

# Table of Contents

Uralstech.UXR.QuestCamera .....	2
CameraDevice .....	4
CameraDevice.ErrorCode .....	12
CameraInfo .....	13
CameraInfo.CameraEye .....	17
CameraInfo.CameraIntrinsics .....	18
CameraInfo.CameraSource .....	20
CameraSupport .....	21
CapturePipeline<T> .....	23
CaptureTemplate .....	25
ContinuousCaptureSession .....	26
JNIExtensions .....	33
NativeWrapperState .....	38
OnDemandCaptureSession .....	39
TaskExtensions .....	41
UCameraManager .....	45
YUVToRGBAConverter .....	49
YUVToRGBAConverter.CPUDepthFrame .....	56
Uralstech.UXR.QuestCamera.SurfaceTextureCapture .....	60
OnDemandSurfaceTextureCaptureSession .....	61
STCaptureSessionNative .....	65
STCaptureSessionNative.AdditionalUpdateCallbackData .....	68
STCaptureSessionNative.NativeEventId .....	70
STCaptureSessionNative.NativeSetupCallbackType .....	71
STCaptureSessionNative.NativeSetupData .....	72
STCaptureSessionNative.NativeUpdateCallbackType .....	74
STCaptureSessionNative.NativeUpdateCallbackWithTimestampType .....	75
STCaptureSessionNative.NativeUpdateData .....	76
SurfaceTextureCaptureSession .....	77

# Namespace Uralstech.UXR.QuestCamera

## Classes

### [CameraDevice](#)

A wrapper for a native Camera2 CameraDevice.

### [CameraInfo](#)

Wrapper for Camera2's CameraCharacteristics.

### [CameraInfo.CameraIntrinsics](#)

Defines the camera's intrinsic properties. All values are in pixels.

### [CameraSupport](#)

Utility to check if the current Meta Quest device supports the Passthrough Camera API.

### [CapturePipeline<T>](#)

Simple class for grouping a capture session and its texture converter.

### [ContinuousCaptureSession](#)

A wrapper for a native Camera2 CaptureSession and ImageReader.

### [JNIExtensions](#)

QOL extensions for the JNI.

### [OnDemandCaptureSession](#)

A wrapper for a native Camera2 CaptureSession and ImageReader.

### [TaskExtensions](#)

Extensions for System.Action.

### [UCameraManager](#)

Class for interfacing with the native Camera2 API on Android.

### [YUVToRGBAConverter](#)

The default YUV 4:2:0 to RGBA converter that uses a compute shader to convert the camera texture to RGBA.

### [YUVToRGBAConverter.CPUDepthFrame](#)

Handles YUV frame data.

## Enums

### [CameraDevice.ErrorCode](#)

Error codes that can be returned by the native CameraDevice wrapper.

### [CameraInfo.CameraEye](#)

The camera eye.

### [CameraInfo.CameraSource](#)

The source of the camera feed.

### [CaptureTemplate](#)

Capture template to use when recording.

### [NativeWrapperState](#)

The current assumed state of a native wrapper.

# Class CameraDevice

Namespace: [Uralstech.UXR.QuestCamera](#)

A wrapper for a native Camera2 CameraDevice.

```
public class CameraDevice : AndroidJavaProxy
```

## Inheritance

object ← CameraDevice

## Constructors

### CameraDevice(string)

```
public CameraDevice(string id)
```

## Parameters

**id** string

## Fields

### CameraId

The ID of the camera being wrapped.

```
public readonly string CameraId
```

## Field Value

string

### \_cameraDevice

```
protected AndroidJavaObject? _cameraDevice
```

Field Value

AndroidJavaObject?

## Properties

### CurrentState

The current assumed state of the native CameraDevice wrapper.

```
public NativeWrapperState CurrentState { get; }
```

Property Value

[NativeWrapperState](#)

## Methods

### CreateContinuousCaptureSession(Resolution, CaptureTemplate)

Creates a new repeating/continuous capture session for use.

```
public CapturePipeline<ContinuousCaptureSession>? CreateContinuousCaptureSession(Resolution  
resolution, CaptureTemplate captureTemplate = CaptureTemplate.Preview)
```

Parameters

**resolution** Resolution

The resolution of the capture.

**captureTemplate** [CaptureTemplate](#)

The capture template to use for the capture

## Returns

[CapturePipeline](#)<[ContinuousCaptureSession](#)>

A new capture session wrapper, or [null](#) if any errors occurred.

## Remarks

Once you have finished using the capture session, call [DisposeAsync\(\)](#) to close and dispose the session to free up native and compute shader resources.

## CreateOnDemandCaptureSession(Resolution)

Creates a new on-demand capture session for use.

```
public CapturePipeline<OnDemandCaptureSession>? CreateOnDemandCaptureSession(Resolution resolution)
```

## Parameters

**resolution** Resolution

## Returns

[CapturePipeline](#)<[OnDemandCaptureSession](#)>

## CreateOnDemandSurfaceTextureCaptureSession(Resolution, CaptureTemplate)

Creates a new on-demand OpenGL SurfaceTexture based capture session for use. Equivalent to [OnDemandCaptureSession](#).

```
public OnDemandSurfaceTextureCaptureSession?  
CreateOnDemandSurfaceTextureCaptureSession(Resolution resolution, CaptureTemplate captureTemplate = CaptureTemplate.Preview)
```

## Parameters

**resolution** Resolution

`captureTemplate` [CaptureTemplate](#)

Returns

[OnDemandSurfaceTextureCaptureSession](#)

## CreateSurfaceTextureCaptureSession(Resolution, CaptureTemplate)

Creates a new OpenGL SurfaceTexture based capture session for use. Equivalent to [ContinuousCaptureSession](#).

```
public SurfaceTextureCaptureSession? CreateSurfaceTextureCaptureSession(Resolution
resolution, CaptureTemplate captureTemplate = CaptureTemplate.Preview)
```

Parameters

`resolution` Resolution

The resolution of the capture.

`captureTemplate` [CaptureTemplate](#)

The capture template to use for the capture

Returns

[SurfaceTextureCaptureSession](#)

A new capture session wrapper, or [null](#) if any errors occurred.

Remarks

This experimental capture session uses a native OpenGL texture to capture images for better performance and requires OpenGL ES 3.0 as the project's graphics API. Works with single and multi-threaded rendering.

Once you have finished using the capture session, call [DisposeAsync\(\)](#) to dispose the session to free up native resources.

## DisposeAsync()

Closes and disposes the camera device.

```
public ValueTask DisposeAsync()
```

Returns

ValueTask

## ~CameraDevice()

```
protected ~CameraDevice()
```

## Invoke(string, nint)

```
public override nint Invoke(string methodName, nint javaArgs)
```

Parameters

**methodName** string

**javaArgs** nint

Returns

nint

## WaitForInitialization()

Waits until the CameraDevice opens or errs out.

```
public WaitUntil WaitForInitialization()
```

Returns



WaitUntil

## WaitForInitialization(TimeSpan, Action, WaitTimeoutMode)

Waits until the CameraDevice opens or errs out.

```
public WaitUntil WaitForInitialization(TimeSpan timeout, Action onTimeout, WaitTimeoutMode timeoutMode = null)
```

### Parameters

**timeout** TimeSpan

Maximum time to wait.

**onTimeout** Action

The action to perform when the **timeout** is reached.

**timeoutMode** WaitTimeoutMode

Mode in which to measure time to determine **timeout**.

### Returns

WaitUntil

## WaitForInitializationAsync(Cancellation\_token)

Waits until the CameraDevice opens or errs out.

```
public Task<bool> WaitForInitializationAsync(Cancellation_token token = default)
```

### Parameters

**token** Cancellation\_token

### Returns

Task<bool>

[true](#) if the device was opened successfully, [false](#) otherwise.

## Events

### OnDeviceClosed

Invoked when the CameraDevice is closed, along with the camera ID.

```
public event Action<string?>? OnDeviceClosed
```

Event Type

Action<string>

Remarks

The camera ID may be [null](#) if the camera could not be opened in the first place.

### OnDeviceDisconnected

Invoked when the CameraDevice is disconnected, along with the camera ID.

```
public event Action<string?> OnDeviceDisconnected
```

Event Type

Action<string>

### OnDeviceErred

Invoked when the CameraDevice encounters an error, along with the camera ID.

```
public event Action<string?, CameraDevice.ErrorCode>? OnDeviceErred
```

Event Type

Action<string, [CameraDevice.ErrorCode](#)>

## Remarks

The camera ID may be [null](#) if the camera could not be opened in the first place.

## OnDeviceOpened

Invoked when the CameraDevice is opened, along with the camera ID.

```
public event Action<string>? OnDeviceOpened
```

## Event Type

Action<string>

# Enum CameraDevice.ErrorCode

Namespace: [Uralstech.UXR.QuestCamera](#)

Error codes that can be returned by the native CameraDevice wrapper.

```
public enum CameraDevice.ErrorCode
```

## Fields

**CameraAccessException = 1000**

The native code encountered a CameraAccessException.

**CameraDeviceError = 4**

The camera device has encountered a fatal error.

**CameraDisabled = 3**

The camera device could not be opened due to a device policy.

**CameraInUse = 1**

The camera device is in use already.

**CameraServiceError = 5**

The camera service has encountered a fatal error.

**MaxCamerasInUse = 2**

The camera device could not be opened because there are too many other open camera devices.

**SecurityException = 1001**

The native code encountered a SecurityException.

**Unknown = 0**

Unknown error.

# Class CameraInfo

Namespace: [Uralstech.UXR.QuestCamera](#)

Wrapper for Camera2's CameraCharacteristics.

```
public record CameraInfo
```

## Inheritance

object ← CameraInfo

## Constructors

### CameraInfo(AndroidJavaObject)

```
public CameraInfo(AndroidJavaObject cameraInfo)
```

## Parameters

**cameraInfo** AndroidJavaObject

## Fields

### CameraId

The actual device ID of this camera.

```
public readonly string CameraId
```

## Field Value

string

## Eye

(Meta Quest) The eye which the camera is closest to.

```
public readonly CameraInfo.CameraEye Eye
```

Field Value

[CameraInfo.CameraEye](#)

## Intrinsics

The intrinsic data for this camera.

```
public readonly CameraInfo.CameraIntrinsics? Intrinsics
```

Field Value

[CameraInfo.CameraIntrinsics](#)

## LensPoseRotation

The orientation of the camera relative to the sensor coordinate system.

```
public readonly Quaternion? LensPoseRotation
```

Field Value

Quaternion?

## LensPoseTranslation

The position of the camera's optical center.

```
public readonly Vector3? LensPoseTranslation
```

Field Value

Vector3?

## NativeCameraCharacteristics

The native CameraCharacteristics object.

```
public readonly AndroidJavaObject NativeCameraCharacteristics
```

Field Value

AndroidJavaObject

## Source

(Meta Quest) The source of the camera feed.

```
public readonly CameraInfo.CameraSource Source
```

Field Value

[CameraInfo.CameraSource](#)

## SupportedResolutions

The resolutions supported by this camera.

```
public readonly Resolution[] SupportedResolutions
```

Field Value

Resolution[]

## Methods

Dispose()

Releases native plugin resources.

```
public void Dispose()
```

## Operators

### implicit operator string(CameraInfo)

```
public static implicit operator string(CameraInfo camera)
```

#### Parameters

**camera** [CameraInfo](#)

#### Returns

string



# Enum CameraInfo.CameraEye

Namespace: [Uralstech.UXR.QuestCamera](#)

The camera eye.

```
public enum CameraInfo.CameraEye
```

## Fields

**Left = 0**

The leftmost camera.

**Right = 1**

The rightmost camera.

**Unknown = -1**

Unknown.

# Class CameraInfo.CameraIntrinsics

Namespace: [Uralstech.UXR.QuestCamera](#)

Defines the camera's intrinsic properties. All values are in pixels.

```
public record CameraInfo.CameraIntrinsics
```

## Inheritance

object ← CameraInfo.CameraIntrinsics

## Constructors

CameraIntrinsics(Vector2, Vector2, Vector2, float)

```
public CameraIntrinsics(Vector2 resolution, Vector2 focalLength, Vector2 principalPoint,  
float skew)
```

## Parameters

**resolution** Vector2

**focalLength** Vector2

**principalPoint** Vector2

**skew** float

## Fields

FocalLength

Focal length in pixels.

```
public readonly Vector2 FocalLength
```

Field Value

Vector2

## PrincipalPoint

Principal point in pixels from the image's top-left corner.

```
public readonly Vector2 PrincipalPoint
```

Field Value

Vector2

## Resolution

Resolution in pixels.

```
public readonly Vector2 Resolution
```

Field Value

Vector2

## Skew

Skew coefficient for axis misalignment.

```
public readonly float Skew
```

Field Value

float

# Enum CameraInfo.CameraSource

Namespace: [Uralstech.UXR.QuestCamera](#)

The source of the camera feed.

```
public enum CameraInfo.CameraSource
```

## Fields

**PassthroughRGB = 0**

Meta Quest Passthrough RGB cameras.

**Unknown = -1**

Unknown.

# Class CameraSupport

Namespace: [Uralstech.UXR.QuestCamera](#)

Utility to check if the current Meta Quest device supports the Passthrough Camera API.

```
public static class CameraSupport
```

## Inheritance

object ← CameraSupport

## Remarks

Requires the Meta XR Core SDK.

## Fields

### MINSUPPORTOSVERSION

```
public const int MINSUPPORTOSVERSION = 74
```

Field Value

int

## Properties

### HorizonOSVersion

Get the Horizon OS version number on the headset

```
public static int? HorizonOSVersion { get; }
```

Property Value

int?

## Remarks

Requires the Meta XR Core SDK.

## IsSupported

Returns true if the current headset supports Passthrough Camera API

```
public static bool IsSupported { get; }
```

## Property Value

bool

## Remarks

Requires the Meta XR Core SDK.

# Class CapturePipeline<T>

Namespace: [Uralstech.UXR.QuestCamera](#)

Simple class for grouping a capture session and its texture converter.

```
public class CapturePipeline<T> where T : ContinuousCaptureSession
```

## Type Parameters

T

### Inheritance

object ← CapturePipeline<T>

## Constructors

### CapturePipeline(T, YUVToRGBAConverter)

```
public CapturePipeline(T captureSession, YUVToRGBAConverter textureConverter)
```

## Parameters

captureSession T

textureConverter [YUVToRGBAConverter](#)

## Fields

### CaptureSession

The capture session wrapper.

```
public readonly T CaptureSession
```

## Field Value

## TextureConverter

The YUV to RGBA texture converter.

```
public readonly YUVToRGBAConverter TextureConverter
```

Field Value

[YUVToRGBAConverter](#)

## Methods

### DisposeAsync()

Closes and disposes the capture session and texture converter.

```
public ValueTask DisposeAsync()
```

Returns

ValueTask



# Enum CaptureTemplate

Namespace: [Uralstech.UXR.QuestCamera](#)

Capture template to use when recording.

```
public enum CaptureTemplate
```

## Fields

**Default = 0**

Default value, do not use.

**Preview = 1**

Creates a request suitable for a camera preview window.

**Record = 3**

Creates a request suitable for video recording.

**StillCapture = 2**

Creates a request suitable for still image capture.

**VideoSnapshot = 4**

Creates a request suitable for still image capture while recording video.

# Class ContinuousCaptureSession

Namespace: [Uralstech.UXR.QuestCamera](#)

A wrapper for a native Camera2 CaptureSession and ImageReader.

```
public class ContinuousCaptureSession : AndroidJavaProxy
```

## Inheritance

object ← ContinuousCaptureSession

## Derived

[OnDemandCaptureSession](#)

## Remarks

This is different from [OnDemandCaptureSession](#) as it returns a continuous stream of images.

## Constructors

### ContinuousCaptureSession()

```
public ContinuousCaptureSession()
```

## Fields

### \_captureSession

The native capture session object.

```
protected AndroidJavaObject? _captureSession
```

## Field Value

AndroidJavaObject?

# Properties

## CurrentState

The current assumed state of the native CaptureSession wrapper.

```
public NativeWrapperState CurrentState { get; }
```

Property Value

[NativeWrapperState](#)

# Methods

## DisposeAsync()

Closes and disposes the capture session.

```
public ValueTask DisposeAsync()
```

Returns

ValueTask

## ~ContinuousCaptureSession()

```
protected ~ContinuousCaptureSession()
```

## Invoke(string, nint)

```
public override nint Invoke(string methodName, nint javaArgs)
```

Parameters

**methodName** string

`javaArgs` `nint`

Returns

`nint`

## WaitForInitialization()

Waits until the CaptureSession is open or erred out.

```
public WaitUntil WaitForInitialization()
```

Returns

`WaitUntil`

## WaitForInitialization(TimeSpan, Action, WaitTimeoutMode)

Waits until the CaptureSession opens or errs out.

```
public WaitUntil WaitForInitialization(TimeSpan timeout, Action onTimeout, WaitTimeoutMode timeoutMode = null)
```

Parameters

`timeout` `TimeSpan`

`onTimeout` `Action`

`timeoutMode` `WaitTimeoutMode`

Returns

`WaitUntil`

## WaitForInitializationAsync(Cancellation\_token)

Waits until the CaptureSession opens or errs out.

```
public Task<bool> WaitForInitializationAsync(CancellationToken token = default)
```

## Parameters

**token** CancellationToken

## Returns

Task<bool>

[true](#) if the session was opened successfully, [false](#) otherwise.

# Events

## OnDisposeCompleted

Called when the native wrapper has been completely disposed.

```
protected event Action? OnDisposeCompleted
```

## Event Type

Action

## OnFrameReady

Callback for processing the YUV 4:2:0 frame.

```
public event Action<nint, long, nint, nint, long, int, int, int, long>? OnFrameReady
```

## Event Type

Action<nint, long, nint, nint, long, int, int, int, long>

## Remarks

This callback may not be called from the main thread.

Parameters	
yBuffer (IntPtr)	The pointer to this frame's Y (luminance) data.
yBufferSize (long)	The size of the Y buffer in bytes.
uBuffer (IntPtr)	The pointer to this frame's U (color) data.
vBuffer (IntPtr)	The pointer to this frame's V (color) data.
uvBufferSize (long)	The size of the U and V buffers in bytes.
yRowStride (int)	The size of each row of the image in yBuffer in bytes.
uvRowStride (int)	The size of each row of the image in uBuffer and vBuffer in bytes.
uvPixelStride (int)	The size of a pixel in a row of the image in uBuffer and vBuffer in bytes.
timestamp (long)	The timestamp the frame was captured at in nanoseconds.

# OnSessionActive

Called when the session has started actively processing capture requests.

```
public event Action? OnSessionActive
```

Event Type

Action

# OnSessionClosed

Called when the session is closed.

```
public event Action? OnSessionClosed
```

Event Type

Action

## OnSessionConfigurationFailed

Called when the session could not be configured, and a boolean value indicating if the failure was caused due to a camera access/security exception.

```
public event Action<bool>? OnSessionConfigurationFailed
```

Event Type

Action<bool>

## OnSessionConfigured

Called when the session has been configured.

```
public event Action? OnSessionConfigured
```

Event Type

Action

## OnSessionRequestFailed

Called when the session request could not be set.

```
public event Action? OnSessionRequestFailed
```

Event Type

Action

## OnSessionRequestSet

Called when the session request has been set.

```
public event Action? OnSessionRequestSet
```

Event Type

Action



# Class JNIExtensions

Namespace: [Uralstech.UXR.QuestCamera](#)

QOL extensions for the JNI.

```
public static class JNIExtensions
```

## Inheritance

object ← JNIExtensions

## Methods

### GetNullableFloat(AndroidJavaObject, string)

Unboxes a native nullable float field into an float?.

```
public static float? GetNullableFloat(this AndroidJavaObject current, string fieldName)
```

#### Parameters

**current** AndroidJavaObject

**fieldName** string

#### Returns

float?

### GetNullableInt(AndroidJavaObject, string)

Unboxes a native nullable integer field into an int?.

```
public static int? GetNullableInt(this AndroidJavaObject current, string fieldName)
```

#### Parameters

**current** AndroidJavaObject

**fieldName** string

The field to unbox.

Returns

int?

The unboxed value.

## UnboxAndCreateGlobalRefForByteBufferElement(nint, int)

Unboxes and creates a global ref of a native ByteBuffer from a native Object array, and returns its direct buffer address.

```
public static (nint obj, nint ptr) UnboxAndCreateGlobalRefForByteBufferElement(nint args,
int index)
```

Parameters

**args** nint

The native array to take the buffer from.

**index** int

The index of the buffer object in the native array.

Returns

(nint obj, nint ptr)

The global reference and the direct buffer address.

## UnboxBoolElement(nint, int)

Unboxes a boolean from a native Object array.

```
public static bool UnboxBoolElement(nint args, int index)
```

## Parameters

**args** nint

The native array to take the boolean from.

**index** int

The index of the boolean object in the native array.

## Returns

bool

The unboxed boolean.

## UnboxIntElement(nint, int)

Unboxes an integer from a native Object array.

```
public static int UnboxIntElement(nint args, int index)
```

## Parameters

**args** nint

The native array to take the integer from.

**index** int

The index of the integer object in the native array.

## Returns

int

The unboxed integer.

## UnboxLongElement(nint, int)

Unboxes a long from a native Object array.

```
public static long UnboxLongElement(nint args, int index)
```

### Parameters

**args** nint

The native array to take the long from.

**index** int

The index of the long object in the native array.

### Returns

long

The unboxed long.

## UnboxStringElement(nint, int)

Unboxes a string from a native Object array.

```
public static string? UnboxStringElement(nint args, int index)
```

### Parameters

**args** nint

The native array to take the string from.

**index** int

The index of the string object in the native array.

### Returns

string

The unboxed string.

# Enum NativeWrapperState

Namespace: [Uralstech.UXR.QuestCamera](#)

The current assumed state of a native wrapper.

```
public enum NativeWrapperState
```

## Fields

**Closed = 2**

The native wrapper failed with an error, was disconnected or is being/was closed normally.

**Initializing = 0**

The native wrapper is still initializing.

**Opened = 1**

The native wrapper is open and ready.

# Class OnDemandCaptureSession

Namespace: [Uralstech.UXR.QuestCamera](#)

A wrapper for a native Camera2 CaptureSession and ImageReader.

```
public class OnDemandCaptureSession : ContinuousCaptureSession
```

## Inheritance

object ← [ContinuousCaptureSession](#) ← OnDemandCaptureSession

## Inherited Members

[ContinuousCaptureSession.CurrentState](#) , [ContinuousCaptureSession.OnSessionConfigured](#) ,  
[ContinuousCaptureSession.OnSessionConfigurationFailed](#) ,  
[ContinuousCaptureSession.OnSessionRequestSet](#) , [ContinuousCaptureSession.OnSessionRequestFailed](#) ,  
[ContinuousCaptureSession.OnSessionActive](#) , [ContinuousCaptureSession.OnSessionClosed](#) ,  
[ContinuousCaptureSession.OnFrameReady](#) , [ContinuousCaptureSession.OnDisposeCompleted](#) ,  
[ContinuousCaptureSession.captureSession](#) , [ContinuousCaptureSession.Invoke\(string, nint\)](#) ,  
[ContinuousCaptureSession.WaitForInitialization\(\)](#) ,  
[ContinuousCaptureSession.WaitForInitialization\(TimeSpan, Action, WaitTimeoutMode\)](#) ,  
[ContinuousCaptureSession.WaitForInitializationAsync\(CancellationToken\)](#) ,  
[ContinuousCaptureSession.DisposeAsync\(\)](#)

## Remarks

This is different from [ContinuousCaptureSession](#) as it only returns a frame from the native plugin when required. This is recommended for single-image capturing or on-demand capturing where you don't need a continuous stream of images.

Why does [OnDemandCaptureSession](#) inherit from [ContinuousCaptureSession](#)? Because under the hood, both do the same thing - a repeating capture session. A true on-demand capture results in a black image, so [OnDemandCaptureSession](#) runs a repeating capture request running on an dummy texture natively, and reads the actual image through an ImageReader only when requested to do so. This means that while the [ContinuousCaptureSession](#) processes each and every frame sent to it, converting it to RGBA, [OnDemandCaptureSession](#) only does it when required.

## Methods

# RequestCapture(CaptureTemplate)

Requests a new capture from the session.

```
public bool RequestCapture(CaptureTemplate captureTemplate = CaptureTemplate.StillCapture)
```

## Parameters

**captureTemplate** [CaptureTemplate](#)

The capture template to use for the capture

## Returns

**bool**

If the capture request was set successfully, [true](#)<sup>↗</sup>, otherwise, [false](#)<sup>↗</sup>.



# Class TaskExtensions

Namespace: [Uralstech.UXR.QuestCamera](#)

Extensions for System.Action.

```
public static class TaskExtensions
```

## Inheritance

object ← TaskExtensions

## Methods

### HandleAnyException(Task)

Adds a continuation to a task to log exceptions.

```
public static void HandleAnyException(this Task current)
```

## Parameters

**current** Task

### InvokeOnMainThread(Action?)

Invokes the current action on the main thread.

```
public static Task InvokeOnMainThread(this Action? current)
```

## Parameters

**current** Action

## Returns

Task

## InvokeOnMainThread<T>(Action<T>?, T)

Invokes the current action on the main thread.

```
public static Task InvokeOnMainThread<T>(this Action<T>? current, T arg0)
```

### Parameters

**current** Action<T>

**arg0** T

### Returns

Task

### Type Parameters

T

## InvokeOnMainThread<T0, T1>(Action<T0, T1>?, T0, T1)

Invokes the current action on the main thread.

```
public static Task InvokeOnMainThread<T0, T1>(this Action<T0, T1>? current, T0 arg0,  
T1 arg1)
```

### Parameters

**current** Action<T0, T1>

**arg0** T0

**arg1** T1

### Returns

Task

### Type Parameters

T0

T1

## Yield(Task)

Allows for "yielding" a System.Threading.Tasks.Task using a WaitUntil object.

```
public static WaitUntil Yield(this Task current)
```

### Parameters

**current** Task

### Returns

WaitUntil

## Yield(Task, TimeSpan, Action, WaitTimeoutMode)

Allows for "yielding" a System.Threading.Tasks.Task using a WaitUntil object.

```
public static WaitUntil Yield(this Task current, TimeSpan timeout, Action onTimeout,
    WaitTimeoutMode timeoutMode = null)
```

### Parameters

**current** Task

**timeout** TimeSpan

**onTimeout** Action

**timeoutMode** WaitTimeoutMode

### Returns

WaitUntil

# Yield(ValueTask)

Allows for "yielding" a System.Threading.Tasks.ValueTask using a WaitUntil object.

```
public static WaitUntil Yield(this ValueTask current)
```

## Parameters

**current** ValueTask

## Returns

WaitUntil

# Yield(ValueTask, TimeSpan, Action, WaitTimeoutMode)

Allows for "yielding" a System.Threading.Tasks.ValueTask using a WaitUntil object.

```
public static WaitUntil Yield(this ValueTask current, TimeSpan timeout, Action onTimeout,
    WaitTimeoutMode timeoutMode = null)
```

## Parameters

**current** ValueTask

**timeout** TimeSpan

**onTimeout** Action

**timeoutMode** WaitTimeoutMode

## Returns

WaitUntil

# Class UCameraManager

Namespace: [Uralstech.UXR.QuestCamera](#)

Class for interfacing with the native Camera2 API on Android.

```
public class UCameraManager : DontCreateNewSingleton<UCameraManager>
```

## Inheritance

object ← UCameraManager

## Fields

### AvatarCameraPermission

The permission required to access the Meta Quest's avatar camera.

```
public const string AvatarCameraPermission = "android.permission.CAMERA"
```

Field Value

string

### HeadsetCameraPermission

The permission required to access the Meta Quest's Passthrough cameras.

```
public const string HeadsetCameraPermission = "horizonos.permission.HEADSET_CAMERA"
```

Field Value

string

### YUVToRGBAComputeShader

The compute shader to use to convert the camera's YUV 4:2:0 images to RGBA.

```
public ComputeShader YUVToRGBAComputeShader
```

Field Value

ComputeShader

## Properties

### Cameras

Gets a cached array of the available cameras and their characteristics.

```
public CameraInfo[]? Cameras { get; }
```

Property Value

[CameraInfo\[\]](#)

Remarks

The disposal of the [CameraInfo](#) objects generated by this property are managed by the [UCamera Manager](#) instance. If you require control of the [CameraInfo](#) objects, use [GetCameraInfos\(\)](#) instead.

## Methods

### Awake()

```
protected override void Awake()
```

### GetCamera(CameraEye)

Gets a camera device by the eye it is closest to.

```
public CameraInfo? GetCamera(CameraInfo.CameraEye eye)
```

## Parameters

**eye** [CameraInfo.CameraEye](#)

The eye.

## Returns

[CameraInfo](#)

A [CameraInfo](#) object or [null](#) if none were found.

## Remarks

The [CameraInfo](#) object returned by this method is managed by the [UCameraManager](#) instance, so do not dispose it manually.

# GetCameraInfos()

Gets all available cameras and their characteristics. This is **not** cached.

```
public CameraInfo[]? GetCameraInfos()
```

## Returns

[CameraInfo](#)[]

An array of [CameraInfo](#) objects or [null](#) if any errors occurred.

## Remarks

You will have to manage the disposal of the [CameraInfo](#) objects returned by this method. Use [Cameras](#) if you don't want to handle the objects yourself.

# OnDestroy()

```
protected void OnDestroy()
```

## OpenCamera(string)

Opens a camera device for use.

```
public CameraDevice? OpenCamera(string camera)
```

### Parameters

**camera** string

The ID of the camera to open; accepts [CameraInfo](#) objects through an implicit cast.

### Returns

[CameraDevice](#)

A [CameraDevice](#) object or [null](#) if any errors occurred.

### Remarks

Once you have finished using the camera, close and dispose of it using [DisposeAsync\(\)](#).

## RefreshCachedCameraInfos()

Refreshes the cached array of camera devices.

```
public bool RefreshCachedCameraInfos()
```

### Returns

bool

[true](#) if the refresh was successful; [false](#) otherwise.



# Class YUVToRGBAConverter

Namespace: [Uralstech.UXR.QuestCamera](#)

The default YUV 4:2:0 to RGBA converter that uses a compute shader to convert the camera texture to RGBA.

```
public class YUVToRGBAConverter
```

## Inheritance

object ← YUVToRGBAConverter

## Constructors

### YUVToRGBAConverter(Resolution)

```
public YUVToRGBAConverter(Resolution resolution)
```

## Parameters

**resolution** Resolution

## Fields

### \_kernelHandle

```
protected int _kernelHandle
```

## Field Value

int

### \_threadGroupsX

```
protected readonly int _threadGroupsX
```

Field Value

int

## \_threadGroupsY

```
protected readonly int _threadGroupsY
```

Field Value

int

## \_uComputeBuffer

Buffer containing U (color) data of the frame being processed.

```
protected readonly ComputeBuffer _uComputeBuffer
```

Field Value

ComputeBuffer

## \_uvBufferSize

```
protected readonly int _uvBufferSize
```

Field Value

int

## \_vComputeBuffer

Buffer containing V (color) data of the frame being processed.

```
protected readonly ComputeBuffer _vComputeBuffer
```

Field Value

ComputeBuffer

## \_yBufferSize

```
protected readonly int _yBufferSize
```

Field Value

int

## \_yComputeBuffer

Buffer containing Y (luminance) data of the frame being processed.

```
protected readonly ComputeBuffer _yComputeBuffer
```

Field Value

ComputeBuffer

## Properties

### FrameCaptureTimestamp

The timestamp the last frame processed was captured at in nanoseconds.

```
public long FrameCaptureTimestamp { get; protected set; }
```

Property Value

long

## FrameRenderTexture

The RenderTexture which will contain the RGBA camera frames.

```
public RenderTexture FrameRenderTexture { get; protected set; }
```

### Property Value

RenderTexture

## Shader

The shader used to convert YUV 4:2:0 to an RGBA RenderTexture. Uses [YUVToRGBAComputeShader](#) if not specified here.

```
public ComputeShader Shader { get; set; }
```

### Property Value

ComputeShader

## Methods

### Dispose()

Releases the frame RenderTexture and buffers.

```
public void Dispose()
```

### ~YUVToRGBAConverter()

```
protected ~YUVToRGBAConverter()
```

# GetNextFrameAsync(CancellationToken)

Returns the next frame to be received by this processor.

```
public Task<(RenderTexture, long)> GetNextFrameAsync(CancellationToken token = default)
```

## Parameters

**token** CancellationToken

## Returns

Task<(RenderTexture, long)>

The frame's RenderTexture and capture timestamp, in nanoseconds..

# OnFrameReady(nint, long, nint, nint, long, int, int, int, long)

Processes a frame received from the native capture session.

```
public virtual void OnFrameReady(nint yBuffer, long yBufferSize, nint uBuffer, nint vBuffer, long uvBufferSize, int yRowStride, int uvRowStride, int uvPixelStride, long timestamp)
```

## Parameters

**yBuffer** nint

The pointer to this frame's Y (luminance) data.

**yBufferSize** long

The size of the Y buffer in bytes.

**uBuffer** nint

The pointer to this frame's U (color) data.

**vBuffer** nint

The pointer to this frame's V (color) data.

**uvBufferSize** long

The size of the U and V buffers in bytes.

**yRowStride** int

The size of each row of the image in **yBuffer** in bytes.

**uvRowStride** int

The size of each row of the image in **uBuffer** and **vBuffer** in bytes.

**uvPixelStride** int

The size of a pixel in a row of the image in **uBuffer** and **vBuffer** in bytes.

**timestamp** long

The timestamp the frame was captured at in nanoseconds.

## PrepareDataForComputeBuffer(CPUDepthFrame, int, int, int, long)

Copies the given data into the shader's buffers and dispatches it.

```
protected virtual Task PrepareDataForComputeBuffer(YUVToRGBAConverter.CPUDepthFrame frame,
int yRowStride, int uvRowStride, int uvPixelStride, long timestamp)
```

### Parameters

**frame** [YUVToRGBAConverter.CPUDepthFrame](#)

The frame data on the CPU.

**yRowStride** int

The size of each row of the image in [\\_yComputeBuffer](#) in bytes.

**uvRowStride** int

The size of each row of the image in [\\_uComputeBuffer](#) and [\\_vComputeBuffer](#) in bytes.

**uvPixelStride** int

The size of a pixel in a row of the image in [\\_uComputeBuffer](#) and [\\_vComputeBuffer](#) in bytes.

**timestamp** long

The timestamp the frame was captured at in nanoseconds.

Returns

Task

## Events

### OnFrameProcessed

Called when a capture is dispatched for conversion to RGBA, with the capture's timestamp in nanoseconds.

```
public event Action<RenderTexture, long>? OnFrameProcessed
```

Event Type

Action<RenderTexture, long>

# Class YUVToRGBAConverter.CPUDepthFrame

Namespace: [Uralstech.UXR.QuestCamera](#)

Handles YUV frame data.

```
protected record YUVToRGBAConverter.CPUDepthFrame
```

## Inheritance

object ← YUVToRGBAConverter.CPUDepthFrame

## Remarks

The NativeArrays used by this object are allocated using Allocator.TempJob.

## Constructors

### CPUDepthFrame(int, int)

```
public CPUDepthFrame(int yBufferSize, int uvBufferSize)
```

## Parameters

yBufferSize int

uvBufferSize int

## Fields

### UBuffer

Represents YUV frame data on the CPU.

```
public readonly NativeArray<byte> UBuffer
```

## Field Value



NativeArray<byte>

## VBuffer

Represents YUV frame data on the CPU.

```
public readonly NativeArray<byte> VBuffer
```

Field Value

NativeArray<byte>

## YBuffer

Represents YUV frame data on the CPU.

```
public readonly NativeArray<byte> YBuffer
```

Field Value

NativeArray<byte>

## Methods

### CopyFrom(nint, long, nint, nint, long)

Copies YUV data from native pointers.

```
public void CopyFrom(nint yNativeBuffer, long yLength, nint uNativeBuffer, nint vNativeBuffer, long uvLength)
```

Parameters

**yNativeBuffer** nint

The Y channel data.

**yLength** long

The length of the Y channel data in bytes.

**uNativeBuffer** nint

The U channel data.

**vNativeBuffer** nint

The V channel data.

**uvLength** long

The length of the U and V channel data in bytes.

## CopyTo(ComputeBuffer, ComputeBuffer, ComputeBuffer)

Copies this data to ComputeBuffers.

```
public void CopyTo(ComputeBuffer yComputeBuffer, ComputeBuffer uComputeBuffer,  
    ComputeBuffer vComputeBuffer)
```

### Parameters

**yComputeBuffer** ComputeBuffer

The Y channel buffer.

**uComputeBuffer** ComputeBuffer

The U channel buffer.

**vComputeBuffer** ComputeBuffer

The V channel buffer.

## Dispose()

Performs application-defined tasks associated with freeing, releasing, or resetting unmanaged resources.

```
public void Dispose()
```

# Namespace Uralstech.UXR.QuestCamera. SurfaceTextureCapture

## Classes

### [OnDemandSurfaceTextureCaptureSession](#)

On-demand version of [SurfaceTextureCaptureSession](#).

### [STCaptureSessionNative](#)

Class to interact with the native graphics plugin for SurfaceTexture rendering.

### [SurfaceTextureCaptureSession](#)

This experimental capture session uses a native OpenGL texture to capture images for better performance.

## Structs

### [STCaptureSessionNative.AdditionalUpdateCallbackData](#)

Additional data tracked in C# related to a native renderer update event.

### [STCaptureSessionNative.NativeSetupData](#)

Data for setting up a native renderer.

### [STCaptureSessionNative.NativeUpdateData](#)

Data for updating a native renderer.

## Enums

### [STCaptureSessionNative.NativeEventId](#)

Event ID for native rendering events.

## Delegates

### [STCaptureSessionNative.NativeSetupCallbackType](#)

Callback type for [SetupNativeTexture](#) events.

### [STCaptureSessionNative.NativeUpdateCallbackType](#)

Callback type for [CleanupNativeTexture](#) and [RenderTextures](#) events.

### [STCaptureSessionNative.NativeUpdateCallbackWithTimestampType](#)

Same as [STCaptureSessionNative.NativeUpdateCallbackType](#), but can include a timestamp tracked from C#.

# Class OnDemandSurfaceTextureCaptureSession

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

On-demand version of [SurfaceTextureCaptureSession](#).

```
public class OnDemandSurfaceTextureCaptureSession : SurfaceTextureCaptureSession
```

## Inheritance

object ← [SurfaceTextureCaptureSession](#) ← OnDemandSurfaceTextureCaptureSession

## Inherited Members

[SurfaceTextureCaptureSession.CurrentState](#) , [SurfaceTextureCaptureSession.Texture](#) ,  
[SurfaceTextureCaptureSession.CaptureTimestamp](#) , [SurfaceTextureCaptureSession.OnSessionConfigured](#) ,  
[SurfaceTextureCaptureSession.OnSessionConfigurationFailed](#) ,  
[SurfaceTextureCaptureSession.OnSessionRequestSet](#) ,  
[SurfaceTextureCaptureSession.OnSessionRequestFailed](#) ,  
[SurfaceTextureCaptureSession.OnSessionRegistrationFailed](#) ,  
[SurfaceTextureCaptureSession.OnSessionActive](#) , [SurfaceTextureCaptureSession.OnSessionClosed](#) ,  
[SurfaceTextureCaptureSession.OnFrameReady](#) , [SurfaceTextureCaptureSession.OnDisposeCompleted](#) ,  
[SurfaceTextureCaptureSession.\\_commandBuffer](#) , [SurfaceTextureCaptureSession.\\_captureSession](#) ,  
[SurfaceTextureCaptureSession.\\_nativeTextureId](#) ,  
[SurfaceTextureCaptureSession.OnFrameReadyInvk\(Texture2D, long\)](#) ,  
[SurfaceTextureCaptureSession.InitializeNative\(long\)](#) ,  
[SurfaceTextureCaptureSession.SendNativeUpdate\(STCaptureSessionNative.NativeEventId, STCaptureSessionNative.NativeUpdateCallbackWithTimestampType, long\)](#) ,  
[SurfaceTextureCaptureSession.WaitForInitialization\(\)](#) ,  
[SurfaceTextureCaptureSession.WaitForInitialization\(TimeSpan, Action, WaitTimeoutMode\)](#) ,  
[SurfaceTextureCaptureSession.WaitForInitializationAsync\(CancellationToken\)](#) ,  
[SurfaceTextureCaptureSession.Disposed](#) , [SurfaceTextureCaptureSession.DisposeAsync\(\)](#)

## Remarks

This experimental capture session uses a native OpenGL texture to capture images for better performance and requires OpenGL ES 3.0 as the project's graphics API. Works with single and multi-threaded rendering.

## Constructors

# OnDemandSurfaceTextureCaptureSession(Resolution)

```
public OnDemandSurfaceTextureCaptureSession(Resolution resolution)
```

## Parameters

**resolution** Resolution

## Properties

### HasFrame

Has the capture session received its first frame?

```
public bool HasFrame { get; }
```

## Property Value

bool

## Methods

### Invoke(string, nint)

```
public override nint Invoke(string methodName, nint javaArgs)
```

## Parameters

**methodName** string

**javaArgs** nint

## Returns

nint

## RequestCapture()

Updates the unity texture with the latest capture from the camera.

```
public WaitUntil? RequestCapture()
```

### Returns

WaitUntil?

Returns a WaitUntil operation if the renderer was invoked, [null](#) otherwise.

## RequestCapture(Action<Texture2D, long>)

Updates the unity texture with the latest capture from the camera.

```
public bool RequestCapture(Action<Texture2D, long> onDone)
```

### Parameters

**onDone** Action<Texture2D, long>

Called when the capture has been rendered in unity, with its timestamp.

### Returns

bool

[true](#) if the renderer was invoked, [false](#) otherwise.

## RequestCapture(TimeSpan, Action, WaitTimeoutMode)

Updates the unity texture with the latest capture from the camera.

```
public WaitUntil? RequestCapture(TimeSpan timeout, Action onTimeout, WaitTimeoutMode timeoutMode = null)
```

### Parameters

**timeout** TimeSpan

**onTimeout** Action

**timeoutMode** WaitTimeoutMode

Returns

WaitUntil?

Returns a WaitUntil operation if the renderer was invoked, [null](#) otherwise.

## RequestCaptureAsync(CancellationTokens)

Updates the unity texture with the latest capture from the camera.

```
public Task<(Texture2D?, long)> RequestCaptureAsync(CancellationTokens token = default)
```

Parameters

**token** CancellationTokens

Returns

Task<(Texture2D?, long)>

The rendered texture and timestamp, or default values if the renderer could not be invoked.



# Class STCaptureSessionNative

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Class to interact with the native graphics plugin for SurfaceTexture rendering.

```
public static class STCaptureSessionNative
```

## Inheritance

object ← STCaptureSessionNative

## Fields

### NativeSetupCallbacksQueue

Event queues for renderer setup events, mapped to the IDs of the unity textures they are for.

```
public static readonly ConcurrentDictionary<uint,  
STCaptureSessionNative.NativeSetupCallbackType> NativeSetupCallbacksQueue
```

#### Field Value

ConcurrentDictionary<uint, [STCaptureSessionNative.NativeSetupCallbackType](#)>

### NativeUpdateCallbacksQueue

Event queues for update events, mapped to the IDs of the native textures they are for.

```
public static readonly ConcurrentDictionary<uint,  
ConcurrentQueue<STCaptureSessionNative.AdditionalUpdateCallbackData>>  
NativeUpdateCallbacksQueue
```

#### Field Value

ConcurrentDictionary<uint, ConcurrentQueue<[STCaptureSessionNative.AdditionalUpdateCallbackData](#)>>

# Methods

## DeregisterNativeUpdateCallbacks(uint)

Deregisters a native update queue for a texture and disposes allocated data.

```
public static void DeregisterNativeUpdateCallbacks(uint textureId)
```

### Parameters

**textureId** uint

The ID of the native texture to deregister updates of.

## GetRenderEventFunction()

Gets the pointer to the native render event handler.

```
public static extern nint GetRenderEventFunction()
```

### Returns

nint

## NativeSetupCallback(bool, bool, uint, uint, bool)

The actual callback for native renderer setup events.

```
public static void NativeSetupCallback(bool glIsClean, bool sessionCallSent, uint  
unityTextureId, uint textureId, bool idIsValid)
```

### Parameters

**glIsClean** bool

Was the GL context successfully cleaned up in this call?

**sessionCallSent** bool

Was the call to start the capture session sent to the Kotlin class?

**unityTextureId** uint

The unity texture associated with the event.

**textureId** uint

The native texture created by the call, may be invalid.

**idIsValid** bool

Is **textureId** a valid texture?

## Remarks

This will dequeue from [NativeSetupCallbacksQueue](#) and process the callbacks.

## NativeUpdateCallback(uint, bool)

The actual callback for native rendering updates.

```
public static void NativeUpdateCallback(uint textureId, bool success)
```

## Parameters

**textureId** uint

The ID of the native texture which was updated.

**success** bool

If the operation was successful.

## Remarks

This will dequeue from [NativeUpdateCallbacksQueue](#) and process the callback data.

# Struct

## STCaptureSessionNative.AdditionalUpdateCallbackData

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Additional data tracked in C# related to a native renderer update event.

```
public readonly struct STCaptureSessionNative.AdditionalUpdateCallbackData
```

## Constructors

AdditionalUpdateCallbackData(NativeUpdateCallbackWithTimestampType?, nint, long)

```
public  
AdditionalUpdateCallbackData(STCaptureSessionNative.NativeUpdateCallbackWithTimestampType?  
nextCall, nint nativeData, long timestamp)
```

## Parameters

**nextCall** [STCaptureSessionNative.NativeUpdateCallbackWithTimestampType](#)

**nativeData** nint

**timestamp** long

## Fields

### NativeData

Native data that should be disposed as part of this callback.

```
public readonly nint NativeData
```

Field Value

nint

## NextCall

Optional callback that should be called after processing for the current native callback is done.

```
public readonly STCaptureSessionNative.NativeUpdateCallbackWithTimestampType? NextCall
```

Field Value

[STCaptureSessionNative.NativeUpdateCallbackWithTimestampType](#)

## Timestamp

Timestamp value which will be provided in [NextCall](#).

```
public readonly long Timestamp
```

Field Value

long

# Enum STCaptureSessionNative.NativeEventId

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Event ID for native rendering events.

```
public enum STCaptureSessionNative.NativeEventId
```

## Fields

CleanupNativeTexture = 2

Disposes a native renderer.

RenderTextures = 3

Renders the textures.

SetupNativeTexture = 1

Sets up a native renderer.

# Delegate STCaptureSessionNative.NativeSetupCallbackType

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Callback type for [SetupNativeTexture](#) events.

```
public delegate void STCaptureSessionNative.NativeSetupCallbackType(bool glIsClean, bool sessionCallSent, uint unityTextureId, uint textureId, bool idIsValid)
```

## Parameters

**glIsClean** bool

Was the GL context successfully cleaned up in this call?

**sessionCallSent** bool

Was the call to start the capture session sent to the Kotlin class?

**unityTextureId** uint

The unity texture associated with the event.

**textureId** uint

The native texture created by the call, may be invalid.

**idIsValid** bool

Is **textureId** a valid texture?

# Struct

## STCaptureSessionNative.NativeSetupData

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Data for setting up a native renderer.

```
public struct STCaptureSessionNative.NativeSetupData
```

## Fields

### Height

The height of the texture.

```
public int Height
```

### Field Value

int

### OnDoneCallback

Callback for when the operation is done, type: [STCaptureSessionNative.NativeSetupCallbackType](#).

```
public nint OnDoneCallback
```

### Field Value

nint

### Timestamp

Timestamp associated with the STCaptureSessionWrapper which will be the source for rendering.



```
public long Timestamp
```

Field Value

long

## UnityTexture

The unity texture to render to.

```
public uint UnityTexture
```

Field Value

uint

## Width

The width of the texture;

```
public int Width
```

Field Value

int

# Delegate STCaptureSessionNative.NativeUpdateCallback Type

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Callback type for [CleanupNativeTexture](#) and [RenderTextures](#) events.

```
public delegate void STCaptureSessionNative.NativeUpdateCallbackType(uint textureId,  
bool success)
```

## Parameters

**textureId** uint

The ID of the native texture which was updated.

**success** bool

If the operation was successful.

# Delegate STCaptureSessionNative.NativeUpdateCallback WithTimestampType

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Same as [STCaptureSessionNative.NativeUpdateCallbackType](#), but can include a timestamp tracked from C#.

```
public delegate void STCaptureSessionNative.NativeUpdateCallbackWithTimestampType(uint textureId, bool success, long timestamp)
```

## Parameters

**textureId** uint

The ID of the native texture which was updated.

**success** bool

If the operation was successful.

**timestamp** long

The timestamp tracked from C#.

# Struct

## STCaptureSessionNative.NativeUpdateData

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

Data for updating a native renderer.

```
public struct STCaptureSessionNative.NativeUpdateData
```

## Fields

### NativeTexture

The native texture to update.

```
public uint NativeTexture
```

### Field Value

uint

### OnDoneCallback

Callback for when the operation is done, type: [STCaptureSessionNative.NativeUpdateCallbackType](#).

```
public nint OnDoneCallback
```

### Field Value

nint

# Class SurfaceTextureCaptureSession

Namespace: [Uralstech.UXR.QuestCamera.SurfaceTextureCapture](#)

This experimental capture session uses a native OpenGL texture to capture images for better performance.

```
public class SurfaceTextureCaptureSession : AndroidJavaProxy
```

## Inheritance

object ← SurfaceTextureCaptureSession

## Derived

[OnDemandSurfaceTextureCaptureSession](#)

## Remarks

Requires OpenGL ES 3.0 as the project's graphics API. Works with single and multi-threaded rendering.

## Constructors

### SurfaceTextureCaptureSession(Resolution)

```
public SurfaceTextureCaptureSession(Resolution resolution)
```

## Parameters

**resolution** Resolution

## Fields

### Texture

The texture being rendered to.

```
public readonly Texture2D Texture
```

Field Value

Texture2D

## `_captureSession`

The native capture session object.

```
protected AndroidJavaObject? _captureSession
```

Field Value

AndroidJavaObject?

## `_commandBuffer`

CommandBuffer for invoking native renderer events.

```
protected readonly CommandBuffer _commandBuffer
```

Field Value

CommandBuffer

## `_nativeTextureId`

The native texture which captures YUV 4:2:0 data.

```
protected uint? _nativeTextureId
```

Field Value

uint?

## Properties

# CaptureTimestamp

The timestamp the last frame processed was captured at in nanoseconds.

```
public long CaptureTimestamp { get; protected set; }
```

## Property Value

long

# CurrentState

The current assumed state of the native CaptureSession wrapper.

```
public NativeWrapperState CurrentState { get; }
```

## Property Value

[NativeWrapperState](#)

# Disposed

```
protected bool Disposed { get; }
```

## Property Value

bool

# Methods

## DisposeAsync()

Closes and releases the capture session..

```
public ValueTask DisposeAsync()
```

Returns

ValueTask

## ~SurfaceTextureCaptureSession()

```
protected ~SurfaceTextureCaptureSession()
```

## InitializeNative(long)

Initializes the native renderer.

```
protected virtual void InitializeNative(long timestamp)
```

Parameters

**timestamp** long

The timestamp corresponding to the native capture session wrapper.

## Invoke(string, nint)

```
public override nint Invoke(string methodName, nint javaArgs)
```

Parameters

**methodName** string

**javaArgs** nint

Returns

nint

## OnFrameReadyInvk(Texture2D, long)



Invokes [OnFrameReady](#) for child classes.

```
protected void OnFrameReadyInvk(Texture2D texture, long timestamp)
```

## Parameters

**texture** Texture2D

**timestamp** long

## SendNativeUpdate(NativeEventId, NativeUpdateCallbackWithTimestampType?, long)

Sends an update event to the native renderer.

```
protected virtual Task SendNativeUpdate(STCaptureSessionNative.NativeEventId eventId,  
    STCaptureSessionNative.NativeUpdateCallbackWithTimestampType? callback, long timestamp = 0)
```

## Parameters

**eventId** [STCaptureSessionNative.NativeEventId](#)

The type of the event.

**callback** [STCaptureSessionNative.NativeUpdateCallbackWithTimestampType](#)

An optional callback for the event's completion.

**timestamp** long

An optional timestamp to be tracked in C# code, to be forwarded to **timestamp**.

## Returns

Task

## WaitForInitialization()

Waits until the CaptureSession is open or erred out.

```
public WaitUntil WaitForInitialization()
```

Returns

WaitUntil

## WaitForInitialization(TimeSpan, Action, WaitTimeoutMode)

Waits until the CaptureSession opens or errs out.

```
public WaitUntil WaitForInitialization(TimeSpan timeout, Action onTimeout, WaitTimeoutMode timeoutMode = null)
```

Parameters

**timeout** TimeSpan

**onTimeout** Action

**timeoutMode** WaitTimeoutMode

Returns

WaitUntil

## WaitForInitializationAsync(Cancellation\_token)

Waits until the CaptureSession opens or errs out.

```
public Task<bool> WaitForInitializationAsync(Cancellation_token token = default)
```

Parameters

**token** Cancellation\_token

Returns

Task<bool>

[true](#) if the session was opened successfully, [false](#) otherwise.

## Events

### OnDisposeCompleted

Called when the native Kotlin wrapper has been completely disposed.

```
protected event Action? OnDisposeCompleted
```

Event Type

Action

### OnFrameReady

Called when a frame is ready, with its capture timestamp in nanoseconds.

```
public event Action<Texture2D, long>? OnFrameReady
```

Event Type

Action<Texture2D, long>

Remarks

This callback may not be called from the main thread.

### OnSessionActive

Called when the session has started actively processing capture requests.

```
public event Action? OnSessionActive
```

Event Type

Action

## OnSessionClosed

Called when the session is closed.

```
public event Action? OnSessionClosed
```

Event Type

Action

## OnSessionConfigurationFailed

Called when the session could not be configured, and a boolean value indicating if the failure was caused due to a camera access/security exception.

```
public event Action<bool>? OnSessionConfigurationFailed
```

Event Type

Action<bool>

## OnSessionConfigured

Called when the session has been configured.

```
public event Action? OnSessionConfigured
```

Event Type

Action

## OnSessionRegistrationFailed

Called when the session could not be registered with the native renderer.

```
public event Action? OnSessionRegistrationFailed
```

Event Type

Action

## OnSessionRequestFailed

Called when the session request could not be set.

```
public event Action? OnSessionRequestFailed
```

Event Type

Action

## OnSessionRequestSet

Called when the session request has been set.

```
public event Action? OnSessionRequestSet
```

Event Type

Action