Interaction in OpenXR

Ryan A. Pavlik, Ph.D.
Principal Software Engineer, Collabora, Ltd.
OpenXR Working Group Spec. Editor
XR Kaigi, November 2020
Agenda

- About Me
- Handle and atom types
- Modeling interaction:
  - Actions, Action Sets, and Interaction Profiles
  - Sample Walkthrough

Slides, with speaker notes and links, will be available at khronos.org
About Me: Ryan Pavlik

• Open-source VR software developer since 2009
• OpenXR working group
  - participant since the first official meeting in January 2017
  - elected specification editor in April 2019
• Principal Software Engineer at Collabora
  - Focusing on XR client projects
  - Leading our OpenXR contributions
  - Developer on Monado
OpenXR Handle Types

Instance

- ActionSet
  - Action
- Session
  - Space
  - Swapchain
Atoms

- **Instance**
  - **Path**
  - **SystemId**
  - **ActionSet**
    - **Action**
  - **Session**
    - **Space**
    - **Swapchain**
Modeling Interaction with Actions

- Focus first on what users do, not the hardware they do it with
- Important for hardware-independence.
- **Action**: A semantic (meaningful) bit of interaction
  - Types: Boolean (button), Float (analog), Vec2, Pose (tracked object), Haptic
  - e.g. “grab_object”, “teleport”, “hand_pose”
- **ActionSet**: a group of related actions for a context, environment, etc.
  - e.g. “menu”, “gameplay”, “driving”
  - One or more active at any time
Suggested Bindings and Interaction Profiles

- How you customize for hardware you’ve tested, without excluding the rest
- For each controller type you’ve tested (“interaction profile”), suggest bindings for actions
  - With as many or few action-binding pairs as you like - OK if not all actions have a suggested binding
  - Can suggest multiple bindings per action in a call: e.g. both left and right hands can “grab_object”
  - Binding is an XrPath atom representing a path string like /user/hand/right/input/select/click
- If your application is used on different hardware, the runtime may re-map your actions to the available hardware
- Set up actions, action sets, and suggested bindings once, at startup

interaction profiles added by vendor extensions XR_MSFT_hand_interaction, XR_HUAWEI_controller_interaction, and multi-vendor extensions XR_EXT_eye_gaze_interaction, XR_EXT_hp_mixed_reality_controller, XR_EXT_samsung_odyssey_controller
One last Action setup step

- Set up Actions, Action Sets, and provide suggested bindings at application start.
- Before you can use them, one more call is required later:
  - `xrAttachSessionActionSets`
  - Associates them with the session
  - Makes them immutable
  - Editor authors: tear down session, actions, action sets and re-create to modify them
- Why is action setup done all up front and immutable?
  - Good rebinding experience needs maximum information on interaction early in execution
Sample of Actions

- These are the actions from “hello_xr” - see OpenXrProgram::InitializeActions
- All in one action set, “gameplay”, due to simplicity of the app
- All except “Quit” are specified for both left and right hand as “subaction paths” because we might want to know which hand did an action
  - which hand grabbed object, etc.

<table>
<thead>
<tr>
<th>actionName</th>
<th>localizedActionName</th>
<th>actionType</th>
<th>subaction path</th>
</tr>
</thead>
<tbody>
<tr>
<td>grab_object</td>
<td>Grab Object</td>
<td>Float Input</td>
<td>/user/hand/left</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/user/hand/right</td>
</tr>
<tr>
<td>hand_pose</td>
<td>Hand Pose</td>
<td>Pose Input</td>
<td>/user/hand/left</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/user/hand/right</td>
</tr>
<tr>
<td>quit_session</td>
<td>Quit Session</td>
<td>Boolean Input</td>
<td></td>
</tr>
<tr>
<td>vibrate_hand</td>
<td>Vibrate Hand</td>
<td>Vibration Output</td>
<td>/user/hand/left</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/user/hand/right</td>
</tr>
</tbody>
</table>
xrSuggestInteractionProfileBindings 1

- Standard defines “khr/simple_controller” as a minimal subset profile
- Note here that grab_object is float, but suggested to bind to “select/click” (boolean)
  - Runtime will automatically convert boolean to a 1 or 0.

<table>
<thead>
<tr>
<th>actionName</th>
<th>actionType</th>
<th>subaction path</th>
<th>/interaction_profiles/khr/simple_controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>grab_object</td>
<td>Float Input</td>
<td>/user/hand/left</td>
<td>/user/hand/left/input/select/click</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/input/select/click</td>
</tr>
<tr>
<td>hand_pose</td>
<td>Pose Input</td>
<td>/user/hand/left</td>
<td>/user/hand/left/input/grip/pose</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/input/grip/pose</td>
</tr>
<tr>
<td>quit_session</td>
<td>Boolean Input</td>
<td>/user/hand/left</td>
<td>/user/hand/left/input/menu/click</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/input/menu/click</td>
</tr>
<tr>
<td>vibrate_hand</td>
<td>Vibration Output</td>
<td>/user/hand/left</td>
<td>/user/hand/left/output/haptic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/output/haptic</td>
</tr>
</tbody>
</table>
xrSuggestInteractionProfileBindings 2

- HTC Vive controller
- The `grab_object` action is here suggested for the “trigger/value” input
  - trigger/value instead of select/click
  - float instead of boolean: no conversion required

<table>
<thead>
<tr>
<th>actionName</th>
<th>actionType</th>
<th>subaction path</th>
<th>/interaction_profiles/htc/vive_controller</th>
</tr>
</thead>
<tbody>
<tr>
<td>grab_object</td>
<td>Float Input</td>
<td>/user/hand/left</td>
<td>/user/hand/left/input/trigger/value</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/input/trigger/value</td>
</tr>
<tr>
<td>hand_pose</td>
<td>Pose Input</td>
<td>/user/hand/left</td>
<td>/user/hand/left/input/grip/pose</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/input/grip/pose</td>
</tr>
<tr>
<td>quit_session</td>
<td>Boolean Input</td>
<td></td>
<td>/user/hand/left/input/menu/click</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>/user/hand/right/input/menu/click</td>
</tr>
<tr>
<td>vibrate_hand</td>
<td>Vibration Output</td>
<td>/user/hand/left</td>
<td>/user/hand/left/output/haptic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
<td>/user/hand/right/output/haptic</td>
</tr>
</tbody>
</table>
xrSuggestInteractionProfileBindings 3

- Oculus Touch controller
- Has a float input suitable for `grab_object` action - called “squeeze/value”
- Only left controller has a menu button, so not suggesting a binding for `quit_session` on the right hand.

<table>
<thead>
<tr>
<th>actionName</th>
<th>actionType</th>
<th>subaction path</th>
</tr>
</thead>
<tbody>
<tr>
<td>grab_object</td>
<td>Float Input</td>
<td>/user/hand/left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
</tr>
<tr>
<td>hand_pose</td>
<td>Pose Input</td>
<td>/user/hand/left</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right</td>
</tr>
<tr>
<td>quit_session</td>
<td>Boolean Input</td>
<td>/user/hand/left/input/menu/click</td>
</tr>
<tr>
<td>vibrate_hand</td>
<td>Vibration Output</td>
<td>/user/hand/left/output/haptic</td>
</tr>
<tr>
<td></td>
<td></td>
<td>/user/hand/right/output/haptic</td>
</tr>
</tbody>
</table>
Wrap-up

- Outline
  - About Me
  - Interaction in OpenXR
  - Sample Actions Walkthrough
- Resources
  - Landing page with news: [khronos.org/openxr](http://khronos.org/openxr)
  - API registry (links to the spec, ref pages, all the repos, etc) [khronos.org/registry/openxr](http://khronos.org/registry/openxr)

Thank you!

- Community
  - Source, issue trackers, etc [github.com/KhronosGroup?q=openxr](https://github.com/KhronosGroup?q=openxr)
  - Chat [khr.io/slack](https://khr.io/slack)
  - Forum [community.khronos.org/c/openxr](https://community.khronos.org/c/openxr)

- Open-Source Runtime for Linux: Monado
  - Community project founded by Collabora, not a Khronos/OpenXR WG project
  - Repos, including additional (cross-platform) OpenXR-related projects [gitlab.freedesktop.org/monado](https://gitlab.freedesktop.org/monado)
Interaction in OpenXR

Ryan A. Pavlik, Ph.D.
Principal Software Engineer, Collabora, Ltd.
OpenXR Working Group Spec. Editor