Debugging Closed Source Code

a whodunnit mystery

THE FOLLOWING PRESENTATION HAS BEEN APPROVED FOR

ALL AUDIENCES

edited for brevity

and narration purposes

we’ll go on a journey

to find the culprit

some will be familiar

and comforting

some will be new

and exciting

Photo by Pok Rie from Pexels
https://www.pexels.com/photo/car-driving-along-curvy-road-in-lush-forest-4847538/

it is a hot summer day

British Summer Time (UTC+1)

a messaging app

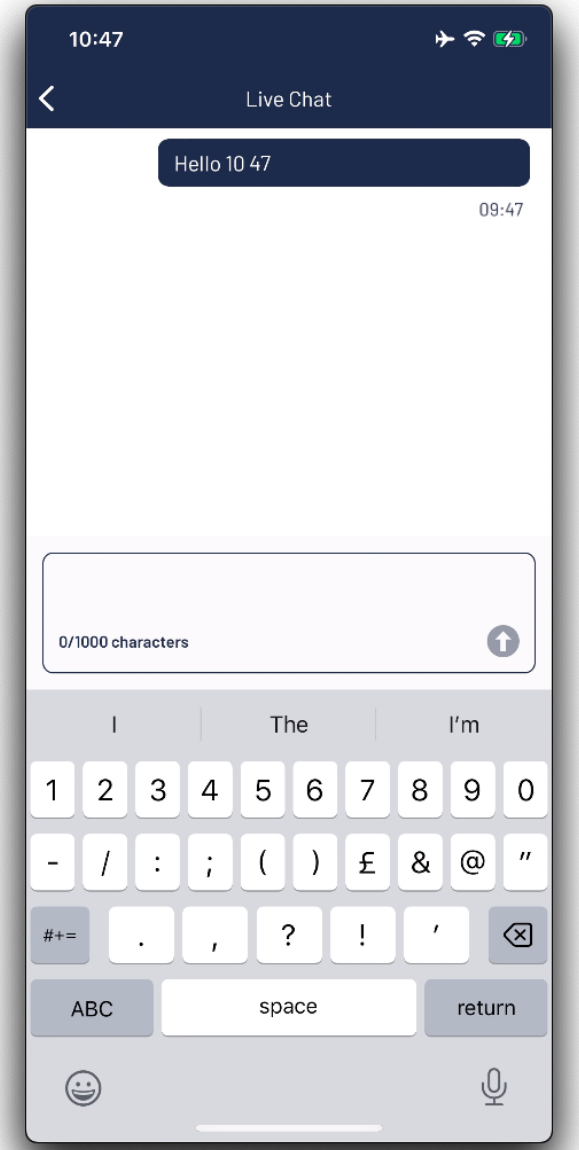
the scene

a bug is raised

the report

the message timestamp is wrong

the mystery


10:47
Live Chat
Hello 10 47
09:47
0/1000 characters
I
The
I'm
1 2 3 4 5 6 7 8 9 0
- / : ; ( ) £ & @ "
#+= . , ? ! '
ABC
space
return

10:47

time on the device

09:47

time on message

10:47

BST (UTC+1)

09:47

BST (UTC+1)

seemingly 1 hour apart

let’s look at the data

timestamp

```swift
let createdAt = message.createdAt

let timestamp = createdAt.string(
     template: "HHmm",
     locale: Locale(identifier: "en_GB"),
     using: .gregorianAutoUpdatingCurrentTimezone)
```

09:47

indeed off by 1 hour

let’s look at the code

parsing and formatting dates

```swift
let message = ICMessage() //IMIconnectCoreSDK.ICMessage
message.message = “Hello 10:47”

let before = message.createdAt.string(
     template: "HHmm",
     locale: Locale(identifier: "en_GB"),
     using: .gregorianAutoUpdatingCurrentTimezone)

try await ICMessaging.shared().publishMessage(message) //IMIconnectCoreSDK.ICMessage

let after = message.createdAt.string(
     template: "HHmm",
     locale: Locale(identifier: "en_GB"),
     using: .gregorianAutoUpdatingCurrentTimezone)
```

10:47

before

09:47

after

what does the server say?

response

let’s look at the network

JSON

using a proxy

e.g. Charles

```
{
  "code": 0,
  "created_at": "2023-06-15T09:47:16.910Z",
  status: "Success",
  "tid": "RDFC56842-CF7C-4F4E-8C3A-7D44728849E6"
}
```

```
{
  "code": 0,
  "created_at": "2023-06-15T09:47:16.910Z",
  status: "Success",
  "tid": "RDFC56842-CF7C-4F4E-8C3A-7D44728849E6"
}
```

```
{
  "code": 0,
  "created_at": "2023-06-15T09:47:16.910Z",
  status: "Success",
  "tid": "RDFC56842-CF7C-4F4E-8C3A-7D44728849E6"
}
```

09:47

created_at *string* in UTC

```swift
let createdAt = message.createdAt

let timestamp = createdAt.string(
     template: "HHmm",
     locale: Locale(identifier: "en_GB"),
     using: .gregorianUTCTimezone)
```

08:47

created_at *date* in UTC

```swift
let createdAt = message.createdAt

let timestamp = createdAt.string(
     template: "HHmm",
     locale: Locale(identifier: "en_GB"),
     using: .gregorianAutoUpdatingCurrentTimezone)
```

09:47

British Summer Time (UTC+1)

something seems off

in the SDK

can’t reproduce

response

raise your hand 

if you have heard (or said) this before

a piece to the puzzle mystery

“We know inconsistencies can arise when **parsing ISO8601** or any other fixed-format **dates** on iOS depending on how the parsing is done. To **guarantee consistency** across simulators, devices and user settings, you should either be using an ISO8601DateFormatter or a **NSDateFormatter with its Locale set to en_US_POSIX** and a timezone of UTC (or GMT)."

SDK

how does it parse the date?

“The attribute is parsed in the aforementioned date time parsing strategy using en-US_POSIX and UTC.”

response

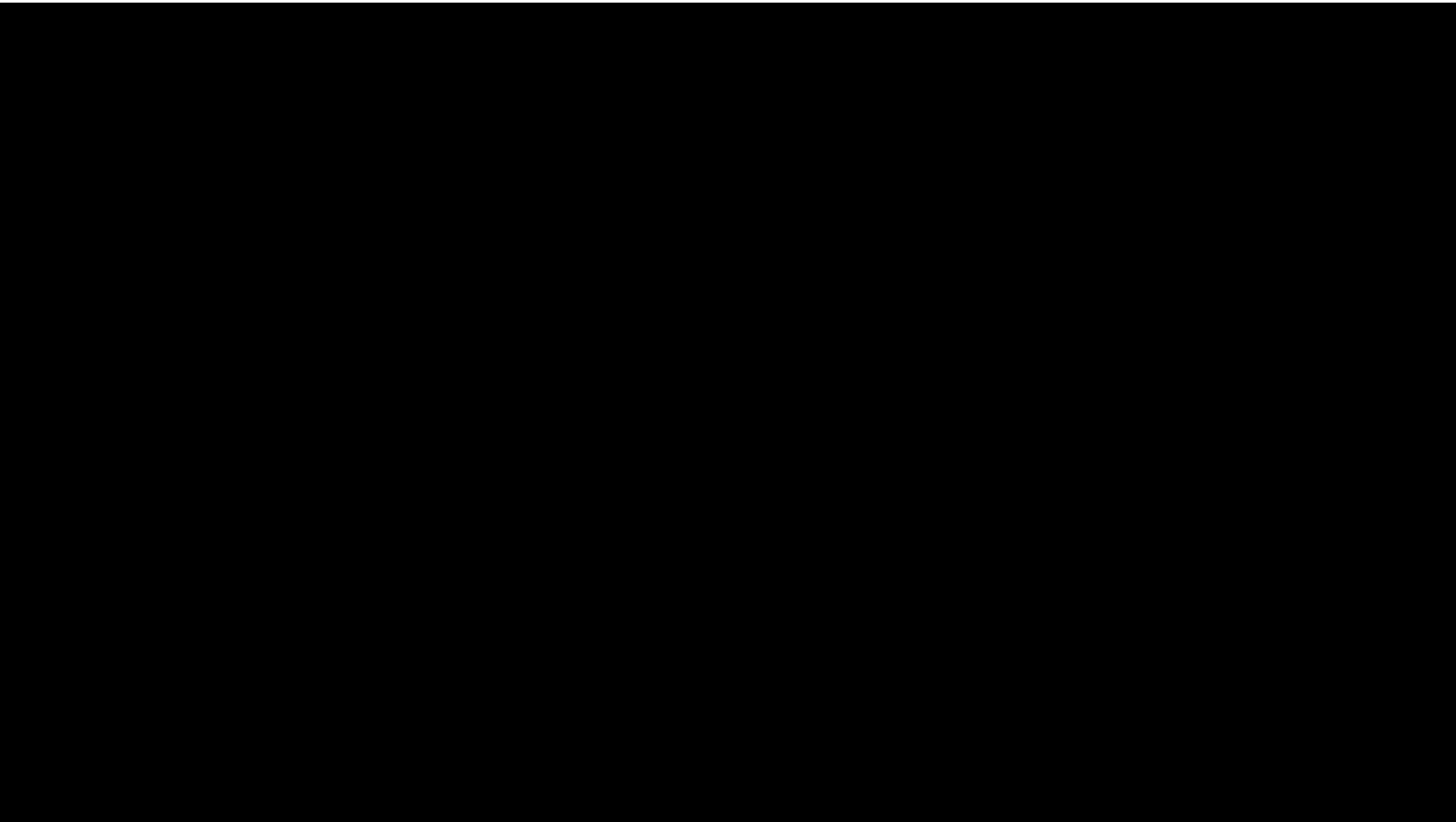
a piece to the puzzle mystery

version

are we testing the same?

“Checked my app, and it was using an **earlier** version of the SDK”

response



“However, I’ve now updated it to use the same version you are using, and I still do not see any issue.”

response

I ain’t got time for that

I am going in

let’s debug the SDK

but how?

we don’t have the source

Symbolic Breakpoint

what is a symbolic breakpoint?

where do I find the symbols?

a symbol table

nm

display name list (symbol table)

whereis nm

nm: /usr/bin/nm /Applications/Xcode.app/Contents/Developer/Toolchains/XcodeDefault.xctoolchain/usr/share/man/man1/nm.1

symbol table

nm ./MysteryApp.app/MysteryApp

-[ICMessage setCreatedAt:]

symbolic breakpoint

let’s look at the codebase

at runtime

bt (backtrace)

```
(lldb) bt
* thread #1, queue = 'com.apple.main-thread', stop reason = breakpoint 6.1
 * frame #0: 0x0000000117424434 IMIconnectCoreSDK`-[ICMessage setCreatedAt:]
   frame #1: 0x0000000117332e8b IMIconnectCoreSDK`__53-[MessagingManager publishMessage:completionHandler:]_block_invoke.410 + 501
   frame #2: 0x000000011741e2a2 IMIconnectCoreSDK`-[RequestManager handleSuccessResponse:completionHandler:] + 675
   frame #3: 0x0000000120992d18 libdispatch.dylib`_dispatch_call_block_and_release + 12
   frame #4: 0x0000000120993f5b libdispatch.dylib`_dispatch_client_callout + 8
   frame #5: 0x00000001209a4e5c libdispatch.dylib`_dispatch_main_queue_drain + 1726
   frame #6: 0x00000001209a4790 libdispatch.dylib`_dispatch_main_queue_callback_4CF + 31
   frame #7: 0x000000011aa82b1f CoreFoundation`__CFRUNLOOP_IS_SERVICING_THE_MAIN_DISPATCH_QUEUE__ + 9
   frame #8: 0x000000011aa7d436 CoreFoundation`__CFRunLoopRun + 2482
   frame #9: 0x000000011aa7c6a7 CoreFoundation`CFRunLoopRunSpecific + 560
   frame #10: 0x000000011e24c28a GraphicsServices`GSEventRunModal + 139
   frame #11: 0x000000013e506ad3 UIKitCore`-[UIApplication _run] + 994
   frame #12: 0x000000013e50b9ef UIKitCore`UIApplicationMain + 123

(lldb)
```

Objective-C Runtime

objc_msgSend

objc_msgSend

parameters

self

first

ICMessage

first

op (selector)

second

setCreatedAt:

second

variable arguments

third

NSDate instance

third

registers

objc_msgSend

message

$rdi (first)

method

$rsi (second)

date set

$rdx (third)

po $rdx (createdAt)

```
(lldb) po $rdx
2023-06-15 08:47:16 +0000

(lldb)
```

frame select 2

```
(lldb) frame select 2
frame #2: 0x00000001153bb2a2 IMIconnectCoreSDK`-[RequestManager handleSuccessResponse:completionHandler:] + 675
IMIconnectCoreSDK`-[RequestManager handleSuccessResponse:completionHandler:]:
-> 0x1153bb2a2 <+675>: movq   0x4612f(%rip), %r12   ;(void *)0x0000000112cfe5c0: objc_release
   0x1153bb2a9 <+682>: movq   -0x40(%rbp), %rdi
   0x1153bb2ad <+686>: callq  *%r12
   0x1153bb2b0 <+689>: movq   %r12 %r14
```

bt (backtrace)

```
(lldb) bt
* thread #1, queue = 'com.apple main-thread', stop reason = breakpoint 5.1
   frame #0: 0x00000001153c1434 IMIconnectCoreSDK`-[ICMessage setCreatedAt:]
   frame #1: 0x00000001152cfe8b IMIconnectCoreSDK`__53-[MessagingManager publishMessage:completionHandler:]_block_invoke.410 + 501
 * frame #2: 0x00000001153bb2a2 IMIconnectCoreSDK`-[RequestManager handleSuccessResponse:completionHandler:] + 675
   frame #3: 0x000000011e92fd18 libdispatch.dylib`_dispatch_call_block_and_release + 12
   frame #4: 0x000000011e930f5b libdispatch.dylib`_dispatch_client_callout + 8
   frame #5: 0x000000011941e5c libdispatch.dylib`_dispatch_main_queue_drain + 1726
   frame #6: 0x000000011941790 libdispatch dylib`_dispatch_main_queue_callback_4CF + 31
   frame #7: 0x0000000118a1fbif CoreFoundation`CFRunLoopRun + 2482
   frame #9: 0x0000000118a196a7 CoreFoundation`CFRunLoopRunSpecific + 560
   frame #10: 0x000000011c1e928a GraphicsServices GSEventRunModal + 139
   frame #11: 0x0000000134a3ad3 UIKitCore`-[UIApplication _run] + 994
   frame #12: 0x000000013c4a89ef UIKitCore`UIApplicationMain + 123

(lldb)
```

register read (local variables)

```
(lldb) register read
General Purpose Registers:
   rbx = 0x0000600001ba49c0
   rbp = 0x000000030f28d800
   rsp = 0x000000030f28d7c0
   r12 = 0x0000000112ce07c0    libobjc.A.dylib`objc_msgSend
   r13 = 0x0000000000000000
   r14 = 0x0000600002189600
   r15 = 0x0000600003c945a0
   rip = 0x000000011153bb2a2    IMIconnectCoreSDK`-[RequestManager handleSuccessResponse:completionHandler:] + 675
13 registers were unavailable.
```

po $r14

```
(lldb) po $r13
<nil>

(lldb) po $r14
{
   code = 0;
   "created_on" = "2023-06-15T09:47:16.910Z";
   status = Success;
   tid = "RDFC56842-CF7C-4F4E-8C3A-7D44728849E6";
}

(lldb) po $r15
<7b22636f 6465223a 20302c20 22737461 74757322 3a202253 75636365 7373222c 20227469 64223a20
22524446 43353638 34322d43 4637432d 34463445 2d384333 412d3744 34343732 38383439 4536222c
20226372 65617465 645f6f6e 223a2022 32303233 2d30362d 31355431 343a3533 3a31362e 3931305a
227d>
```

register $r14

```
{
  code = 0;
  "created_on" = “2023-06-15T09:47:16.910Z”;
  status = Success;
  tid = "RDFC56842-CF7C-4F4E-8C3A-7D44728849E6";
}
```

createdAt

2023-06-15T08:47:16.910Z

let’s look at the clues

unable to reproduce

outside of our codebase 🧩

date parsing is correct

in the SDK source code 🧩

that leaves us

with

one thing

that can alter

runtime execution

and that is…

Objective-C category

aka extension

“Given how the SDK team has been **unable to reproduce** 🧩 this issue and **how the date parsing is correct** 🧩, I am now considering whether this is caused by an **Objective-C category at runtime.**”

“**This category** belongs either to our codebase or in some third party dependency that customises any existing classes that are **involved in the parsing.”**

“**Potential candidates** may be NSDate, NSDateFormatter and related methods that are **used to create either a date instance or used to parse string dates.**”

“Our SDK basically does this”

response

```objc
+ (NSDate *)dateFromString:(NSString *)dateString withFormat:(NSString *)format
{
    NSDateFormatter *dateFormatter = [[NSDateFormatter alloc] init];
    NSLocale *locale = [[NSLocale alloc] initWithLocaleIdentifier:@"en_US_POSIX"];
    [dateFormatter setLocale:locale];
    [dateFormatter setTimeZone:[NSTimeZone timeZoneWithAbbreviation:@"GMT"]];
    [dateFormatter setDateFormat:format];
    return [dateFormatter dateFromString:dateString];
}
```

meanwhile…

“Is that method defined as
an extension to the
NSDate type?”

"Yes, it is defined on the NSDate type.”

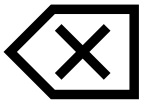
response

!

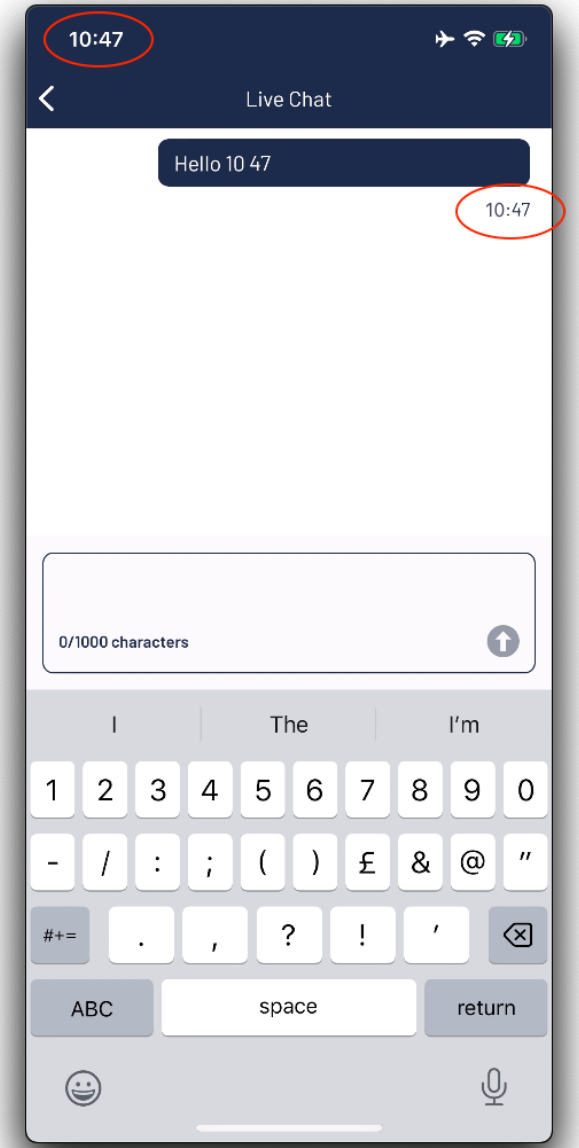
```objc
+ (NSDate *)dateFromString:(NSString *)string withFormat:(NSString *)format
{
    NSDateFormatter *formatter = [[NSDateFormatter alloc] init];
    formatter.dateFormat = format;
    return [formatter dateFromString:string];
}
```

Notice the method signature

```
+ (NSDate *)dateFromString:(NSString *)dateString withFormat:(NSString *)format
+ (NSDate *)dateFromString:(NSString *)string     withFormat:(NSString *)format
```



delete!


10:47
Live Chat
Hello 10 47
10:47
0/1000 characters
I
The
I'm
1 2 3 4 5 6 7 8 9 0
- / : ; ( ) £ & @ "
#+= . , ? ! '
ABC
space
return

what’s the moral

of the story?

always be prefixing

your class extensions

Thank you

and for more debugging…

www.youtube.com/**@qnoid**

Sound Debugging

References

- https://developer.apple.com/forums/thread/729071
- https://developer.apple.com/documentation/xcode/setting-breakpoints-to-pause-your-running-app#Pause-on-a-symbol-outside-your-code
- https://developer.apple.com/videos/play/wwdc2022/110370/
- https://developer.apple.com/documentation/objectivec/1456712-objc_msgsend
- https://releases.llvm.org/14.0.0/tools/clang/docs
- https://lldb.llvm.org/man/lldb.html
- https://math.hws.edu/eck/cs220/f22/registers.html
- https://developer.apple.com/library/archive/documentation/Cocoa/Conceptual/ProgrammingWithObjectiveC/CustomizingExistingClasses/CustomizingExistingClasses.html