Building AR Experiences with Reality Composer

Michelle C., Designer
Pau Sastre Miguel, Software Engineer
Abhi Ashutosh, Software Engineer
Reality Composer

Get started with AR and 3D
Layout and pre-visualization
Content library
Simple interactions
Xcode integration
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Building Scenes

Adding Behaviors

Using Physics

Building Apps
Anatomy of a Scene

Scene

Anchor  Objects  Behaviors  Physics World
Anchoring the Scene
Anchoring the Scene

Choose an Anchor

- **Horizontal**
  For tabletop experiences.

- **Vertical**
  Can be placed on a wall or float.

- **Image**
  Augment 2D media in your environment.

- **Face**
  For augmenting faces – uses front-facing camera.

[Buttons: Cancel, Choose]
Working with Scenes
Demo
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Building Scenes

Adding Behaviors

Using Physics

Building Apps
Adding Behaviors

Pau Sastre Miguel, Software Engineer
Anatomy of a Scene

- Anchor
- Objects
- Behaviors
- Physics World
Behaviors

- Trigger
- Action Sequence
Triggers

- Start
- Tap
- Proximity
- Collision
- Notification
Triggers

- Start
- Tap
- Proximity
- Collision
- Notification
Triggers

- Start
- Tap
- Proximity
- Collision
- Notification
Triggers

- Start
- Tap
- Proximity
- Collision
- Notification
Triggers

- Start
- Tap
- Proximity
- Collision
- Notification
Triggers

Start
Tap
Proximity
Collision
Notification
Action Sequence

Groups

Looping

Exclusive action sequences
Looping

Action Sequence

Action 1

Action 2

Action 3

Action 4

Action 5
Looping

Action Sequence

Action 1

Action 2

Action 3

Action 4

Action 5
Exclusive Action Sequences

Exclusive Action Sequence 1

Exclusive Action Sequence 2

Action Sequence 3
Visibility Actions

Show
Hide
Visibility Actions

Show

Hide
Visibility Actions

Show

Hide
Visibility Actions

Show

Hide
Animate Actions

- Emphasis
- Spin
- Orbit
- Usdz Animation
Animate Actions

- Emphasis
- Spin
- Orbit
- Usdz Animation
Animate Actions

- Emphasis
- Spin
- Orbit
- Usdz Animation
Animate Actions

- Emphasis
- Spin
- Orbit
- Usdz Animation
Move Actions

- Move By
- Move To
Move Actions

- Move By
- Move To
Move Actions

Move By

Move To
Move Actions

- Move By
- Move To
Look At Action
Look At Action

Look At
Audio Actions

- Play Sound
- Play Ambient
- Play Music
Audio Actions

- Play Sound
- Play Ambient
- Play Music
Demo
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Physics
Physics

Physical materials

Forces

Collisions
Materials

Concrete  Ice  Lead  Plastic  Wood  Rubber
Forces
Gravity
Forces
Force action

Add Force
Forces

Force action

Add Force
Collisions
Collides vs. simulates
Collisions
Collides vs. simulates
Collisions

Shapes

Box

Sphere

Capsule
Collisions
Trigger

Collide
Collisions
Trigger

Collide
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Building Scenes
Adding Behaviors
Using Physics
Building Apps
Building Apps

Abhi Ashutosh, Software Engineer
Getting Started

RealityKit AR or Game template in Xcode
New project from Reality Composer
Reality File exported from Reality Composer
Getting Started

Reality Composer Project

Reality File
Reality Composer Project

Project file for Reality Composer

Included in RealityKit AR and Game templates

Preview available in Xcode

Xcode automatically exports as a Reality File
Chapter Completed!

- Moon
- Earth
- Sun
Reality Composer Project
Automatic export in Xcode

SolarSystem.rcproject → Build in Xcode → SolarSystem.reality
Reality Composer Project
Automatic export in Xcode

SolarSystem.rcproject → Build in Xcode → MyExperience.app
Reality File

Contains all data required for rendering and simulation, optimized for RealityKit

Exported from Reality Composer

Automatically exported by Xcode

Reference directly in an application

Preview in AR Quick Look
Reality File

Contains all data required for rendering and simulation, optimized for RealityKit

Exported from Reality Composer

Automatically exported by Xcode

Reference directly in an application

Preview in AR Quick Look
Reality File

Structure

Experience.reality
Reality File

Structure

Experience.reality

My First Scene

My Second Scene
Reality File

Structure

Experience.reality

Anchor

My First Scene

Entity

Entity

Entity

Anchor

My Second Scene

Entity

Entity

Entity
Reality Composer Project in Xcode

Build in Xcode

SolarSystem.rcproject → SolarSystem.reality
Code Generation in Xcode

Build in Xcode

SolarSystem.rcproject → SolarSystem.reality

SolarSystem.swift
Code Generation in Xcode

Automatically generated by Xcode

Application-specific, generated API for
- Scenes
- Named entities
- Notify Actions
- Notification Triggers
Chapter Completed!

- Earth
  - Mass: $5.972 \times 10^{24}$ kg
  - Radius: 6,371 km

- Sun
  - Mass: $1.989 \times 10^{30}$ kg
  - Radius: 695,510 km

Configure
- Name: Sun
Chapter Completed!

Name: Sun
Code Generation in Xcode

Generated code structure

SolarSystem.rcproject

SizeChapter

Sun
Earth
Moon
Code Generation in Xcode

Generated code structure

SolarSystem.swift

SizeChapter

Sun

Earth

Moon
Code Generation in Xcode

Generated code structure

SolarSystem.swift

Entity, HasAnchoring

SizeChapter

Sun
Earth
Moon
Code Generation in Xcode

Generated code structure

- `SolarSystem.swift`
- `Entity, HasAnchoring`
  - `SizeChapter`
    - `Entity`
      - `sun`
    - `Entity`
      - `earth`
    - `Entity`
      - `moon`
Code Generation in Xcode
Generated code structure

SolarSystem.swift

- loadSizeChapter
- Entity, HasAnchoring
- loadSizeChapterAsync

Entity, Entity, Entity
sun, earth, moon
Code Generation in Xcode

Loading a Reality File synchronously

```swift
let seasonsChapter = try? SolarSystem.loadSeasonsChapter()

// Use the loaded anchor here
```
Code Generation in Xcode
Loading a Reality File synchronously

```swift
let seasonsChapter = try? SolarSystem.loadSeasonsChapter()

// Use the loaded anchor here
```
Code Generation in Xcode
Loading a Reality File asynchronously

```swift
SolarSystem.loadSeasonsChapterAsync { result in
    switch result {
    case .success(let anchor):
        // Use loaded anchor here
    case .failure(let error):
        // Handle failure
    }
}
```
Code Generation in Xcode
Loading a Reality File asynchronously

```swift
SolarSystem.loadSeasonsChapterAsync { result in
    switch result {
    case .success(let anchor):
        // Use loaded anchor here
    case .failure(let error):
        // Handle failure
    }
}
```
SolarSystem.loadSeasonsChapterAsync { result in
    switch result {
        case .success(let anchor):
            // Use loaded anchor here
        case .failure(let error):
            // Handle failure
    }
}
Code Generation in Xcode

Loading a Reality File asynchronously

```swift
SolarSystem.loadSeasonsChapterAsync { result in
    switch result {
    case .success(let anchor):
        // Use loaded anchor here
    case .failure(let error):
        // Handle failure
    }
}
```
Code Generation in Xcode
Accessing entities

// Load Reality File anchor, `seasonsChapter`, above...

let sun = seasonsChapter.sun
let earth = seasonsChapter.earth
let moon = seasonsChapter.moon
// Load Reality File anchor, `seasonsChapter`, above...

let sun = seasonsChapter.sun
let earth = seasonsChapter.earth
let moon = seasonsChapter.moon
// Load the "Seasons Chapter" scene from the "SolarSystem" Reality File
let seasonsChapter = try! SolarSystem.loadSeasonsChapter()

// Add the seasons chapter anchor into the scene
arView.scene.anchors.append(seasonsChapter)
Code Generation in Xcode

Getting started

```swift
// Load the "Seasons Chapter" scene from the "SolarSystem" Reality File
let seasonsChapter = try! SolarSystem.loadSeasonsChapter()

// Add the seasons chapter anchor into the scene
arView.scene.anchors.append(seasonsChapter)
```
// Load the "Seasons Chapter" scene from the "SolarSystem" Reality File
let seasonsChapter = try! SolarSystem.loadSeasonsChapter()

// Add the seasons chapter anchor into the scene
arView.scene.anchors.append(seasonsChapter)
Code Generation in Xcode

SolarSystem.rcproject → Build in Xcode → MyExperience.app
```swift
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let anchor = try? Entity.loadAnchor(contentsOf: url, withName: "SeasonsChapter")

// Use the loaded anchor here...
```
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let anchor = try? Entity.loadAnchor(contentsOf: url, withName: "SeasonsChapter")

// Use the loaded anchor here...
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let anchor = try? Entity.loadAnchor(contentsOf: url, withName: "SeasonsChapter")

// Use the loaded anchor here...
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let loadRequest = Entity.loadAnchorAsync(contentsOf: url, withName: "SeasonsChapter")
_ = loadRequest.sink(receiveCompletion: { completion in
  // Handle completion state
}, receiveValue: { anchor in
  // Use loaded anchor here
})
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let loadRequest = Entity.loadAnchorAsync(contentsOf: url, withName: "SeasonsChapter")
_ = loadRequest.sink(receiveCompletion: { completion in
    // Handle completion state
}, receiveValue: { anchor in
    // Use loaded anchor here
})
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let loadRequest = Entity.loadAnchorAsync(contentsOf: url, withName: "SeasonsChapter")
_ = loadRequest.sink(receiveCompletion: { completion in
    // Handle completion state
}, receiveValue: { anchor in
    // Use loaded anchor here
})
```swift
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let loadRequest = Entity.loadAnchorAsync(contentsOf: url, withName: "SeasonsChapter")
_ = loadRequest.sink(receiveCompletion: { completion in
    // Handle completion state
}, receiveValue: { anchor in
    // Use loaded anchor here
})
```
guard let url = Bundle.main.url(forResource: "SolarSystem", withExtension: "reality") else {
    return
}

let loadRequest = Entity.loadAnchorAsync(contentsOf: url, withName: "SeasonsChapter")
_ = loadRequest.sink(receiveCompletion: { completion in
    // Handle completion state
}, receiveValue: { anchor in
    // Use loaded anchor here
})
Reality File
Accessing entities

// Load Reality File anchor above...

let sun = anchor.findEntity(named: "Sun")
let earth = anchor.findEntity(named: "Earth")
let moon = anchor.findEntity(named: "Moon")

// Use fetched entities below...
Reality File
Accessing entities

// Load Reality File anchor above...

let sun = anchor.findEntity(named: "Sun")
let earth = anchor.findEntity(named: "Earth")
let moon = anchor.findEntity(named: "Moon")

// Use fetched entities below...
UI Applications and AR Content

Actions

Application Logic

SWIFT
UI Applications and AR Content

Actions

- Application Logic
- Notify Action

- Swift

Icons:
- Eye
- Cube
- Rotation
- Pages
- Video
- Sound
- Notification
Notify Action

Set up in Reality Composer
Invoked in action sequences
Settable closure in application code
Accessible by name in code
Chapter Completed!

Notify

Affected Objects
1 Object

Identifier
Display Earth Details

Action will notify your application code using the name above when it occurs (Xcode)
Notify Action
Generated code structure

SolarSystem.swift

loadSizeChapter
Entity, HasAnchoring
loadSizeChapterAsync

SizeChapter

Entity
sun

Entity
earth

Entity
moon
Notify Action
Generated code structure

SolarSystem.swift

- loadSizeChapter
- Entity, HasAnchoring
- loadSizeChapterAsync

SizeChapter
- Entity
  - sun
- Entity
  - earth
- Entity
  - moon
- Actions
  - actions
Notify Action
Generated code structure
Notify Action
Defining closure

```javascript
seasonsChapter.actions.displayEarthDetails.onAction = { entity in
    // Display details about Fall
}
```
Notify Action

Defining closure

```javascript
seasonsChapter.actions.displayEarthDetails.onAction = { entity in
  // Display details about Fall
}
```
UI Applications and AR Content

Triggers

Application Logic

SWIFT
UI Applications and AR Content

Triggers

- Application Logic Notification Trigger

- SWIFT

- Application Logic
- Notification Trigger
Notification Trigger

Set up in Reality Composer
Starts action sequences
Posted from the application
Accessible by name in code
Chapter Completed!

Action Sequence begins when the above notification is posted from your application code (Xcode).
Notification Trigger
Generated code structure

SolarSystem.swift

- loadSizeChapter
- Entity, HasAnchoring
- loadSizeChapterAsync

SizeChapter

- Entity
  - sun
- Entity
  - earth
- Entity
  - moon
- Actions
  - actions
Notification Trigger
Generated code structure

SolarSystem.swift

- loadSizeChapter
- Entity, HasAnchoring
- loadSizeChapterAsync

SizeChapter

- Entity
  - sun
- Entity
  - earth
- Entity
  - moon
- Actions
  - actions
- Notifications
  - notifications
Notification Trigger
Generated code structure

Notifications

notifications

NotificationTrigger
sizeToRelativeSize

NotificationTrigger
chapterCompleted

[NotificationTrigger]
allNotifications
Notification Trigger
Starting an action sequence

```javascript
seasonsChapter.notifications.showGoldStar.post()

// Replaces targets in the action sequence named `originalTarget.name` with `newTarget`
seasonsChapter.notifications.showGoldStar.post(overrides: [originalTarget.name: newTarget])
```
Notification Trigger
Starting an action sequence

seasonsChapter.notifications.showGoldStar.post()

// Replaces targets in the action sequence named `originalTarget.name` with `newTarget`  
seasonsChapter.notifications.showGoldStar.post(overrides: [originalTarget.name: newTarget])
Notification Trigger

Starting an action sequence

```javascript
seasonsChapter.notifications.showGoldStar.post()
```

```javascript
// Replaces targets in the action sequence named `originalTarget.name` with `newTarget`
seasonsChapter.notifications.showGoldStar.post(overrides: [originalTarget.name: newTarget])
```
Demo
Reality Composer

Get started with AR and 3D
Layout and pre-visualization
Content library
Simple interactions
Xcode integration
## More Information

developer.apple.com/wwdc19/609

<table>
<thead>
<tr>
<th>Event</th>
<th>Date</th>
</tr>
</thead>
<tbody>
<tr>
<td>RealityKit and Reality Composer Lab</td>
<td>Thursday, 3:00</td>
</tr>
<tr>
<td>Advances in AR Quick Look</td>
<td>Friday, 9:00</td>
</tr>
<tr>
<td>AR Quick Look and Reality Composer Lab</td>
<td>Friday, 11:00</td>
</tr>
</tbody>
</table>