Going Social with ReplayKit and Game Center

What’s new in social gaming

Session 605

Edwin Iskandar Software Engineer
Megan Gardner Software Engineer
Agenda

What’s new in Game Center
Agenda

What’s new in Game Center

• Guest players
Agenda

What’s new in Game Center

• Guest players
• Unified server environment
Agenda

What’s new in Game Center
• Guest players
• Unified server environment

Introducing ReplayKit
Agenda

What’s new in Game Center
• Guest players
• Unified server environment

Introducing ReplayKit
• New way to add social to your games
Agenda

What’s new in Game Center
• Guest players
• Unified server environment

Introducing ReplayKit
• New way to add social to your games
• Share game experiences
What’s New in Game Center
Game Center

Social gaming

Friends
Leaderboards
Achievements
Real time multiplayer
Turn based multiplayer
Challenges
What’s New?

Guest players
Unified environment
Guest Players

Adding more to your multiplayer
Players

Current usage

GKLocalPlayer

• Only one per device

• Must be an authenticated Game Center user
Players

Current usage

GKLocalPlayer

- Only one per device
- Must be an authenticated Game Center user

GKPlayer

- Friend of the GKLocalPlayer
- Participant in a match
Players

Current usage

GKLocalPlayer
- Only one per device
- Must be an authenticated Game Center user

GKPlayer
- Friend of the GKLocalPlayer
- Participant in a match

Multiplayer
- Real time
- Turn based
Guest Players

What are they?

GKPlayer instances

• Do not require authentication
• First class participants in multiplayer games

Can fill up all the slots in a game

• Up to three for real time
• Up to fifteen for turn based
Guest Players
What you need to know

Great for pass and play and handling AI players
You define the identifier—needs to be unique across the game
Cannot earn achievements or post scores
Sessions with guest players only compatible with other players running iOS 9
• Game still compatible with players on older systems
Game Center Sandbox Environment
Unification with production
Sandbox

Original purpose

Environment for pre-release testing with Game Center

- Duplicated production Game Center functionality
- Same iTunes Connect metadata
- Different servers
- Different accounts
Unified Environment

Benefits

Simplifies accounts
Compatible with TestFlight
Multiplayer works across versions (if enabled)
Automatic
Unified Environment
Considerations

New games appear in friend’s games list
• Including unreleased
Scores post to existing leaderboards
iOS 8 users still need to flip the switch
Leaderboards

All versions will post to the same leaderboards
New leaderboard only visible to those who have that version
Remove test scores via iTunes Connect
Multiplayer

Play against any other version, specified in the compatibility matrix
Release versions can match against unreleased versions
iTunes Connect

Up to two sets of metadata will be maintained

• Currently released version

• Unreleased version
Up to two sets of metadata will be maintained
• Currently released version
• Unreleased version

Which data you are vended as a user will depend on your CFBundleVersion
• CFBundleVersion > Released Bundle Version—unreleased data
• CFBundleVersion <= Released Bundle Version—currently released data
### Post-Sandbox
**Compatibility and visibility**

<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>0.9 “Beta”</td>
<td>Game Center Friends</td>
<td>Other beta versions</td>
<td>Beta leaderboards</td>
</tr>
<tr>
<td>1.0 “Released”</td>
<td>Game Center Friends</td>
<td>All versions, including future updates, if specified</td>
<td>Released leaderboards</td>
</tr>
<tr>
<td>2.0 “Update”</td>
<td>Game Center Friends</td>
<td>All specified versions</td>
<td>Released leaderboards and update leaderboards, not viewable by Game Center friends</td>
</tr>
</tbody>
</table>
Recap

What’s new

Guest players

• Expands multiplayer opportunity
Recap
What’s new

Guest players
• Expands multiplayer opportunity

Unified environment
• Fulfills developers requests
• Simple and automatic
• Works great with TestFlight
ReplayKit
Sharing game experiences

Edwin Iskandar  Software Engineer
ReplayKit
ReplayKit

Record your running app
ReplayKit

Record your running app
Add voice commentary
ReplayKit

Record your running app
Add voice commentary
Playback, scrub, and trim
ReplayKit

Record your running app
Add voice commentary
Playback, scrub, and trim
Share
  • Social networks
  • Video destination sites
ReplayKit

HD quality
  • Low performance impact
  • Minimal power usage
ReplayKit

HD quality
- Low performance impact
- Minimal power usage

Privacy safeguards
ReplayKit

HD quality
• Low performance impact
• Minimal power usage
Privacy safeguards
Available in iOS 9
ReplayKit

HD quality
• Low performance impact
• Minimal power usage
Privacy safeguards
Available in iOS 9
A7 and A8 based devices
Privacy Safeguards

Allow screen recording in "DemoBots"?
You can save the recording to the camera roll or share it with friends. Record microphone audio if you want to add comments as you go.

Record Screen & Microphone

Record Screen Only

Don’t Allow
Privacy Safeguards

Permission required

• User consent prompt
• Parental controls

Allow screen recording in "DemoBots"?
You can save the recording to the camera roll or share it with friends. Record microphone audio if you want to add comments as you go.

- Record Screen & Microphone
- Record Screen Only
- Don’t Allow
Privacy Safeguards

Permission required
- User consent prompt
- Parental controls

Recording excludes system UI
- Notifications
- Keyboard entry
Privacy Safeguards

Permission required
• User consent prompt
• Parental controls
Recording excludes system UI
• Notifications
• Keyboard entry
No direct access to recordings
• Share sheet only

Allow screen recording in "DemoBots"?
You can save the recording to the camera roll or share it with friends. Record microphone audio if you want to add comments as you go.

- Record Screen & Microphone
- Record Screen Only
- Don’t Allow
Architecture
Application
Using ReplayKit

Getting started
Classes and Protocols
Classes and Protocols

RPScreenRecorder

• Start, stop, discard recording
• Check ability to record
• Enable microphone for commentary
Classes and Protocols

RPScreenRecorder
- Start, stop, discard recording
- Check ability to record
- Enable microphone for commentary

RPScreenRecorderDelegate
- If availability changes
- If recording stops (due to error)
Classes and Protocols

RPScreenRecorder
- Start, stop, discard recording
- Check ability to record
- Enable microphone for commentary

RPPreviewViewController
- Preview the recording
- Edit and trim
- Share

RPScreenRecorderDelegate
- If availability changes
- If recording stops (due to error)
Classes and Protocols

**RPScreenRecorder**
- Start, stop, discard recording
- Check ability to record
- Enable microphone for commentary

**RPPreviewViewController**
- Preview the recording
- Edit and trim
- Share

**RPScreenRecorderDelegate**
- If availability changes
- If recording stops (due to error)

**RPPreviewViewControllerDelegate**
- After view controller dismissal
Example

DemoBots
DemoBots

Taskbots keep the circuit board running
Bots are buggy, debug them
Bite size 2–3 minute levels
Source available at developer.apple.com
Main Menu
Main Menu -> Game Level -> Level End
class RPScreenRecorder : NSObject {
    class func sharedRecorder() -> RPScreenRecorder
}

let sharedRecorder = RPScreenRecorder.sharedRecorder()
class RPScreenRecorder: NSObject {
    class func sharedRecorder() -> RPScreenRecorder
}

let sharedRecorder = RPScreenRecorder.sharedRecorder()
Start Recording

class RPScreenRecorder : NSObject {
    func startRecordingWithMicrophoneEnabled(microphoneEnabled: Bool, handler: ((NSError?) -> Void)?) {

    }

    func levelDidStart() {
        sharedRecorder.startRecordingWithMicrophoneEnabled(true) { (error:) in
            if error != nil {
                // pause game and show error
            }
        }
    }
}
Start Recording

class RPScreenRecorder : NSObject {
    func startRecordingWithMicrophoneEnabled(microphoneEnabled: Bool, 
       handler: ((NSError?) -> Void)?)

    func levelDidStart() {
        sharedRecorder.startRecordingWithMicrophoneEnabled(true) { (error:) in
            if error != nil {
                // pause game and show error
            }
        }
    }
}
Allow screen recording in "DemoBots"?

You can save the recording to the camera roll or share it with friends. Record microphone audio if you want to add comments as you go.

- Record Screen & Microphone
- Record Screen Only
- Don't Allow

NEW GAME
Allow screen recording in "DemoBots"?

You can save the recording to the camera roll or share it with friends. Record microphone audio if you want to add comments as you go.

- Record Screen & Microphone
- Record Screen Only
- Don’t Allow

NEW GAME
class RPScreenRecorder : NSObject {
    func stopRecordingWithHandler(
        handler: ((RPPreviewViewController?, NSError?) -> Void)?)

    func levelWillEnd() {
        sharedRecorder.stopRecordingWithHandler {
            (previewViewController, error) -> Void in

                // Handle error
                if previewViewController != nil {
                    // Keep a reference for later use
                    self.previewViewController = previewViewController
                }
            }
        }
    }
}
class RPScreenRecorder : NSObject {
    func stopRecordingWithHandler(
        handler: ((RPPreviewViewController?, NSError?) -> Void)?) {
        sharedRecorder.stopRecordingWithHandler {
            (previewViewController, error) -> Void in

                // Handle error
                if previewViewController != nil {
                    // Keep a reference for later use
                    self.previewViewController = previewViewController
                }
            }
        }
    }
}

func levelWillEnd() {
    sharedRecorder.stopRecordingWithHandler {
        (previewViewController, error) -> Void in

            // Handle error
            if previewViewController != nil {
                // Keep a reference for later use
                self.previewViewController = previewViewController
            }
        }
    }
}
LEVEL COMPLETE!

NEXT LEVEL
REPLAY LEVEL
QUIT
class UIViewController: ... {
    func presentViewController(viewControllerToPresent: UIViewController,
        animated flag: Bool, completion: (() -> Void)?)
}

@IBAction func didPressViewRecordingButton() {

    // Present the preview view controller we made a reference to before
    self.presentViewController(previewViewController, animated: true,
        completion:nil)
}
Present Preview UI

class UIViewController : ... {
    func presentViewController(viewControllerToPresent: UIViewController,
        animated flag: Bool, completion: (() -> Void)?)

@IBAction func didPressViewRecordingButton() {

    // Present the preview view controller we made a reference to before
    self.presentViewController(previewViewController, animated: true,
        completion:nil)
}

Finally beat that level! 😁😁😁
protocol RPPreviewViewControllerDelegate : NSObjectProtocol {

    optional func previewControllerDidFinish(
        previewController: RPPreviewViewController)

    @IBAction func didPressViewRecordingButton() {
        previewViewController.previewViewControllerDelegate = self
    }

    func previewControllerDidFinish(previewController: RPPreviewViewController) {
        previewViewController.dismissViewControllerAnimated(true, completion: nil)
    }
}
protocol RPPreviewViewControllerDelegate : NSObjectProtocol {

    optional func previewControllerDidFinish(
        previewController: RPPreviewViewController)

}@IBAction func didPressViewRecordingButton() {
    previewViewController.previewViewControllerDelegate = self
}

func previewControllerDidFinish(previewController: RPPreviewViewController) {
    previewViewController.dismissViewControllerAnimated(true, completion: nil)
Using ReplayKit

Getting started

Getting a shared instance of the recorder
Starting and stopping the recorder
Presenting and dismissing the Preview UI
Using ReplayKit

Fine tuning
Verifying Availability
Verifying Availability

Recording may be unavailable

- AirPlay in use
- TV-out in use
- Unsupported device
Verifying Availability

Recording may be unavailable
• AirPlay in use
• TV-out in use
• Unsupported device

Guidance
• Use **available** property to check for availability
• Disable recording UI if false
• Use **screenRecorderDidChangeAvailability** to listen for changes
class RPScreenRecorder : NSObject {
    var available: Bool { get }

    func updateButtonUI() {
        recordingToggleButton.hidden = !sharedRecorder.available
    }
}
Verifying Availability

Example

class RPScreenRecorder : NSObject {
    var available: Bool { get }
}

func updateButtonUI() {
    recordingToggleButton.hidden = !sharedRecorder.available
}
Discarding the Recording

Automatically discarded when new recording starts

• Only one recording at a time per app

Discard when preview no longer accessible

• Use discardRecordingWithHandler
Discarding the Recording

Automatically discarded when new recording starts

- Only one recording at a time per app

Discard when preview no longer accessible

- Use `discardRecordingWithHandler`
Discarding the Recording

Example

class RPScreenRecorder : NSObject {
    func discardRecordingWithHandler(handler: () -> Void)

    func willTransitionToNextLevel() {
        sharedRecorder.discardRecordingWithHandler {
            // start next level
            self.transitionToNextLevel()
        }
    }
}
class RPScreenRecorder : NSObject {
    func discardRecordingWithHandler(handler: () -> Void)

    func willTransitionToNextLevel() {
        sharedRecorder.discardRecordingWithHandler {
            // start next level
            self.transitionToNextLevel()
        }
    }
}
Recording Indicator

Indicate that recording is on
Indicate that microphone is on
class RPScreenRecorder: NSObject {

    var recording: Bool { get }
    var microphoneEnabled: Bool { get }

    func recordingDidStart() {
        recordingIndicator.hidden = !sharedRecorder.recording
        microphoneIndicator.hidden = !sharedRecorder.microphoneEnabled
    }
}
class RPScreenRecorder : NSObject {

var recording: Bool { get }
var microphoneEnabled: Bool { get }

func recordingDidStart() {
    recordingIndicator.hidden = !sharedRecorder.recording
    microphoneIndicator.hidden = !sharedRecorder.microphoneEnabled
}
Excluding UI
Excluding UI

Hide elements that are uninteresting to a spectator

• Recording indicators
• Virtual controls
• Pause and menu buttons
Excluding UI

Hide elements that are uninteresting to a spectator

• Recording indicators
• Virtual controls
• Pause and menu buttons

ReplayKit only records your application’s main UIWindow

• Use a separate UIWindow to hide UI
When to Record
Automatic vs. user-initiated
When to Record
Automatic vs. user-initiated

App controlled (automatic)
• Short gameplay sessions
When to Record

Automatic vs. user-initiated

App controlled (automatic)
• Short gameplay sessions

User-initiated
• Longer gameplay sessions
• Spread out interesting events
When to Record
Automatic vs. user-initiated

App controlled (automatic)
• Short gameplay sessions

User-initiated
• Longer gameplay sessions
• Spread out interesting events

Choose what is appropriate
Using ReplayKit

Getting Started
• Getting a shared instance of the recorder
• Starting and stopping the recorder
• Presenting and dismissing the Preview UI

Fine Tuning
• Verifying availability
• Discarding the recording
• Showing indicator
• Excluding UI
• When to record
Demo
Call of Champions

Cinco Barnes
Chief Vision Officer, Spacetime Studios
What’s New in Social Gaming
What’s New in Social Gaming

New in Game Center

• Guest players allow for new modes in multiplayer
• Unified server environment to streamline development and testing
What’s New in Social Gaming

New in Game Center

• Guest players allow for new modes in multiplayer
• Unified server environment to streamline development and testing

Introducing ReplayKit

• Records audio and visuals of running application
• Efficient with privacy safeguards
• Compact framework with simple adoption
More Information

Documentation and Videos
http://developer.apple.com

Apple Developer Forums
http://developer.apple.com/forums

Developer Technical Support
http://developer.apple.com/support/technical

General Inquiries
Allan Schaffer, Game Technologies Evangelist
aschaffer@apple.com
<table>
<thead>
<tr>
<th>Related Labs</th>
<th>Graphics, Games, and Media Lab D</th>
<th>Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>ReplayKit Lab</td>
<td></td>
<td>Wednesday 2:30PM</td>
</tr>
<tr>
<td>ReplayKit Lab</td>
<td></td>
<td>Thursday 9:00AM</td>
</tr>
</tbody>
</table>