Table of Contents

Altizure Development Platform

Welcome to Altizure Development Platform！

On Altizure, you can not only experience the most advanced 3D reconstruction technology to turn 2D photos to realistic 3D models, but also use the online 3D publishing service to share and enjoy the 3D models.

Join Altizure open platform to integrate the powerful 3D reconstruction and publishing services on Altizure to your business workflow. Unleash the power of realistic 3D models!

Let's start the journey.

Learn more about Altizure at:

- Explore the 3D world on Altizure: altizure.com/explore
- Facebook page
- Official blog: blog.altizure.com
- Offline documentations: pdf, epub

Contact us support@altizure.com

Last modified at Thu Dec 21 2017 21:30:16 GMT+0800 (HKT)

Concepts

Before starting to develop with Altizure, here we introduce a few concepts. If you have used some development SDK for other online platform, e.g. facebook app or github sdk, you should be familiar with them.

1. Developer account

If you do not have an altizure account yet, please get one on sign up page.

Then you can apply for developer account. All development tools and privileges will be linked with this account. Please keep this account secure and safe.

2. App

To be added

App key

To be added

App secret

To be added

3. User token

To be added

4. GraphQL API

Altizure GraphQL API is a set of API in the syntax of GraphQL. The API allows developers to fetch and modify the data on Altizure. Learn more at GraphGL API

5. Javascript SDK

Altizure Javascript SDK allows you to integrate rich 3D experience with our realistic 3D models to your business workflow. Learn more at Javascript SDK

Last modified at Mon Oct 30 2017 19:30:15 GMT+0800 (HKT)

Developer Account

We are sorry that the developer account is not yet for public application. It is only for invited partners at the moment.

For latest information about our open platform, please keep following us at:

- Facebook page
- Official blog: blog.altizure.com

Last modified at Mon Oct 30 2017 19:30:15 GMT+0800 (HKT)

Altizure Graphql API

Altizure GraphQL API is a set of API in the syntax of GraphQL. The API allows developers to fetch and modify the data on Altizure.

1. Prerequisite

- Altizure developer account
- App key
- User token (optional)

2. API Endpoint and Documentations

API endpoints and documentations: api.altizure.com/graphql.

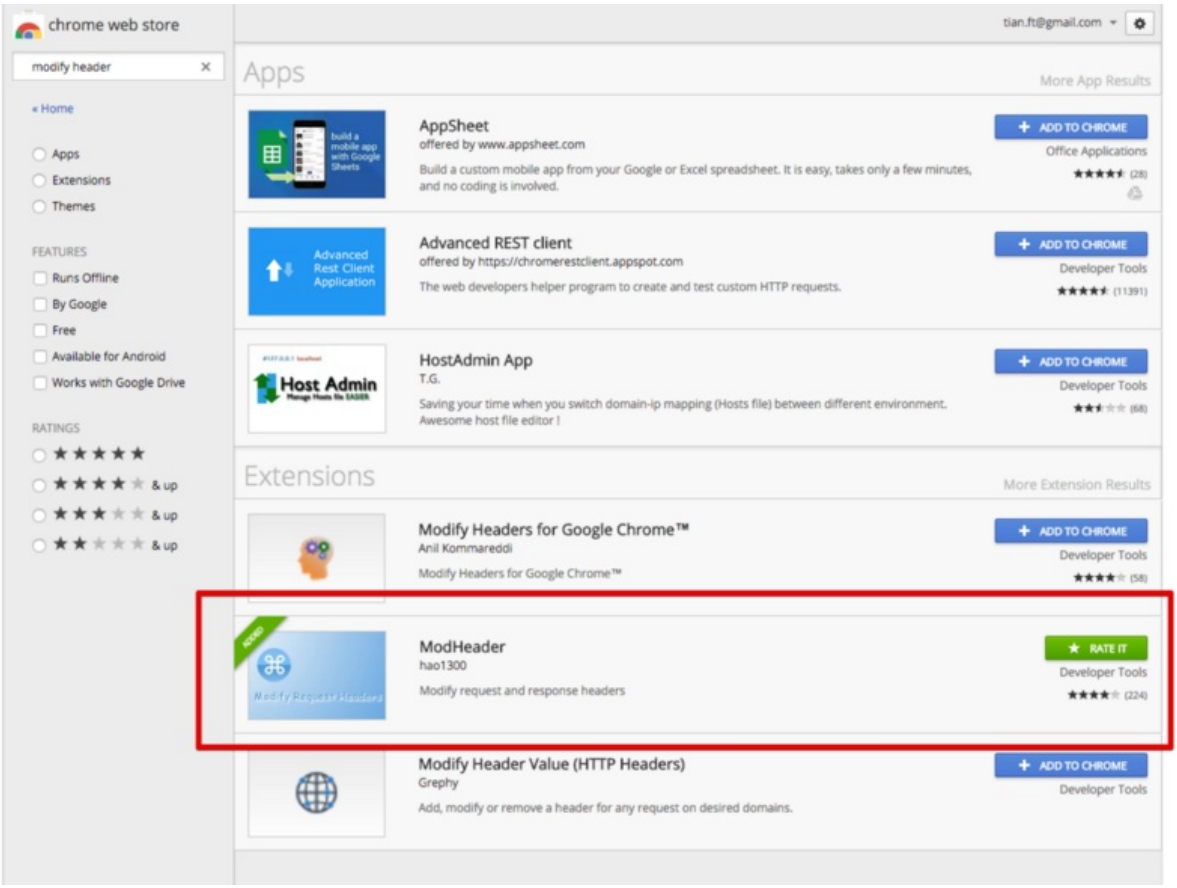
3. Try out API in your browser

It is very convenience to test the API in browsers, because it provides instant feedback on the query results and detailed inline documentations. After the testing, you can easily copy and paste the query string to your code and trigger the API call.

We take Google Chrome as an example. Other browsers supporting extensions, e.g. Firefox, should work too.

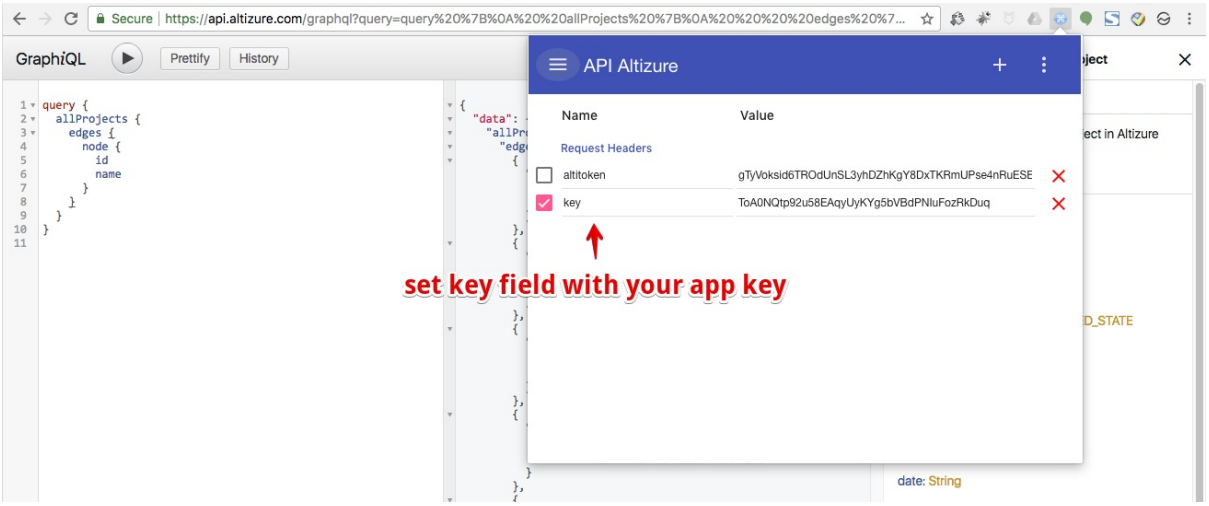
Install extension

First, please install an extension that can modify the http request header. Here ModHeader is used. Please search and install `ModHeader` in the extension store of Google Chrome.



Modify http header

Please use ModHeader to add `key` field in request header with app key as the value.



Visit api.altizure.com/graphql after setting the key. You can find three sections: "query section", "result section", and "documentation", on the page.

Now please fill the following query string to the query section to calculate how many gigapixels there are in a set of images.

```
query {
  support {
    sizeToGigaPixel(width: 4000, height: 3000, numImg: 100)
  }
}
```

Please click the action button to get the query results. This api call will compute how many giga pixels there are in 100 images at 4000x3000 resolution.

4. Integrate the API in your code

The API can be called by any libs and programming languages that can issue a http post request.

For example:

JQurey in Javascript

```
$.ajax({
    type: 'POST',
    url: 'https://api.altizure.com/graphql',
    headers: {
      altitoken: 'user token',
      key: 'app key'
    },
    data: 'query=' + 'GraphGL query string'
  })
```

5. Obtain user token

User token is obtained via the standard OAuth 2 flow.

The authorization endpoint is the following url:

```
`https://api.altizure.com/auth/start?client_id=${appKey}&response_type=token&redirect_uri=${redirect_uri}`
```

where **appKey** is your application key, and **redirect_uri** is one of the domains associated with your application.

Your application needs to route/open this url. A form will be shown to your users asking for their authorizations. After your users have authorized the request, the page will be redirected back to your **redirect_uri** with a url hash variable of key: **access_token**.

For mobile application, the **redirect_uri** will be your application's bundle identifier name (iOS) or your package name (android).

For a vanilla JS implementation, please refer to here.

6. FAQ

6.1 How to access the api in Mainland China?

Please use `https://api.altizure.cn/graphql`. It is better to choose a reachable and faster endpoint whenever possible in the logic of your application instead of hardcoding the api endpoint.

6.2 Where to find more detailed documentations on GraphQL API

Please follow the above tutorial and browse api.altizure.cn/graphql with your browser. All documentations are embedded in the web frontend of our api endpoint.

7. Learn More

- Learn more about GraphQL
- Use Altizure Javascript SDK to developer rich 3D application
- More tools on GraphQL: Awesome GraphGL

Last modified at Sun Dec 10 2017 23:30:16 GMT+0800 (HKT)

Using the upload API

Overview

To speed up data transfer, images are uploaded directly from clients to Amazon or Aliyun. Your image are protected securely. Others could not read or modify your images. That is why you need to obtain the authorizations from our api. As filename is the only identifier in the buckets, the image is required to be uploaded to a specific url (S3) or prefix (OSS) so that Altizure knows which image is which.

1. Choose bucket

Choose one of the fastest S3 or OSS buckets. This is largely correlated to your network zone with the edge-points. You may simply query `GeoIPInfo.nearestBuckets` for hints for auto selecting the best edge for your clients.

2. Compute image checksum

Compute the sha1sum hash for each image. Before each upload, check if the current image has already been uploaded by calling the mutation `hasImage(pid, checksum)`. If it is not, proceed to the next step. Otherwise, skip to the next image.

3. Upload to OSS

If an OSS bucket is chosen, obtain STS and the related meta image info (e.g. id and hashed filename) by calling mutation `uploadImageOSS(pid, bucket, filename, type, checksum)`. STS is a temporary (1 hour) security token for the write only permission on the `/pid` prefix. If it has expired, renew with the same mutation. Otherwise, only request the `image` gql fragment from the result, and re-use the same STS for performance reason. As signing from Aliyun is slow, one should not sign a new STS for each image. If the `Wrapper` in Aliyun OSS Javascript SDK is used, please do not recreate new wrappers before STS expires to avoid an error about too many sockets.

Given the STS, images could be uploaded via any compatible protocol or library to OSS. It is required to upload with the specific hashed filename according to the image info returned from the `uploadImageOSS` mutation. Specifically, it is required to be put to the path `${pid}/${image.filename}` inside the bucket.

In order to keep track of the state, just before an image is uploaded, call mutation `startImageUpload(id)` to signal the start of the process. When the upload is done, call mutation `doneImageUpload(id)`.

3. Upload to S3

If a S3 bucket is chosen, uploading is much simpler. For each image, call the mutation `uploadImageS3(pid, bucket, filename, type, checksum)` to obtain a temporary (3 hours) signed url and the related meta image info.

Given the signed url, use the standard HTTP put to put the file to this url with `Content-type: JPEG` or other formats accordingly.

Just before each upload, it is required to call mutation `startImageUpload(id)`. There is no need to signal the end of uploading.

4. Wait for image processing

Uploaded images will be copied, verified and processed. Eventually, the image state will become either `Ready` or `Invalid`. If the total count of `Ready` matches the desired number, you may call mutation `startReconstructionWithError(id, options)` to start the reconstruction. Otherwise, you may consider re-uploading any outstanding image as indicated by the mutation `hasImage`.

If you do not concern about a few number of missing images and want to start the reconstruction immediately once all the images are ready, you could call mutation `preStartReconstruction(id, options)` .

Learn more

- Learn more about STS
- Aliyun OSS Javascript SDK: link

Last modified at Mon Dec 11 2017 18:30:15 GMT+0800 (HKT)

Altizure Javascript SDK

Altizure Javascript SDK allows you to integrate rich 3D experience with our realistic 3D models to your business workflow.

- Simplify the loading and rendering of large-scale realistic 3D models.
- Simplify the fusion and rendering data from different sources.
- Simplify the development of integrating realistic 3D models to different business workflows.

Combined with tools like Electron and React Native, you can easily develop high quality 3D applications with realistic models for desktop and mobile apps.

1. Quick Start

Include SDK

In the `<head>` section of the html page, include our SDK as

```
<!-- Set the encoding to make sure that utf8 character is shown correctly -->
<meta charset="utf-8">
<!-- Set the viewport to make the page look correct on mobile -->
<meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
<!-- Include SDK -->
<script type="text/javascript" src="https://beta.altizure.com/sdk"></script>
```

Here we provide three links for our SDK.

- Latest：`<script type="text/javascript" src="https://beta.altizure.com/sdk"></script>`
- Stable：`<script type="text/javascript" src="https://www.altizure.com/sdk"></script>`
- Mainland China：`<script type="text/javascript" src="https://www.altizure.cn/sdk"></script>`

Create a div as a container for 3D rendering

Our SDK will take over the downloading and rendering of realstic 3D models. Developers should create a `div` and attach a sandbox object to it. This div specifies where 3D contents will be rendered.

```
<body>
  <div id="page-content"></div>
</body>
```

Create a sandbox object

We now create a sandbox object and attach it to the div we created to hold this sandbox. Then the sandbox object will render all 3D contents in such a div.

```
// Create an option for configuring the sandbox
let options = {
  altizureApi:{
    // your app key
    key: 'your-app-key'
  }
}

// Create a sanbox object and attach it to the div page-content.
let sandbox = new altizure.Sandbox('page-content', options)
```

`'page-content'` is the id of `div` where the 3D contents are rendered. `options` is used to configure the sandbox. Please refer to our sample section for more information about how to configure a sandbox.

Summary

All codes can be put together as:

```
<html lang="en">
<head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1.0, user-scalable=no">
  <script type="text/javascript" src="https://beta.altizure.com/sdk"></script>
</head>
<body>
  <div id="page-content"></div>
  <script>
    let options = {
      altizureApi:{
        // your app key
        key: 'your-app-key'
      }
    }

    let sandbox = new altizure.Sandbox('page-content', options)
  </script>
</body>
</html>
```

Save this code fragment to `<path>/altizure-sdk-test/earth.html`. Then start a http server in your console by typing

```
cd <path>/altizure-sdk-test/
python -m SimpleHTTPServer
```

This page can be accessed at `http://127.0.0.1:8000/earth.html` to load Altizure Earth

You can check this demo on our site to see how it looks.

In the following, you can learn more about our SDK via samples and demos.

2. Samples

Up-to-date live samples are at altizure.github.io/sdk.examples/examples.sdk.html.

2.1 Concepts

Here we start with some concepts in our SDK.

- Sandbox: altizure.Sandbox is the core of the 3D engine. It is the entry point of the development and handles all 3D rendering and data management.

The source code of these samples can be downloaded at github.com/altizure/sdk.examples. You can easily setup your server and run these samples following the tutorial.

Any questions and bugs, please feel free to write on our issue page.

2.2 List of Samples

- 2.2.1 Sandbox setting
  - Default loading

3. FAQ

Please check FAQ page for more information and demo page for more sophisticated demos. Please also check the Reference documents.

4. Learn More

- ThreeJS
- WebGL
- OpenGL
- Vulkan
- OpenGL Transformation

Last modified at Thu Dec 21 2017 20:04:53 GMT+0800 (HKT)

Altizure JavaScript SDK Demo

Here we show how to use Altizure Javascript SDK to solve some real applications to demonstrate how to integrate realistic 3D models to business application.

If you have any questions, please feel free to write on our issue page.

1. Timeline

Timeline is very useful for presenting a time series of a changing site. This is a essential tool for monitoring the progress of construction. We show several variation of timeline visualization.

1.1 Timeline in Grid Style

- Demo link: altizure.github.io/experimental-demo/timeline.html
- Keywords: Loading and setting multiple Altizure projects, text tags.
- Brief introduction: In this demo, a series of 3D models of a construction site are laid out in grid style. Each model is attached with a text tag to show the time stamp of the construction site.

1.2 Timeline with Sliders

- Demo link: altizure.github.io/experimental-demo/timeline-slider.html
- Keywords: Visibility of markers, interaction, camera flight, marker loading and destruction.
- Brief introduction: In this demo, a series of 3D models of a construction site are laid out with a slider. Users can drag the slider to switch the models in different time. The demo also shows how to control the camera flight animation to help users to fly around 3D models.

2. Sensors

Modern handheld mobile devices contain lots of sensors, measuring positions, poses, and velocities. In this section, we demonstrate how to integrate the information of sensors with 3D models to make the scenes more interactive.

2.1 GPS Tracking

- Demo link: altizure.github.io/experimental-demo/gps-tracker.html
- Keywords: Image tag, tag position, camera following, coordinate transformation, canvas with 3D elements, altitude reading.
- Brief introduction: This demo is only on mobile phones. After you press tracking, the demo will track your GPS location and show your track with aspect to the center of the scene. Your position is shown at realtime as an Altizure logo.

3. Performance Benchmark

- Demo link: altizure.github.io/experimental-demo/performance-test.html
- Keywords: Text tag, Rendering benchmark, Altitude reading.
- Brief introduction: The demo renders 1000 text tags by default. Users can change the number of text tags and click `Generate` to generate a new scene. The more tags can be rendered smoothly, the more powerful is your system.

4. How to Get the Code?

All the demos are open source. You can get the codes by running:

```
git clone https://github.com/altizure/experimental-demo.git
cd experimental-demo
python -m SimpleHTTPServer
```

Open your browser and access `http://127.0.0.1:8000/` to run and check the demos.

5. Learn More

- Altizure Javascript SDK
- Altizure Javascript SDK FAQ
- Altizure Javascript SDK Reference

—

Last modified at Thu Dec 21 2017 20:04:53 GMT+0800 (HKT)

Frequently Asked Questions

1 General Questions

1.1 How to start a testflight on Altizure Javascript SDK?

Please follow the guide altizure.github.io/sdk.examples/examples.sdk.html to test all samples. The source code can be cloned to your local computer for editing and testing.

1.2 The remote model cannot be loaded when I test the samples on my local computer.

Please make sure that you start a local http server and access the sample page via `127.0.0.1` instead of `localhost`

- Check whether a local http server is setup for testing. The simplest way to start a local http server is to use python and follow the guide at altizure.github.io/sdk.examples/.
- Use `127.0.0.1` instead of `localhost` to access the sample page hosted on your local computer.

1.3 The link of SDK script cannot be accessed.

If `https://www.altizure.com/sdk` and `https://beta.altizure.com/sdk` are not reachable. Please switch to `https://www.altizure.cn/sdk` 。

1.4 Where can I find the reference manual?

Please check our samples and demos. They are detailed documented. We are constantly updating the samples and demos to illustrate the latest features.

1.5 How to get the user token?

Please refer to Section `5. how to get user tokens` on GraphQL API.

1.6 In which programming language is the 3D rendering engine on Altizure developed?

We use Javascript. The underlying 3D API is WebGL. But we hardly write raw WebGL codes. Most of the features are developed on top of ThreeJS.

1.7 Can I implement some elements in three.js and insert it to the scene created using Altizure Javascript SDK?

No, at the moment. We did lots of optimization on memory management to enable fast transfer and rendering of large-scale realistic 3D models, so it is not easy to integrate customized three.js objects to altizure scenes. However, we have plan to do so.

1.8 Do I have to learn about WebGL and computer graphics to use Altizure Javascript SDK?

In general, no, we design the SDK for non-computer graphics experts, anyone with the knowledge of Javascript and HTML should be able to use Altizure Javascript SDK to integrate the realistic 3D models to the business workflow. Nevertheless, it is always a plus if you know some basic knowledge on WebGL and computer graphics.

2 Common Use Cases

Coming soon

3. Learn More

- List of samples
- Demos illustrate more sophisticated examples.
- Altizure Javascript SDK Reference

—

Last modified at Thu Dec 21 2017 21:30:16 GMT+0800 (HKT)