3DSS-API 技術ドキュメント

3D Shape Share 共通データAPI

Base URL:

```
https://us-central1-shapeshare3d.cloudfunctions.net/api
```

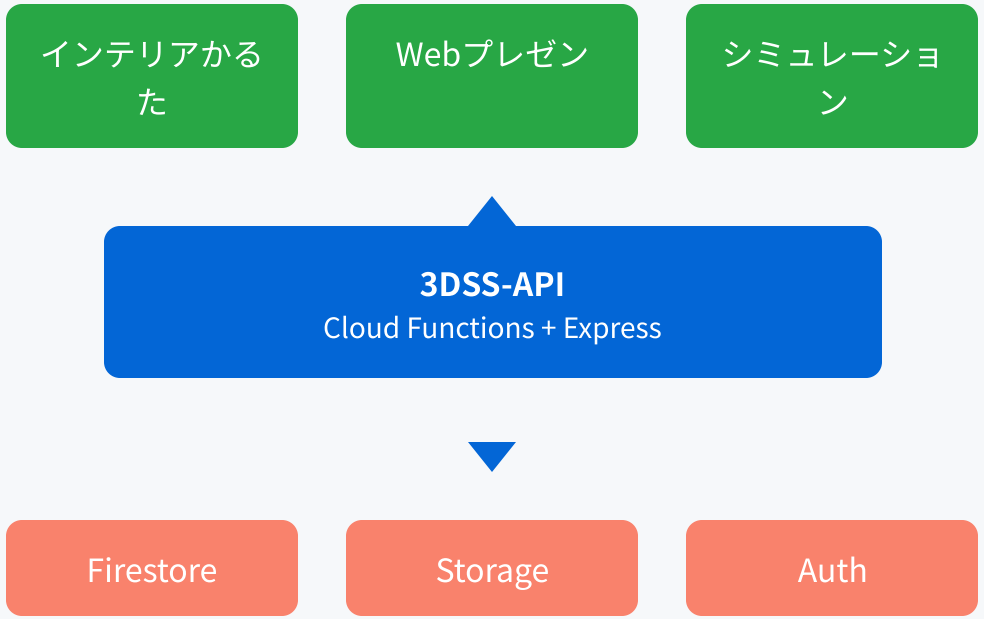
構成情報:

Firebase プロジェクト: shapeshare3d
Cloud Functions エントリ: api（Express を内包）

> **役割:**
> Firestore に保存された公開3Dモデルのメタ情報を JSON で提供。
> 将来的に users / boards / articles / analytics を拡張。

目次

3DSS-APIの全体像（何者か？）



- **プロジェクト名:** 036-3dss-api（ローカルのAPI専用プロジェクト）
- **Firebase プロジェクト:** shapeshare3d（本番の3D Shape Shareと同じ）
- **Cloud Functions のエントリ:** api
- **主要ルート:**

  `GET /models/v1` - モデル一覧取得

  `GET /models/v1/:id` - 単一モデル取得

役割:

3D Shape Share の Firestore に保存されているモデル情報を JSON で返す「窓口」として機能

将来的には `/users/v1`, `/boards/v1`, `/articles/v1` なども追加予定

プラットフォーム層とアプリ層の関係

`https://us-central1-shapeshare3d.cloudfunctions.net/api/...`

- **プラットフォーム層**（3DSS-API）

  3Dモデル、ユーザー、チーム、ボード、共通タグなど

  すべてのアプリで共有されるデータリソース

- **アプリ層**（インテリアかるた / Webプレゼン / シミュレーション）

  それぞれ独自の Firestore / DB を持ってOK

  ただし「3Dモデル」など共通で使いたい情報は 3DSS-API 経由で取得

- **原則:** 各アプリは ID のみを保持

  modelId / userId / boardId 等のIDを参照として保存

  詳細データは API から動的に取得

メリット:

どのアプリからでも Base URL を叩けば、同じ共通3Dデータにアクセス可能

データの整合性が保たれ、アプリ間でのデータ連携が容易に

現在のAPI仕様 - エンドポイント一覧

Base URL

`https://us-central1-shapeshare3d.cloudfunctions.net/api`

GET `/models/v1?limit=...`

モデル一覧（学習/デモ用）を取得します。

クエリパラメータ: `limit` - 返す件数（任意。デフォルト10）

GET `/models/v1/:id`

単一モデルの詳細情報を取得します。

パスパラメータ: `id` - モデルのID（必須）

レスポンス要約

- `id` モデルの一意識別子
- `slug` URL用識別子
- `title` モデル名
- `thumbnailUrl` サムネイルURL
- `category` カテゴリパス
- `modelFormats[]` 提供フォーマット
- `dimensions` サイズ情報
- `tags[]` 関連タグ

`200` 正常にデータ取得　`404` モデルが見つからない場合

API仕様詳細 - モデル一覧取得

GET `/models/v1?limit=5`

モデル一覧を取得するためのエンドポイントです。学習やデモ用として使用します。

クエリパラメータ：

`limit` - 返す件数（任意。デフォルトは10）

レスポンス例 200 OK


```
{
    "items": [
        {
          "id": "0052b495-4cf6-42be-b597-ea83340a86c9",
          "slug": "0052b495-4cf6-42be-b597-ea83340a86c9",
          "title": "ハンギングポット",
          "thumbnailUrl": "https://firebasestorage.googleapis.com/...",
          "category": "lighting/pendant",
          "modelFormats": [],
          "dimensions": null,
          "tags": []
        },
        {
          "id": "43547c5c-d73a-47d8-890d-73ef8a54e07d",
          "slug": "43547c5c-d73a-47d8-890d-73ef8a54e07d",
          "title": "BlueUnicorn",
          "thumbnailUrl": "https://firebasestorage.googleapis.com/...",
          "category": null,
          "modelFormats": [],
          "dimensions": null
```


ℹ レスポンスは`items`配列内のオブジェクトとして返され、ページネーションや検索機能は今後拡張予定です。

API仕様詳細 - 単一モデル取得

GET `/models/v1/:id`

IDを指定して単一の3Dモデル情報を取得するためのエンドポイントです。

例：
`/models/v1/0052b495-4cf6-42be-b597-ea83340a86c9`

成功時のレスポンス例 200 OK


```
{
    "id": "0052b495-4cf6-42be-b597-ea83340a86c9",
    "title": "ハンギングポット",
    "thumbnailUrl": "https://firebasestorage.googleapis.com/...",
    "category": "lighting/pendant",
    "modelFormats": [],
    "dimensions": null,
```


存在しない場合のレスポンス例 404 Not Found


```
{
    "error": "Model not found"
}
```


ℹ インテリアかるたアプリでは、このエンドポイントを使用して3Dモデルのメタデータを取得しています。

基本の使い方（fetch）

ブラウザまたはNode.jsでの基本的な使い方です。標準のfetch APIを使用した実装例を示します。

基本実装 JavaScript/TypeScript

```
const BASE_URL =
    import.meta.env.VITE_3DSS_API_BASE_URL ||
    "https://us-central1-shapeshare3d.cloudfunctions.net/api";
// モデル一覧取得
async function fetchModels({ limit = 10 } = {}) {
    const res = await fetch(`${BASE_URL}/models/v1?limit=${limit}`);
    if (!res.ok) {
        throw new Error(`Failed: ${res.status} ${res.statusText}`);
    }
    return await res.json(); // { items: [...] }
}
// 単一モデル取得
async function fetchModelById(id) {
    const res = await fetch(`${BASE_URL}/models/v1/${encodeURIComponent(id)}`);
    if (!res.ok) {
        throw new Error(`Failed: ${res.status} ${res.statusText}`);
    }
    return await res.json(); // { id, title, ... }
}
```

環境変数 `VITE_3DSS_API_BASE_URL` を設定すると、開発環境やテスト環境など異なる環境でAPIのベースURLを切り替えることができます。

使用例：`fetchModels({ limit: 20 }).then(data => console.log(data.items))` でモデル一覧を取得できます。

React/Vite実装例（共通クライアント）

FILE `src/threeDssApi.js`

どのReactアプリでも共通で使える3DSS-APIクライアントモジュール

共通クライアント実装 threeDssApi.js

```
export const THREE_DSS_API_BASE_URL =
  import.meta.env.VITE_3DSS_API_BASE_URL ||
  "https://us-central1-shapeshare3d.cloudfunctions.net/api";
export async function fetchModelById(modelId) {
  if (!modelId) throw new Error("modelId is required");
  const url = `${THREE_DSS_API_BASE_URL}/models/v1/${encodeURIComponent(modelId)}`;
  const res = await fetch(url);
  if (!res.ok) {
    const text = await res.text().catch(() => "");
    throw new Error(
      `Failed to fetch model ${modelId}: ${res.status} ${res.statusText} ${text}`
    );
  }
  return res.json();
}
export async function fetchModels({ limit = 10 } = {}) {
  const url = `${THREE_DSS_API_BASE_URL}/models/v1?limit=${limit}`;
  const res = await fetch(url);
```

利用例（React） ModelList.jsx

```
import { useState, useEffect } from 'react';
import { fetchModels, fetchModelById } from './threeDssApi';
export default function ModelList() {
  const [items, setItems] = useState([]);
  const [loading, setLoading] = useState(true);
  useEffect(() => {
    fetchModels({ limit: 20 })
      .then(data => setItems(data.items))
      .catch(err => console.error(err))
      .finally(() => setLoading(false));
  }, []);
  return (
    {loading ? '読み込み中...' : `${items.length}個のモデルを表示中`}
  );
}
```

環境変数 `VITE_3DSS_API_BASE_URL` を.envファイルで設定することで、開発環境や本番環境で異なるAPIエンドポイントを指定できます。

CORS とセキュリティ設定

現状のアプローチ 現在の設定

- ✓ 公開モデルを誰でも読めることを目的としています
- ✓ Express 側で CORS を全許可設定（`origin: true`）
- ✓ Firestore セキュリティルールで「public モデルだけ read を許可」

Express 設定例

```
const functions = require("firebase-functions/v2/https");
const express = require("express");
const cors = require("cors");
const app = express();

// CORS を全世界許可（練習用）
app.use(cors({ origin: true }));
app.use(express.json());

// ここに /models/v1 などのルートを定義
// ...

exports.api = functions.onRequest(app);
```

将来の強化ポイント 計画中

- → CORS の `origin` をホワイトリスト方式に変更
  例: `["https://3dshapeshare.com", "https://your-other-app.com"]`
- → 読み取り以外の操作に認証を要求
  ・Firebase Auth トークン または
  ・独自の API キー
- → お気に入り登録・ボード保存・削除などの書き込み系操作は厳格に保護

新規アプリ開発の作法（テンプレ）

1 クライアントモジュールを置く

各アプリにAPI呼び出し用のモジュールを配置します：

`threeDssApi.js`（JS版）または `threeDssApi.ts`（TS版）

> このモジュールがSDKの役割を果たし、全アプリで統一した呼び出し方ができます

2 環境変数で BASE_URL を設定

フレームワーク別の環境変数設定：

```
# Vite / React (.env.local)
VITE_3DSS_API_BASE_URL="https://us-central1-shapeshare3d.cloudfunctions.net/api"
# Next.js (.env.local)
NEXT_PUBLIC_3DSS_API_BASE_URL="https://us-central1-shapeshare3d.cloudfunctions.net/api"
# Node バックエンド (.env)
THREE_DSS_API_BASE_URL=https://us-central1-shapeshare3d.cloudfunctions.net/api
```

3 アプリ固有のデータと modelId の紐付け

アプリのエンティティに 3DSS モデル ID を参照フィールドとして持たせます：

```
{
id: "ne-nekko",
syllable: "ね",
keyword: "根っこのコンセプト",
modelId: "0052b495-4cf6-42be-b597-ea83340a86c9", // ← 3DSS モデルとの紐づけ
// その他アプリ固有のフィールド
}
```

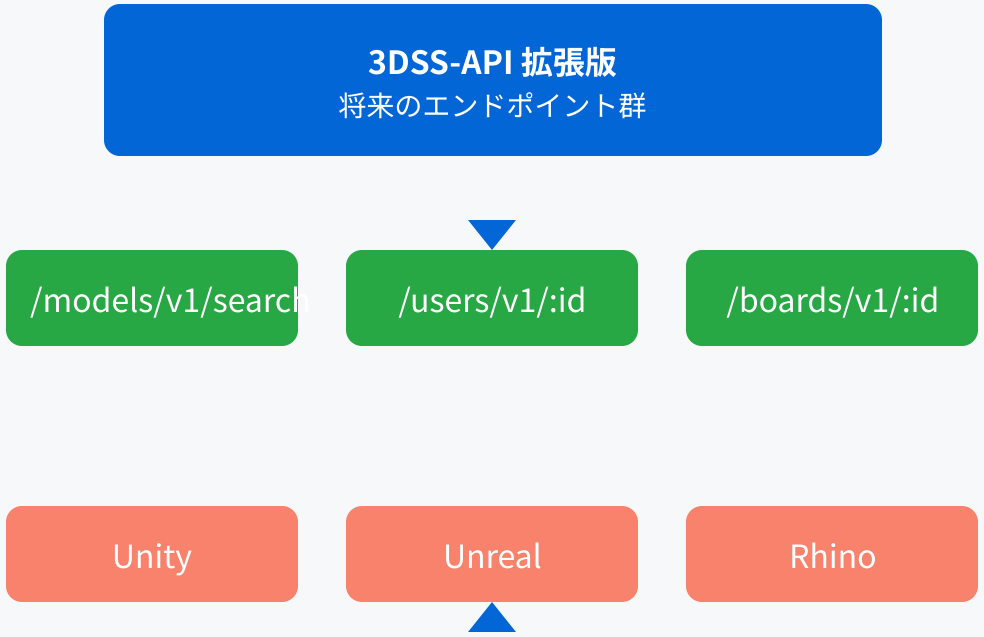
4 画面で必要なときに API を呼び出す

インポートして使用：

```
import { fetchModelById, fetchModels } from "./threeDssApi";
// 一覧表示画面などで
useEffect(() => {
fetchModels({ limit: 20 }).then((data) => setItems(data.items));
}, []);
// 詳細画面では
useEffect(() => {
if (modelId) {
fetchModelById(modelId).then((model) => setModelData(model));
}
}, [modelId]);
```

これでどのアプリからも統一的なインターフェースで 3D モデルデータにアクセスできます。

今後の拡張計画



- **エンドポイント拡張:**

  `/models/v1/search` - キーワード・タグ・カテゴリで検索

  `/models/v1/:id/files` - glb, 3dm等のダウンロードURLやメタデータ

  `/users/v1/:id` - ユーザーの公開プロフィール

  `/boards/v1/:id` - 公開ボードのサマリ

- **運用ルール:**

  共通データはぜんぶ「3DSS-API 側」に集める

  各アプリは「modelId / userId / boardId を持つだけ」で詳細はAPIから取得

次のステップ候補:

`/models/v1/search` の追加 - カテゴリ/キーワード検索API

`/boards/v1/:id` の作成 - ボード単位でモデル一覧を取得する機能