readme.md for @apiclient.xyz/zitadel

An unofficial client for interacting with Zitadel API.

Install

To get started with `@apiclient.xyz/zitadel`, ensure you have Node.js installed on your machine. Then, you can add this package to your project by running:

```
npm install @apiclient.xyz/zitadel --save
```

This module is written in TypeScript to take advantage of type safety and autocompletion in IDEs. If you haven't installed TypeScript globally, you can do so by running:

```
npm install -g typescript
```

Usage

The `@apiclient.xyz/zitadel` package provides an unofficial TypeScript client for easy interaction with Zitadel's API, allowing for convenient management operations such as user and project management.

Getting Started

First, ensure you import the core client class from the package:

```
import { ZitaldelClient } from '@apiclient.xyz/zitadel';
```

Initialize the `ZitaldelClient` with your Zitadel instance URL and access token:

```
const zitadelClient = new ZitaldelClient({
  url: 'https://your-zitadel-instance.com', // Replace with your Zitadel instance URL.
  accessToken: 'your_access_token_here' // Replace with your actual access token.
});
```

User Management

The library allows performing user operations such as listing all users, getting info of the current user, and creating new users.

List All Users

```
async function listUsers() {
  const users = await zitadelClient.listUsers();
  console.log(users); // Outputs user list
}
listUsers();
```

Get Current User Information

```
async function getCurrentUser() {
  const user = await zitadelClient.listOwnUser();
  console.log(user); // Outputs current user information.
}
getCurrentUser();
```

Create a New User

```
async function createUser() {
  await zitadelClient.createUser({
    email: 'newuser@example.com',
    firstName: 'First',
    lastName: 'Last'
  });
  console.log('User created successfully.');
}
createUser();
```

Project Management

Beyond user management, the client also supports actions on projects—like listing and managing project roles.

List Projects

```
async function listProjects() {
  const projects = await zitadelClient.listProjects();
  console.log(projects); // Outputs list of projects
}
listProjects();
```

Project Roles

Each project comes with roles that can be managed through the client.

Listing Project Roles

To inspect roles within a project:

```
async function listProjectRoles(projectId: string) {
  const projectRoles = await someProject.listProjectRoles(); // Assume `someProject` is an
instance of `ZitadelProject`.
  console.log(projectRoles);
}
```

Note: The `ZitadelProject` object would typically be obtained as part of the list from `listProjects`.

Advanced Topics

For more complex scenarios, such as direct interaction with Zitadel's gRPC APIs, or detailed configuration, please refer to the comprehensive official Zitadel documentation. This client offers a simplified abstraction over Zitadel's API functionalities but can be extended or used in conjunction with direct API calls to meet specific needs.

For instance, the `ZitaldelClient` class can be tweaked to support custom interceptors for logging, authentication flow enhancements, or to integrate with Zitadel's event system for real-time updates on resources.

This is a basic introduction to `@apiclient.xyz/zitadel`. The client simplifies the interaction with Zitadel's API, focusing on core functionalities like user and project management. For specific use cases, detailed configurations, or advanced features, you may need to extend the client or use it as a foundation for more complex interactions with the Zitadel API. undefined

---

Revision #5
Created 2026-03-28 10:48:19 UTC by foss.global Team
Updated 2026-03-28 12:08:57 UTC by foss.global Team