

Papervision 3D

October 23, 2008

Luis Hierro
Stuart Menten
Ransom Christofferson
Swamy Ananthanarayan

Papervision 3D

- Developed to enable 3D graphics with Flash
- Uses ActionScript as the language
- Developed by a core team and released to public in 2007, has since become open source
- Report/presentation is based on 1.5 but 2.0 is in public beta

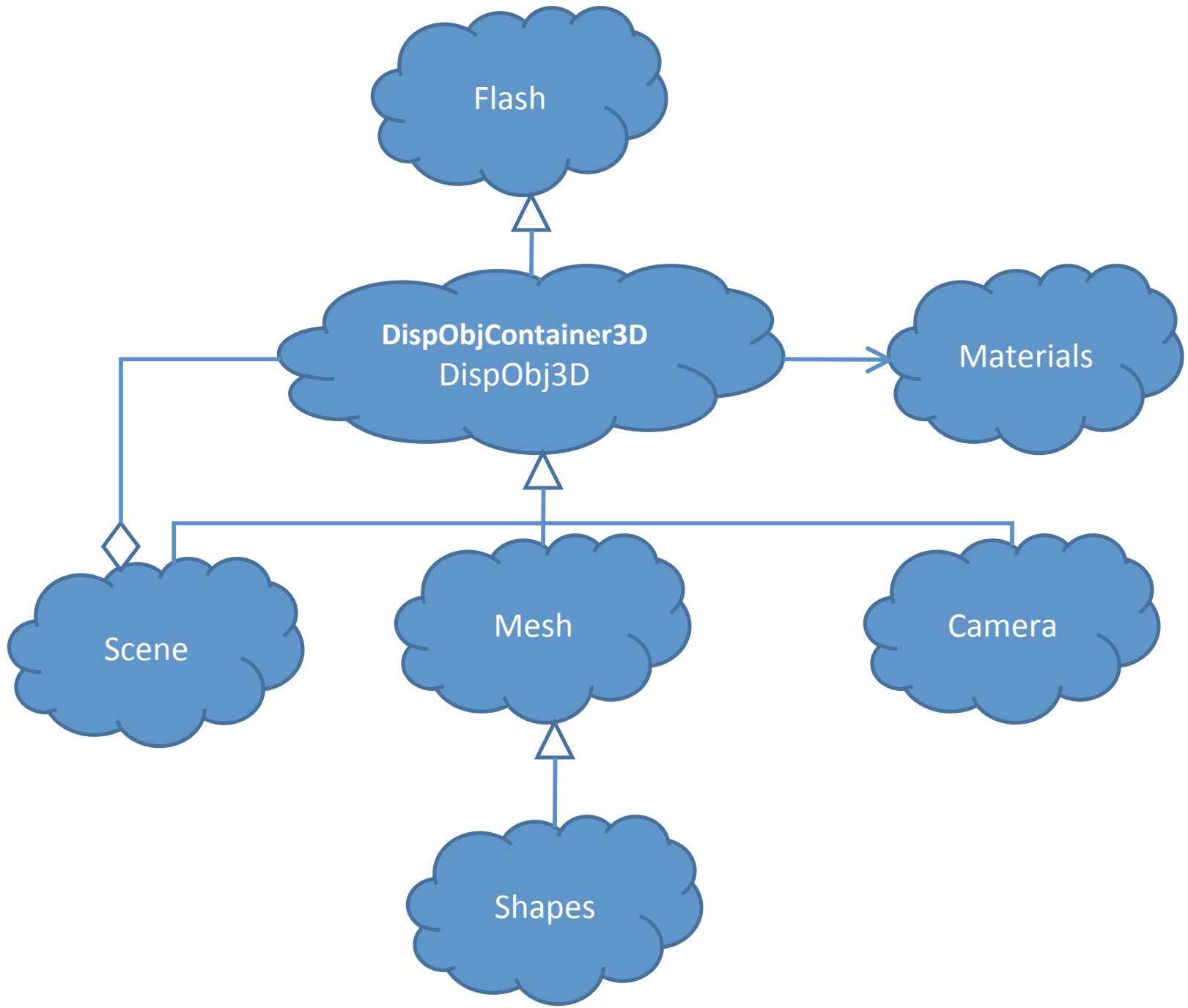
Abilities

- Papervision allows you to make 3D objects, and manipulate them.
- Works by making a scene, having a camera object and then adding any other needed objects to the scene.
- Demo tease

Demo

Flash & Papervision

- Papervision takes care of all calculations and keeps track of all objects in the scene
- Flash presents Papervision's objects and calculations
- Flash handles any I/O

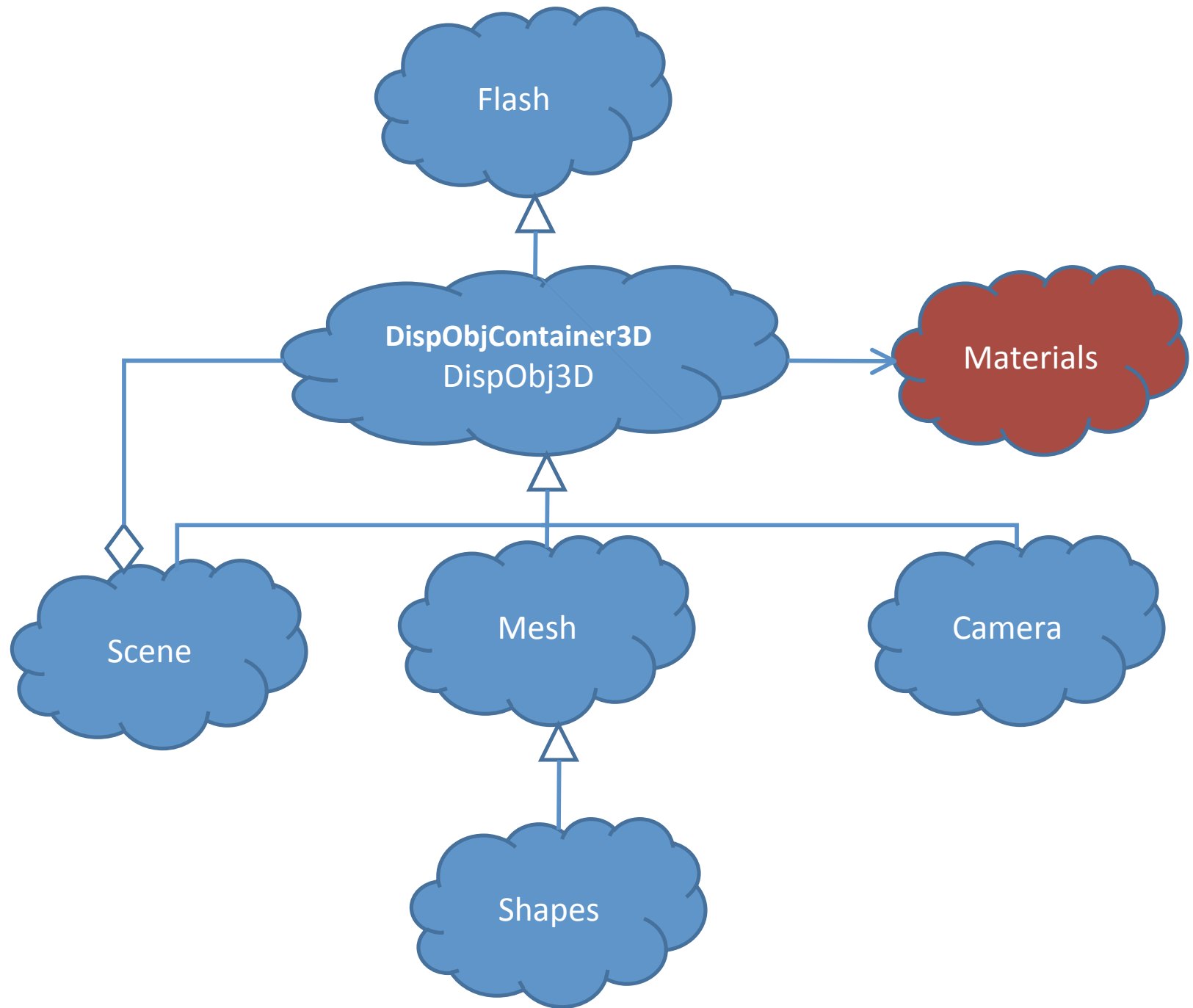


DisplayObject3D

- Directly below flash in our Hierarchy is the DisplayObjectContainer3D (abstract class)
 - Is the base class for anything that can contain 3D objects (scenes, shapes, etc...)
- Its children are displayObject3D and sceneObject3D which are the base for the objects needed to create a “movie”

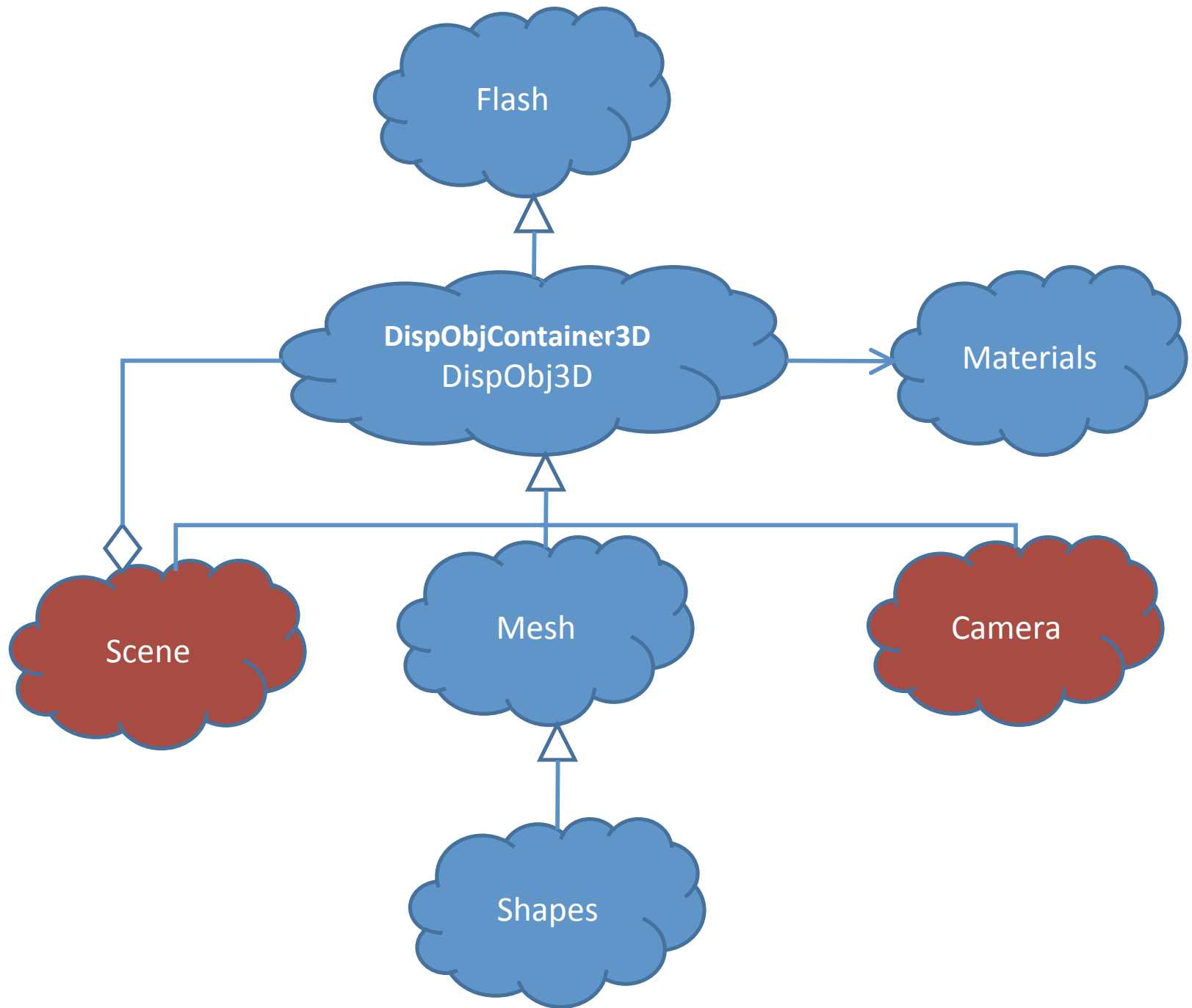
DisplayObject3D Services

- A `displayObjectContainer3D` is a container class for `displayObject3Ds`
 - Think of the space where all the 3D objects “live” (this includes the camera and any shapes you might want)
 - Anything and everything that you see in a Papervision movie is a `displayObject3D`



Materials

- Materials collects data about how objects appear when rendered.
- Materials get assigned to entire objects or faces of an object.
- The materials package provides different ways to create and manage materials
- Ex. Bitmap, color, movie, etc.

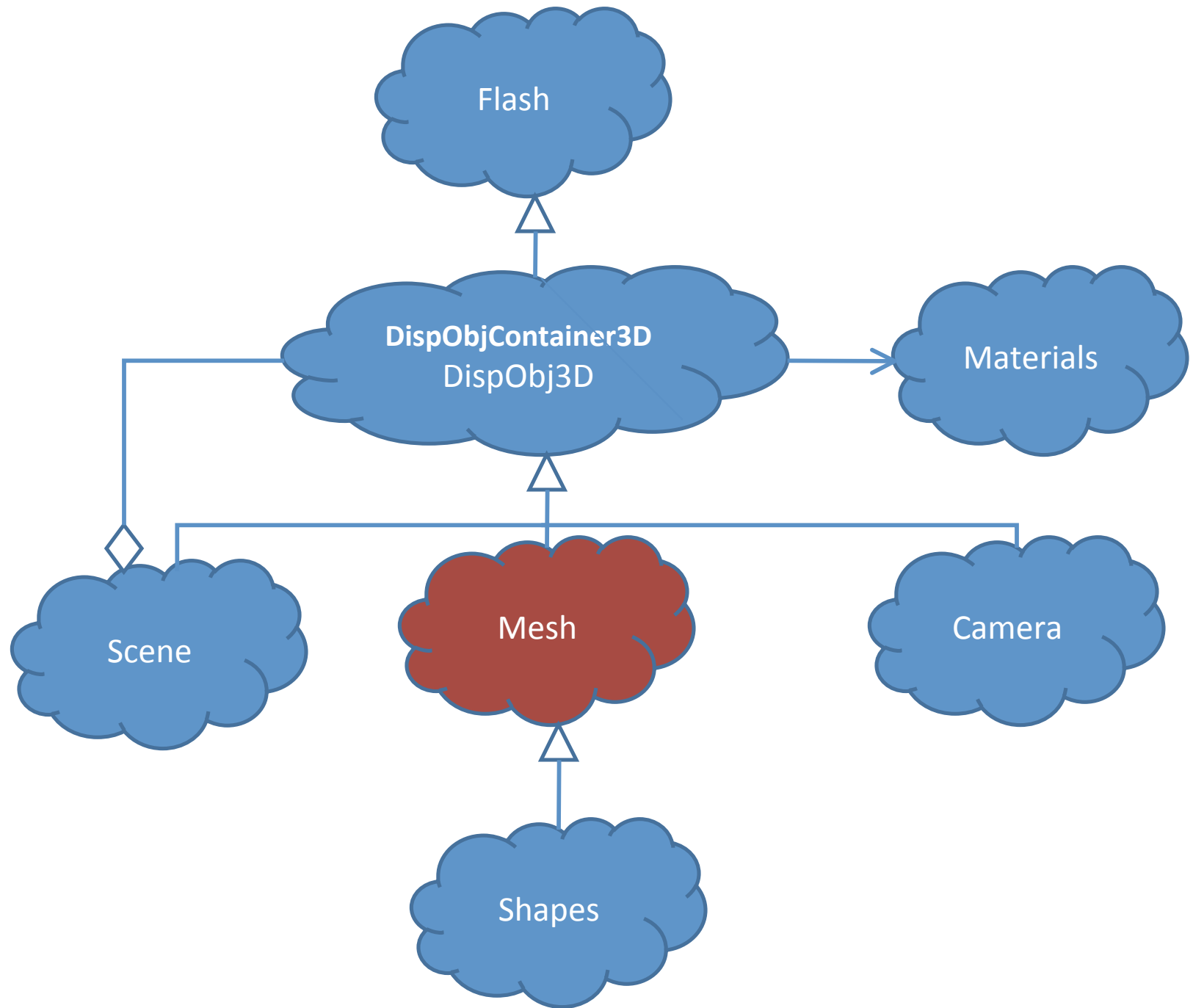


Scenes

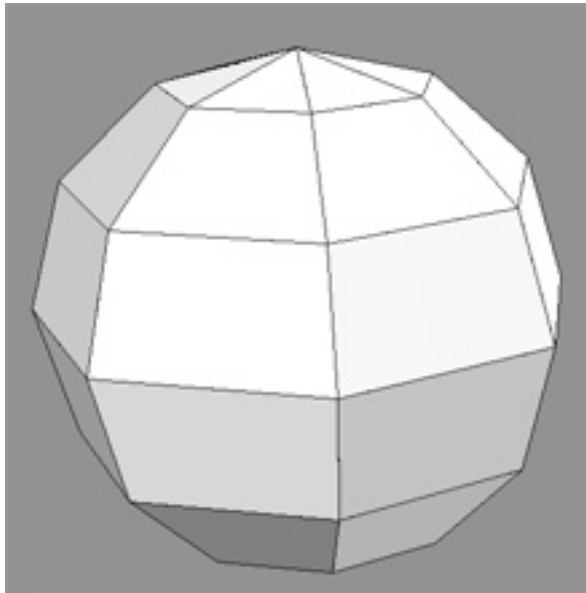
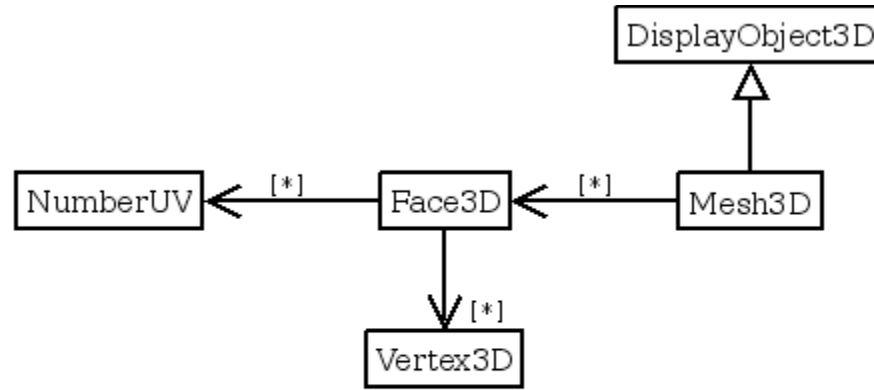
- A scene is the place where objects are placed, it contains the 3D environment.
- [MovieScene3D](#) lets you create a scene where each object is rendered in its own container.
- [Scene3D](#) lets you create a scene where all objects are rendered in the same container.
 - Contains multiple display objects (shapes)

Cameras (passed into scene)

- A camera defines the view from which a scene will be rendered. Different camera settings would present a scene from different points of view.
- The Camera3D class creates a camera that views the area around a target object.
- The FreeCamera3D class creates a camera that views the area in the direction the camera is aimed.

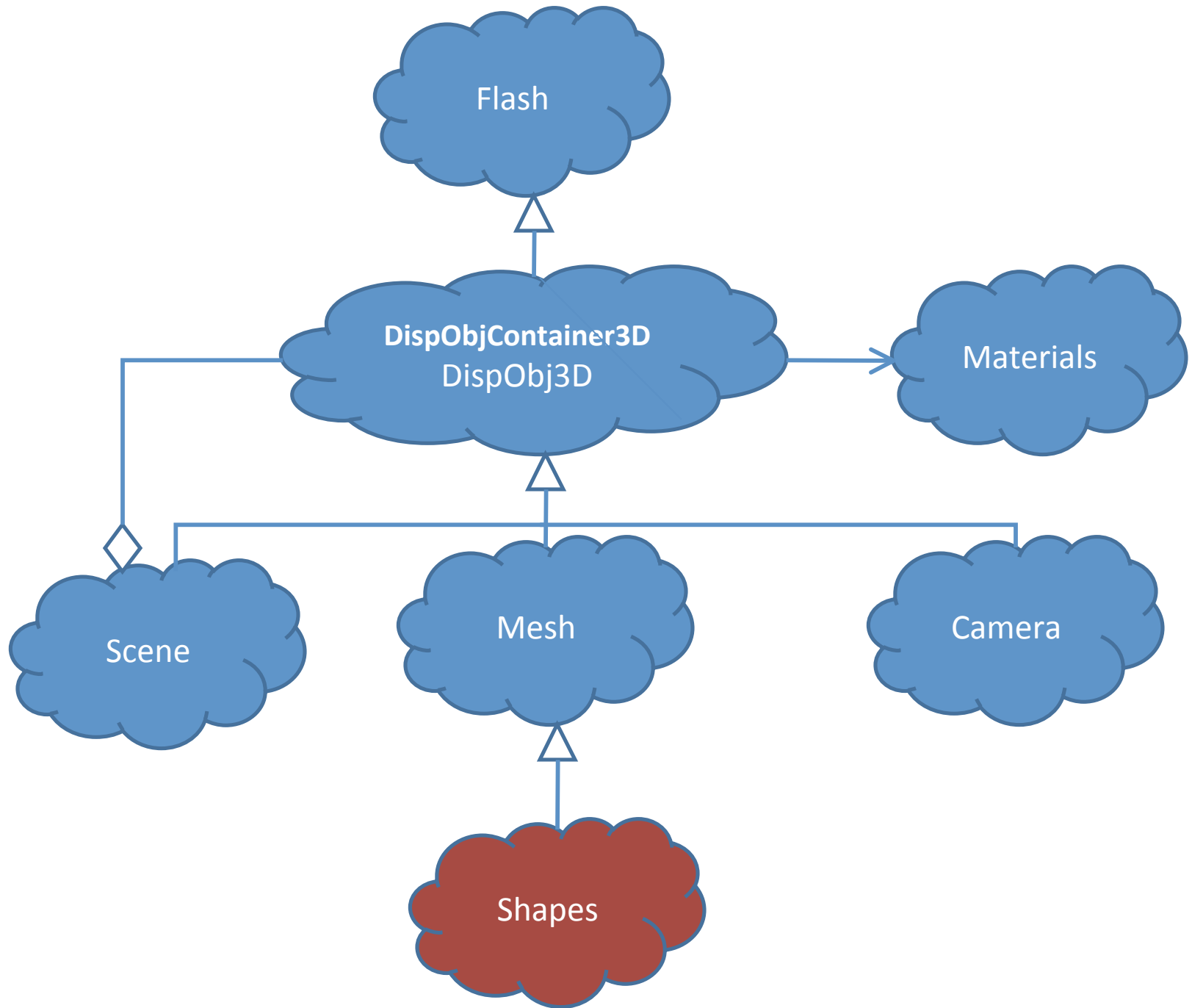


Mesh3D



Services Exported by Mesh3D

- Project the object from 3D to 2D
- Move the object in space
- Add/remove objects composing the 3D shape
- Render the object onto a scene

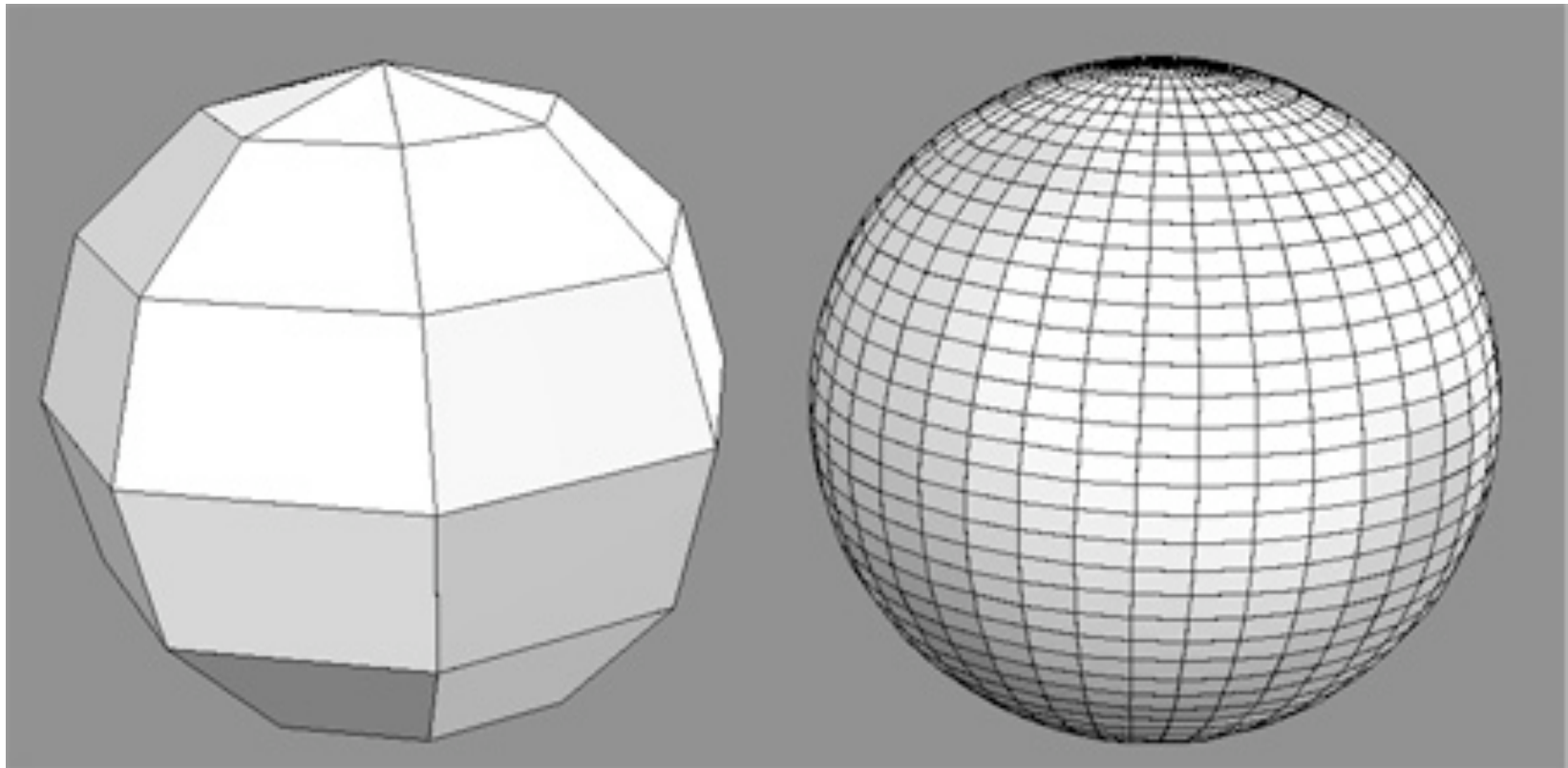


Shapes

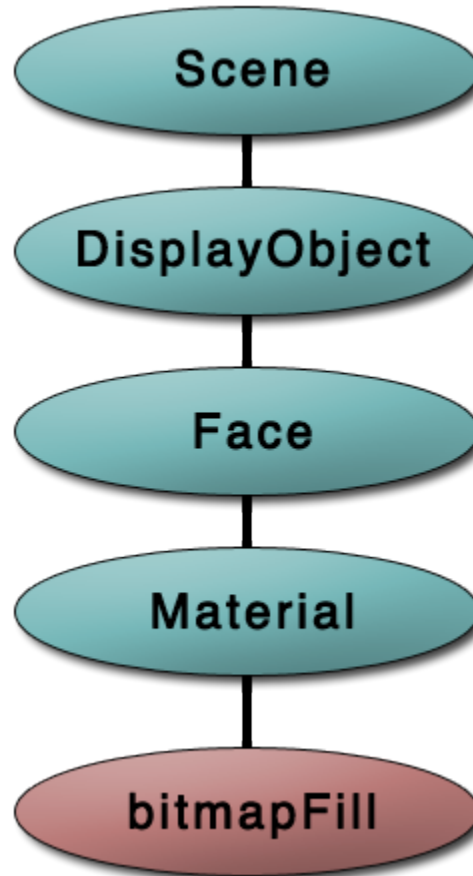
- All shapes inherit from Mesh3D
- The shape classes allow you to easily construct common objects
 - Cylinder
 - Cone
 - Cube
 - Plane
 - Sphere
 - Paper Plane
 - Collada
 - Ase

Shapes

- The shape constructors allow you to define segmentation



Rendering



Getting Papervision to Work

- `#include` papervision source code
- Create a Scene
- Create a Camera
- Create shapes & add them as children to the Scene
- On every frame, call scene's 'render' method and pass it the Camera object

Demo

Critique

- Kick ass documentation
- Excellent use of inheritance and polymorphism
- Easier to use than OpenGL
- Has the advantages and disadvantages of Flash
- Easy to create/extend framework

Why use Papervision?

- Open source with a large community
- Well maintained
- Agile core developers