Advanced Dates and Times in Swift

Jeff Kelley @SlaunchaMan

HEALTHYISH

Squash with Dates and Thyme

52 RATINGS

Dates and Times in Swift Recap

Jeff Kelley @SlaunchaMan

`Date`

A `Date` is a **moment in time**, measured in seconds since January 1, 2001.

Those measurements of seconds are `TimeIntervals`, which are just `Doubles` under the hood.

Calendar

The things we think about when we think about dates and times—months, days, years, hours, minutes, etc.—are called `DateComponents`. To get `DateComponents` from a `Date` and vice versa, you need a `Calendar`.

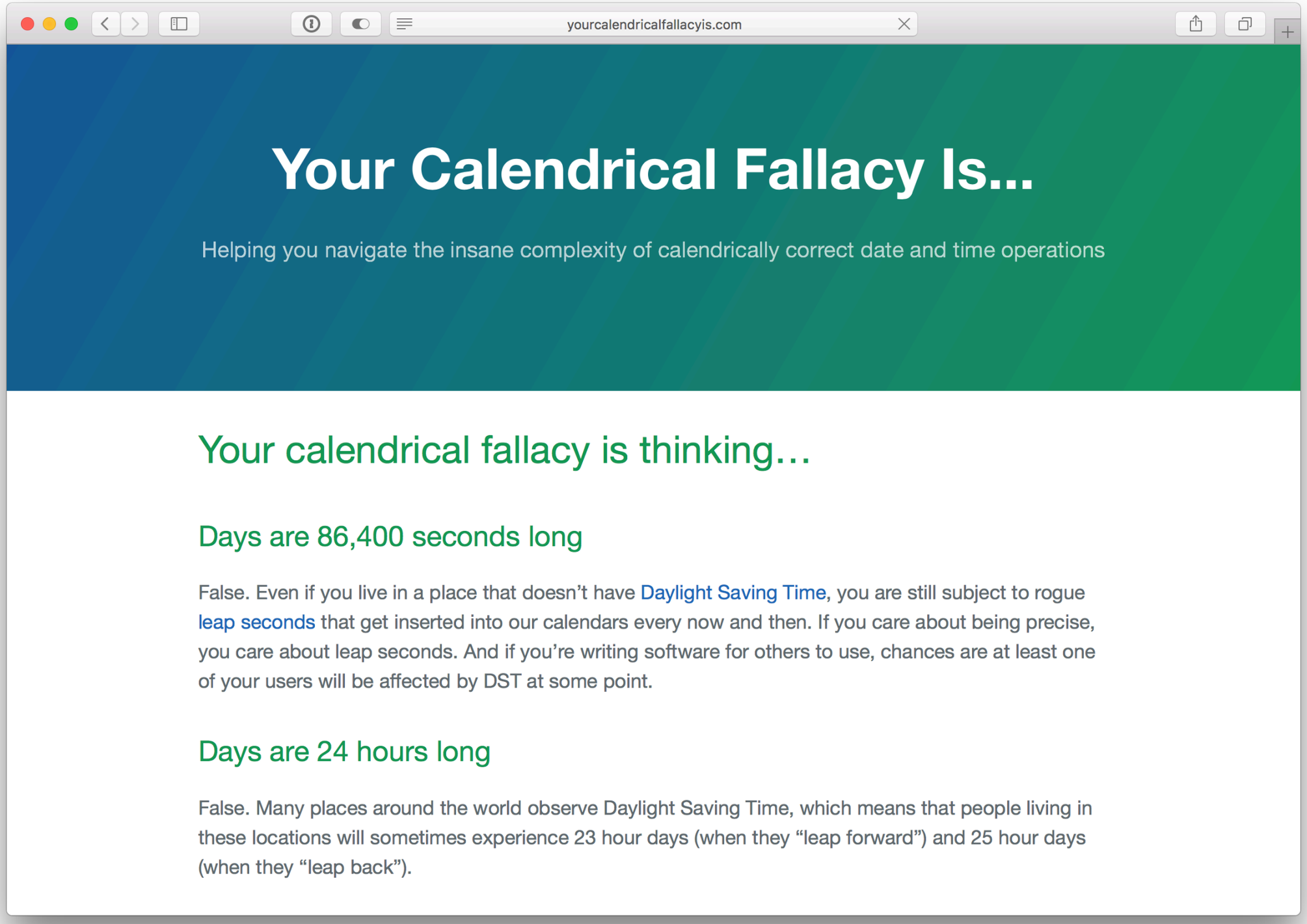
A `Calendar` knows about `TimeZones`, how many months are in a year, etc.

Calendar.Component

```
public enum Component {
    case era
    case year
    case month
    case day
    case hour
    case minute
    case second
    case weekday
    case weekdayOrdinal
    case quarter
    case weekOfMonth
    case weekOfYear
    case yearForWeekOfYear
    case nanosecond
    case calendar
    case timeZone
}
```

Why are dates and times so hard?

Jeff Kelley @SlaunchaMan



Jeff Kelley @SlaunchaMan

Indiana Time Zones

— America/Indiana/ Indianapolis

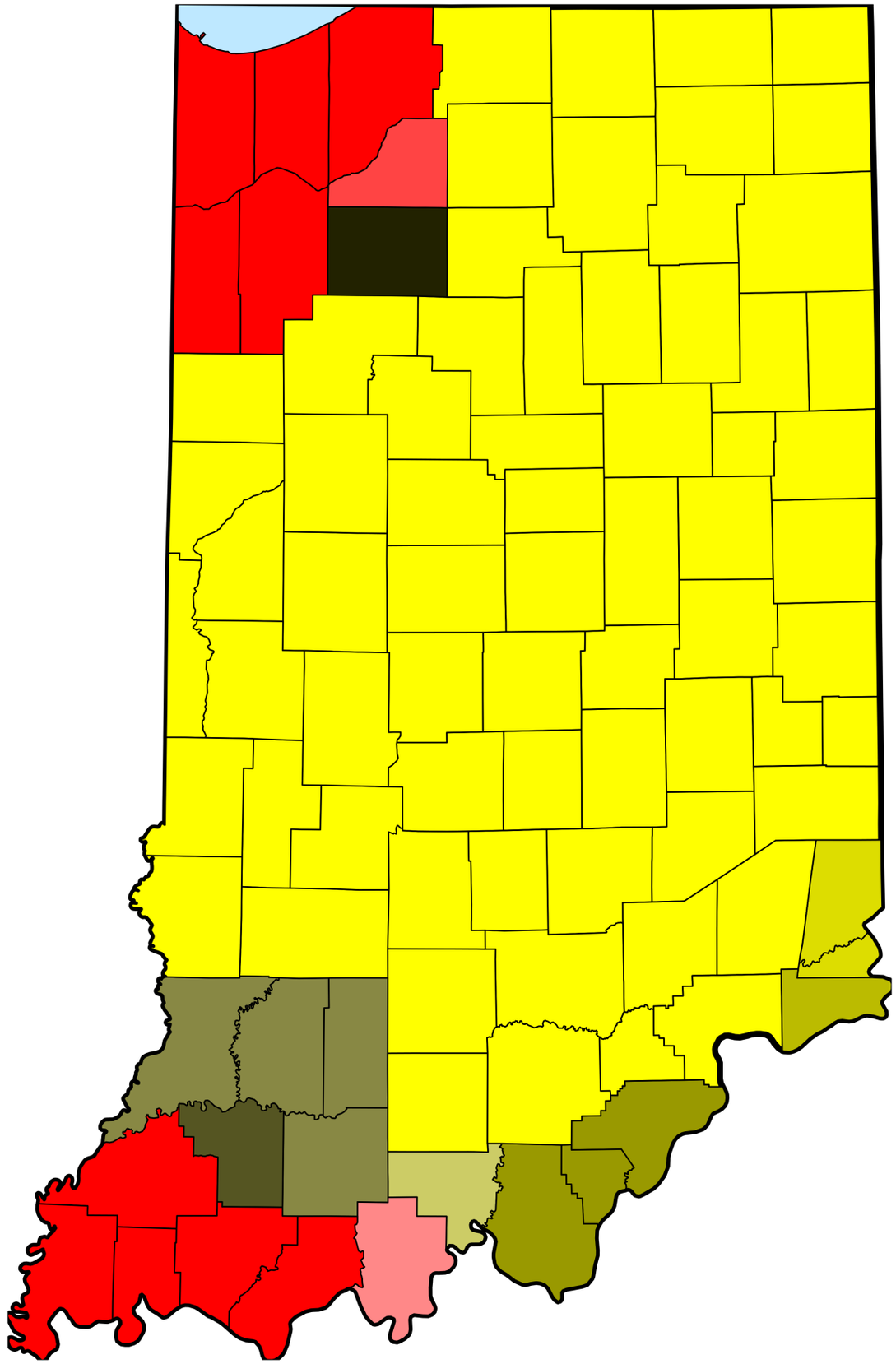
— America/Indiana/ Vincennes

— America/Indiana/ Winamac

— America/Indiana/ Marengo

— America/Indiana/ Petersburg

— America/Indiana/ Vevay



Jeff Kelley @SlaunchaMan

Central Time Coalition

Central Time is the Right Time

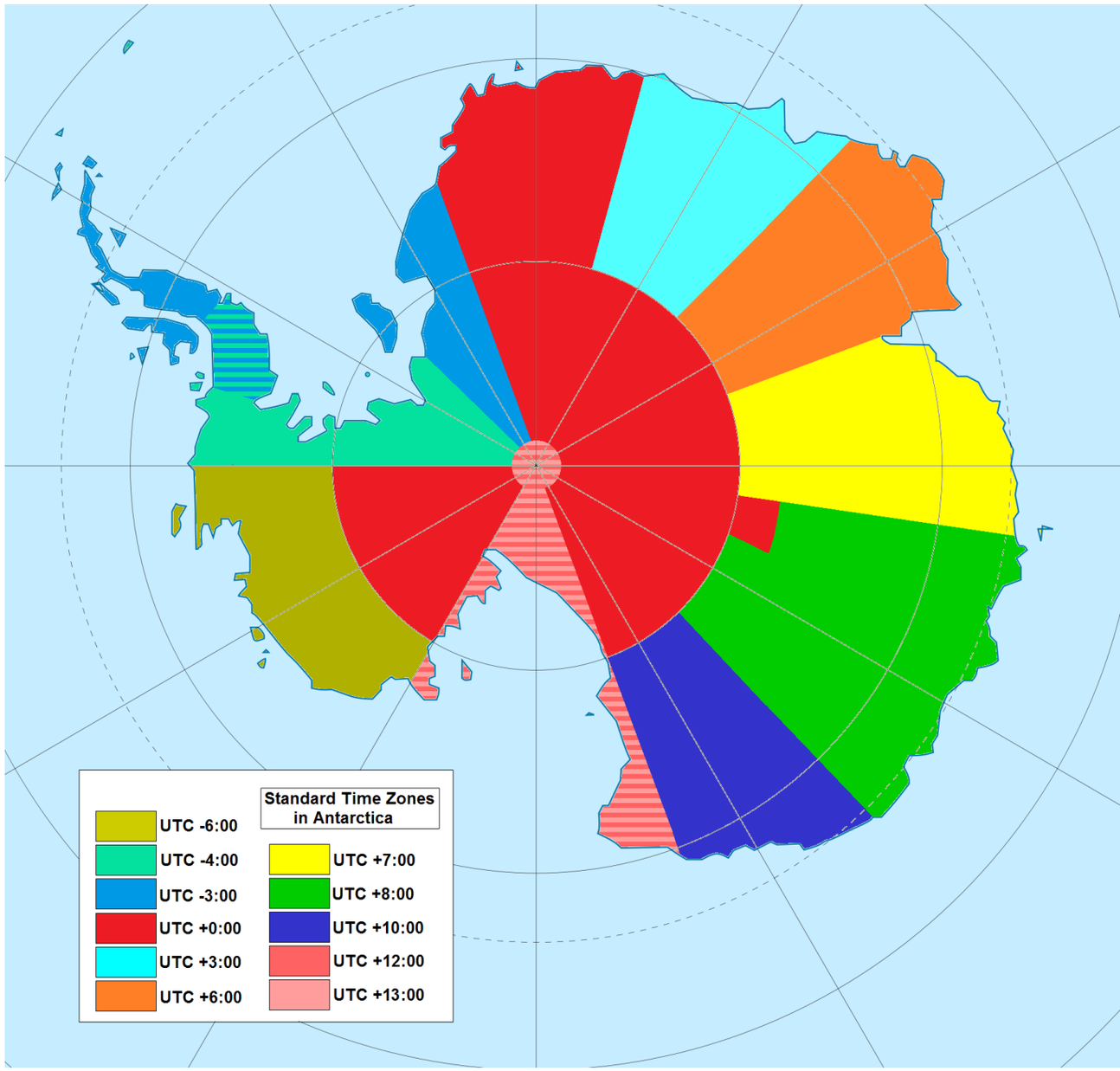
CENTRAL TIME

Indiana's Right Time

Antarctica Time Zones

For the most part, daylight saving time (DST) is not observed ... 95 percent of the continent is located south of the Antarctic Circle and the midnight sun phenomenon renders the use of DST unnecessary ... a few regions ... observe the time and use of DST of the countries they are supplied from.

— Wikipedia, Time in Antarctica



Jeff Kelley @SlaunchaMan

Easter

Let’s write a method to find the month and day for Easter in a given year.

Easter falls on the first Sunday after the full moon following the March equinox—the “paschal full moon.”

Easy, right?

Computus

Expression	$Y = 1961$	$Y = 2018$
$a = Y \bmod 19$	$a = 4$	$a = 4$
$b = Y \operatorname{div} 100$	$b = 19$	$b = 20$
$c = Y \bmod 100$	$c = 61$	$c = 18$
$d = b \operatorname{div} 4$	$d = 4$	$d = 5$
$e = b \bmod 4$	$e = 3$	$e = 0$
$f = (b + 8) \operatorname{div} 25$	$f = 1$	$f = 1$
$g = (b - f + 1) \operatorname{div} 3$	$g = 6$	$g = 6$
$h = (19a + b - d - g + 15) \bmod 30$	$h = 10$	$h = 10$
$i = c \operatorname{div} 4$	$i = 15$	$i = 4$
$k = c \bmod 4$	$k = 1$	$k = 2$
$l = (32 + 2e + 2i - h - k) \bmod 7$	$l = 1$	$l = 0$
$m = (a + 11h + 22l) \operatorname{div} 451$	$m = 0$	$m = 0$
$month = (h + l - 7m + 114) \operatorname{div} 31$	$month = 4$ (April)	$month = 4$ (April)
$day = ((h + l - 7m + 114) \bmod 31) + 1$	$day = 2$	$day = 1$
Gregorian Easter	2 April 1961	1 April 2018

“Anonymous Gregorian Algorithm” from Wikipedia

Jeff Kelley @SlaunchaMan

SwiftyComputus

```
extension DateComponents {
    init(forEasterIn year: Int) {
        self.init()
        self.calendar = Calendar(identifier: .gregorian)

        // Source: http://en.wikipedia.org/wiki/Computus#Anonymous_Gregorian_algorithm
        let a = year % 19;
        let b = year / 100;
        let c = year % 100;
        let d = b / 4;
        let e = b % 4;
        let f = (b + 8) / 25;
        let g = (b - f + 1) / 3;
        let h = ((19 * a) + b - d - g + 15) % 30;
        let i = c / 4;
        let k = c % 4;
        let L = (32 + (2 * e) + (2 * i) - h - k) % 7;
        let m = (a + (11 * h) + (22 * L)) / 451;

        self.month = (h + L - (7 * m) + 114) / 31;
        self.day = ((h + L - (7 * m) + 114) % 31) + 1;
        self.year = year
    }
}
```

Jeff Kelley @SlaunchaMan

SwiftyComputus

```
let easterThisYear = DateComponents(forEasterIn: 2018)

print("Easter this year was on day \(easterThisYear.day!) of month \(easterThisYear.month!)")

// Output: Easter this year was on day 1 of month 4
```

Jeff Kelley @SlaunchaMan

Advanced Dates and Times

Jeff Kelley @SlaunchaMan

DateInterval

```
let calendar = Calendar.autoupdatingCurrent
let timeZone = TimeZone(identifier: "America/New_York")!

let talkStart = DateComponents(
    timeZone: timeZone,
    year: 2018, month: 6, day: 19,
    hour: 13, minute: 00, second: 00)

let talkEnd = DateComponents(
    timeZone: timeZone,
    year: 2018, month: 6, day: 19,
    hour: 13, minute: 45, second: 00)

let startDate = calendar.date(from: talkStart)!
let endDate = calendar.date(from: talkEnd)!

let talkInterval = DateInterval(
    start: startDate,
    end: endDate)
```

DateInterval

```
talkInterval.duration
// 2700

talkInterval.description
// "2018-06-19 17:00:00 +0000 to 2018-06-19 17:45:00 +0000"

talkInterval.contains(Date())
// true
```

DateIntervalFormatter

```
let formatter = DateIntervalFormatter()

formatter.string(from: talkInterval)
// "6/19/18, 1:00 - 1:45 PM"
```

DateIntervalFormatter

```
var prefix = ""

if calendar.isDateInToday(startDate) && calendar.isDateInToday(endDate) {
    formatter.dateStyle = .none
    prefix = "Today "
}
else if calendar.isDateInYesterday(startDate) && calendar.isDateInYesterday(endDate) {
    formatter.dateStyle = .none
    prefix = "Yesterday "
}
else if calendar.isDateInTomorrow(startDate) && calendar.isDateInTomorrow(endDate) {
    formatter.dateStyle = .none
    prefix = "Tomorrow "
}
else {
    formatter.dateStyle = .short
}

prefix + formatter.string(from: talkInterval)!
// "Today 1:00 - 1:45 PM"
```

Converting Between `DateComponents`

```
// How many days are there in this month?
calendar.range(of: .day, in: .month, for: Date())

// Range(1..<32)
```

Converting Between `DateComponents`

```
// How many days are there in this year?
calendar.range(of: .day, in: .year, for: Date())

// Range(1..<366)
```

How Many Seconds Are in a Year?

```
calendar.maximumRange(of: .second)
// Range(0..<60)

calendar.minimumRange(of: .day)
// Range(1..<29)
```

Jeff Kelley @SlaunchaMan

How Many Seconds Are in a Year?

```
// How many seconds are there in this year?
calendar.range(of: .second, in: .year, for: Date())

// Range(0..<60)
```

How Many Seconds Are in a Year?

```
let monthRange = calendar.range(of: .month, in: .year, for: Date())!

for month in monthRange.lowerBound ..< monthRange.upperBound {
    let date = calendar.date(bySetting: .month, value: month, of: Date())!

    let dayRange = calendar.range(of: .day, in: .month, for: date)!

    for day in dayRange.lowerBound ..< dayRange.upperBound {
        let date = calendar.date(bySetting: .day, value: day, of: date)!

        let hourRange = calendar.range(of: .hour, in: .day, for: date)!

        for hour in hourRange.lowerBound ..< hourRange.upperBound {
```


THIS IS GETTING OUT OF HAND
quickmeme.com

Demo

How many seconds are in a year?

Jeff Kelley @SlaunchaMan

When to use `range(of:in:for:)`

- Creating your own date picker
  - But you should probably use Apple's if you can
- Creating a calendar view
  - Find number of months in a year, days in a week, etc.
- Enumerating values, e.g. every day in a month

When not to use `range(of:in:for)`

- Credit Card Expiration Date pickers
  - No matter what, for US cards, the valid months are 01 through 12
  - The actual domain is valid credit card expiration dates

Apple to fix DST alarm bug

Ben Grubb

Were you woken up either early or late this week by your iPhone's alarm app?

If so, Apple plans soon to patch **that bug**. But it's unlikely Apple will give Australian users an IOU for putting some into sleep debt.

As NSW, Victoria, Tasmania, the ACT and South Australia **switched to daylight saving time** at the weekend, a bug in the Apple iPhone caused some workers to run late on Monday.

The bug appeared to affect only those with a "recurring" alarm, which is useful for when you want to be woken up at the same time each and every day of the week.

Online forums were filled with Australians discovering the bug; they have found a simple fix - to set their alarms one hour late. Setting a manual alarm each day - instead of having it occur automatically - **was also said to work**, Macworld said.

Notifications at Specific Times

```
let beginningOfAnyYearComponents = DateComponents(
    month: 1,
    day: 1,
    hour: 0,
    minute: 0,
    second: 0,
    nanosecond: 0)

calendar.nextDate(after: Date(),
                  matching: beginningOfAnyYearComponents,
                  matchingPolicy: Calendar.MatchingPolicy.nextTime)
// "Jan 1, 2018 at 12:00 AM"
```

Notifications at Specific Times

```
let trigger =
    UNCalendarNotificationTrigger(dateMatching:beginningOfAnyYearComponents,
                                  repeats: true)

trigger.nextTriggerDate()
// "Jan 1, 2018 at 12:00 AM"
```

Notifications at Specific Times

```
let content = UNMutableNotificationContent()
content.title = "Happy new year!"

let request = UNNotificationRequest(identifier: "com.slaunchaman.happynewyear",
                                    content: content,
                                    trigger: trigger)

UNUserNotificationCenter.current().add(request) { (error) in
    if let error = error {
        NSLog("Error: \(error.localizedDescription)")
    }
}
```

Searching for `Date`s

The same methods `UNCalendarNotificationTrigger` uses are available to us:

```
let calendar = Calendar.autoupdatingCurrent
let thirteenthComponents = DateComponents(day: 13, weekday: 6)

let next13th = calendar.nextDate(after: Date(),
                                 matching: thirteenthComponents,
                                 matchingPolicy: .nextTime)
// Jul 13, 2018 at 12:00 AM

let the13thAfterThat = calendar.nextDate(after: next13th!,
                                         matching: thirteenthComponents,
                                         matchingPolicy: .nextTime)
// Sep 13, 2019 at 12:00 AM
```

Enumerating Dates

```
func enumerateDates(startingAfter start: Date,
                    matching components: DateComponents,
                    matchingPolicy: Calendar.MatchingPolicy,
                    repeatedTimePolicy: Calendar.RepeatedTimePolicy = default,
                    direction: Calendar.SearchDirection = default,
                    using block: (Date?, Bool, inout Bool) -> Void)
```

Jeff Kelley @SlaunchaMan

Demo

Drop It Like It’s Clock

Jeff Kelley @SlaunchaMan

More Resources

— Date and Time Programming Guide, Apple
— YourCalendricalFallacyIs.com, Dave DeLong
— Chronology, Dave DeLong

Thank You!

Jeff Kelley,
jeff@detroitlabs.com

@SlaunchaMan on Twitter