API Collection

ARKit in visionOS C API

Integrate ARKit with low-level libraries and functionality.

Overview

ARKit in visionOS includes a full C API for compatibility with C and Objective-C apps and frameworks.

Topics

Sessions

`ar_session_t`

The main entry point for receiving data from ARKit.

`ar_session_create`

Creates a new session.

`ar_session_create_with_device`

Create a session connected to the specified device.

`ar_session_query_authorization_results`

Checks whether the current session is authorized for particular authorization types without requesting authorization.

`ar_session_query_authorization_results_f`

Checks whether the current session is authorized for particular authorization types without requesting authorization.

`ar_session_request_authorization`

Requests authorization from the user to use the specified kinds of ARKit data.

`ar_session_request_authorization_f`

Requests authorization from the user to use the specified kinds of ARKit data.

`ar_session_run`

Runs a session with the data providers you supply.

`ar_session_set_authorization_update_handler`

Sets the handler for receiving updates in authorization status for a specific authorization type.

`ar_session_set_authorization_update_handler_f`

Sets the handler for receiving updates in authorization status for a specific authorization type.

`ar_session_copy_data_providers`

Get a copy of the collection of all data providers on this session.

`ar_session_set_data_provider_state_change_handler`

Sets the handler for responding to a state change of one or more data providers.

`ar_session_set_data_provider_state_change_handler_f`

Sets the handler function for responding to a state change of one or more data providers.

`ar_session_data_provider_state_change_handler_t`

A handler that the session calls when one or more data providers associated with it change state.

`ar_session_stop`

Stops a session.

`typealias ar_device_t`

Memory Management

`ar_release`

Releases a reference count on an ARKit object.

`ar_retain`

Adds a reference count to an ARKit object.

Authorization

`ar_authorization_status_t`

The authorization states for a type of ARKit data.

`enum AuthorizationStatus`

The authorization states for a type of ARKit data.

`ar_authorization_type_t`

The authorization types you can request from ARKit.

`ar_authorization_result_get_authorization_type`

Gets the authorization type associated with an authorization result.

`ar_authorization_result_get_status`

Gets the authorization status associated with an authorization result.

`ar_authorization_results_enumerate_results`

Enumerates a collection of authorization results.

`ar_authorization_results_enumerate_results_f`

Enumerates a collection of authorization results.

`ar_authorization_results_get_count`

Gets the number of authorization results in a collection.

`ar_authorization_status_allowed`

Your app has permission to use the associated kind of ARKit data.

`ar_authorization_status_denied`

Your app doesn’t have permission to use the associated kind of ARKit data.

`ar_authorization_status_not_determined`

Permission for your app to use the associated kind of ARKit data is undetermined.

`ar_authorization_result_t`

An authorization result.

`ar_authorization_results_enumerator_t`

A handler for enumerating a collection of authorization results.

`ar_authorization_results_handler_t`

A handler to call upon completion of an authorization request.

`ar_authorization_results_t`

A collection of authorization results.

`ar_authorization_update_handler_t`

A handler for receiving updates in authorization status for a specific authorization type.

`ar_authorization_results_handler_function_t`

`ar_authorization_update_handler_function_t`

`ar_authorization_results_enumerator_function_t`

Anchors

`ar_anchor_get_identifier`

Gets the unique identifier that distinguishes this anchor from all other anchors.

`ar_anchor_get_timestamp`

Gets the timestamp corresponding to the anchor.

`ar_anchor_get_origin_from_anchor_transform`

Gets the transform from the anchor to the origin coordinate system.

`ar_trackable_anchor_is_tracked`

Returns a Boolean value that indicates whether ARKit is tracking an anchor.

`ar_anchor_t`

The identity, location, and orientation of an object in world space.

`ar_mesh_anchor_t`

A surface’s position in a person’s surroundings.

`ar_mesh_anchors_t`

A collection of mesh anchors.

`ar_mesh_anchors_enumerator_t`

A handler for enumerating a collection of mesh anchors.

`ar_image_anchor_t`

A 2D image’s position in a person’s surroundings.

`ar_image_anchors_t`

A collection of image anchors.

`ar_image_anchors_enumerator_t`

A handler for enumerating a collection of image anchors.

`ar_hand_anchor_t`

A hand's position in a person's surroundings.

`ar_trackable_anchor_t`

An anchor that can gain and lose its tracking state over the course of a session.

`ar_world_anchor_t`

A fixed location in a person's surroundings.

`ar_world_anchors_t`

A collection of world anchors.

`ar_world_anchors_enumerator_t`

A handler for enumerating a collection of world anchors.

`ar_world_anchor_sharing_availability_t`

Enumeration indicating the availability of world anchor sharing.

`ar_plane_anchor_t`

An anchor that represents horizontal and vertical planes.

`ar_plane_anchors_t`

A collection of plane anchors.

`ar_plane_anchors_enumerator_t`

A handler for enumerating a collection of plane anchors.

`ar_plane_detection_provider_t`

A source of live data about planes in a person's surroundings.

`ar_hand_skeleton_joint_name_t`

An enumeration that describes hand joint names.

`ar_hand_anchor_query_status_t`

An enumeration that describes the status of a hand anchor query.

Data providers

`ar_data_provider_state_t`

The possible states of a data provider.

`ar_data_provider_get_required_authorization_type`

The kinds of authorization you need to use a particular data provider type.

`ar_data_provider_get_state`

Gets the current status of data coming from this provider.

`ar_data_providers_enumerator_t`

A handler for enumerating a collection of data providers.

`ar_data_providers_t`

A collection of data providers.

`ar_data_providers_add_data_provider`

Adds a data provider to a collection.

`ar_data_providers_add_data_providers`

Adds multiple data providers to a collection.

`ar_data_providers_create`

Creates an empty collection of data providers.

`ar_data_providers_create_with_data_providers`

Creates a collection of data providers that contains the data providers you supply.

`ar_data_providers_enumerate_data_providers_f`

Enumerates a collection of data providers.

`ar_data_providers_get_count`

Gets the number of data providers in a collection.

`ar_data_providers_enumerate_data_providers`

Enumerates a collection of data providers.

`ar_data_providers_remove_data_provider`

Removes a data provider from a collection.

`ar_data_providers_remove_data_providers`

Removes multiple data providers from a collection.

`ar_data_providers_enumerator_function_t`

`ar_session_data_provider_state_change_handler_function_t`

A handler function that the session calls when one or more data providers associated with it change state.

`ar_data_provider_t`

A source of live data from ARKit.

Geometry

`ar_geometry_primitive_type_t`

The kinds of geometry primitives that a geometry element can contain.

`ar_geometry_element_get_buffer`

Gets a Metal buffer that contains index data that defines the geometry of an object.

`ar_geometry_element_get_bytes_per_index`

Gets the number of bytes that represent an index value.

`ar_geometry_element_get_count`

Gets the number of primitives in the Metal buffer for a geometry element.

`ar_geometry_element_get_index_count_per_primitive`

Gets the number of indices for each primitive.

`ar_geometry_element_get_primitive_type`

Gets the kind of primitive, lines or triangles, that a geometry element contains.

`ar_geometry_source_get_buffer`

Gets a Metal buffer that contains per-vector data for a geometry source.

`ar_geometry_source_get_components_per_vector`

Gets the number of scalar components in each vector in a geometry source.

`ar_geometry_source_get_count`

Gets the number of vectors in a geometry source.

`ar_geometry_source_get_format`

Gets the vertex format for data in a geometry source’s buffer.

`ar_geometry_source_get_offset`

Gets the offset, in bytes, from the beginning of a geometry source’s buffer.

`ar_geometry_source_get_stride`

Gets the number of bytes between one vector and another in a geometry source’s buffer.

`ar_geometry_primitive_type_line`

Two vertices that connect to form a line.

`ar_geometry_primitive_type_triangle`

Three vertices that connect to form a triangle.

`ar_geometry_element_t`

A container for vertex indices of lines or triangles.

`ar_geometry_source_t`

A container for geometrical vector data.

`ar_mesh_geometry_t`

The shapes that make up a mesh anchor.

`ar_mesh_anchors_enumerator_function_t`

Plane detection

`ar_plane_alignment_t`

The kinds of alignment — horizontal or vertical — that a plane anchor can have.

`ar_plane_classification_t`

The kinds of object classification a plane anchor can have.

`ar_plane_classification_ceiling`

A ceiling.

`ar_plane_classification_door`

A door.

`ar_plane_classification_floor`

A floor.

`ar_plane_classification_seat`

A seat.

`ar_plane_classification_status_not_available`

A plane classification is currently unavailable.

`ar_plane_classification_status_undetermined`

A plane classification is undetermined.

`ar_plane_classification_status_unknown`

A plane classification isn’t one of the known classes.

`ar_plane_classification_table`

A table.

`ar_plane_classification_wall`

A wall.

`ar_plane_classification_window`

A window.

`ar_plane_anchor_get_alignment`

Gets the alignment — horizontal or vertical — of a plane anchor relative to gravity.

`ar_plane_anchor_get_geometry`

Gets the shape of a plane anchor.

`ar_plane_anchor_get_plane_classification`

Gets the kind of real-world object that ARKit determines the plane anchor might be.

`ar_plane_anchors_enumerate_anchors`

Enumerates a collection of plane anchors using the collection of plane anchors and the enumeratin handler you provide.

`ar_plane_anchors_enumerate_anchors_f`

Enumerates a collection of plane anchors.

`ar_plane_anchors_get_count`

Gets the number of plane anchors in a collection.

`ar_plane_anchor_get_surface_classification`

Get the surface classification of a plane anchor.

`ar_plane_detection_configuration_t`

`ar_plane_detection_provider_create`

Creates a plane detection provider for the types of planes you want to detect.

`ar_plane_detection_provider_is_supported`

Returns a Boolean value that indicates whether the current runtime environment supports plane detection providers.

`ar_plane_geometry_get_plane_extent`

Gets the size of a plane.

`ar_plane_geometry_get_mesh_vertices`

Gets the vertices in the mesh that describes a plane.

`ar_plane_geometry_get_mesh_faces`

Gets the faces in the mesh that describes a plane.

`ar_plane_detection_provider_get_required_authorization_type`

Gets the types of authorizations necessary for detecting planes.

`ar_plane_detection_configuration_create`

Creates a plane detection configuration.

`ar_plane_detection_configuration_set_alignment`

Sets the plane alignments that you want the provider to detect.

`ar_plane_detection_provider_set_update_handler`

Sets the handler for receiving plane detection updates.

`ar_plane_detection_provider_set_update_handler_f`

Sets the handler for receiving plane detection updates using the plane detection provider and plane detection update queue you provide.

`ar_plane_extent_t`

The size of a plane.

`ar_plane_extent_get_height`

Gets the height of a plane.

`ar_plane_extent_get_plane_anchor_from_plane_extent_transform`

Gets the transform of a plane anchor from its extent transform.

`ar_plane_extent_get_width`

Gets the width of a plane.

`ar_plane_geometry_t`

The geometry of a plane anchor.

`ar_plane_detection_update_handler_t`

A handler for receiving updates to plane anchors.

`ar_plane_anchors_enumerator_function_t`

`ar_plane_detection_update_handler_function_t`

`ARPlaneClassificationStatus`

Possible states of ARKit's process for classifying plane anchors.

`ar_surface_classification_t`

A value describing the classification of a surface.

World tracking

`ar_world_anchor_create_with_origin_from_anchor_transform`

Creates a world anchor from a position and orientation in world space.

`ar_world_anchor_is_shared_with_nearby_participants`

Check if a world anchor is marked to be shared with nearby participants.

`ar_world_anchor_shared_with_nearby_participants_create`

Initialize a world anchor that should be shared with nearby participants.

`ar_world_anchor_sharing_availability_update_handler_function_t`

Function pointer called when there is a change in world anchor sharing availability.

`ar_world_anchor_sharing_availability_update_handler_t`

Handler called when there is a change in world anchor sharing availability.

`ar_world_tracking_add_anchor_completion_handler_t`

A handler to call upon completion of a request to add a world anchor.

`ar_world_tracking_remove_anchor_completion_handler_t`

A handler to call upon completion of a request to remove a world anchor.

`ar_world_tracking_remove_all_anchors_completion_handler_function_t`

Function called when a request to remove all known world anchors has completed (successfully or not).

`ar_world_tracking_remove_all_anchors_completion_handler_t`

Function called when a request to remove all known world anchors has completed (successfully or not).

`ar_world_tracking_anchor_update_handler_t`

A handler for receiving updates to world anchors.

`ar_world_tracking_configuration_create`

Creates a world tracking configuration.

`ar_world_anchors_enumerate_anchors`

Enumerates a collection of world anchors.

`ar_world_anchors_enumerate_anchors_f`

Enumerates a collection of world anchors.

`ar_world_anchors_get_count`

Gets the number of world anchors in the collection.

`ar_world_tracking_provider_create`

Creates a world tracking provider.

`ar_world_tracking_provider_is_supported`

Returns a Boolean value that indicates whether the current runtime environment supports world tracking providers.

`ar_world_tracking_provider_query_device_anchor_at_timestamp`

Queries the predicted pose of the current device at a given time.

`ar_world_tracking_provider_add_anchor`

Adds a world anchor you supply to the set of currently tracked anchors.

`ar_world_tracking_provider_add_anchor_f`

Adds a world anchor you supply to the set of currently tracked anchors.

`ar_world_tracking_provider_get_required_authorization_type`

Gets the types of authorizations required to track world anchors.

`ar_world_tracking_provider_set_anchor_update_handler`

Sets the handler for receiving world tracking updates.

`ar_world_tracking_provider_set_anchor_update_handler_f`

Sets the handler for receiving world tracking updates.

`ar_world_tracking_provider_remove_anchor_with_identifier`

Removes a world anchor from a world tracking provider based on its ID.

`ar_world_tracking_provider_remove_anchor`

Removes a world anchor from a world tracking provider.

`ar_world_tracking_provider_remove_anchor_f`

Removes a world anchor from a world tracking provider.

`ar_world_tracking_provider_remove_all_anchors`

Removes all known world anchors from the world tracking provider asynchronously.

`ar_world_tracking_provider_remove_all_anchors_f`

Removes all known world anchors from the world tracking provider asynchronously.

`ar_world_tracking_provider_remove_anchor_with_identifier_f`

Remove a world anchor from the world tracking provider using its identifier.

`ar_world_tracking_provider_set_world_anchor_sharing_availability_update_handler`

Set the handler for receiving world anchor sharing availability updates.

`ar_world_tracking_provider_set_world_anchor_sharing_availability_update_handler_f`

Set the function for receiving world anchor sharing availability updates

`ar_world_tracking_configuration_t`

`ar_device_anchor_t`

`ar_device_anchor_create`

`ar_device_anchor_query_status_t`

`ar_world_anchors_enumerator_function_t`

`ar_world_tracking_add_anchor_completion_handler_function_t`

`ar_world_tracking_anchor_update_handler_function_t`

`ar_world_tracking_remove_anchor_completion_handler_function_t`

`ar_world_tracking_provider_t`

A source of live data about the device pose and anchors in a person's surroundings.

`ar_camera_position_t`

An enumeration that describes possible camera positions.

`ar_camera_type_t`

An enumeration that describes camera types.

`ar_device_anchor_tracking_state_t`

Values that describe the tracking states of a device anchor.

`ar_world_tracking_error_code_t`

The error codes for errors that world tracking providers throw.

`ar_session_error_code_t`

The error codes for ARKit sessions.

Scene reconstruction

`ar_scene_reconstruction_configuration_t`

`ar_scene_reconstruction_mode_t`

The additional kinds of information you can request about a person's surroundings.

`ar_scene_reconstruction_provider_create`

Creates a provider that reconstructs the person's surroundings.

`ar_scene_reconstruction_provider_is_supported`

Returns a Boolean value that indicates whether the current runtime environment supports scene reconstruction providers.

`ar_scene_reconstruction_provider_get_required_authorization_type`

Gets the types of authorizations needed to run scene reconstruction.

`ar_scene_reconstruction_configuration_create`

Creates a scene reconstruction configuration.

`ar_scene_reconstruction_configuration_get_scene_reconstruction_mode`

Gets the scene reconstruction mode.

`ar_scene_reconstruction_configuration_set_scene_reconstruction_mode`

Sets the scene reconstruction mode.

`ar_scene_reconstruction_provider_set_update_handler`

Sets the handler for receiving scene reconstruction updates.

`ar_scene_reconstruction_provider_set_update_handler_f`

Sets the handler for receiving scene reconstruction updates.

`ar_mesh_classification_t`

The kinds of classification a mesh anchor can have.

`ar_mesh_anchor_get_geometry`

Gets the shape of a mesh anchor.

`ar_mesh_anchors_enumerate_anchors`

Enumerates a collection of mesh anchors.

`ar_mesh_anchors_enumerate_anchors_f`

Enumerates a collection of mesh anchors.

`ar_mesh_anchors_get_count`

Gets the number of mesh anchors in the collection.

`ar_mesh_geometry_get_classification`

Gets the classification of each face in the mesh.

`ar_mesh_geometry_get_faces`

Gets the faces of the mesh.

`ar_mesh_geometry_get_normals`

Gets the normals of the mesh.

`ar_mesh_geometry_get_vertices`

Gets the vertices of the mesh.

`ar_scene_reconstruction_provider_t`

A source of live data about the shape of a person’s surroundings.

`ar_scene_reconstruction_update_handler_t`

A handler for receiving updates to mesh anchors.

`ar_scene_reconstruction_update_handler_function_t`

Hand tracking

`ar_hand_chirality_t`

The values identifying hand chirality.

`ar_hand_anchor_create`

Creates a hand anchor.

`ar_hand_anchor_get_chirality`

Gets the value that indicates whether the hand is a left or right hand.

`ar_hand_anchor_get_fidelity`

Get the fidelity of the hand anchor.

`ar_hand_tracking_configuration_t`

`ar_hand_tracking_provider_create`

A source of live data about the position of a person's hands and hand joints.

`ar_hand_tracking_provider_is_supported`

Returns a Boolean value that indicates whether the current runtime environment supports hand tracking providers.

`ar_hand_tracking_provider_get_latest_anchors`

Fetches the most recent hand anchors for each hand.

`ar_hand_tracking_provider_get_required_authorization_type`

Gets the types of authorizations required to track hands.

`ar_hand_tracking_configuration_create`

`ar_hand_tracking_provider_set_update_handler`

Sets the handler for receiving hand tracking updates.

`ar_hand_tracking_provider_set_update_handler_f`

Sets the handler for receiving hand tracking updates.

`ar_hand_skeleton_joint_name_forearm_arm`

`ar_hand_skeleton_joint_name_forearm_wrist`

`ar_hand_skeleton_joint_name_index_finger_intermediate_base`

`ar_hand_skeleton_joint_name_index_finger_intermediate_tip`

`ar_hand_skeleton_joint_name_index_finger_knuckle`

`ar_hand_skeleton_joint_name_index_finger_metacarpal` `ar_hand_skeleton_joint_name_index_finger_tip`

`ar_hand_skeleton_joint_name_little_finger_intermediate_base`

`ar_hand_skeleton_joint_name_little_finger_intermediate_tip`

`ar_hand_skeleton_joint_name_little_finger_knuckle`

`ar_hand_skeleton_joint_name_little_finger_metacarpal`

`ar_hand_skeleton_joint_name_little_finger_tip`

`ar_hand_skeleton_joint_name_middle_finger_intermediate_base`

`ar_hand_skeleton_joint_name_middle_finger_intermediate_tip`

`ar_hand_skeleton_joint_name_middle_finger_knuckle`

`ar_hand_skeleton_joint_name_middle_finger_metacarpal`

`ar_hand_skeleton_joint_name_middle_finger_tip`

`ar_hand_skeleton_joint_name_ring_finger_intermediate_base`

`ar_hand_skeleton_joint_name_ring_finger_intermediate_tip`

`ar_hand_skeleton_joint_name_ring_finger_knuckle`

`ar_hand_skeleton_joint_name_ring_finger_metacarpal`

`ar_hand_skeleton_joint_name_ring_finger_tip`

`ar_hand_skeleton_joint_name_thumb_intermediate_base`

`ar_hand_skeleton_joint_name_thumb_intermediate_tip`

`ar_hand_skeleton_joint_name_thumb_knuckle`

`ar_hand_skeleton_joint_name_thumb_tip`

`ar_hand_skeleton_joint_name_wrist`

`ar_hand_chirality_left`

A left hand.

`ar_hand_chirality_right`

A right hand.

`ar_hand_tracking_provider_t`

A source of live data about the position of a person’s hands and hand joints.

`ar_hand_tracking_update_handler_t`

A handler for receiving updates to hand anchors.

`ar_hand_tracking_update_handler_function_t`

`ar_hand_fidelity_t`

Enum for hand fidelity.

Image tracking

`ar_image_anchor_get_estimated_scale_factor`

Gets the estimated scale factor between the tracked image's physical size and the reference image's size.

`ar_image_anchor_get_reference_image`

Gets the reference image that this image anchor tracks.

`ar_image_anchors_enumerate_anchors`

Enumerates a collection of image anchors.

`ar_image_anchors_enumerate_anchors_f`

Enumerates a collection of image anchors.

`ar_image_anchors_get_count`

Gets the number of image anchors in the collection.

`ar_image_tracking_provider_create`

Creates an image tracking provider that tracks the reference images you supply.

`ar_image_tracking_provider_is_supported`

Returns a Boolean value that indicates whether the current runtime environment supports image tracking providers.

`ar_image_tracking_provider_get_required_authorization_type`

Gets the types of authorizations required to track images.

`ar_image_tracking_provider_set_update_handler`

Sets the handler for receiving image tracking updates.

`ar_image_tracking_provider_set_update_handler_f`

Sets the handler for receiving image tracking updates.

`ar_image_tracking_configuration_add_reference_images`

Adds reference images to the set to be tracked.

`ar_image_tracking_configuration_create`

Creates an image tracking configuration.

`ar_image_tracking_configuration_t`

`ar_reference_images_t`

A collection of reference images.

`ar_reference_images_enumerator_t`

A handler for enumerating a collection of reference images.

`ar_reference_images_add_image`

Adds a reference image to a collection.

`ar_reference_images_add_images`

Adds multiple reference images to a collection.

`ar_reference_images_create`

Creates an empty collection of reference images.

`ar_reference_images_enumerate_images`

Enumerates a collection of reference images.

`ar_reference_images_enumerate_images_f`

Enumerates a collection of reference images.

`ar_reference_images_get_count`

Gets the number of reference images in a collection.

`ar_reference_images_load_reference_images_in_group`

Creates multiple reference images based on their group name in an asset catalog.

`ar_reference_image_create_from_cgimage`

Creates a reference image from a Core Graphics image.

`ar_reference_image_create_from_pixel_buffer`

Creates a reference image from a pixel buffer.

`ar_reference_image_get_name`

Gets the name of a reference image.

`ar_reference_image_set_name`

Sets the name of a reference image.

`ar_reference_image_get_physical_height`

Gets the height, in meters, of a reference image in the real world.

`ar_reference_image_get_physical_width`

Gets the height, in meters, of a reference image in the real world.

`ar_image_tracking_provider_t`

A source of live data about a 2D image’s position in a person’s surroundings.

`ar_image_tracking_update_handler_t`

A handler for receiving updates to image anchors.

`ar_reference_image_t`

A 2D image the system uses as a reference to find the same image in a person’s surroundings.

`ar_image_anchors_enumerator_function_t`

`ar_image_tracking_update_handler_function_t`

`ar_reference_images_enumerator_function_t`

Accessory tracking

`ar_accessories_enumerator_function_t`

Function for enumerating a collection of accessories.

`ar_accessories_enumerator_t`

Handler for enumerating a collection of accessories.

`ar_accessories_t`

`ar_accessory_anchor_t`

`ar_accessory_anchors_enumerator_function_t`

Function for enumerating a collection of accessory anchors.

`ar_accessory_anchors_enumerator_t`

Handler for enumerating a collection of accessory anchors.

`ar_accessory_anchors_t`

`ar_accessory_device_load_completion_handler_function_t`

Function triggered when a request to load an accessory from a `GCDevice` has completed.

`ar_accessory_device_load_completion_handler_t`

Handler triggered when a request to load an accessory from a `GCDevice` has completed.

`ar_accessory_t`

`ar_accessory_tracking_configuration_t`

`ar_accessory_tracking_provider_t`

`ar_accessory_tracking_update_handler_function_t`

Function called when there are updates to accessory anchors.

`ar_accessory_tracking_update_handler_t`

Handler called when there are updates to accessory anchors.

`ar_accessory_location_name_aim`

Pre-defined accessory location name for spatial gamepad and stylus aim point.

`ar_accessory_location_name_grip`

Pre-defined accessory location name for spatial gamepad grip.

`ar_accessory_location_name_grip_surface`

Pre-defined accessory location name for spatial gamepad grip surface.

`ar_accessories_add_accessories`

Add accessories to a collection.

`ar_accessories_add_accessory`

Add an accessory to a collection.

`ar_accessories_create`

Create an empty collection of accessories.

`ar_accessories_enumerate_accessories`

Enumerate a collection of accessories.

`ar_accessories_enumerate_accessories_f`

Enumerate a collection of reference objects.

`ar_accessories_get_count`

Get the count of accessories in the collection.

`ar_accessories_remove_accessory`

Remove an accessory from a collection.

`ar_accessories_remove_accessories`

Remove accessories from a collection.

`ar_accessory_anchor_create`

Create an uninitialized accessory anchor.

`ar_accessory_anchor_get_accessory`

Returns the accessory that is being tracked by the anchor.

`ar_accessory_anchor_get_anchor_from_location_transform_with_correction`

Get the transform from an anchor to the coordinate system of a location on the accessory.

`ar_accessory_anchor_get_angular_velocity`

Get the estimated angular velocity of the accessory in the local coordinate system.

`ar_accessory_anchor_get_held_chirality`

Get the held state of this accessory anchor.

`ar_accessory_anchor_get_identifier`

Get the identifier of an anchor.

`ar_accessory_anchor_get_origin_from_anchor_transform`

Get the transform from an anchor to the origin coordinate system.

`ar_accessory_anchor_get_origin_from_anchor_transform_with_correction`

Get the transform from an anchor to the origin coordinate system.

`ar_accessory_anchor_get_timestamp`

Get the timestamp corresponding to the accessory anchor.

`ar_accessory_anchor_get_tracking_state`

Get the tracking state of the accessory anchor.

`ar_accessory_anchor_get_velocity`

Get the estimated velocity of the accessory in the local coordinate system.

`ar_accessory_anchor_is_equal_to_accessory_anchor`

Returns a Boolean value that indicates whether the two accessory anchors are equal.

`ar_accessory_anchor_is_held`

Returns a Boolean value that indicates whether the accessory is held.

`ar_accessory_anchor_is_tracked`

Determine whether an accessory anchor is tracked.

`ar_accessory_anchors_enumerate_anchors`

Enumerate a collection of accessory anchors.

`ar_accessory_anchors_enumerate_anchors_f`

Enumerate a collection of acessory anchors.

`ar_accessory_anchors_get_count`

Get the count of accessory anchors in the collection.

`ar_accessory_copy_location_names`

Gets the names of locations with pre-defined coordinate systems available for this accessory.

`ar_accessory_get_identifier`

Get identifier for the accessory.

`ar_accessory_get_inherent_chirality`

Get inherent chirality for the accessory (what hand it is designed to be held in).

`ar_accessory_get_name`

Get an accessory's name.

`ar_accessory_get_source_device`

Gets the `GCDevice` that was used to initialize the accessory.

`ar_accessory_get_source_type`

Gets type of source that was used to load an accessory.

`ar_accessory_get_usdz_file_path`

Get path to a USDZ file for the accessory, if it has one.

`ar_accessory_is_equal_to_accessory`

Indicates whether two accessories are equal.

`ar_accessory_load_from_device`

Load an accessory from a `GCDevice`.

`ar_accessory_load_from_device_f`

Load an accessory from a `GCDevice`.

`ar_accessory_tracking_configuration_create`

Create an accessory tracking configuration.

`ar_accessory_tracking_configuration_set_accessories`

Set accessories to track for the configuration.

`ar_accessory_tracking_provider_create`

Create an accessory tracking provider.

`ar_accessory_tracking_provider_get_latest_anchors`

Retrieves the latest anchors seen by the provider.

`ar_accessory_tracking_provider_get_required_authorization_type`

Get the authorization type required by the accessory tracking provider.

`ar_accessory_tracking_provider_is_supported`

Determines whether this device supports the accessory tracking provider.

`ar_accessory_tracking_provider_predict_anchor_at_timestamp`

Predict an accessory anchor for the given anchor identifier at the given timestamp. The accessory's origin is a right-handed coordinate system and is defined such that X points to the right, Y points up, and Z points backward.

`ar_accessory_tracking_provider_set_update_handler`

Set the handler for receiving accessory anchor updates.

`ar_accessory_tracking_provider_set_update_handler_f`

Set the function for receiving accessory tracking updates.

`ar_accessory_anchor_tracking_state_t`

Tracking status for accessory anchors.

`ar_accessory_chirality_t`

The chirality of an accessory.

`ar_accessory_source_type_t`

The type of input that was used to initialize an `ar_accessory_t`.

`ar_accessory_tracking_error_code_t`

Error codes specific to accessory tracking.

`ar_transform_correction_t`

A correction type to apply for transforms returned from ARKit APIs.

Camera sampling

`ar_camera_frame_sample_enumerator_function_t`

Function for enumerating camera frame samples.

`ar_camera_frame_sample_enumerator_t`

Handler for enumerating camera frame samples.

`ar_camera_frame_samples_t`

`ar_camera_frame_get_frame_samples`

Get the collection of camera frame samples for this camera frame.

`ar_camera_frame_samples_enumerate_frame_samples`

Enumerate all camera frame samples in this collection.

`ar_camera_frame_samples_enumerate_frame_samples_f`

Enumerate all supported camera frame samples for this configuration using a function.

`ar_camera_frame_samples_get_count`

Get the count of camera frame samples in the collection.

`ar_camera_video_format_get_camera_rectification_type`

Get the camera rectification type for this video format.

`ar_camera_rectification_type_t`

A value describing the type of rectification applied to a video format.

Camera region

`ar_camera_region_add_anchor_completion_handler_function_t`

Function called when a request to add a camera region anchor has completed (successfully or not).

`ar_camera_region_add_anchor_completion_handler_t`

Handler called when a request to add a camera region anchor has completed (successfully or not).

`ar_camera_region_anchor_t`

`ar_camera_region_anchor_update_handler_function_t`

Function called when there are updates to a specific camera region anchor.

`ar_camera_region_anchor_update_handler_t`

Handler called when there are updates to a specific camera region anchor.

`ar_camera_region_anchors_enumerator_function_t`

Function for enumerating a collection of camera region anchors.

`ar_camera_region_anchors_enumerator_t`

Handler for enumerating a collection of camera region anchors.

`ar_camera_region_anchors_t`

`ar_camera_region_configuration_t`

`ar_camera_region_provider_t`

`ar_camera_region_remove_anchor_completion_handler_function_t`

Function called when a request to remove a camera region anchor has completed (successfully or not).

`ar_camera_region_remove_anchor_completion_handler_t`

Handler called when a request to remove a camera region anchor has completed (successfully or not).

`ar_camera_region_remove_anchor_with_identifier_completion_handler_function_t`

Function called when a request to remove a camera region anchor by its identifier has completed (successfully or not).

`ar_camera_region_remove_anchor_with_identifier_completion_handler_t`

Handler called when a request to remove a camera region anchor by its identifier has completed (successfully or not).

`ar_camera_region_anchor_create_with_parameters`

Create a camera region anchor using a transform from the anchor to the origin coordinate system, a specified size, and a camera enhancement.

`ar_camera_region_anchor_get_camera_enhancement`

Get the camera enhancement type for a given anchor.

`ar_camera_region_anchor_get_height`

Get the height of a given anchor.

`ar_camera_region_anchor_get_identifier`

Get the identifier of an anchor.

`ar_camera_region_anchor_get_origin_from_anchor_transform`

Get the transform from an anchor to the origin coordinate system.

`ar_camera_region_anchor_get_pixel_buffer`

Get the `CVPixelBufferRef` for this anchor.

`ar_camera_region_anchor_get_timestamp`

Get the timestamp corresponding to an anchor.

`ar_camera_region_anchor_get_width`

Get the width of a given anchor.

`ar_camera_region_anchor_is_equal_to_camera_region_anchor`

Returns a bool value that indicates whether the two camera region anchors are equal.

`ar_camera_region_anchors_enumerate_anchors`

Enumerate a collection of camera region anchors.

`ar_camera_region_anchors_enumerate_anchors_f`

Enumerate a collection of camera region anchors using a function.

`ar_camera_region_anchors_get_count`

Get the count of camera region anchors in a collection.

`ar_camera_region_configuration_create`

Create a camera region configuration.

`ar_camera_region_provider_add_camera_region_anchor`

Add a camera region anchor to a camera region provider.

`ar_camera_region_provider_add_camera_region_anchor_f`

Add a camera region anchor to an camera region provider using a function.

`ar_camera_region_provider_create`

Create a camera region provider.

`ar_camera_region_provider_get_required_authorization_type`

Get the authorization type required by the camera region provider.

`ar_camera_region_provider_is_supported`

Determine whether this device supports the camera region provider.

`ar_camera_region_provider_remove_camera_region_anchor`

Remove a camera region anchor from a camera region provider.

`ar_camera_region_provider_remove_camera_region_anchor_f`

Remove a camera region anchor from a camera region provider using a function.

`ar_camera_region_provider_remove_camera_region_anchor_with_identifier`

Remove a camera region anchor from a camera region provider by its identifier.

`ar_camera_region_provider_remove_camera_region_anchor_with_identifier_f`

Remove a camera region anchor from a camera region provider by its identifier using a function.

`ar_camera_region_provider_set_update_handler_for_anchor_with_identifier`

Set the handler for receiving camera region updates for a specific anchor identifier.

`ar_camera_region_provider_set_update_handler_for_anchor_with_identifier_f`

Set the function for receiving camera region updates for a specific anchor identifier.

`ar_camera_region_camera_enhancement_t`

Enum describing camera enhancement types.

`ar_camera_region_error_code_t`

Error codes specific to the camera region provider.

Coordinate spaces

`ar_coordinate_space_data_t`

`ar_coordinate_space_data_copy_cfdata`

Copy out a `CFDataRef` that archives the coordinate space data.

`ar_coordinate_space_data_copy_recipient_identifers`

Copy the list of participant identifiers of the intended recipient for this data. Data should be broadcast if the list is empty.

`ar_coordinate_space_data_create_from_cfdata`

Create and initialize an `ar_coordinate_space_data_t` from a `CFDataRef`.

Shared coordinate spaces

`ar_shared_coordinate_space_configuration_t`

`ar_shared_coordinate_space_connected_participants_update_handler_function_t`

Function called when there are updates to shared coordinate space participant status.

`ar_shared_coordinate_space_connected_participants_update_handler_t`

Handler called when there is an update to shared coordinate space connected participants.

`ar_shared_coordinate_space_provider_t`

`ar_shared_coordinate_space_sharing_status_update_handler_function_t`

Function called when there is an update to shared coordinate space sharing status.

`ar_shared_coordinate_space_sharing_status_update_handler_t`

Handler called when there is an update to shared coordinate space sharing status.

`ar_shared_coordinate_provider_set_connected_participants_update_handler_f`

Set the function for receiving shared coordinate space connected participants updates.

`ar_shared_coordinate_space_configuration_create`

Create a shared coordinate space configuration.

`ar_shared_coordinate_space_provider_copy_next_coordinate_space_data`

Copy the next collaboration data.

`ar_shared_coordinate_space_provider_create`

Create a shared coordinate space provider.

`ar_shared_coordinate_space_provider_get_participant_identifier`

Get the identifier used to identify the participant in the shared coordinate space.

`ar_shared_coordinate_space_provider_get_required_authorization_type`

Get the authorization type required by the shared coordinate space provider.

`ar_shared_coordinate_space_provider_is_supported`

Determines whether this device supports the shared coordinate space provider.

`ar_shared_coordinate_space_provider_push_data`

Push data to shared coordinate space.

`ar_shared_coordinate_space_provider_set_connected_participants_update_handler`

Set the handler for receiving shared coordinate space connected participants updates.

`ar_shared_coordinate_space_provider_set_sharing_status_update_handler`

Set the handler for receiving sharing status updates.

`ar_shared_coordinate_space_provider_set_sharing_status_update_handler_f`

Set the function for receiving sharing status updates.

Rendering

`ar_stereo_properties_configuration_t`

`ar_stereo_properties_provider_t`

`ar_stereo_properties_configuration_create`

Create a stereo properties configuration object.

`ar_stereo_properties_provider_create`

Create a stereo properties provider.

`ar_stereo_properties_provider_get_required_authorization_type`

Get the authorization type required by the stereo properties provider.

`ar_stereo_properties_provider_get_viewpoint_properties`

Returns the latest viewpoint properties.

`ar_stereo_properties_provider_is_supported`

Determines whether this device supports the stereo properties provider.

`ar_viewpoint_properties_create`

Create an `ar_viewpoint_properties_t`.

`ar_viewpoint_properties_get_device_from_left_viewpoint_transform`

Get the transformation matrix that converts from the left viewpoint to the device's coordinate space.

`ar_viewpoint_properties_get_device_from_right_viewpoint_transform`

Get the transformation matrix that converts from the right viewpoint to the device's coordinate space.

`ar_viewpoint_properties_t`

Strings

`ar_strings_enumerator_function_t`

Function for enumerating a collection of strings.

`ar_strings_enumerator_t`

Handler for enumerating a collection of strings.

`ar_strings_t`

`ar_strings_enumerate_strings`

Enumerate a collection of strings.

`ar_strings_enumerate_strings_f`

Enumerate a collection of strings using a function.

`ar_strings_get_count`

Returns the number of strings in this collection.

Objective-C compatibility

- ARKit Functions
- ARKit Data Types
- Objective-C compatibility

Errors

`ar_error_t`

An error reported by ARKit.

`ar_error_code_t`

Codes that identify errors in ARKit.

`ar_error_domain`

A string that indicates the error domain in Core Foundation.

`ar_error_get_error_code`

Gets the error code associated with an error.

`ar_error_copy_cf_error`

Copies a reference to a Core Foundation error object that represents the specified ARKit error.

Protocols

`AR_EXTERN_C_BEGIN`

`AR_EXTERN_C_END`

See Also

Related Documentation

- Setting up access to ARKit data

  Check whether your app can use ARKit and respect people's privacy.

- ARKit in visionOS C API

  Integrate ARKit with low-level libraries and functionality.

- ARKit in iOS

  Integrate iOS device camera and motion features to produce augmented reality experiences in your app or game.