donmccurdy /
glTF-Transform

Code   Issues   46   Pull requests   7   Discussions   Actions   Projects   Security   Insights

glTF 2.0 SDK for JavaScript and TypeScript, on Web and Node.js.

🔗 **gltf-transform.dev**

⚖️ MIT license

Code of conduct

⭐ **1.2k** stars   **135** forks   👁 **28** watching   **18** Branches   **192** Tags   Activity

🌐 Public repository

⑂ main ▾   ⑂ **18** Branches   ◌ **192** Tags   ⑂   ◌   🔍 Go to file   t   Go to file   +   Add file ▾   Code   ⋯

🙂 **donmccurdy** v4.0.0-alpha.10 ✓	15 hours ago	⋯ 🕚
📁 .github	chore(deps): update codecov/codecov-ac...	last month
📁 benchmarks	chore(functions): Add benchmarks for qu...	last week
📁 packages	v4.0.0-alpha.10	15 hours ago
📁 scripts	chore(scripts): Move test/ to scripts/ (#12...	2 months ago
📄 .editorconfig	Update .editorconfig	4 years ago
📄 .eslintrc.json	Add testing docs to contributing.md	last month
📄 .gitignore	Migrate tests to Ava (#799)	last year
📄 .prettierignore	Documentation: Rebuild documentation ...	10 months ago
📄 .prettierrc.json	Documentation: Rebuild documentation ...	10 months ago
📄 .vercelignore	Documentation: Rebuild documentation ...	10 months ago
📄 CHANGELOG.md	Require Node.js v18+ (#1209)	3 months ago
📄 LICENSE	Clean up documentation for v3 (#781)	last year
📄 README.md	chore(docs): Fixes for docs	2 days ago
📄 lerna.json	v4.0.0-alpha.10	15 hours ago
📄 package.json	fix(core,extensions): Extension order inde...	yesterday
📄 tsconfig.json	chore(ts): Set module=nodenext (#1248)	2 months ago
📄 yarn.lock	chore(deps): update devdependencies (n...	2 days ago

📖 README   🏷 Code of conduct   ⚖️ MIT license   ✏️ ☰

glTF Transform

`npm v3.10.0` `minzipped size 20.3 kB` `license MIT` 🔘 `build passing` 🔀 `codecov 94%`

glTF 2.0 SDK for JavaScript and TypeScript, on Web and Node.js.

Introduction

glTF Transform supports reading, editing, and writing 3D models in glTF 2.0 format. Unlike 3D modeling tools — which are ideal for artistic changes to geometry, materials, and animation — glTF Transform provides fast, reproducible, and lossless control of the low-level details in a 3D model. The API automatically manages array indices and byte offsets, which would otherwise require careful management when editing files. These traits make it a good choice for bundling, splitting, or optimizing an existing model. It can also be used to apply quick fixes for common issues, to build a model procedurally, or to easily develop custom extensions on top of the glTF format. Because the core SDK is compatible with both Node.js and Web, glTF Transform may be used to develop offline workflows and web applications alike.

Packages:

- `@gltf-transform/core` : Core SDK, providing an expressive API to read, edit, and write glTF files.
- `@gltf-transform/extensions` : Extensions (optional glTF features) for the Core SDK.

- `@gltf-transform/functions` : Functions for common glTF modifications, written using the core API.
- `@gltf-transform/cli` : Command-line interface (CLI) to apply functions to glTF files quickly or in batch.
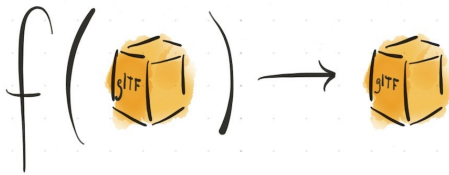
Commercial Use

Using glTF Transform for a personal project? That's great! Sponsorship is neither expected nor required. Feel free to share screenshots if you've made something you're excited about — I enjoy seeing those!

Using glTF Transform in for-profit work? That's wonderful! Your support is important to keep glTF Transform maintained, independent, and open source under MIT License. Please consider a subscription or GitHub sponsorship.

Learn more in the glTF Transform Pro FAQs.

Scripting API

Install the scripting packages:

```
npm install --save @gltf-transform/core @gltf-transform/extensions @gltf-transform/functions
```

Read and write glTF scenes with platform I/O utilities WebIO, NodeIO, or DenoIO:

```
import { Document, NodeIO } from '@gltf-transform/core';
import { ALL_EXTENSIONS } from '@gltf-transform/extensions';
import draco3d from 'draco3dgltf';

// Configure I/O.
const io = new NodeIO()
    .registerExtensions(ALL_EXTENSIONS)
    .registerDependencies({
        'draco3d.decoder': await draco3d.createDecoderModule(), // Optional.
        'draco3d.encoder': await draco3d.createEncoderModule(), // Optional.
    });

// Read from URL.
const document = await io.read('path/to/model.glb');

// Write to byte array (Uint8Array).
const glb = await io.writeBinary(document);
```

To perform changes to an existing glTF Document, import off-the-shelf scripts from the Functions package, or write your own using API classes like Material, Primitive, and Texture.

```
import { resample, prune, dedup, draco, textureCompress } from '@gltf-transform/functions';
import sharp from 'sharp'; // Node.js only.

await document.transform(
    // Losslessly resample animation frames.
    resample(),
    // Remove unused nodes, textures, or other data.
    prune(),
    // Remove duplicate vertex or texture data, if any.
    dedup(),
    // Compress mesh geometry with Draco.
    draco(),
```

```
        // Convert textures to WebP (Requires glTF Transform v3 and Node.js).
        textureCompress({
            encoder: sharp,
            targetFormat: 'webp',
            resize: [1024, 2024],
        }),
        // Custom transform.
        backfaceCulling({cull: true}),
    );

    // Custom transform: enable/disable backface culling.
    function backfaceCulling(options) {
        return (document) => {
            for (const material of document.getRoot().listMaterials()) {
                material.setDoubleSided(!options.cull);
            }
        };
    }
```

To learn how glTF Transform works, and the architecture of the scripting API, start with Concepts. To try out the scripting API without installing anything, visit gltf.report/, load a glTF model, and open the *Script* tab.

Command-line API

Install the CLI, supported in Node.js LTS versions.

```
npm install --global @gltf-transform/cli
```

List available CLI commands:

```
gltf-transform --help
```

Optimize everything all at once:

```
gltf-transform optimize input.glb output.glb --texture-compress webp
```

Or pick and choose your optimizations, building a custom pipeline.

Compress mesh geometry with Draco or Meshoptimizer:

```
# Draco (compresses geometry).
gltf-transform draco input.glb output.glb --method edgebreaker

# Meshopt (compresses geometry, morph targets, and keyframe animation).
gltf-transform meshopt input.glb output.glb --level medium
```

Resize and compress textures with Sharp, or improve VRAM usage and performance with KTX2 and Basis Universal:

```
# Resize textures.
gltf-transform resize input.glb output.glb --width 1024 --height 1024

# Compress textures with WebP.
gltf-transform webp input.glb output.glb --slots "baseColor"

# Compress textures with KTX2 + Basis Universal codecs, UASTC and ETC1S.
gltf-transform uastc input.glb output1.glb \
    --slots "{normalTexture,occlusionTexture,metallicRoughnessTexture}" \
    --level 4 --rdo --rdo-lambda 4 --zstd 18 --verbose
gltf-transform etc1s output1.glb output2.glb --quality 255 --verbose
```

… and much more.

Credits

See *Credits*.

License

Copyright 2023, MIT License.

Releases

🏷 **192** tags

Sponsor this project
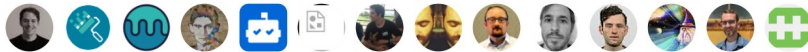
👤 **donmccurdy** Don McCurdy

♡ Sponsor

Learn more about GitHub Sponsors

Used by 1.9k

+ 1,844

Contributors 29

+ 15 contributors

Languages

● **TypeScript** 98.9%   ○ **Other** 1.1%