BEUTH HOCHSCHULE FÜR TECHNIK BERLIN
University of Applied Sciences

MAD7

MapKit

Prof. Dr. Dragan Macos

Wiederholung

```
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??

Wiederholung

Resolution ist eine Struktur

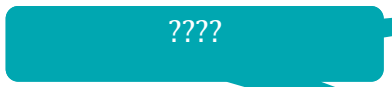
```
let hd = Resolution(width: 1920, height: 1080)
var cinema = hd
cinema.width = 2048
println("cinema is now \(cinema.width) pixels wide")
```

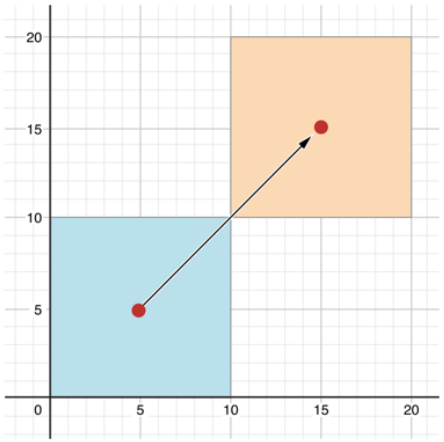
Ausgabe??

```
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)"
```

Wiederholung


```
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
```

Wiederholung

???

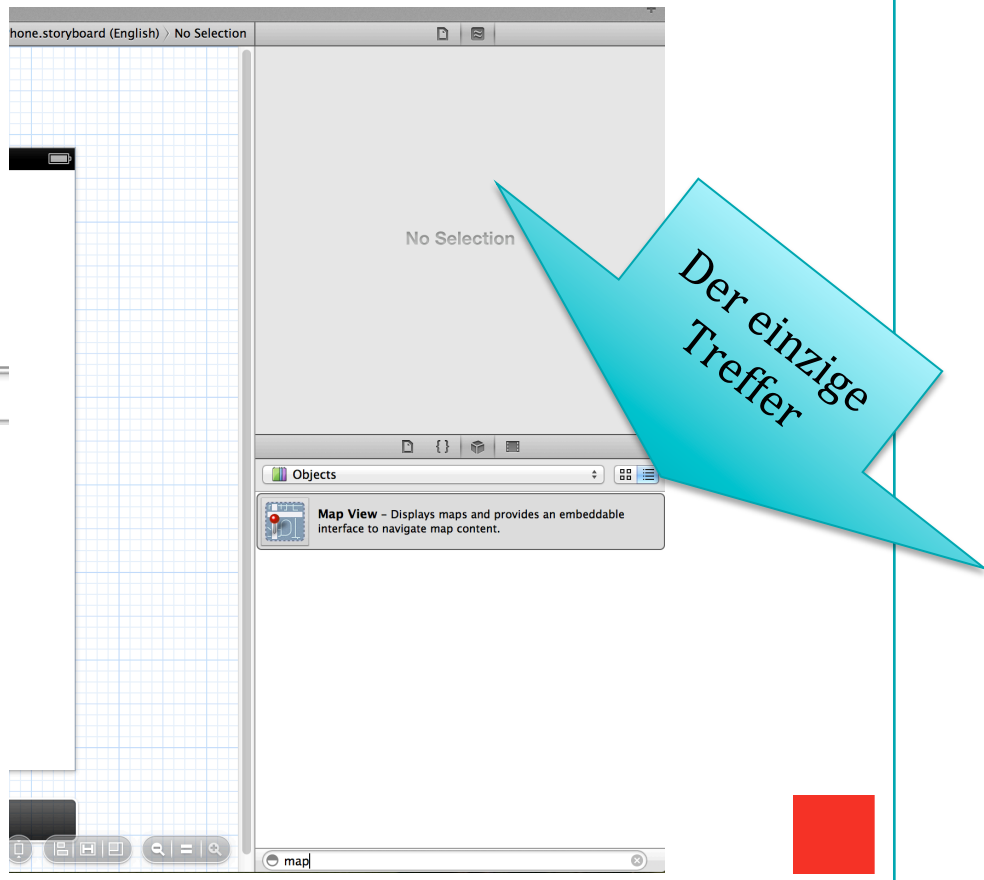
```
struct Point {
    var x = 0.0, y = 0.0
    mutating func moveByX(deltaX: Double, y
        deltaY: Double) {
        x += deltaX
        y += deltaY
    }
}
var somePoint = Point(x: 1.0, y: 1.0)
somePoint.moveByX(2.0, y: 3.0)
println("The point is now at (\(somePoint.x),
        \(somePoint.y))")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

MapKit

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

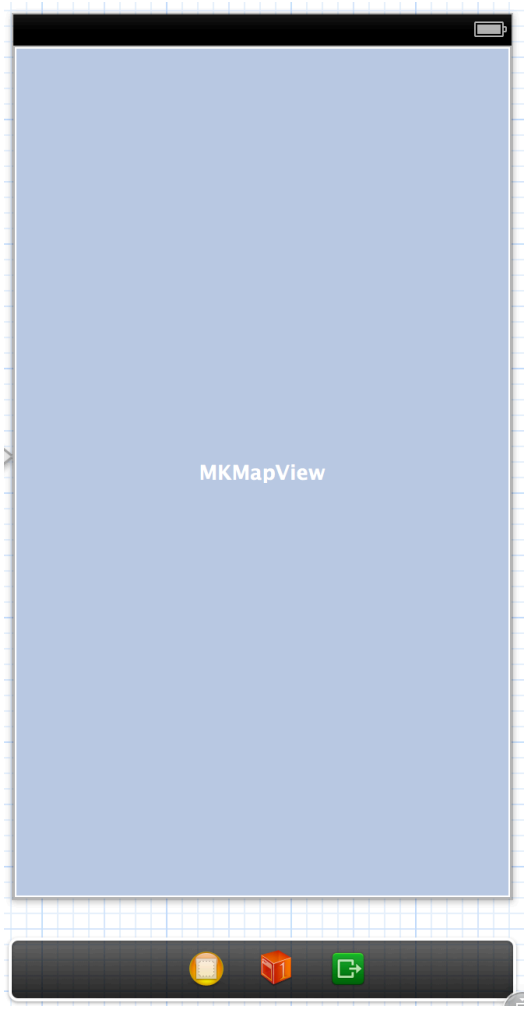
MapView im Storyboard

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



MapKit

MkMapView in unserem View.



Beuth Hochschule für Technik Berlin – University of Applied Sciences

.. 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
```

.. Nächstes Problem

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

MKMapView, Klassenreferenz

Displaying the User's Location

`showsUserLocation`

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

Declaration

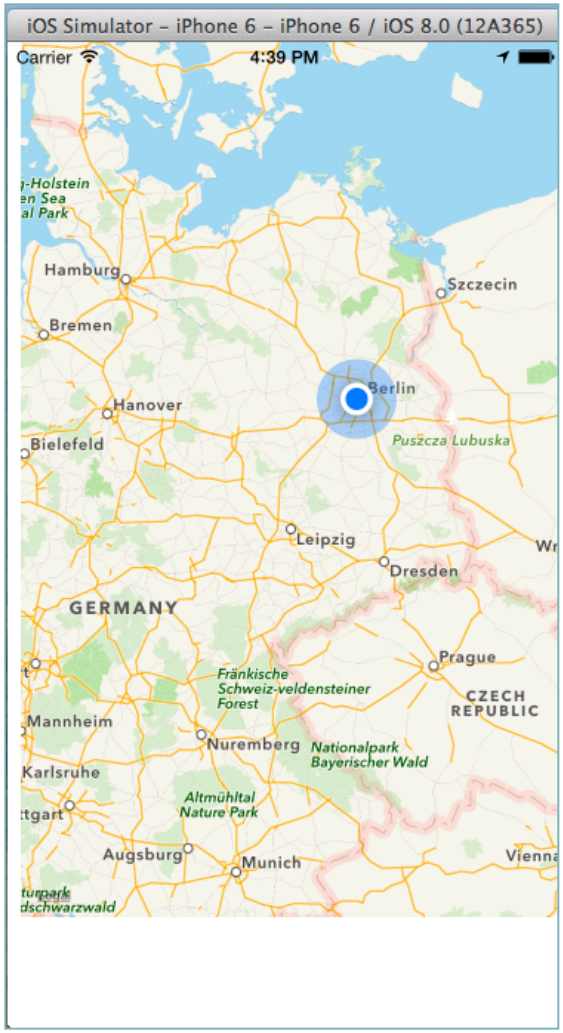
SWIFT

```
var showsUserLocation: Bool
```

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

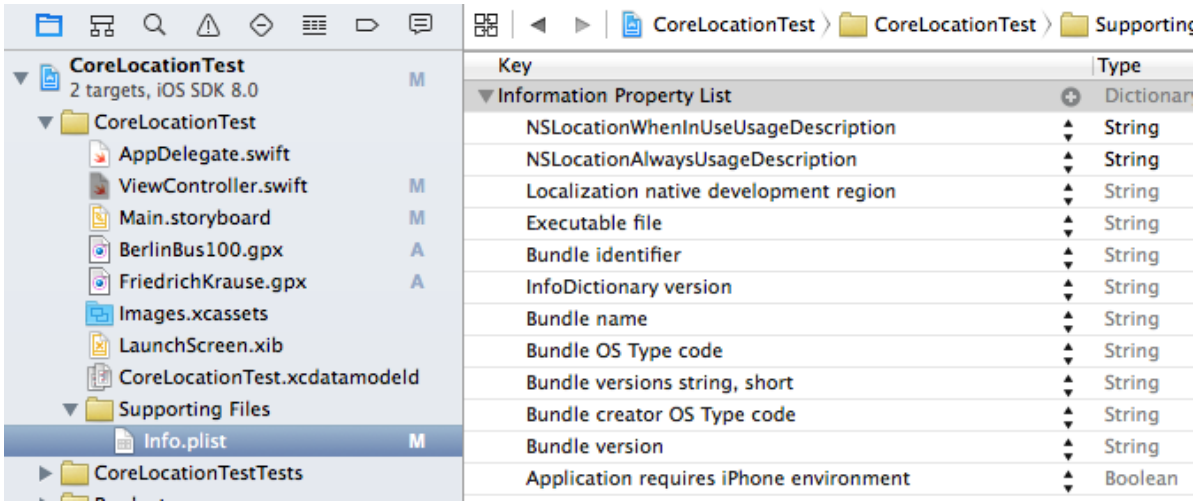
```
override func viewDidLoad() {
    super.viewDidLoad()


    super.viewDidLoad()

    locationManager.requestAlwaysAuthorization()


    map.showsUserLocation = true

}
```

.. Und die Position wird angezeigt



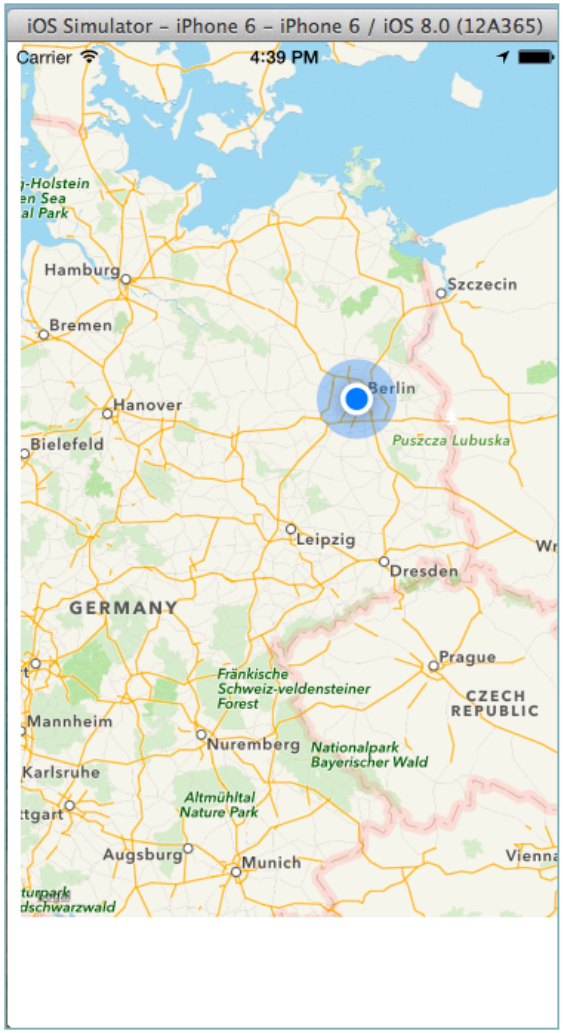
...Noch 2 Sachen

- Wichtige Einstellung im xCode:



- 2 Parameter eintragen (Sicherheit) ☺
  - NSLocationWhenInUseUsageDescription und
  - NSLocationAlwaysUsageDescription

Wir möchten doch die Karte auf die aktuelle Position zoomen



region

Manipulating the Visible Portion of the Map

`region`

The area currently displayed by the map view.

Declaration

SWIFT

```
var region: MKCoordinateRegion
```

OBJECTIVE-C

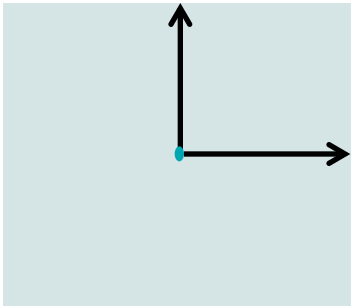
```
@property(nonatomic) MKCoordinateRegion region
```

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.

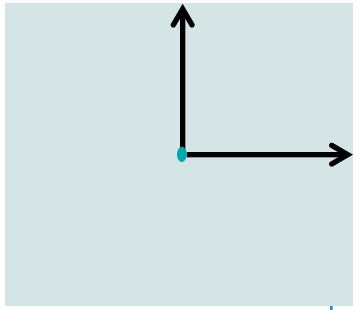
Declaration

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.



Declaration

SWIFT
```
func MKCoordinateRegionMakeWithDistance(_ centerCoordinate: CLLocationCoordinate2D,
                                        _ latitudinalMeters: CLLocationDistance,
                                        _ longitudinalMeters: CLLocationDistance) -> MKCoordinateRegion
```

OBJECTIVE-C
```
MKCoordinateRegion MKCoordinateRegionMakeWithDistance ( CLLocationCoordinate2D centerCoordinate,
CLLocationDistance latitudinalMeters, CLLocationDistance longitudinalMeters );
```

Parameters

centerCoordinate	The center point of the new coordinate region.
latitudinalMeters	The amount of north-to-south distance (measured in meters) to use for the span.
longitudinalMeters	The amount of east-to-west distance (measured in meters) to use for the span.

Return Value

A region with the specified values.

Import Statement

```
import MapKit
```

In der Doku gefunden...

Declaration

SWIFT

```
func MKCoordinateRegionMakeWithDistance(_ centerCoordinate: CLLocationCoordinate2D,
                                        _ latitudinalMeters: CLLocationDistance,
                                        _ longitudinalMeters: CLLocationDistance) -> MKCoordinateRegion
```

OBJECTIVE-C

```
MKCoordinateRegion MKCoordinateRegionMakeWithDistance ( CLLocationCoordinate2D centerCoordinate,
CLLocationDistance latitudinalMeters, CLLocationDistance longitudinalMeters );
```

Parameters

centerCoordinate	The center point of the new coordinate region.
latitudinalMeters	The amount of north-to-south distance (measured in meters) to use for the span.
longitudinalMeters	The amount of east-to-west distance (measured in meters) to use for the span.

Return Value

A region with the specified values.

Import Statement

```
import MapKit
```

In der Doku gefunden...

Wo soll das hin??
Diskussion

Wie kriegen wir das her??

```
self.map.region  = MKCoordinateRegionMakeWithDistance(curentLocation, 1000, 1000)
```

MKMapViewDelegate

Inherits from: None
Conforms to: None
Framework: MapKit in iOS 3.0 and later. More related items...

- mapView:didUpdateUserLocation:

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

Declaration

SWIFT

```
optional func mapView(_ mapView: MKMapView!,
didUpdateUserLocation userLocation: MKUserLocation!)
```

OBJECTIVE-C

```
- (void)mapView:(MKMapView *)mapView
didUpdateUserLocation:(MKUserLocation *)userLocation
```

Parameters

mapView	The map view that is tracking the user's location.
userLocation	The location object representing the user's latest location. This property may be `nil`.

Discussion

While the `showsUserLocation` property is set to `YES`, 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 `MKUserTrackingModeFollowWithHeading` 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  = MKCoordinateRegionMakeWithDistance(curentLocation, 1000, 1000)
```

MKMapViewDelegate

Inherits from: None
Conforms to: None
Framework: MapKit in iOS 3.0 and later. More related items...

- mapView:didUpdateUserLocation:

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

Declaration

SWIFT

```
optional func mapView(_ mapView: MKMapView!,
didUpdateUserLocation userLocation: MKUserLocation!)
```

OBJECTIVE-C

```
- (void)mapView:(MKMapView *)mapView
didUpdateUserLocation:(MKUserLocation *)userLocation
```

Parameters

mapView	The map view that is tracking the user's location.
userLocation	The location object representing the user's latest location. This property may be `nil`.

Discussion

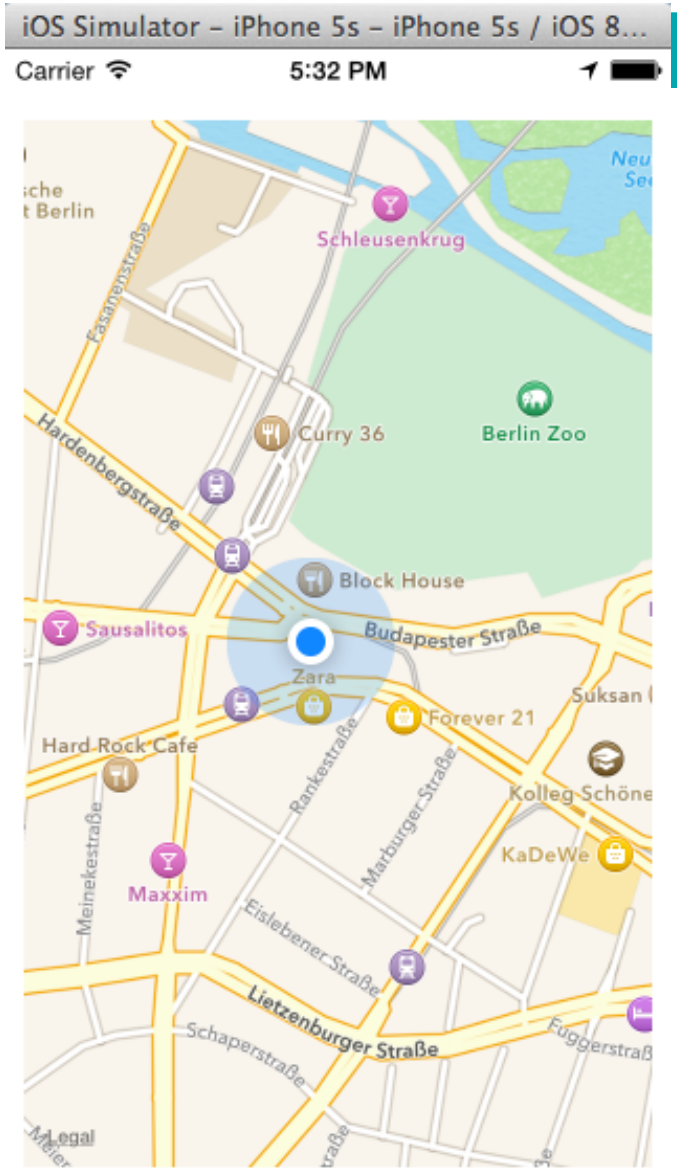
While the `showsUserLocation` property is set to `YES`, this method is called whenever a new location update is received by the map view. This method is also called if the map view's user tracking mode is set to `MKUserTrackingModeFollowWithHeading` and the heading changes.

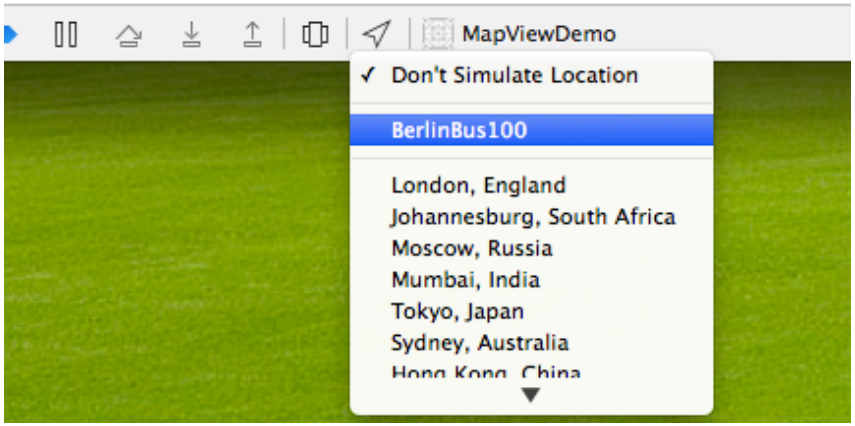
This method is not called if the application is currently running in the background. If you want to receive location updates while running in the background, you must use the Core Location framework.

```
self.map.region  = MKCoordinateRegionMakeWithDistance(curentLocation, 1000, 1000)
```

Die ganze Lösung

```
import UIKit
import MapKit
import CoreLocation



class ViewController: UIViewController, MKMapViewDelegate{

    @IBOutlet weak var map: MKMapView!
    var locationManager = CLLocationManager()

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

    override func didReceiveMemoryWarning() {
        super.didReceiveMemoryWarning()
        // Dispose of any resources that can be recreated.
    }

    func mapView(mapView: MKMapView!, didUpdateUserLocation userLocation: MKUserLocation!) {
        let curentLocation = userLocation.location.coordinate
        self.map.region  = MKCoordinateRegionMakeWithDistance(curentLocation, 1000, 1000)

    }
}
```


iOS Simulator – iPhone 5s – iPhone 5s / iOS 8...
Carrier
5:32 PM
Schleusenkrug
Curry 36
Berlin Zoo
Hardenbergstraße
Block House
Sausalitos
Budapester Straße
Zara
Forever 21
Suksan
Hard Rock Cafe
Rankestraße
Marburger Straße
Kolleg Schöne
KaDeWe
Maxxim
Meinekestraße
Eislebener Straße
Lietzenburger Straße
Schaperstraße
Legal


MapViewDemo
Don't Simulate Location
BerlinBus100
London, England
Johannesburg, South Africa
Moscow, Russia
Mumbai, India
Tokyo, Japan
Sydney, Australia

Review

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 ....

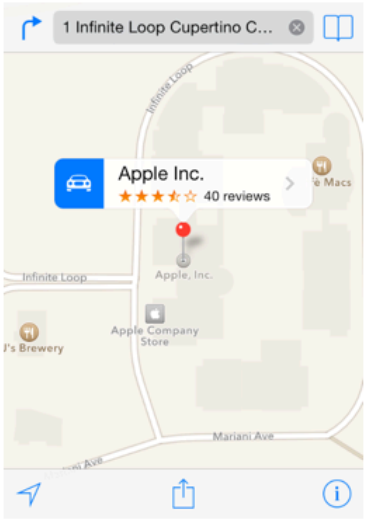
Wir möchten noch etwas..

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

Adding Annotations to a Map

Annotations offer a way to highlight specific coordinates
annotations typically have some sort of image to identify
and provides a callout bubble that displays additional info

Figure 6-1 Displaying an annotation in a map



4 Schritte bei der Anzeige eines POI

1. Annotations-Objekt definieren.

2. 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

Language: Swift Obj-C Both On This Page Options

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	Conforms To	Import Statement
Not Applicable	`NSObjectProtocol`	`import MapKit`

Availability

Available in OS X v10.9 and later.

Position Attributes

`coordinate`

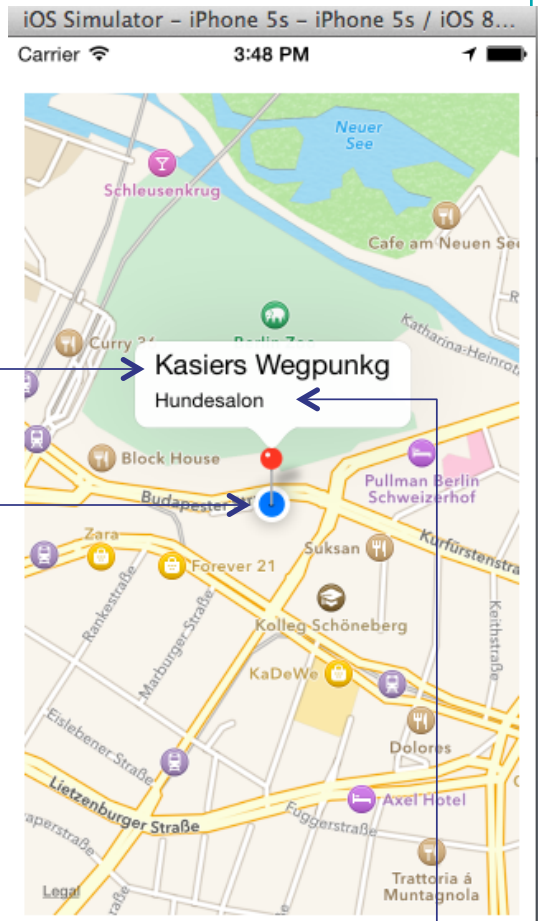
`setCoordinate(_:)`

Title Attributes

`title`

`subtitle`

Feedback



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

Die Klasse MKPointAnnotation

- 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 Erwieterung.

```
import Foundation
import MapKit

class KaisersWegpunkt: MKPointAnnotation
{

    let nameDerAnnotation = "Kaiser"
}
```

Hält das Protokoll MKAnnotation ein.

2. Annotationsobjekt der Karte zuordnen

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

Property des ViewControllers

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

In VieiwDidLoad

3. Annotations-View definieren
4. mapView:viewForAnnotation: realisieren

- Wir werden das Annotationsview im mapView:viewForAnnotation: defieniren 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(mapView: MKMapView!,
             viewForAnnotation annotation: MKAnnotation!)
             -> MKAnnotationView!
{
    if annotation is Kaiser || annotation !=
        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
}
```

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.

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
}
```

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

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

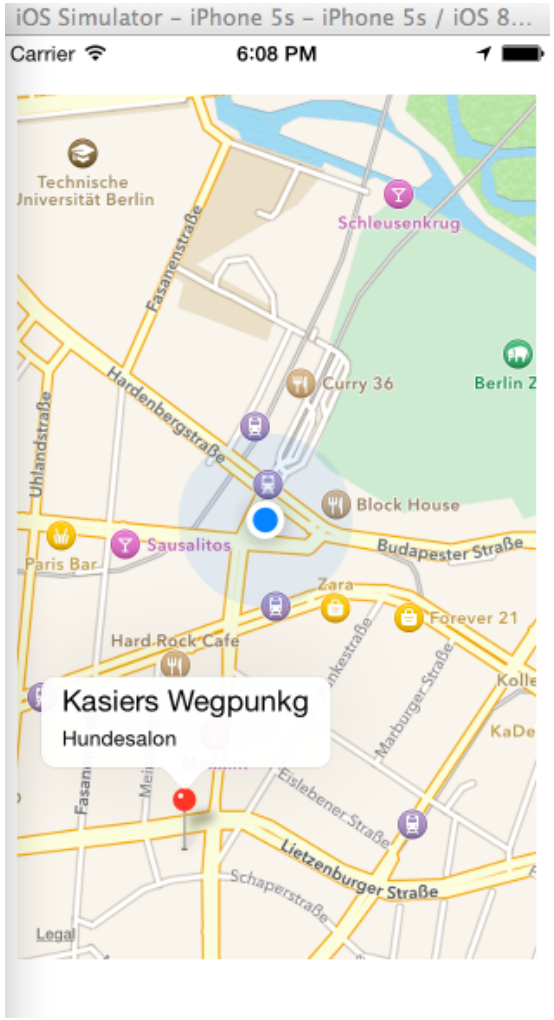
Die Annotation wird dem neuen View zugeordnet.

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“

Fertig


iOS Simulator – iPhone 5s – iPhone 5s / iOS 8...
Carrier
6:08 PM
Technische Universität Berlin
Schleusenkrug
Fasanenstraße
Curry 36
Hardenbergstraße
Uhlandstraße
Block House
Sausalitos
Paris Bar
Budapester Straße
Zara
Forever 21
Hard Rock Cafe
Kasiers Wegpunkg
Hundesalon
Marburger Straße
Eislebener Straße
Lietzenburger Straße
Schaperstraße
Legal