Universal Scene Description

USD Basics in an Hour(ish)

Here’s what we think you should know if you only had an hour to learn the basics of Universal Scene Description (USD).

We’ve also provided additional resource links in case you want to dig in deeper when you have more time.

Please note that these materials contain links to third-party content and/or sites. These links are provided for convenience only. Pixar does not endorse or take responsibility for such content, sites or any opinions expressed by the third parties.

USD 101

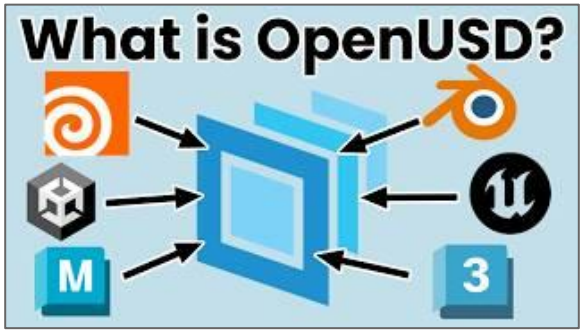
Table of contents

Short Introduction Videos

What is USD and why do people care?

Pick just one video or watch ‘em all!

USD Primer for Artists
https://youtu.be/SaBXE4yQetk
(10 min)



What is USD: A Simple Explanation
https://youtu.be/tRhJW2qRunA
(11 min)

Foundry’s Guide to USD
https://youtu.be/SaBXE4yQetk
(8 min)

Key Terminology

Not an exhaustive list but hopefully a good start

Prim	Primary USD container object; organizes a scene
Property	Describes the attributes and relationships of a Prim; the “real data” of USD
Stage	The composed view of the scene description; i.e. what you see in the end
Layer Stack	Ordered set of layers; allows breaking work down modularly
Reference	Way to compose smaller units of scene description into larger aggregates
Variant(Set)	Sets of alternatives on a prim (e.g. geometry variants, material variants)
Opinions	Authored values for metadata/properties on prims in their respective layers; LIVeRPS strength ordering informs which opinion “wins”
Composition	Process to assemble the layers of the scene graph together into the Stage
Schema	Means to author and retrieve structured data for a USD object
LIVeRPS	Mnemonic for strength ordering and composition arcs; “Liver Peas”

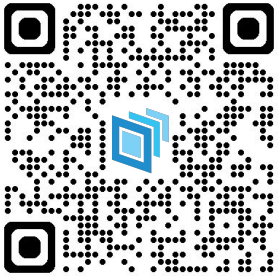
Also see

- Book of USD
- USD Glossary

Where to get USD

Source code

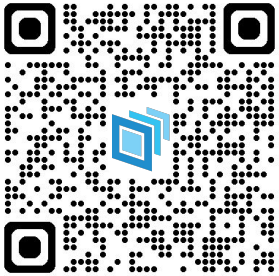
Clone and run `build_usd.py`
or customize your build



github.com/PixarAnimationStudios/OpenUSD

Pip install usd-core

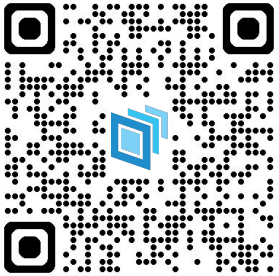
Get core USD libraries
excludes imaging/usdView and optional plug-ins like Alembic



pypi.org/project/usd-core/

Products with USD support

Community-maintained list of products with USD support



openusd.org/release/usd_products.html

USD 102

USD Asset Pipeline Example

Inspired by how Pixar builds USD assets

USD: Building Asset Pipelines

https://vimeo.com/211022588
(23 min)

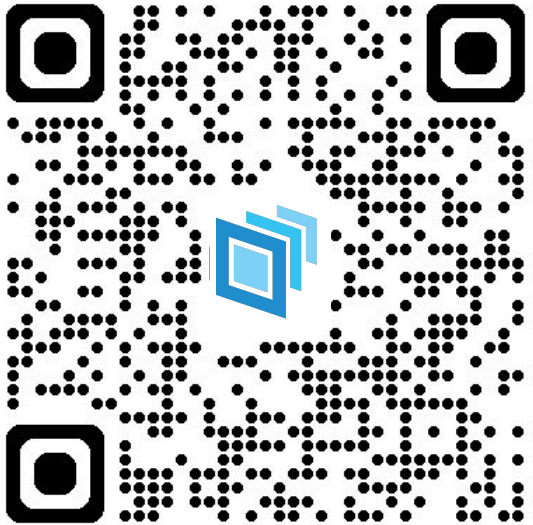
How USD could be used for building an asset pipeline, including descriptions of:

- relevant composition and underlying features of USD
- the role of schemas
- how ModelAPI and Kind work with example asset and shot structures

USD vs Other 3d Formats

TL;DR

USD offers more robust preservation of authorial choices and decisions



nickporcino.com/papers/lastmile-v2/

Extra Credit

Extending your knowledge

Deconstruct an example: Download an example Pixar asset and

- explore it in usdview to conceptualize how a USD asset fits together (see next slide)
- unzip and review the file/folder structure as inspiration for modular asset creation/collaboration

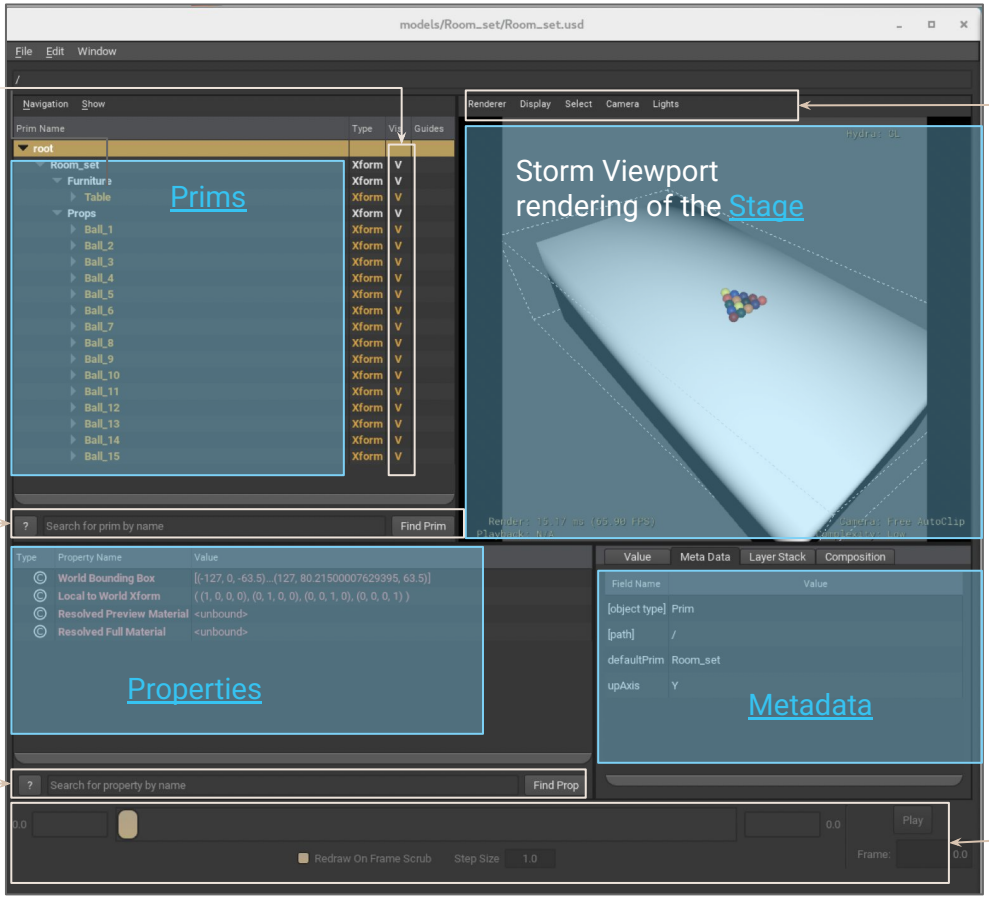
Narrow down areas of study

- Artists might focus on “USD-optimized ways” to create assets, e.g. when to use reference, variants, sublayers, instancing, etc.
- Pipeline TDs might focus on how to tailor USD to their studio’s workflows, e.g. setting up layers and custom asset resolvers/schemas/plugins; leveraging composition arcs and LIVeRPS strength ordering

Reviewing an Asset in usdView

Toggle prim visibility

USDVIEW NAVIGATION
- **F11** to pull up the scoping to hide or show geometry or parts.
- **Option (or alt) +2** to collapse all in outliner



Prim/Property Legend + Search bar

Inspect/change Camera, Lights, Selection, Render delegate, etc.

USDVIEW NAVIGATION

Trackpad/single-button mouse
- option (alt) + drag: tumble/rotate cam
- ⌘ (ctrl) + option (alt) + drag: truck/pan
- scroll: zoom

F to frame
- If hovering over viewport, F will focus on asset in viewport
- If hovering over outliner, F will find the item in the hierarchy

Play/scrub animation (if any)

Starting your own USD pipeline

- **Take advantage of products that already have USD support**
  - https://openusd.org/release/usd_products.html
    Note that different DCCs interpret USD slightly differently, so test your roundtrips early and often
- **Start small and build incrementally**
  - BYU Animation Case Study: https://dl.acm.org/doi/10.1145/3721242.3734008
- **Leverage the community**
  - See Where to go with questions?
  - Ask questions, attend working/interest group meetings, contribute back!
  - Take advantage of community-developed best practice guidelines: https://github.com/usd-wg/assets/blob/main/docs/asset-structure-guidelines.md

Where to go with questions?

USD Resources

Below is a short-list of links to get you started on the USD journey.

- Wiki: openusd.org/release
- Presentations & Videos: openusd.org/release/dl_downloads.html
- API docs: openusd.org/release/api
- Community materials
  - USD Survival Guide
  - Book of USD
  - Awesome-OpenUSD

Community

The USD community is incredibly helpful. ❤️

Pixar engineers also attend/monitor these community spaces and chime in when we can :)

- Forum: forum.aousd.org
- ASWF USD Working Group
  - Wiki: wiki.aswf.io/display/WGUSD
  - Slack: slack.aswf.io
- Feature proposals: OpenUSD-proposals
- Formal specifications: aousd.org

Appendix

How did USD come to be?

1994	2004	2008	2012
Marionnette	**Presto**	**TidScene**	**Universal Scene Description**
Shots described as single, linear program file for Toy Story in proprietary animation system, Marionette. Started to add and evolve concepts for referencing, layering, editing, and variation.	Unified referencing, overriding, variation, and other operations at all granularities into a single text format evaluated with a single composition engine.	A geometry schema with a lightweight scenegraph that enabled preview rendering across pipeline and had a referencing feature used for layering, scenegraph “isolation”, asset referencing, and some variation support.	Married the composition engine and low-level data model from Presto with the lazy-access, time-sampled data model and lightweight scenegraph from TidScene, introducing parallel computation.

What can USD do?

Represent scene description and 3D graphics concepts

Combine layers of artistic decisions through a versatile set of provided mechanisms (e.g. layer stacks, variant sets, payloads, etc.)

Image (with Hydra) to provide a “ground truth” rendering of any scene

Be extended/customized with asset resolvers, file format plugins, custom schemas and metadata

Why use USD?

- **Facilitates the use of multiple digital content creation (DCC) applications** by extensively and flexibly preserving data through the 3d pipeline.
- Allows multiple artists to **collaborate on the same assets and scenes** .
- Maximizes artistic iteration by **minimizing latency** .

