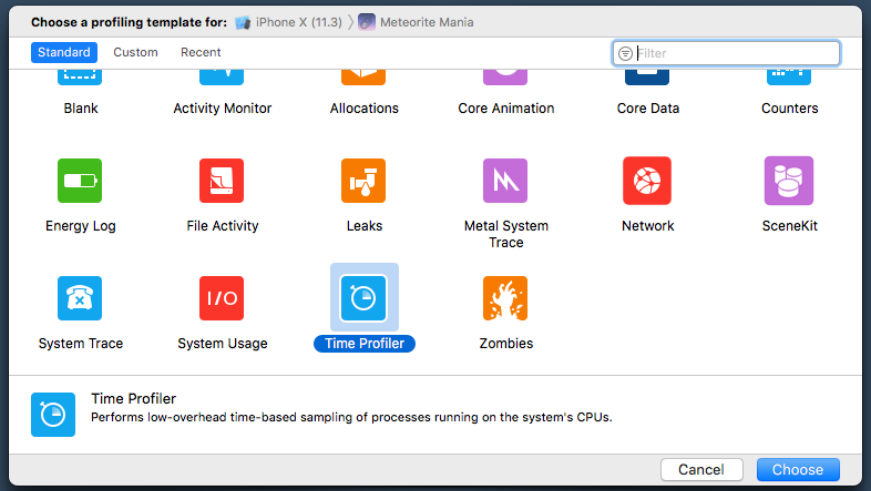
Profile with Instruments

- Press ⌘+ I in Xcode
- Or select “Profile” by long clicking the run button in Xcode



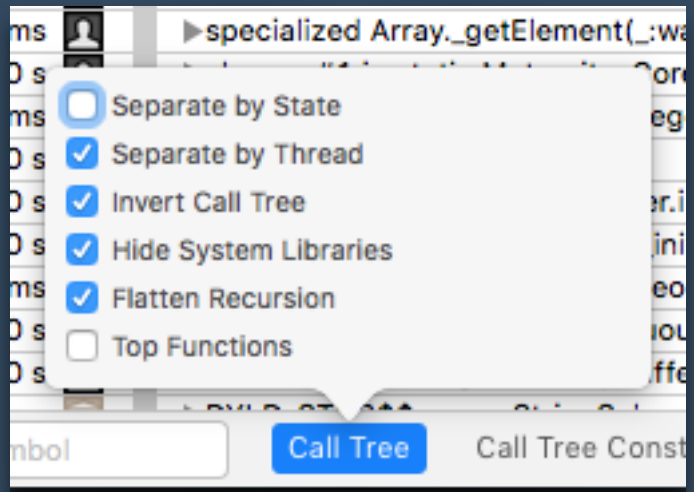
Time Profile for Performance

Use the time profile template in Instruments to identify performance issues



Control Your Call Tree

Filter out underperforming code that you can actually change by adjusting your call tree options



Find Bottlenecks

Find the bottlenecks in your code by looking at the top functions using CPU time and double clicking on their results

Time Profiler 〉 Profile 〉 Root 〉 specialized closure #1 in static MeteoritesCoreDataStore.importNASAMeteoritesCSV(csvFileName:)

MeteoritesCoreDataStore.swift

```
27          for row in rows {
28              let columns = row.components(separatedBy: ",")
29              result.append(columns)
30          }
31
32          result.remove(at: 0)
33
34          result.forEach { (meteoriteRowStrings) in
35
36              let name = meteoriteRowStrings.count > 0 ? meteoriteRowStrings[0] : nil
37              let nasaID = meteoriteRowStrings.count > 1 ? Int32(meteoriteRowStrings[1]) : nil
38              let nameType = meteoriteRowStrings.count > 2 ? meteoriteRowStrings[2] : nil
39              let recClass = meteoriteRowStrings.count > 3 ? meteoriteRowStrings[3] : nil
40              let massInGrams = meteoriteRowStrings.count > 4 ? Double(meteoriteRowStrings[4]) : nil
41              let fall = meteoriteRowStrings.count > 5 ? meteoriteRowStrings[5] : nil
42              let year = meteoriteRowStrings.count > 6 ? csvDateFormatter.date(from: meteoriteRowStrings[6]) : nil    1328x
43              let recLat = meteoriteRowStrings.count > 7 ? Double(meteoriteRowStrings[7]) : nil
44              let recLong = meteoriteRowStrings.count > 8 ? Double(meteoriteRowStrings[8]) : nil    1x
45              let geoLocation = meteoriteRowStrings.count > 10 ? meteoriteRowStrings[9] + "," + meteoriteRowStrings[10] : nil
46
47              createNewMeteorite(name: name,
48                                 nasaID: nasaID,
49                                 nameType: nameType,
50                                 recClass: recClass,
51                                 massInGrams: massInGrams,
52                                 fall: fall,
53                                 year: year,
54                                 recLat: recLat,
55                                 recLong: recLong,
56                                 geoLocation: geoLocation)
```

MeteoritesCoreDataStore.swift, Line 166- : 0 Samples

Input Filter | Involves Symbol | Call Tree | Call Tree Constraints | Data Mining

Now, let's build the map screen...

TO THE CODE!

BEST PRACTICE #3

Think outside the box.

Can we approach this issue differently?...

TO THE CODE!

WE DID IT! BUT HOW AGAIN?

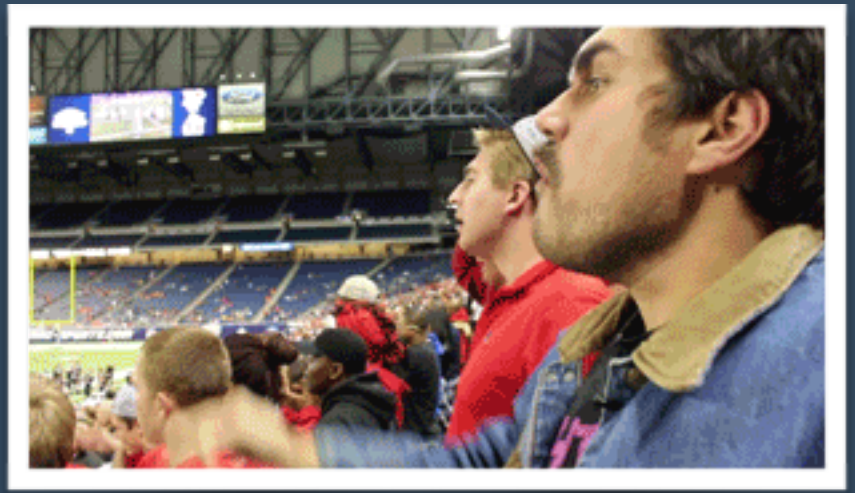
Let's Recap

BEST PRACTICE #1

Don't prematurely optimize

BEST PRACTICE #2

Don’t guess. Measure and plan.



BEST PRACTICE #3

Think outside the box.

Use Instruments

- Time profiler for speed
- Allocations for memory
- Zombies for bad access
- Just press ⌘+ I

SPEED 5: THE SEQUEL

Code Challenge

Speed 5: The Sequel - Code Challenge

- We've stubbed out the graphs and currently they're very fancy...and slow

- MAKE
- THEM
- SPEEDIER

LunarLincoln

Questions?

@LunarLincoln | www.LunarLincoln.com