



BEUTH HOCHSCHULE FÜR TECHNIK BERLIN  
University of Applied Sciences



# MAD7

MapKit

Prof. Dr. Dragan Macos



```
let tenEighty = VideoMode()
tenEighty.resolution = hd
tenEighty.interlaced = true
tenEighty.name = "1080i"
tenEighty.frameRate = 25.0
let alsoTenEighty = tenEighty
alsoTenEighty.frameRate = 30.0
println("The frameRate property of tenEighty is now \
      (tenEighty.frameRate)")
```

```
class VideoMode {
    var resolution = Resolution()
    var interlaced = false
    var frameRate = 0.0
    var name: String?
}
```

Ausgabe??



```
let tenEighty = VideoMode()
tenEighty.resolution = hd
tenEighty.interlaced = true
tenEighty.name = "1080i"
tenEighty.frameRate = 25.0
let alsoTenEighty = tenEighty
alsoTenEighty.frameRate = 30.0
println("The frameRate property of tenEighty is now \
      (tenEighty.frameRate)")
// prints "The frameRate property of tenEighty is now
      30.0"
```

```
class VideoMode {
    var resolution = Resolution()
    var interlaced = false
    var frameRate = 0.0
    var name: String?
}
```



Resolution ist  
eine Struktur

```
let hd = Resolution(width: 1920, height: 1080)
var cinema = hd
cinema.width = 2048
println("cinema is now \(cinema.width) pixels wide")
```

Ausgabe:

```
println("hd is still \(hd.width) pixels wide")
```

Ausgabe:

# Wiederholung



```
struct Point {
    var x = 0.0, y = 0.0
}

struct Size {
    var width = 0.0, height = 0.0
}

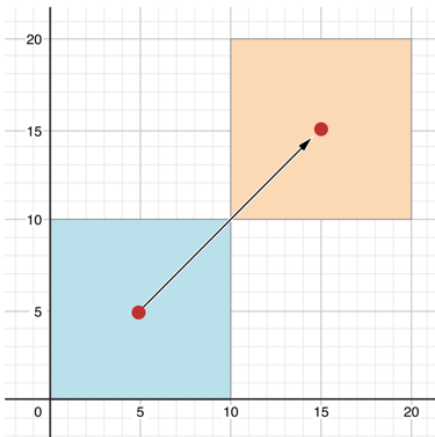
struct Rect {
    var origin = Point()
    var size = Size()
    var center: Point {
        get {
            let centerX = origin.x + (size.width / 2)
            let centerY = origin.y + (size.height / 2)
            return Point(x: centerX, y: centerY)
        }
        set(newCenter) {
            origin.x = newCenter.x - (size.width / 2)
            origin.y = newCenter.y - (size.height / 2)
        }
    }
}

var square = Rect(origin: Point(x: 0.0, y: 0.0),
    size: Size(width: 10.0, height: 10.0))

let initialSquareCenter = square.center
square.center = Point(x: 15.0, y: 15.0)
println("square.origin is now at \(square.origin.x), \
    (square.origin.y)")

// prints "square.origin is now at (10.0, 10.0)"
```

???





???

```
1  class StepCounter {
2      var totalSteps: Int = 0 {
3          willSet(newTotalSteps) {
4              println("About to set totalSteps to \
              (newTotalSteps)")
5          }
6          didSet {
7              if totalSteps > oldValue {
8                  println("Added \((totalSteps - oldValue) steps")
9              }
10         }
11     }
12 }
13 let stepCounter = StepCounter()
14 stepCounter.totalSteps = 200
15 // About to set totalSteps to 200
16 // Added 200 steps
17 stepCounter.totalSteps = 360
18 // About to set totalSteps to 360
19 // Added 160 steps
20 stepCounter.totalSteps = 896
21 // About to set totalSteps to 896
22 // Added 536 steps
```



???

```
struct Point {  
    var x = 0.0, y = 0.0  
    mutating func moveByX(deltaX: Double, y  
        deltaY: Double) {  
        x += deltaX  
        y += deltaY  
    }  
}  
  
var somePoint = Point(x: 1.0, y: 1.0)  
somePoint.moveByX(2.0, y: 3.0)  
println("The point is now at \(somePoint.x),  
    \(somePoint.y)")  
// prints "The point is now at (3.0, 4.0)"
```



```
func locationManager(manager:CLLocationManager,
    didUpdateLocations locations:[AnyObject])
{
    var loc = locations.last as CLLocation
    var latValue = loc.coordinate.latitude
    var lonValue = loc.coordinate.longitude
    println("**** from locations:")
    println("LAT:" + latValue.description + "    LON:" + lonValue.description)

    println("**** from locationManager:")
    latValue = locationManager.location.coordinate.latitude
    lonValue = locationManager.location.coordinate.longitude
    println("LAT:" + latValue.description + "    LON:" + lonValue.description)
}
}
```

Wie binden wir diese Funktion in unsere App ein, so dass sie die GPS-Koordinaten auf dem Display ausgibt??

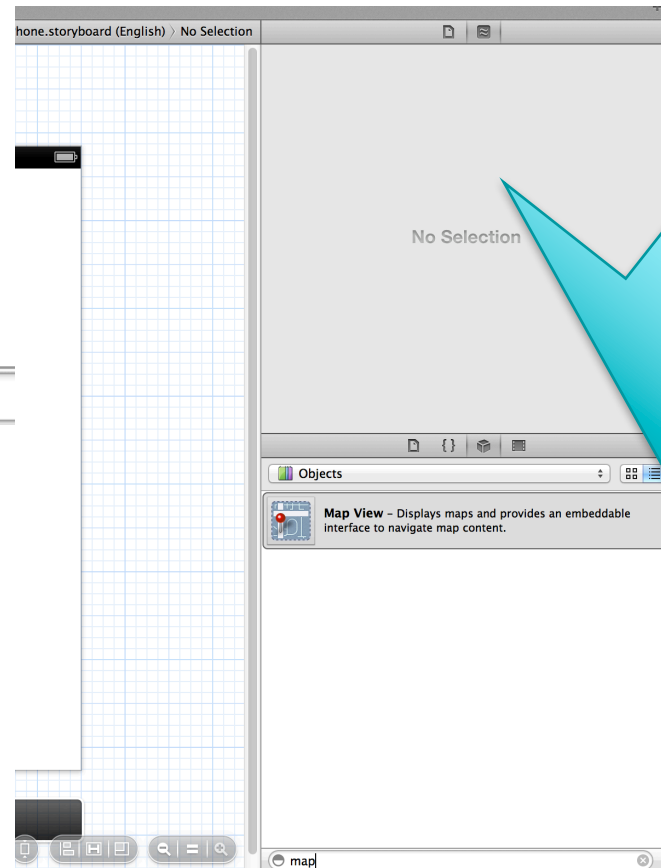
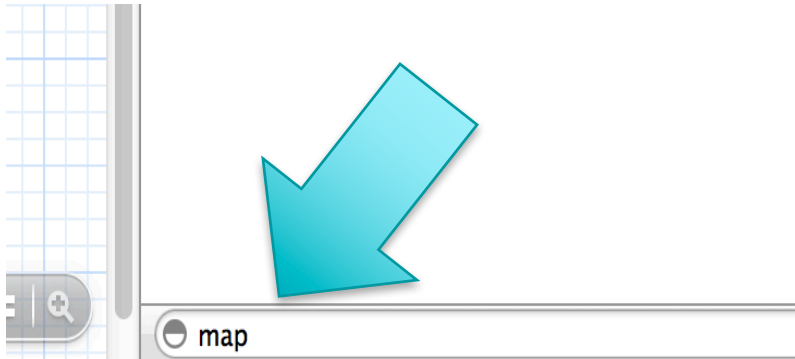


- Framework für die Anzeige von Karten.
- Wichtige Information: Dieses Framework ist grundsätzlich unabhängig von CoreLocation



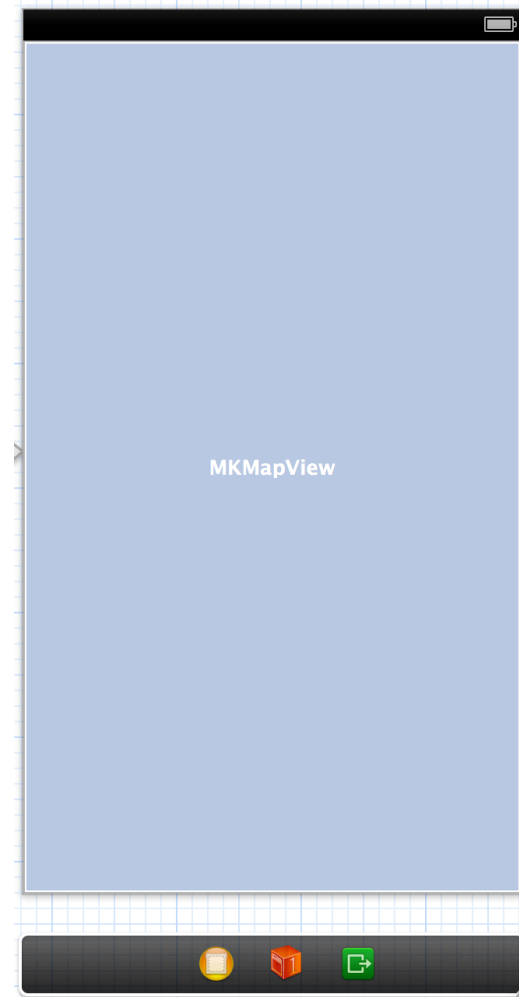


- ... In unserer Objektbibliothek können wir einfach „map“ eintippen:



Der einzige  
Treffer

MkMapView in unserem View.



# .. Outlet deklarieren...



```
10 import MapKit
11 import CoreLocation
12
13
14
15 class ViewController: UIViewController, MKMapViewDelegate{
16
17     @IBOutlet weak var map: MKMapView!
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
```



- Wie zeigen wir die Benutzerposition auf der Karte?
- ... Entwicklerdokumentation





## Displaying the User's Location

`showsUserLocation`

A Boolean value indicating whether the map should try to display the user's location.

### Declaration

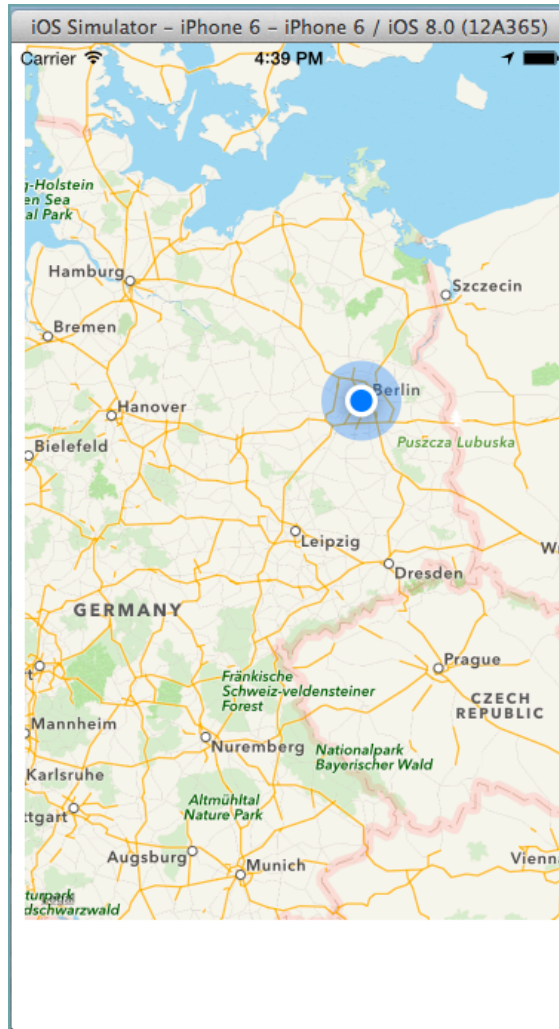
SWIFT

```
var showsUserLocation: Bool
```

Früher war das  
nicht notwendig.  
Früher war  
Mapkit völlig  
unabhängig von  
CoreLocation

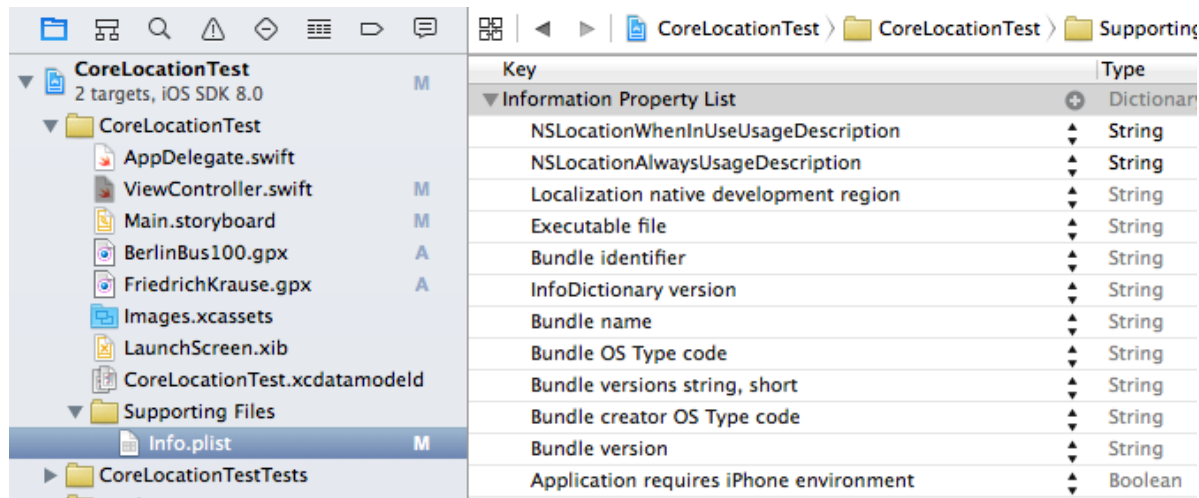
```
override func viewDidLoad() {  
  
    super.viewDidLoad()  
  
    locationManager.requestAlwaysAuthorization()  
  
    map.showsUserLocation = true  
  
}
```

# .. Und die Position wird angezeigt





- Wichtige Einstellung im Xcode:

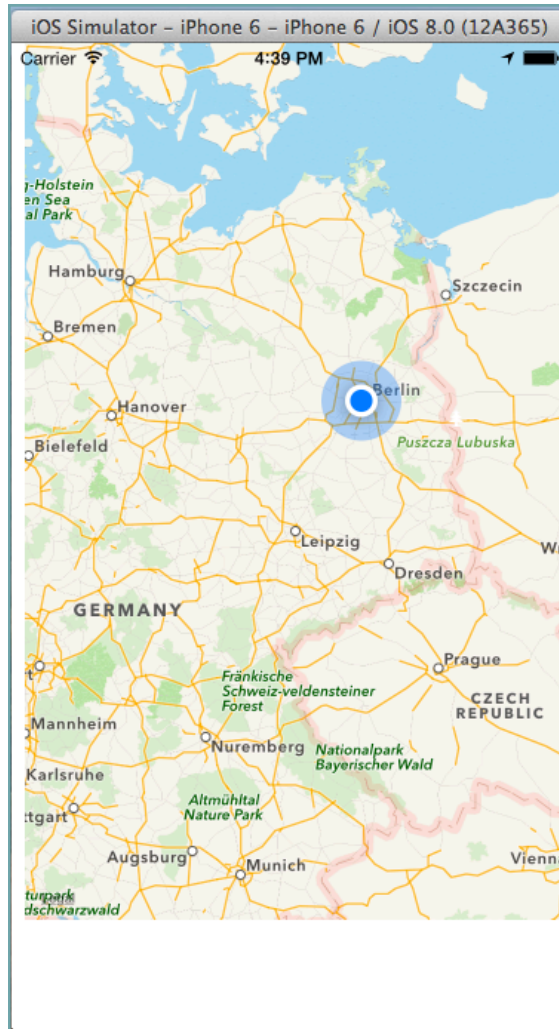


- 3 Parameter wie im MAD 6 eintragen (Sicherheit) ☺





# Wir möchten doch die Karte auf die aktuelle Position zoomen



## Manipulating the Visible Portion of the Map

`region`

The area currently displayed by the map view.

### Declaration

**SWIFT**

```
var region: MKCoordinateRegion
```

**OBJECTIVE-C**

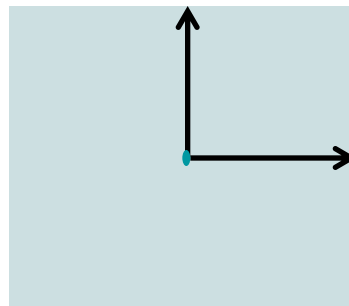
```
@property(nonatomic) MKCoordinateRegion region
```

### Discussion

The region encompasses both the latitude and longitude point on which the map is centered and the span of coordinates to display. The span values provide an implicit zoom value for the map. The larger the displayed area, the lower the amount of zoom.

Property  
von  
MKMapView

Eine Struktur, die aus dem  
Mittelpunkt der zu  
zoomenden Region, dessen  
Höhe und Breite besteht.



# Manipulating the Visible Portion of the Map

## region

The area currently displayed by the map view.

### Declaration

SWIFT

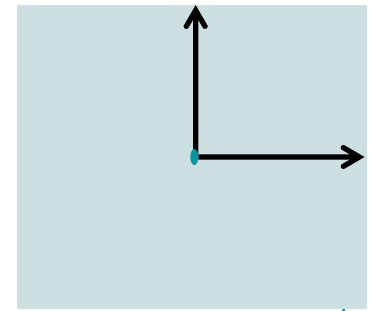
```
var region: MKCoordinateRegion
```

OBJECTIVE-C

```
@property(nonatomic) MKCoordinateRegion region
```

### Discussion

The region encompasses both the latitude and longitude point on which the map is centered and the span of coordinates to display. The span values provide an implicit zoom value for the map. The larger the displayed area, the lower the amount of zoom.



## MKCoordinateRegion

A rectangular geographic region centered around a specific latitude and longitude.

### Language

Swift | [Objective-C](#)

### SDKs

iOS 3.0+  
macOS 10.9+  
tvOS 9.2+  
watchOS 2.0+

### Framework

MapKit

## Topics

### Creating a Region

```
init()
```

```
init(center: CLLocationCoordinate2D, span: MKCoordinateSpan)
```

Creates a new [MKCoordinateRegion](#) from the specified coordinate and span values.

```
init(center: CLLocationCoordinate2D, latitudinalMeters: CLLocationDistance,  
longitudinalMeters: CLLocationDistance)
```

Creates a new [MKCoordinateRegion](#) from the specified coordinate and distance values.

```
init(MKMapRect)
```

Returns the region that corresponds to the specified map rectangle.





So wird die Region „gezoomt“:

```
self.map.region = MKCoordinateRegion(center: currentLocation, latitudinalMeters: 1000,  
    longitudinalMeters: 1000)
```





# MKMapViewDelegate

Inherits from: None  
Conforms to: None



## mapView(\_:didUpdate:)

and later. [More related items...](#)

Tells the delegate that the location of the user was updated.

## Declaration

```
optional func mapView(_ mapView: MKMapView, didUpdate userLocation:
```

## Parameters

<b>mapView</b>	The map view that is tracking the user's location.
<b>userLocation</b>	The location object representing the user's latest location. This property may be <code>nil</code> .

## Discussion

While the `showsUserLocation` property is set to `true`, this method is called whenever a new location update is received by the map view. This method is also called if the map view's user tracking mode is set to `followWithHeading` and the heading changes.

This method is not called if the application is currently running in the background. If you want to receive location updates while running in the background, you must use the Core Location framework.

```
self.map.region = MKCoordinateRegion(center: currentLocation, latitudinalMeters: 1000,  
    longitudinalMeters: 1000)
```



# MKMapViewDelegate

Inherits from: None

Conforms to: None



## mapView(\_:didUpdate:)

and later. [More related items...](#)

Tells the delegate that the location of the user was updated.

### Declaration

```
optional func mapView(_ mapView: MKMapView, didUpdate userLocation:
```

### Parameters

<b>mapView</b>	The map view that is tracking the user's location.
<b>userLocation</b>	The location object representing the user's latest location. This property may be <code>nil</code> .

### Discussion

While the `showsUserLocation` property is set to `true`, this method is called whenever a new location update is received by the map view. This method is also called if the map view's user tracking mode is set to `followWithHeading` and the heading changes.

This method is not called if the application is currently running in the background. If you want to receive location updates while running in the background, you must use the Core Location framework.

```
self.map.region = MKCoordinateRegion(center: currentLocation, latitudinalMeters: 1000,  
longitudinalMeters: 1000)
```



```
...  
map.delegate = self  
...  
func mapView(_ mapView: MKMapView, didUpdate userLocation: MKUserLocation) {  
    let currentLocation = userLocation.coordinate  
    self.map.region = MKCoordinateRegion(center: currentLocation, latitudinalMeters: 1000,  
        longitudinalMeters: 1000)  
}
```

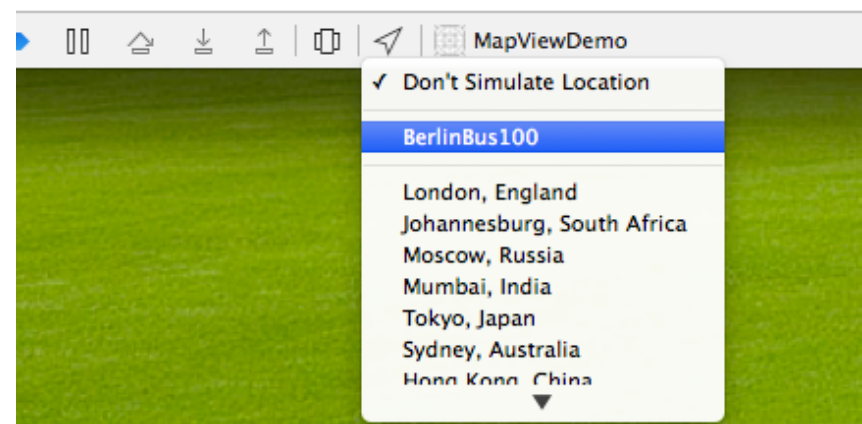
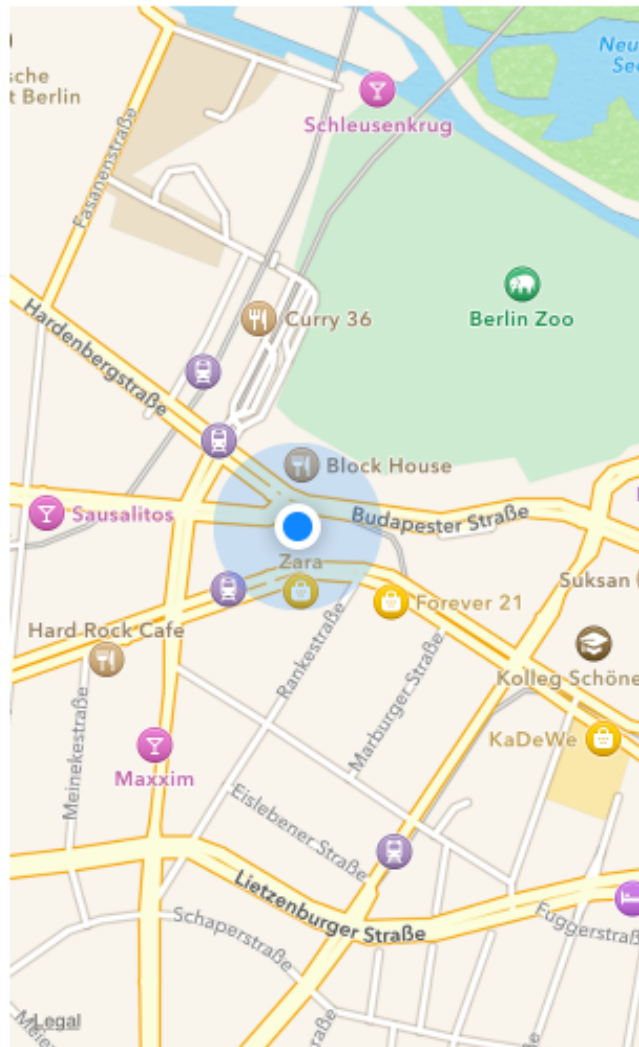




iOS Simulator – iPhone 5s – iPhone 5s / iOS 8...

Carrier

5:32 PM







Wir haben Folgendes geschafft

- Weltkarte anzeigen
- Die Weltkarte wird auf 1km x 1km um die Benutzerposition gezoomt....Immer, wenn die Benutzerposition auf der Karte sich geändert hat.
- Effizient, elegant ....



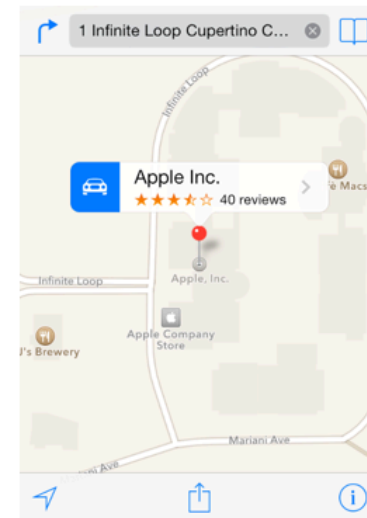


- Die POIs auf der Karte als Pins anzeigen.
- Annotations

## Adding Annotations to a Map

Annotations offer a way to highlight specific coordinates on a map. Annotations typically have some sort of image to identify the location and provides a callout bubble that displays additional information.

**Figure 6-1** Displaying an annotation in a map





1. Annotations-Objekt definieren.
1. Annotationsobjekt der Karte zuordnen.  
In diesem Objekt sind die Koordinaten der Annotation vorhanden.
3. Annotations-View definieren. Er ist für die Anzeige der Annotation auf der Karte zuständig.
4. Im delegate vom map-View soll die Methode `mapView:viewForAnnotation:` umgesetzt werden.  
Diese Methode wird aufgerufen, wenn im Sichtbaren Kartenbereich anzuzeigenden Annotationen vorhanden sind (Holywood) . Wir müssen dem Framework sagen, wie die Annotationen anzuzeigen sind.



# 1. Annotationsobjekt definieren



- Irgendein Objekt, das das Protokoll MKAnnotation realisiert

## MKAnnotation

The `MKAnnotation` protocol is used to provide annotation-related information to a map view. To use this protocol, you adopt it in any custom objects that store or represent annotation data. Each object then serves as the source of information about a single map annotation and provides critical information, such as the annotation's location on the map. Annotation objects do not provide the visual representation of the annotation but typically coordinate (in conjunction with the map view's delegate) the creation of an appropriate `MKAnnotationView` object to handle the display. [More...](#)

Inheritance  
Not Applicable

Conforms To  
`NSObjectProtocol`

Import Statement  
`import MapKit`

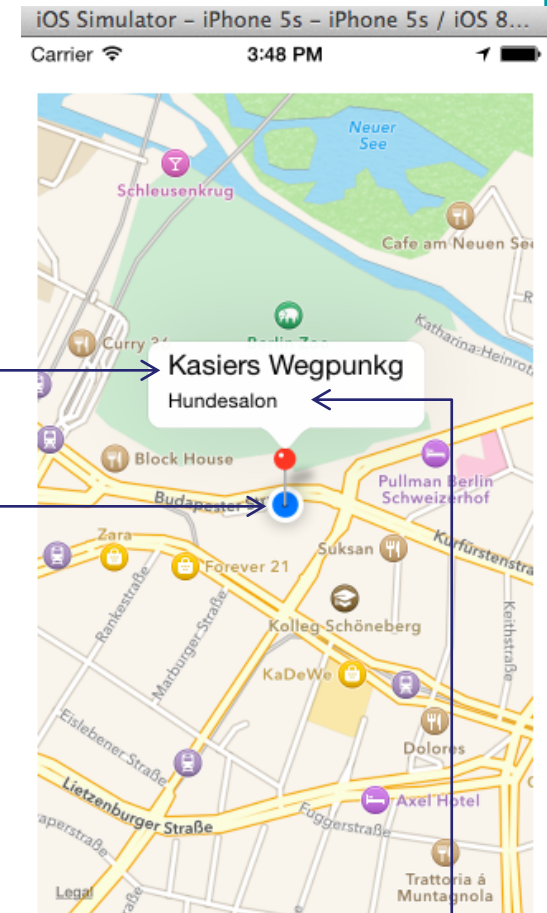
Availability  
Available in OS X v10.9 and later.

### Position Attributes

`coordinate`  
`setCoordinate(_:)`

### Title Attributes

`title`  
`subtitle`



In die eigene Klasse für Annotationen kann man beliebig viele notwendigen Informationen speichern



- Hält das Protokoll MKAnnotation ein.
- Hat nur das notwendige, was man für einfache Annotations braucht.
- Das ist schon eine fertige Annotationsklasse für einfache Annotationen
- Als unsere Annotationsklasse definieren wir eine Unterklasse von MKPointAnnotation mit einer minimalen Erweiterung.



# 1. Annotationsobjekt definieren



- Die Klasse MKPointAnnotation hält das Protokoll MKAnnotation ein.
- Hat nur das notwendige, was man für einfache Annotations braucht.
- Als unsere Annotationsklasse definieren wir eine Unterklasse von MKPointAnnotation mit einer minimalen Erweiterung.

```
import Foundation
import MapKit
```

Unterklasse von  
MKAnnotation.

```
class KaisersWegpunkt: MKPointAnnotation
{
    let nameDerAnnotation = "Kaiser"
}
```

## 2. Annotationsobjekt der Karte zuordnen



```
var poi = KaisersWegpunkt()
```

Property des ViewControllers

```
self.poi.coordinate = CLLocationCoordinate2DMake(52.50, 13.33)
self.poi.title = "Kasiers Wegpungk"
self.poi.subtitle = "Hundesalon"
map.addAnnotation(poi)
```

In ViewDidLoad

```
import Foundation
import MapKit

class KaisersWegpunkt: MKPointAnnotation
{
    let nameDerAnnotation = "Kaiser"
}
```



- Wir werden das Annotationsview im mapView:viewForAnnotation: definieren und gleichzeitig die restliche Implementierung dieser Methoden zeigen.





### 3. Annotations-View definieren

### 4. mapView:viewForAnnotation: realisieren



```
func mapView(mapView: MKMapView!,
             viewForAnnotation annotation: MKAnnotation!)
    -> MKAnnotationView!
{
    if (!(annotation is KaisersWegpunkt) ||
        ((annotation as KaisersWegpunkt).nameDerAnnotation != "Kaiser"))
    {
        return nil
    }
    let reuseId = "Kaiser"
    var annotationView = mapView.dequeueReusableAnnotationViewWithIdentifier(reuseId)
    if annotationView == nil
    {
        annotationView = MKPinAnnotationView(annotation: annotation,
                                             reuseIdentifier: reuseId)
        annotationView.canShowCallout = true
    }
    else
    {
        annotationView.annotation = annotation
    }

    return annotationView
}
```

### 3. Annotations-View definieren

### 4. `mapView:viewForAnnotation:` realisieren



```
func mapView(MKMapView!,
             viewForAnnotation annotation: MKAnnotation!)
    -> MKAnnotationView!
```

Das Kartenobjekt

Die Methode wird aufgerufen,  
wenn eine Annotation  
angezeigt werden soll.

Sie liefert das View für die  
Annotation.

Das angeforderte View ist ein  
View für diese Annotation.

```
{
    if annotation is MKPointAnnotation {
        return nil
    }
    let reuseId = "Kaiser"
    var annotationView = mapView.dequeueReusableAnnotationViewWithIdentifier(reuseId)
    if annotationView == nil
    {
        annotationView = MKPinAnnotationView(annotation: annotation,
                                              reuseIdentifier: reuseId)
        annotationView.canShowCallout = true
    }
    else
    {
        annotationView.annotation = annotation
    }

    return annotationView
}
```

### 3. Annotations-View definieren

### 4. mapView:viewForAnnotation: realisieren



```
func mapView(MKMapView!,
             viewForAnnotation annotation: MKAnnotation
             -> MKAnnotationView!
{
    if (!(annotation is KaisersWegpunkt) ||
        ((annotation as KaisersWegpunkt).nameDerAnnotation != "Kaiser"))
    {
        return nil
    }
    let reuseId = "Kaiser"
    var annotationView = mapView.dequeueReusableAnnotationViewWithIdentifier(reuseId)
    if annotationView == nil
    {
        annotationView = MKPinAnnotationView(annotation: annotation,
                                              reuseIdentifier: reuseId)
        annotationView.canShowCallout = true
    }
    else
    {
        annotationView.annotation = annotation
    }

    return annotationView
}
```

Wir möchten sicher sein, dass die Annotation „unsere“ Annotation ist.

Um nicht immer die Annotationsviews neu zu erzeugen, kann man diese in einer Queue speichern.

Die Annotation wird dem neuen View zugeordnet.

### 3. Annotations-View definieren

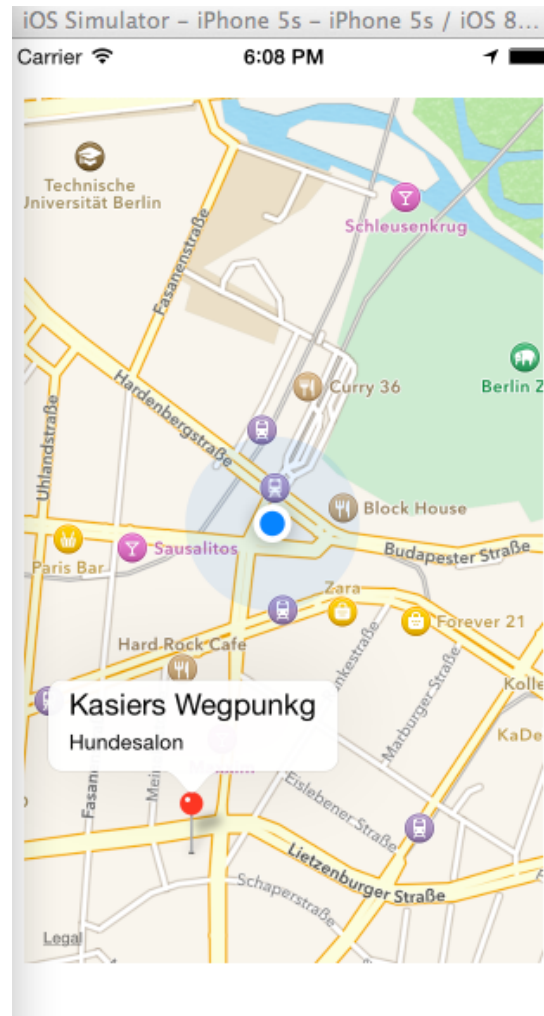
### 4. mapView:viewForAnnotation: realisieren



```
func mapView(mapView: MKMapView!,
             viewForAnnotation annotation: MKAnnotation!)
    -> MKAnnotationView!
{
    if (!(annotation is KaisersWegpunkt) ||
        ((annotation as KaisersWegpunkt).nameDerAnnotation != "Kaiser"))
    {
        return nil
    }
    let reuseId = "Kaiser"
    var annotationView = mapView.dequeueReusableAnnotationViewWithIdentifier(reuseId)
    if annotationView == nil
    {
        annotationView = MKPinAnnotationView(annotation: annotation,
                                             reuseIdentifier: reuseId)
        annotationView.canShowCallout = true
    }
    else
    {
        annotationView.annotation = annotation
    }

    return annotationView
}
```

Die Annotation wird beim berühren  
„aufgeklappt“



Die meisten Sourcecode-Beispiele und die Sprachdefinition der Sprache Swift wurden aus:

Apple Inc. „The Swift Programming Language.“ iBooks. <https://itun.es/de/jEUH0.I>

genommen.

Eventuelle andere Quellen bzw. eigene Beispiele werden an den entsprechenden Stellen direkt angegeben bzw. gekennzeichnet.