

Javascript Examples

The following code sample was written in java using the fetch method.

It requires an API_KEY and FILE_NUMBER_OF_UPLOADED_VIDEO

Depending on your operating system you will need node.js loaded <https://nodejs.org/en> and this code can be run:

```
node se-localfile.mjs
```

```
node --experimental-modules se-localfile.mjs
```

```
se-localfile.mjs
```

```
import fetch from 'node-fetch';

const url = 'https://streamengine.igolgi.com/api/v0/job';
const username = 'API_KEY';
const password = 'a';

const headers = {
  'Authorization': 'Basic ' + Buffer.from(username + ":" + password).toString('base64'),
  'Content-Type': 'application/json'
};

const data = {
  videoProfiles: [
    {
      width: 960,
      height: 544,
      video_bitrate: 2500,
      video_framerate: "1x",
      audio_profiles: "0"
    }
  ],
  separate_audio: false,
```

```
segmented_output_dash: false,
output_container: "mp4",
picture_transform: "none",
logo_url: null,
audio_volume: 100,
closed_captions: false,
create_tar_file: false,
gop_length: 2,
h264_quality: "good",
scte35_pid_remap: -1,
video_aspect_ratio: "rotate_counterclockwise + invert_aspect_ratio",
rotation_blackness: 0,
scte35_passthrough: false,
segmented_output_hls: false,
ip_distance: 2,
audioProfiles: [
  {
    audio_codec: "aac",
    audio_channels: 2,
    audio_bitrate: 256,
    primary_audio_downmix_to_stereo: false,
    source_stream: 1
  }
],
input: "https://streamengine.igolgi.com/uploads/FILE_NUMBER_OF_UPLOADED_VIDEO",
cloud_credentials: {
  input: {
    cloud_provider: "file-upload",
    access_key: null,
    secret_key: null,
    region: "ca-toronto-1",
    tenancy_ocid: null,
    user_ocid: null,
    oci_fingerprint: null
  },
  output: {
    cloud_provider: "igolgi-store",
    access_key: null,
    secret_key: null,
    region: "ca-toronto-1",
```

```

    tenancy_ocid: null,
    user_ocid: null,
    oci_fingerprint: null
  }
},
master_variant_mode: false,
video_codec: "h.264"
};

fetch(url, {
  method: 'POST',
  headers: headers,
  body: JSON.stringify(data)
})
.then(response => response.json())
.then(data => console.log(data))
.catch(error => console.error('Error:', error));

```

se-job-status.mjs

```
$node se-job-status.mjs job-id
```

This sample script will run every 10 seconds to see if a job is still running, when its finished it will display finished, feel free to modify this.

```

import fetch from 'node-fetch';

const checkJobStatus = async (jobId) => {
  const url = `https://streamengine.igolgi.com/api/v0/job/${jobId}`; // Adjust this URL as needed
  const username = 'API_KEY';
  const password = 'a';

  const headers = {
    'Authorization': 'Basic ' + Buffer.from(username + ":" + password).toString('base64'),

```

```
'Content-Type': 'application/json'
};

try {
  const response = await fetch(url, {
    method: 'GET',
    headers: headers
  });

  if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
  }

  const data = await response.json();
  console.log('Job status:', data);
  return data;
} catch (error) {
  console.error('Error checking job status:', error);
}

};

const checkUntilComplete = async (jobId) => {
  let status;
  do {
    const result = await checkJobStatus(jobId);
    status = result.status; // Adjust this based on the actual API response structure
    if (status !== 'completed') {
      console.log('Job still in progress. Waiting 10 seconds before checking again...');
      await new Promise(resolve => setTimeout(resolve, 10000)); // Wait for 10 seconds
    }
  } while (status !== 'completed');
  console.log('Job completed!');
};

// Get job ID from command line argument
const jobId = process.argv[2];

if (!jobId) {
  console.error('Please provide a job ID as a command-line argument.');
  console.error('Usage: node se-job-status.mjs <job_id>');
```

```
process.exit(1);  
}
```

```
console.log(`Checking status for job ID: ${jobId}`);  
checkUntilComplete(jobId);
```

Revision #2

Created 8 July 2024 22:29:24 by Admin

Updated 25 July 2024 16:43:34 by Admin