



FORGE DevCon 2017

Using the Max Interactive Engine and Forge to create powerful AR/VR Applications

Paul Kind – Autodesk, Inc.

Benjamin Slapcoff – Autodesk, Inc.

Cyrille Fauvel – Autodesk, Inc.

November, 2017

Outline

Max Interactive Overview

Max Interactive's Rendering Pipeline

- Overview
- Stereo Rendering for AR/VR

Forge AR/VR Toolkit

- What is it?
 - Forge AR/VR Toolkit Server
 - Forge AR/VR Toolkit Client
 - Get the Beta

Demos

3ds Max Interactive

Brief History

- Bit Squid = Stingray = Max Interactive
 - Bit Squid was the engine acquired by Autodesk in 2014
 - Tools Re-Written almost entirely
 - Core of the Engine was expanded on.
 - VR Added
 - Rendering Improvements
 - Other Developments
 - In this presentation, we are referring to Max Interactive

Game Engines... What exactly are they?

- What is a Game Engine?

In the early days, game engines were effectively renderers and some basic infrastructure.

- Game engines today have expanded substantially!

- Game Engines at their core are Real Time Interaction Engines and tools which help feed them content.

- Game Engines are Complicated!

- **Rendering**

- **Audio**

- **Animation**

- **UI**

- **Physics**

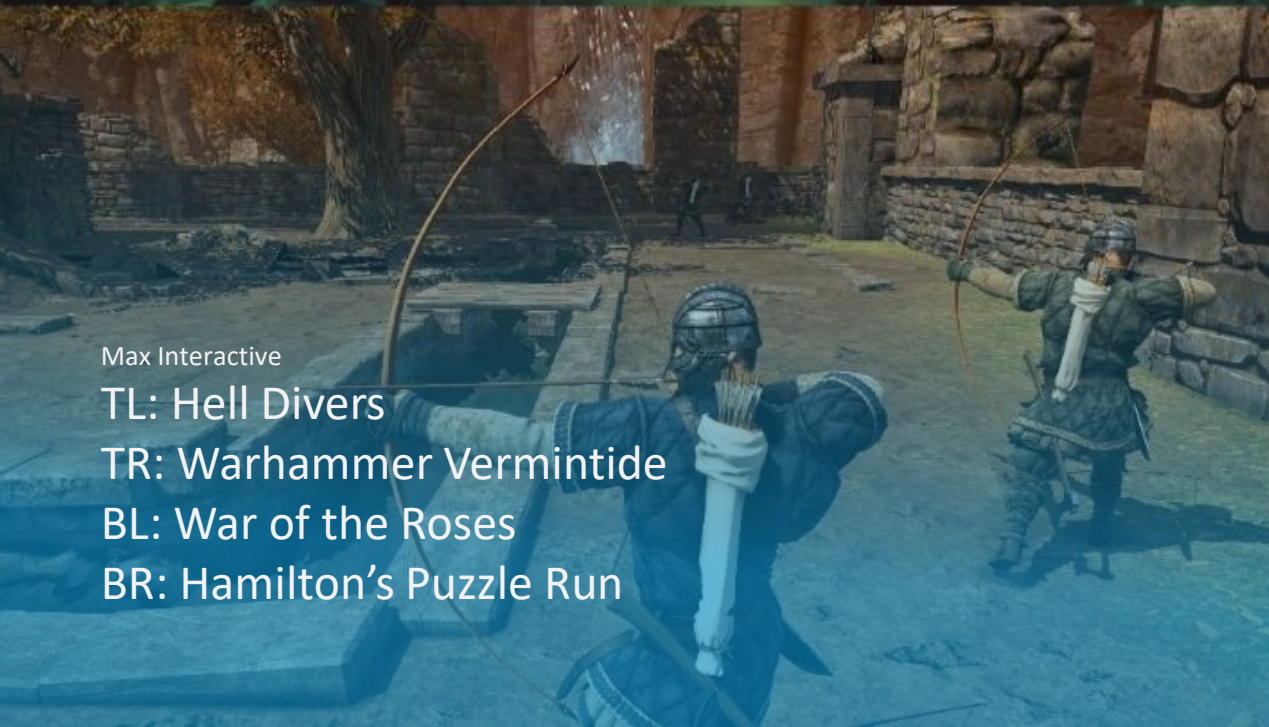
- **AI (Artificial Intelligence)**

- **Inverse Kinematics**

- **Scripting Engines**

- **Networking**

- **Much Much More!**



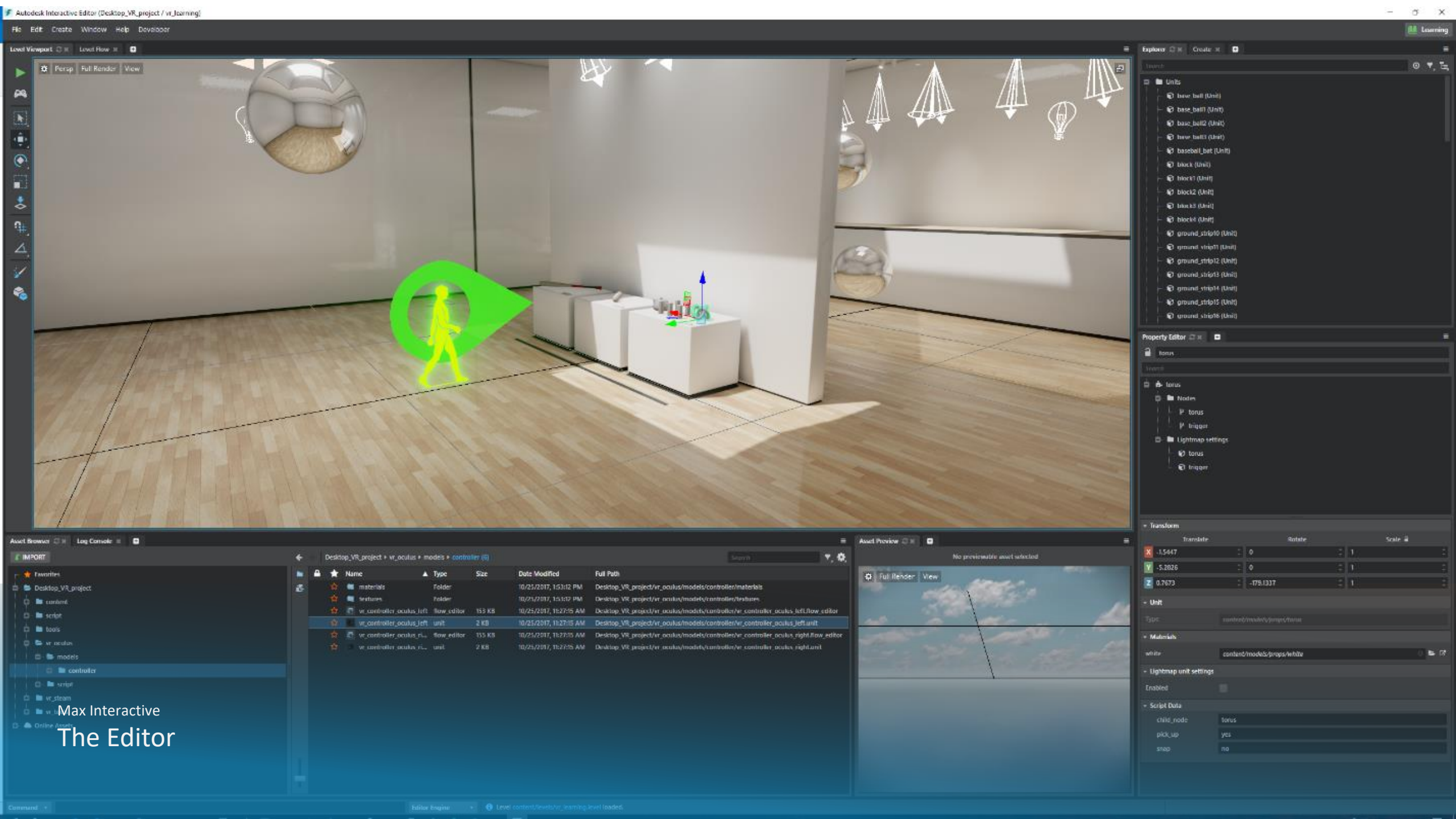
Max Interactive

TL: Hell Divers

TR: Warhammer Vermintide

BL: War of the Roses

BR: Hamilton's Puzzle Run



Max Interactive
The Editor

Use Cases

Not just for games anymore!

- Realtime Visualization
- Cinema
- Animation
- Serious Games
- Simulation
- Training
- AR/VR Increasing the Domain



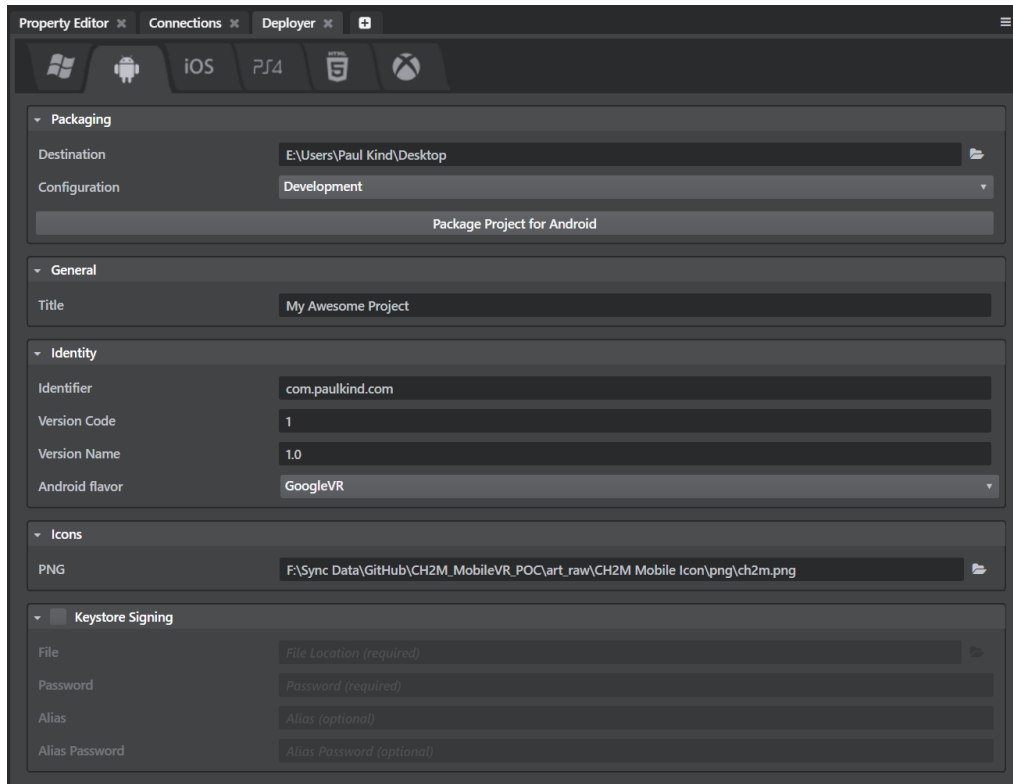
Engine Considerations

The key aspects of Max Interactive revolve around being game agnostic, ease to update, and high performance. The result is a lightweight engine with a high performance easy to learn scripting language (lua) and beginner friendly visual scripting interface called flow.

- **Lightweight**
- **Easy to Maintain**
- **Blazing Fast Performance**
- **Easy and Fast Iteration**
- **Game Agnostic**
- **Robust**

Platforms

Modern game engines are expected to support a multitude of platforms and Max Interactive is no different.



- **Windows**
- **iOS**
- **Android**
- **Oculus**
- **HTC Vive**
- **Gear VR**
- **Google Cardboard**
- **Google Daydream**
- **Hololens**
- **Windows Mixed Reality**
- **ARKit**
- **WebGL**
- **More to come.**

Data Driven Renderer

Produce Dramatically Different Content by simply changing the data!



Lua Scripting and C-API: Happy Together

Lua

Lightweight, beginner friendly, high performance, super awesome!

```
-- Returns the viewport position from the Editor, if available for example when
-- run as an editor Test Level.
function Appkit.get_editor_view_position()
    if Appkit.boxed_editor_view_position then return Appkit.boxed_editor_view_position:unbox() end
    return nil
end
```

C-API

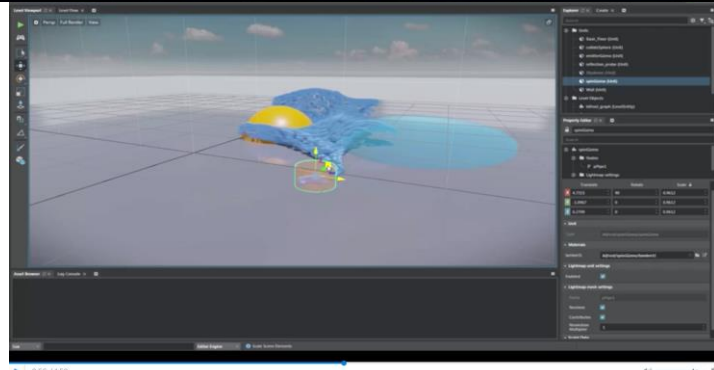
Super powerful, industry standard, allows for expansion.

```
struct UnitCApi
{
    ConstVector3Ptr → (*local_position) (UnitRef, unsigned index);
    CApiQuaternion (*local_rotation) (UnitRef, unsigned index);
    ConstVector3Ptr → (*local_scale) (UnitRef, unsigned index);
    ConstLocalTransformPtr (*local_pose) (UnitRef, unsigned index);
}
```

Neither Lua nor C have code modifications!

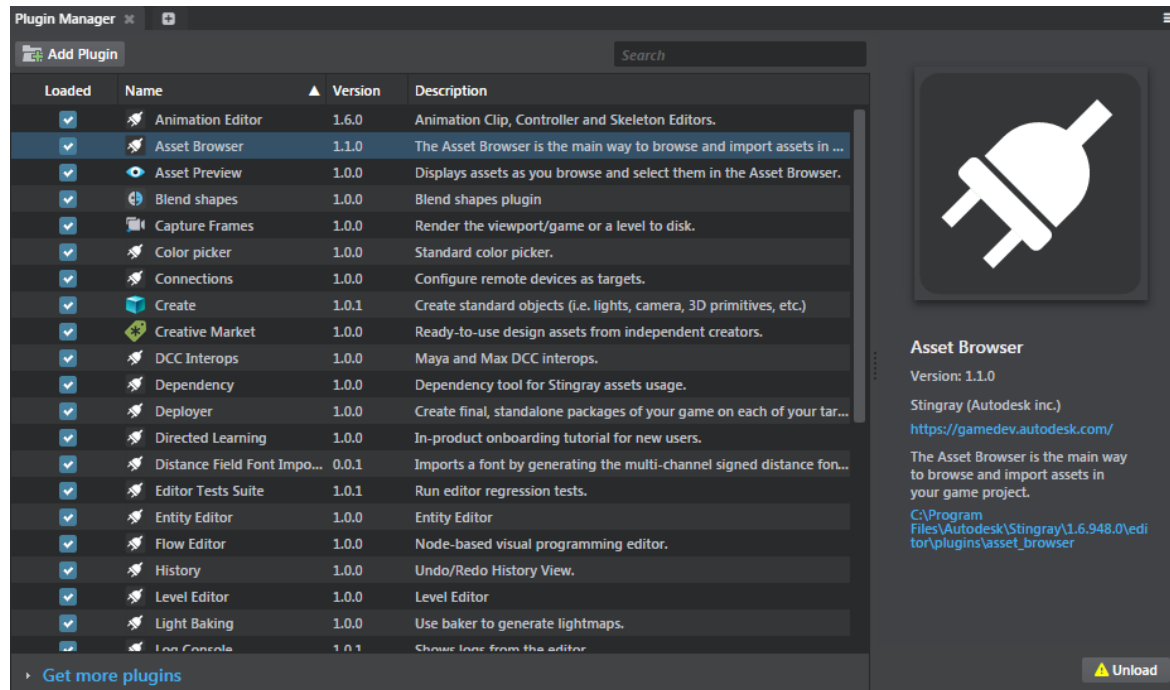
Plugin System

With the C-API you can extend both the editor and the runtime!



Plugin Manager

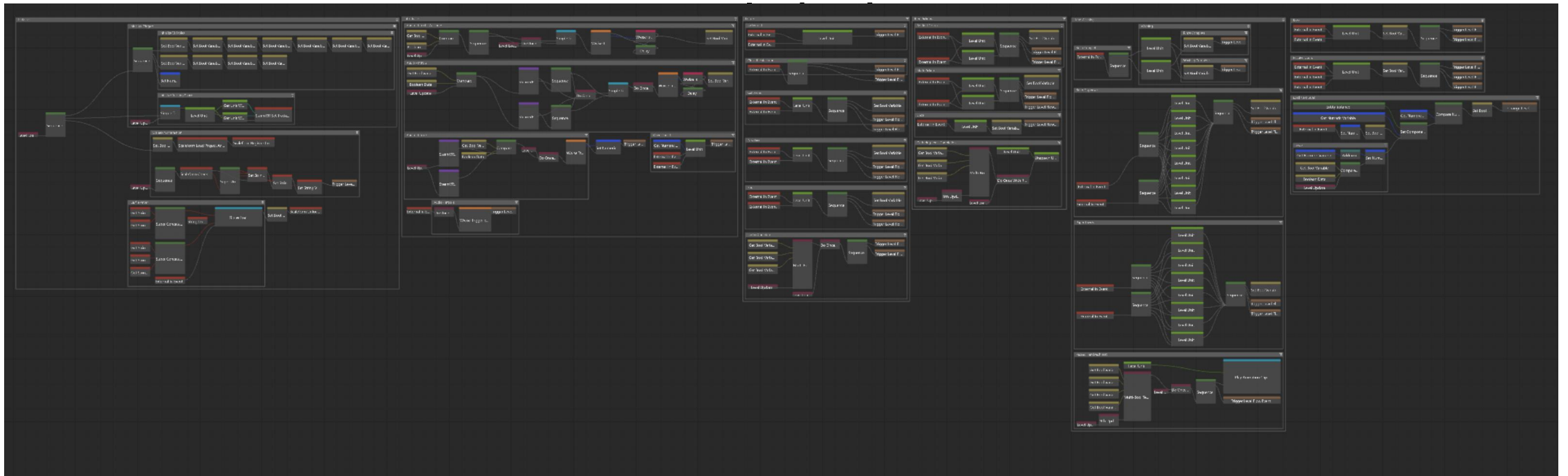
Adding plug-ins to a project is as easy as checking a box.



The SDK for writing plugins is publicly available on GitHub!
github.com/AutodeskGames/stingray-plugin

Go with the Flow

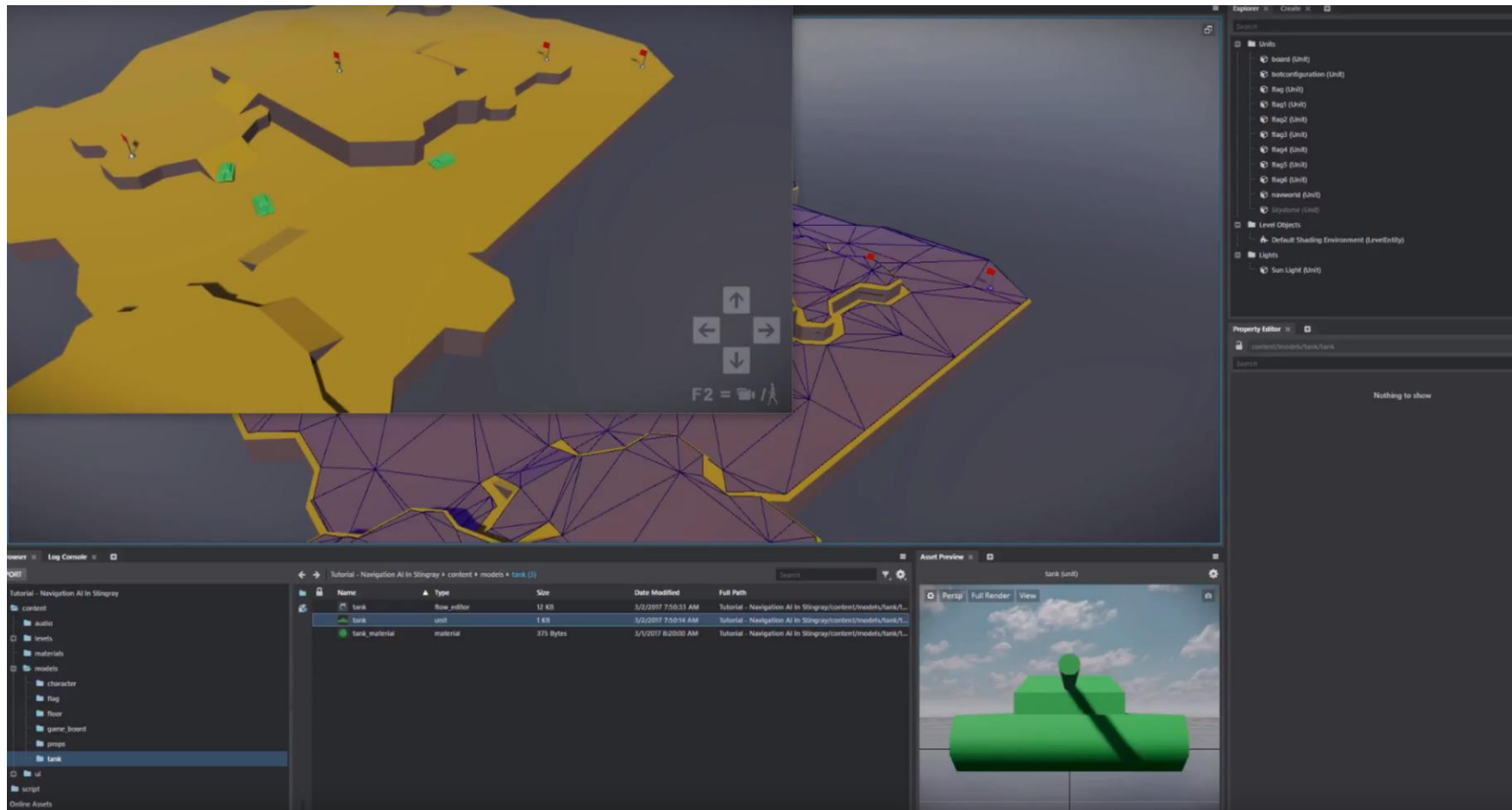
Not a coder? Want a tool to make quick work of complicated tasks? Want to enable “non-technical” users on your team to do more? Flow is the



Easily add new and powerful nodes to your project by crafting your own with Lua!

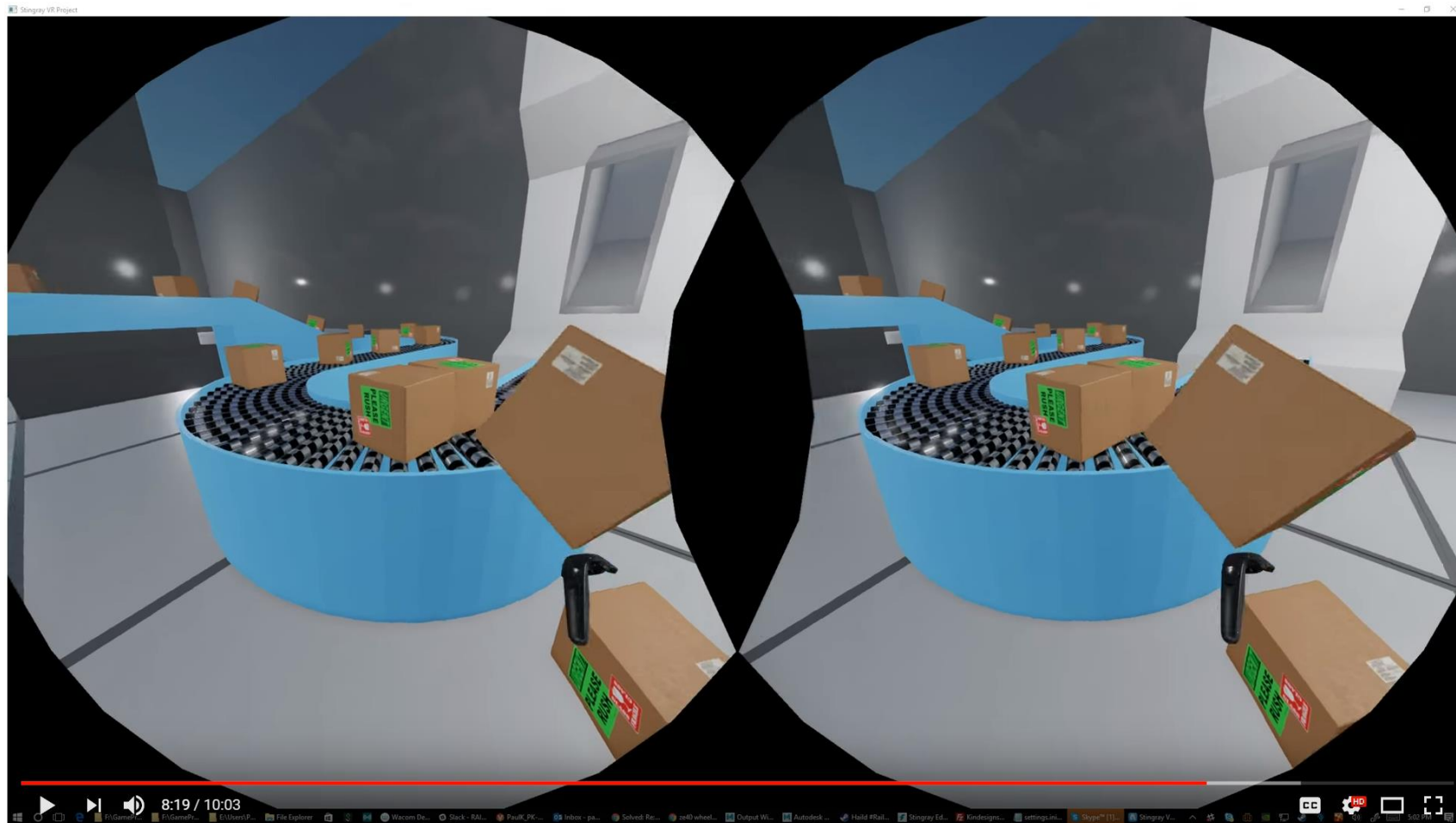
Autodesk Navigation AI

Compute and follows paths, obstacle avoidance, dynamic obstacles and more.



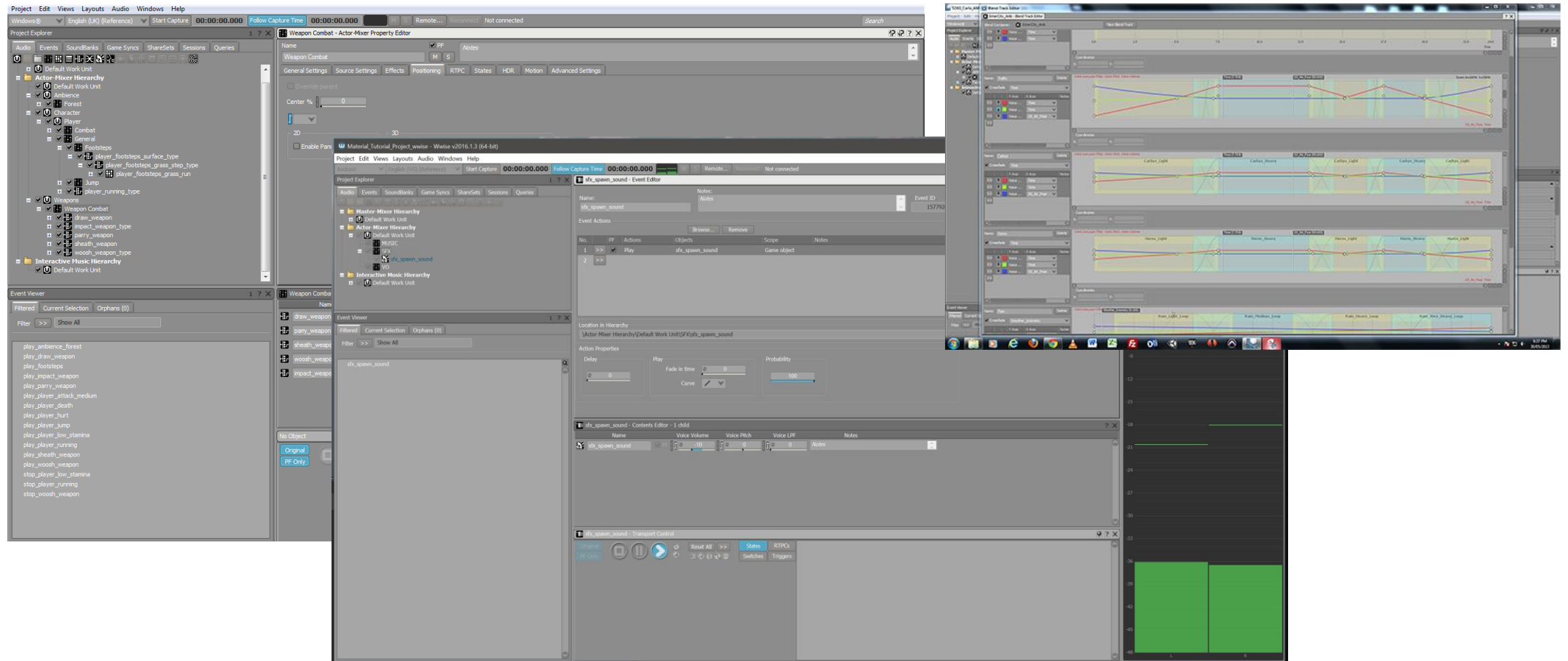
Physics using PhysX

Nvidia's PhysX SDK brings your simulations to life.



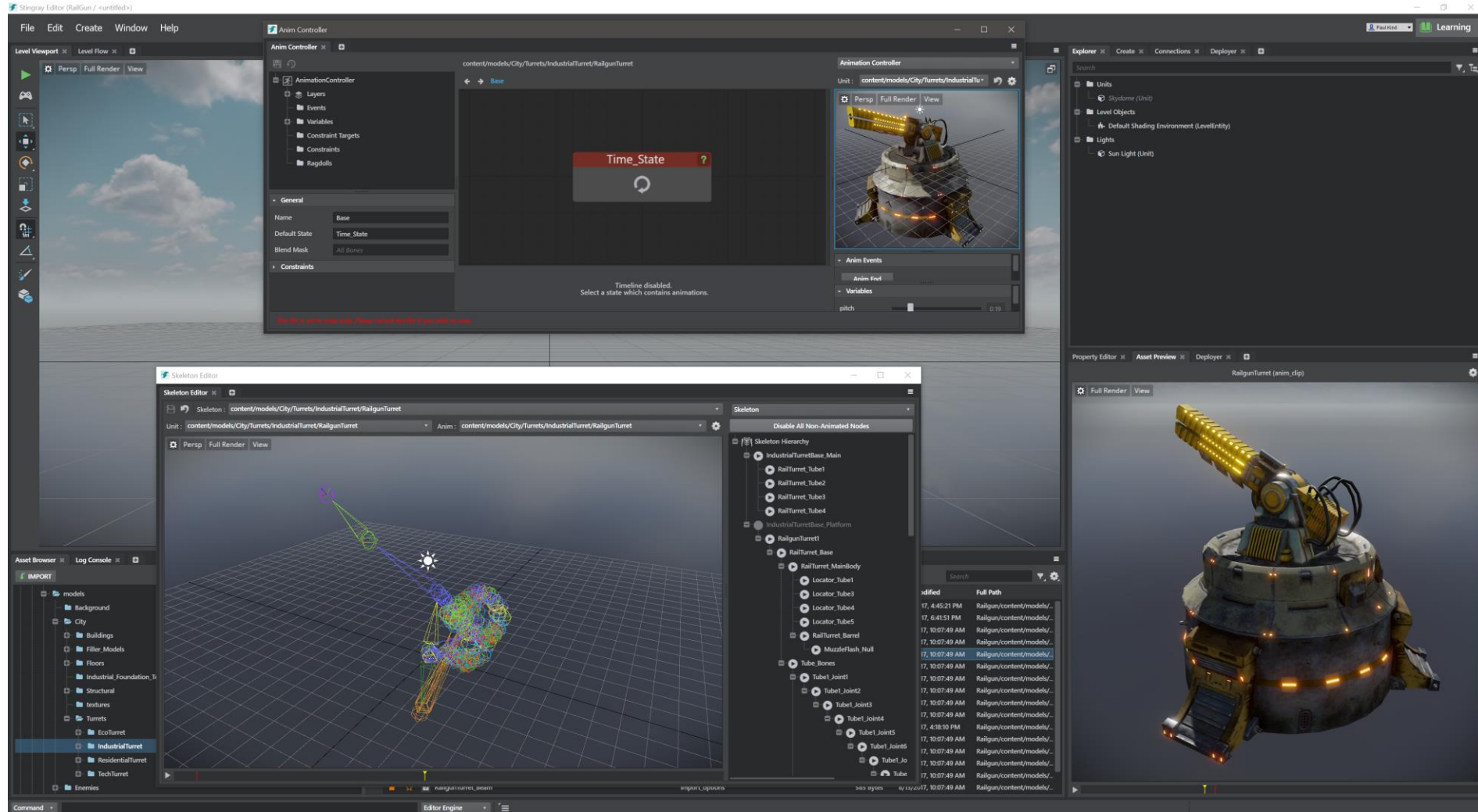
Audiokinetic Wwise

Industry leading sound engine for 3D engines.



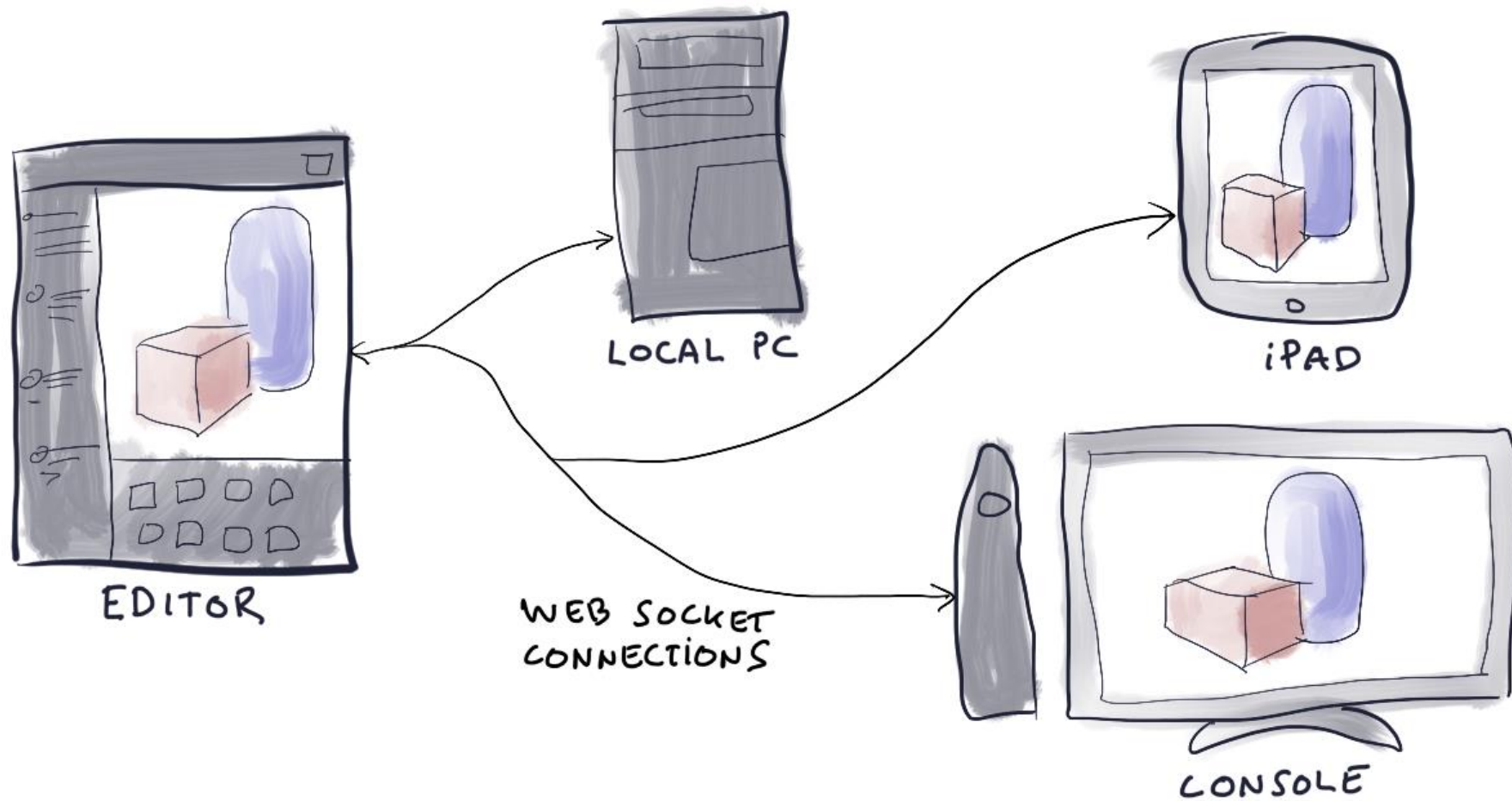
Animation

Human IK, State Machine, Story Tools and Much More!



Live Link & Hot Reload

Rapid Iteration Made Easy



Conclusion

Max Interactive is Awesome!

- Easy To Use and Learn!
- Powerful and Flexible!
- Is Loaded With Modern Tools!
- Looks Great!
- Comfortable to use, familiar UI!
- High Performance!
- Scriptable, Codeable and Visual Scripting!
- Deploys Anywhere!
- Fits Beautifully into Your Ecosystem!
- Is VR/AR Ready!
- Is part of your 3DS max subscription!

Max Interactive: Rendering Pipeline

Flexible Data-Driven Renderer

Shaders, resource creation, resource manipulation and flow of rendering pipeline entirely defined in ***data***

Data is expressed in SJSON

- Hot-reloadable for quick iteration times
- Allows for fast experimentation and debugging

Max Interactive's "Render Config"

- Ties all rendering sub-systems
- Dictates the order of operations for a frame
- Defines quality settings, device capabilities and default shader libraries to load
- Three key ingredients
 - Resource sets – memory allocations / deallocations
 - Resource generators – resource updating
 - Layer configurations – frame scheduling

er.render_config *

```
// G-buffer targets
{ name="gbuffer0" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R8G8B8A8" }
{ type="static_branch" platforms=["win", "uwp", "ps4", "xb1"]
  pass = [
    // Will move to smarter g-buffer encoding of normals but for now we'll just keep them in half precision
    { name="gbuffer1" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R16" }
  ]
  fail = [
    { name="gbuffer1" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R8" }
  ]
}
{ type="static_branch" platforms=["win", "uwp", "ps4", "xb1", "web", "linux", "macosx"]
  pass = [
    // Need hi precision motion vectors on high-end platforms
    { name="gbuffer2" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R16" }
  ]
  fail = [
    { name="gbuffer2" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R8" }
  ]
}
{ name="gbuffer3" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R8G8B8A8" }

// Render target containing linear depth, populated by the linearize_depth pass
{ type="static_branch" platforms=["win", "uwp", "ps4", "xb1"]
  pass = [
    { name="linear_depth" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R16" }
    { name="hiz_depth" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 mip_levels=1 }
  ]
  fail = [
    { name="linear_depth" type="render_target" depends_on="output_target" w_scale=1 h_scale=1 format="R8" }
  ]
}

// Shadow map for cascaded shadow mapping from sun light
{ type="static_branch" render_settings={ sun_shadows = true }
  pass = [
    { name="sun_shadow_map" type="render_target" size_from_render_setting="sun_shadow_map_size" format="R16" }
  ]
}
{ type="static_branch" render_settings={ deferred_local_lights_cast_shadows = true }
  pass = [
```


Challenges in VR

High refresh rates: 90Hz (11.11ms per frame)

Frame buffer: 2160x1200

Super sampled to reduce aliasing artifacts

- Off-screen buffer scaled by 1.5x in each dimension (3240x1800)

Shaded visible pixels

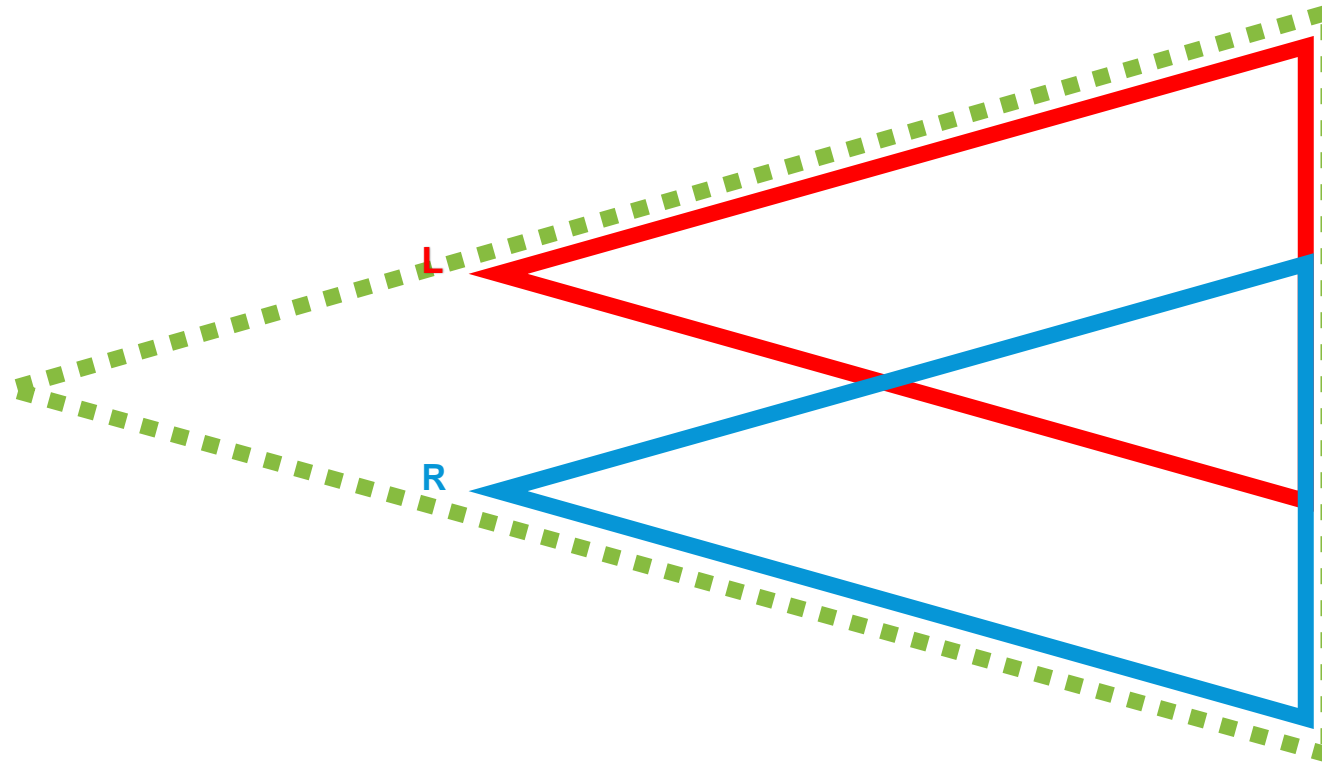
Resolution	Refresh Rate (Hz)	MPixels per Second
720p	60	55
1080p	60	124
2160p	60	489
VR (3240x1800)	90	525

Max Interactive VR: Optimizations

- Frustums are a big deal...

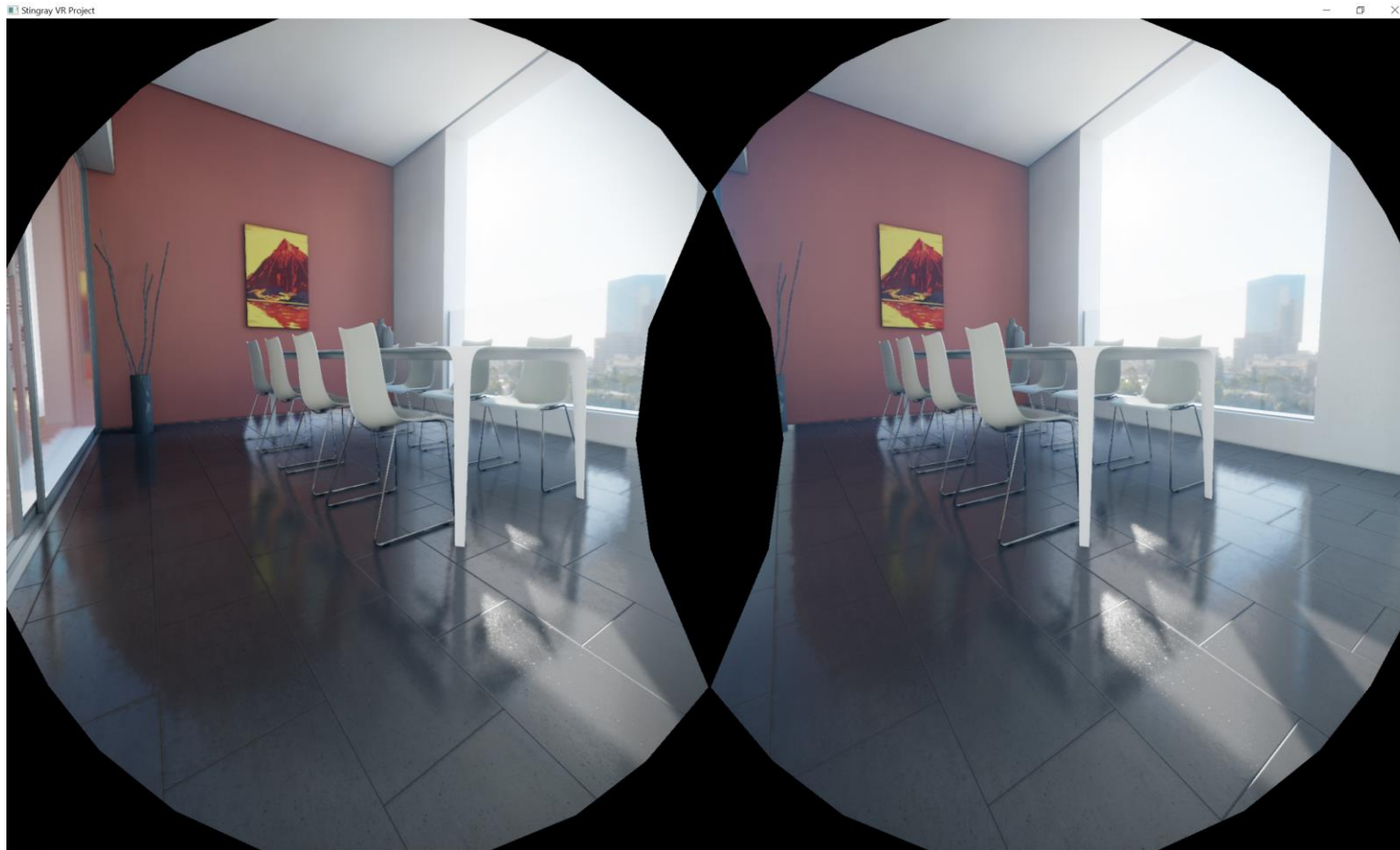
Max Interactive VR: Optimizations

- Compute single compound frustum to incorporate both eyes



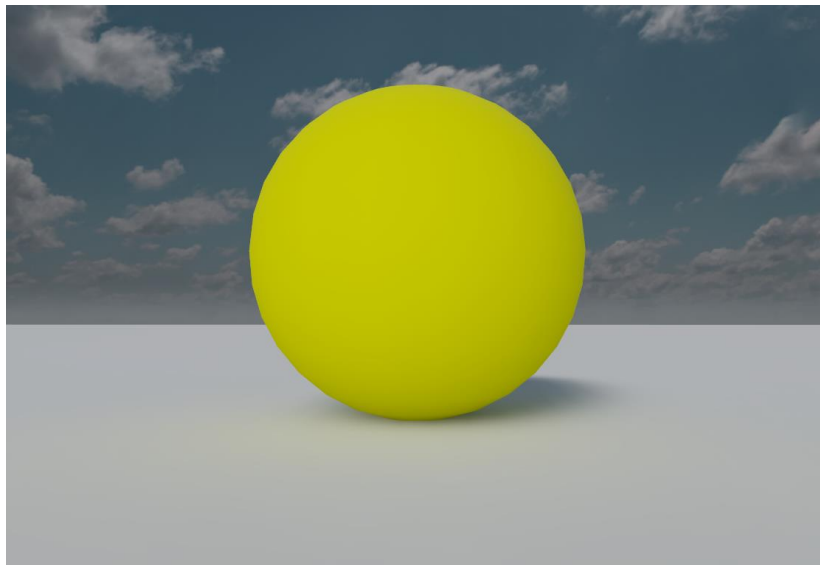
Max Interactive VR: Optimizations

- Bail early on non-visible (depth and stencil compare)

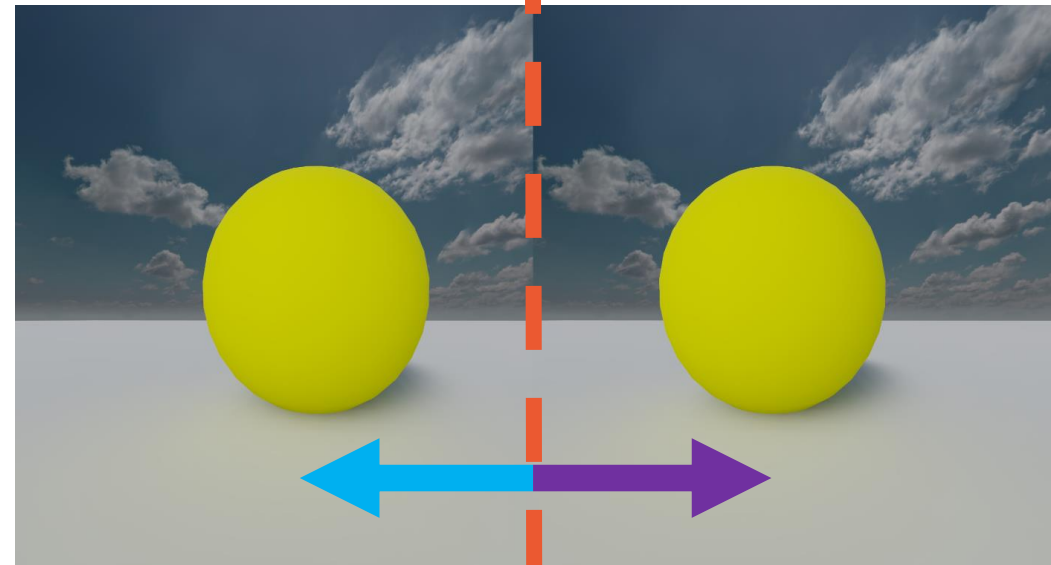



Max Interactive VR: Optimizations

- GPU geometry instancing to render in stereo



Geometry
instance count
doubled



Even instances clip
position scaled and
shifted left

Odd instances clip
position scaled and
shifted right

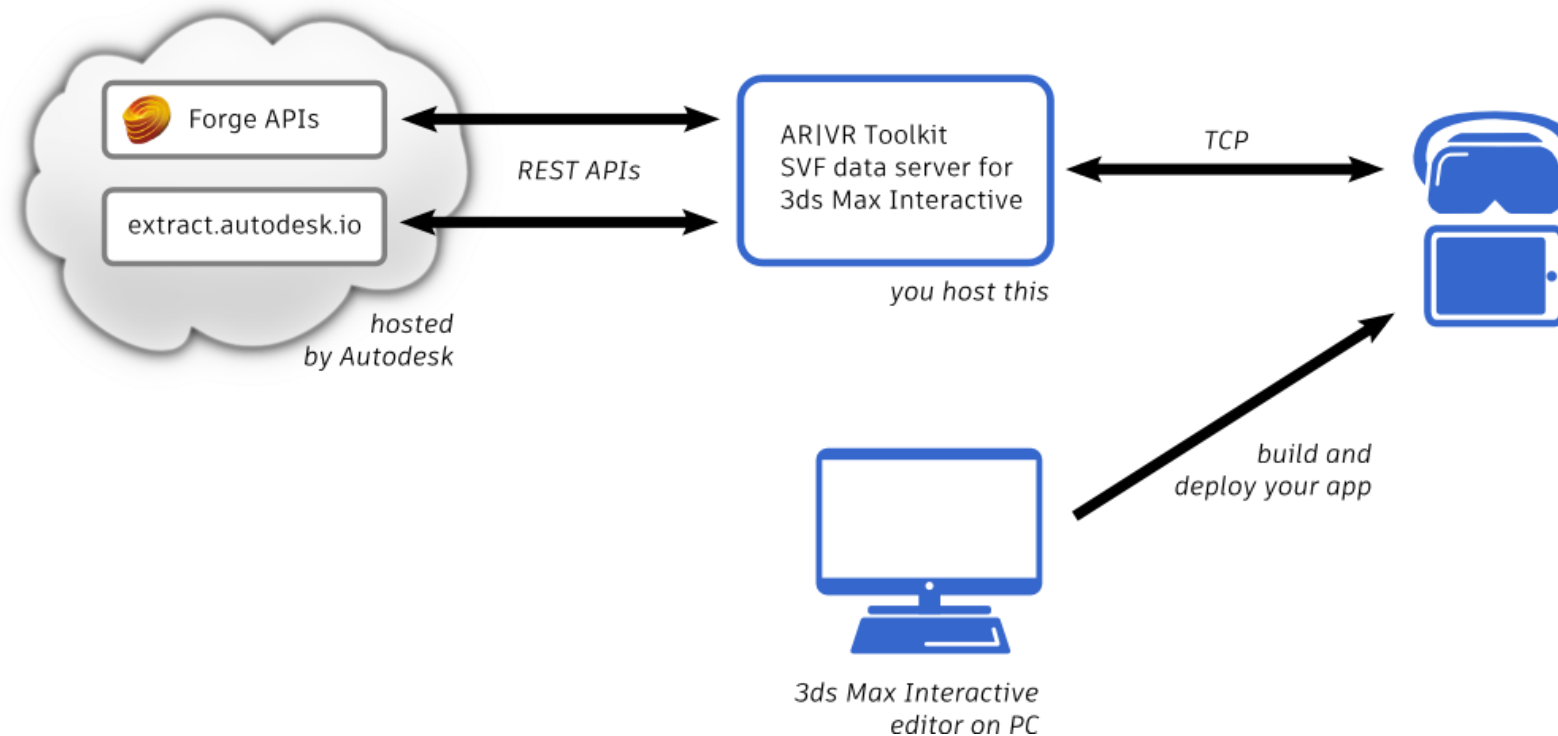
Dynamic clip plane

Augmented Reality (AR) and Low-End Devices



Forge AR/VR Toolkit with Max Interactive

- Forge integration with Max Interactive
- Easily view all your Forge data in AR and VR (and any other Max Interactive platform)
- Max Interactive “Client” apps connect to a small Server app that interfaces with Forge



Forge AR/VR Toolkit: Server

- Standalone application that interfaces with Forge
- `server_config.json` file read in at boot for initial setup.

```
{  
  "version":1,  
  "client_id":<client-id>,  
  "secret":<client-secret>,  
  "data_folder":"C:/TEMP/data/",  
  "cache_folder":"C:/TEMP/cache/",  
  "temp_folder":"C:/TEMP/temp/"  
}
```

Forge Developer Client ID and Secret

Data Prep Folders

Uploading your Design Data to Forge

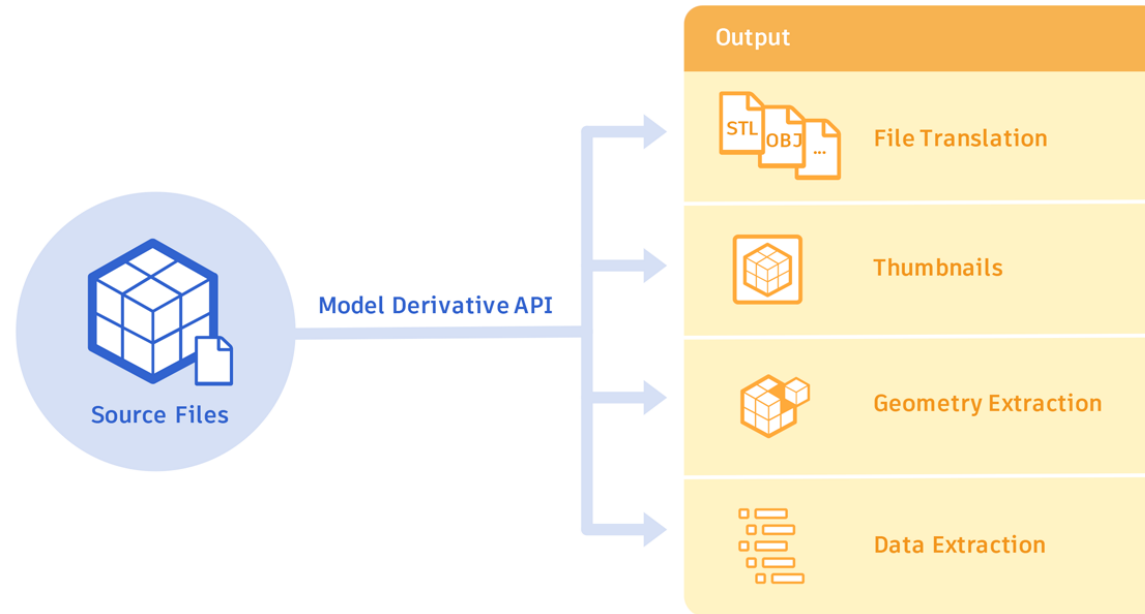
- All of your design data goes in the **data** folder, to be uploaded to Forge
- Can be any of the file formats supported by the Forge Model Derivative service

3DS, 3DM, ASM, CATPART, CATPRODUCT, CGR, COLLABORATION, DAE, DGN, DLV3, DWF, DWFX, DWG, DWT, DXF, EXP, F2D, F3D, FBX, G, GBXML, IAM, IDW, IFC, IFW, IGE, IGES, IGS, IPT, JT, MFR, MODEL, NEU, NWC, NWD, OBJ, PDF, PRT, PRT, RCP, RVT, SAB, SAT, SESSION, SKP, SLDASM, SLDPRT, SMB, SMT, STE, STEP, STL, STLA, STLB, STP, STPZ, X_B, X_T, XAS, XPR, WIRE, ZI

- Obtain an access token from the Forge OAuth API
- Create buckets for the resources with the Forge Object Storage Service (OSS)
- Upload the files and receive a URN corresponding to the uploaded object

Using the Model Derivative Service

- Use the Model Derivative API to convert our resources to **SVF** format



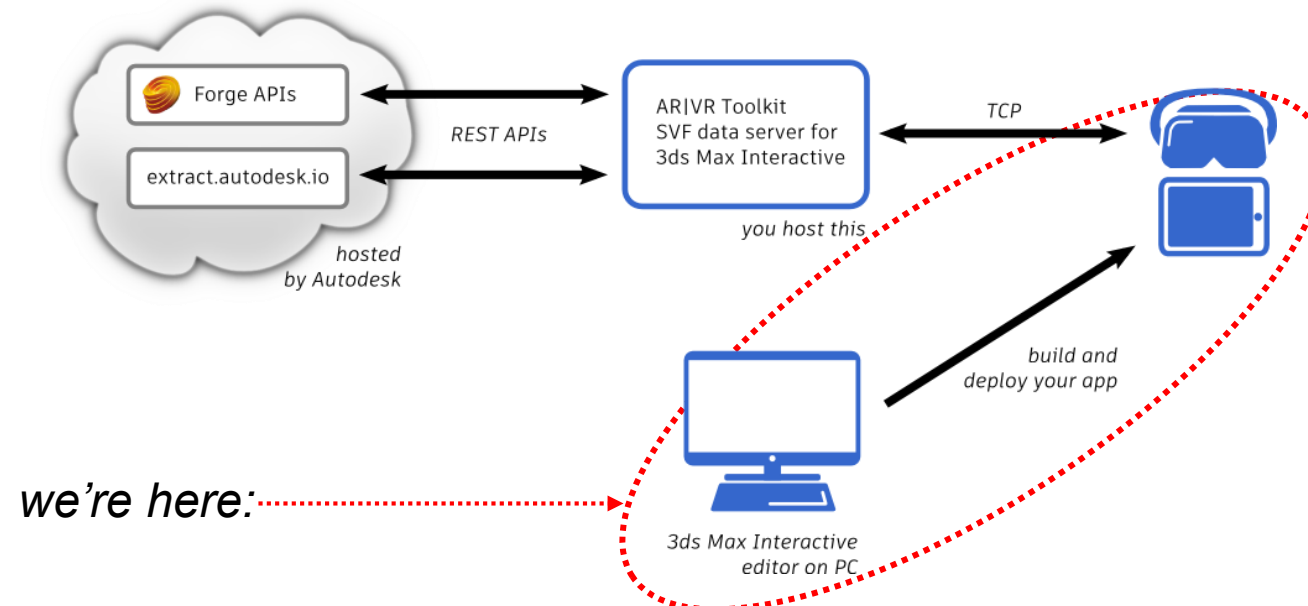
- Use the Extract API (*extract.autodesk.io*) to convert the SVF file into a “viewable” format, known as bubbles.
- Download these bubbles in the **temp** folder defined in the *server_config.json*

Last Step: Ready the bubbles for Max Interactive

- Two libraries are used to extract relevant information for Max Interactive clients
 - LMVTK (Large Model Viewer Toolkit)
 - PropertyDB Reader
- The Goal: Convert SVF model into our own “Scene Model” representation that is shared between the client and server.
- Use assortment of different readers available in LMVTK to extract necessary data
 - FragmentReader, InstanceTreeNodeReader, GeometryReader, MaterialReader, etc.
- Serialize the Scene Model into a “.extract” file in our **cache** folder (specified in the *server_config.json*)
- We’re done! Open up TCP socket and wait for clients

Forge AR/VR Toolkit: Client

- Use Max Interactive's engine plugin system to create a Forge plugin to act as a client connecting to our Server.
 - **ExtractIO** plugin
- Engine plugins are C modules that interface with Max Interactive through its Plugin API system
- Plugins can expose Lua APIs that can be used in project scripts



Forge Plugin Lua Bindings

- First step: request connection to the server

ExtractIO.server_connect(<ip>, <port>)

- Next step: get available resources from server

ExtractIO.server_resource_names()

- Last step to having your Forge data in your app:

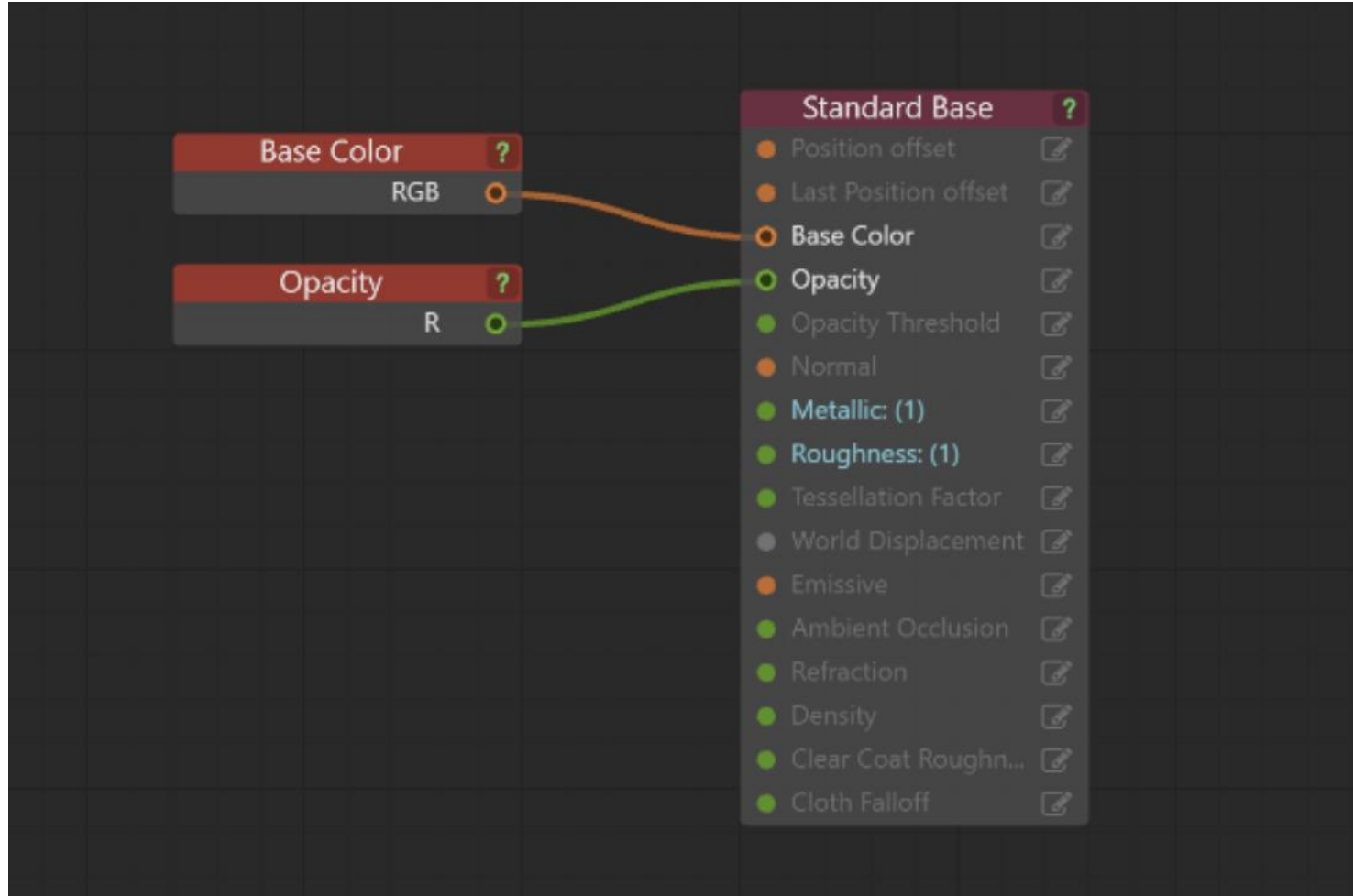
ExtractIO.load_server_resource(<resource_id>)

Max Interactive Plugin API

- **SceneGraphAPI** to create nodes for each element in the SVF's instance tree
- **MeshAPI / RenderBufferAPI** to create the meshes for each fragment of the SVF
- **MaterialAPI** to fill in material properties in a premade shadergraph

Max Interactive Plugin API

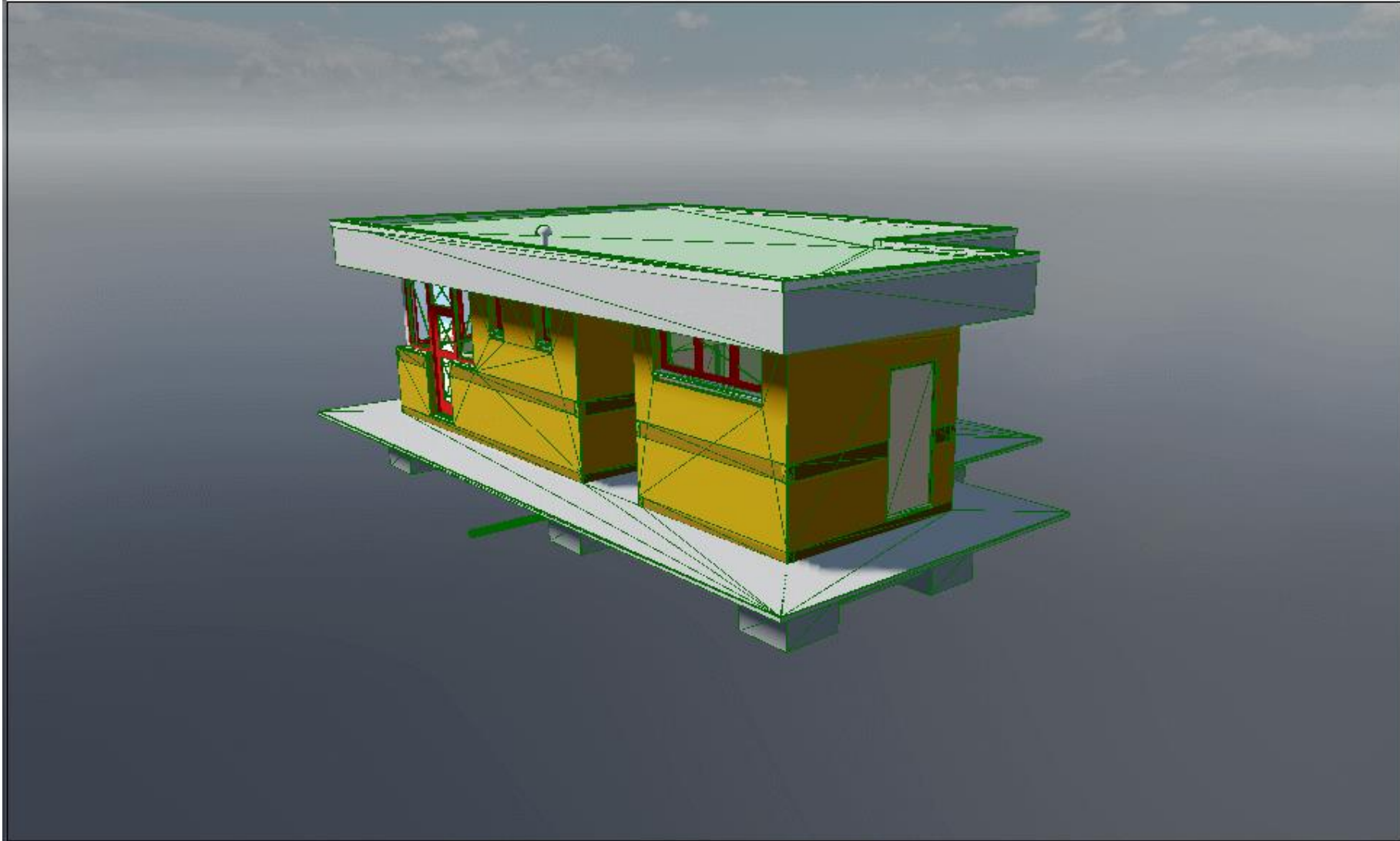
Example of a material graph with two parameters exposed, to be filled in with the **MaterialAPI**



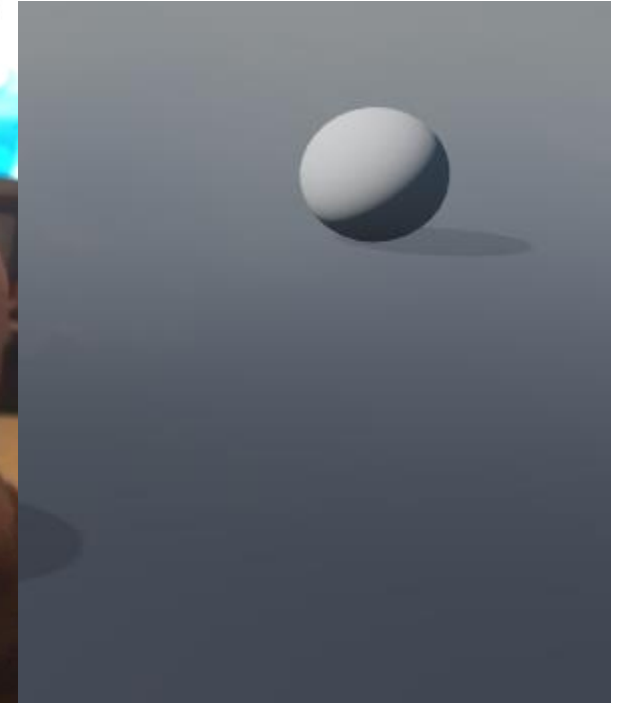
Max Interactive Plugin API

- **SceneGraphAPI** to create nodes for each element in the SVF's instance tree
- **MeshAPI / RenderBufferAPI** to create the meshes for each fragment of the SVF
- **MaterialAPI** to fill in material properties in a premade shadergraph
- **RuntimePhysicsCooking API** to create physics representation for the Forge models

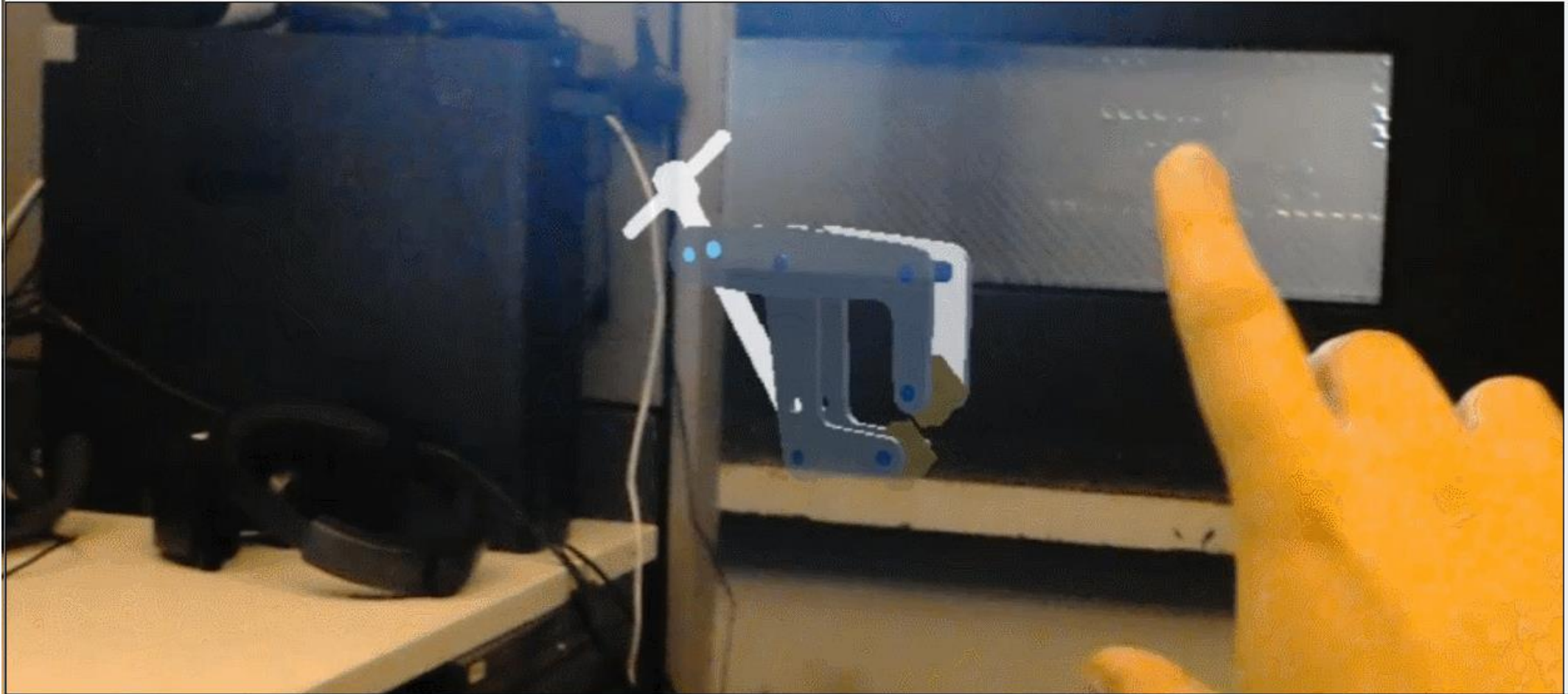
Max Interactive Plugin API



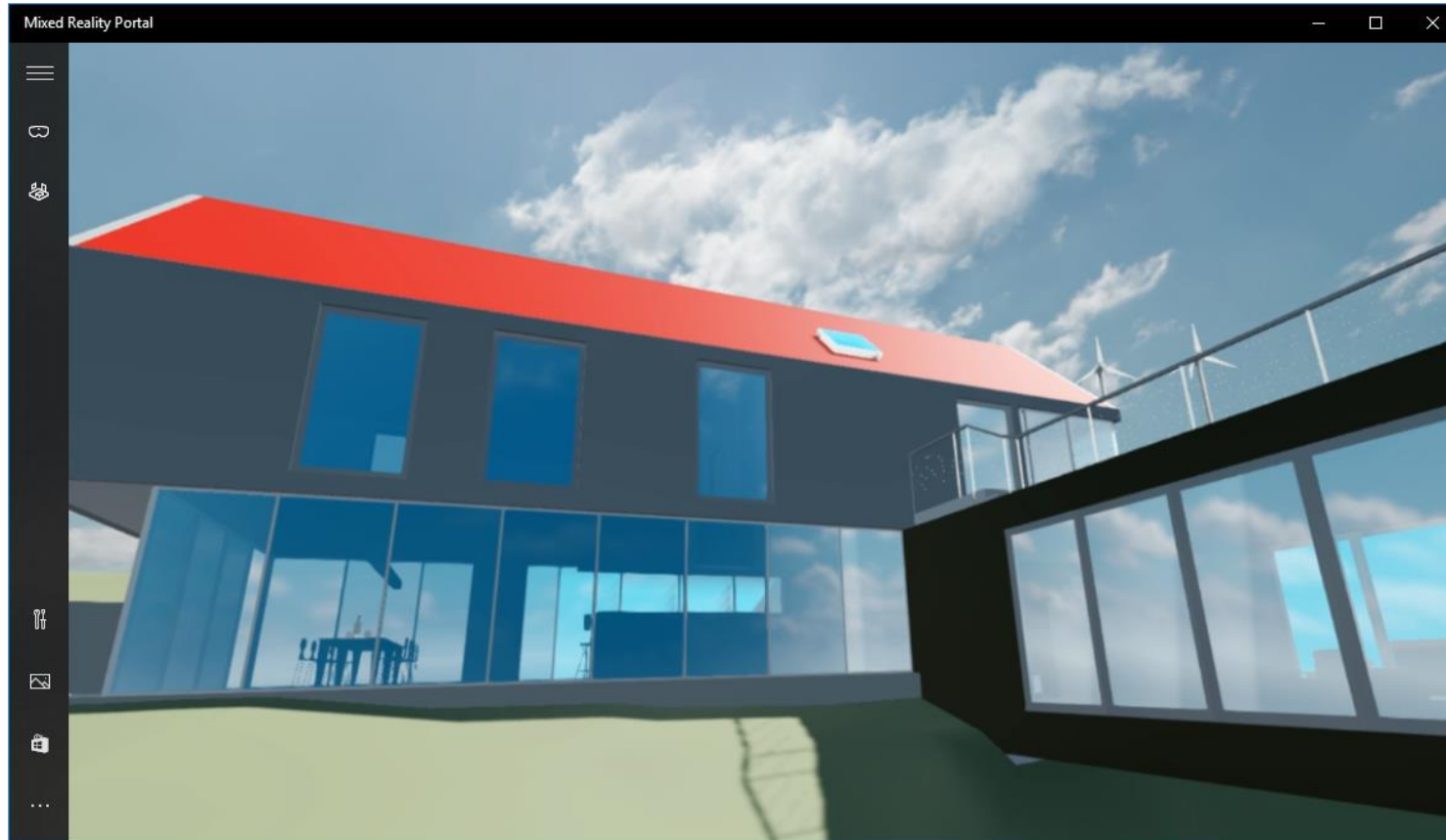
Forge on iOS ARKit



Forge on HoloLens



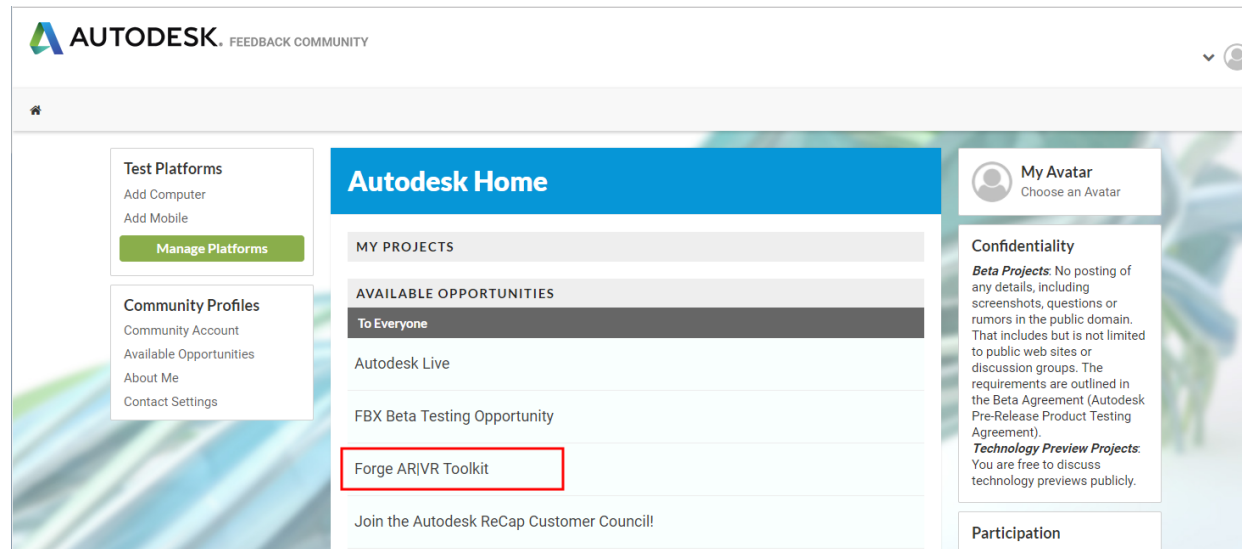
Forge with MR/VR



Forge AR/VR Toolkit with Max Interactive BETA

Visit <http://beta.autodesk.com>, sign in with your Autodesk account, and look for an opportunity to join the **Forge AR|VR Toolkit** tech preview.

Also, keep your eyes on <https://forge.autodesk.com/blog> -- as we work on the toolkit we'll be posting all our news there.



Acknowledgments

James Park – Sr. SW Engineer, DCP-IXG

Olivier Dionne – Sr. SW Development Manager, DCP-IXG

Raman Grewal – Software Engineer, DCP-IXG

Anis Benyoub – (former) Software Engineer, DCP-IXG

Robb Surridge – Principal Learning Content Developer, DCP-IXG

Matthew Carpenter – QA Manager, DCP-IXG

Daniel Cotnoir – Director, Product Development, DCP-IXG

Vincent Lu – Sr. Software developer in Test, DCP-IXG

DEMO

Q & A