What is glTF?

- glTF is the *GL Transmission Format*
  - An open standard, developed by Khronos: [khronos.org/gltf](http://khronos.org/gltf)
- Designed for the efficient transfer of 3D assets
  - Versatile, compact, and easy to process by the client
Outline

- The full tutorial is available at [github.com/KhronosGroup/glTF-Tutorials](https://github.com/KhronosGroup/glTF-Tutorials)
- Explains the concepts of glTF, step by step
  - Each one demonstrated with an actual glTF asset
- Geometry, animations, materials, textures, skins...
- Targeting glTF 2.0: Pull request at [github.com/KhronosGroup/glTF/pull/826](https://github.com/KhronosGroup/glTF/pull/826)
Introduction and motivation

3D data sources
E.g. laser scanners

Authoring applications

Blender
Maya
LightWave3D
3DSMAX
...

 Runtime applications

Application 1
Application 2
...
Application N

Graphics APIs

OpenGL
WebGL
OpenGL ES
Vulkan
Microsoft DirectX
Introduction and motivation

3D data sources
E.g. laser scanners

Authoring applications
- Blender
- Maya
- LightWave3D
- 3DSMAX
- ...

Importers and converters
Hundreds of them!

Runtime applications
- Application 1
- Application 2
- ...
- Application N

Graphics APIs
- OpenGL
- WebGL
- OpenGL ES
- Vulkan
- DirectX
Introduction and motivation
Goals of glTF

- Versatile:
  - Plain geometry
  - Complex scenes with animations, materials, ...

- Compact:
  - OBJ : 90MB
  - glTF: 19MB
  - COLLADA: 5.3MB (+1.9MB textures)
  - glTF: 2.2MB (+1.9MB textures)

- Easy to parse
  - Core format based on JSON
  - Geometry data stored in binary form: No decoding overhead!
Users and supporters of glTF: Libraries

three.js

BabylonJS

GrimoireGL

Cesium

xeogl

GLBoost

Test and comparison of loaders and viewers at [github.com/cx20/gltf-test](https://github.com/cx20/gltf-test)
Users and supporters of glTF: Applications

VisCircle

Sketchfab

Archilogic

Cesium

BioDigital

Supporters:
Basic file structure

- External resources can be embedded into JSON, as data URIs

.glTF (JSON) file

```
"scenes": [ ... ],
"nodes": [ ... ],
"cameras": [ ... ],
"animations": [ ... ],
...

"buffers": [
  {
    "uri": "buffer01.bin",
    "byteLength": 102040
  }
],

"images": [
  {
    "uri": "image01.png"
  }
],
```

The JSON part describes the general scene structure, and elements like cameras and animations. Additionally, it contains links to files with binary data and images:

- .bin files
  - Raw data for geometry, animations and skins
- .jpg or .png files
  - Images for the textures of the models
Basic JSON structure
This is a complete glTF asset with an embedded buffer

(Supposed to be the minimal glTF asset)
Scenes and nodes

- Nodes stored in JSON
  - Define a node hierarchy
  - Can have local transforms
- Meshes, cameras, etc. are attached to nodes

```
gltf nodes
"nodes" : [
  {
    "children": [ 1, 2 ]
  },
  {
    "camera": 0,
    "matrix": [ ... ]
  },
  {
    "mesh": 0,
    "children": [ 3, 4 ]
  },
  {
    "mesh": 2,
    "rotation": [ ... ],
    "translation": [ ... ]
  },
  {
    "mesh": 2,
    "rotation": [ ... ]
    "translation": [ ... ]
  }
]
```
Cameras

- Perspective and orthographic cameras, attached to nodes
Buffers, bufferViews, and accessors

- A buffer is stored in an external file, in binary form
- A bufferView defines a part of a buffer
- An accessor defines the data layout of a bufferView

```c
glVertexAttribPointer( ... , bufferView );
```
A simple animation

```
1.8 ... 
```

time accessor

```
0.0 0.0 0.707 0.707 ... 
```

rotation accessor
Meshes

- Consist of mesh primitives that refer to accessors with vertex attribute data

<table>
<thead>
<tr>
<th>positions accessor:</th>
<th>1.0</th>
<th>1.0</th>
<th>-5.0</th>
<th>4.0</th>
<th>3.0</th>
<th>-2.0</th>
<th>8.0</th>
<th>1.8</th>
<th>-2.0</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>normals accessor:</td>
<td>0.0</td>
<td>1.0</td>
<td>0.0</td>
<td>0.0</td>
<td>-1.0</td>
<td>0.0</td>
<td>0.71</td>
<td>0.71</td>
<td>0.0</td>
<td>...</td>
</tr>
<tr>
<td>texCoords accessor:</td>
<td>0.8</td>
<td>0.2</td>
<td>0.6</td>
<td>0.2</td>
<td>0.9</td>
<td>0.1</td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>

- Position: (1.0, 1.0, -5.0)
- Normal: (0.0, 1.0, 0.0)
- TexCoord: (0.8, 0.2)
Materials

- A **material** stores material parameters
  - For example: Metallic-ness and roughness
  - Can also be given as textures
- Physically based rendering (PBR) part of glTF 2.0
  - Coordinated effort to define a standard for PBR!
Techniques (extension)

- An extension for GL-based rendering
- Fast forward:
  - A material is an instance of a technique
  - A technique refers to a program
  - The program refers to GLSL shader objects
Techniques (extension)

- Technique parameters describe shader attributes and uniforms

- Shaders are stored in external (GLSL) files
Materials and Techniques (extension)

- Materials are „instances“ of techniques

Diagram showing the relationship between techniques, their parameters, and instances.

- Technique Parameters:
  - position
  - color
  - lightDirection

- Parameter values:
  - spherePositions
  - vec4(1,0,0,1)
  - vec3(1,-1,-1)

- Instances:

- Material Parameters:
  - boxPositions
  - vec4(0,0,1,1)
  - vec3(-1,1,-1)
Textures

- A texture can be an input for a `sampler2D` uniform variable:

```
Technique
Parameters:
  position
  diffuseTexture
  lightDirection

Texture image:
```

```
Material
Parameter values:
  planePositions
  exampleTexture
  vec3(-1,1,-1)

```

- Reminder: Texture images are stored as external files (JPG, PNG...)

© Copyright Khronos Group 2017 - Page 22
Skins

Skeleton + Mesh = Skinned mesh: Vertex skinning
Skins

- The JOINTS: Which joints affect the vertices
- The WEIGHTS: How strongly the joints affect the vertices
glTF resources

- Landing page: khronos.org/gltf
- GitHub repository: github.com/KhronosGroup/glTF
  - Links to resources, samples and the specification
- Sample models repository: github.com/KhronosGroup/glTF-Sample-Models
  - Simple models for learning, complex models for testing
- Asset validator: github.com/KhronosGroup/glTF-Validator
- Converters:
  - COLLADA: github.com/KhronosGroup/COLLADA2GLTF/
  - OBJ: github.com/AnalyticalGraphicsInc/OBJ2GLTF
  - + many others!
  - Full list at github.com/KhronosGroup/glTF#gltf-tools
- Try it out with the online drag-and-drop converter:
  - cesiumjs.org/convertmodel.html
Getting started with glTF

- Get an overview of the glTF concepts and their relationships:
  - [github.com/KhronosGroup/glTF#overview](https://github.com/KhronosGroup/glTF#overview)
- Explore each concept using the simple test models:
  - [github.com/KhronosGroup/glTF-Sample-Models](https://github.com/KhronosGroup/glTF-Sample-Models)
- Dive deeper into each topic using the tutorials:
  - [github.com/KhronosGroup/glTF-Tutorials](https://github.com/KhronosGroup/glTF-Tutorials)
- Look up the details in the specification:
  - [github.com/KhronosGroup/glTF/tree/master/specification](https://github.com/KhronosGroup/glTF/tree/master/specification)
Writing a glTF loader or viewer

- Have a look at the existing loaders and viewers:
  - [github.com/KhronosGroup/glTF#loaders-and-viewers](https://github.com/KhronosGroup/glTF#loaders-and-viewers)
  - For JavaScript/WebGL, C++, C#, Go, Rust, Haxe, Java...
Contributing to glTF

- Create loaders, exporters, converters or viewers
  - To be listed at github.com/KhronosGroup/glTF
- Contribute sample models
  - To be added to github.com/KhronosGroup/glTF-Sample-Models
- Write tutorials
  - To be published at github.com/KhronosGroup/glTF-Tutorials
- You are already using glTF?
  - Let us know and share your story!
Khronos glTF Webinar: Questions?

- Download the spec, header files, tutorials, and more: [khronos.org/gltf](http://khronos.org/gltf)

- Sign up for the Khronos newsletter: [khronos.org/news/subscribe](http://khronos.org/news/subscribe)

- Learn about becoming a Khronos member: [khronos.org/members](http://khronos.org/members)

- Slides by Marco Hutter [gltf@marco-hutter.de](mailto:gltf@marco-hutter.de)

- Thanks to all glTF contributors! [Neil Trevett and Patrick Cozzi will now join to answer your questions](mailto:neil.trevett@khronos.org, patrick.cozzi@khronos.org)