Meta Wearables Login Register

API reference

Wearables Object

Main entry point for the Wearables Device Access Toolkit. Provides registration, device management, permissions, and session state functionality. Must be initialized before use via Wearables.initialize.

Signature

```
object Wearables
```

Properties

devices : StateFlow<Set< DeviceIdentifier> > [Get]	A StateFlow that provides the devices that can be potentially used by the Wearables SDK. **Signature** `val devices: StateFlow<Set<DeviceIdentifier>>`
devicesMetadata : Map [Get]	A map from DeviceIdentifier to StateFlow that provides device metadata. **Signature** `val devicesMetadata: Map`

StateFlow<RegistrationState>
[Get]

Signature

```
val registrationState:
StateFlow<RegistrationState>
```

Functions

checkPermissionStatus (permission)

Checks whether the specified device permission is granted on any linked device.
Since permissions are managed via constellation, this function returns the first valid permission grant state found among the linked devices.

Signature

```
suspend fun
checkPermissionStatus(permission:
Permission): PermissionStatus
```

Parameters

`permission:` `Permission` The device permission to check if exists on device

Returns

`PermissionStatus`PermissionStatus.Granted if permission is granted, PermissionStatus.Denied if denied, PermissionStatus.Error if errors occurs

Throws

`WearablesException` if SDK not initialized via Wearables.initialize

getDeviceSessionState (deviceIdentifier)

Gets the session state for a specific device.

Signature

```
fun
getDeviceSessionState(deviceIdentifier:
```

Parameters

`deviceIdentifier:` `DeviceIdentifier` The identifier of the device to get session state for

Returns

`StateFlow<SessionState>` StateFlow providing the session state for the specified device

Throws

`WearablesException` if SDK not initialized via Wearables.initialize

initialize (context)

Initializes the Wearables SDK with the provided context. This method must be called before using any other SDK functionality.

Signature

```
fun initialize(context: Context)
```

Parameters

`context: Context` The Android context used to initialize the SDK

reset ()

Resets the Wearables SDK instance.

Signature

```
fun reset()
```

startRegistration (context)

Initiates the registration process with AI glasses.
This method opens the Meta AI app where the user completes the registration flow. When the user completes the flow, the result will be automatically processed and the registration state will be updated.
The Wearables.registrationState will be updated throughout the registration process.

Signature

Parameters

`context: Context` The Android context used to start the registration activity

Throws

`WearablesException` if SDK not initialized via Wearables.initialize

startStreamSession (context , deviceSelector , streamConfiguration)

Creates and starts a new streaming session with the specified device selector and configuration.
The session is automatically started and will attempt to connect to the device selected by the device selector. Video frames are delivered through the StreamSession.videoStream Flow.

Signature

```
fun
Wearables.startStreamSession(context:
Context, deviceSelector: DeviceSelector,
streamConfiguration: StreamConfiguration
= StreamConfiguration()): StreamSession
```

Parameters

`context: Context` The Android context for the streaming session
`deviceSelector: DeviceSelector` The device selector to determine which device to stream from
`streamConfiguration: StreamConfiguration` The configuration for the streaming session (optional, defaults to default configuration)

Returns

`StreamSession` A new StreamSession instance that is already started

startUnregistration (context)

Initiates the unregistration process with AI glasses.
This method opens the Meta AI app where the user completes the unregistration flow. When the user completes the flow, the result will be automatically processed and the registration state will be updated.
The Wearables.registrationState will be updated throughout the unregistration process.

```
fun startRegistration(context:
Context)
```

Parameters

`context: Context` The Android context used to start registration activity

Throws

`WearablesException` if SDK not initialized via Wearables.initialize

Inner Class

RequestPermissionContract Class

Modifiers: final

An ActivityResultContract for requesting permissions from AI glasses through the Meta AI companion application.

This contract handles the process of launching the Meta AI app with a permission request and parsing the user's response. It creates the appropriate intent to open the permission request flow and processes the result to determine the permission grant state.

Example usage:

```
private val permissionsResultLauncher =
    registerForActivityResult(Wearables.RequestPermissionContract()) { result -
>
        // Handle the permission result
        when (result) {
            PermissionStatus.GRANTED -> // Permission granted
            PermissionStatus.DENIED -> // Permission denied
        }
    }
```

Signature

```
class RequestPermissionContract
```

Constructors

```
constructor()
```

Returns

`Wearables.RequestPermissionContract`

Functions

createIntent (context , permission)

Signature

```
open fun createIntent(context: Context,
permission: Permission): Intent
```

Parameters

`context: Context`
`permission: Permission`

Returns

`Intent`

getSynchronousResult (context , input)

Signature

```
open fun getSynchronousResult(context:
Context, input: Permission):
SynchronousResult<PermissionStatus>?
```

Parameters

`context: Context`
`input: Permission`

Returns

`SynchronousResult<PermissionStatus>?`

parseResult (resultCode , intent)

Signature

Parameters

```
resultCode: Int
intent: Intent?
```

Returns

`PermissionStatus`