ByteNite API docs

Authentication

1. Open https://app.bytenite.com and login
2. Open the profile page by clicking on your avatar (top right of the screen)
3. Click **Get API key**
4. Copy the access token
5. Create a file named `token.txt` in this folder with the token inside

To send authenticated requests set the http header `Authorization`

API usage

Get user info

Send an authenticated request to the endpoint http://api.bytenite.com/v1/auth/user

```
import requests

access_token='...'
resp = requests.get('http://api.bytenite.com/v1/auth/user', headers={'Authorization': access_token})
user = resp.json()
print(user)
```

Example response

```
{
  "user": {
    "id": "...",
    "createdAt": "2023-01-04T16:35:26.928Z",
    "lastLogin": "2023-03-15T17:13:45.913Z",
    "displayName": "n.castelli",
    "email": "...",
    "phoneNumber": "",
    "photoUrl": "",
    "providerId": "firebase",
    "emailVerified": true,
    "customClaims": {}
  }
}
```

Get your balance

Send an authenticated request to the endpoint https://api.bytenite.com/v1/customer/wallet/balance

```
import requests

access_token='...'
resp = requests.get('https://api.bytenite.com/v1/customer/wallet/balance', headers={'Authorization': access_toke
balance = resp.json()
print(balance)
```

Example response

```
{
  "accountId":"...",
  "totalBalance":"18335",
  "availableBalance":"18335"
}
```

Run a job

Step 1: Create a new job

To create a new job send an authenticated `POST` request to `http://api.bytenite.com/v1/customer/jobs`.

The request accepts two parameters: - `templateId` should be set to `video-transcoding@v0.2` for a video encoding job - `name` is a mnemonic string (min 4 characters)

Both parameters are required.

If the request is successful a response with the new job id is returned.

Step 2: Set data source and destination

Data source and data destination are dynamic fields that have different parameters according to the type chosen.

Supported data sources: - Url - descriptor: `url` - type: `type.googleapis.com/bytenite.data_source.HttpDataSource` - parameters: `{"url": "https://..."}` - Local file - descriptor: `local` - type: `type.googleapis.com/bytenite.data_source.LocalFileData` - parameters: `{"temp_url": "https://...", "name": "..."}`

Supported data destination - Temporary bucket - descriptor: `bucket` - parameters: `{}` - GCP - descriptor: `gcp` - type: `type.googleapis.com/bytenite.data_source.S3DataSource` - parameters: `{"name": "...", "secret_key":"" "access_key":"", "bucket_name":"" "cloud_region":""}` - Storj - descriptor: `storj` - type: `type.googleapis.com/bytenite.dat` - parameters: `{"name": "...", "secret_key":"" "access_key":"", "bucket_name":"" "cloud_region":""}` - S3 - descriptor: `s3` - type: `type.googleapis.com/bytenite.data_source.S3DataSource` - parameters: `{"name": "...", "secret_key":"" "access_key":"", "bucket_name":"" "cloud_region":""}`

To set data source and destination send a `POST` request to `https://api.bytenite.com/v1/customer/jobs/datasource/{job_id}`

Step 3: Set encoding parameters

Use the GUI to generate a valid set of encoding parameters - Open https://app.bytenite.com/encoding - Create a new job and select a random data source and destination - Select a preset or "unpackaged" - Configure encoding parameters - Click **Next** - Click **Switch to JSON** - The parameters are the object inside the `params` property

To set encoding parameters send a `POST` request to `https://api.bytenite.com/v1/customer/jobs/run/{job_id}`

Step 4: Run

Send a `POST` request to `https://api.bytenite.com/v1/customer/jobs/params/{job_id}` with an empty object as body

Example

```python
import requests

access_token='...'

# Create a new video transcoding job
create_job_body={"name": "Example job", "templateId": "video-transcoding@v0.2"}
resp = requests.post('http://api.bytenite.com/v1/customer/jobs', json=create_job_body, headers={'Authorization':
job_data = resp.json()['job']

# Set a data source and a data destination
job_id = job_data['id']
data_source_body={
    "dataSource": {
        "dataSourceDescriptor":"url",
        "params":{"@type":"type.googleapis.com/bytenite.data_source.HttpDataSource","name":"","url":"https://sto
    },
    "dataDestination": {
        "dataSourceDescriptor":"bucket",
        "params":{}
    }
}

resp = requests.post(f'https://api.bytenite.com/v1/customer/jobs/datasource/{job_id}', json=data_source_body, he
resp.raise_for_status()

# Set video encoding parameters
job_params_body={
    "data": {
```

```
"output_code_template": "",
"output_template": "{{.job_name}}_{{.output_params.Aspect.Resolution.Height}}",
"outputs": [
  {
    "output_params": {
      "aspect": {
        "cropping": {
          "bottom": 0,
          "left": 0,
          "right": 0,
          "top": 0
        },
        "orientation": {
          "flip": "",
          "rotation": ""
        },
        "padding": {
          "fill_color": "#000000",
          "final_aspect_ratio": ""
        },
        "resolution": {
          "aspect_ratio": "",
          "avoid_upscaling": False,
          "height": "720p"
        }
      },
      "audio": {
        "audio_bitrate": "320k",
        "audio_channel": "2.0",
        "audio_codec": "AAC",
        "audio_sample_rate": "48khz"
      },
      "video": {
        "bitrate": {
          "avg_bitrate": "3000k",
          "buffer_size": "8000k",
          "max_bitrate": "8000k",
          "min_bitrate": "700k",
          "rate_control_mode": "variable_bitrate"
        },
        "codec": "libx264",
        "codec_params": {
          "level": "",
          "preset": "slow",
          "profile": "",
          "tune": "film"
        },
        "frame_rate": {
          "fps": "",
          "up_mode": "avoid"
        }
      }
    },
    "output_type": "mp4"
  },
  {
    "output_params": {
      "aspect": {
        "cropping": {
          "bottom": 0,
          "left": 0,
                "right": 0,
                "top": 0
              },
              "orientation": {
                "flip": "",
                "rotation": ""
              },
              "padding": {
                "fill_color": "#000000",
                "final_aspect_ratio": ""
              },
              "resolution": {
                "aspect_ratio": "",
                "avoid_upscaling": False,
                "height": "480p"
              }
            },
            "audio": {
              "audio_bitrate": "320k",
              "audio_channel": "2.0",
              "audio_codec": "AAC",
              "audio_sample_rate": "48khz"
            },
            "video": {
              "bitrate": {
                "avg_bitrate": "800k",
                "buffer_size": "2000k",
                "max_bitrate": "2000k",
                "min_bitrate": "400k",
                "rate_control_mode": "variable_bitrate"
              },
              "codec": "libvpx-vp9",
              "codec_params": {
                "cpu_used": 1
              },
              "frame_rate": {
                "fps": "",
                "up_mode": "avoid"
              }
            }
          },
          "output_type": "mp4"
        }
      ]
    },
    "preset": "unpackaged"
}
resp = requests.post(f'https://api.bytenite.com/v1/customer/jobs/params/{job_id}', json=job_params_body, headers
resp.raise_for_status()

# Run the job
requests.post(f'https://api.bytenite.com/v1/customer/jobs/run/{job_id}', json={}, headers={'Authorization': acce
```

Check the notebook `run_job.ipynb` for a working example

Get job results

Send an authenticated request to the endpoint `https://api.bytenite.com/v1/customer/jobs/{job_id}/results`

```
import requests

access_token='...'
job_id='...'
```

```
resp = requests.get(f'https://api.bytenite.com/v1/customer/jobs/{job_id}/results', headers={'Authorization': acc
results = resp.json()
```

Example response

```
{
  "results": [
    {
      "name": "out_0/Example_job_480p.mp4",
      "link": "..."
    },
    {
      "name": "out_1/Example_job_720p.mp4",
      "link": "..."
    }
  ]
}
```

Online docs

To see all endpoints and data structures visit https://api.bytenite.com/v1/customer/docs/