```
arkitSession = ARKitSession()
```

```
worldTracking = WorldTrackingProvider()
planeDetection = PlaneDetectionProvider()
```

```
arkitSession.run([worldTracking, planeDetection])
```

```
planeAnchorHandler = PlaneAnchorHandler(rootEntity: root)
```

PlaneAnchorHandler是sample中定义的类，用于处理得到平面锚点，渲染检测到的平面

```
for await anchorUpdate in planeDetection.anchorUpdates {
        await planeAnchorHandler.process(anchorUpdate)
}

for await anchorUpdate in worldTracking.anchorUpdates {
        persistenceManager.process(anchorUpdate)
}
```

```
var anchorUpdates : AsyncSequence<AnchorUpdate<PlaneAnchor>>

struct AnchorUpdate{
        AnchorType            anchor;
        TimeInterval          timestamp;
        AnchorUpdate.Event  event;
}
```

```
struct PlaneAnchor {
        simd_float4x4             originFromAnchorTransform;
        PlaneAnchor.Alignment     alignment;
        PlaneAnchor.Geometry      geometry;
        UUID                      id;
        TimeInterval              timestamp;
        SurfaceClassification     surfaceClassification;
}
```

```
Class  PlaneAnchorHandler {
    ......
    func process(_ anchorUpdate: AnchorUpdate<PlaneAnchor>) async {
        let anchor = anchorUpdate.anchor
        if anchorUpdate.event == .removed {
        planeAnchorsByID.removeValue(forKey: anchor.id)
          if let entity = planeEntities.removeValue(forKey: anchor.id) {
            entity.removeFromParent()
             }
           return
          }
      let entity = Entity()
      entity.name = "Plane \(anchor.id)"
      entity.setTransformMatrix(anchor.originFromAnchorTransform, relativeTo: nil)
    }
      ......

}
```

根据event对entity 进行更新