

[GameKit](#) / GKGameActivity

## Class

# GKGameActivity

An object that represents a single instance of a game activity for the current game.

iOS 26.0+ | iPadOS 26.0+ | Mac Catalyst 26.0+ | macOS 26.0+ | tvOS 26.0+ | visionOS 26.0+

```
class GKGameActivity
```

## Mentioned in

 [Creating activities for your game](#)

## Topics

### Creating an activity

```
init(definition: GKGameActivityDefinition)
```

Creates a game activity with definition.

```
class func start(definition: GKGameActivityDefinition) throws -> GKGameActivity
```

Creates and starts a game activity with a definition.

```
class func start(definition: GKGameActivityDefinition, partyCode: String) throws -> GKGameActivity
```

Creates and starts a new game activity with a custom party code.

## Getting the activity definition

```
var activityDefinition: GKGameActivityDefinition
```

The activity definition that this activity instance is based on.

## Getting the activity state

```
var state: GKGameActivity.State
```

The state of the game activity.

```
enum State
```

The state of a game activity.

## Updating the activity state

```
func start()
```

Starts the game activity if it's not already started.

```
func pause()
```

Pauses the game activity if it's not already paused.

```
func resume()
```

Resumes the game activity if it was paused.

```
func end()
```

Ends the game activity if it's not already ended.

## Getting and removing achievements

```
var achievements: Set<GKAchievement>
```

All achievements that have been associated with this activity.

```
func removeAchievements([GKAchievement])
```

Removes all achievements if they exist.

```
func progress(on: GKAchievement) -> Double
```

Get the achievement progress from a specific achievement of the local player if previously set.

```
func setProgress(on: GKAchievement, to: Double)
```

Set a progress for an achievement for a player.

```
func setAchievementCompleted(GKAchievement)
```

Set progress to 100% for an achievement for a player.

## Getting and removing leaderboard scores

```
var leaderboardScores: Set<GKLeaderboardScore>
```

All leaderboard scores that have been associated with this activity.

```
func score(on: GKLeaderboard) -> GKLeaderboardScore?
```

Get the leaderboard score from a specific leaderboard of the local player if previously set.

```
func setScore(on: GKLeaderboard, to: Int)
```

Set a score of a leaderboard for a player.

```
func setScore(on: GKLeaderboard, to: Int, context: Int)
```

Set a score of a leaderboard with a context for a player.

```
func removeScores(from: [GKLeaderboard])
```

Removes all scores from leaderboards for a player if exist.

## Getting and verifying the party code

```
var partyCode: String?
```

If the game supports party code, this is the party code that can be shared among players to join the party.

```
var partyURL: URL?
```

If the game supports party code, this is the URL that can be shared among players to join the party.

```
class var validPartyCodeAlphabet: [String]
```

Allowed characters for the party code to be used to share this activity.

```
class func isValidPartyCode(String) -> Bool
```

Checks whether a party code is in valid format.

## Getting the activity properties

```
var duration: TimeInterval
```

The total time elapsed while in active state.

```
var startDate: Date?
```

The date when the activity was initially started.

```
var endDate: Date?
```

The date when the activity was officially ended.

```
var creationDate: Date
```

The date when the activity was created.

```
var lastResumeDate: Date?
```

The date when the activity was last resumed.

## Getting the custom user data

```
var properties: [String : String]
```

Properties that contain additional information about the activity.

## Getting the activity identifiers

```
var identifier: String
```

The identifier of this activity instance.

## Checking for an activity

```
class func checkPendingGameActivityExistence(completionHandler: (Bool) -> Void)
```

Checks whether there is a pending activity to handle for the current game.

## Creating a matchmaking request

```
func makeMatchRequest() -> GKMatchRequest?
```

Makes a match request object with information from the activity, which you can use to find matches for the local player.

## Performing a matchmaking request

```
func findMatch(completionHandler: (GKMatch?, (any Error)?) -> Void)
```

Use information from the activity to find matches for the local player.

```
func findPlayersForHostedMatch(completionHandler: ([GKPlayer]?, (any Error)?) -> Void)
```

Use information from the activity to find server hosted players for the local player.

---

## Relationships

### Inherits From

NSObject

### Conforms To

CVarArg

CustomDebugStringConvertible

CustomStringConvertible

Equatable

Hashable

NSObjectProtocol

Sendable

SendableMetatype

---

## See Also

### Activities



Creating activities for your game

Use activities to surface game content to players and encourage them to connect with each other.

```
class GKGameActivityDefinition
```

An object that represents the static metadata you define for the activity.

```
protocol GKGameActivityListener
```

An object that responds to activity events.