// ARKit 1.5

delete everthing after viewWillDisappear

in Assets.xcassets → click "+" → right click → New AR Resource Group

add a picture with known size enter size in ⊕ ⇒ the actual size in meters

// viewController.swift // only detection

```
import UIKit
import SceneKit
import ARKit

class ViewController: UIViewController, ARSCNViewDelegate {
    var sceneView: ARSCNView!

    private var hud : MBProgressHUD!

    override func viewDidLoad(){
        super.viewDidLoad()
        self.sceneView = ARSCNView(frame: self.view.frame)
        self.view.addSubview(self.sceneView)
        self.sceneView.autoenablesDefaultLighting = true
        sceneView.delegate = self
        let scene = SCNScene()
        sceneView.scene = scene

    }

    // ⇒ called every time find an anchor
    func renderer(_ renderer : SCNSceneRenderer, didAdd node: SCNNode, for anchor : ARAnchor ){
        if let imageAnchor = anchor as? ARImageAnchor{
            if let name = imageAnchor.referenceImage.name{
                DispatchQueue.main.async{
                    self.hud = MBProgressHUD.showAdded(to: self.view, animated: true)
                    self.hud.label.text = name
                    self.hud.hide(animated: true, afterDelay: 2.0)
                }
            }
        }
    }

    override func viewWillAppear(_ animated : Bool){
        super.viewWillAppear(animated)
        //guard let referenceImages = ARReferenceImage.referenceImages(inGroupNamed: "AR Resources", bundle: nil) else {
            fatalError("missing expected asset catalog resources.")
        }
        let configuration = ARWorldTrackingConfiguration()
        let blackBerryRefImage = ARReferenceImage((UIImage(named: "BlackBerry.JPG")?.cgImage!)!,
            orientation: CGImagePropertyOrientation.up, physicalWidth: 0.28)

        let javaRefImage = ARReferenceImage((UIImage(named: "Java.JPG")?.cgImage!)!,
```

// MBProgressHUD written in objectC

header .h
implimation .m
⇒ those two needed

↓ drag into the xcode and create Bridging header

↓ in the Bridging header

```
#import "MBProgressHUD.h"
```

// UI API can not be called on a background thread

should do a safer way

```
        orientation: CGImagePropertyOrientation.up, physicalWidth: 0.16)

    // configuration.detectionImages = referenceImages
    configuration.detectionImages = Set([blackBerryRefImage, javaRefImage])
    sceneView.session.run(configuration)
}
```

// with model displayed above image (not moving with the reference image)
// remove things before viewWillDisappear

```
// ViewController.swift
import UIKit
import SceneKit
import ARKit

class ViewController: UIViewController, ARSCNViewDelegate {
    var sceneView: ARSCNView!

    private var hud: MBProgressHUD!

    override func viewDidLoad() {
        super.viewDidLoad()
        self.sceneView = ARSCNView(frame: self.view.frame)
        self.view.addSubview(self.sceneView)
        self.sceneView.autoenablesDefaultLighting = true
        sceneView.delegate = self
        let scene = SCNScene()
        sceneView.scene = scene

    }

    func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNode, for anchor: ARAnchor) {
        if anchor is ARImageAnchor {
            let phoneScene = SCNScene(named: "Phone-01.scn")!
            let phoneNode = phoneScene.rootNode.childNode(withName: "parentNode", recursively: true)!
            let rotationAction = SCNAction.rotateBy(x: 0, y: 0.5, z: 0, duration: 1)
            let infiniteAction = SCNAction.repeatForever(rotationAction)
            phoneNode.runAction(infiniteAction)

            phoneNode.position = SCNVector3(anchor.transform.columns.3.x,
                                            anchor.transform.columns.3.y,
                                            anchor.transform.columns.3.z)
            self.sceneView.scene.rootNode.addChildNode(phoneNode)
        }
    }
}
```

```
override func viewWillAppear(_ animated: Bool){
    super.viewWillAppear(animated)
    let configuration = ARWorldTrackingConfiguration()

    guard let referenceImages = ARReferenceImage.referenceImages(inGroupNamed: "AR Resources", bundle: nil) else {
        fatalError("missing expected asset catalog resources.")
    }
    configuration.detectionImages = referenceImages
    sceneView.session.run(configuration)
}
```

// ARKit 2.0 change device version => ios 11 to ios 12 (current one)

```
// viewController.swift
import UIKit
import SceneKit
import ARKit

class ViewController: UIViewController, ARSCNViewDelegate{
    var sceneView: ARSCNView!

    private var hud : MBProgressHUD!

    override func viewDidLoad(){
        super.viewDidLoad()
        self.sceneView = ARSCNView(frame: self.view.frame)
        self.view.addSubview(self.sceneView)
        self.sceneView.autoenablesDefaultLighting = true
        sceneView.delegate = self
        let scene = SCNScene()
        sceneView.scene = scene

    }

    func renderer(_ renderer: SCNSceneRenderer, didAdd node: SCNNode, for anchor: ARAnchor){
        if anchor is ARImageAnchor{
            let phoneScene = SCNScene(named: "phone-01.scn")!
            let phoneNode = phoneScene.rootNode.childNode(withName: "parentNode", recursively: true)!
            let rotationAction = SCNAction.rotateBy(x: 0, y: 0.5, z: 0, duration: 1)
            let infiniteAction = SCNAction.repeatForever(rotationAction)
            phoneNode.runAction(infiniteAction)

            phoneNode.position = SCNVector3(anchor.transform.columns.3.x,
                                            anchor.transform.columns.3.y,
                                            anchor.transform.columns.3.z)
            //self.sceneView.scene.rootNode.addChildNode(phoneNode)
            node.addChildNode(phoneNode)          // ← new in ARKit 2.0
        }
    }
```

```
override func viewWillAppear(_ animated: Bool){
    super.viewWillAppear(animated)
    //let configuration = ARWorldTrackingConfiguration()

    let configuration = ARImageTrackingConfiguration() ← new in ARKit 2.0
    guard let referenceImages = ARReferenceImage.referenceImages(inGroupNamed: "AR Resources", bundle: nil) else {
        fatalError("missing expected asset catalog resources.")
    }
    // configuration.detectionImages = referenceImages
    configuration.trackingImages = referenceImages ← new in ARKit 2.0
    configuration.maximumNumberOfTrackedImages = 1 ← default is 1

    sceneView.session.run(configuration)
}
```