



Hi everyone, and welcome to Pixar's 2023 Birds of a Feather on USD, Hydra and OpenSubdiv.

My name is Florian, and I am a Director of Engineering in Pixar's Software R&D department.



Agenda

- 2:05pm **Introduction**
- 2:10pm **Alliance for OpenUSD**
- 2:15pm **USD Update**
- 2:25pm **Hydra Update**
- 2:30pm **OpenSubdiv Update**
- 2:35pm **Partner Presenters**
- 3:20pm **Panel + Q&A**

© 2023 Disney/Pixar. All rights reserved.

Here is what we have on the agenda for today:

- In a moment, **Steve May** is going to talk about the **Alliance for OpenUSD**
- Then, we are going to get an **update** on USD, Hydra and OpenSubdiv respectively
- Up next we have our slate of partners presenters share their awesome work around these technologies
- And lastly, we are going to open the floor, answer your questions, and discuss



Should out to all of our Pixarians who have been hard at work on these awesome open-source technologies.

And of course, thanks to all of you: Thank you for showing up today... for contributing... and for engaging with us and helping us build a community around these technologies.

And without further ado, I'll hand it off to Steve May, who will talk to us about the Alliance for OpenUSD!



- suspect a lot of you have heard about AOUSD by this point
- wondering, I get it but what does it mean for me? why is it important?

AOUSD

- The **Alliance for OpenUSD** is an open, non-profit organization dedicated to fostering the standardization, development, evolution, and growth of OpenUSD
- Founding companies: Pixar, Adobe, Apple, Autodesk, NVIDIA

© 2023 Disney/Pixar. All rights reserved.

- last week, announced the launch of the Alliance for OpenUSD — AOUSD, for short
- The Alliance for OpenUSD is an open, non-profit organization dedicated to fostering the standardization, development, evolution, and growth of OpenUSD
- 5 founding members: Pixar, Adobe, Apple, Autodesk, NVIDIA
- all have been big contributors to USD and/or proponents of USD adoption
- we have come together to create this organization to serve you and the industry at large

AOUSD

- Hosted by Linux Foundation as a Joint Development Foundation project
- Incoming members: Cesium, Epic, Foundry, Hexagon, IKEA, SideFX, Unity
- Liaison relationships with: ASWF, Khronos

© 2023 Disney/Pixar. All rights reserved.

- a few other details:

- AOUSD is hosted by the Linux Foundation, as a JDF project
- JDF provides a specific structure to develop and create a standard that will eventually go to some larger standards body
- note: ASWF was not chartered to develop standards so that's why we're taking this path
- BUT we want to have very close ties with ASWF and both being under the LF is one of the reasons we chose this path
- new members are encouraged to join
- this may be out of date, but as of launch, these co's have joined — Cesium, Epic, Foundry, Hexagon, IKEA, SideFX, Unity
- want to thank all of those companies for being early supporters
- and we are pursuing liaison relationships with ASWF, Khronos and possibly others so that we can work with them to find solutions that work best for community



- our goal w/open source'ing USD was to make it a standard in our industry - that's OpenUSD - the code, the implementation that Pixar created and maintains
- extremely happy with how USD has been adopted by studios and companies and many of you in the room have been instrumental to its success
- USD needs to take the next step
- to achieve the full potential of OpenUSD we need to formalize as a standard
- want people to have confidence that it will work across many platforms, on many kinds of devices, reliably, for many years into the future
- the goals of the alliance include 2 things
 1. create a formal specification of the functionality already in OpenUSD
 2. have open discussions and create new working groups about how we want to expand the capabilities of USD going forward
- It's a huge milestone for USD



Hi everybody! On behalf of the USD team, I'd like to welcome you today as we talk about just a little bit of the intense amount of activity that's been happening in the USD ecosystem!



- Let's start with a quick overview of what we'll be going over today.



- We'll start out with a checkin on the projects we had on our roadmap from last year
- Then have a lightning review of just a *few* of the many other recent developments in OpenUSD
- We'll follow that with an update on many of the positive changes arising from the "USD Initiative" that we announced at last year's BoF
- Including not only progress, but several of the initiatives we expect to move forward over the coming year
- We'll end with a more informative look at what to expect from OpenExec, which will be a major evolution for USD



- So let's glide into it, and review our past-year roadmap progress, which is all about user-level features!

A presentation slide titled "Roadmap Progress" with a background image of Buzz Lightyear. The slide lists several development milestones. In the top left corner, there is a small icon of a lamp and a globe. In the bottom left corner, there is a small logo for "PIXAR OPENUSD". In the bottom right corner, there is a copyright notice: "© 2023 Disney/Pixar. All rights reserved."

Roadmap Progress

- Schema Versioning: **Complete in 23.05**
- Pattern-Based Collections: **Complete in dev branch / 23.11**
- Stage Variable Expressions: **Complete in dev branch / 23.11**
- Animation Splines: **Proposal just published on OpenUSD-proposals**
- Namespace Editing: **Proposal coming imminently to OpenUSD-proposals**
- Validation Framework: **In R&D**

- Schema versioning was completed in 23.05, and we think we will have some first core uses of it in evolving the Cylinder and Capsule schemas in collaboration with the UsdPhysics community
- Pattern-based Collections are feature-complete in the dev branch.
- Stage Variable substitutions are usable in 23.08, and support for the full range of expressions described in the proposal are landing imminently in the dev branch
- We've been in R&D and collaborative discussions with Autodesk and others partners for awhile now on native animation spline support in USD, and have just published our proposal on OpenUSD-proposals. Be sure to check it out!
- Our long-awaited namespace-editing project kicked off recently, and our proposal will be appearing shortly on OpenUSD-proposals
- Finally, we are in active R&D for adding a core validation framework to USD that is both performant and extensible, so you can not only run it on full scenes, but customize it to site or department needs.



- Now in our lightning round, we'll mention some of the *other* tasty developments in OpenUSD over the past year!



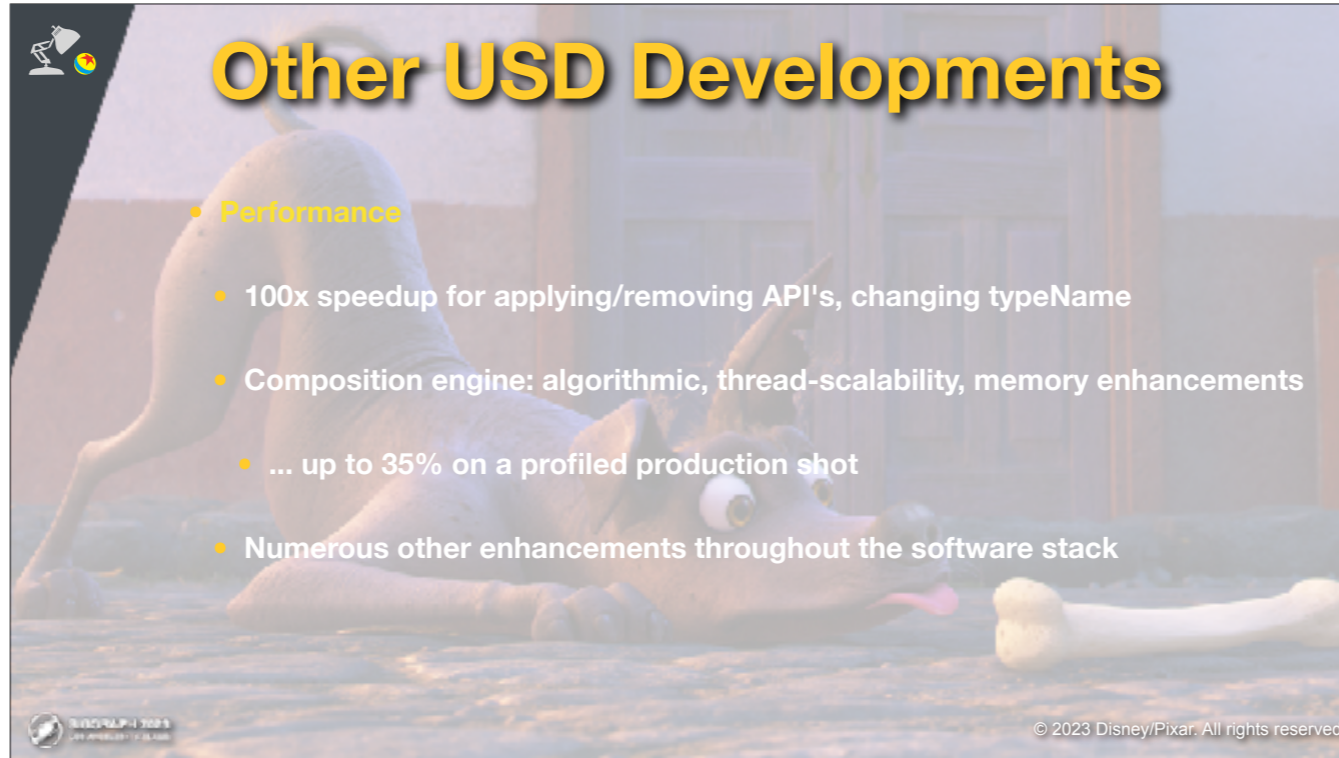
In the documentation and collaboration space,

- Thanks to an Apple PR, we have a central listing of USD-infused products and projects
- The USD build script can now optionally inject doxygen doc strings into USD python bindings, as we have done internally at Pixar for some time
- With help from another Apple PR, we now leverage Doxygen Awesome so that our phones can be useful for USD API inquiries!
- Made progress on our backlog of doc fixes
- We added a new gitHub repo, OpenUSD-proposals, which has already greatly facilitated collaboration on new USD proposals, and is where Pixar has and will be using for all of our new OpenUSD proposals.



In the developer space, we've made a number of improvements to the API's and ecosystem, such as

- VtArray now uses operator new, making it easier for clients to largely substitute their own memory management into USD
- The new VtVisitValue visitor provides a much cleaner and more compact pattern for many cases of processing VtValue containers
- Any USD layer can now be greedily loaded and "detached" from its backing datastore, facilitating workflows where one needs to keep regenerating a layer that is in-use in another application
- OpenUSD now contains a script that is a one-stop-solution for upgrading layers or USDZ packages with respect to the handful of non-backwards-compatible schema changes we have needed to make.
- DCC's can now present just the relevant USD-supported file formats for different operations like reading and writing, as the plugins can now report their capabilities.



Other USD Developments

- **Performance**
 - 100x speedup for applying/removing API's, changing typeName
 - Composition engine: algorithmic, thread-scalability, memory enhancements
 - ... up to 35% on a profiled production shot
 - Numerous other enhancements throughout the software stack

© 2023 Disney/Pixar. All rights reserved.

In our continuing efforts to make USD ever-faster:

- We delivered an almost 100x speedup in time to apply or remove an API schema, or change a prim's typeName
- We made a set of algorithmic enhancements to the core composition engine that increased thread scalability, reduced memory consumption,
 - and on our 32 core workstations, delivered a 35% speedup in composition time on a typical production shot
- along with many other small optimizations as we discovered them



Last year we announced our USD Initiative at this BoF. Let's review what that's all about, and talk about the effects it has had on USD's evolution over the past year.



What Is the Initiative?

- Commitment from Pixar & Disney to grow USD so that we can:
 - Move faster and engage more
 - Embrace other industries
 - Help make USD a broadly applicable standard

© 2023 Disney/Pixar. All rights reserved.

- The Initiative is a commitment from Disney and Pixar to grow USD resources so that we can...
- Move faster on both internal **and** external priorities, and engage more externally
- ... and bring other industries into the USD community
- Help make USD a recognized standard suitable for many 3D domains



Progress Update

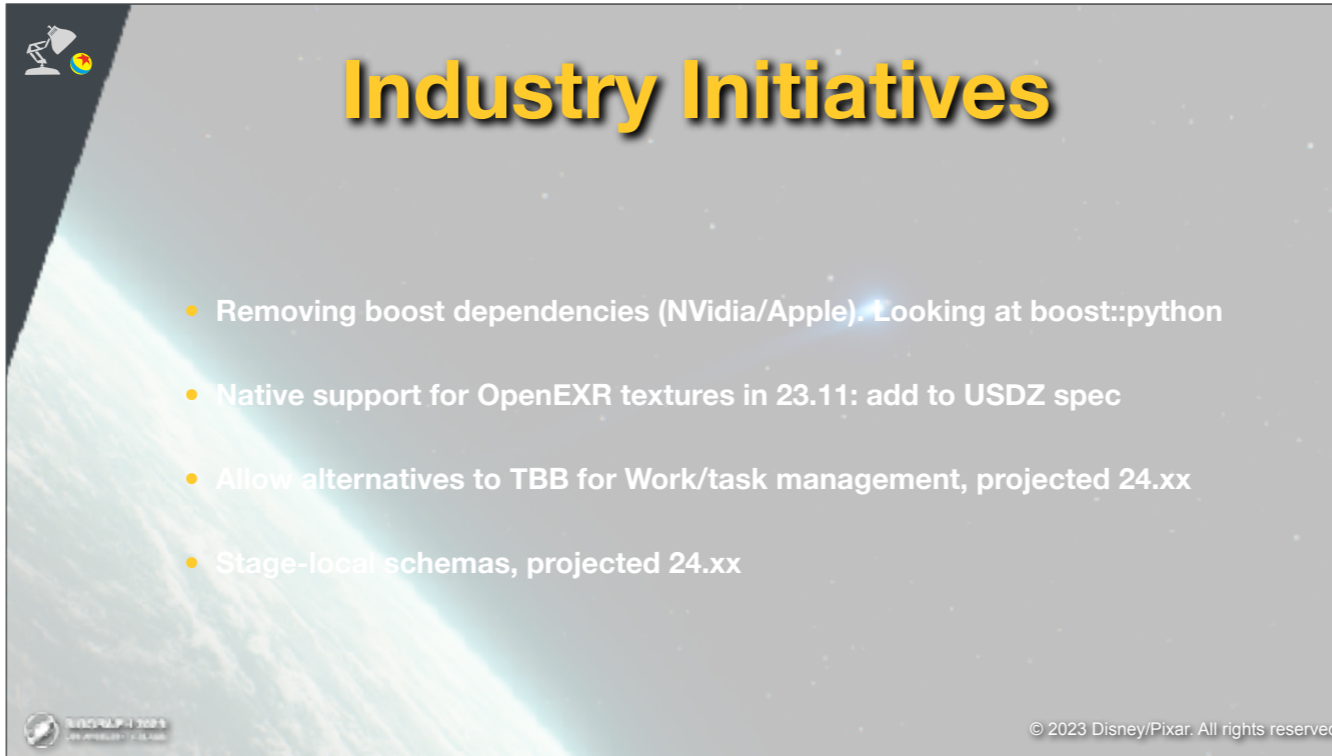
- Added full-time staff to Pixar R&D
- Partnered with Adobe, Apple, Autodesk, and NVidia to launch the Alliance for OpenUSD
- Key goal: Formal USD specification
- Engaged with MSF/Khronos on glTF/USD interop
- Addressing several key industry issues/initiatives:



Technical Relations Manager
Jess Wang

© 2023 Disney/Pixar. All rights reserved.

- We've added a handful of new developers so far across the USD, Hydra, and other related teams in our R&D group, and also filled some roles we've been sorely lacking, such as dedicated USD build and QA support, and...
- Our new technical relations manager, Jessica Wang! She's already been helping us really up our collaboration game, and she's here today, so if you'd like, come on up and introduce yourself after the presentations!
- As you heard from Steve, we've partnered with several big USD proponents and developers to create the Alliance for OpenUSD
 - The one thing I'll repeat here is that a key goal of the alliance is to create and maintain a formal specification for USD, to expand USD's reach and applicability
- We've engaged with some groups in Khronos, particularly around mutual glTF and USD understanding, and hopefully, interoperability.
- We also have updates on a number of industry initiatives, some of which we've talked about in the past



Industry Initiatives

- Removing boost dependencies (Nvidia/Apple). Looking at boost::python
- Native support for OpenEXR textures in 23.11: add to USDZ spec
- Allow alternatives to TBB for Work/task management, projected 24.xx
- Stage-local schemas, projected 24.xx

© 2023 Disney/Pixar. All rights reserved.

- Thanks largely to numerous and fantastic contributions from NVidia and Apple, we've been chipping away at USD's boost dependencies. In the coming year, Pixar plans to examine the boost::python dependency, a particularly tricky one due to our large internal codebase that relies on it.
- In 23.11, you'll get native support in Hydra's texture manager for OpenEXR textures, *without* the need for any external dependencies or plugins. OpenEXR is the HDR texture format of choice for OpenUSD, and we are working with our partners to add OpenEXR to the USDZ spec.
- Within the coming year we plan to support OneTBB (with help from some great PR's), and to start thinking about allowing our Work library to substitute an application's own desired task management system for TBB.
- We also hope to add the feature requested particularly by the games industry to package codeless schemas with stages, so that different DCC's do not need to share common USD plugin installations to share schemas.



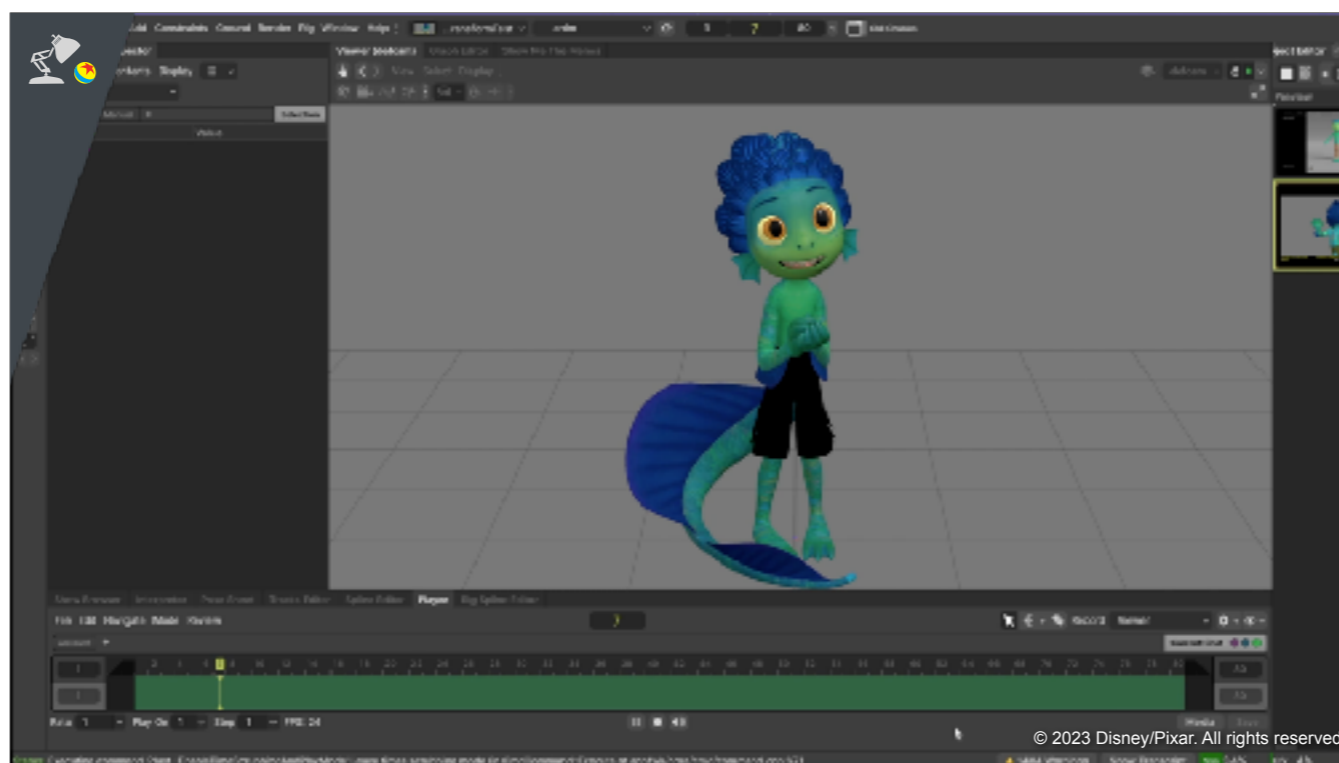
- And finally, we want to fill you in on our thinking so far about OpenExec. For that, I'll hand it over to Florian, who for years guided the development of Presto's execution engine, the inspiration for OpenExec.



Thank you, Spiff!

At last year's BOF, we teased a project called "OpenExec". After this announcement, we got a lot of questions and decide it would be worth sharing a little more detail this time around.

So, what is this "OpenExec" thing:



Well, first off: It already exists... sort of. Our plan is to open-source the production proven execution engine that is at the heart of our in-house content creation application called Presto.

It allows us to author execution graphs with computations that can be evaluated in real-time and produce cool looking, live scenes like the sea monster transitions on this Luca character rig... and the system can perform at scale like you see in these crowds workflows.



Nerdy Details



<https://tinyurl.com/openexec1>

<https://tinyurl.com/openexec2>

© 2023 Disney/Pixar. All rights reserved.

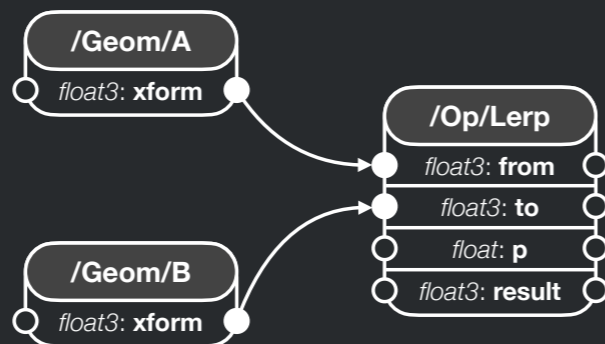
We shared details about the architecture of this system at SIGGRAPH in previous years. If you are curious to learn more, I encourage you to visit these links.

tinyurl.com, forward slash **openexec1** and **openexec2**.



Computed Values

Scene



Schema Registration

```
RegisterExecutionBehavior("Lerp")  
  .Attribute("result")  
  .Expression()  
    .AttributeInput<GfVec3f>("from")  
    .AttributeInput<GfVec3f>("to")  
    .AttributeInput<float>("p")  
    .Callback(+[](const GfVec3f &from,  
                 const GfVec3f &to,  
                 const float p){  
    return from+p*(to-from);  
});
```

© 2023 Disney/Pixar. All rights reserved.

What does this mean for USD? Rather than being a separate open-source project, we plan on making this available as functionality in USD.

In a nutshell, the system will allow you to optionally retrieve computed values about properties in the scene (like for example computed attribute values), in addition to the authored values USD returns today.

And at the risk of sharing too much detail, let me give you a flavor of what this may look like: On the left we have a Lerp prim, with some attributes connected up to other attributes in the scene. The Lerp prim also exposes an attribute called result, providing a computed value. How that value is computed is expressed in Lerp's schema registration, similar to what is illustrated on the right.



OpenExec

What it **IS**

- + General purpose computation engine with caching and invalidation capabilities
- + Supports custom types and custom computation callbacks
- + Enables scenes to return **computed** values in addition to authored values

What it is **NOT**

- Procedural generation of namespace, or culling of existing namespace
- Complete rigging system (though foundation thereof)

© 2023 Disney/Pixar. All rights reserved.

So what is this system, and what is it not:

- Our execution system is a general purpose computation engine that supports caching of previously computed values, and sending of invalidation notices as a result of dependencies changing
- The system supports custom data types and custom computation callback logic
- In essence, it enables scenes to return computed values in addition to authored values

- Architecturally, the system is NOT designed to procedurally generate namespace or remove existing subtrees
- As-is, the system is also not a complete rigging system. Although, in Presto our rigging schemas and deformation code are expressed through this system.



What's the Plan

- **In-progress on restaging system on-top of USD**
- **Looking forward to engaging with community at key decision points**
- **Open-Source as part of USD distribution**
- **Collaborate with community to evolve**

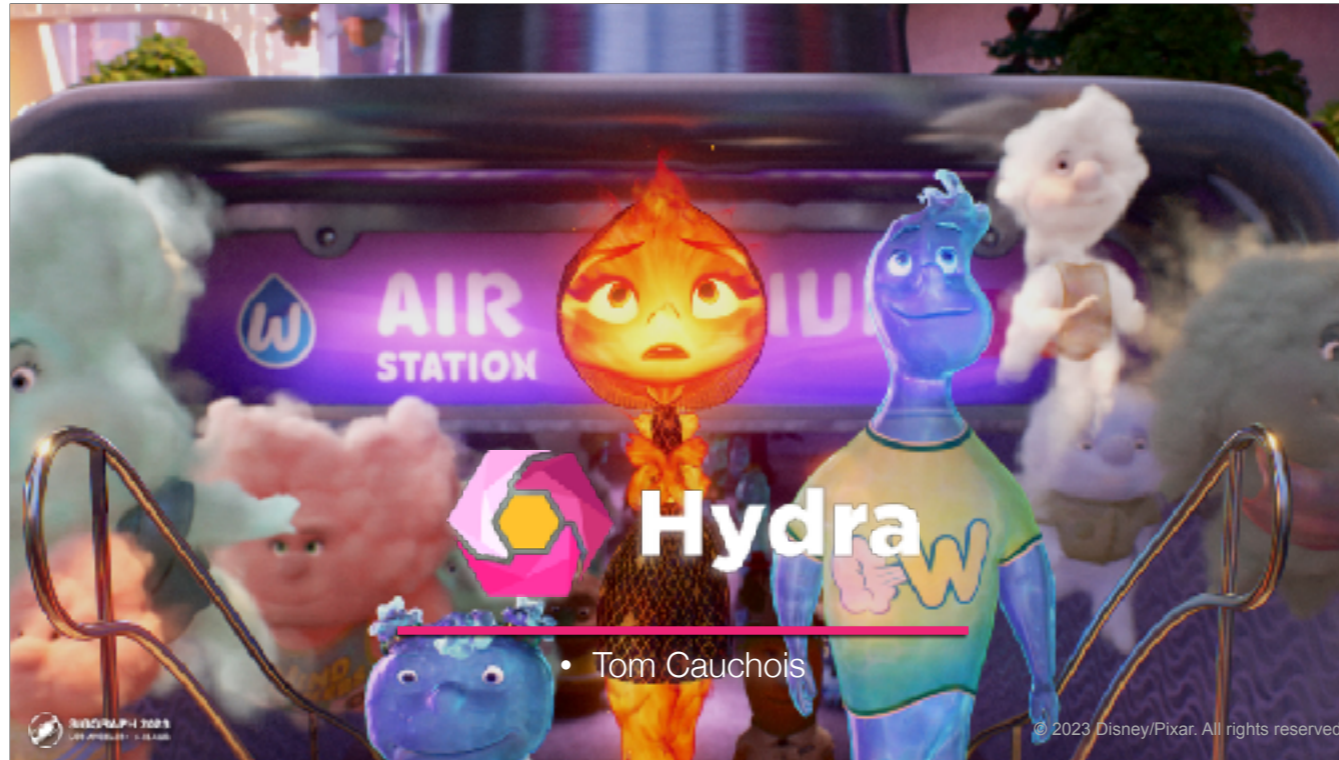
© 2023 Disney/Pixar. All rights reserved.

Okay, so what's the plan:


- We are currently in-progress on applying the scalpel to de-couple this system from Presto and restage it on-top of USD
- During this process, we anticipate encountering some Presto-isms that we instead want to imbue with a USD flavor. And we look forward to engaging with the community.
- We plan to make this system a feature of USD and will open-source it with the USD distribution
- And once that has happened we want to collaborate with the community to make it even more awesome



Thank you for listening, thanks for being part of the USD community, and stay tuned for ... Tom Cauchois who unfortunately was not able to attend SIGGRAPH in person this year, but will give us a recorded update on Hydra!



Hi! I'm Tom from Pixar, here to update you all on the progress of the Hydra project this year.



An open source framework to **transport** live scene graph **data** to **renderers**

Our goal is to support both **viewport** and **final frame** rendering

© 2023 Disney/Pixar. All rights reserved.

To recap the goals of the hydra project, it's an abstraction that transports and evaluates scene graph data and rendering commands. We ship it with the important USD and hdStorm scene and renderer integrations, but it's also an important piece of glue in our pipeline between different scene graphs, execution engines, renderers, and applications, letting each integration build off the others.

Our goal with hydra is for it to drive both viewport and final frame rendering, letting us bring the two closer together using shared technology.



Hydra 2023 Highlights

- Hydra Scene Index API/Hydra 2.0
- UsdImaging 2.0
- usdrecord and UsdRenderSettings
- Hydra Storm backends - Metal, Vulkan
- Hydra Prman features - mesh lights, instancing, and more
- MaterialX - Metal support, custom nodes, and more
- Pinned curves
- More tests open-sourced.

© 2023 Disney/Pixar. All rights reserved.

Hydra has seen a lot of big API changes this year, with the continuing rollout of the scene index API and the beginning of UsdRenderSettings support. In addition to the new workflows these support, we hope that they will lead to C++ API stability and more pipeline customizability, making it easier to maintain hydra integrations as the ecosystem continues to evolve.

Scene-index wise, the biggest news is the development of UsdImagingStageSceneIndex, which is still cooking but which already has some cool new features and eventually will be a drop-in replacement for UsdImagingDelegate.

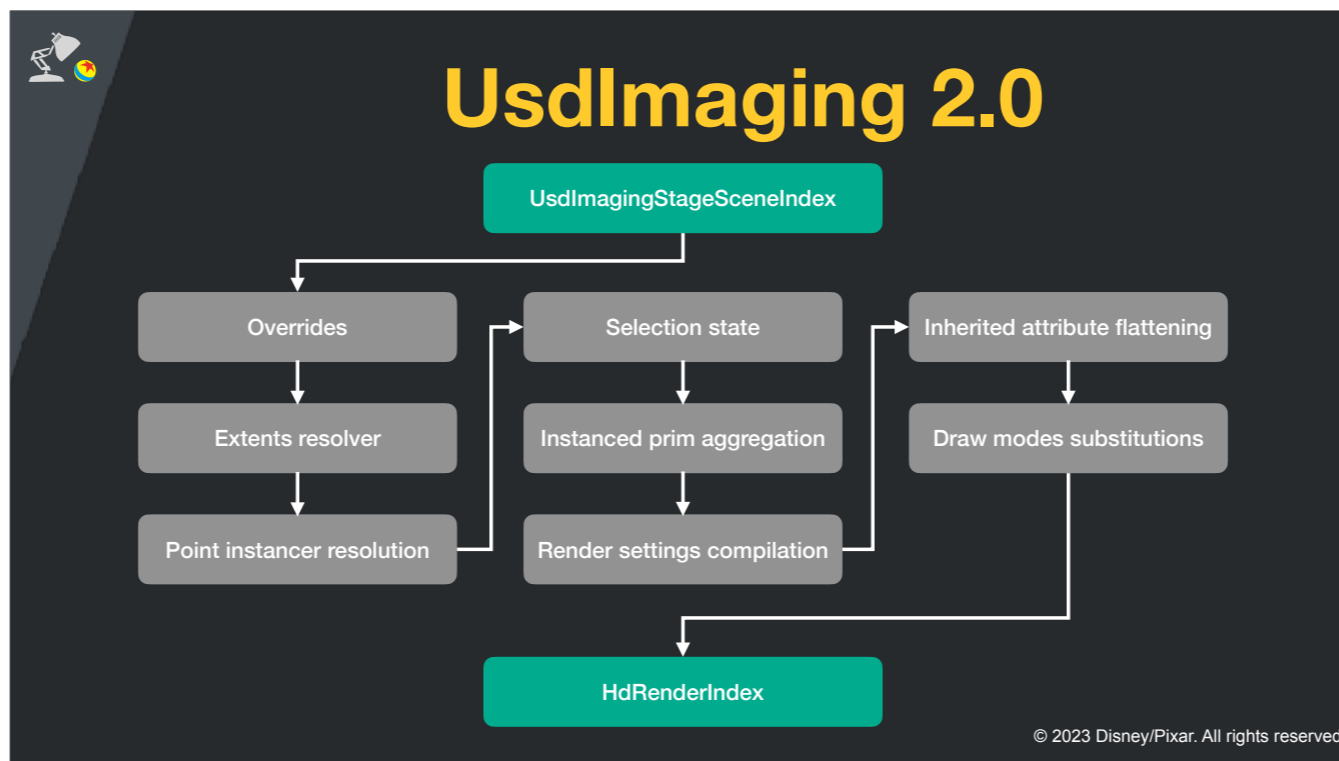
We've also made a lot of progress this year expanding the capabilities of usdrecord, and the capabilities of hydra to kick off a final render from USD.

Our focus with Storm has been to push the development of the Metal and Vulkan backends, with lots of amazing contributions to Storm from Apple's Metal Ecosystem team. For hdPrman, our Renderman integration, we've got some great new scene features.

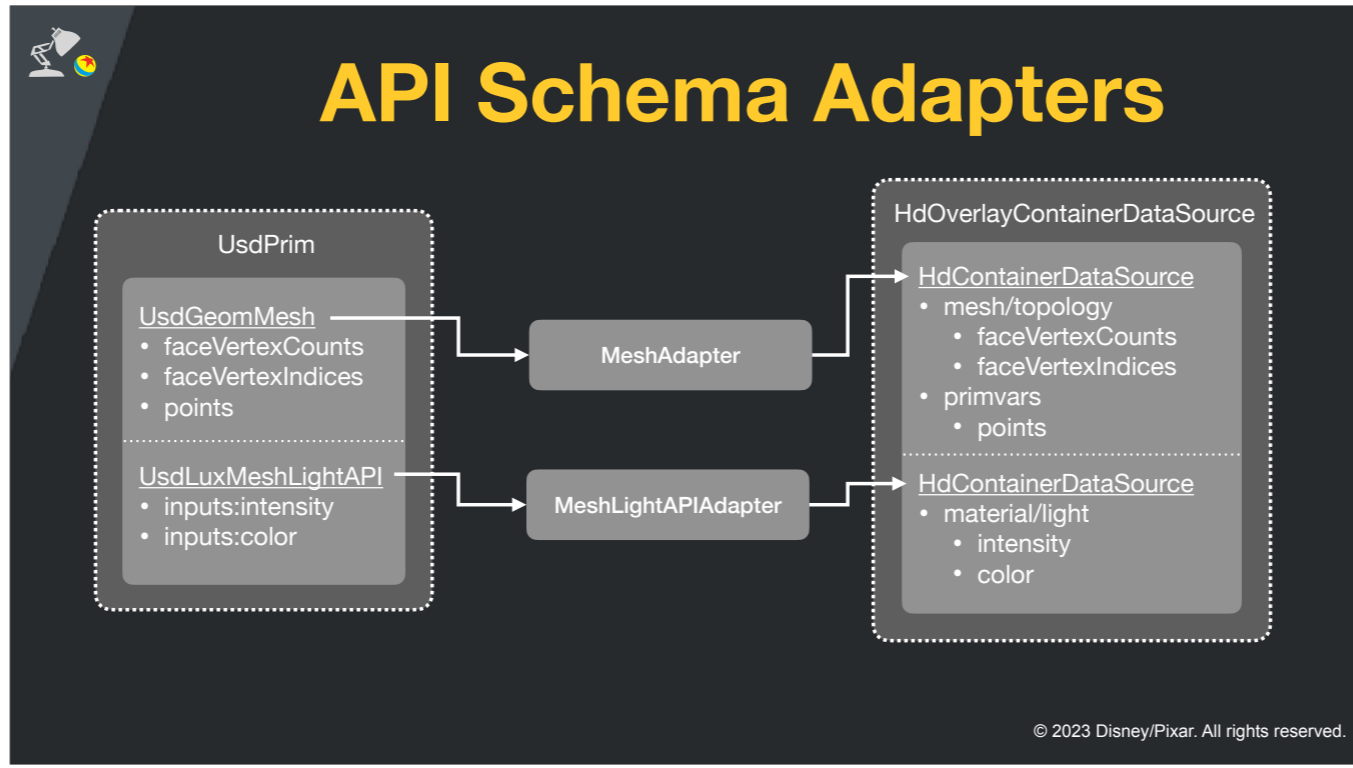
Our MaterialX integration has seen the addition of Metal support, as well as better custom node support and other integration improvements.

We've added support for USD's pinned curves concept, which extend cubic curves to their first and last vertices and represent a big space savings for things like caching hair geometry.

Finally, we're continuing efforts to open source our unit test suite, so that developers outside of Pixar can make use of it.



As I mentioned, we've developed `UsdImagingStageSceneIndex` to replace `UsdImagingDelegate`; but `usdImaging` is a complicated library that evaluates a lot of tricky residual semantics of USD scenes. When we started this work, we decided to pry apart all of those layers of evaluation into self-contained scene index filters. This picture represents the current evaluation topology of USD scenes. The really exciting thing about this is that it's easy to modify this topology, for example to add application-specific overrides or to modify our instancing logic; and it's vastly easier to maintain since the global scene transformations are self-contained in their resolving filters.



Another great feature of the new UsdImaging code is that we can register imaging adapters and behaviors for USD API schemas, and the API imaging behaviors will compose onto the base prim imaging behaviors. Just like we register a mesh adapter to turn a UsdGeomMesh into a hydra mesh, we can also register an adapter for the MeshLightAPI, which will decorate that hydra mesh with useful lighting info like intensity and color. This is enabled by the composability of container datasources, and it's been a powerful tool for us internally. It's also a way to extend the hydra object model of prims, if your pipeline has attributes that the base prim adapters don't pick up.

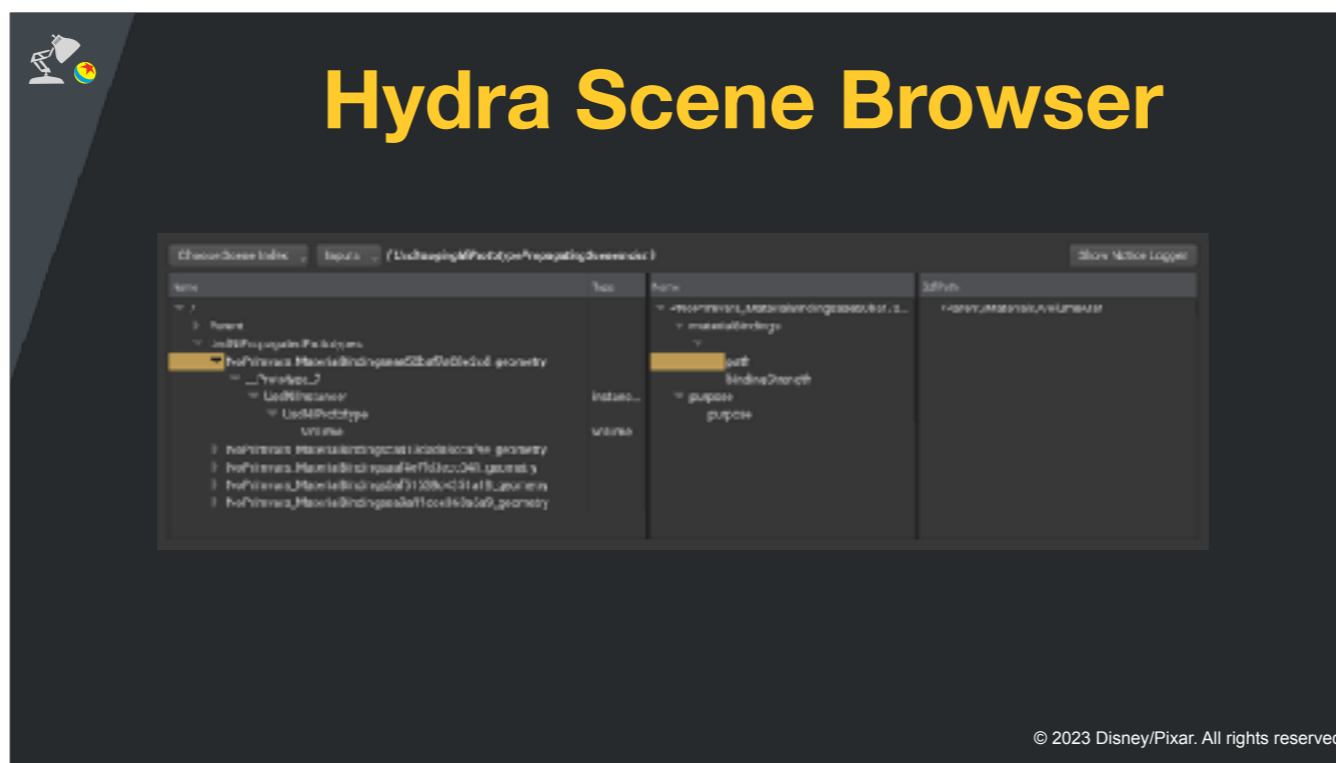


UsdImaging 2.0

- Usable in usdview via `USDIMAGINGGL_ENGINE_ENABLE_SCENE_INDEX`
- Missing a few features still:
 - UsdSkel
 - UsdGeomSubset
 - Collection support for material/light linking
 - Motion blur
- Working through edge cases
- Target is to replace `UsdImagingDelegate` & be semantically equivalent

© 2023 Disney/Pixar. All rights reserved.

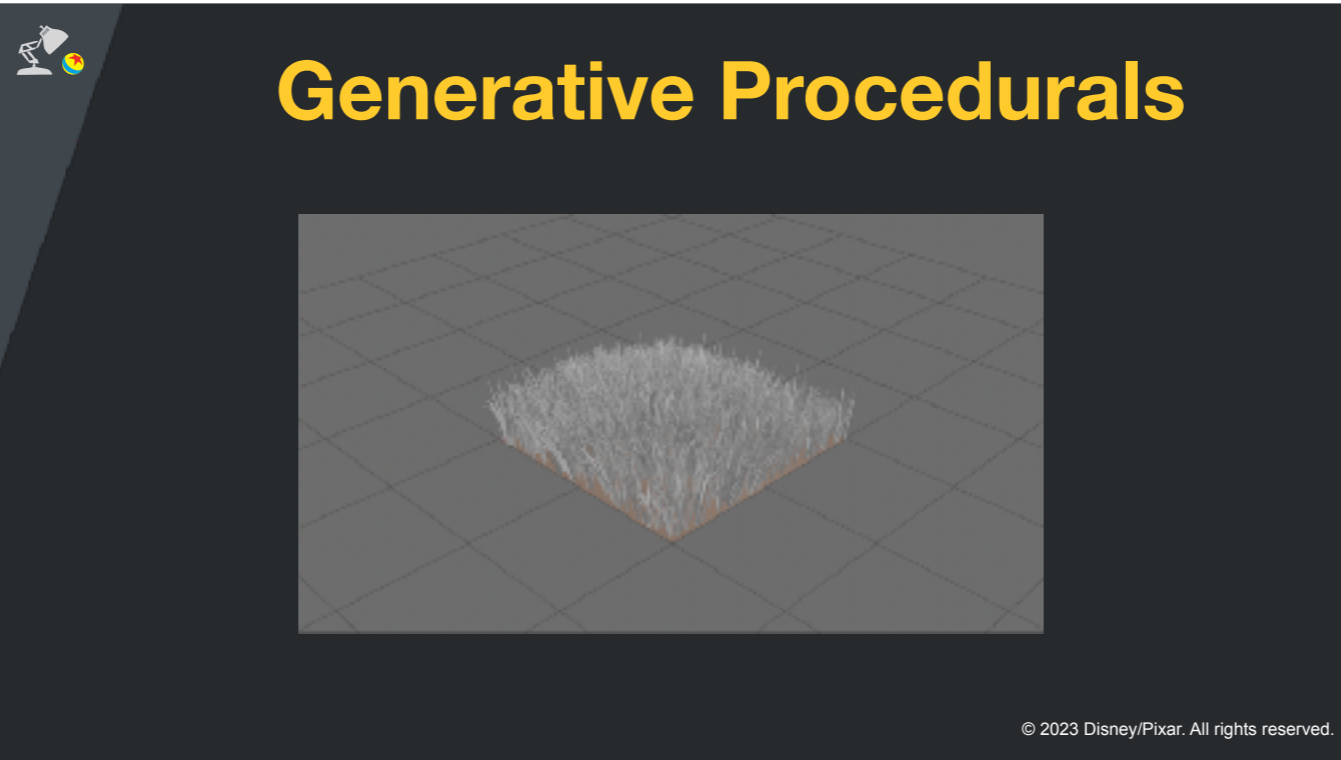
As of USD 23.08, you can test this new code in usdview by setting an environment variable. It's missing a few major features: Skel, subsets, collections, motion blur; and we're still tracking down semantic edge case differences in our test suite. Our goal is for it to be semantically equivalent to `UsdImagingDelegate`, and eventually replace that class in usdview.



NOTE - These are screen caps from the hydra scene browser (part of the Usdview UI) applied to an open source test scene in OpenUSD.

The scene index API has also enabled other really great tools in our pipeline. We talked last year about the Hydra Scene Browser, which is a UI we've added that lets you inspect the hydra view of the scene at any point in the evaluation chain. This has been a super useful debugging tool: here, I've got a screenshot of a scene before `|||` and after native instance aggregation, which is useful for following what the aggregation logic is doing.

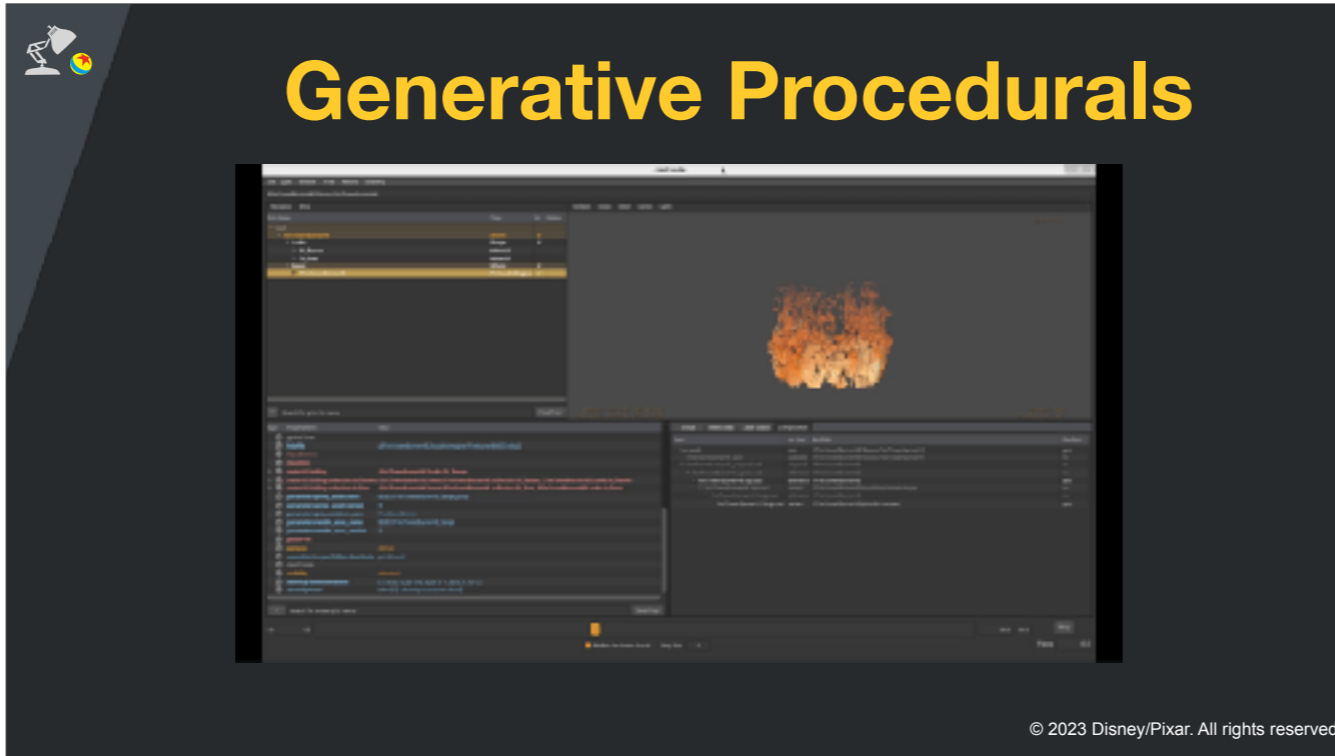
Since 23.02 the scene browser has shipped as a standard part of usdview, and we include both a python and C++ reference implementation for folks who want to implement this UI in their own app.



NOTE: this is moss running in Presto, growing a generic grass curve.

Another important use case of the scene index API is procedural evaluation. At Pixar we're building this off of the Generative Procedural system we talked about last year. This work won't land on GitHub but it's an important illustration of the possibilities of the new API.

This screenshot is one of our standard studio procedurals, which grows basis curves on a surface; here, it's grass growing on a square patch.



Another fun example is our Houdini Engine procedural, which is a Usd prim that takes a hip file and some parameters and calls out to Houdini for evaluation. Here we've got a Fire Burner from Elemental, which is being triangulated in real time for display in usdview.

The great thing about running these in hydra is that they automatically show up in any of our DCC apps; and by throwing in a level of detail control it's possible to have the same code running between a GL view for a modeling artist and a Renderman view for a lighting artist at opposite ends of the pipeline.

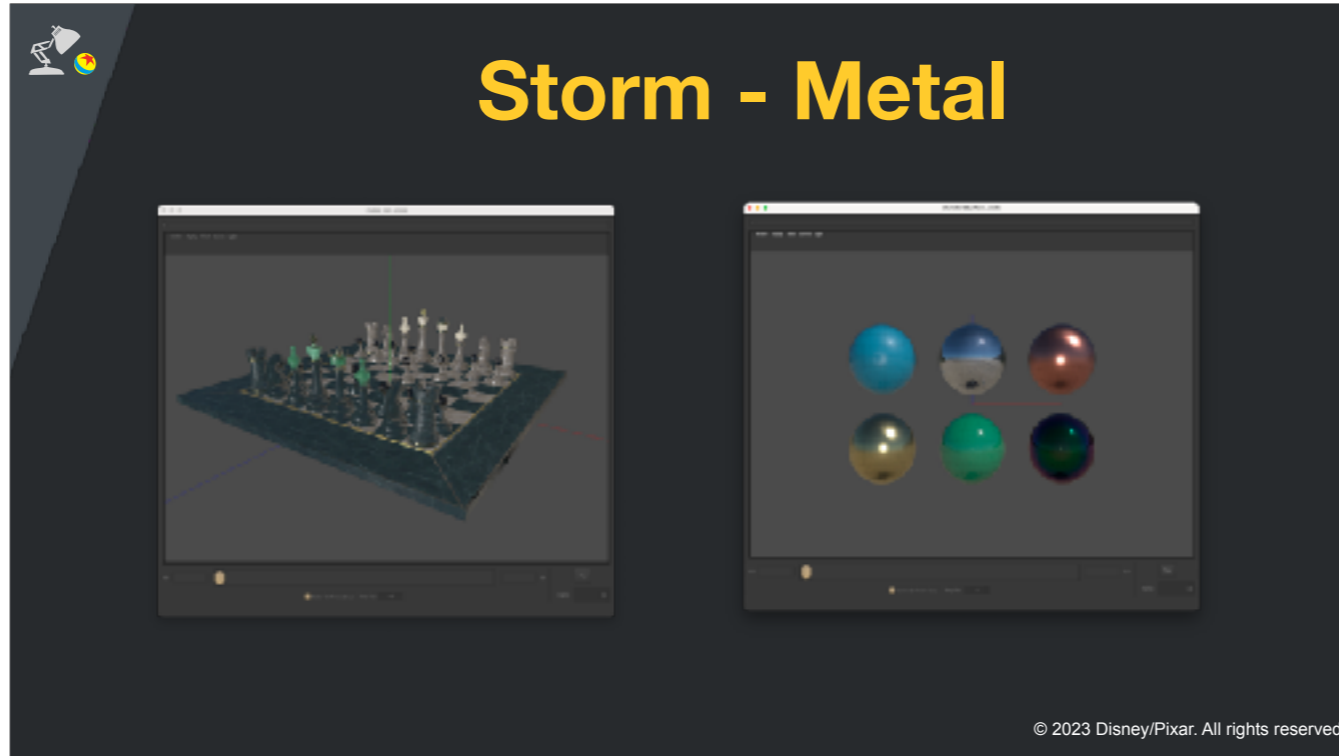


usdrecord & Render Settings

- `usdrecord --disableGPU`
- `UsdRenderSettings` support
 - `RenderSettings`, `RenderProduct`, `RenderVar`
 - For `Renderman`: sample filter, display filter, integrator
 - `HdPrman` support for integrator and hider settings
 - (Soon!) `HdPrman` support for AOV configuration from `RenderProduct`

© 2023 Disney/Pixar. All rights reserved.

Another important initiative this year has been to build up `hydra` and `usdrecord` as tools that can kick off batch renders. To that end, we've added a flag to run `usdrecord` without a GPU, which is important for cloud-based rendering; and we've also been adding support for `UsdRenderSettings` as a way to drive render configuration, targeting `usdrecord` and `HdPrman` at first. USD 23.08 supports transport of the render settings bundle, and by 23.11 we'll have `HdPrman` using render settings to configure global options, render targets, and AOVs.



NOTE: these are all open source MaterialX test assets, e.g. https://github.com/usd-wg/assets/tree/main/full_assets/OpenChessSet

Our big push this year with Storm was to round out the Metal support. In collaboration with the amazing folks on Apple's Metal Ecosystem team, we've added support for tessellating pipelines through OpenSubdiv and our basis curve tessellators, which brings us up to date on geometry support. We've also added support for GPU frustum culling, and integrated the MaterialX support for Metal shader generation. The screenshots here are MaterialX test assets that have gone through a few iterations of uniform catmull-clark refinement, as rendered by usdview on a MacBook.

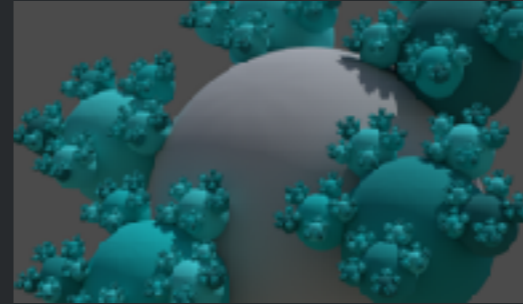
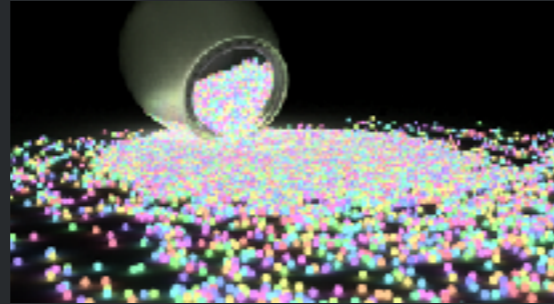


NOTE: these are all open source USDZ test assets.

Our Vulkan backend is also showing signs of life! Note that these screenshots are from a local branch that hasn't been fully merged yet, but we're passing part of our imaging test suite so we've passed the daunting first pixels hurdle. This fall we'll be adding a few more important abstractions to Hgi and bringing it up to spec. We're very excited about Vulkan's potential to unlock new lighting and tessellation pipelines for us!



Hydra Prman



- Display filters
- Integrators
- Mesh lights
- Instanced lights
- Nested instancing

© 2023 Disney/Pixar. All rights reserved.

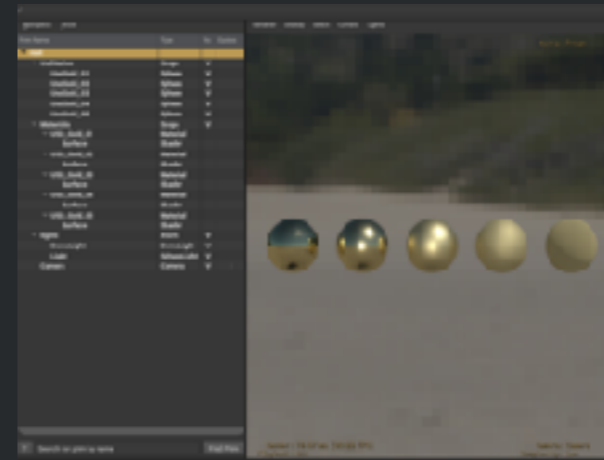
HdPrman is one of the first beneficiaries of the render settings support I mentioned earlier, and as part of that we've added support for display filters and scene-configured integrators. We've also added a cluster of exciting features: mesh lights, targeting UsdLuxMeshLightAPI; support for translating hydra nested instancing into Renderman group instancing; and support for instanced lights. This last one is great for letting artists add lights to model groups that might later be instanced.

The screenshots here are a mess of glowing marbles made of instanced mesh lights; and a fractal whose exponentially large number of spheres is enabled by Renderman nested group instancing.



MaterialX

- Hydra Storm Metal support
- Custom nodes & node graphs
- Shader generation caching
- Distant lights support
- UsdPreviewSurface reference!



© 2023 Disney/Pixar. All rights reserved.

Karen gave an excellent update on our MaterialX integration at the ASWF MaterialX town hall, but I'll give a recap. As I mentioned we're very excited about the Storm Metal support for MaterialX materials, enabled by MaterialX Metal shadergen. We've also improved support for custom nodes and node graphs, added caching around MaterialX shadergen, and improved integration with our lighting code by adding distant light support to MaterialX materials in Storm. We've also made a reference implementation of UsdPreviewSurface as a MaterialX node graph, which has been a great tool for standardizing its look across renderers.



What's next?

- Bringing UsdImagingStageSceneIndex to feature parity
- Bringing HgiVulkan to feature parity
- Finishing UsdRenderSettings support
- HdPrman as a scene index observer
- Cleaning up the UsdImagingGLEngine API
- MaterialX colorspace support
- HdPrman camera projections

© 2023 Disney/Pixar. All rights reserved.


This coming year we'll be wrapping up some of these big projects, including the new version of UsdImaging and our Vulkan backend. As the render settings support continues to land we'll start exploring the consequences for the rest of the Hydra API. We'll also start deploying some transitional API for reading scene indices from a render delegate, targeting hdPrman. We'll be updating the UsdImagingGLEngine API in support of the scene delegate changes, although its functionality will remain the same.

Featurewise, we'll be adding MaterialX colorspace support, which will lay the foundations for colorspace support more generally; and we'll be rounding out the UsdRenderSettings scene transport work by adding support for USD defined camera projections.

Lots of exciting stuff! Thanks all!

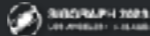
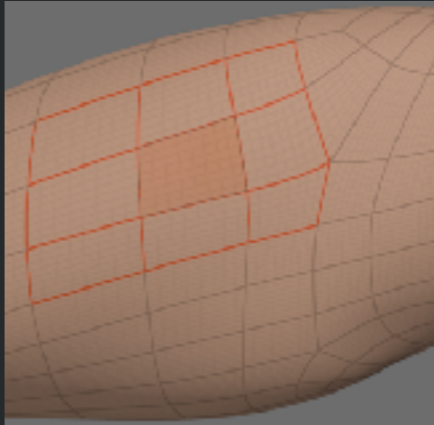


Here's a quick update on OpenSubdiv.



Current Releases

- **v3.5.0**
 - **OpenSubdiv::Bfr** - *Base Face Representation*
 - Fall 2022
- **v3.5.1**
 - **OpenSubdivConfig.cmake** and other improvements
 - Summer 2023



© 2023 Disney/Pixar. All rights reserved.

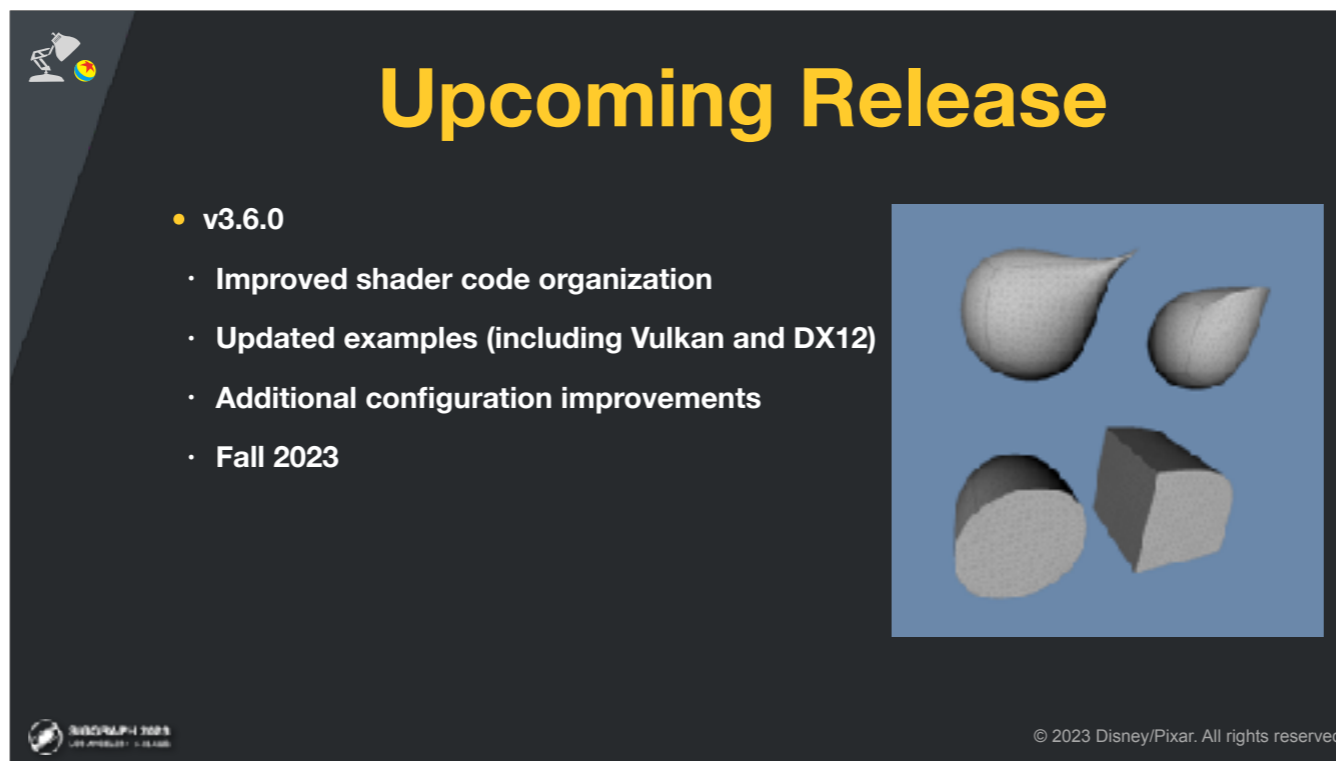
Last year we released version 3.5.0

The highlight of that release was the introduction of the `OpenSubdiv::Bfr` API, presented in more detail at last year's BoF, which allows easy access to the subdivision limit surface with no global mesh pre-processing or allocation of large tables and allows efficient use from multithreaded CPU code.

Pixar's RenderMan XPU uses OpenSubdiv and the RenderMan team has been a great partner helping to evaluate and test the Bfr API. Preliminary results from that work in progress have shown improved CPU performance and reduced memory usage for mesh tessellation using Bfr.

We recently released version 3.5.1 with additional configuration improvements and minor bug fixes.

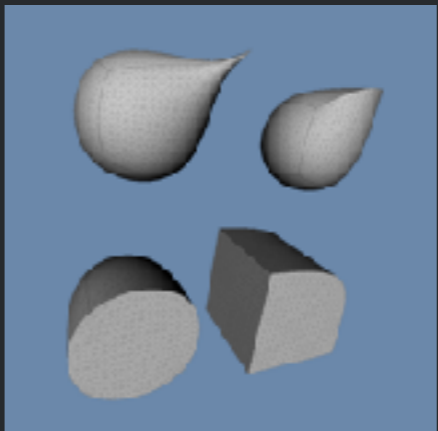
One improvement from these recent releases is that OpenSubdiv now creates and installs a CMake config so that your CMake based projects can use OpenSubdiv without needing to maintain your own CMake `find_package` module for OpenSubdiv.



The slide features a dark background with a yellow title 'Upcoming Release' at the top center. In the top left corner, there is a small icon of a microscope and a color wheel. Below the title, a bulleted list details the v3.6.0 release. To the right of the list is a blue square containing four 3D rendered objects: two teardrop shapes and two rectangular blocks. At the bottom left, there is a small circular logo with the text 'SUBDIVISION' and 'LIFE APPROXIMATE'. At the bottom right, there is a copyright notice: '© 2023 Disney/Pixar. All rights reserved.'

Upcoming Release

- **v3.6.0**
 - Improved shader code organization
 - Updated examples (including Vulkan and DX12)
 - Additional configuration improvements
 - Fall 2023



© 2023 Disney/Pixar. All rights reserved.

We've been working on version 3.6.0, which we're planning to release later this fall.

Our focus for this upcoming release is to further improve how OpenSubdiv can be incorporated into your projects.

For example, OpenSubdiv provides GPU shader source code to support evaluation of piecewise parametric patches.

We're planning some changes to the organization of that shader code to improve compatibility, for example when compiling GLSL to SPIR-V for Vulkan or when compiling HLSL using the latest DX12 features.

Another benefit of this work is that these shader source strings will be smaller for typical uses cases, improving shader compile times and readability.

This is an aspect of the work that we've previously described as "non-intrusive back-end integration" which also will make it easier for developers to work with different and newer versions of TBB, CUDA, etc.

We're also continuing to work on overall build and configuration aspects, for example to address compiler warnings and to true-up compiler settings across compilers and build systems.



Future Work

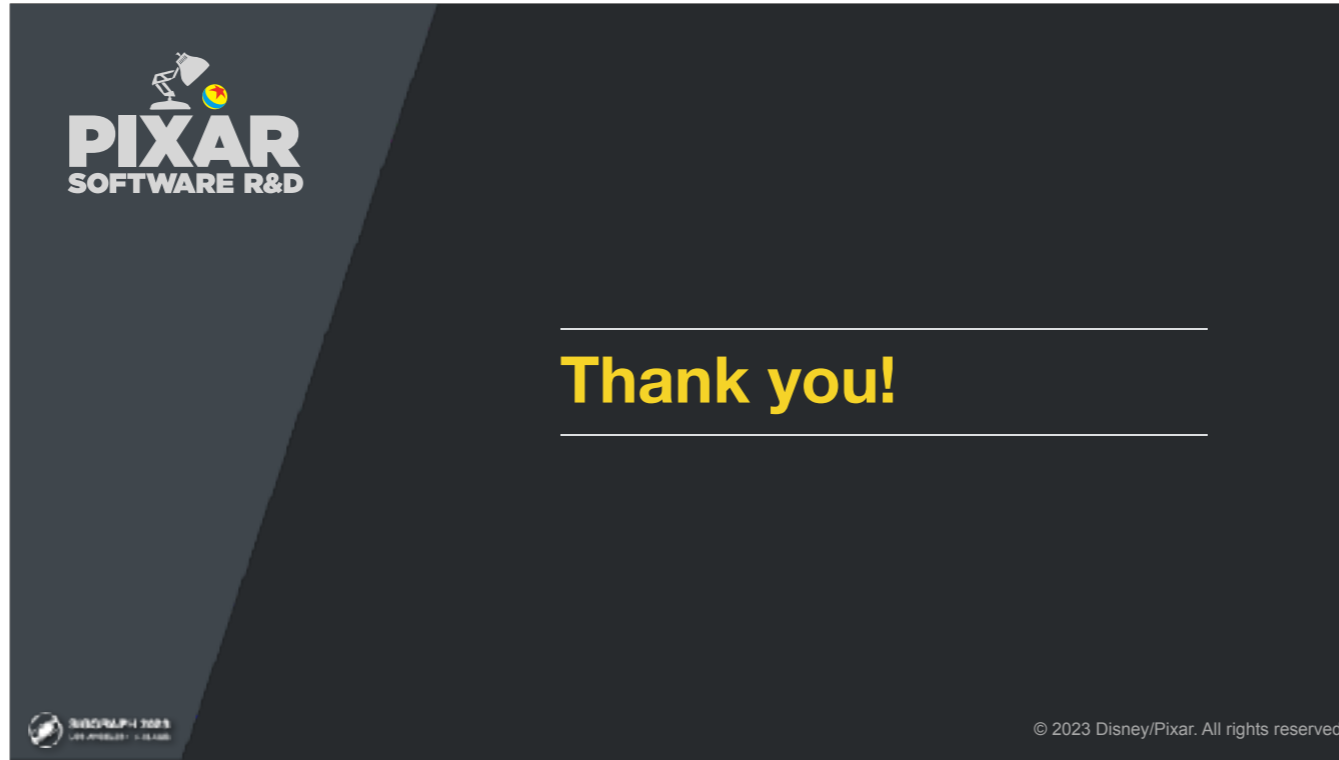
- Tessellation to triangles for GPU Ray Tracing
- Meshlet generation for GPU Mesh Shaders



© 2023 Disney/Pixar. All rights reserved.

We've been working with OpenSubdiv internally to prototype use cases like tessellating to triangles for GPU Ray Tracing and generating meshlet data for GPU Mesh Shaders.

We'll be incorporating some of this work into the examples and as always we're looking forward to seeing what the community builds on top of OpenSubdiv.



Thanks to all of the open source contributors and discussion forum participants

