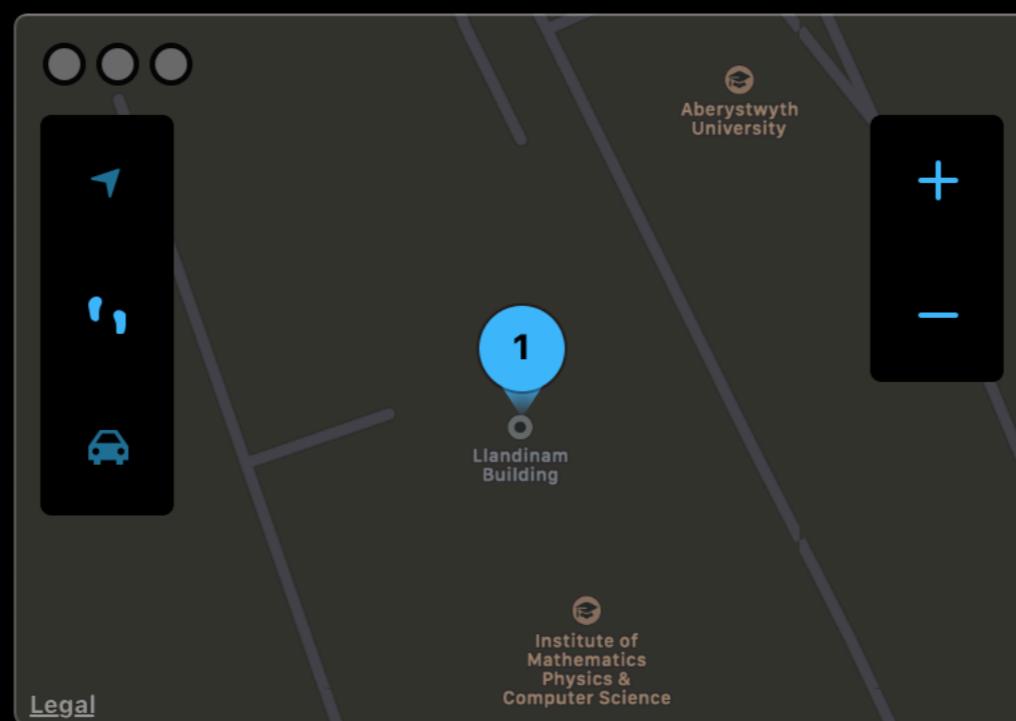


Offline maps and navigation



Vicente Garcia
zero.griffin@gmail.com
twitter: [_vaux](#)

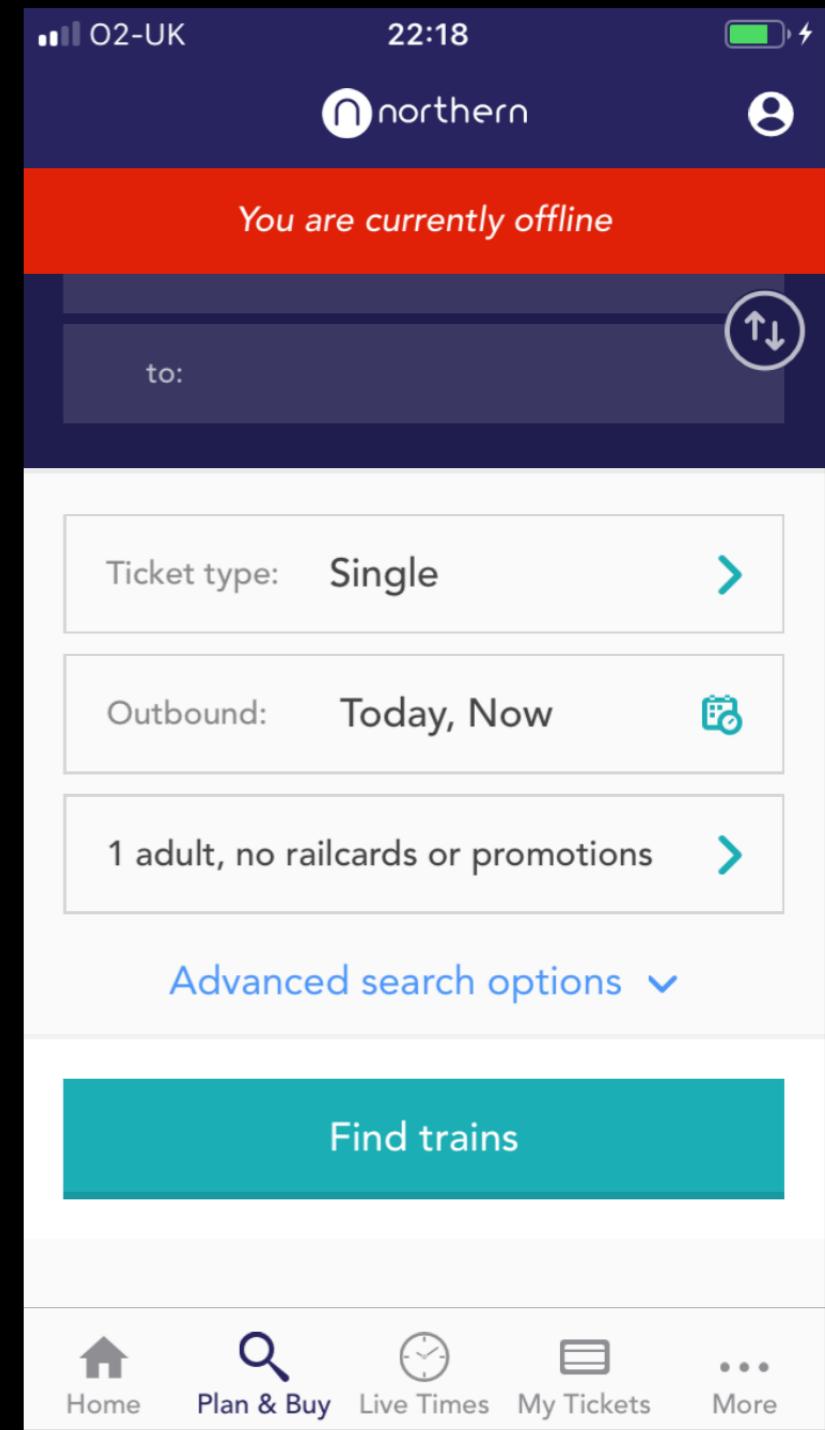
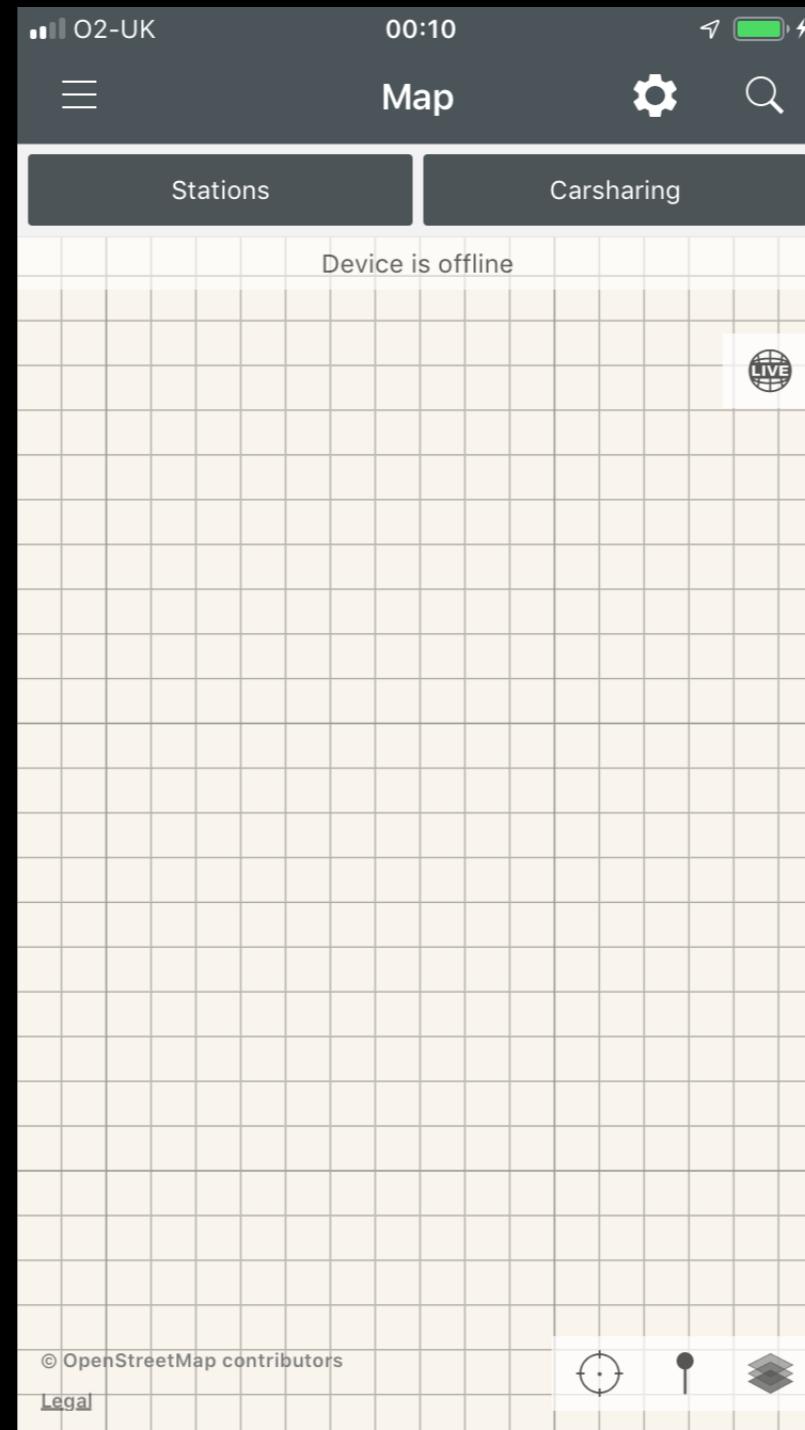
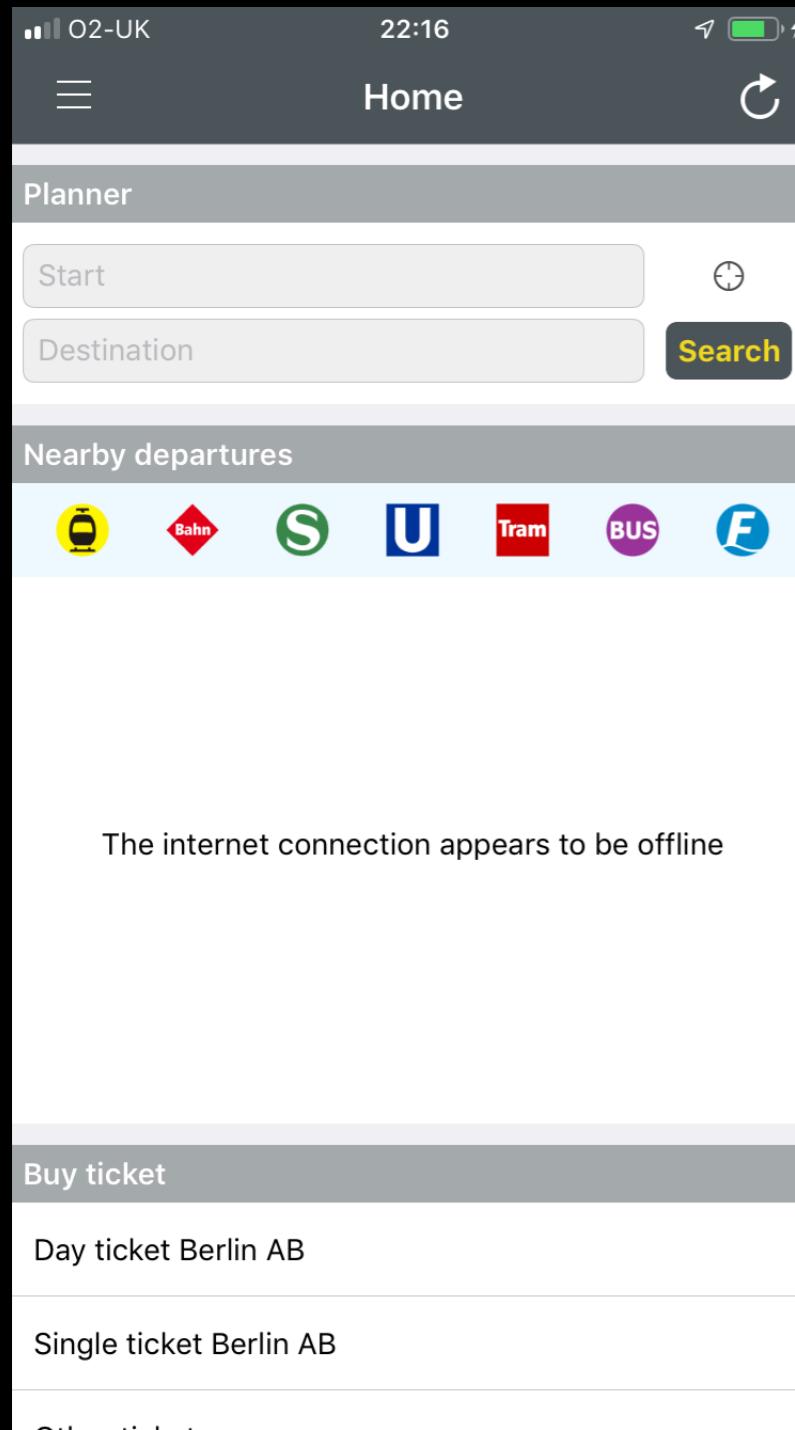
iOSDevUK 9

Aberystwyth University, 4th September 2019

Agenda

- Problem
- Solution
- Implementation
- Conclusions

Problem



Problem

We make apps that navigate you,
if you are online.

Overlooking

- Traveling abroad
- Bad signal
- You don't want to use your mobile data

What could we do?

Have static information.

- Stations
- Schedules (maybe)
- Points of interest
(Aberystwyth Castle)

Use your location to estimate.

- Points of interest close to you
- Distance to those points
- Should you turn left or right
(heading)

Store maps.

- Show where you are

Store directions.

- From and to points of interests
- On demand

Solution

We can navigate you,
even if you are offline.

- How does it work online
- What is usable offline
- What can we store

How does it work online

- Components of navigation
- Interactions
- Flow

Navigation

- GPS
- Maps
- Points of interest
- Directions

GPS

Global Positioning System,
maintained by the US government,
it is free and open.

Satellites acting as beacons
that devices can listen to
calculate their own
latitude and longitude.

SUPPORT:

Frequently Asked Questions

Address, Route, & Map Problems

Service Outages & Status Reports

Civil GPS Service Interface Committee (CGSIC)

Technical Documentation

External Links

About This Website

Contact Us

TAKE ACTION:

 Bookmark this page

 Share this page

Help with Address, Route, and Map Problems in GPS Devices and Apps

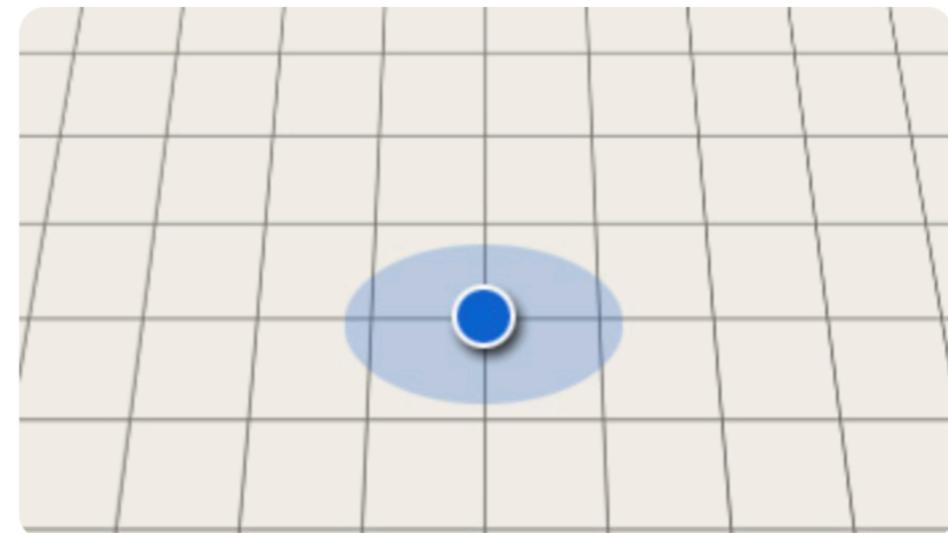
Frustrated by missing/wrong locations and directions in GPS devices and apps?

The problem is in the mapping software used by the devices/apps. That software comes from the private sector (e.g., Google, Apple), not from the GPS satellites.

GPS satellites are simply beacons, like lighthouses, that a device uses to calculate its own latitude and longitude. The satellites do not transmit any mapping information.

The U.S. government cannot correct mapping errors in consumer devices/apps, but the following links will help you report them to the responsible parties.

GPS only gives you the blue dot.



It does not provide the map!

HOW GPS WORKS

1 GPS satellites broadcast radio signals providing their locations, status, and precise time $\{t_1\}$ from on-board atomic clocks.

2 The GPS radio signals travel through space at the speed of light $\{c\}$, more than 299,792 km/second.

3 A GPS device receives the radio signals, noting their exact time of arrival $\{t_2\}$, and uses these to calculate its distance from each satellite in view.

4 Once a GPS device knows its distance from at least four satellites, it can use geometry to determine its location on Earth in three dimensions.



GPS
IS A CONSTELLATION
OF 24 OR MORE
SATELLITES FLYING
20,350 KM ABOVE THE
SURFACE OF THE EARTH.
EACH ONE CIRCLES THE
PLANET TWICE A DAY IN
ONE OF SIX ORBITS TO
PROVIDE CONTINUOUS,
WORLDWIDE
COVERAGE.

To calculate its distance from a satellite, a GPS device applies this formula to the satellite's signal:

$$\text{distance} = \text{rate} \times \text{time}$$

where **rate** is $\{c\}$ and **time** is how long the signal traveled through space.

The signal's travel **time** is the difference between the time broadcast by the satellite $\{t_1\}$ and the time the signal is received $\{t_2\}$.

The Air Force launches new satellites to replace aging ones when needed. The new satellites offer upgraded accuracy and reliability.

How does GPS help farmers? Learn more about the Global Positioning System and its many applications at

WWW.GPS.GOV



The GPS Master Control Station tracks the satellites via a global monitoring network and manages their health on a daily basis.

Ground antennas around the world send data updates and operational commands to the satellites.

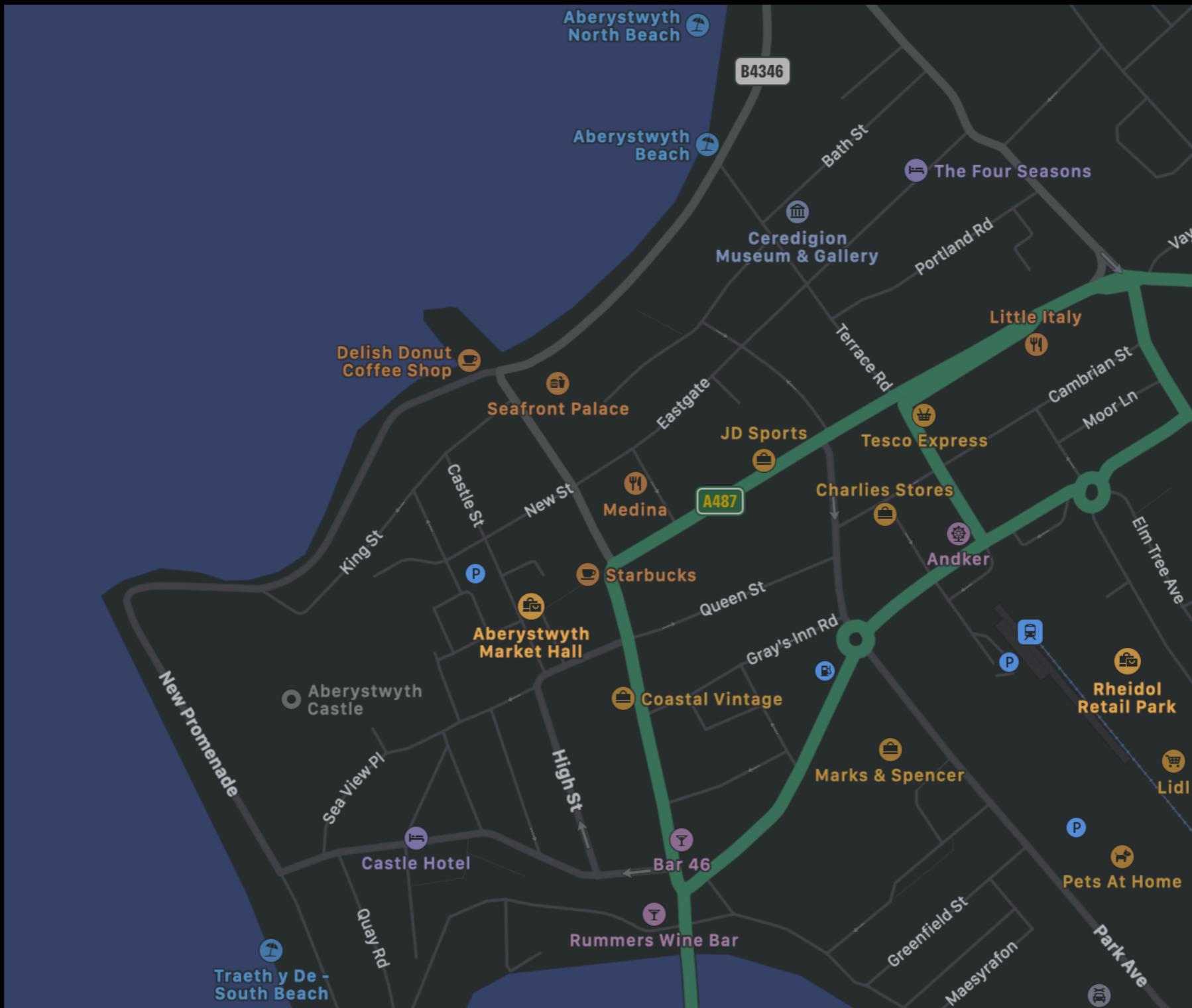


Map

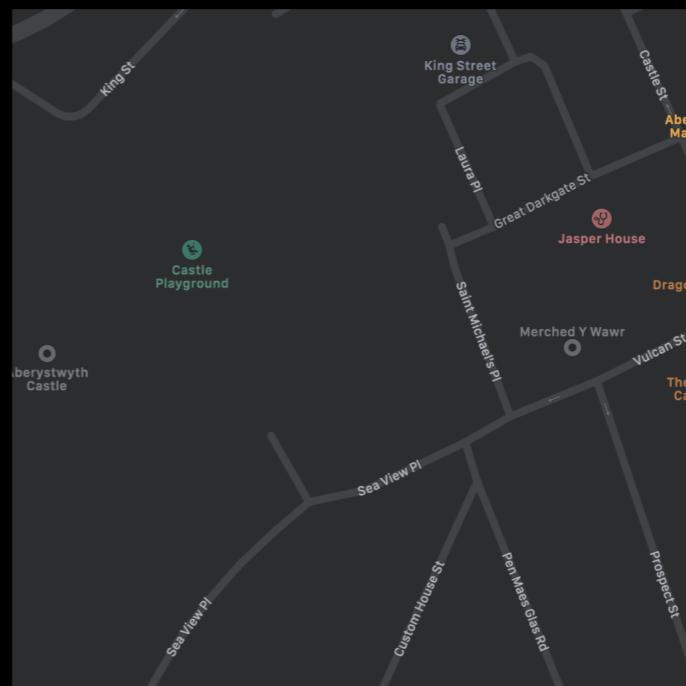
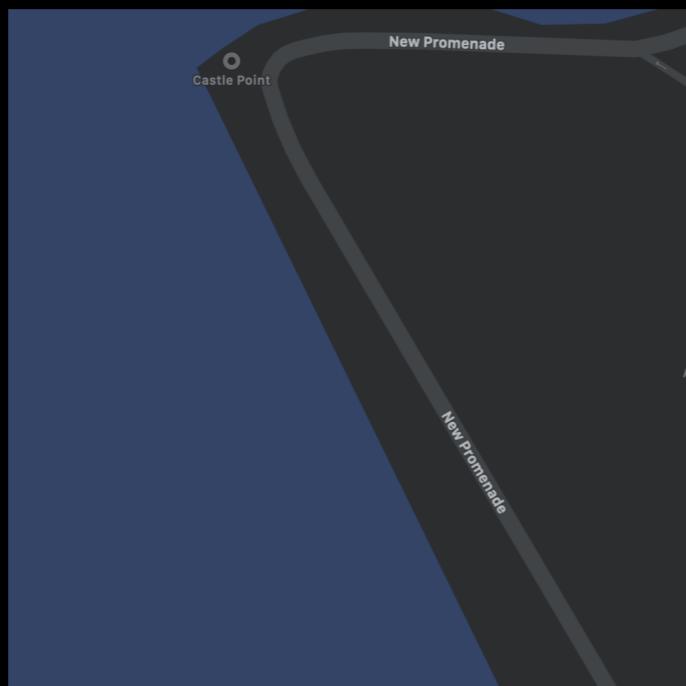
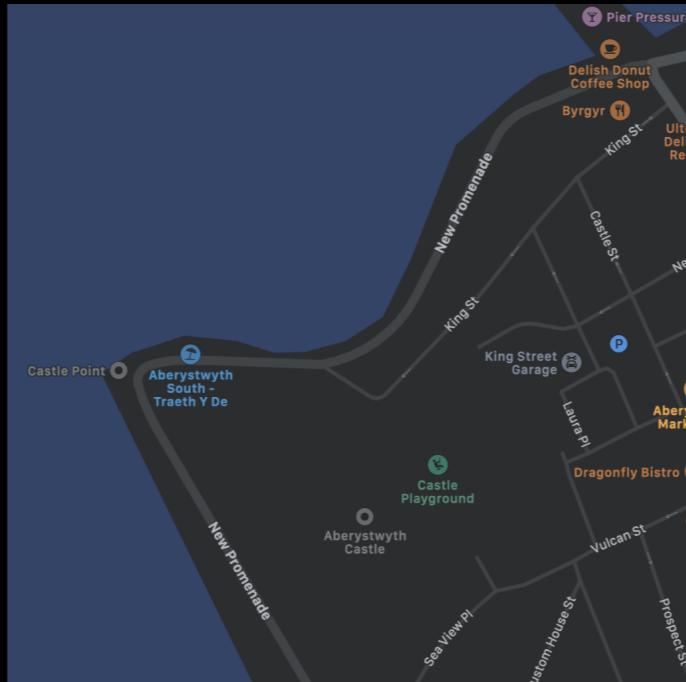
Set of images that we can associate with a region.

- Computer generated images
- Satellite imagery
- Aerial photography
- Vector rendering

How a map looks



What a map actually is



Each image represents a region
with a certain amount of detail.

The smaller the region, the more
detail you could add.

Region

A geometrical representation of
the world in 2d.

A rectangle from this
coordinate to this other.

Providers

- Apple Maps
- Google Maps
- OpenStreetMap
- Here
- TomTom

Point of interest

Information associated
with a location.

Location

- Latitude (north - south)
- Longitude (east - west)

Aberystwyth Castle

Latitude: 52.4135223

Longitude: -4.0880009

Description

13th-century fortress built by Edward I

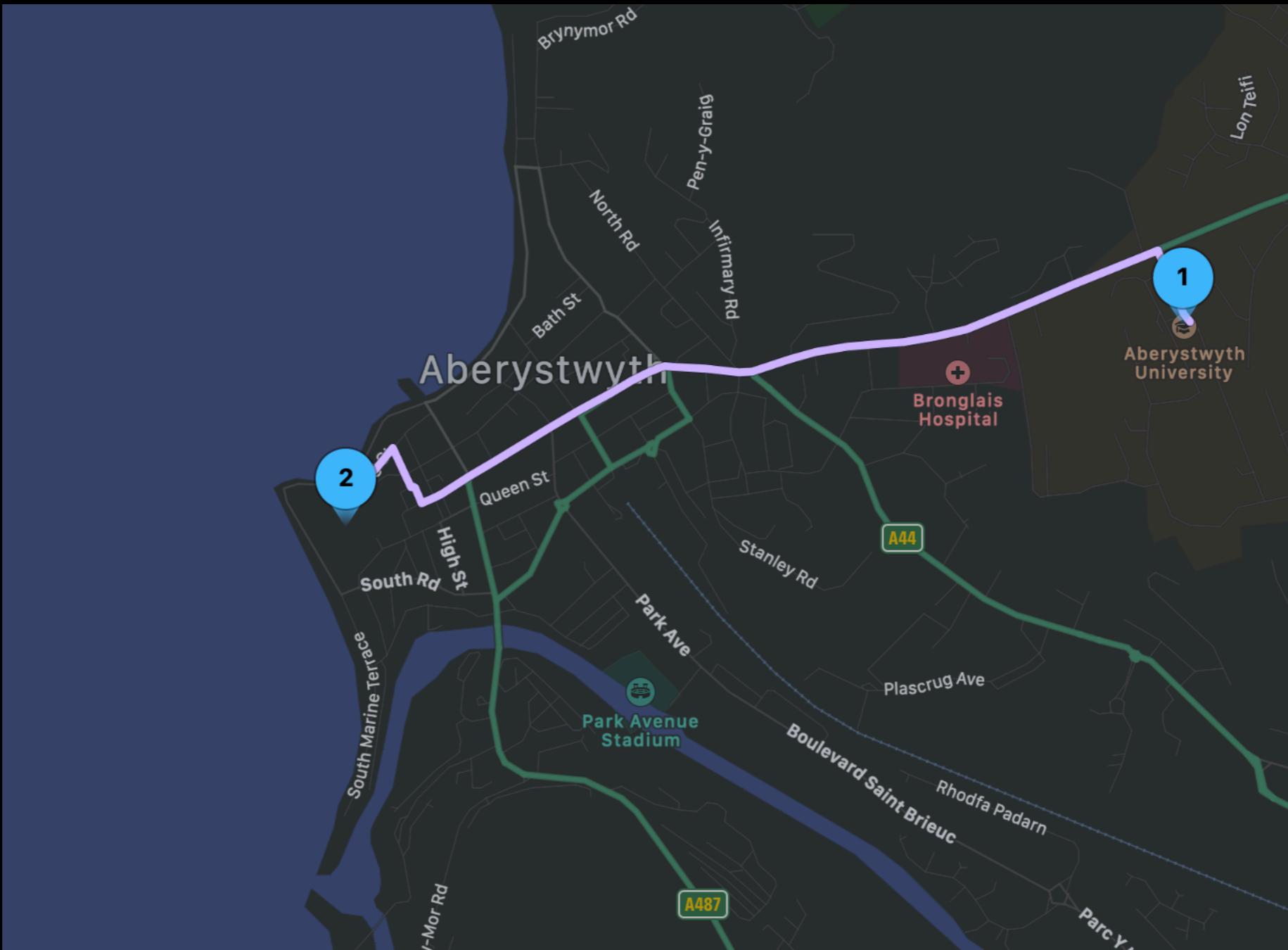
Address

King St, Aberystwyth, SY23 2AU, Wales

Directions

How to go from a point of interest to another.

From Aberystwyth University to the Castle



A route consisting of a set of coordinates to pass when traveling between 2 locations.

Can also contain

- Distance
- Expected duration
- Type of travel

Walking

Driving

Cycling

And indications specific per point in the route.

- Here turn left

Interactions

Coordinates

Map

Image 1

Image 2



Points of interest

Aberystwyth University

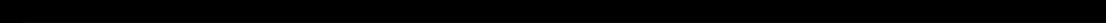
You



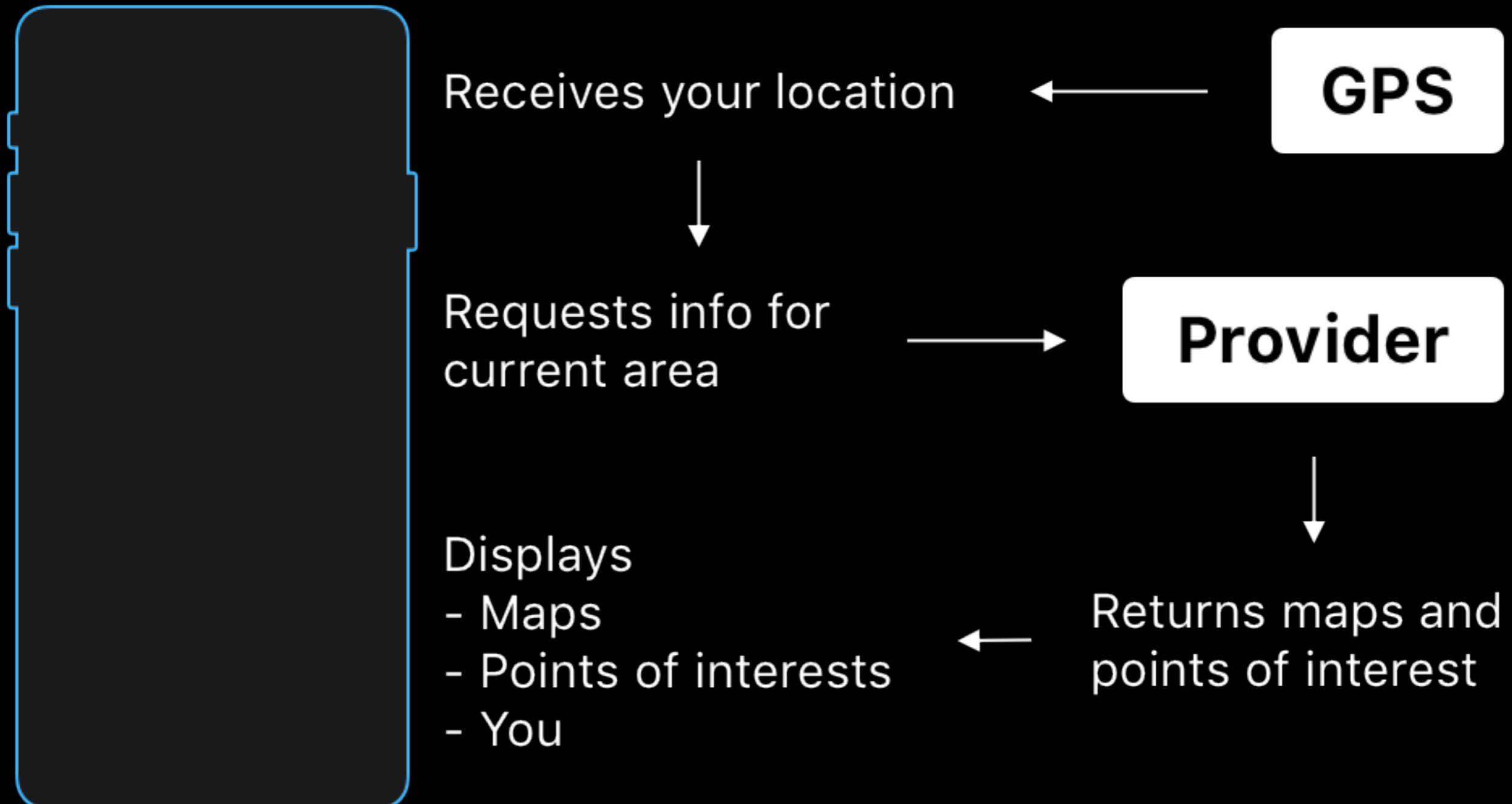
Directions

From

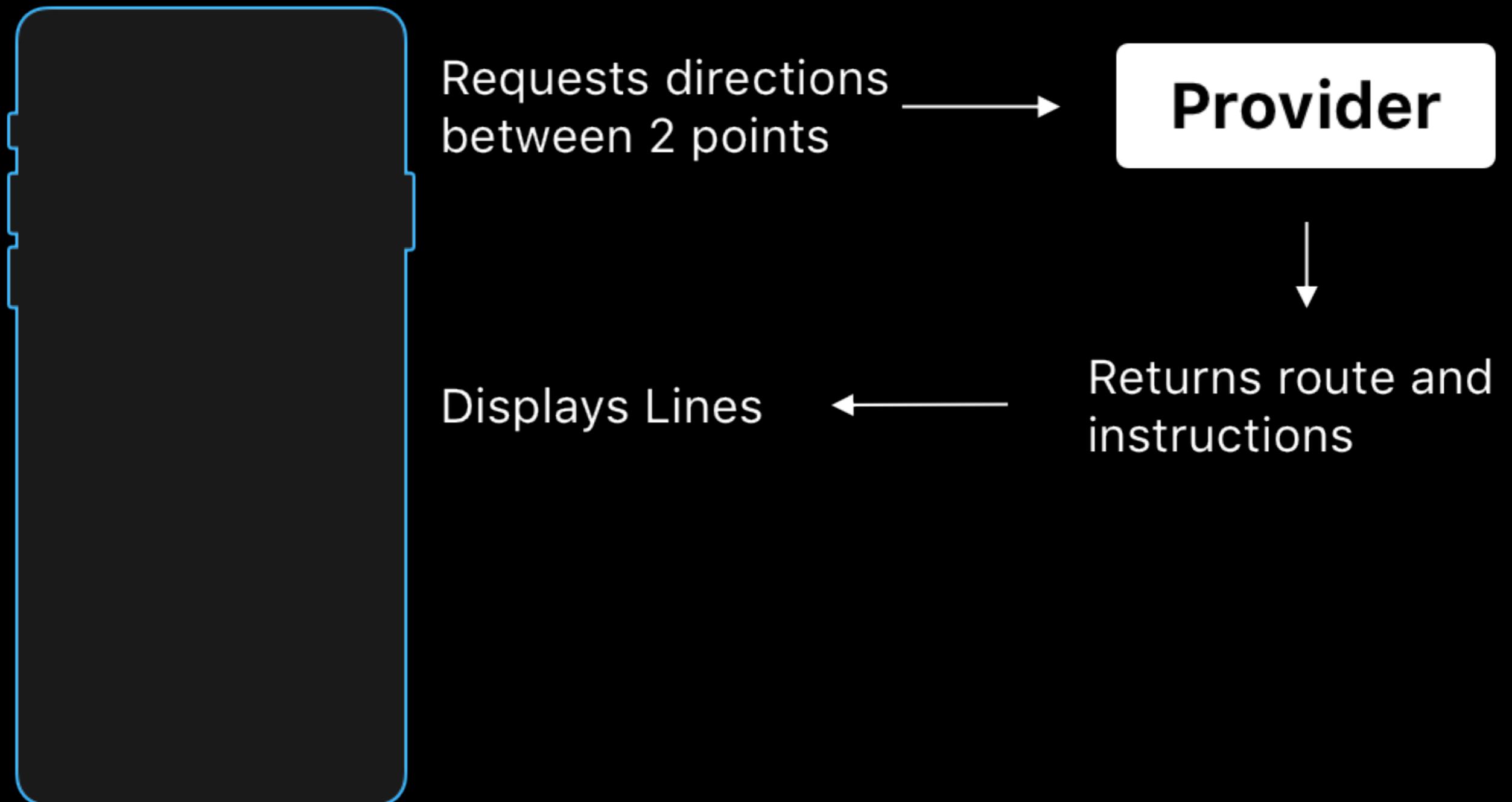
To



Flow



On demand



Offline

- Your location
- Area around you

Online

- Images
- Points of interest
- Routes

But these are static

- Images
- Points of interest

And routes between to points of interests are also static.*

- Without real time information like traffic and construction works.

What can we store

- Maps important for our app
- Maps requested by you
- Points of interests for our app
- Directions between points
- Directions requested by you

Disadvantages

- Uses memory storage
- Might not be free (provider)
- No real time

Implementation

- MapKit
- Map
- Points of interests
- Directions

MapKit

Cons

- Google Maps is more popular
- Not as detailed.

Pros

- Out of the box
- No quota limit

info.plist

iOS 8 - 10

NSLocationAlwaysUsageDescription

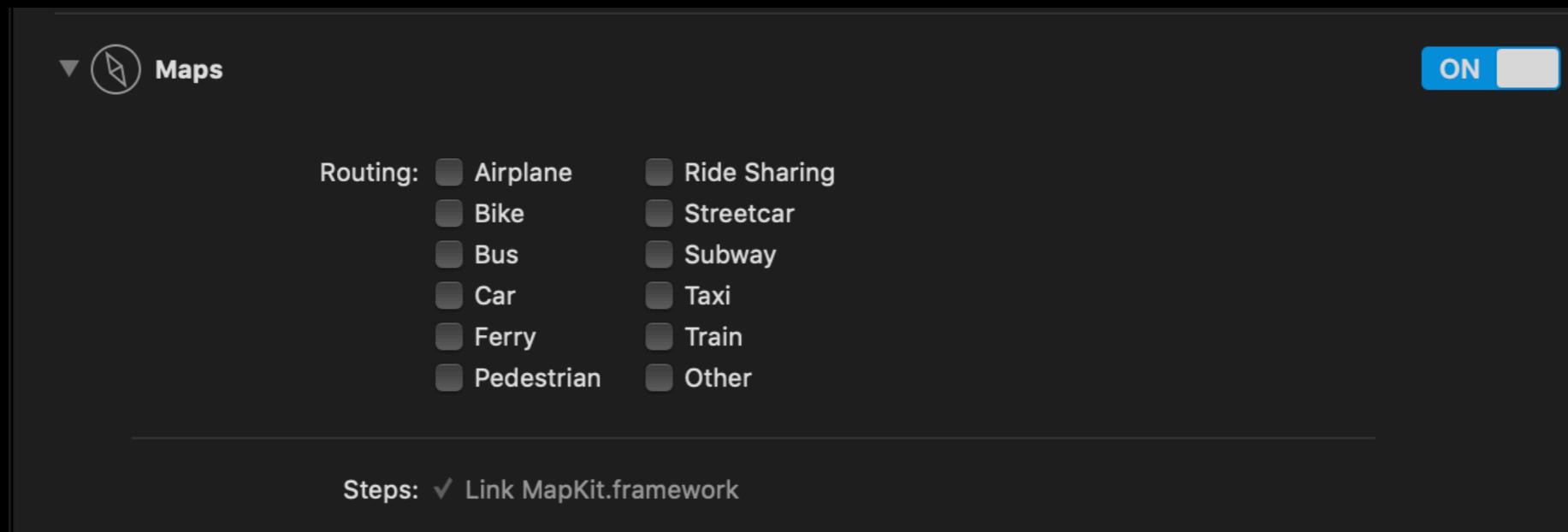
iOS 11+

NSLocationWhenInUseUsageDescription

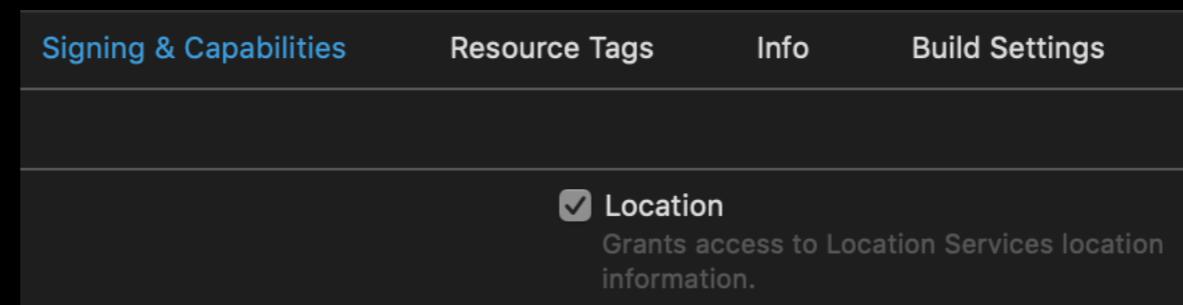
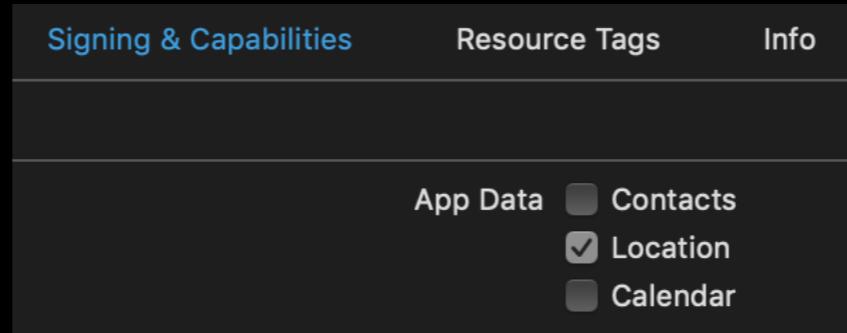
macOS 10.14+

NSLocationUsageDescription

In Xcode 10 or below you need to enable Maps capabilities.
No longer necessary in Xcode 11.



For macOS you also need to enable access to location in capabilities.



Map

MKMapViewSnapshotter

- iOS 7+
- macOS 10.9+

```
let options = MKMapSnapshotter.Options()
options.mapRect = mapRect

let snapshotter = MKMapSnapshotter(options: options)
snapshotter.start { snapshot, error in
    if let image = snapshot?.image {
        try? image.pngData()?.write(to: url)
    }
}
```

MKMapRect

2d representation of the curved surface of the globe.

Often used to show the entire surface of the globe all at once.

But, before using MKMapRect
we need to understand

- MKMapPoint
- Size of the world
- Zoom level

MKMapPoint

This data structure represents a point on the map.

You can create them with coordinates, perform calculations with them and convert back to coordinate values.

The entire surface of the globe,
in points is

```
print(MKMapRect.world.size)
// MKMapSize(width: 268435456.0, height: 268435456.0)
```

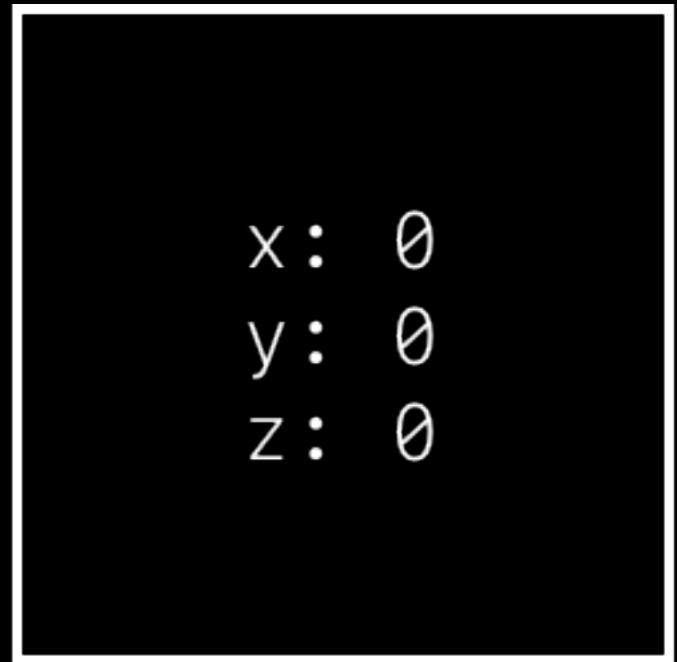
Zoom level

We can use a **z coordinate** or **zoom** level to define how close or far we want to generate or map images.

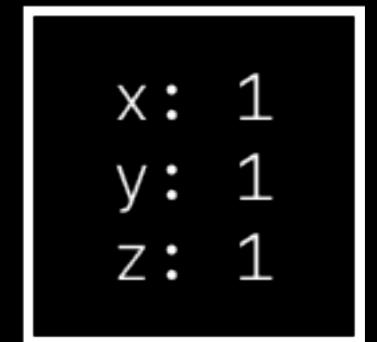
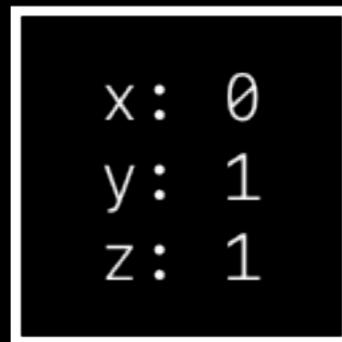
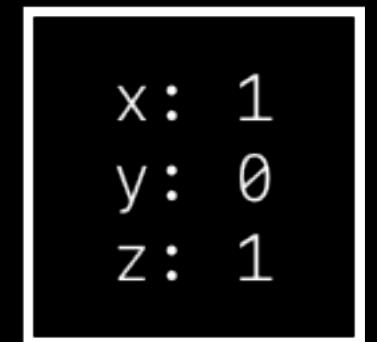
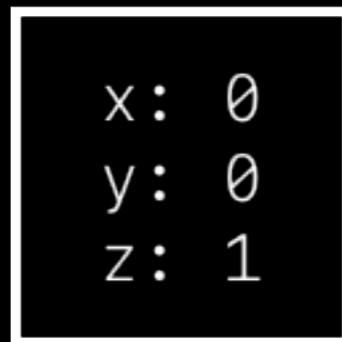
If we want to display the whole world at once, in a just one image, we could call that a **tile** at the position

x: 0
y: 0
z: 0

Zoom 1 divides zoom 0
into 4 tiles.



Zoom 2 divides each tile
of zoom 1 into 4 again,
and so on.



If we want our image to be **256x256** pixels that would mean we can have **20 zoom levels**, or:

```
let maxZoom = log2(MKMapRect.world.width / 256)
```

And the size of tiles in map points:

```
let tileSize = MKMapRect.world.width / pow(2, zoomLevel)
```

256x256 is the default for MKMapSnapshotter and is also the standard with other providers.

That also means we would need to have

4^{20} or

1,099,511,627,776

images at the maximum zoom to display the whole globe.

```
// Aberystwyth castle
let coordinate = CLLocationCoordinate2D(latitude: 52.413244,
                                         longitude: -4.089675)
let point = MKMapPoint(coordinate)

let zoomLevel = 16.0
let sizeAtZoom = MKMapRect.world.width / pow(2, zoomLevel)
let mapSize = MKMapSize(width: sizeAtZoom, height: sizeAtZoom)

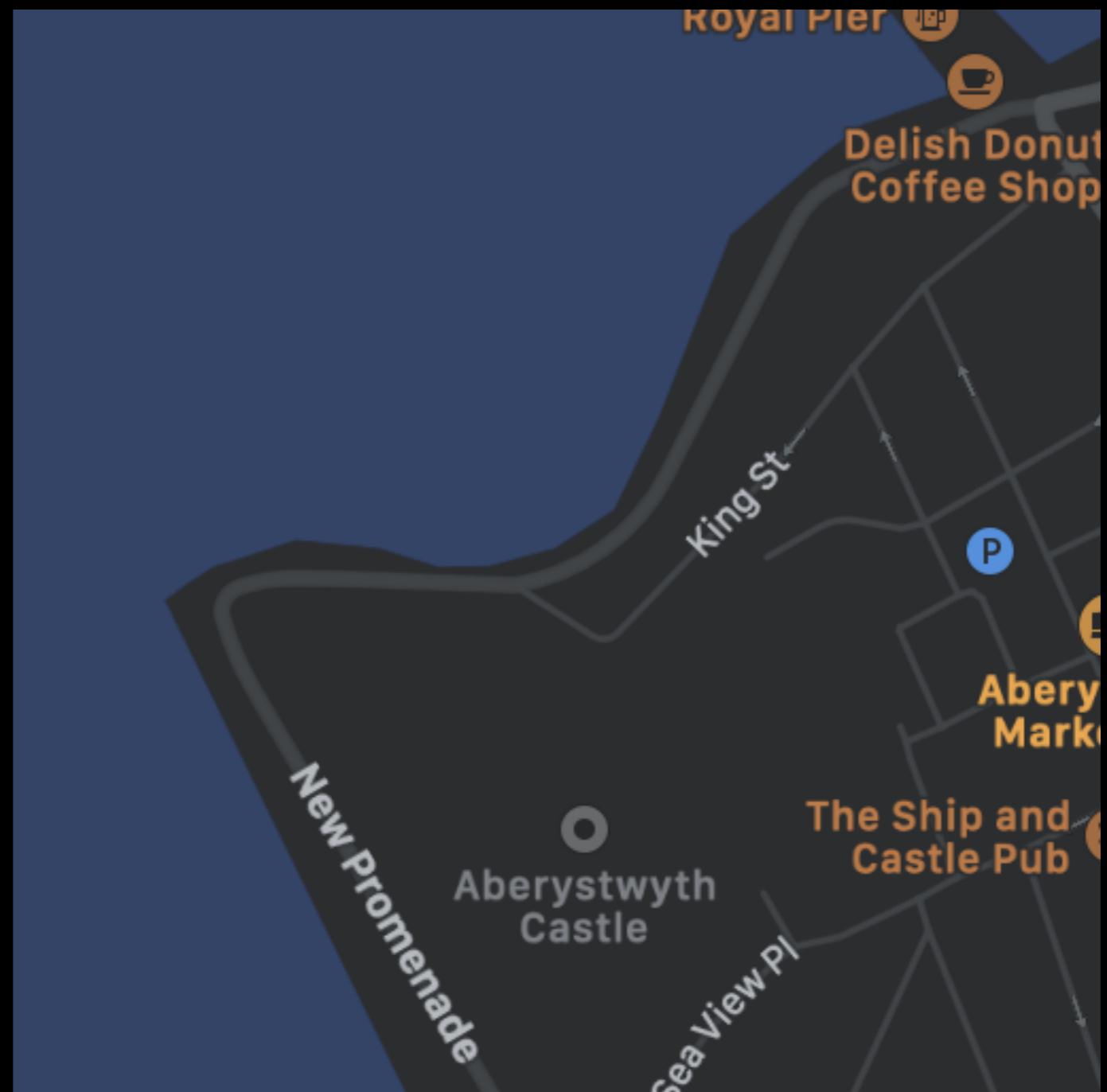
let pointAsTile = MKMapPoint(
    x: floor(point.x / sizeAtZoom) * sizeAtZoom,
    y: floor(point.y / sizeAtZoom) * sizeAtZoom)

let options = MKMapSnapshotter.Options()
options.mapRect = MKMapRect(origin: pointAsTile, size: mapSize)

let snapshotter = MKMapSnapshotter(options: options)
snapshotter.start { snapshot, error in
    if let image = snapshot?.image {
        try? image.pngData()?.write(to: url)
    }
}
```

Aberystwyth castle

x: 131166208
y: 88162304
z: 16
size: 4096



To use these tiles in our **MKMapView**

Assuming we saved every tile with a name like
x_y_x.png in the folder **myFolder**, for example:

myFolder/1_0_1.png

```
let tileOverlay = MKTileOverlay(urlTemplate:  
    "myFolder/{x}_{y}_{z}.png")
```

```
tileOverlay.canReplaceMapContent = true  
mapView.addOverlay(tileOverlay)
```

In our **MKMapViewDelegate**

```
func mapView(_ mapView: MKMapView, rendererFor overlay:  
MKOverlay) -> MKOverlayRenderer {  
  
    if let tileOverlay = overlay as? MKTileOverlay {  
        return MKTileOverlayRenderer(tileOverlay: tileOverlay)  
    } else {  
        return MKOverlayRenderer(overlay: overlay)  
    }  
}
```

Points of interest

We could store those relevant to our app.

User's location will be shown, if granted permission, and don't store it.

```
struct PointOfInterest: Codable {  
    var name = ""  
    var latitude = 0.0  
    var longitude = 0.0  
}
```

```
var point = PointOfInterest()  
point.name = "Aberystwyth castle"  
point.latitude = 52.413244  
point.longitude = -4.089675  
// store point
```

```
var point = PointOfInterest()
point.name = "Aberystwyth castle"
point.latitude = 52.413244
point.longitude = -4.089675
// store point

/*
 *
 */

// load point
let annotation = MKPointAnnotation()
annotation.title = point.name
annotation.coordinate = CLLocationCoordinate2D(
    latitude: point.latitude, longitude: point.longitude)

mapView.addAnnotation(annotation)
```

We can also let users select places they want to store.

Directions

```
// coordinateA and B are CLLocationCoordinate2D
let placeA = MKPlacemark(coordinate: coordinateA)
let placeB = MKPlacemark(coordinate: coordinateB)

let request = MKDirections.Request()
request.transportType = .walking
request.source = MKMapItem(placemark: placeA)
request.destination = MKMapItem(placemark: placeB)
```

```
let directions = MKDirections(request: request)
directions.calculate { response, error in

    if let route = response?.routes.first {

        // route is MKRoute
        // route.polyline.points() is a
        // UnsafeMutablePointer<MKMapPoint>

        for i in 0 ..< route.polyline.pointCount {
            // route.polyline.points()[i].coordinate
            // is a CLLocationCoordinate2D
        }
    }
}
```

Once they are stored

```
// load route coordinates and map them  
// back to CLLocationCoordinate2D  
  
let polyline = MKPolyline(coordinates: coordinates,  
                           count: coordinates.count)  
mapView.addOverlay(polyline, level: .aboveRoads)
```

In our MKMapViewDelegate

```
func mapView(_ mapView: MKMapView, rendererFor overlay:  
MKOverlay) -> MKOverlayRenderer {  
  
    if let tileOverlay = overlay as? MKTileOverlay {  
  
        return MKTileOverlayRenderer(tileOverlay: tileOverlay)  
  
    } else if let polyline = overlay as? MKPolyline {  
  
        return MKPolylineRenderer(polyline: polyline)  
  
    } else {  
  
        return MKOverlayRenderer(overlay: overlay)  
    }  
}
```

Things to consider

- When creating a snapshot it might take up to **20 seconds** to respond, and there is no timeout support.
- You need a different **MKMapSnapshotter** for each image.
- Size of images should be divisible by **256**, but preferably smaller than **2560**. Maximum supported is **4928**.

Conclusions

- We can can navigate you,
even if you are offline.
- **Argonaut** in the AppStore,
it is free and OpenSource

<https://github.com/oinkhub/argonaut>



Thanks