

Painstakingly copied from <https://github.com/ValveSoftware/openvr/releases> because they have no central changelog and some of their documentation hasn't been updated since the initial release of the SDK. This serves as a single searchable changelog with bookmarks.

## v1.16.8

Feb 24, 2021

IVRCompositor:

- Added support for OpenGL array textures via the Submit\_GLArrayTexture flag.

IVROverlay:

- Added support for overlays that support projections via SetOverlayTransformProjection

## v1.14.15

Sep 18, 2020

New events:

- VREvent\_DesktopViewUpdating
- VREvent\_DesktopViewReady

IVRDriverDirectModeComponent:

- Updated CreateSwapTextureSet to (optionally) allow NTHandle support

IVRServerDriverHost:

- Added SetDisplayProjectionRaw
- Added SetRecommendedRenderTargetSize
- Renamed TrackedDeviceDisplayTransformUpdated -> SetDisplayEyeToHead

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 6100641]

## v1.12.5

Jun 03, 2020

Vulkan:

- Added support for texture arrays

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5908460]

## v1.11.11

Apr 24, 2020

New events:

- Added `VREvent_WindowsMRSectionSettingChanged`, which is sent whenever a setting in the "driver\_holographic" section changes.
- Added `VREvent_OtherSectionSettingChanged`, which is sent when a setting changes in a section that doesn't have an event of its own.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5825862]

## v1.10.30

Mar 16, 2020

`IVRHeadsetView`:

- Allows scene applications access the contents of the "VR View" window, including modifying properties of their own instances of that window. See the header file for details.

`IVROverlayView`:

- Allows applications to read overlay textures and send mouse events to those overlays. See the header file for details.

`IVRCompositor`:

- Added `Submit_FrameDiscontinuity` flag which can be passed to `Submit` to identify a discontinuity with the previous frame (e.g. when the player teleports). This prevents SteamVR's Motion Smoothing from being applied using that pair of frames.
- `IVRCompositor::Submit` – Can now handle DirectX Texture Arrays. Index 0 for the left eye, Index 1 for the right eye

`IVRRenderModels`:

- Added `VRComponentProperty_IsHighlighted`, which is set to true for any render model component that should be highlighted for an input binding.

## IVROverlay:

- The DualAnalog input method is no longer supported.
- Added VROverlayFlags\_WantsModalBehavior. This flag causes an overlay to get a VREvent\_Modal\_Cancel event whenever the user clicks off of the overlay. This is ignored for dashboard overlays.
- Added flags for keyboard usage. “minimal mode” is now specified with KeyboardFlag\_Minimal.
- Made modal keyboard behavior optional. To get the behavior from pre-1.10 SDKs, pass KeyboardFlag\_Modal.
- Added VREvent\_LockMousePosition and VREvent\_UnlockMousePosition, which are sent to overlays when a user is laser mousing over the overlay and performs the LockMousePosition compositor action.
- Removed support for rendermodels attached to overlays. This feature is no longer supported.

## IVRInput:

- Added global action set priority. When this is enabled in settings, overlay applications can override part or all of the input available to scene applications. See the header file comments for more details.
- Added GetDominandHand/SetDominantHand. This allows an application to flip left and right hands on bindings. Applications that support this must add “supports\_dominant\_hand\_setting” to true in their action manifest file.
- Added GetComponentStateForBinding. This allows applications to get the render model component transform for any component on a render model for a set of input bindings.
- Added GetBindingVariant. This allows applications to provide multiple binding files for a controller type and determine which of them the user currently has selected. The value returned here is the “variant” field from the options block in the binding JSON file.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5759046]

# v1.9.16

Jan 06, 2020

*This is identical to 1.9.15, except with the correct version information in the commit message*

## IVRDriverInput:

- Added TrackedControllerRole\_Stylus, which allows a device to appear as /user/stylus in the binding system.

## IVRSettings:

- Removed sync function. Saving of settings is controlled automatically by the system now. All existing calls to Sync can be deleted. Calls to sync using old APIs do nothing.

#### IVRCompositor:

- Added the ability for apps to skin the construct during loading transitions using a single model (limit 65k vertices) and a single texture (rendered unlit). See <https://github.com/ValveSoftware/openvr/wiki/Compositor-Skinning> for details.

#### IVROverlay:

- Added transform types for dashboard tabs and dashboard thumbs. These no longer use absolute transforms, so they no longer have getters for their transforms.
- Added SetOverlayTransformCursor, which will set an overlay's transform to "cursor", which means it will be centered around its hotspot and automatically positioned by the laser mouse system.
- Added GetOverlayTransformCursor in case an application needs to read back the cursor hotspot of a cursor.
- Added SetOverlayCursor, which allows an overlay to specify which cursor overlay should be used whenever the laser mouse is pointed at that overlay. The desktop overlay uses this to show custom cursors based on context.
- Added SetOverlayCursorPositionOverride, which lets an overlay set the position to show its cursor at a position that is independent from where the laser mouse is pointing. The desktop overlay uses this to allow the physical mouse to override the laser mouse.
- Added ClearOverlayCursorPositionOverride, which clears any cursor position override on the overlay and returns it to using the laser mouse position.
- Added TriggerLaserMouseHapticVibration, which will trigger a haptic vibration on the laser mouse device in the context of a particular transform.

## v1.9.15

Dec 27, 2019

#### IVRDriverInput:

- Added TrackedControllerRole\_Stylus, which allows a device to appear as /user/stylus in the binding system.

#### IVRSettings:

- Removed sync function. Saving of settings is controlled automatically by the system now. All existing calls to Sync can be deleted. Calls to sync using old APIs do nothing.

#### IVRCompositor:

- Added the ability for apps to skin the construct during loading transitions using a single model (limit 65k vertices) and a single texture (rendered unlit). See <https://github.com/ValveSoftware/openvr/wiki/Compositor-Skinning> for details.

#### IVROverlay:

- Added transform types for dashboard tabs and dashboard thumbs. These no longer use absolute transforms, so they no longer have getters for their transforms.
- Added SetOverlayTransformCursor, which will set an overlay's transform to "cursor", which means it will be centered around its hotspot and automatically positioned by the laser mouse system.
- Added GetOverlayTransformCursor in case an application needs to read back the cursor hotspot of a cursor.
- Added SetOverlayCursor, which allows an overlay to specify which cursor overlay should be used whenever the laser mouse is pointed at that overlay. The desktop overlay uses this to show custom cursors based on context.
- Added SetOverlayCursorPositionOverride, which lets an overlay set the position to show its cursor at a position that is independent from where the laser mouse is pointing. The desktop overlay uses this to allow the physical mouse to override the laser mouse.
- Added ClearOverlayCursorPositionOverride, which clears any cursor position override on the overlay and returns it to using the laser mouse position.
- Added TriggerLaserMouseHapticVibration, which will trigger a haptic vibration on the laser mouse device in the context of a particular transform.

## v1.8.19

Nov 05, 2019

#### Properties:

- Added Prop\_DashboardScale\_Float. Each headset has the opportunity to provide a scale for 'comfortable dashboard UI legibility'. For reference, Vive is 1.0 scale and Index is 0.75 scale.
- Added Prop\_IpdUIRangeMinMeters\_Float, Prop\_IpdUIRangeMaxMeters\_Float to control the IPD user interface range, in-headset.
- Added Prop\_Audio\_DefaultPlaybackDeviceId\_String, Prop\_Audio\_DefaultRecordingDeviceId\_String for HMDs to provide identifiers for associated audio playback and recording devices. On Windows these are Endpoint ID Strings.
- Added Prop\_Audio\_DefaultPlaybackDeviceVolume\_Float for HMDs to specify a default volume level which will be set the first time the HMDs associated audio playback device is selected. Prop\_Audio\_DefaultPlaybackDeviceId\_String must also be set. The volume range is 0 to 1.

#### IVRApplications:

- Added GetSceneApplicationState(), which returns the running application state ( None, Starting, Quitting, Running, etc). See VREvent\_SceneApplicationStateChanged
- Removed GetTransitionState()
- Removed IsQuitUserPromptRequested()

#### IVRSystem:

- Removed AcknowledgeQuit\_UserPrompt(). "Prompt user on quit" functionality is no longer supported.

#### IVROverlay:

- Added VROverlayFlags\_HideLaserIntersection - This causes the laser mouse to not draw the intersection blob for this overlay. The overlay is responsible for drawing its own cursor, if appropriate.
- Added support for rendering overlays as gravity aligned curved surfaces. See new Set/GetOverlayCurvature definitions for details.
- Removed gamepad support in overlays. This includes GetGamepadFocusOverlay(), SetGamepadFocusOverlay(), SetOverlayNeighbor(), and MoveGamepadFocusToNeighbor(). Going forward gamepads will be supported via lasermouse instead of with these custom calls.

#### IVRInput:

- Added OpenBindingUI - This function opens the binding user interface.

#### Events:

- Added VREvent\_SceneApplicationStateChanged
- Removed VREvent\_SceneFocusLost, VREvent\_SceneFocusGained, VREvent\_SceneApplicationSecondaryRenderingStarted, VREvent\_ApplicationTransition{\*}
- Removed VREvent\_QuitAborted\_UserPrompt

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5469045]

## v1.7.15

Sep 17, 2019

#### Events:

- VREvent\_Compositor\_MirrorWindowShown and VREvent\_Compositor\_MirrorWindowHidden are no longer supported and have been removed.

#### IVRApplications:

- Added `VRApplicationProperty_IsHidden_Bool`, which is true for applications which the user has chosen to hide in their Steam library.

#### IVRCompositor:

- Disabled the following functions: `ShowMirrorWindow()`, `HideMirrorWindow()`, and `IsMirrorWindowVisible()`. These are no longer support with the new VR View window.

#### IVROverlay:

- Removed support for so-called "high quality" overlays, along with the functions that create them. These overlays never supported laser mouse interaction, were unable to act as dashboard overlays, and generally didn't play well with the rest of SteamVR. This approach to rendering overlays also didn't scale to modern displays. Any existing apps that still use the APIs in old versions of the SDK will fall back to being rendered as standard overlays.

#### IVRTrackedCamera:

- Added support for setting which tracking universe poses are returned in. Set the camera tracking universe with `SetCameraTrackingSpace(...)` and retrieve the current tracking universe with `GetCameraTrackingSpace(...)`. The camera tracking universe defaults to standing.

#### Driver Interface:

- Added support for overriding HMD properties from display redirect drivers. Any properties written to the `k_ulDisplayRedirectContainer` property container will be read instead of the properties on the HMD itself. This container is only valid after `Activate` has been called on a display redirect device.
- Created an optional `Prop_NamedIconPathDeviceStandbyAlert_String` that drivers can use to declare a device icon for a combination of standby + alert states. It's recommended to visually be a combination of the `Prop_NamedIconPathDeviceReadyAlert_String` and `Prop_NamedIconPathDeviceStandby_String` icons.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5356236]

## v1.6.10

Aug 02, 2019

#### IVRSystem:

- Added `GetAppContainerFilePaths` – Call this function to get a semicolon-delimited list of file paths that any app container that intends to act as a SteamVR application will need access to.
- Added `GetRuntimeVersion` – This returns the version of the SteamVR runtime as a string.

#### IVROverlay:

- `VROverlayFlags_ProtectedContent` – This flag prevents the overlay in question from being readable by any OpenVR SDK, and prevents it from showing up in the overlay viewer.

#### CVRPropertyHelpers:

- Added getters and setters for 2, 3, and 4 element vectors

#### Driver Interface:

- Added `IVRServerDriverHost::GetFrameTimings`, which allows drivers to retrieve frame timing data
- Added tracked device class to `IVRWatchdogHost::WatchdogWakeUp` – This allows more precise logging of how the user caused SteamVR to wake up from hardware activity.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5271985]

## v1.5.17

Jul 11, 2019

#### Properties:

- Deprecated `Prop_DoNotApplyPrediction_Bool`. Drivers should provide poses with zero velocity instead.
- Added `Prop_DriverProvidedChaperoneVisibility_Bool` – Drivers that set `Prop_DriverProvidedChaperonePath_String` can also set this property to indicate when the driver provided chaperone is visible or not visible.
- Added `Prop_HmdTrackingStyle_Int32` - Drivers should set this to an `EHmdTrackingStyle` value to control what message is shown to users when the HMD isn't tracking.

Add a per-driver “loadPriority” (higher is earlier loading) to control the order that we check drivers for available HMDs. The default loadPriority is 0. Drivers can set a different default in `resources/settings/default.vrsettings`. The user can override those settings in their personal



steam/config/steamvr.vrsettings file. Drivers with the same priority continue to be loaded in alphabetical order as before.

IVRDriverManager interface:

- Added IsEnabled, which returns true if the driver is enabled.

IVRInput interface:

- Added GetActionBindingInfo - This function allows the caller to learn details about exactly how an action is bound, including what input source the binding is for, what mode on that source, and what slot on that mode. For example, a dpad binding to a trackpad would be "/input/trackpad", "dpad", and "north".

IVRDebug Interface:

- Initial version of IVRDebug, an interface intended to collect SteamVR debugging functionality. It currently provides methods to interact with the VR Profiler and the driver debug interface.
- Added EmitVrProfilerEvent - Applications can use this method to emit a discrete event to the VR Profiler.
- Added BeginVrProfilerEvent/FinishVrProfilerEvent - Applications can use this pair of functions to create a duration based VR Profiler event. The methods signal the beginning and the end of the event respectively.
- Added DriverDebugRequest - Migrated this function from IVRSystem to IVRDebug.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5214411]

## **v1.4.18**

May 24, 2019

General:

- vr::VR\_GetRunTimePath now takes a buffer to fill with the returned path. This form of the function is now thread-safe.

Driver Interface:

- Prop\_DashboardLayoutPathName\_String - specified the name of the dashboard layout control file. Allows per-HMD control over settings like dashboard distance, etc.

HelloVR Sample:

- Added an example of generic bindings that will work if no device-specific bindings are specified.

#### Valve Index Controller:

- Renamed some references from Knuckles to Index Controller

#### IVRChaperoneSetup:

- Added RoomSetupStarting - This fires an event that the tracking system can use to know room setup is about to begin. This lets the tracking system make any last minute adjustments that should be incorporated into the new setup. If the user is adjusting live in HMD using a tweak tool, keep in mind that calling this might cause the user to see the room jump.

#### IVRCompositor:

- Added IsMotionSmoothingSupported - This returns true if Motion Smoothing is supported by the current hardware.
- Added IsCurrentSceneFocusAppLoading - This indicates whether or not the current scene focus app is currently loading. The return value is inferred from its use of FadeGrid to explicitly fade to the compositor to cover up the fact that it cannot render at a sustained full framerate during this time.

#### IVRInput:

- Split GetPoseActionData into two functions
  - GetPoseActionRelativeToNow - returns the data for the pose action for any time, given a relative number of seconds.
  - GetPoseActionDataForNextFrame - returns the data for the pose action that matches the application's most recent call to WaitGetPoses
- Added a eSummaryType argument to GetSkeletalSummaryData - This allows applications to specify how much filtering should be applied to the data:
  - VRSummaryType\_FromAnimation - The data should match the animated transforms in the skeleton transforms. This data will probably be smoothed and may be more latent
  - VRSummaryType\_FromDevice - The data should be the unprocessed values from the device when available. This data may include more jitter but may be provided with less latency.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5123272]

## v1.3.22

Apr 11, 2019

#### General:

- Removed DLL exports in the static library version of openvr\_api.dll

#### Driver Interface:

- Added Prop\_AdditionalSystemReportData\_String, which allows drivers to put additional information about its devices in the system report.
- Added VREvent\_SystemReport\_Started, which gives drivers the opportunity to log before a system report is generated.
- Added IVRServerDriverHost::RequestRestart, which allows a driver to request a restart with a message that will be shown to the user, and an optional executable to run.

#### IVROverlay:

- Added viewportscale to scroll events. This is the overlay's vertical size relative to the overlay height.

#### IVRInput:

- Added IsUsingLegacyInput, which returns true if the application is using legacy input.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 5016734]

## v1.2.10

Jan 30, 2019

#### General:

- Added new, more specific error messages for many compositor startup failures.
- Fix for IVRCompositor.GetFrameTimings C# binding ([#1001](#))
- Added VRNotifications and VRIOBuffer C# accessors.

#### Event and Overlay changes:

- Scroll events have been split into two types. The first, VREvent\_ScrollDiscrete, is meant for applications that are tuned to accept primarily detented mousewheel events, which replaces the previous VREvent\_Scroll event. The second, VREvent\_ScrollSmooth, is for applications that are tuned for more touchscreen-like, analog scrolling.
- VROverlayFlags have been updated to allow overlays to indicate their preferred scroll event type. The VROverlayFlags\_SendVRDiscreteScrollEvents flag renames the VROverlayFlags\_SendVRScrollEvents flag, and the overlay will receive VREvent\_ScrollDiscrete events when this flag is set. The

VROverlayFlags\_SendVRSmoothScrollEvents flag is added, and the overlay's owning application will receive VREvent\_ScrollSmooth events when this flag is set.

- VREvent\_Input\_BindingsUpdated is sent when a user has updated controller bindings using the system input binding UI.

IVRIOBuffer:

- Adds ability for writers to detect if an IOBuffer has readers (IVRIOBuffer::HasReaders), to avoid potentially expensive writing work

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4926575]

## v1.1.3b

Nov 27, 2018

General:

- Added required SteamVR version number to the header file.

IVRCompositor:

- New VRCompositor\_FrameTiming ReprojectionFlag: VRCompositor\_ReprojectionMotion. This flag will be set for application frames where motion smoothing was applied at least one of the times it was displayed.
- New interface IsMotionSmoothingEnabled added to determine if that user has enabled motion smoothing or not.

IVRChaperone:

- Added VREvent\_ChaperoneFlushCache, which is sent when the application should reload any cached data they loaded from the chaperone API. This event indicates the user's current chaperone settings have changed.
- The VREvent\_ChaperoneDataHasChanged event will no longer be sent, and IVRChaperone::ReloadInfo no longer has any effect.

IVRChaperoneSetup:

- Removed some unimplemented functions:
- SetWorkingCollisionBoundsTagsInfo
- GetLiveCollisionBoundsTagsInfo
- SetWorkingPhysicalBoundsInfo
- GetLivePhysicalBoundsInfo
- Added ShowWorkingSetPreview/HideWorkingSetPreview, which will cause the application's current working set to show as the chaperone bounds rendered by the

compositor. Unless your application is modifying the user's chaperone bounds, you won't need to call these functions. They are independent of bounds turning on and off based on how close the user is to them.

#### IVROverlay:

- Added flag `VROverlayFlags_MakeOverlaysInteractiveIfVisible`. If this flag is set on an overlay and that overlay is visible, SteamVR will be placed into laser mouse mode. This will prevent the scene application from receiving any input, so use this flag carefully.
- Changed `SetOverlayDualAnalogTransform` to take a pointer to better support the C API.

#### IVRTrackedCamera:

- for headsets (like Vive Pro) which include multiple camera images in a single video stream, `GetCameraIntrinsics` and `GetCameraProjection` now support a `uint32_t nCameraIndex` parameter to get data about the specific camera.

#### IVRInput:

- Added an argument to `GetOriginLocalizedName` to allow the caller to specify which parts of the name they want in the returned string. The possible values are:
- `VRInputString_Hand` - Which hand the origin is in. E.g. "Left Hand"
- `VRInputString_ControllerType` - What kind of controller the user has in that hand. E.g. "Vive Controller"
- `VRInputString_InputSource` - What part of that controller is the origin. E.g. "Trackpad"
- `VRInputString_All` - All of the above. E.g. "Left Hand Vive Controller Trackpad"
- Skeletal Input:
- Added `GetBoneCount` to return the number of bones in the skeleton associated with an action
- Added `GetBoneHierarchy` which returns the index of each bone's parent in a user-provided array
- Added `GetBoneName` to retrieve the name of the bones in the skeleton
- Added `GetSkeletalReferenceTransforms` to retrieve the transforms for several specific hand skeleton poses:
  - Bind Pose
  - Open Hand
  - Fist
  - Grip Limit, which is the shape of the hand when closed around the controller
- Added `GetSkeletalTrackingLevel` to retrieve an estimate of the level of detail with which the controller associated with an action can track actual the movement of the user's body. The levels are:
  - Estimated: body part location can't be directly determined by the device. Any skeletal pose provided by the device is estimated by assuming the position required to active buttons, triggers, joysticks, or other input sensors. e.g. Vive Controller, Gamepad

- Partial: body part location can be measured directly but with fewer degrees of freedom than the actual body part. Certain body part positions may be unmeasured by the device and estimated from other input data. e.g. Knuckles, gloves that only measure finger curl
- Full: body part location can be measured directly throughout the entire range of motion of the body part. e.g. Mocap suit for the full body, gloves that measure rotation of each finger segment
- Added GetSkeletalSummaryData which returns meta data about the current pose of the hand such as finger curl and splay
- Removed ulRestrictToDevice as a parameter from all skeletal input functions

#### Driver API:

- Added TrackedControllerRole\_Treadmill, which lets a driver specify that a device is intended to function as a treadmill. This opts the device out of hand selection. The new input path /user/treadmill is automatically assigned to the first treadmill device to activate.

#### IVRCameraComponent:

- CVS\_FORMAT\_BAYER16BG for cameras which support delivering raw sensor data
- Added camera index to GetCameraDistortion, GetCameraProjection, and GetCameraIntrinsics to support multiple cameras on the same device (see also IVRTrackedCamera)
- Added the ability for cameras to return one of a small set of programmatic distortion function types and function parameters in addition to or instead of UV-sampling distortion through GetCameraDistortion. See EVRDistortionFunctionType and IVRCameraComponent::GetCameraIntrinsics and refer to OpenCV camera calibration and undistortion documentation.

#### IVRDriverInput:

- Added parameter to CreateSkeletonComponent to allow the driver to specify the skeletal tracking level that the controller supports

#### Samples:

- Fixed texture corruption bug with hellovr\_vulkan when controller is turned on after starting the application.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4837234]

#### OpenVR SDK 1.1.3b:

- Fixed issue with EVRSkeletalTrackingLevel not being defined in openvr\_driver.h Fixes [#947](#)

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4837752]

## v1.0.17

Oct 16, 2018

IVRProperties:

- Prop\_CameraStreamFormat\_Int32 - value from the openvr\_driver.h ECameraVideoStreamFormat indicating the expected delivery format of the device
- TrackedProp\_IPCReadFailure

IVRTrackedCamera:

- CameraVideoStreamFrameHeader\_t:: ulFrameExposureTime - additional field which carries the time in absolute system ticks of when the frame exposure happened, and the time of the frames pose.
- CVS\_FORMAT\_YUYV16 - new opener\_driver.h ECameraVideoStreamFormat value indicating 16-bit YUYV raw image encoding

OpenVR C API:

- Fixed cNewInput and rchRenderModelComponentName having an incorrect type

IVRCompositor FrameTiming:

- Added frame prediction and throttling bits to reprojection flags. Use the macros VR\_COMPOSITOR\_ADDITIONAL\_PREDICTED\_FRAMES and VR\_COMPOSITOR\_NUMBER\_OF\_THROTTLED\_FRAMES for easy access. These values can be used to provide a better job at updating game simulation time for rendered frames when unable to meet native refresh rate requirements.
- Added m\_nNumVSyncsReadyForUse for tracking how long each frame took to render.
- Added m\_nNumVSyncsToFirstView for tracking how many vsync intervals before a given frame was first viewed (i.e. scanned out). This may differ from NumVSyncsReadyForUse if the frame was predicted further ahead since frames will never be displayed earlier than the time they were predicted to.
- See [https://developer.valvesoftware.com/wiki/SteamVR/Frame\\_Timing](https://developer.valvesoftware.com/wiki/SteamVR/Frame_Timing) for more details.

Driver API:

- Added TrackingResult\_Fallback\_RotationOnly, which drivers can return if they have lost positional tracking but wish to still provide rotation-only tracking.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4759625]

# v1.0.16

Jul 27, 2018

## SteamVR Input (IVRInput/IVRDriverInput):

- Added motion range parameter to UpdateSkeletonComponent in IVRDriverInput. This allows the driver to specify which range of motion set the given transforms are for, with controller or without controller
- Added boneCount member InputSkeletalActionData struct. This lets users know how many bones are in the target skeleton by calling GetSkeletalActionData.
- Added GetSkeletalBoneData as a separate call to get the bone transforms. User code should now call GetSkeletalActionData first to determine if the action is active, and if so then call GetSkeletalBoneData to retrieve the transforms.
- Added an input example to the sample driver. This includes an input profile, localized strings, icons, etc.
- Added ulRestrictToDevice argument to GetDigitalActionData, GetAnalogActionData, GetSkeletalActionData, GetSkeletalBoneData, GetSkeletalBoneDataCompressed, and TriggerHapticVibration. This allows applications to use the same action for both hands and differentiate them when calling to fetch the actual data. To get these handles, pass "/user/hand/right" or "/user/hand/left" to IVRInput::GetInputSourceHandle.
- Added nPriority to VRActiveActionSet\_t. This allows applications to control how multiple bindings from different actions sets block each other. If action set A and action set B both have something bound to the trigger, the action set with the larger priority number will be returned to the application and the action set with the smaller priority number will receive no input. If multiple action sets use the same priority number, the application will get inputs from all actions in those sets even if they're bound to the same input source.

## IVRRenderModels:

- Added GetComponentStateForDevicePath, which allows applications to animate controller components while using the new input system. To get these handles, pass "/user/hand/right" or "/user/hand/left" to IVRInput::GetInputSourceHandle or use the devicePath field in the structure filled out by IVRInput::GetOriginTrackedDeviceInfo.

## IVRSpatialAnchors:

- Add a new interface that lets clients request driver descriptors for physical locations. These descriptors are opaque strings that are stored by the client until they want to recover that position for future use. The driver can update the coordinates whenever it has better information (for example when the HMD gets physically closer to the original location). The client should treat active anchors like object poses that could update at any time. Drivers don't have to implement any special support for this, in which case the runtime will provide fallback descriptors which are no better or worse than just having



the application record virtual coordinates. As of this release, no drivers are providing special descriptors. This interface is present for prototyping but is not recommended for production use at this time. Further documentation is provided in the associated headers.

#### IVRVirtualDisplay:

- Added the ability for DisplayRedirect devices to disable mura correction in the compositor (set Prop\_DriverRequestedMuraCorrectionMode\_Int32 to EVRMuraCorrectionMode\_NoCorrection).
- Added the ability for DisplayRedirect devices to specify parameters to use for mura feathering. This is to allow DisplayRedirect devices that use the IVRVirtualDisplay interface to handle only a subset of mura correction in the middle of the display, while the compositor continues to correct the outer area. Mura correction values are linearly interpolated to zero over the specified ranges in pixels; corners use the max value.

#### IVRCompositorPluginProvider:

- Initial support for loading SteamVR drivers in the compositor process. Currently only supports IVRVirtualDisplay components in order to provide async reprojection support.

#### New texture submission formats for macOS 10.14:

- macOS 10.14 now supports cross-process sharable MTLTexture objects. Applications on macOS 10.14 or higher should prefer submitting textures to the compositor using TextureType\_Metal rather than TextureType\_IOSurface.
- TextureType\_Metal textures of type MTLTextureType2D and MTLTextureType2DArray can be submitted to IVRCompositor::Submit. When submitting MTLTextureType2DArray textures, the compositor assumes layer 0 is the left eye texture (vr::EVREye::Eye\_Left) and layer 1 is the right eye texture (vr::EVREye::Eye\_Right). Applications must still call IVRCompositor::Submit twice, once for each eye, when submitting MTLTextureType2DArray textures, and the two layers of the texture will only be utilized if the same MTLTextureType2DArray texture is submitted to each call.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4637165]

## v1.0.15

May 14, 2018

Added support for the new SteamVR Input system. Please see the documentation for details:

- <https://github.com/ValveSoftware/openvr/wiki/SteamVR-Input>

Added support for tracked devices to provide access to their raw inertial measurement unit (IMU) data stream. Please see the documentation for details:

- [https://github.com/ValveSoftware/openvr/wiki/ImuSample\\_t](https://github.com/ValveSoftware/openvr/wiki/ImuSample_t)

The stream of IMU data in the form of `vr::ImuSample_t` structures is provided through a new way of sharing data between applications and drivers called `IVRIOBuffers`. For details, see the documentation:

- <https://github.com/ValveSoftware/openvr/wiki/IVRIOBuffer>

A new section in the `steamvr.vrsettings` file supports tracked devices overriding the tracking of other devices. Please see the documentation for details:

- <https://github.com/ValveSoftware/openvr/wiki/TrackingOverrides>

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4505287]

## v1.0.14

Apr 05, 2018

### Tracked Camera

- Added property `Prop_NumCameras_Int32` for drivers to specify the number of cameras a given headset provides.
- Added property `Prop_CameraFrameLayout_Int32` for drivers to specify the frame layout of images delivered (use provided `EVRTrackedCameraFrameLayout` values).

### IVRVirtualDisplay

- Added vsync mode, frame id, and vsync timing data to `Present` call.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4455565]

## v1.0.13

Feb 15, 2018

### General

- Added `VREvent_RoomViewShown` and `VREvent_RoomViewHidden`. Indicates that the "room view" pass-through camera mode has been started or stopped by the user. An application or driver can use this to enable or disable other functionality.

### IVROverlay

- Added TextureType\_DXGISharedHandle. This is used to submit a shared texture handle directly and avoid an extra copy on the resource. This texture type is current only supported for overlays on Windows.

## Driver Interface

- Added TrackedControllerRole\_OptOut. Set a controller to this if you want that controller to be ignored for the purposes of right/left hand selection.
- Added Prop\_NeverTracked\_Bool. Set that property to true on a device if you want to inform the system that the device is never going to have a valid pose. For instance, you might use this for an untracked gamepad.
- Prop\_DistortionMeshResolution\_Int32. Set this property on an HMD device if you want the system to generate a distortion mesh with a different resolution than the default, which is an approximately 50x50 grid across the full texture which is then trimmed to the area within the hidden area mesh.
- Added Prop\_MinimumIpdStepMeters\_Float. Set this to the minimum IPD change in meters that should cause the IPD UI to be shown.
- IVRDriverDirectModeComponent.CreateSwapTextureSet added the ability to specify sample count.
- IVRDriverDirectModeComponent.SubmitLayer

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4366099]

## v1.0.12

Jan 09, 2018

## IVRSystem

- Removed the concept of "input focus" and all the associated APIs.
- Added more granular "system behavior" functions to allow applications to query whether or not they should modify their behavior based on the state of the system. The new function are:
- IsInputAvailable() -- This will return true if the application is receiving input events. For instance, it will return false when the SteamVR dashboard is visible.
- IsSteamVRDrawingControllers() -- This will return true if SteamVR is drawing the user's actual controllers for any reason. It will be true if the dashboard is visible or if the SteamVR keyboard is visible over the application.
- ShouldApplicationPause() -- This will return true is applications should pause (where appropriate.) It generally indicates that the user's attention is being taken by something like the SteamVR dashboard. Multiplayer games and other applications where "pause" is unavailable can ignore this.
- ShouldApplicationReduceRenderingWork() -- This will return true when SteamVR is using more GPU and CPU resources than normal. It is a hint to the application to reduce its own rendering workload. A common way of doing this is to submit smaller render

targets per eye.

IVRCompositor

- Added ability to pass depth info for scene textures. Use `VRTextureWithDepth_t` (or `VRTextureWithPoseAndDepth_t`) and pass `Submit_TextureWithDepth` flag to `Submit`.

## Driver API

- Replaced the API that drivers use to report input (button, trackpad, joystick, etc.) state. See `IVRDriverInput` documentation] for more information:  
<https://github.com/ValveSoftware/openvr/wiki/IVRDriverInput-Overview>
- Replaced the haptic API into drivers with an events. Drivers will receive an event of type `VREvent_Input_HapticVibration` which uses the `hapticVibration` member of the data until to specify parameters.
- `Prop_CameraToHeadTransforms_Matrix34_Array` - HMD devices which support multiple cameras expose camera extrinsic information in this property as an array of 3x4 transforms, one for each camera sensor. It is suggested that new devices also choose one of their cameras as a `primary` and expose its extrinsics as `Prop_CameraToHeadTransform_Matrix34` for backwards compatibility with applications designed to recognize a single camera sensor.
- `Prop_DriverIsDrawingControllers_Boolean` - Drivers should set this property if they are rendering controllers on top of the scene in their own compositor. This will hint the application to stop drawing them separately.
- `Prop_DriverRequestsApplicationPause_Boolean` - Drivers can set this to true in order to hint applications to pause their game (e.g. when bringing up a custom dashboard).
- `Prop_DriverRequestsReducedRendering_Boolean` - Drivers can set this to true in order to hint applications to reduce rendering (e.g. when bringing up a custom dashboard).

## IVROverlay

- Removed `HandleControllerOverlayInteractionAsMouse`. Overlay applications should use the mouse overlay input method instead.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4305765]

# v1.0.11

Dec 05, 2017

General:

- `k_unMaxTrackedDeviceCount` is now 64 to support larger numbers of trackers.

macOS:

- resolves memory leaks with caller/callee MRC/ARC mismatches - consider re-integrating if you bundle or build libopenvr\_api from source

#### IVRCompositor:

- extend SetExplicitTimingMode() with an extra mode to allow the application to instruct the runtime to never implicitly call PostPresentHandoff() instead of relying on sequencing of other calls to decide if it should.

#### Driver Interface:

- Added several new driver properties to support better status UI layout and device-specific default idle icons for custom headset and tracked controller drivers:
- Prop\_ExpectedTrackingReferenceCount\_Int32 - specifies the expected number of tracking sensors or basestations to reserve UI space for, though it can later offer more or fewer actual tracking reference devices.
- Prop\_ExpectedControllerCount\_Int32 - specifies the expected number of tracked controllers to reserve UI space for, though it can later offer more or fewer actual tracked controllers.
- Prop\_NamedIconPathControllerLeftDeviceOff\_String - specifies the placeholder inactive icon for the expected "left" controller when no controller has yet been found or activated.
- Prop\_NamedIconPathControllerRightDeviceOff\_String - specifies the placeholder inactive icon for the expected "right" controller when no controller has yet been found or activated.
- Prop\_NamedIconPathTrackingReferenceDeviceOff\_String - specifies the placeholder inactive icon for the expected tracking reference sensor/base when no tracking references have yet been found or activated.

#### IVRDriverDirectModeComponent:

- Added separate PostPresent call which can be implemented to allow Present to return early after having called AcquiredSync on the provided syncTexture to unblock the compositor from continuing its work. Added GetFrameTiming to allow DriverDirectMode implementations to return additional timing information that only the driver's compositor knows about.
- Added new property Prop\_DoNotApplyPrediction\_Bool - Can be set by drivers which implement DriverDirectMode to tell the compositor not to apply any prediction to the poses provided by the driver.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4268903]

**v1.0.10**

Sep 01, 2017

## IVRCompositor:

- New IVRCompositor::SetExplicitTimingMode and IVRCompositor::SubmitExplicitTimingData for DX12 (<https://github.com/ValveSoftware/openvr/wiki/DirectX12#explicit-timing>) and Vulkan (<https://github.com/ValveSoftware/openvr/wiki/Vulkan#explicit-timing>)
- Adds Submit\_TextureWithPose flag and VRTextureWithPose\_t structure so applications can specify the pose the texture was rendered for when calling IVRCompositor::Submit

## IVROverlay:

- New IVROverlay::CloseMessageOverlay for closing modal dashboard dialogs the process owns

## IVRSystem:

- Adds context to IVRSystem::GetOutputDevice required to discriminate graphics device in Vulkan

## Miscellaneous

- Fixes crashes under some multithreading circumstances using OpenVR VR\_\* APIs
- Updates hellovr\_vulkan sample to use IVRSystem::GetOutputDevice
- Fixes a crash with hellovr\_opengl sample under macOS

# v1.0.9

Jul 19, 2017

## Drivers:

- Added TrackedDeviceDisplayTransformUpdated, which allows HMD drivers to explicitly specify the 4x3 eye-to-head transforms. The classic mechanism of setting Prop\_UserIpzMeters\_Float is still supported.
- Added Prop\_DriverProvidedChaperonePath\_String, which a driver can set on the HMD device to specify a fallback chaperone setup that will be used if no other chaperone data is available for the universe returned by that driver. This string is a JSON object in the same format as the .vrchap file.
- Added VREvent\_WirelessDisconnect and VREvent\_WirelessReconnect events for display redirect drivers to notify the runtime of display connection status.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4059277]

# v1.0.8

Jun 14, 2017

#### General:

- Added VRCompositorError\_InvalidBounds - This is returned when the application passes texture bounds to Submit that are outside the range of -100.0 to 100.0 or are otherwise invalid.
- Added sample programs: hellovr\_vulkan and hellovr\_dx12.

#### IVRVirtualDisplay:

- The IVRVirtualDisplay interface is provided to allow OpenVR driver authors access to the final composited backbuffer intended for the headset's display. The primary expected use case is for wireless transport, though this could also be used for saving output to disk or streaming. From the perspective of the runtime, the VR compositor is interfacing with a virtual rather than an actual display. See [https://github.com/ValveSoftware/virtual\\_display](https://github.com/ValveSoftware/virtual_display)

#### IVRSystem:

- GetOutputDevice interface added for applications to know which primary graphics adapter to use. This returns a LUID on Win32, vk::PhysicalDevice for Vulkan and id on OSX.

#### IVRDriverManager:

- Public API to query installed device drivers.

#### Drivers:

- Drivers can now set Prop\_DriverDirectModeSendsVsyncEvents\_Bool to true to indicate they will call VsyncEvent on their own. This enables avoiding the hardcoded 2.8ms vsync offset for IVRDriverDirectModeComponent implementations.
- Allow drivers to specify which graphics adapter to use by settings Prop\_GraphicsAdapterLuid\_Uint64. This is a LUID on Win32.

#### MacOS/OSX:

- includes a private framework compatible with Xcode 8 and higher. Embedding frameworks as binaries in your macOS app bundle can be error prone - see detailed instructions in issue [#543](#) [#543](#) (comment)

#### Cmake Build System (optional):

- cmake script changes to better support Cygwin
- supports building universal 32/64 binaries on OSX

- supports building as a private OSX framework

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 4009583]

## v1.0.7

May 01, 2017

### General:

- Updated structure packing to be more consistent across platforms.
- New property (Prop\_SecondsFromPhotonsToVblank\_Float) to support rolling illumination displays.
- New property (Prop\_ResourceRoot\_String). When present identifies the properly named driver subdir to be used for sourcing resources.
- New properties which are automatically set based on components a given driver implements.
  - Prop\_HasDisplayComponent\_Bool
  - Prop\_HasControllerComponent\_Bool
  - Prop\_HasCameraComponent\_Bool
  - Prop\_HasDriverDirectModeComponent\_Bool
  - Prop\_HasVirtualDisplayComponent\_Bool

### IVROverlay:

- Added SetOverlayName method.
- Added SetOverlayRenderModel method. Sets render model to draw behind this overlay and the vertex color to use, pass null for pColor to match the overlays vertex color. The model is scaled by the same amount as the overlay, with a default of 1m.
- Added GetOverlayRenderModel accessor.
- Added SetOverlayTransformOverlayRelative method. Sets the transform to relative to the transform of the specified overlay. This overlays visibility will also track the parents visibility.
- Added GetOverlayTransformOverlayRelative accessor.

### IVRServerDriverHost:

- Added GetRawTrackedDevicePoses. Provides access to device poses for drivers. Poses are in their "raw" tracking space which is uniquely defined by each driver providing poses for its devices. It is up to clients of this function to correlate poses across different drivers. Poses are indexed by their device id, and their associated driver and other properties can be looked up via IVRProperties.

### IVRVirtualDisplay:



- New interface drivers can implement to redirect display output.
- Added new device type (TrackedDeviceClass\_DisplayRedirect) for accessories which redirect display output.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3936268]

## v1.0.6

Jan 30, 2017

### General:

- Updated source code and cmake configurations for openvr\_api.dll (and dylib/so) for applications that need a static library.
- Added VREvent\_PropertyChanged event, which is sent when any property changes.
- Added VREvent\_PrimaryDashboardDeviceChanged event, which is sent when the user changes the dashboard laser pointer from one controller to another.

### IVRCompositor:

- Added initial support for DirectX 12 and OSX IOSurfaces. Use at your own risk. Forward compatibility is not guaranteed.
- Added IVRCompositor::ReleaseMirrorTextureD3D11(). Call ReleaseMirrorTextureD3D11 instead of calling Release directly on the texture.

### IVRApplications:

- Added GetCurrentSceneProcessId(), which returns the process ID of the latest process to call VR\_Init with the Scene application type.

### Server driver Interface:

- Greatly simplified IServerTrackedDeviceProvider::Init and its arguments. This function now takes only an IVRDriverContext. From there it can call GetGenericInterface to get the rest of the interface.
- Added global accessor functions for drivers that are similar to those used by applications. Put this line at the start of your IServerTrackedDeviceProvider::Init function (and the equivalent line in Cleanup) to enable them:  
VR\_INIT\_SERVER\_DRIVER\_CONTEXT( pContext );
- IServerTrackedDeviceProvider no longer has enumeration functions for drivers. If the provider contains an HMD it should call TrackedDeviceAdded with the details of that HMD before Init returns. Other devices can be added at any time by calls to TrackedDeviceAdded.
- IVRServerDriverHost::TrackedDeviceAdded now takes all the required values for a new tracked device, including the device class and device driver interface pointer.

- Replace the property functions on ITrackedDeviceServerDriver with the IVRProperties interface and the CPropertyHelpers helper functions. This should result in significantly less boilerplate code in drivers and allows drivers to invalidate properties immediately instead of waiting for the client-side cache to expire. Use vr::VRProperties() to get the new helper interface. See the sample driver for details.
- Added a new "enable" setting to all drivers that will prevent the driver DLL from being loaded. The enable flag has been removed from the sample driver.
- IServerDriverHost has been renamed to IVRServerDriverHost no longer contains a few functions that are now handled by property setters.
  - GetSettings() is now handled with vr::VRSettings()
  - PhysicalIpdSet() is now handled by setting the Prop\_UserIpzMeters\_Float property.
  - TrackedDevicePropertiesChanged() is now handled automatically when a property is set.
  - MCImageUpdated() was undocumented and not useful outside the Lighthouse driver. It has been removed.

#### CVRHiddenAreaHelpers:

- This new helper class provides access to the hidden area mesh via the property system. You can access it with vr::VRHiddenArea() in a server driver.

#### IDriverLog:

- This interface has been renamed to IVRDriverLog

#### IClientTrackedDeviceProvider:

- Client drivers have been removed from the system. Drivers are no longer loaded into client processes. The functionality that used to be held in client drivers has moved:
  - BIsHmdPresent is implemented by looking for USB VID and PID values as specified in the driver manifest file:  
<https://github.com/ValveSoftware/openvr/wiki/DriverManifest>
  - GetHiddenAreaMesh is implemented via properties and the CVRHiddenAreaHelpers class. (See above)
  - GetMCImage was undocumented and not useful outside of the Lighthouse driver. It has been removed.
  - Watchdog mode, which allows SteamVR to start automatically on hardware activity, has been moved to a new driver type IVRWatchdogProvider.

#### IVRWatchdogProvider:

- This provider is only loaded in app type SteamWatchdog. It monitors the hardware for changes and calls vr::VRWatchdogHost()->WatchdogWakeUp() if an event occurs that should start SteamVR. This is entirely optional. A driver that doesn't implement this provider will just not wake up SteamVR on hardware activity.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3811839]

## v1.0.5

Dec 12, 2016

General:

- Added final support for submitting Vulkan overlays and eye images. See this document for more information: <https://github.com/ValveSoftware/openvr/wiki/Vulkan>
- Added TextureType\_Vulkan, VRVulkanTextureData\_t
- Rename EGraphicsAPIConvention -> ETextureType.
- New synchronous MessageOverlay API. Use IVROverlay::ShowMessageOverlay to display a message with up to four buttons.
- Added ETrackedDeviceClass, GenericTracker
- Added ETrackedPropertyError, PermissionDenied

IVRSystem (v15):

- GetProjectionMatrix signature change, removed EGraphicsAPIConvention eProjType

IVRCompositor (v19):

- Added GetVulkanInstanceExtensionsRequired, GetVulkanDeviceExtensionsRequired

IVROverlay (v14):

- Added VROverlayFlags\_VisibleInDashboard
- GetOverlayTexture signature change (includes overlay texture bounds)
- Added GetOverlayFlags
- Added ShowMessageOverlay

IVRTrackedCamera (v13):

- Renamed GetCameraIntrinsics

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3739504]

## v1.0.4

Nov 16, 2016

General:

- Further refinement to Vulkan support. Note: the interface is still subject to change.

#### IVRSystem (v14):

- ComputeDistortion signature change. Previous version used nan as error signaling. New version returns explicit success.
- GetHiddenAreaMesh takes optional EHiddenAreaMeshType argument to request Standard, Inverse, or LineLoop mesh.

#### IVRCompositor (v18):

- New submit error, VRCompositorError\_AlreadySubmitted (if you submit the same eye twice)
- GetFrameTimings signature change
- Exposed GetCurrentFadeColor, GetCurrentGridAlpha

#### IVROverlay (v13):

- Added SetOverlayIntersectionMask

#### IServerDriverHost:

- Added PollNextEvent

#### IClientDriverHost:

- Removed GetSettings
- Added GetGenericInterface

#### ClientTrackedDeviceProvider (v5):

- GetHiddenAreaMesh takes additional argument, EHiddenAreaMeshType

#### Pull Requests integrated:

[#48](#) Format README in Markdown

[#58](#) correct detection of GCC

[#68](#) helloworldoverlay: Specify c++11 to fix gcc errors

[#76](#) Remove duplicate, unneeded glBindBuffer call.

[#135](#) Invert m\_mat4HMDPose after assignment copying.

[#191](#) Document additional methods

[#278](#) include stdbool.h on non windows platform

[#306](#) Multi-platform CMake support.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3697868]

## v1.0.3

Oct 10, 2016

General:

- Added preliminary Vulkan support
- Added Button ID `k_EButton_ProximitySensor`. Use this for raw access to a proximity sensor if there is one.
- Fixed spelling error in `EVROverlayError`, "`VROVerlayError_KeyboardAlreadyInUse`"

IVRSettings:

- Improved default value handling. Defaults are no longer passed with each individual call to `GetBool`, `GetInt32`, `GetFloat`, and `GetString`. Instead, system-wide defaults are specified in "`default.vrsettings`" (in the `resources/setting/` directory of either the runtime or the `driver_xxx` directory.) If not explicitly defined, the default will be `false`, `0`, `0.0` or `""` with an error of `VRSettingsError_UnsetSettingHasNoDefault`.
- Added explicit `EVRSettingsError` for `VRSettingsError_JsonParseFailed`

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3647965]

## v1.0.2

Jul 05, 2016

General:

- Split several `VR_Init` error return values out into unique causes.
- Added `Prop_ControllerRoleHint_Int32`, which a controller driver can optionally use to return the best matching `ETrackedControllerRole` type for the controller. Use this to specify a controller that is physically shaped for one hand only or when the driver is able to determine which hand is holding the controller through sensors or other means.
- Added `VREvent_OverlayFocusChanged`, which is sent when an overlay was under the laser mouse and isn't anymore or when there was no overlay under the laser mouse and now there is.

IVRApplications:

- Added `LaunchApplicationFromMimeType`. This allows the caller (usually Steam) to start the application that is associated with that mime type if it is not already running. If the application is not running it will be started with the specified arguments appended to existing arguments for the app. If the application is already running it will receive `VREvent_ApplicationMimeTypeLoad` and can retrieve the arguments by passing

event.applicationLaunch.unArgsHandle to IVRApplications::  
GetApplicationLaunchArguments.

- Added other support functions for mime types.

#### IVRCompositor:

- Split out Compositor\_FrameTiming's m\_flSceneRenderGpuMs into m\_flPreSubmitGpuMs and m\_flPostSubmitGpuMs. Post-Submit time was not previously being recorded.

#### IVROverlay:

- Added VROverlayFlags\_SortWithNonSceneOverlays. This allows the scene application's overlays to sort with other non-scene overlays.
- Added SetOverlayTexelAspect/GetOverlayTexelAspect. This allows an overlay to control the aspect ratio of its texels. By default all overlay texels are square (i.e. 1.0)
- Added SetOverlaySortOrder/GetOverlaySortOrder. This allows an overlay to be forced to be sorted above or below other overlays. It is most useful for overlays from the scene application that want to explicitly control their draw order.

#### IVRTrackedCamera:

- Added accessors to work with tracked camera gpu resources:
  - \*\* GetVideoStreamTextureSize
  - \*\* GetVideoStreamTextureD3D11
  - \*\* GetVideoStreamTextureGL
  - \*\* ReleaseVideoStreamTextureGL

#### IVRResources:

- New interface to allow access to resource files in the runtime or user-installed drivers.

#### IVRCameraComponent:

- Simplified interface by removing duplicate functions already provided by properties and deprecating the arbitrary distortion methods.

#### IClientDriverHost:

- Added WatchdogWakeUp. When a client driver is in Watchdog mode it should call this if a hardware event happens that should cause the system to wake up. The Lighthouse driver, for instance, calls this method when a system button is pressed or a controller is turned on.

#### IClientTrackedDeviceProvider:

- Added eDriverMode argument to Init. When this is passed as ClientDriverMode\_Watchdog the driver should enter a low-power state where hardware is being monitored. If the driver does not support watchdog mode it should return VRInitError\_Init\_LowPowerWatchdogNotSupported.

## v1.0.1

Jun 10, 2016

IVROverlay:

- Added event VREvent\_ImageFailed which is sent when SetOverlayFromFile is called and the file load fails.

IVRScreenshots:

Added a new API via IVRScreenshots for screenshot support in VR. This feature is still being developed and is currently in a beta state. You can enable screenshot capture by selecting Enable Screenshots in the general settings of the VRMonitor. Documentation for this new API is available at, [https://github.com/ValveSoftware/openvr/wiki/IVRScreenshots\\_Overview](https://github.com/ValveSoftware/openvr/wiki/IVRScreenshots_Overview)

Driver API:

- Split Oculus driver direct mode into a new component and out of IVRDisplayComponent. Other drivers generally don't need to implement this component.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3492473]  
master v1.0.1

## v1.0.0

May 24, 2016

IVRApplications:

- Added VREvent\_ApplicationListUpdated event. This is sent whenever an application manifest is added, removed, or updated.

IVRRenderModels:

- Added VREvent\_ModelSkinSettingsHaveChanged event. This is sent when the user has selected a new controller or base station/camera replacement model. If your application uses the IVRRenderModel interface to load the user's actual controllers it should reload them after receiving this event.

- Added `GetRenderModelOriginalPath` function. This returns the non-overridden full path to the render model so controller/base station/camera skinning can be avoided if the application requires it.
- Added `GetRenderModelThumbnailURL` function. This returns a URL (usually to a file) for a 4x3 aspect ratio thumbnail image that can be used to preview the render model.
- Added `GetRenderModelErrorNameFromEnum` so applications don't need to provide their own lookup table to interpret these errors.

#### IVRCompositor:

- Added cumulative stats tracked per-application and associated accessor.
- Added interface to access the mirror textures (per-eye). This is the undistorted view with chaperone and overlays (e.g. dashboard) drawn on top.
- Added preliminary support for screenshots. This interface is not stable yet and should be ignored for now.

#### IVRTrackedCamera:

- Exposed the initial version. Provides developers access to poll streaming frames from the HMD front facing camera with the associated tracked HMD pose. The image is available as either a pre-corrected distorted view or corrected undistorted view.
- Added a thin Qt example for getting and showing the camera image/pose.

#### IVROverlay:

- Added interface to get size of a texture used by an overlay. This is useful when setting the overlay texture by filename.
- Added ability to render side-by-side stereo content and panoramas.

#### Driver Interface:

- `IVRCameraComponent` is still in a development phase.
- Added provider method `GetInterfaceVersions`. This contains all the version numbers of the other interfaces in the driver. Drivers should return `vr::k_InterfaceVersions` from the version of `openvr_driver.h` that they were compiled against.
- `GetTrackedDeviceDriver` and `FindTrackedDeviceDriver` no longer take an interface version. The caller will expect the version of the interface returned in `GetInterfaceVersions()`

#### HelloVR sample:

- Fixed compile error around `glDebugMessageCallback` that could happen with certain versions of the OpenGL headers.
- Fixed compile error from `LoadRenderModel->LoadRenderModel_Async` name change.
- Fixed compile error from `LoadTexture->LoadTexture_Async` name change.



# v0.9.20

May 03, 2016

General:

- Added `VREvent_InputFocusChanged` - This event is sent whenever input focus changes from one process to another.
- Added `VREvent_AudioSettingsHaveChanged` - This event is sent whenever the user changes audio settings.
- Added `VREvent_ReprojectionSettingHasChanged` - This event is sent when the user turns interleaved reprojection on or off.
- Added `VRInitError_Compositor_FirmwareRequiresUpdate` - This error code will be returned by the compositor when there is a mandatory firmware update pending.

IVRCompositor:

- `Compositor_FrameTiming` now contains `m_nReprojectionFlags`. This will be 0 or more of:
  - . `VRCompositor_ReprojectionReason_Cpu`
  - . `VRCompositor_ReprojectionReason_Gpu`
- Added `ForceReconnectProcess`. Call this to force a disconnect and reconnect to the compositor.
- Added `SuspendRendering`. Call this when you know your process isn't going to render frames for a while. This often happens while loading new maps.

IVROverlay:

- Added overlay flag `VROverlayFlags_TransferOwnershipToInternalProcess`. This flag causes overlays owned by one process to be transferred to the new process spawned by a call to `IVRApplications::LaunchInternalProcess`. This can be useful for smoothing over such transitions from the user's point of view.

Driver Interface:

- Added `VREvent_DriverRequestedQuit` - A driver can post this event to ask the system to shut down.
- Added `IVRDisplayComponent::GetNextSwapTextureSetIndex` to retrieve the index in the texture set to use for the next frame's render targets.
- Changed `IVRDisplayComponet::CreateSwapTextureSet` to create three textures to allow a slightly longer round-robin list.

NOTE: As of SDK 0.9.20 the driver interfaces will remain backward-compatible indefinitely. Future runtime changes will not break the ability to load drivers built against at least SDK 0.9.20.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3426544]

## v0.9.19

Mar 09, 2016

### IVRRenderModels:

- Added LoadIntoTextureD3D11\_Async. Loads render model texture asynchronously and copies into provided texture (as opposed to creating one and returning it).

### IVROverlay:

- Added SetOverlayRenderingPid/GetOverlayRenderingPid to let you delegate rendering of an overlay's texture to another process
- Added new event VREvent\_OverlaySharedTextureChanged, this is fired when the backing shared texture for an overlay target changes, which is useful when using GetOverlayTexture
- Added GetOverlayTexture/ReleaseNativeOverlayHandle to let you retrieve a native texture pointer backing an overlay target

### IVRApplications:

- Added new application type, templates. Use the "is\_template" key in your manifest to define an application of this type, and the LaunchTemplateApplication call to launch this application. Template applications let you override the keys in the manifest at runtime, useful for launching viewer style apps that want optional command line arguments and for you to transition from the application back to itself again.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3339101]

## v0.9.18

Mar 04, 2016

### General:

- Added VREvent\_Compositor\_ChaperoneBoundsShown and VREvent\_Compositor\_ChaperoneBoundsHidden, which are sent when the chaperone bounds become visible/hidden.
- Added VREvent\_SeatedZeroPoseReset, which is sent whenever the user resets the seated zero pose in the dashboard or via the API.

### IVROverlay:

- Added new overlay flag, `VROverlayFlags_ShowTouchPadScrollWheel`, to let you cause a scrollwheel to be draw on the active controller even if it isn't in scroll mode

#### IVRApplications:

- Added `LaunchInternalProcess`, which allows an application to switch which process is providing 3D frames for that application without showing any application transition UI.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3330105]

## [v0.9.17](#)

Feb 26, 2016

#### General:

- Added events for application to temporarily hide and re-show render models (`VREvent_HideRenderModels`, `VREvent_ShowRenderModels`).
- Added `VREvent_KeyboardDone` which is sent when DONE button clicked on keyboard.
- There is a change to how the `openvr.h` manages talking to the `openvr_api.dll`. We now have inline functions that acquire the interface pointers so when you build you will get the correct interface pointer even if the underlying `openvr_api.dll` is newer than the one you built against. This will enable environments where multiple codebases need to coexist in the same process such as some game engines.

#### IVRApplications:

#### Compositor:

- Added new error code for detecting non-D3D11 class hardware (`VRInitError_Compositor_D3D11HardwareRequired`).
- New interface for individually getting last tracked device poses returned by `WaitGetPoses` (`GetLastPoseForTrackedDeviceIndex`).
- New API used to determine if the current app should render using lower resources (`ShouldAppRenderWithLowResources`). Typically this will return true when the dashboard is showing but other scenarios could result in this behavior also.

#### Settings:

- New speaker related settings (`usingSpeakers`, `speakersForwardYawOffsetDegrees`).

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3321024]

## [v0.9.16](#)

Feb 15, 2016

Note: This update includes significant changes to the OpenVR driver interface to simplify driver development and support backward compatibility. Drivers will need to update to at least this version to continue to be supported.

#### General:

- Added `VREvent_TrackedDeviceRoleChanged`, which is sent when right/left handedness of a controller changes.

#### Settings:

- Added `RemoveSection`, which removes an entire section from the settings.

#### Render Models:

- Changed render model and texture loading to be asynchronous. `LoadRenderModel` has become `LoadRenderModel_Async` and `LoadTexture` has become `LoadTexture_Async`. Check the `EVRRenderModelError` returned by these functions to determine if the requested resource is loaded asynchronously.
- `LoadTextureD3D11` has become `LoadTextureD3D11_Async` which loads a texture into a D3D11 resource asynchronously.
- Added `VRComponentProperty_IsScrolled`, which can be returned for render model components that are currently scrolling.
- Added `RenderModel_ControllerMode_State_t` to `GetComponentState`. This allows the caller to specify whether the component is scrolled and is used for devices sending scroll events to overlays. Pass `NULL` if you don't care about this.

#### Driver Interface:

- Removed `TrackedDeviceDriverInfo_t`. The information that used to be in this struct is now provided via properties. Added `Prop_DeviceClass_Int32` for the one member of the struct that didn't already have a matching property.
- `IServerDriverHost::TrackedDeviceAdded` now takes a serial number string. The driver will be called back to get specifics via properties.
- `Prop_SerialNumber_String` and `Prop_DeviceClass_Int32` are now required for every tracked device driver.
- Added `deviceIsConnected` to `DriverPose_t`. When a connection to a device is lost the driver should send another pose with this bool set to false.
- Removed `ITrackedDeviceServerDriver::GetId`. This information is retrieved via `Prop_SerialNumber_String` now.
- Functions that aren't universal to all tracked devices have been moved onto driver components. These are retrieved with the `GetComponent` function.
- Display-related functions have moved to `IVRDisplayComponent`. This includes:
  - `GetWindowBounds`

- IsDisplayOnDesktop
- IsDisplayRealDisplay
- GetRecommendedRenderTargetSize
- GetEyeOutputViewport
- GetProjectionRaw
- ComputeDistortion
- CreateSwapTextureSet
- DestroySwapTextureSet
- DestroyAllSwapTextureSets
- SubmitLayer
- Present
- Controller-related functions have been moved to `IVRControllerComponent`. This includes:
  - `GetControllerState`
  - `TriggerHapticPulse`
- Camera-related functions have moved onto `IVRCameraComponent`, but this interface is still in flux, so you should avoid implementing it in your driver.
- `IServerDriverHost` lost the `TrackedDeviceInfoUpdated` function. If property values change, the driver should call `TrackedDevicePropertiesChanged` instead.
- `GetSettings` on `IServerDriverHost` and `IClientDriverHost` now take an interface string. Pass the version of `IVRSettings` that your driver was compiled against.
- Interface versions were added to a few other places to support backward compatibility.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3299073]

## v0.9.15

Jan 29, 2016

General:

- Several event members have been changes from enum values to plain-old-data types to guarantee compatibility across compiler vendors.

`IVRSystem`:

- `PollNextEvent()` and `PollNextEventWithPose()` now take a parameter for the size in bytes of the `VREvent` object you are passing in
- The data member in `VREvent_t` is now variable length and at the end of the structure.

`IVRApplications`:

- Added `VRApplicationProperty_LastLaunchTime_UInt64`, which returns the last launch time for an application in seconds since January 1, 1970.

IVRCompositor:

- Compositor\_FrameTiming is totally different. What should we tell people, Aaron?

IVROverlay:

- Added the ability to render an overlay wrapped around a component on a tracked controller when in the dashboard or compositor.
- k\_EGamepadTextInputModeSubmit added for chat-style use of the keyboard where instead of a "Done" button there is a "Submit" button and it doesn't make the keyboard go away.
- PollNextOverlayEvent() now takes a parameter for the size in bytes of the VREvent object you are passing in
- For overlay applications there are now two new event types you can listen to, VREvent\_Scroll and VREvent\_TouchPadMove. These events are generated if the VROverlayFlags\_SendVRScrollEvents or VROverlayFlags\_SendVRTouchpadEvents flag is set on your overlay window.

IVRRenderModels:

- Added RenderModelHasComponent, which returns true if the named model has the named component.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3265813]

## [v0.9.14](#)

Dec 30, 2015

- IVRSystem
  - Added PerformanceTestEnableCapture(bool) for performance testing tools to toggle when frame timing data should be recorded.
  - Added PerformanceTestReportFidelityLevelChange(int) for performance testing tools to report when they've changed fidelity settings.
- General
  - Added present count field to Compositor\_FrameTiming.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3205113]

## [v0.9.13](#)

Dec 23, 2015

- IVRSystem

- Added new APIs `GetTrackedDeviceIndexForControllerRole()` and `GetControllerRoleForTrackedDeviceIndex()` to get the correct devices for the left and right hand controllers.
- `IVRChaperone`
  - API `GetBoundsColor` has changed to also return `pOutputCameraColor`.
- `IVROverlay`
  - Added `GetTransformForOverlayCoordinates()` to return the 3d space transform for a given 2d point in an overlay.
  - Added `SetKeyboardTransformAbsolute()` to set the keyboard position in 3d space
  - Added `SetKeyboardPositionForOverlay()` to set the keyboard position appropriate for an overlay
- `IVRControlPanel` removed
- Misc new types added

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3201751]

## v0.9.12

Nov 30, 2015

- Renamed many enums to a more consistent naming scheme. Typedefs are in place to allow old code to continue building, but those will be removed at some point in the future.
  - `HmdError` -> `EVRInitError`
  - `Hmd_Eye` -> `EVREye`
  - `GraphicsAPIConvention` -> `EGraphicsAPIConvention`
  - `HmdTrackingResult` -> `ETrackingResult`
  - `TrackedDeviceClass` -> `ETrackedDeviceClass`
  - `TrackingUniverseOrigin` -> `ETrackingUniverseOrigin`
  - `TrackedDeviceProperty` -> `ETrackedDeviceProperty`
  - `TrackedPropertyError` -> `ETrackedPropertyError`
  - `VRSubmitFlags_t` -> `EVRSubmitFlags`
  - `VRState_t` -> `EVRState`
  - `CollisionBoundsStyle_t` -> `ECollisionBoundsStyle`
  - `VROverlayError` -> `EVROverlayError`
  - `VRFirmwareError` -> `EVRFirmwareError`
  - `VRCompositorError` -> `EVRCompositorError`
- Renamed all `HmdError_*` enum values to `VRInitError_*` to match the new enum name.
- `VR_GetStringForHmdError` was renamed to `VR_GetVRInitErrorAsEnglishDescription` to match the new enum name and make it clear what this function actually does.
- `VR_GetVRInitErrorAsSymbol` was added. This returns the literal `EVRInitError` value name instead of a user-facing string. Use this to look up localized strings to display to users.
- `IVRRenderModels`

- Introduced Component interface, which allows client applications to draw, label, or interact with tracked object components.  
Components may define coordinate systems, renderable geometry, associations with button state, and supports button / axis dynamics.

#### Example uses of the IVRRenderModels Component API

- Compositor controller rendering reflects live button/trackpad/trigger motion dynamics
  - Client applications can query controller-agnostic render label "attachment points"
  - Client applications can query controller-agnostic hand-pose
- LoadRenderModel does not load textures automatically, instead an additional call to LoadTexture is required. This enables texture reuse across model components.
- Added new event VREvent\_IpdChanged, which is sent when the physical IPD has changed on hardware that supports that. Because IVRSystem::GetEyeToHeadTransform is based on the user's IPD, applications which cache the matrix returned from that function will need to query it again. Applications that call GetEyeToHeadTransform each frame can safely ignore this event.
- Functions that are used by an application to create its own window have moved to IVRExtendedDisplay. Calls to get this interface will return NULL and error VRInitError\_Init\_NotSupportedWithCompositor if VR Compositor is already running. Applications will need to use the compositor in that case. The functions that were moved are:
  - GetWindowBounds
  - GetEyeOutputViewport
  - GetDXGIOutputInfo
- IVRNotifications has been changed to support different types (transient and persistent) and styles of notifications. The notification system remains a work-in-progress and is not yet fully implemented.
- Changed several functions that take a texture pointer to take a Texture\_t struct instead. This struct allows the caller to identify the API and color space of the source texture in addition to the texture itself.
- IVROverlay::SetOverlayGamma has been replaced with SetOverlayTextureColorSpace which allows applications to specify that their texture is in linear or SRGB space.
- IVRCompositor::Submit may return the following two new errors now rather than silently failing:
  - TextureUsesUnsupportedFormat
  - SharedTexturesNotSupported (if your app does not use DXGI 1.1 or greater)
- IVRCompositor::GetLastError has been removed. Errors are reported in the log.
- IVRCompositor::Get/SetVSync has been removed.
- IVRCompositor::Get/SetGamma has been removed (specify color space when submitting textures now instead).
- Added IVRCompositor::GetLastPoses - returns the last set of poses returned by WaitGetPoses.



- Added new application type `VRApplication_Background`, which will not start SteamVR if it isn't already running and will not keep it running if it would otherwise shutdown.
- Added `IVRSystem::AcknowledgeQuit_Exiting` to allow an application to acknowledge an incoming Quit event. This extends the timeout before vrserver kills the app from 5 seconds to 60 seconds.
- Added `IVRSystem::AcknowledgeQuit_UserPrompt` to allow an application to tell the system that the user should be prompted to save data before exiting. This will cause the dashboard to be shown with UI that prompts the user to cancel the quit/transition request or to quit the app anyway. All timeouts will be cancelled once this call is made to give the user a chance to respond.
- Quit events now have a `bForced` field (under `event.data.process`). If this is this field is true calls to `AcknowledgeQuit_UserPrompt` will be ignored and the app will be killed automatically after a timeout if it doesn't exit on its own.
- The concept of a "home application" has been removed from the API. No 3D application will be launched automatically when SteamVR starts.
- Chaperone data is automatically reloaded in all applications when `VRServer` receives `VREvent_ChaperoneSettingsHaveChanged`. It's no longer necessary to watch for this event in each application.
- `IVRChaperoneSetup` has been added to the API which includes access to Chaperone Bounds polygons.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3147743]

## v0.9.11

Oct 23, 2015

IVROverlay:

- Added the concept of gamepad focus to dashboard overlays. Only the overlay with gamepad focus should pay attention to `Xinput` or other gamepad events. These are supported by new events and these `IVROverlay` functions:
  - `GetGamepadFocusOverlay` - Returns the overlay that currently has gamepad focus.
  - `SetGamepadFocusOverlay` - Sets the overlay that currently has gamepad focus. The dashboard will do this automatically for the active overlay.
  - `SetOverlayNeighbor` - Set the up/down/left/right relationship between two overlays.
  - `MoveGamepadFocusToNeighbor` - Overlays should call this when the user would navigate off of that overlay. OpenVR will shift the focus to the neighbor in that direction if there is one.

IVRCompositor:

- Added PostPresentHandoff - Should be called immediately after Present, only if you can't call WaitGetPoses instead (e.g. due to not having a separate render thread).
- Added GetLastPoses - Returns the last set of poses that would normally have been gotten through calling WaitGetPoses.

#### IVRTrackedCamera:

- This is a work in progress API for forward facing cameras.

#### IVRChaperone:

- Added GetPlayAreaSize - Returns the size of the Play Area in X/Z dimensions.
- Renamed GetSoftBoundsInfo to GetPlayAreaRect - Returns 4 corners of Play Area centered around the Standing Center.
- Removed GetHardBoundsInfo
- Removed GetSeatedBoundsInfo

#### IVRChaperoneSetup:

- Added GetWorkingPlayAreaSize - Returns the size of the working Play Area in X/Z dimensions.
- Renamed GetWorkingSoftBoundsInfo to GetWorkingPlayAreaRect - Returns 4 corners of working Play Area centered around the Standing Center.
- Replaced SetWorkingSoftBoundsInfo with SetWorkingPlayAreaSize - Sets the X/Z dimensions of the working Play Area.
- Renamed GetWorkingHardBoundsInfo to GetWorkingCollisionBoundsInfo - Returns quads for the working Collision Bounds.
- Renamed SetWorkingHardBoundsInfo to SetWorkingCollisionBoundsInfo - Returns quads for the working Collision Bounds.
- Added GetLiveCollisionBoundsInfo - Returns quads for the current Collision Bounds (formerly known as Hard Bounds).
- Removed all TagPoses interfaces and functionality.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3047315]

## v0.9.10

Oct 07, 2015

- Split VREvent\_SceneApplicationChanged into two events. SceneApplicationChanged will be sent when switching from one app drawing the 3D scene to another drawing the 3D scene. SceneFocusChanged will be sent when the process that is allowed to draw in 3D has changed.
- Added events to notify launchers when about ongoing application transitions.

- Added button IDs for DPad and A buttons that will be sent when the user presses on the trackpad while in the VR dashboard.
- Added new HmdError\_ShuttingDown, which will be returned when a new application calls VR\_Init while SteamVR is in the process of shutting down.
- Added new Application errors for when an application can't start because another application transition is already in progress.
- Added new IVRApplications methods that are intended for launchers, but may be of more general use:
  - GetTransitionState() - returns what phase of the transition process the system is in
  - GetStartingApplication() - returns the app key of the application that is starting up
  - PerformApplicationPrelaunchCheck() - kicks off the application transition process and returns what kind of event the caller should wait for before starting its new application
  - GetApplicationsTransitionStateNameFromEnum() - Turns the results of GetTransitionState() into a string for logging purposes
- Added more const to the settings strings to avoid compiler warnings on some platforms.
- Added IVRCompositor methods:
  - ShowMirrorWindow/HideMirrorWindow - Shows or hides a window that displays the left undistorted eye that the user is seeing in the HMD
  - CompositorDumpImages - Writes all textures that the compositor knows about (including overlays) to a "screenshots" folder in the root of the runtime
  - GetFrameTimeRemaining - Returns the time left (in seconds) until the end of the current frame
  - GetLastFrameRenderer - Returns the process ID of the process that rendered the most recent frame to the HMD. If the compositor is showing its grid, this function will return 0 even if another app has scene focus.
- Added preliminary support for controller-based keyboard input
  - Use IVROverlay::ShowKeyboard() to show the keyboard.
  - Currently only minimal mode is supported
  - Listen for KeyboardCharInput events to receive key input while the keyboard is up. The cNewInput buffer will contain input.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 3016204]

## v0.9.9

Sep 10, 2015

- Added events (VREvent\_TrackedDeviceUserInteractionStarted and VREvent\_TrackedDeviceUserInteractionEnded) and a new interface method (IVRSystem:: GetTrackedDeviceActivityLevel) to hint an application about whether a device is being held or worn by a user.
- Added events (VREvent\_FocusEnter and VREvent\_FocusLeave). These are sent to overlays with automatic mouse processing when the cursor moves onto or off of an overlay. Use them to clean up hover states in your overlay UI.

- Dashboard overlays will now be send pressed and unpressed events when they are the current overlay.
- Added `IVRSystem::GetRawZeroPoseToStandingAbsoluteTrackingPose()`, which allows an application to transform between standing and the raw coordinate system.
- Added method (`IVRChaperone::SetSceneColor`) to let the app provide a hint to the compositor about the scene color so it can pick reasonable bounds colors.
- Added methods to (`GetBoundsColor`, `AreBoundsVisible`, `ForceBoundsVisible`) to let applications query the current state of the bounds.
- `IVRCompositor::Submit` now takes a flag parameter that allows an application to submit images that have already been distorted.
- Added methods (`SetSkyboxOverride` and `ClearSkyboxOverride`) to `IVRCompositor` to allow applications to control the skybox that will be visible when the compositor draws. This is useful for when an application knows it is not going to render for a while (because it is loading a level or doing some other blocking action.)

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2961972]

## v0.9.8

Aug 13, 2015

General:

- Added new event `VREvent_SceneApplicationChanged`. It is sent whenever the application drawing the 3D scene changes.
- `VR_Init` now takes an optional argument to specify application type. The types are `Scene` (an app drawing in 3D), `Overlay` (a dashboard overlay), and `Other`. The default is `Scene`.
- For `Scene` applications, `VR_Init` will now block until the current scene application exits.
- Added new error return code (`HmdError_Init_InitCanceledByUser`) from `VR_Init` when the user decides to not exit the current `Scene`.
- Only one `Scene` application is allowed to be blocking on `VR_Init` at a time. If a second `Scene` application tries `VR_Init` will return `HmdError_Init_AnotherAppLaunching`.

`IVRCompositor`:

- Only `Scene` applications are allowed to call `WaitGetPoses` or `Submit`. `VRCompositorError_IsNotSceneApplication` will be returned if a non-`Scene` application tries to call these functions.

`IVROverlay`:

- Exposed curve parameters for high-quality overlay.

`IVRApplications`:

- This new interface allows the following operations:
  - Applications, installers, and distribution systems can register applications with the system.
  - Launchers can enumerate the list of installed applications and information about those applications
  - Launchers can start applications by key instead of needing to know the details of how to launch the application.
  - Dashboard overlays can be marked to auto-launch at startup so they are always available.
  - Scene applications can be set as the Home application so they start when the system does. The user can also access the home application at any time via the Home button on the dashboard.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2918409]

## [v0.9.7](#)

Aug 06, 2015

- Added new function VR\_IsRuntimeInstalled() to openvr.h. It returns true if the VR runtime directory was found.
- Added EVREventType::EVREventType\_VREvent\_StatusUpdate to notify subscribers of the event that the OpenVR overall system entered a new status state. Possible states are defined by VRStatusState\_t and stored in VREvent\_status\_t
- Added EVREventType::VREvent\_ChaperoneUniverseHasChanged to notify when the universe ID of the chaperone system changed. This message is usually sent once, when the ID changes from 0 to the actual universe ID.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2912031]

## [v0.9.6](#)

Jul 15, 2015

Rearranged type declarations in the headers so they're in the right order.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2879780]

## [v0.9.5](#)

Jul 14, 2015

IVROverlay:

- Added GetOverlayKey and GetOverlayName which return the key or name of an overlay from its handle.
- Added GetOverlayImageData which returns the current raw image data from an overlay.
- Improved reliability of SetOverlayRaw and SetOverlayFromFile.
- Added VREvent\_ImageLoaded, which is sent when SetOverlayRaw or SetOverlayFromFile finish updating the overlay texture.
- Added VROverlayError\_UnableToLoadFile, which may be returned if SetOverlayFromFile fails.
- Added ShowDashboard, which will make the dashboard visible if a dashboard manager is present. We aren't shipping a dashboard manager quite yet, so this function currently does nothing.

#### IVRNotifications:

- This new interface allows applications to post notifications to the user. Details and documentation are coming soon.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2877976]

## v0.9.4

Jul 07, 2015

Removed Microsoft C Runtime DLL dependency from openvr\_api.dll. Applications should be able to use this with any C runtime and won't require that the VS2010 C runtime be installed.

## v0.9.3

Jul 06, 2015

#### General:

- Added accessors for all public OpenVR interfaces. These eliminate the need to call VR\_GetGenericInterface and then cache a pointer to the result. The functions are:
  - vr::IVRSystem \*VRSystem()
  - vr::IVRChaperone\*VRChaperone ()
  - vr::IVRCompositor\*VRCompositor ()
  - vr::IVROverlay\*VROverlay ()
  - vr::IVRRenderModels \*VRRenderModels()
  - vr::IVRControlPanel\*VRControlPanel()
- Added VREvent\_Quit to tell applications when they should exit. Applications that don't exit will be killed after five seconds.
- Added VREvent\_ProcessQuit to tell applications when another OpenVR process has exited.

## C API:

- Added missing "VR\_" prefixes.
- Added missing global entry points (VR\_Init, VR\_Shutdown, etc.).

## IVRSystem:

- Added GetSortedTrackedDeviceIndicesOfClass helper function. Useful for determining left vs right controller, etc.
- Added properties to get battery and changing state for controllers. These are currently returning dummy values in the Lighthouse driver.
  - Prop\_DeviceIsWireless\_Bool
  - Prop\_DeviceIsCharging\_Bool
  - Prop\_DeviceBatteryPercentage\_Float
  - Prop\_StatusDisplayTransform\_Matrix34

## IVRNotifications:

- Experimental version of Notifications is now in the SDK. There WILL be a breaking change to this API in a future SDK.
- Notifications API will enable creating Notification Sources that send alerts to users while they are in a VR experience.



## IVRControlPanel:

- Added QuitProcess(), which tells another OpenVR process to quit. Pass in 0 to tell all OpenVR processes to quit. In the all case,

## IVRCompositor:

- Compositor\_FrameTiming now additionally reports the following values:
  - m\_flFrameIntervalMs
  - m\_flSceneRenderCpuMs
  - m\_flSceneRenderGpuMs
  - m\_flCompositorRenderCpuMs
  - m\_flCompositorRenderGpuMs
  - m\_flPresentCallCpuMs
  - m\_flRunningStartMs
- Removed IVRCompositor::SetGraphicsDevice. Applications pass in the type of a texture with each Submit call instead.
- Submit's TextureBounds no longer need to be vertically flipped for OpenGL textures.

## IVROverlay:

- Renamed the system overlay to the Dashboard. Added CreateDashboardOverlay() to allow applications to create overlays that are intended to appear in the dashboard.

- Removed Set/GetOverlayVisibility. This used to be used to set an overlay as a system overlay.
- Added Set/GetOverlayColor. The overlay color (including alpha) is multiplied by the overlay's output texture value.
- Applications that use IVROverlay no longer need to call IVRCompositor::SetGraphicsDevice(), but SetOverlayTexture takes a graphics API to identify what kind of texture is being passed in.
- SetOverlayTexture now takes a texture type. SetOverlayTextureBounds still needs to be flipped vertically for OpenGL textures.

#### IVRChaperone:

\*Added ReloadInfo() for forcing reload of bounds data from .vrchap. This is automatically handled by VRMonitor and the Compositor system when the config file is modified.

[git-p4: depot-paths = "//vr/steamvr/sdk\_release/": change = 2865502]

## v0.9.2

Jun 11, 2015

#### General:

- Added openvr\_driver.h. This is the API to write an OpenVR driver. It is still very much a work in progress. No backward compatibility is guaranteed for drivers using this interface. It may be useful for development, however. See the documentation for details.
- Added include guards around VR types in case openvr.h and openvr\_driver.h end up included in the same file.

#### IVRSystem:

- Added properties to get information about connected wireless dongles.
- Added properties to determine if the HMD display extends the desktop.
- Both render model functions have moved to the new IVRRenderModel interface.
- HandleControllerOverlayInteractionAsMouse has moved to IVROverlay.

#### IVRCompositor

- Removed all overlay related functions. Use the new IVROverlay interface.
- Added the concept of 'scene focus'. Only the application with scene focus can render the 3D scene in the compositor.
- Focus goes to application that started rendering most recently.
- The old scene focus app will receive a VREvent\_SceneFocusLost when it loses focus. When an app regains focus (because the application with focus exited) it receives a VREvent\_SceneFocusGained.



- Applications can check for scene focus with the `CanRenderScene` method. Applications without scene focus should do little to no rendering work and allow the application with focus full access to the CPU and GPU.
- `WaitGetPoses` will now block for a while for the application that doesn't have focus and then return an error. If you want to avoid the blocking, call `CanRenderScene` in your render loop.
- `Submit` will now return an error if the current application doesn't have focus.

`IVRRenderModel`:

- Added functions to enumerate the list of available render models.

`IVROverlay`:

- This is a new interface in this SDK revision. It offers:
  - Support for multiple overlays in the compositor
  - The ability to attach an overlay to a controller or other tracked device
  - Support for registering system overlays, which is what comes up when you hit the tiny black system button on the SteamVR controllers. See the `helloworldoverlay` sample for an example of how to implement one of these using Qt.
  - See the documentation for more details.

## [v0.9.1](#)

May 11, 2015

Made `openvr_api.dylib` a universal binary.

## [v0.9.0](#)

Apr 29, 2015

The API that used to be available in `steamvr.h` in the Steamworks SDK is now the OpenVR SDK. Support for the old API will be maintained indefinitely, but applications that need any of the new features will need to switch to the new SDK.

Several new interfaces were added and new methods were added to existing interfaces. See the API documentation for details.

Changes from the SteamVR interface in SteamWorks SDK 1.31 to OpenVR SDK 0.9.0 include:

- `IHmd` is now [IVRSystem](#).
- Added support for multiple tracked objects. Call [IVRSystem::GetTrackedDeviceClass](#) on numbers between 0 and `k_unMaxTrackedDeviceCount` to determine what kind of

device is in each slot. The active HMD will always be at `k_unHmdTrackedObject` (aka 0) in the list.

- Moved fetching of various values about HMDs and other tracked devices into properties. The `TrackedDeviceProperty` enum defines the available values. Call [`IVRSystem::Get<type>TrackedDeviceProperty`](#) to get the value for an object.
- Added support for providing applications with models and textures for tracked devices. This allows applications to draw the actual devices being tracked. See [`IVRSystem::LoadRenderModel`](#) and [`IVRSystem::FreeRenderModel`](#) for details.
- Clarified and changed tracking prediction. The right value to pass to [`IVRSystem::GetDeviceToAbsoluteTrackingPose`](#) is typically `( - ) + .` All three of those values are available as properties on the HMD.
- Added origin to calls that return poses. This can be Seated, Standing, or Raw. In the first two cases all returned poses will be relative to that configured origin. Raw poses are in whatever tracking system the current driver provides.
- Renamed `IVRSystem::ZeroTracker` to [`IVRSystem::ResetSeatedZeroPose`](#). This function only applies to the seated origin and will have no effect on applications that use the standing origin.
- Added angular velocity and velocity to the data returned with a tracked device's pose.
- CAPI and C# bindings for OpenVR interfaces added to headers. The accompanying json can be used to generate bindings for other languages as well.
- Added [`IVRSystem::PollNextEvent`](#) to the API. This allows applications to receive events when devices are activated, deactivated, or updated.
- Added new interface [`IVRChaperone`](#) to query chaperone hard and soft bounds. Use this to show the user where safe areas to walk are.
- [`IVRSystem::GetHiddenAreaMesh`](#). It returns the stencil mesh to use to minimize pixel rendering for the current HMD.
- Removed `GetIPD`. Use the property `Prop_UserIpdmeters_Float` instead.
- Added [`IVRCompositor`](#) interface. This interface allows an application to use the VR Compositor to perform distortion mapping and manage the HMD window.