Introducing:

Peter Gagliardi - Software Developer, 3D Tiles at Cesium
Email: peter@cesium.com, Twitter: @ptrgags
3D Tiles and Web3D, a history

- 2014 - Cesium presents at Web3D about CesiumJS, CZML, glTF
- 2015 - 3D Tiles specification is released
- 2016 - Cesium presents at Web3D about 3D Tiles
- 2019 - 3D Tiles becomes an OGC Community Standard
- **2021 - 3D Tiles Next is announced** - open specifications for 3D Tiles extensions are now available for community feedback
● Open standard for streaming massive heterogeneous 3D geospatial data
  ○ Terrain & imagery, 3D buildings, photogrammetry, point clouds, BIM models, interiors, etc.
  ○ Multiple source data types, one runtime format

● Visualization + analysis

● Combine:
  ○ Flexible spatial data structure in JSON
  ○ “Runtime ready” binary tile formats
  ○ Khronos glTF open standard for 3D models
  ○ Vertex/polygon-level metadata
Draft extensions for 3D Tiles
Open for community feedback
Goals:
- Cleaner integration with the glTF ecosystem
- Enrich data with more robust and efficient semantic metadata
- Implicit tiling: compact, sparse quadtrees and octrees for massive simulations and analytics

https://github.com/CesiumGS/3d-tiles/tree/main/next
3D Tiles and the glTF ecosystem

- 3DTILES_content_gltf extension allows using glTF files directly in a tileset
- Tile formats moving to glTF + extensions
  - Remove abstraction layers (b3dm, i3dm, pnts, cmpt)
  - Common format for meshes, point clouds, potentially vector data
- Leverage more glTF tools & extensions
  - Compression: KTX 2.0, MESHOPT
  - GPU instancing
  - More physically-based materials: specular, clearcoat, sheen, etc.

### 3D Tiles 1.0
.b3dm file
- b3dm Header
- Batch Table
- glTF

### 3D Tiles Next
.glb file + extension
- glTF
- EXT_mesh_features
Metadata
Augment real-world data with semantics from AI and machine learning

Metadata at many granularities

Decoupled design: semantics, type

Metadata

3DTILES_metadata

EXT_mesh_features (gltF extension)
**Per-texel metadata**

*Property textures* for fine grained surface properties

Data source: Maxar

*Feature ID textures* for feature identification

Data source: Aerometrex

---

```
{
    "schema": {
        "class": {
            "properties": {
                "type": "STRING"
            },
            "component": {
                "type": "STRING"
            },
            "attribute": {
                "type": "STRING"
            }
        }
    },
    "featureTable": {
        "labels": {
            "class": "label",
            "count": 5,
            "properties": {
                "color": {
                    "bufferView": 1,
                    "stringOffsetBufferView": 2
                },
                "component": {
                    "bufferView": 3,
                    "stringOffsetBufferView": 4
                },
                "attribute": {
                    "bufferView": 5
                }
            }
        }
    }
}
```
Demo: Photogrammetry Classification

Data: Aerometrex  
https://demos.cesium.com/ferry-building
Implicit Tiling
Implicit tiling

- Efficient binary representation of quadtrees/octrees
- Tile and content availability stored in bitstreams
- Tiles located directly by (level, x, y, [z])
- Random access provides
  - Accelerated spatial queries
  - Efficient traversal at runtime
  - Efficient partial updates for changing scenes
- S2 bounding volumes avoid distortion at the poles
Demo: S2 Base Globe

Data: Maxar

https://demos.cesium.com/owt-globe
Comparing Explicit and Implicit Tiling

Explicit Tiling: Provides flexibility of tileset structure

Every bounding volume is described explicitly

Implicit Tiling: Provides random access and efficient storage for quadtrees/octrees

Root Bounding volume is implicitly subdivided for each descendant tile

Template URI patterns are listed once

Subtree files contain availability bitstreams
What’s Next
What’s next

- Revise specifications from community feedback
- Continue development on experimental CesiumJS implementation
- Bring 3D Tiles Next to Unreal and other engines
- Create more sample data and tooling
- Talk with the Open Geospatial Consortium (OGC) and the Khronos Group about the standardization process
How to get involved

- Specifications are open for community feedback
  - 3D Tiles Next on GitHub: https://github.com/CesiumGS/3d-tiles/tree/main/next

- Try out the experimental implementation in CesiumJS
  - CesiumJS CHANGES.md

Thanks to Don McCurdy, Marco Hutter, Erik Dahlström, Johan Borg, Patrick Cozzi, Sam Suhag, Sean Lilley for their contributions to 3D Tiles Next