Autodesk University 2020

Universal Scene Description

深入浅出介绍

讲师介绍

翁乃奇

2000年　北京航空航天大学 电气工程学士

2002年　清华大学　软件工程学士

2005年　加拿大多伦多大学 计算机科学硕士

2006—2016年 欧特克加拿大Maya开发总部

Maya API，Motionbuilder SDK 开发咨询师

定制Maya解决方案开发工程师

2017—至今　南京原力三维动画制作公司

高级开发工程师，技术负责

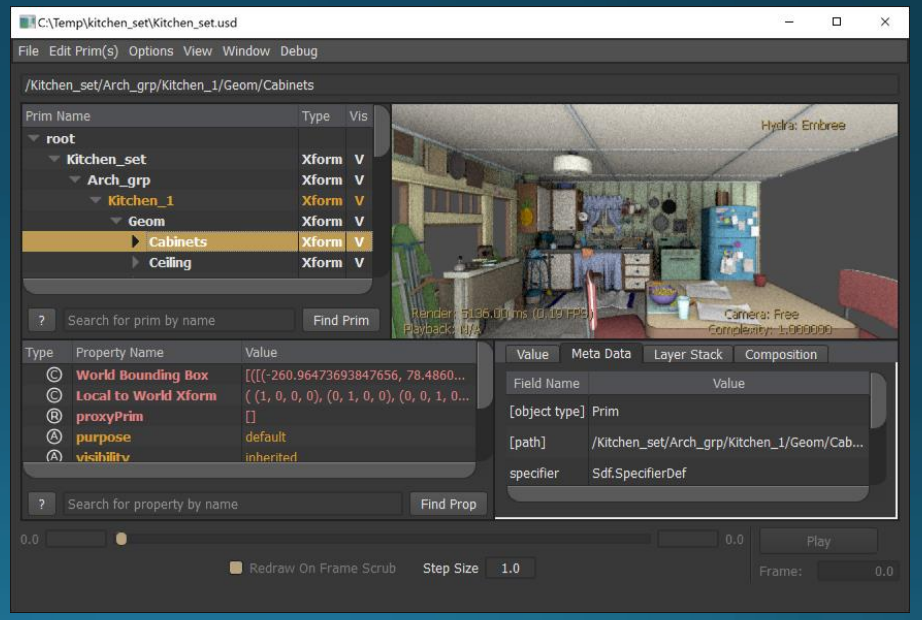
Agenda

- USD 基本术语和概念
- USD 内部框架
- USD 应用场景

USD 基本术语和概念

Universal Scene Description

- Pixar推出的一套开源文件格式库以及相应的数据解析引擎
- 在生产流程中不同应用程序之间传递3D数据的文件格式
- 提供一套C++/Python API 对文件进行 non-destructive 编辑(override)
- 一套对文件进行各种操作的工具集



Why USD

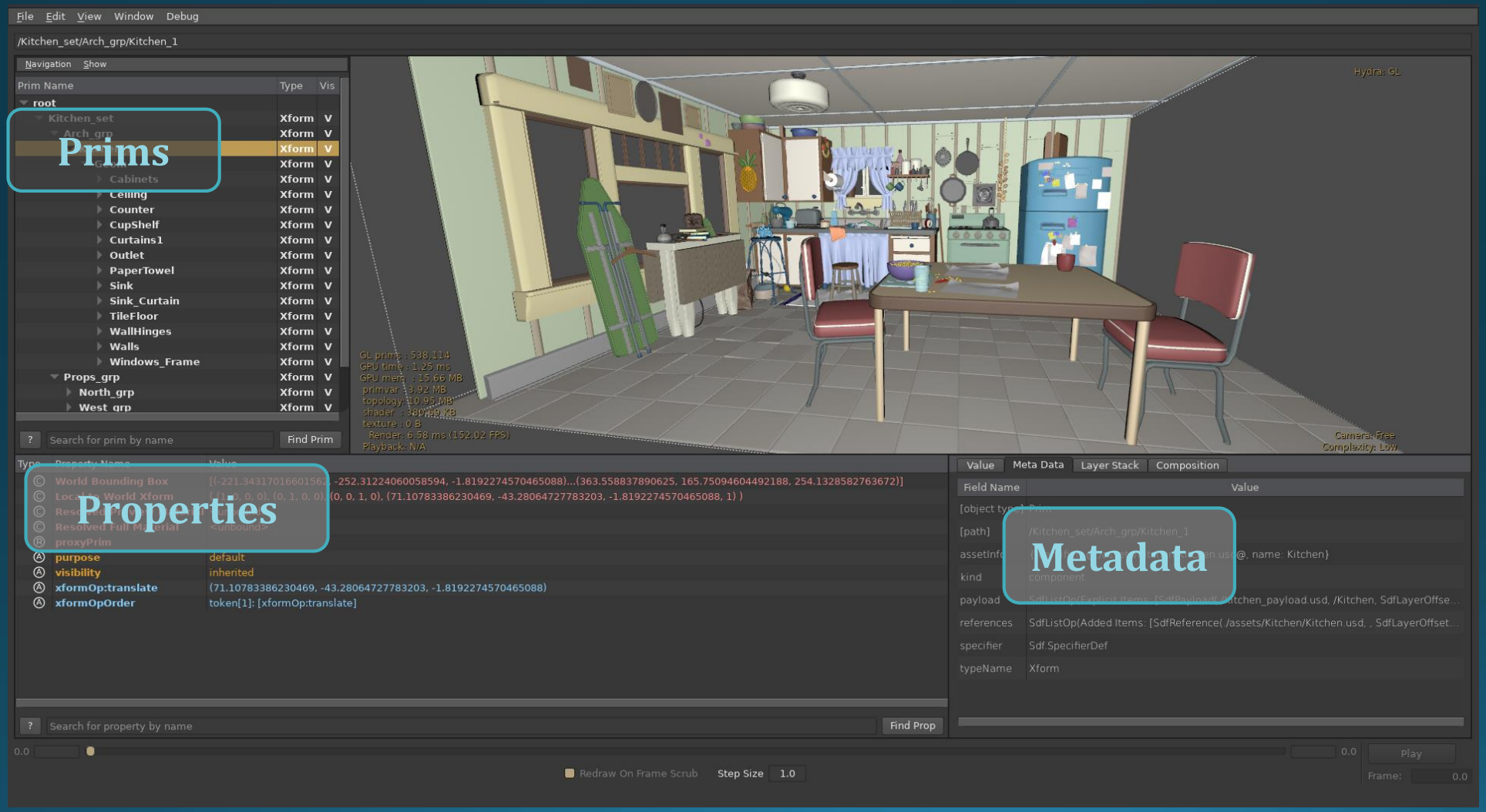
Collaboration

- 统一的语言和文件格式，用来定义打包组装以及编辑3D数据，无缝衔接多个不同应用程序和软件
- 多人对同一资产或者场景进行改动协作
- 最大化加快艺术家迭代速度

USD基本概念

- Stage
- Prim & Properties
- Layer & layer stack
- Composition arcs

Stage


File Edit View Window Debug
/Kitchen_set/Arch_grp/Kitchen_1
Navigation Show
Prim Name
Type
Vis
root
Kitchen_set
Arch_grp
Cabinets
Ceiling
Counter
CupShelf
Curtains1
Outlet
PaperTowel
Sink
Sink_Curtain
TileFloor
WallHinges
Walls
Windows_Frame
Props_grp
North_grp
West_grp
Xform
Prims
Search for prim by name
Find Prim
Hydra: GL
Camera: Free
Complexity: Low
Properties
World Bounding Box
proxyPrim
purpose
default
visibility
inherited
xformOp:translate
(71.10783386230469, -43.28064727783203, -1.8192274570465088)
xformOpOrder
token[1]: [xformOp:translate]
Search for property by name
Find Prop
Value
Meta Data
Layer Stack
Composition
Field Name
Value
[path]
/Kitchen_set/Arch_grp/Kitchen_1
kind
payload
references
SdfListOp(Added Items: [SdfReference(/assets/Kitchen/Kitchen.usd, , SdfLayerOffset...
specifier
Sdf.SpecifierDef
typeName
Xform
Metadata
Redraw On Frame Scrub
Step Size
1.0
Play
Frame:
0.0

Stage

helloWorld.usda

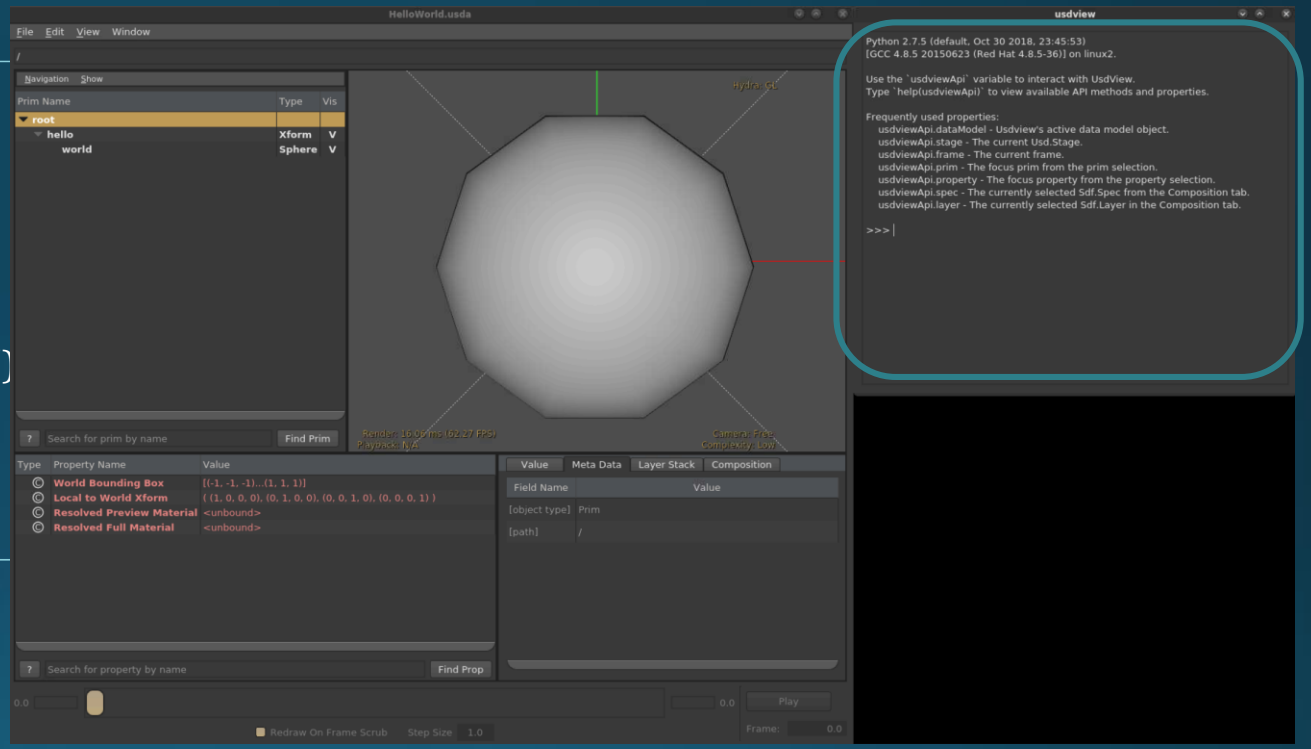
Prims

Properties

```
#usda 1.0

def Xform “hello”
{
   def Sphere “world”
   {
      float3[ ] extent = [(-2, -2, -2), (2, 2, 2)]
      color3f[ ] primvars:displayColor = [(0, 0, 1)]
      double radius = 2
   }
}
```

Stage

```
>>> from pxr import Usd, UsdGeom

>>> stage = Usd.Stage.CreateNew('helloworld.usda')

>>> xformPrim = UsdGeom.Xform.Define(stage, '/hello')

>>> spherePrim = UsdGeom.Sphere.Define(stage, '/hello/world')

>>> stage.GetRootLayer().Save()
```

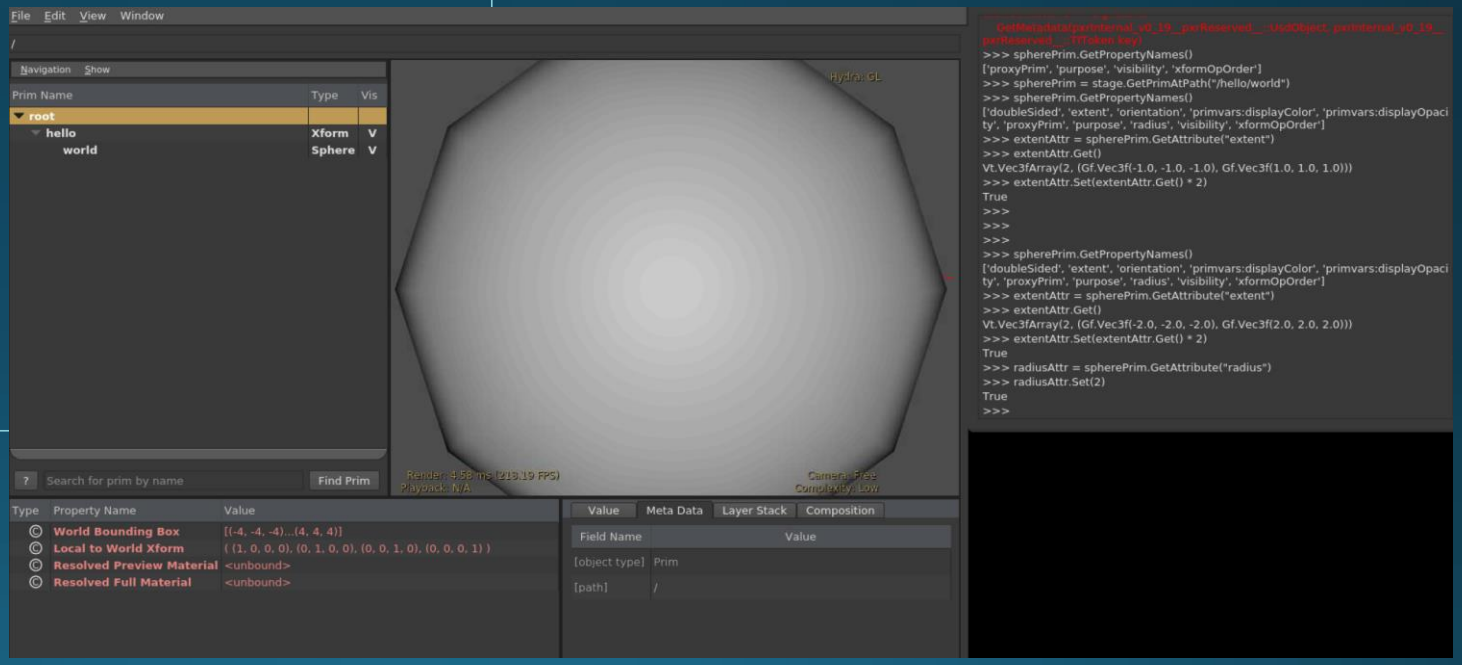


helloworld.py

```
$ python extras/usd/tutorials/helloWorld/helloWorld.py
```

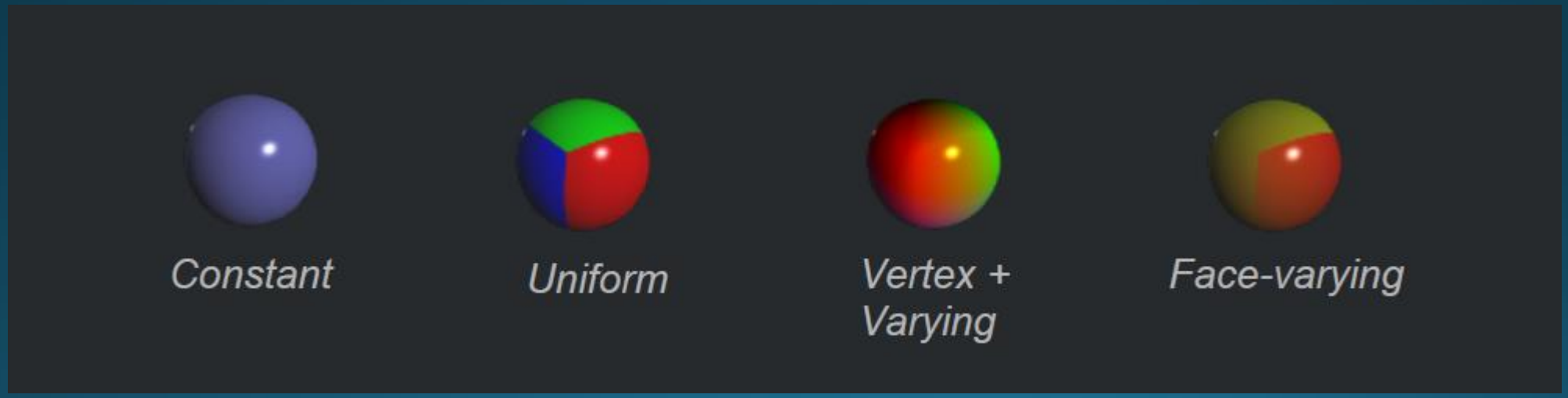
Prims & Properties

```
>>> extentAttr = spherePrim.GetAttribute("extent")
>>> extentAttr.Get()
Vt.Vec3fArray(2, (Gf.Vec3f(-1.0, -1.0, -1.0), Gf.Vec3f(1.0, 1.0, 1.0)))

>>> extentAttr.Set(extentAttr.Get() * 2)
True

>>> radiusAttr = spherePrim.GetAttribute('radius')
>>> radiusAttr.Set(2)
True
```



Primvar

- 非几何拓扑的几何体元素数据： 位置， uv， 颜色
- 随几何体表面变化
- 根据拓扑维度来分类： per-primitive, per-face, per-vertex



Primvar

helloworld.usda

```
#usda 1.0

def Xform "hello"
{
    def Sphere "world"
    {
        float3[ ] extent = [(-2, -2, -2), (2, 2, 2)]
        color3f[ ] primvars:displayColor = [(0, 0, 1)]
        double radius = 2
    }
}
```

PrimVar

Cubemesh.usda

```
#usda 1.0

def Xform "pCube1" (
)
{
   def Mesh "pCubeShape1"
   {
      int[] faceVertexCounts = [4, 4, 4, 4, 4, 4]
      int[] faceVertexIndices = [0, 1, 3, 2, 2, 3, 5, 4, 4, 5, 7, 6, 6, 7, 1, 0, 1, 7, 5, 3, 6, 0, 2, 4]
      normal3f[] normals = [(-0.57735026, -0.57735026, 0.57735026), (0.57735026, -0.57735026, 0.57735026), (0.7071068, 0, 0.7071068), (-0.7071068, 0,
0.7071068), (0, 1, 0), (0, 1, 0), (0, 1, 0), (0, 1, 0), (-0.7071068, 0, -0.7071068), (0.7071068, 0, -0.7071068), (0.57735026, -0.57735026, -0.57735026), (-0.57735026, -
0.57735026, -0.57735026)] (
         interpolation = "faceVarying"
      )
      point3f[] points = [(-0.5, -0.5, 0.5), (0.5, -0.5, 0.5), (-0.5, 0.5, 0.5), (0.5, 0.5, 0.5), (-0.5, 0.5, -0.5), (0.5, 0.5, -0.5), (-0.5, -0.5, -0.5), (0.5, -0.5, -0.5)]
      float2[] primvars:st = [(0.375, 0), (0.625, 0), (0.375, 0.25), (0.625, 0.25), (0.375, 0.5), (0.625, 0.5), (0.375, 0.75), (0.625, 0.75), (0.375, 1), (0.625, 1), (0.875, 0),
(0.875, 0.25), (0.125, 0), (0.125, 0.25)] (
         interpolation = "faceVarying"
      )
      int[] primvars:st:indices = [0, 1, 3, 2, 2, 3, 5, 4, 4, 5, 7, 6, 6, 7, 9, 8, 1, 10, 11, 3, 12, 0, 2, 13]
   }

}
```

Layer

- Every stage has a root layer and root layer stack

```
>>> shotStage =Usd.Stage.CreateNew('helloworldShot.usda')
>>> paths=["helloWorldshot_layout.usd", "helloWorldshot_sets.usd"]
>>> shotStage.GetRootLayer().SetSubLayerPaths(paths)
```

helloworldShot.usda

```
#usda 1.0
(
subLayers = [
      @helloWorldshot_layout.usd@,
      @helloWorldshot_sets.usd@,
    ]
)
```

Sublayers

helloWorldShot.usda

```
#usda 1.0
(
subLayers = [
      @helloWorldshot_layout.usd@,
      @helloWorldshot_sets.usd@,
    ]
)
```

layerStack

```
helloWorldShot.usda
helloWorldShot_layout.usda
helloWorldShot_sets.usda
```

helloWorldShot_layout.usda

```
#usda  1.0
def Xform “World”
{
   def Xform “Character”
   {
      def Mesh “GirlA”
      {…….
      }
   }
}
```

helloWorldShot_sets.usda

```
#usda  1.0
def Xform “World”
{
   def Xform “Sets”
   {
      def Xform“DeskA”
      {…….
      }
   }
}
```

helloWorldShot.usda(composed)

```
#usda 1.0
def Xform “World”
{
   def Xform “Character”
   {
      def Mesh “GirlA”
      {…….
      }
   }

   def Xform “Sets”
   {
      def Xform“DeskA”
      {…….
      }
   }
}
```

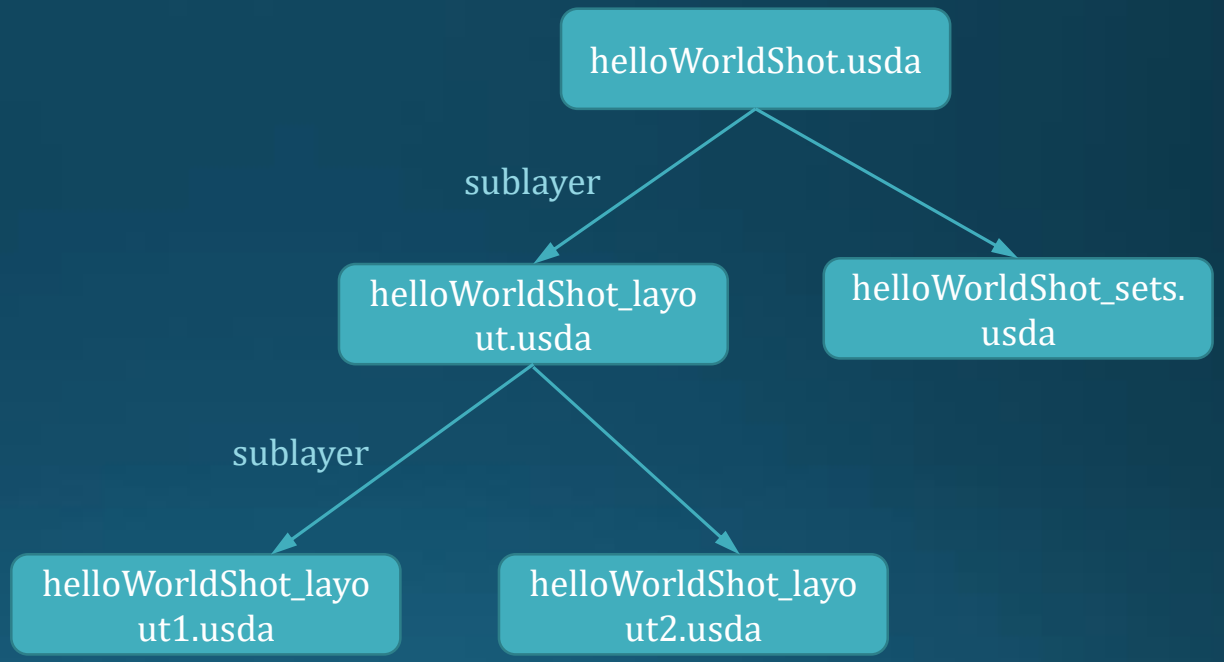
Sublayers

helloWorldShot.usda

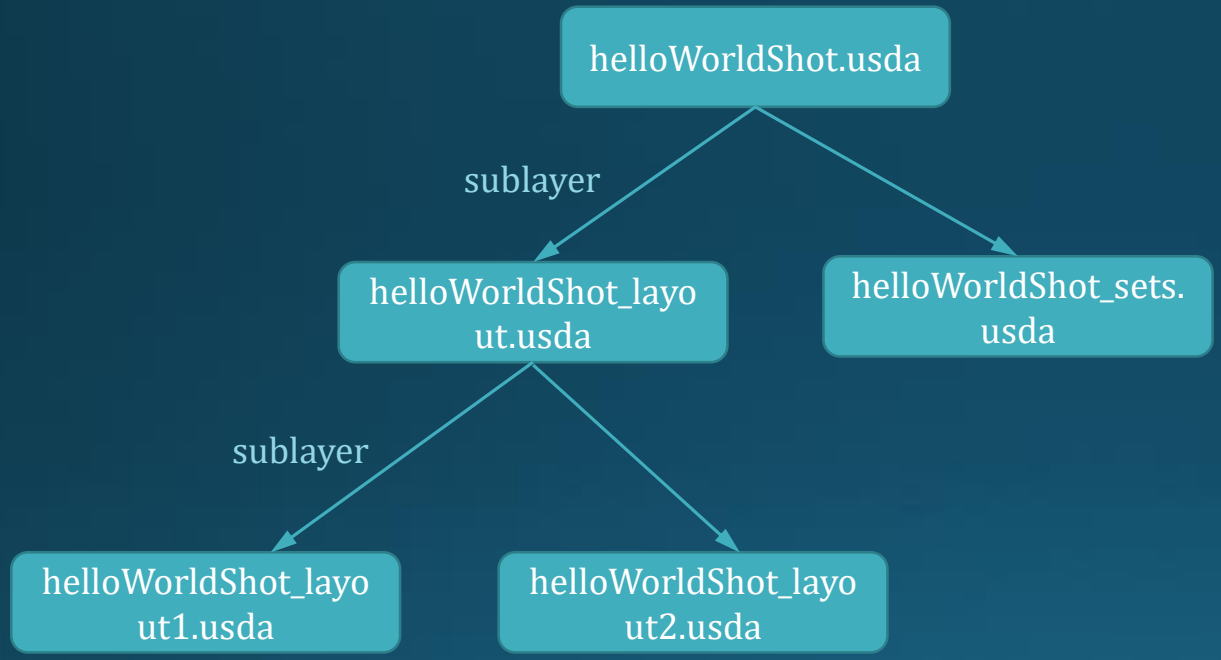
```
#usda 1.0
(
subLayers = [
        @helloWorldshot_layout.usd@,
        @helloWorldshot_sets.usd@,
        ]
)
```

helloWorldShot_layout.usda

```
#usda 1.0
(
subLayers = [
        @helloWorldshot_layout1.usd@,
        @helloWorldshot_layout2.usd@,
        ]
)
```



Sublayers



layerStack

helloWorldShot.usda
helloWorldShot_layout.usda
helloWorldShot_layout1.usda
helloWorldShot_layout2.usda
helloWorldShot_sets.usda

Composition Arcs（合成弧）

- Sublayers
- References　　UsdReferences
- Payloads　　UsdPayloads
- Variants　　UsdVariants UsdVariantSets
- Inherits　　UsdInherits
- Specializes　　UsdSpecializes

References（引用）

helloWorld.usda

```
#usda 1.0

def Xform “hello”
{
    def Sphere “world”
    {
        float3[ ] extent = [(-2, -2, -2), (2, 2, 2)]
        color3f[ ] primvars:displayColor = [(0, 0, 1)]
        double radius = 2
    }
}
```

superWorld.usda

```
#usda 1.0

over "refSphere" (
    prepend references = @./helloWorld.usda@
)
{
    uniform token[] xformOpOrder = []
}

over "refSphere2" (
    prepend references = @./helloWorld.usda@
)
{
    over "world"
    {
        color3f[] primvars:displayColor = [(1, 0, 0)]
    }
}
```

References

referenceLayer.py

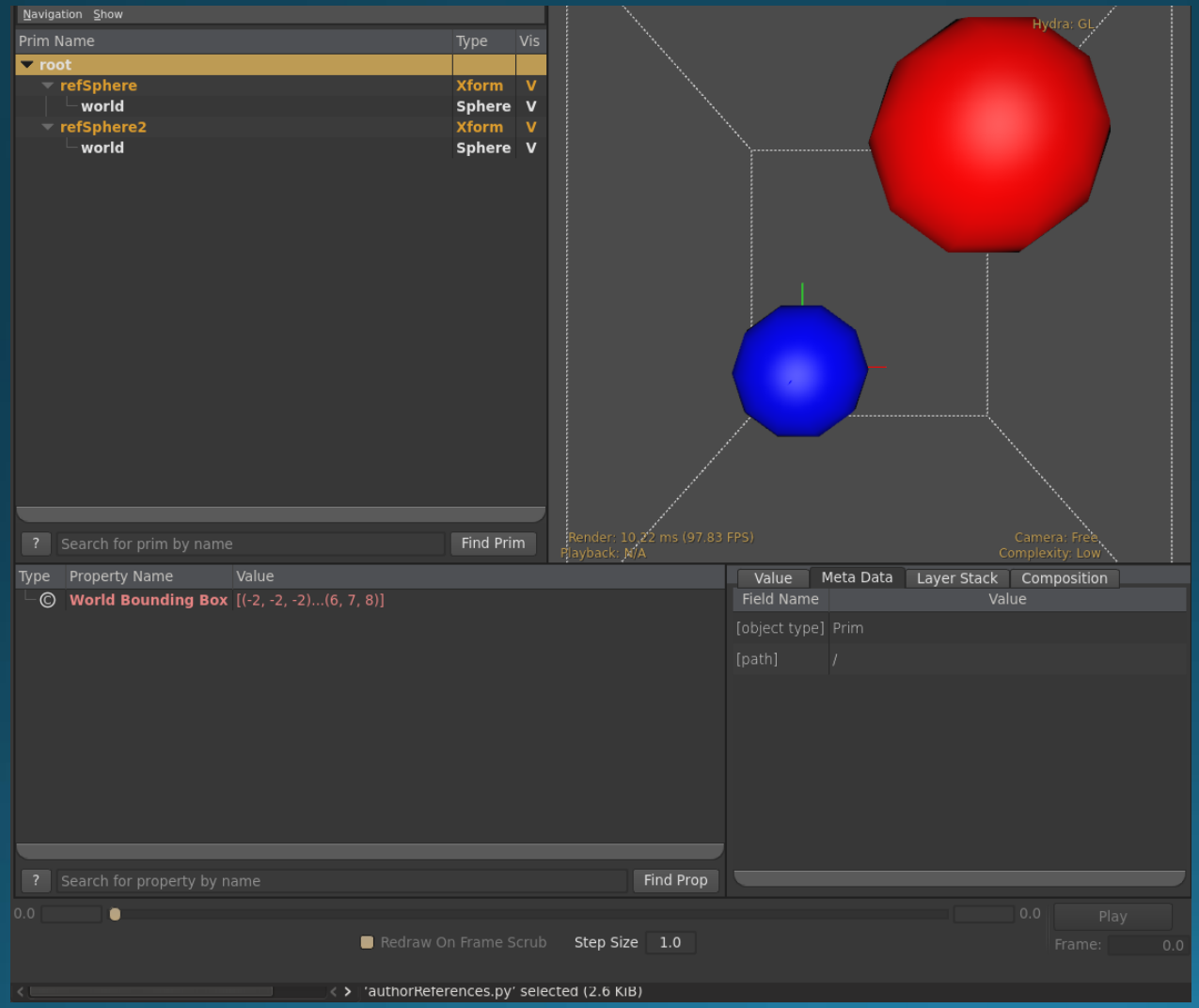
```
>>> refStage = Usd.Stage.CreateNew('superWorld.usda')

>>> refSphere = refStage.OverridePrim('/refSphere')
>>> refSphere.GetReferences().AddReference('./helloWorld.usda')

>>> refXform = UsdGeom.Xformable(refSphere)
>>> refXform.SetXformOpOrder([])
>>> refStage.GetRootLayer().Save()
```

```
>>> refSphere2 = refStage.OverridePrim('/refSphere2')
>>> refSphere2.GetReferences().AddReference('./helloWorld.usda')

>>> overSphere = UsdGeom.Sphere.Get(refStage, '/refSphere2/world')
>>> overSphere.GetDisplayColorAttr().Set( [(1, 0, 0)] )

>>> refStage.GetRootLayer().Save()
```

superWorld.usda

```
#usda 1.0

over "refSphere" (
    prepend references = @./helloWorld.usda@
)
{
    uniform token[] xformOpOrder = []
}

over "refSphere2" (
    prepend references = @./helloWorld.usda@
)
{
    over "world"
    {
        color3f[] primvars:displayColor = [(1, 0, 0)]
    }
}
```

References


Navigation Show
Prim Name
Type
Vis
root
refSphere
Xform
V
world
Sphere
V
refSphere2
Xform
V
world
Sphere
V
Search for prim by name
Find Prim
Hydra: GL
Render: 10.22 ms (97.83 FPS)
Playback: N/A
Camera: Free
Complexity: Low
Type
Property Name
Value
World Bounding Box
[(-2, -2, -2)...(6, 7, 8)]
Search for property by name
Find Prop
Value
Meta Data
Layer Stack
Composition
Field Name
Value
[object type]
Prim
[path]
/
0.0
0.0
Play
Redraw On Frame Scrub
Step Size
1.0
Frame:
0.0
'authorReferences.py' selected (2.6 KiB)

Payloads

- 可选择性加载引用场景
- 实时加载复杂场景

UsdStage::Load()
UsdStage::Unload()

payloadWorld.usda

```
#usda 1.0

def “payloadSphere" (
   payload = [
       @./helloWorld.usda@</hello/world>
    ]
)
```

Variants （变体）

- 在单个资产上实现不同的表现形式
- 用户在合成过程中选择不同的variant

helloWorld.usda

```
#usda 1.0

def Xform “hello”
{
   def Sphere “world”
   {
      float3[ ] extent = [(-2, -2, -2), (2, 2, 2)]
      color3f[ ] primvars:displayColor = [(0, 0, 1)]
      double radius = 2
   }
}
```

clearColor.py

```
>>> from pxr import Usd, UsdGeom
>>> stage = Usd.Stage.Open(“helloWorld.usda”)

>>> spherePrim  = UsdGeom.Sphere.Get(stage, “/hello/world”)

>>> colorAttr = spherePrim.GetDisplayColorAttr()
>>> colorAttr.Clear()

>>> stage.GetRootLayer().Save()
```

Variants

- 在单个资产上实现不同的表现形式
- 创建 variant sets 来记录所有的变体
- 用户在合成过程中选择不同的变体

helloWorld.usda

```
#usda 1.0

def Xform "hello"
{
    def Sphere "world"
    {
        float3[ ] extent = [(-2, -2, -2), (2, 2, 2)]
        color3f[ ] primvars:displayColor
        double radius = 2
    }
}
```

clearColor.py

```
>>> from pxr import Usd, UsdGeom
>>> stage = Usd.Stage.Open("helloWorld.usda")

>>> spherePrim  = UsdGeom.Sphere.Get(stage, "/hello/world")

>>> colorAttr = spherePrim.GetDisplayColorAttr()
>>> colorAttr.Clear()

>>> stage.GetRootLayer().Save()
```

Variants

```
>>> from pxr import Usd, UsdGeom
>>> rootPrim = stage.GetPrimAtPath('/hello')
>>> vset = rootPrim.GetVariantSets().AddVariantSet('shadingVariant')

>>> vset.AddVariant('red')
>>> vset.AddVariant('blue')
>>> vset.AddVariant('green')


>>> vset.SetVariantSelection('red')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(1,0,0)])

>>> vset.SetVariantSelection('blue')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(0,0,1)])

>>> vset.SetVariantSelection('green')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(0,1,0)])
```

```
#usda 1.0

def Xform "hello" (
 variants = {
    string shadingVariant = "green"
  }
  prepend variantSets = "shadingVariant"
)
{
  def Sphere "world"
  {
    float3[] extent = [(-2, -2, -2), (2, 2, 2)]
    color3f[] primvars:displayColor
    double radius = 2
  }

variantSet "shadingVariant" = {
    "blue" {
        over "world"
      {
        color3f[] primvars:displayColor = [(0, 0, 1)]
      }
    }

    "green" {
        over "world"
      {
        color3f[] primvars:displayColor = [(0, 1, 0)]
      }
    }

    "red" {
        over "world"
      {
        color3f[] primvars:displayColor = [(1, 0, 0)]
      }
    }
```

Variants

```
>>> from pxr import Usd, UsdGeom
>>> rootPrim = stage.GetPrimAtPath('/hello')
>>> vset = rootPrim.GetVariantSets().AddVariantSet('shadingVariant')

>>> vset.AddVariant('red')
>>> vset.AddVariant('blue')
>>> vset.AddVariant('green')


>>> vset.SetVariantSelection('red')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(1,0,0)])

>>> vset.SetVariantSelection('blue')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(0,0,1)])

>>> vset.SetVariantSelection('green')
>>> with vset.GetVariantEditContext():
        colorAttr.Set([(0,1,0)])

>>> stage.GetRootLayer().Save()
```

Composed result

```
#usda 1.0

def Xform "hello"
{
    def Sphere "world"
    {
        float3[] extent = [(-2, -2, -2), (2, 2, 2)]
        color3f[] primvars:displayColor = [(0, 1, 0)]
        double radius = 2
    }

}
```

Inherits（继承）

- 类似C++代码中从基类继承
- 用于大批量改动同一父类的资产

helloWorld.usda

```
#usda 1.0

class “class_Geometry”
{
}

def Xform "hello"
{
   def Sphere "world“(
        inherits = [
            </class_Geometry>
         ])
   {
      float3[] extent = [(-2, -2, -2), (2, 2, 2)]
      color3f[] primvars:displayColor = [(0, 1, 0)]
      double radius = 2
   }
}
```

cube.usda

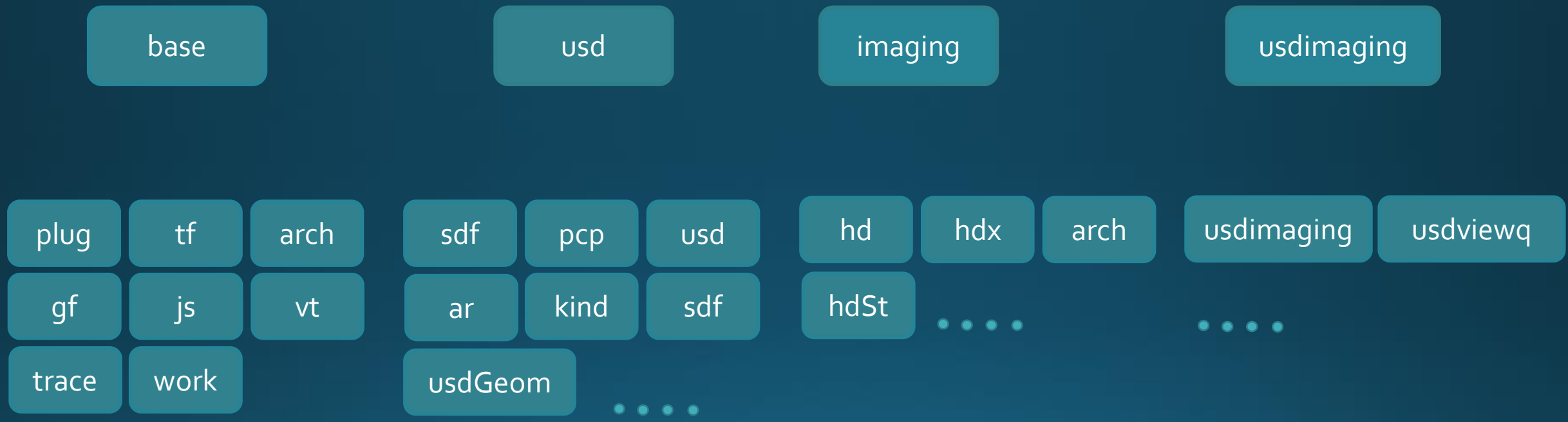
```
#usda 1.0
class “class_Geometry”
{
}


def  Cube “cube1“(
         inherits = [
              </class_Geometry>
          ])
    {
     }
}
```

scene.usda

```
#usda 1.0
class “class_Geometry”
{
      color3f[] primvars:displayColor = [(1, 0, 0)]
}


def  “sceneSphere“ (
           prepend references = @./helloWorld.usda@
      )
   {
    }

def “SceneSphere” (
         prepend references = @./cube.usda@
      )
    {
    }

}
```

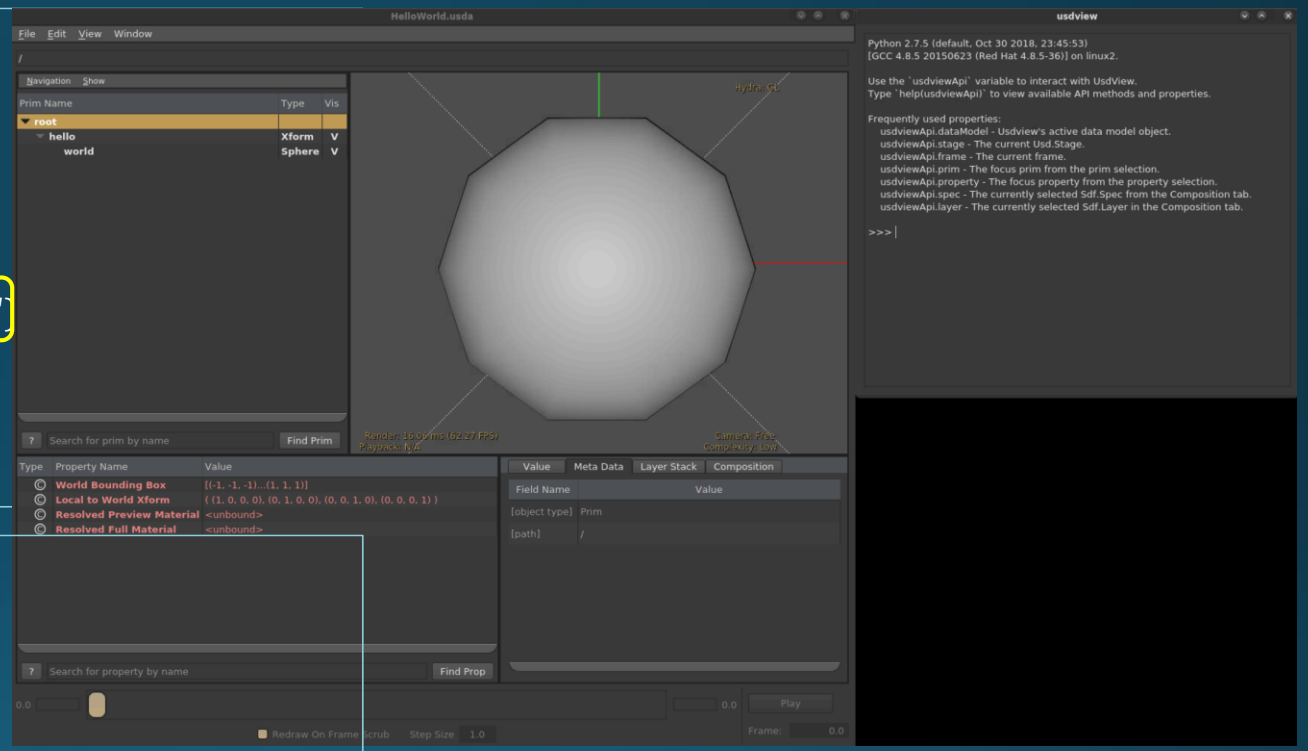
USD 内部框架

USD Framework



Schema

```
>>> from pxr import Usd, UsdGeom

>>> stage = Usd.Stage.CreateNew('helloworld.usda')

>>> xformPrim = UsdGeom.Xform.Define(stage, '/hello')

>>> spherePrim = UsdGeom.Sphere.Define(stage, '/hello/world')

>>> stage.GetRootLayer().Save()
```




```
>>> from pxr import Usd

>>> stage = Usd.Stage.CreateNew('helloworld.usda')

>>> xformPrim = stage.DefinePrim('/hello', 'Xform')

>>> spherePrim = stage.DefinePrim('/hello/world', 'Sphere')

>>> stage.GetRootLayer().Save()
```

Schema

- 带有类型含义(domain-specific) 的元素(Prim)定义
- 定义每种特定类型 Prim 的属性以及可对其进行的操作

Geometry	Volumes	Shading	Lighting	Skeleton Animation	Rendering
UsdGeom	UsdVol	UsdShade	UsdLux	UsdSkel	UsdRender

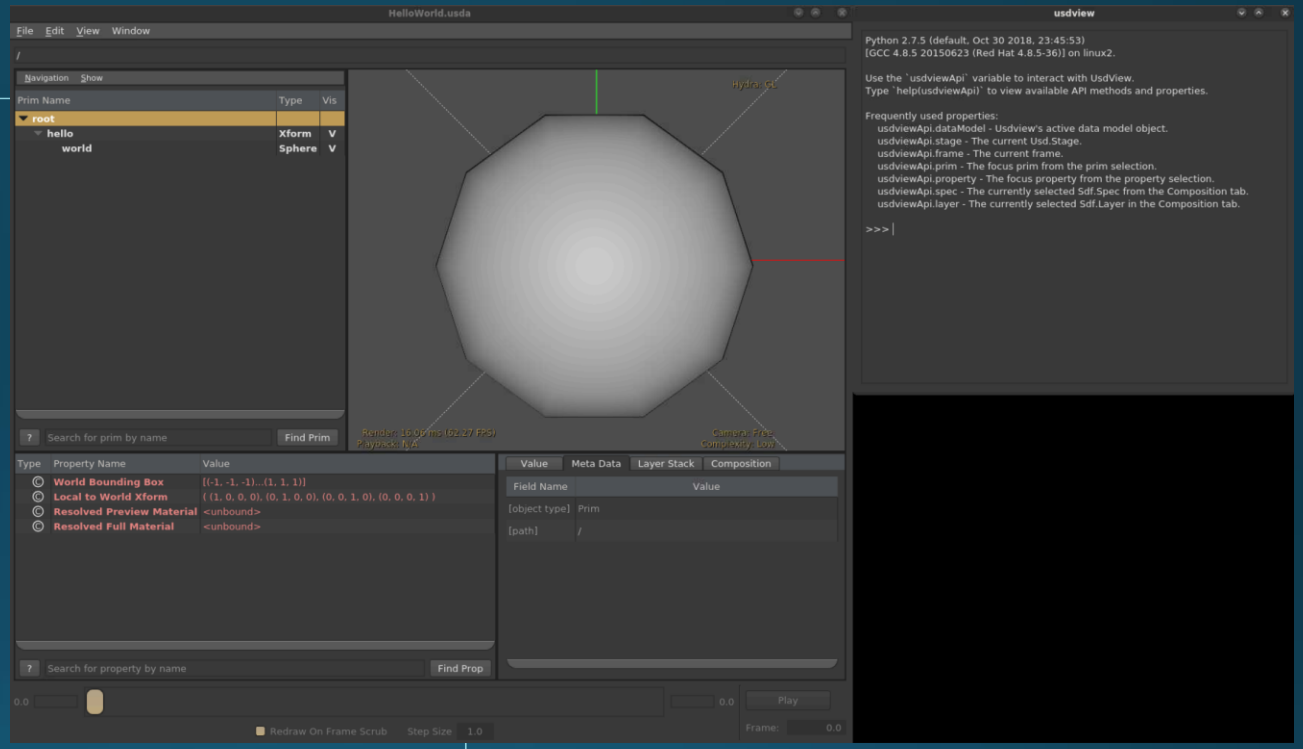
Schema

```
>>> from pxr import Usd, UsdGeom

>>> testPrim = stage.GetPrimAtPath('/hello/world')

>>> if testPrim:
>>>     print “This prim exists!”
```

This prim exists

```
>>>testCamera = UsdGeom.Camera(testPrim)
>>> if testCamera:
>>>     print “This prim is a camera”
>>> else:
>>>     print “This prim is not a camera”
```

This prim is not a camera



C++:
Assert(!testCamera)

Custom Schema

- Create a usda file
- $ usdGenSchema schema.usda
- Build the schema plugin

$ cmake --build . --target install --config Release

```
#usda 1.0
(
subLayers = [
    @usd/schema.usda@
]
)
over "GLOBAL" (
customData = {
    string libraryName = "myLib"
    string libraryPath = "./"
}
) {
}
class "MyCustomPrim" (
    inherits = </Typed>
    customData = {
        string className = "MyBasePrim"
    }
) {
# Some base attributes common to all derived schemas
uniform double uniformScale = 1.0 (
doc = "A double valued uniform attribute representing scale."
)
float3 rotation = (0, 0, 0) (
    doc = "A varying 3D vector in floating-pt precision representing rotation."
)}
```

```
Processing schema classes:
    SimplePrim, ComplexPrim, ParamsAPI
    # This is mainly needed for definition of UsdTyped.
Loading Templates
Writing Schema Tokens:
    unchanged extras/usd/examples/usdSchemaExamples/tokens.h
    unchanged extras/usd/examples/usdSchemaExamples/tokens.cpp
    unchanged extras/usd/examples/usdSchemaExamples/wrapTokens.cpp
Generating Classes:
    unchanged extras/usd/examples/usdSchemaExamples/simple.h
    unchanged extras/usd/examples/usdSchemaExamples/simple.cpp
    unchanged extras/usd/examples/usdSchemaExamples/wrapSimple.cpp
    unchanged extras/usd/examples/usdSchemaExamples/complex.h
    unchanged extras/usd/examples/usdSchemaExamples/complex.cpp
    unchanged extras/usd/examples/usdSchemaExamples/wrapComplex.cpp
    unchanged extras/usd/examples/usdSchemaExamples/paramsAPI.h
    unchanged extras/usd/examples/usdSchemaExamples/paramsAPI.cpp
    unchanged extras/usd/examples/usdSchemaExamples/wrapParamsAPI.cpp
    unchanged extras/usd/examples/usdSchemaExamples/plugInfo.json
Generating Schematics:
    unchanged extras/usd/examples/usdSchemaExamples/generatedSchema.usda
```

Composition Arcs(合成弧)

helloWorldShot_layout.usda

```
#usda  1.0
def Xform “World”
{
   def Xform “Character”
   {
      def Mesh “GirlA”
      {
        ……
        custom double3 xformOp:translate = (4, 5, 6)
        uniform token[] xformOpOrder = ["xformOp:translate"]
      }
   }
}
```

helloWorldShot_sets.usda

```
#usda  1.0
def Xform “World”
{
   def Xform “Sets”
   {
      def Xform“DeskA”
      {…….
      }
   }
}
```

helloWorldShot.usda

```
#usda 1.0
(
subLayers = [
        @helloWorldshot_layout.usd@,
        @helloWorldshot_sets.usd@，
         ]
)
{
        custom double3 xformOp:translate = (1, 1, 1)
        uniform token[] xformOpOrder = ["xformOp:translate"]
}
```

Composition Arcs

- 合成弧相互之间修改值冲突
- 优先级:

  Sublayers > Inherits > Variants > References >Payloads > Specializes
- 优先级高的合成弧覆盖优先级低的合成弧对属性值的改动
- 无损覆盖(non-destructive override)

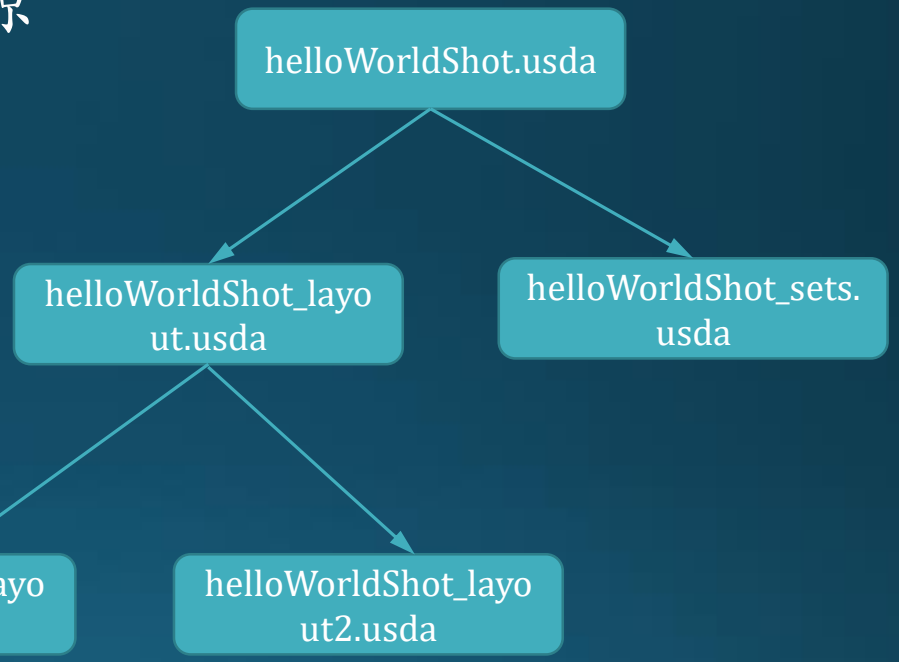
PCP (Prim Cache Population)

- 合成弧算法库，解算合成弧操作的底层库
- 计算“prim index”：对于每一个Prim, 场景描述中各个属性值被改变的位置以及他们相对的优先级
- Prim Index用有方向的节点图表示

PCP(Prim Cache Population)

- 节点代表每一层，属性值改动的来源
- 边代表合成操作，也就是合成弧

UsdPrim::GetPrimIndex
UsdPrim::PcpPrimIndex
PcpCache::ComputePrimIndex
PcpCache::ComputeLayerStack

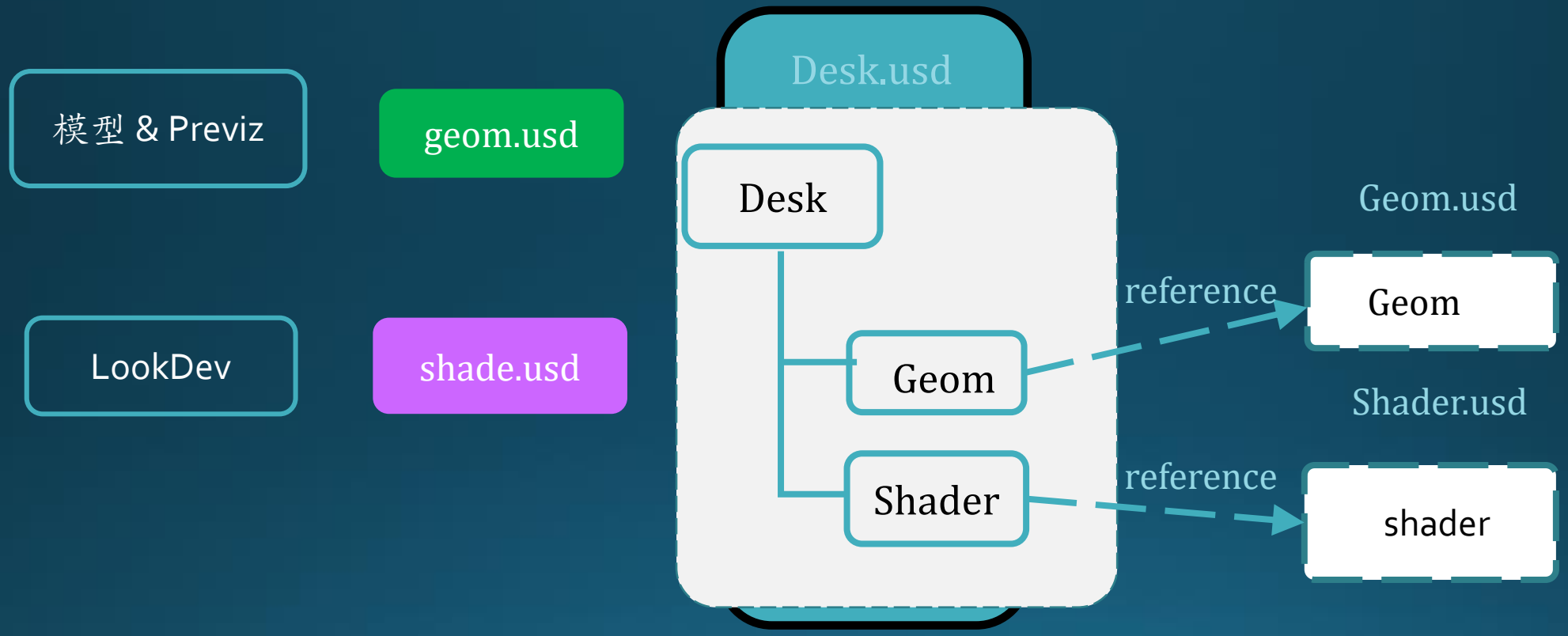



```
>>> primIndex = usdviewApi.prim.GetPrimIndex()
>>> primIndex.DumpToDotGraph('prim_index.dot')
<use graphviz 'dot' command to convert to image>
```

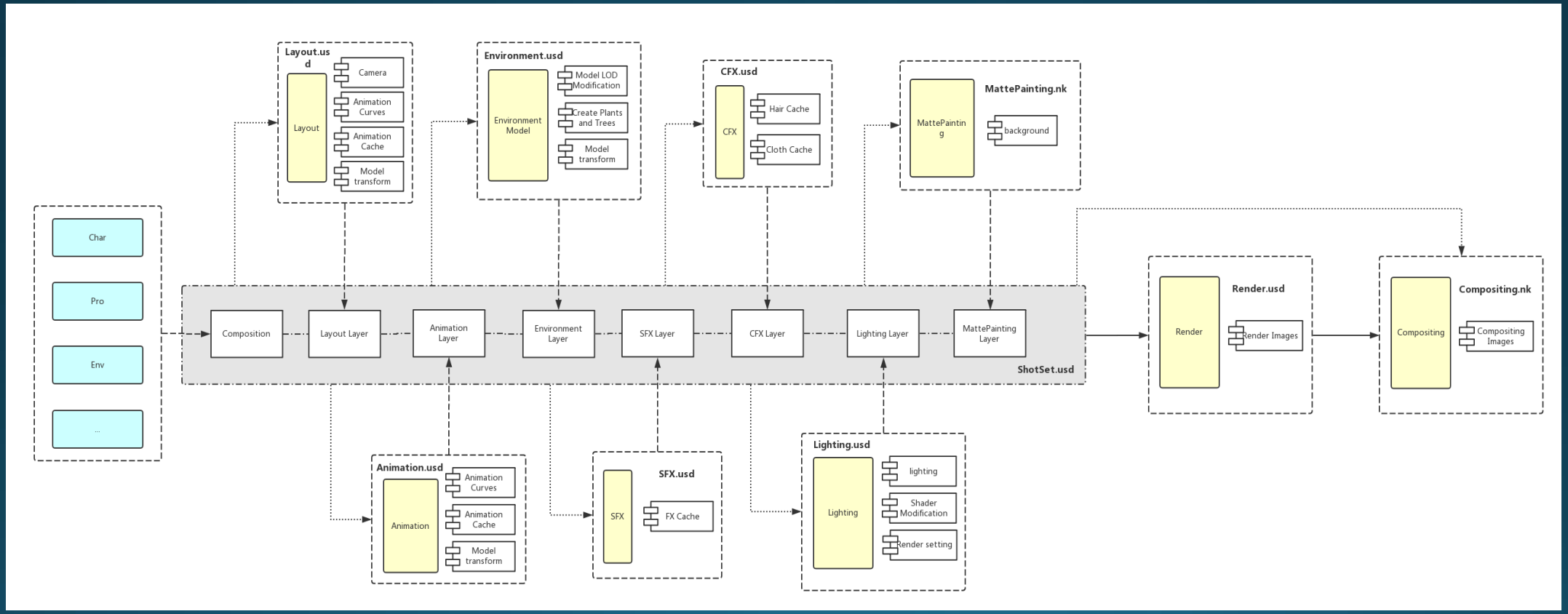
USD 应用场景

- 基于资产的流程
- 基于镜头的流程

基于资产的流程

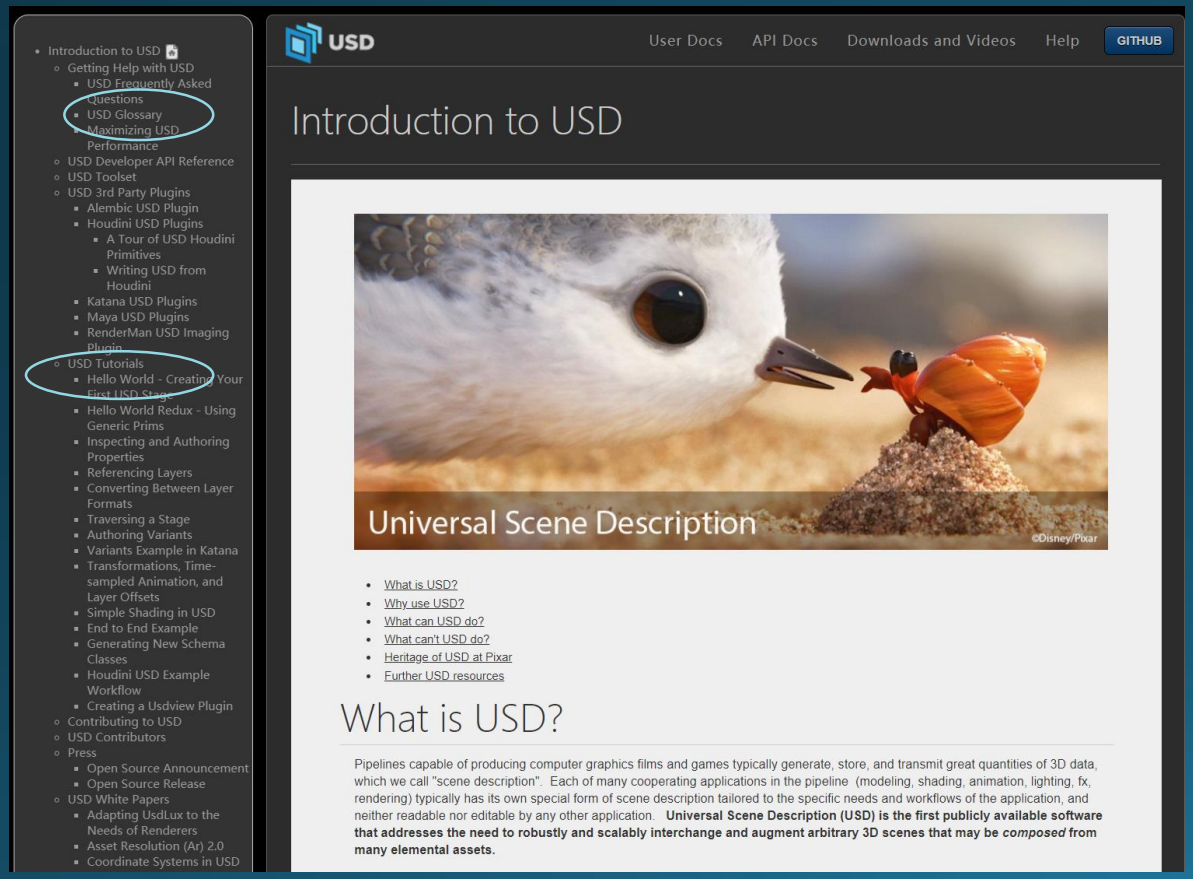

模型 & Previz
geom.usd
LookDev
shade.usd
Desk.usd
Desk
Geom
Shader
reference
Geom.usd
Geom
reference
Shader.usd
shader

基于镜头的流程


Layout.usd
Camera
Animation Curves
Animation Cache
Model transform
Layout
Environment.usd
Model LOD Modification
Create Plants and Trees
Model transform
Environment Model
CFX.usd
Hair Cache
Cloth Cache
CFX
MattePainting.nk
background
MattePainting
Char
Pro
Env
...
Composition
Layout Layer
Animation Layer
Environment Layer
SFX Layer
CFX Layer
Lighting Layer
MattePainting Layer
ShotSet.usd
Render.usd
Render
Render Images
Compositing.nk
Compositing
Compositing Images
Animation.usd
Animation Curves
Animation Cache
Model transform
Animation
SFX.usd
FX Cache
SFX
Lighting.usd
lighting
Shader Modification
Render setting
Lighting

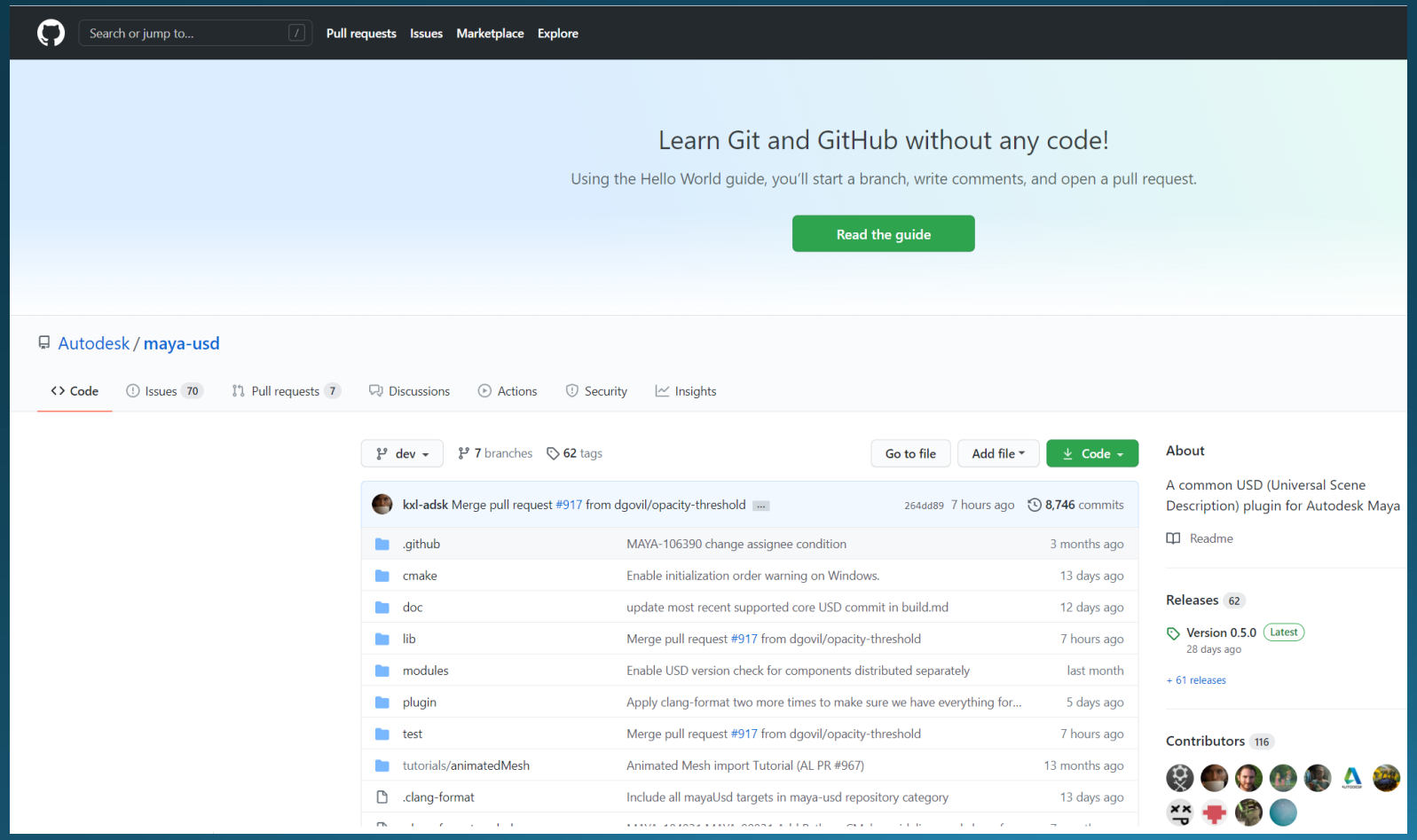
Pixar USD main page

- https://graphics.pixar.com/usd/docs/index.html



Autodesk Maya USD

- https://github.com/Autodesk/maya-usd



USD Demo

- Autodesk Vision Series
  - https://www.youtube.com/watch?v=6EqcwefKMRI
- Nvidia Omniverse
  - https://developer.nvidia.com/nvidia-omniverse-platform