Comparing NSDates without time component

Asked 8 years, 10 months ago Modified 1 year, 9 months ago Viewed 58k times

84

In a swift playground, I have been using

```
NSDate.date()
```

But, this always appears with the time element appended. For my app I need to ignore the time element. Is this possible in Swift? How can it be done? Even if I could set the time element to be the same time on every date that would work too.

Also, I am trying to compare two dates and at the moment I am using the following code:

```
var earlierDate:NSDate = firstDate.earlierDate(secondDate)
```

Is this the only way or can I do this in a way that ignores the time element? For instance I don't want a result if they are the same day, but different times.

ios time swift nsdate foundation

Share Improve this question Follow

edited Oct 5, 2014 at 0:56
jww
96.2k 89 406 876

asked Jul 4, 2014 at 15:06
agf119105
1,742 4 14 22

1 You don't want to "ignore" the time part of the object. It seems that you want to format the date as a string without the time part. You can do that using `NSDateFormatter` . – The Paramagnetic Croissant Jul 4, 2014 at 15:11

17 Answers

Sorted by: Highest score (default)

154

Use this `Calendar` function to compare dates in iOS 8.0+

```
func compare(_ date1: Date, to date2: Date, toGranularity component:
Calendar.Component) -> ComparisonResult
```

passing `.day` as the unit

Use this function as follows:

```
let now = Date()
// "Sep 23, 2015, 10:26 AM"
let olderDate = Date(timeIntervalSinceNow: -10000)
// "Sep 23, 2015, 7:40 AM"

var order = Calendar.current.compare(now, to: olderDate, toGranularity: .hour)

switch order {
case .orderedDescending:
    print("DESCENDING")
case .orderedAscending:
    print("ASCENDING")
case .orderedSame:
    print("SAME")
}

// Compare to hour: DESCENDING

var order = Calendar.current.compare(now, to: olderDate, toGranularity: .day)


switch order {
case .orderedDescending:
    print("DESCENDING")
case .orderedAscending:
    print("ASCENDING")
case .orderedSame:
    print("SAME")
}

// Compare to day: SAME
```

Share Improve this answer Follow edited Oct 22, 2018 at 10:17 answered Jul 4, 2014 at 15:15

Ashley Mills
49.7k 16 126 159

is this from UIKit? I cannot find it in a Foundation project. – vikingosegundo Jul 4, 2014 at 15:30

2 This is one of the new iOS 8.0 `NSCalendar` APIs. See `NSCalendar.h` in *iOS 7.1 to iOS 8.0 API Differences* – Ashley Mills Jul 4, 2014 at 15:36

1 finally, I always missed it. – vikingosegundo Jul 4, 2014 at 15:36

1 @theReverend that was a hangover from an old version of Swift. Updated for 2.0 - thanks! – Ashley Mills Sep 23, 2015 at 9:30

2 I think the two comments, indicating the expected results are wrong. `// Compare to day: DESCENDING` should be `SAME` and vice versa. – jaw Feb 6, 2016 at 13:44

25

Xcode 11.2.1, Swift 5 & Above

Checks whether the date has same day component.

```
Calendar.current.isDate(date1, equalTo: date2, toGranularity: .day)
```

Adjust toGranularity as your need.

Share Improve this answer Follow

answered Jan 31, 2020 at 9:41
Saranjith
11.2k 4 64 120

22

There are several useful methods in NSCalendar in iOS 8.0+:

```
startOfDayForDate, isDateInToday, isDateInYesterday, isDateInTomorrow
```

And even to compare days:

```
func isDate(date1: NSDate!, inSameDayAsDate date2: NSDate!) -> Bool
```

To ignore the time element you can use this:

```
var toDay = Calendar.current.startOfDay(for: Date())
```

But, if you have to support also iOS 7, you can always write an extension

```
extension NSCalendar {
    func myStartOfDayForDate(date: NSDate!) -> NSDate!
    {
        let systemVersion:NSString = UIDevice.currentDevice().systemVersion
        if systemVersion.floatValue >= 8.0 {
            return self.startOfDayForDate(date)
        } else {
            return self.dateFromComponents(self.components(.CalendarUnitYear |
.CalendarUnitMonth | .CalendarUnitDay, fromDate: date))
        }
    }
}
```

Share Improve this answer Follow

edited Nov 12, 2018 at 15:31
beryllium
29.6k 15 104 125

answered Aug 28, 2014 at 0:30
slamor
3,365 2 14 13

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

11

In Swift 4:

```
func compareDate(date1:Date, date2:Date) -> Bool {
    let order = NSCalendar.current.compare(date1, to: date2, toGranularity:
.day)
    switch order {
    case .orderedSame:
        return true
    default:
        return false
    }
}
```

Share Improve this answer Follow

answered Nov 14, 2017 at 0:02

zs2020
53.6k 28 153 219

8

I wrote the following method to compare two dates by borrowing from Ashley Mills solution. It compares two dates and returns true if the two dates are the same (stripped of time).

```
func compareDate(date1:NSDate, date2:NSDate) -> Bool {
    let order = NSCalendar.currentCalendar().compareDate(date1, toDate: date2,
        toUnitGranularity: .Day)
    switch order {
    case .OrderedSame:
        return true
    default:
        return false
    }
}
```

And it is called like this:

```
if compareDate(today, date2: anotherDate) {
    // The two dates are on the same day.
}
```

Share Improve this answer Follow

answered Sep 27, 2016 at 22:48

Emmett Corman
81 1 1

7

Two Dates comparisions in swift.

```
    // Date comparision to compare current date and end date.
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

```
{
    // Current date is smaller than end date.
}
else if dateComparisionResult == NSComparisonResult.OrderedDescending
{
    // Current date is greater than end date.
}
else if dateComparisionResult == NSComparisonResult.OrderedSame
{
    // Current date and end date are same.
}
```

Share Improve this answer Follow edited Sep 3, 2014 at 9:49 answered Sep 3, 2014 at 9:29
abhi
563 6 9

5 Not quite. The NSDate type represents a point in time; it contains date elements and time elements. So your code compares two points in time as opposed to the intended calendar dates in the original question. – Yer00n Oct 21, 2014 at 23:01

7

I wrote a Swift 4 extension for comparing two dates:

```
import Foundation

extension Date {
  func isSameDate(_ comparisonDate: Date) -> Bool {
    let order = Calendar.current.compare(self, to: comparisonDate,
toGranularity: .day)
    return order == .orderedSame
  }

  func isBeforeDate(_ comparisonDate: Date) -> Bool {
    let order = Calendar.current.compare(self, to: comparisonDate,
toGranularity: .day)
    return order == .orderedAscending
  }

  func isAfterDate(_ comparisonDate: Date) -> Bool {
    let order = Calendar.current.compare(self, to: comparisonDate,
toGranularity: .day)
    return order == .orderedDescending
  }
}
```

Usage:

```
startDate.isSameDateAs(endDate) // returns a true or false
```

Share Improve this answer Follow edited Nov 24, 2017 at 16:31 answered Nov 24, 2017 at 16:17

6

For iOS7 support

```
let dateFormatter = NSDateFormatter()
dateFormatter.dateFormat = "yyyy-MM-dd"
let date1String = dateFormatter.stringFromDate(date1)
let date2String = dateFormatter.stringFromDate(date2)
if date1String == date2String {
    println("Equal date")
}
```

Share Improve this answer Follow

answered Dec 31, 2014 at 10:02
Loganathan
1,657 1 13 17

other logic operator, like `<` , `>` , `>=` and `<=` , works as well. – Ângelo Polotto Oct 21, 2020 at 13:39

5

You can compare two dates using it's description.

```
let date1 = NSDate()
let date2 = NSDate(timeIntervalSinceNow: 120)
if date1.description == date2.description {
    print(true)
} else {
    print(false)   // false (I have added 2 seconds between them)
}
```

If you want set the time element of your dates to a different time you can do as follow:

```
extension NSDate {
    struct Calendar {
        static let gregorian = NSCalendar(calendarIdentifier:
NSCalendarIdentifierGregorian)!
    }
    var day:    Int { return Calendar.gregorian.component(.Day,    fromDate:
self)   }
    var month:  Int { return Calendar.gregorian.component(.Month,  fromDate:
self)  }
    var year:   Int { return Calendar.gregorian.component(.Year,   fromDate:
self)  }

    var noon: NSDate {
        return Calendar.gregorian.dateWithEra(1, year: year, month: month, day:
day, hour: 12, minute: 0, second: 0, nanosecond: 0)!
    }
}

let date1 = NSDate()
let date2 = NSDate(timeIntervalSinceNow: 120)
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

or you can also do it using NSDateFormatter:

```
extension NSDate {
    struct Date {
        static let formatterYYYYMMDD: NSDateFormatter = {
            let formatter = NSDateFormatter()
            formatter.dateFormat = "yyyyMMdd"
            return formatter
        }()
    }
    var yearMonthDay: String {
        return Date.formatterYYYYMMDD.stringFromDate(self)
    }
    func isSameDayAs(date:NSDate) -> Bool {
        return yearMonthDay == date.yearMonthDay
    }
}

let date1 = NSDate()
let date2 = NSDate(timeIntervalSinceNow: 120)
print(date1.yearMonthDay == date2.yearMonthDay)   // true

print(date1.isSameDayAs(date2))    // true
```

Another option (iOS8+) is to use calendar method isDate(inSameDayAsDate:):

```
extension NSDate {
    struct Calendar {
        static let gregorian = NSCalendar(calendarIdentifier:
NSCalendarIdentifierGregorian)!
    }
    func isInSameDayAs(date date: NSDate) -> Bool {
        return Calendar.gregorian.isDate(self, inSameDayAsDate: date)
    }
}
let date1 = NSDate()
let date2 = NSDate(timeIntervalSinceNow: 120)
if date1.isInSameDayAs(date: date2 ){
    print(true)   // true
} else {
    print(false)
}
```

Share Improve this answer Follow

edited Jan 25, 2016 at 21:06

answered Jan 3, 2015 at 7:26

Leo Dabus
227k 59 478 560

4

Swift 3

```
let order = NSCalendar.current.compare(date1, to: date2, toGranularity:
```

Join Stack Overflow to find the best answer to your technical question, help others answer theirs.

Sign up

```
    }
    else if order == .orderedDescending {
      // date 1 is newer
    }
    else if order == .orderedSame {
      // same day/hour depending on granularity parameter
    }
```

Share Improve this answer Follow

answered Aug 17, 2017 at 11:10
Mobile Developer
5,720 1 38 45

2

For Swift3

```
var order = NSCalendar.current.compare(firstDate, to: secondDate,
toGranularity: .hour)

if order == .orderedSame {
    //Both the dates are same.
    //Your Logic.
}
```

Share Improve this answer Follow

answered Jun 30, 2017 at 7:00
Uzma
209 2 4

1

Swift:

```
extension NSDate {

    /**
    Compares current date with the given one down to the seconds.
    If date==nil, then always return false

    :param: date date to compare or nil

    :returns: true if the dates has equal years, months, days, hours, minutes
and seconds.
    */
    func sameDate(date: NSDate?) -> Bool {
        if let d = date {
            let calendar = NSCalendar.currentCalendar()
            if NSComparisonResult.OrderedSame == calendar.compareDate(self,
toDate: d, toUnitGranularity: NSCalendarUnit.SecondCalendarUnit) {
                return true
            }

        }
        return false
```

Share Improve this answer Follow

answered Jan 19, 2015 at 0:18
Alexander Volkov
7,754 1 47 42

0

When you `NSDate.date()` in the playground, you see the **default description** printed. Use `NSDateFormatter` to print a localized description of the date object, possibly with only the date portion.

To zero out specific portions of a date (for the sake of comparison), use `NSDateComponents` in conjunction with `NSCalendar`.

Share Improve this answer Follow edited Jul 4, 2014 at 15:21

answered Jul 4, 2014 at 15:13
Léo Natan
56.7k 9 148 195

ah, come on, string operations? – vikingosegundo Jul 4, 2014 at 15:14

this doesn't help as I know how to print a localised description without the time element. I want to ignore the time-element as I need to compare two dates without the time getting in the way – agf119105 Jul 4, 2014 at 15:19

@agf119105 In that case, use `NSDateComponents`. – Léo Natan Jul 4, 2014 at 15:20

0

In my experience, most people's problems with using NSDate comes from the incorrect assumption that an NSDate can be used to represent a date in the 'normal' sense (i.e. a 24 period starting at midnight in the local timezone). In normal (everyday / non-programming) usage, 1st January 2014 in London is the same date as 1st January in Beijing or New York **even though they cover different periods in real time**. To take this to the extreme, the time on Christmas Island is UTC+14 while the time on Midway Island is UTC-11. So 1st January 2014 on these two island are the same date even though one doesn't even start until the other has been completed for an hour.

If that is the kind of date you are recording (and if you are not recording the time component, it probably is), then do not use NSDate (which stores only seconds past 2001-01-01 00:00 UTC, nothing else) but store the year month and day as integers - perhaps by creating your own CivilDate class that wraps these values - and use that instead.

Only dip into NSDate to compare dates and then make sure to explicitly declare the time zone as "UTC" on both NSDates for comparison purposes.

Share Improve this answer Follow edited Oct 11, 2014 at 6:40

answered Oct 11, 2014 at 6:34

-1

Swift 4

```
func compareDate(date1:Date, date2:Date) -> Bool {
    let order = Calendar.current.compare(date1, to: date2,toGranularity: .day)
    switch order {
    case .orderedSame:
        return true
    default:
        return false
    }
}
```

Share Improve this answer Follow

answered May 24, 2019 at 10:16
Asad Jamil
198 9

This is an identical copy of @zs2020 answer from 18 months earlier – Ashley Mills Aug 26, 2020 at 7:53

-1

If you need to compare just if date is in the same day as other date use this:

```
Calendar.current.isDate(date1, inSameDayAs: date2)
```

Share Improve this answer Follow

answered Aug 11, 2021 at 10:21
Robert Dresler
10.5k 2 20 38

-3

To answer your question:

> Is this possible in Swift?

Yes, it is possible

Ahh, you also want to now HOW

```
let cal = NSCalendar.currentCalendar()
cal.rangeOfUnit(.DayCalendarUnit, startDate: &d1, interval: nil, forDate: d1)
// d1 NSDate?
cal.rangeOfUnit(.DayCalendarUnit, startDate: &d2, interval: nil, forDate: d2)
// d2 NSDate?
```

compare with `d1!.compare(d2!)`

To display them without time portion, us NSDateFormatter.

Share Improve this answer Follow edited May 26, 2019 at 15:17 answered Jul 4, 2014 at 15:10
vikingosegundo
52k 14 135 177

2 Thanks, Could you answer the how as well?!? – agf119105 Jul 4, 2014 at 15:12

@agf119105 using Google, for instance. – The Paramagnetic Croissant Jul 4, 2014 at 15:12

2 It answers the question as it wording, isn't it? – vikingosegundo Jul 4, 2014 at 15:13

2 The wording of the question: "Is this possible in Swift? How can it be done?" meaning what code is required? @user3477950 tried Google thanks, but no useful results for Swift ... – agf119105 Jul 4, 2014 at 15:16