Version 2025.2.1

Guide of a FastReport Cloud programmer

Foreword

This documentation contains information about the beta version of FastReport Cloud and may differ from the current product version.

Any code fragments that may be found in this documentation are only for demonstration purposes. These examples are meant to illustrate the functionality of the FastReport Cloud product. Fast Reports Inc. does not guarantee the accuracy and completeness of these examples and is not responsible for any direct or indirect damage that may be caused by them.

The documentation may contain hyperlinks to various Internet resources. These links are up to date as of the writing of the documentation. Fast Reports Inc. is not responsible for their availability at the time of reading the documentation or for any damage caused by them.

Guides

This section is dedicated to guides on working with the cloud and FastReport Cloud integration into the application. Guides are currently available for:

- REST API
- C#/.NET

Guides for other programming languages are planned.

Write to us if you are interested in any specific programming language or platform. Write!

Development tools

There are SDKs available for the following languages:

- Haskell
- JavaScript
- C++
- Python
- Java
- Go
- C#/.NET and ASP.NET

Please note that the development tools are sourced on Github. To use them, you will need to build the package or library from the sources yourself.

REST API

The instructions and guides in this section are designed to help you execute common tasks with FastReport Cloud, regardless of your programming language.

Getting Started

You will need the following tools and features:

1. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

2. Report designer FastReport Community Designer.

   Some guides will require a designer to create the report.

3. Active FastReport Cloud subscription.

4. Access to the Internet.

> Note! These guides assume that you already know how to use curl or use another REST client.

> Note! The paragraphs above describe the recommended tools.

What's next?

We suggest starting with the following articles:

- Status codes.
- Authentication and authorization.

Status codes

This article lists all status codes that can be returned by FastReport Cloud services.

Request processed successfully—2xx

- 200 OK—request processed successfully, used in most cases when a response body is returned along with the result.
- 201 Created—the request was processed successfully, and a new entity was created as a result of its execution.
- 204 No Content—the request was processed successfully, and an empty response body was returned.

User Error—4xx

- 400 Bad Request—the request contains errors, for example:
  - A parameter required to complete the request is missing.
  - `Id` format of the entity is different from Hex 24.
  - The request has a negative skip or take parameter.
- 401 Unauthorized—the request came from an unauthorized user.
- 402 Payment Required—the requested resource is restricted under the subscription plan.
- 403 Forbidden—the requested resource was found, but the user does not have permission to act.
- 404 Not Found—the requested resource was not found.
- 413 Request Entity Too Large—the request body exceeds the limits in the service settings.

Server Error—5xx

- 500 Internal Server Error—the request passed all validation checks, but an Exception occurred during its execution. In this case, please report by writing to - support@fast-report.com

Other

Intermediate services between FastReport Cloud and the user can return other status codes.

Authentication and authorization

The process of authenticating user data and issuing certain rights to the user in FastReport Cloud is carried out using one of two available methods:

1. Via JWT token.

   In this case, authentication must be completed by a person, and the token will only be valid for 5 minutes, during which the user must log in to their application. When connecting to the server, the browser will redirect you to the authentication server, after which it will generate an access token. For security reasons, we limit the possibility of getting a JWT token only by a person.

   > If the user has not logged into the application within 5 minutes, authentication must be repeated. If the user has logged into the application, re-authentication is not required.

2. Via API key.

   In this case, access permissions are obtained for server applications. To get the API key, the presence of a user is required. However, the key itself can remain valid for a long time, for example, a year.

Getting the first API key

To get the first API key, use the user panel. If for some reason there is no access to the user panel, you can request a key as described below.

1. Open the link in a browser: https://fastreport.cloud/account/signin?r=https://fastreport.cloud/api/manage/v1/ApiKeys.

   Clicking on this link will direct you to the browser's automatic authentication process.

2. Once the authentication is passed, you need to request a new key.

   Press `F12` or `Ctrl+Shift+I` to open the developer panel. The keyboard shortcut may differ from the standard ones, in this case, open the developer panel through the browser menu.

3. Copy and execute the code in the JavaScript console.

   This code will make a `POST` request to the URL `https://fastreport.cloud/api/manage/v1/ApiKeys` to generate a new access key before 2030.

4. Refresh the browser page and take the result.

```
{
  "apiKeys": [
    {
      "value": "cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey",
      "description": "Generated by js develop panel",
      "expired": "2030-01-01T07:41:23.399Z"
    }
  ],
  "count": 1
}
```

Now you can use the API key, in the case above it is `cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey`.

ПYou don’t need to get a new API key through the browser again.

How to use the API key

The key should be passed with every request in the `Authorization: Basic` header. The username should be `apikey`, and the password should be the key value. For example.

```
Authorization: Basic Base64Encode(apikey:cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey);
```

Where `Base64Encode` is the function for converting a string to `base64` when using `UTF8` encoding.

Getting a new API key

To get a new key, make a `POST` request to the entry point `https://fastreport.cloud/api/manage/v1/ApiKeys` and pass JSON in the request body as shown below.

```
{
   "description": "string",
   "expired": "string($date-time)"
}
```

Request example.

```
curl -X POST "https://fastreport.cloud/api/manage/v1/ApiKeys" -H "accept: text/plain" -H "authorization: Basic YXBpa2V5OmNjMzU1b2V1MXo1ZDV3bmNheW8zM21lNmMxZzVqdW5xZHFrNHBrdXBpZDd0OHluanNoZXk=" -H "Content-Type: application/json-patch+json" -d "{ \"description\": \"Generated by js develop panel\", \"expired\": \"2030-01-01T07:41:23.399Z\"}"
```

Response scheme.

```
{
  "value": "string",
  "description": "string",
  "expired": "2020-12-02T08:47:43.270Z"
}
```

What's next?

- Uploading a new template.
- Working with groups.
- Adding new users to a workspace.

Uploading a new template

One of the main features of FastReport Cloud is storing templates, reports, and other data in the cloud. This article will cover how to upload your prepared template into the FastReport Cloud template repository.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Report template.

   It can be created using the free program FastReport Community Designer.

4. Active FastReport Cloud subscription.

5. Access to the Internet.

Instruction

1. You need to get the identifier of the workspace root folder. This identifier will never change, so you can save it and not have to request it every time before uploading a new template.

   To request the root directory, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Templates/Root`.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/Root" -H "accept: text/plain"
   ```

   In this case, the directory for the default workspace will be returned. You can specify a specific workspace by passing the `subscriptionId` parameter.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/Root?subscriptionId=your_workspace_id" -H "accept: text/plain"
   ```

   Response example.

   ```
   {
    "name": "RootFolder",
    "parentId": null,
    "tags": [],
    "icon": "",
    "type": "Folder",
    "size": 16384,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "None",
    "id": "5fa919f9292a8300019349b9",
    "createdTime": "2020-11-09T10:29:13.928Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "2020-11-13T15:58:45.69Z",
    "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
   }
   ```

   The directory identifier from the example above is `5fa919f9292a8300019349b9`.

2. To download the required template, make a `POST` request to `https://fastreport.cloud/api/rp/v1/Templates/Folder/{id}/File`, where instead of `{id}` you should substitute the directory identifier, and pass JSON in the request body as shown below.

```
{
  "name": "string",
  "content": "base64encoded(template.frx)"
}
```

   - `name` — the name of the file that will be displayed in the user panel.

     Add the `.frx` extension manually.

   - `content` — file content encoded in base64.

   Request example.

```
curl -X POST "https://fastreport.cloud/api/rp/v1/Templates/Folder/5fa919f9292a8300019349b9/File" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"name\": \"template.frx\", \"content\": \"77u/PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0idXRmLTgiPz4NCjxSZXBvcnQgU2NyaXB0TGFuZ3VhZ2U9IkNTaGFycCIgUmVwb3J0SW5mby5DcmVhdGVkPSIxMi8wNC8yMDIwIDEwOjU4OjU3IiBSZXBvcnRJbmZvLk1vZGlmaWVkPSIxMi8wNC8yMDIwIDExOjAwOjIwIiBSZXBvcnRJbmZvLkNyZWF0b3JWZXJzaW9uPSIyMC4yMC4wLjEiPg0KICA8RGljdGlvbmFyeS8+DQogIDxSZXBvcnRQYWdlIE5hbWU9IlBhZ2UxIiBXYXRlcm1hcmsuRm9udD0iQXJpYWwsIDYwcHQiPg0KICAgIDxDYXRhbG9nIFRvcD0iMzcuOCIvPg0KICAgIDxQYWdlSGVhZGVyQmFuZCBOYW1lPSJQYWdlSGVhZGVyMSIgVG9wPSI0MSIgV2lkdGg9IjcxOC4yIiBIZWlnaHQ9IjI4LjM1Ii8+DQogICAgPERhdGFCYW5kIE5hbWU9IkRhdGExIiBUb3A9IjcyLjU1IiBXaWR0aD0iNzE4LjIiIEhlaWdodD0iNzUuNiI+DQogICAgICA8VGV4dE9iamVjdCBOYW1lPSJUZXh0MSIgV2lkdGg9Ijc1LjYiIFRleHQ9IkhlbGxvLCBXb3JsZCEiIEZvbnQ9IkFyaWFsLCAxMHB0Ii8+DQogICAgPC9EYXRhQmFuZD4NCiAgICA8UGFnZUZvb3Rlck9iamVjdCBOYW1lPSJQYWdlRm9vdGVyMSIgVG9wPSIxMTUxLjM1IiBXaWR0aD0iNzE4LjIiIEhlaWdodD0iMTguOSIvPg0KICA8L1JlcG9ydFBhZ2U+DQo8L1JlcG9ydD4NCg==\"}"
```

   Response example.

```
{
  "reportInfo": {
    "author": null,
    "created": "2020-12-04T10:58:57",
    "creatorVersion": "20.20.4.1",
    "description": null,
    "modified": "2020-12-04T11:00:20",
    "name": null,
    "picture": null,
    "previewPictureRatio": 0,
    "saveMode": "All",
    "savePreviewPicture": false,
    "tag": null,
    "version": null
  },
  "name": "template.frx",
  "parentId": "5fa919f9292a8300019349b9",
  "tags": null,
  "icon": null,
  "type": "File",
  "size": 17159,
  "subscriptionId": "5fa919fa292a8300019349bc",
  "status": "Success",
  "id": "5fc9ece6b792c90001d94b13",
  "createdTime": "2020-12-04T08:01:42.7087229+00:00",
  "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "editedTime": "2020-12-04T08:01:42.7087112+00:00",
  "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
}
```

3. Now this template can be used to prepare (build) a report and export.

What's next?

- Permission management on the example of a template.
- Building a report.

Adding a JSON data source

This article walks you through the process of creating a new JSON data source in FastReport Cloud.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. JSON file.

4. JSON schema.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

Creating a data source

To create a data source, send a CREATE request to `https://fastreport.cloud/api/data/v1/DataSources`.

Request body example:

```
{
  "name": "fakeAPI",
  "connectionString":
"Json=aHR0cHM6Ly9qc29ucGxhY2Vob2xkZXIudHlwaWNvZGUuY29tL3Bvc3Rz;JsonSchema=eyJ0eXBlIjoiYXJyYXkiLCJpdGVtcyI
6eyJ0eXBlIjoib2JqZWN0IiwicHJvcGVydGllcyI6eyJ1c2VySWQiOnsidHlwZSI6Im51bWJlciJ9LCJpZCI6eyJ0eXBlIjoibnVtYmVyIn0sInR
pdGxlIjp7InR5cGUiOiJzdHJpbmcifSwiYm9keSI6eyJ0eXBlIjoic3RyaW5nIn19fX0=;Encoding=utf-8",
  "subscriptionId": "604f52e1261a3c19104c0e25",
  "connectionType": "JSON"
}
```

Where

- `name` — the name of the data source (will be displayed in the designer when selected).
- `connectionString` — connection string, in the case of JSON, it consists of 3 items:
  - `Json` — a JSON file or a http/https link encoded in base64;
  - `JsonSchema` — a schema that describes the structure of a JSON file, encoded in base64;
  - `Encoding` — encoding, you should always pass `utf-8`.
- `subscriptionId` — the id of the workspace (subscription) to which a data source will be attached.
- `connectionType` — connection type, `JSON` is used in this guide.

Request example.

```
curl -X POST "https://https://fastreport.cloud/api/data/v1/DataSources" -H "accept: text/plain" -H "Content-Type:
application/json-patch+json" -d "{ \"name\": \"fakeAPI\", \"connectionString\":
\"Json=aHR0cHM6Ly9qc29ucGxhY2Vob2xkZXIudHlwaWNvZGUuY29tL3Bvc3Rz;JsonSchema=eyJ0eXBlIjoiYXJyYXkiLCJpdGVtcy
I6eyJ0eXBlIjoib2JqZWN0IiwicHJvcGVydGllcyI6eyJ1c2VySWQiOnsidHlwZSI6Im51bWJlciJ9LCJpZCI6eyJ0eXBlIjoibnVtYmVyIn0sInR
pdGxlIjp7InR5cGUiOiJzdHJpbmcifSwiYm9keSI6eyJ0eXBlIjoic3RyaW5nIn19fX0=;Encoding=utf-8\", \"subscriptionId\":
\"604f52e1261a3c19104c0e25\", \"connectionType\": \"JSON\"}"
```

Response example.


```
{
  "id": "60648953db44d83f9c6da98f",
  "name": "fakeAPI",
  "connectionType": "JSON",
  "connectionString":
"Json=aHR0cHM6Ly9qc29ucGxhY2Vob2xkZXIudHlwaWNvZGUuY29tL3Bvc3Rz;JsonSchema=eyJ0eXBlIjoiYXJyYXkiLCJpdGVtcyI
6eyJ0eXBlIjoib2JqZWN0IiwicHJvcGVydGllcyI6eyJ1c2VySWQiOnsidHlwZSI6Im51bWJlciJ9LCJpZCI6eyJ0eXBlIjoibnVtYmVyIn0sInR
pdGxlIjp7InR5cGUiOiJzdHJpbmcifSwiYm9keSI6eyJ0eXBlIjoic3RyaW5nIn19fX0=;Encoding=utf-8",
  "dataStructure": "<JsonDataSourceConnection
ConnectionString=\"rijcmlqrcq6OJBTPt0pNFvRuRtDUSHSHLQy/QlNolifTTaTjsrExzdbf1ifpPblp655sducwkD1lEVzxVZF8qRuE0NT
6UkyTr7kwjGItFOwh7DBsOyL6QkQY4FOZ2ki8Al2R30gpXs6nMUGg1BRwCF0rj3+QvmXbj+2t8x5RerR5y7inP1R+oCuo0wvfcTe
OMfYfZrjdE3whziFh5Qn3mR7vaevmV9peDWQ3LYyK2ec3KpGVeEXSqM+10WyL4ahY7EHuQIzIZROGFGKfW50cUYwdillhKy24g
NdsUzi5klG66DDQtCKEOLbNutDvA0xqCTW3MvRNORSbvckL6g3gM+cStJ5PQ2XUjF9yz9zdwmramnXI6k+MK8V9lrMkc0XFkDM
HOxDlfG2jHhkFuUTgmiKp7hQMg==\">\r\n  <JsonTableDataSource Name=\"JSON\"
DataType=\"FastReport.Data.JsonConnection.JsonParser.JsonArray\" Enabled=\"true\" TableName=\"JSON\">\r\n    <Column
Name=\"index\" DataType=\"System.Int32\"/>\r\n    <Column Name=\"item\" DataType=\"FastReport.JsonBase\">\r\n
<Column Name=\"userId\" DataType=\"System.Double\"/>\r\n      <Column Name=\"id\" DataType=\"System.Double\"/>\r\n
<Column Name=\"title\" DataType=\"System.String\"/>\r\n      <Column Name=\"body\" DataType=\"System.String\"/>\r\n
</Column>\r\n    <Column Name=\"array\" DataType=\"FastReport.JsonBase\"/>\r\n
</JsonTableDataSource>\r\n</JsonDataSourceConnection>\r\n",
  "subscriptionId": "604f52e1261a3c19104c0e25",
  "editedTime": "2021-03-31T14:38:10.5792982Z",
  "editorUserId": "2df79f83-07f1-41ba-96b5-7757bbf377df",
  "createdTime": "0001-01-01T00:00:00",
  "creatorUserId": "2df79f83-07f1-41ba-96b5-7757bbf377df",
  "isConnected": true
}
```

Permission management on the example of a template

Restricting access to private resources is a very important part of FastReport Cloud. A flexible access system allows you to set a limit or grant rights to each resource separately, specifying the circle of people who can access them.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Report template.

   You can learn how to upload a report template in the article Uploading a new template.

4. Active FastReport Cloud subscription.

5. Access to the Internet.

Instruction

1. To view the current permissions for a resource, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Templates/File/{id}/permissions`, where `{id}` should be replaced with the template identifier.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/File/5fc9ece6b792c90001d94b13/permissions" -H "accept: text/plain"
   ```

   Response example.

```
{
 "permissions": {
  "ownerId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "owner": {
   "create": "All",
   "delete": "All",
   "execute": "All",
   "get": "All",
   "update": "All",
   "administrate": "All"
  },
  "groups": null,
  "other": {
   "create": "All",
   "delete": "All",
   "execute": "All",
   "get": "All",
   "update": "All",
   "administrate": "All"
  },
  "anon": null
 }
}
```

In this example, the owner and other workspace members have full access to the report template.

2. To change permissions, make a `POST` request to `https://fastreport.cloud/api/rp/v1/Templates/File/{id}/permissions`, where {id} should be replaced with the template identifier.

   Model example.

```
{
 "newPermissions": {
  "ownerId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "owner": {
   "create": -1,
   "delete": -1,
   "execute": -1,
   "get": -1,
   "update": 0,
   "administrate": -1
  },
  "other": {
   "create": 0,
   "delete": 0,
   "execute": 0,
   "get": 0,
   "update": 0,
   "administrate": 0
  }
 },
 "administrate": "Owner,Other"
}
```

   Note! The `Owner` and `Other` values were passed in the request to update permissions in the `administrate` property. In this case, only the specified permission categories, i.e., `owner` and `other`, will be updated, while the `anon` and `groups` categories will not be changed.

   Here is an example of a request that removes the owner's report update permission (e.g., editing in the online designer), as well as all permissions on the file from other workspace members.

```
curl -X POST "https://fastreport.cloud/api/rp/v1/Templates/File/5fc9ece6b792c90001d94b13/permissions" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"newPermissions\": { \"ownerId\": \"5af5a8dc-8cb0-40f9-ac99-ca2533fa4491\", \"owner\": { \"create\": -1, \"delete\": -1, \"execute\": -1, \"get\": -1, \"update\": 0, \"administrate\": -1 }, \"other\": { \"create\": 0, \"delete\": 0, \"execute\": 0, \"get\": 0, \"update\": 0, \"administrate\": 0 } }, \"administrate\": \"Owner,Other\"}
```

Response example.

```
200 OK
```

3. You can now request permissions again to make sure they have been updated.

   Request example.

```
curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/File/5fc9ece6b792c90001d94b13/permissions" -H "accept: text/plain"
```

   Response example.


```
{
 "permissions": {
   "ownerId": "2df79f83-07f1-41ba-96b5-7757bbf377df",
 "owner": {
     "create": "All",
     "delete": "All",
     "execute": "All",
  "get": "All",
     "update": "None",
  "administrate": "All"
   },
   "groups": null,
   "other": {
     "create": "None",
     "delete": "None",
     "execute": "None",
     "get": "None",
     "update": "None",
     "administrate": "None"
   },
   "anon": null
 }
}
```


What's next?

- Building a report.
- Working with groups.
- Adding new users to a workspace.

Building a report

This article walks you through the process of building a report from a template with the FastReport Cloud report processor.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Report template.

   You can find how to upload a report template in the article Uploading a new template.

4. Active FastReport Cloud subscription.

5. Access to the Internet.

Instruction

1. You need a template identifier to build it. Make a `GET` request to `https://fastreport.cloud/api/rp/v1/Templates/Root` to get the root directory.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/Root" -H "accept: text/plain"
   ```

   Response example.

   ```
   {
    "name": "RootFolder",
    "parentId": null,
    "tags": [],
    "icon": "",
    "type": "Folder",
    "size": 16384,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "None",
    "id": "5fa919f9292a8300019349b9",
    "createdTime": "2020-11-09T10:29:13.928Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "2020-11-13T15:58:45.69Z",
    "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
   }
   ```

   The directory identifier from the example above is `5fa919f9292a8300019349b9`.

2. To get a list of files in a directory, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Templates/Folder/{id}/ListFiles?skip=0&take=10`, where `{id}` should be replaced with the directory identifier.

   Request example.

```
curl -X GET "https://fastreport.cloud/api/rp/v1/Templates/Folder/5fa919f9292a8300019349b9/ListFiles?
skip=0&take=10" -H "accept: text/plain"
```

Response example.


```
[
  {
    "reportInfo": {
      "author": null,
      "created": "2020-12-04T10:58:57Z",
      "creatorVersion": "20.20.4.1",
      "description": null,
      "modified": "2020-12-04T11:00:20Z",
      "name": null,
      "picture": null,
      "previewPictureRatio": 0,
      "saveMode": "All",
      "savePreviewPicture": false,
      "tag": null,
      "version": null
    },
    "name": "template.frx",
    "parentId": "5fa919f9292a8300019349b9",
    "tags": null,
    "icon": null,
    "type": "File",
    "size": 17159,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "Success",
    "id": "5fc9ece6b792c90001d94b13",
    "createdTime": "2020-12-04T08:01:42.708Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "2020-12-04T08:01:42.708Z",
    "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
  }
]
```


The template identifier from the example above is `5fc9ece6b792c90001d94b13`.

3. To build the report, a directory will be needed to put the report to. Request the report root directory by making a `GET` request to `https://fastreport.cloud/api/rp/v1/Reports/Root`.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Reports/Root" -H "accept: text/plain"
   ```

   Response example.


   ```
   {
    "name": "RootFolder",
    "parentId": null,
    "tags": null,
    "icon": null,
    "type": "Folder",
    "size": 16384,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "None",
    "id": "5fa919f9292a8300019349ba",
    "createdTime": "2020-11-09T10:29:13.993Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "0001-01-01T00:00:00Z",
    "editorUserId": null
   }
   ```

The directory identifier from the example above is `5fa919f9292a8300019349ba`.

4. To build a report, make a `POST` request to `https://fastreport.cloud/api/rp/v1/Templates/File/{id}/Prepare`, where `{id}` should be replaced with the template identifier.

   In the request body, pass the JSON as shown below.

```
{
  "name": "string",
  "folderId": "string",
  "reportParameters": {
    "additionalProp1": "string",
    "additionalProp2": "string",
    "additionalProp3": "string"
  }
}
```

   - `folderId` — identifier of the directory where the report will be placed.
   - `name` — the name of the resulting file. Add `.fpx` extension manually.
   - `reportParameters` — parameters that are specified in the report itself using the report designer, we do not need them in this example.

> If you do not specify folderId, then the prepared report will be saved to the root folder.

   Request example.

```
curl -X POST "https://fastreport.cloud/api/rp/v1/Templates/File/5fc9ece6b792c90001d94b13/Prepare" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"name\": \"awesome_report.fpx\", \"folderId\": \"5fa919f9292a8300019349ba\"}"
```

   Response example.

```
{
  "templateId": "5fc9ece6b792c90001d94b13",
  "reportInfo": null,
  "name": "awesome_report.fpx",
  "parentId": "5fa919f9292a8300019349ba",
  "tags": null,
  "icon": null,
  "type": "File",
  "size": 16384,
  "subscriptionId": "5fa919fa292a8300019349bc",
  "status": "InQueue",
  "id": "5fe4614bcd7c55000148e4c6",
  "createdTime": "2020-12-24T09:37:15.7169531+00:00",
  "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "editedTime": "2020-12-24T09:37:15.7169582+00:00",
  "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
}
```

   Note the status of the `InQueue` file, it means that the build task was created and it has been placed in the builder queue. At this stage, the report already received its `5fe4614bcd7c55000148e4c6` identifier for work.

   You should wait some time until the report is built. You can call the get report method in a loop every few milliseconds and check the status.

5. To get information about the report, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Reports/File/{id}`, where `{id}` should be replaced with the report identifier.

   Request example.

```
curl -X GET "https://fastreport.cloud/api/rp/v1/Reports/File/5fe4614bcd7c55000148e4c6" -H "accept: text/plain"
```

Response example.


```
{
 "templateId": "5fc9ece6b792c90001d94b13",
 "reportInfo": null,
 "name": "awesome_report.fpx",
 "parentId": "5fa919f9292a8300019349ba",
 "tags": null,
 "icon": null,
 "type": "File",
 "size": 16927,
 "subscriptionId": "5fa919fa292a8300019349bc",
 "status": "Success",
 "id": "5fe4614bcd7c55000148e4c6",
 "createdTime": "2020-12-24T09:37:15.716Z",
 "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
 "editedTime": "2020-12-24T09:37:15.716Z",
 "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
}
```


Note the `Success` status of the file. The report has been successfully built.

6. To download the report, make a `GET` request to `https://fastreport.cloud/download/r/{id}`, where `{id}` should be replaced with the report identifier.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/download/r/5fe4614bcd7c55000148e4c6" -H "accept: text/plain"
   ```

   The file will be in the response.

What's next?

- Exporting a report to PDF.

Report parameters

In this article, we will look at the process of passing parameters to a report, which represent a dictionary of parameters provided when generating the report. More information on this topic can be found in the FastReport .NET manual in the section Report parameters.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Report template with the specified report parameters.

   It can be created using the free program FastReport Community Designer.

4. Active FastReport Cloud subscription.

5. Access to the Internet.

Passing a value to a report parameter

Parameters can be passed in the following cases:

- Prepare template;
- Export template;
- Working with tasks.

Let's look at the passing of parameters using an example of preparing a template. Detailed instructions on how to prepare a report can be found in the section Building a report.

To build a report, make a `POST` request to `https://fastreport.cloud/api/rp/v1/Templates/File/{id}/Prepare`, where `{id}` should be replaced with the template identifier.

In the request body, pass the JSON as shown below.

```
{
 "name": "string",
 "folderId": "string",
 "reportParameters": {
  "additionalProp1": "string",
  "additionalProp2": "string"
 }
}
```

- `folderId` — identifier of the directory where the report will be placed.
- `name` — the name of the resulting file. Add `.fpx` extension manually.
- `reportParameters` — parameters that are specified in the report itself using the report designer.

> If you do not specify folderId, then the prepared report will be saved to the root folder.

Request example.

```
curl -X POST "https://fastreport.cloud/api/rp/v1/Templates/File/5fc9ece6b792c90001d94b13/Prepare"
-H "accept: text/plain"
-H "Content-Type: application/json-patch+json"
-d "{
 "name": "awesome_report.fpx",
 "folderId": "5fa919f9292a8300019349ba",
 "reportParameters": {
   "Parameter1": "Value1",
   "Parameter2": "Value2"
   }
 }"
```

Response example.


```
{
  "templateId": "5fc9ece6b792c90001d94b13",
  "name": "awesome_report.fpx",
  "parentId": "5fa919f9292a8300019349ba",
  "type": "File",
  "size": 16384,
  "subscriptionId": "5fa919fa292a8300019349bc",
  "status": "InQueue",
  "statusReason": "None",
  "id": "5fe4614bcd7c55000148e4c6",
  "createdTime": "2020-12-24T09:37:15.7169531+00:00",
  "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "editedTime": "2020-12-24T09:37:15.7169582+00:00",
  "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
}
```

Exporting a report to PDF

This article walks you through the process of exporting a report with the FastReport Cloud report processor.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud. You can find how to get and use an API key in the article Authentication and authorization

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Report.

   You can find how to build a report in the article Building a report.

4. Active FastReport Cloud subscription.

5. Access to the Internet.

Comment

Please note that the report can be exported directly from the template, without intermediate saving of the report. To do this, run the same commands for the report template, replacing `Report` in the request strings with `Template`, also use the template identifier, not that of the report.

Instruction

1. You will need a report identifier to export to PDF. Make a `GET` request to `https://fastreport.cloud/api/rp/v1/Report/Root` to get the root directory.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Reports/Root" -H "accept: text/plain"
   ```

   Response example.

   ```
   {
    "name": "RootFolder",
    "parentId": null,
    "tags": null,
    "icon": null,
    "type": "Folder",
    "size": 16384,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "None",
    "id": "5fa919f9292a8300019349ba",
    "createdTime": "2020-11-09T10:29:13.993Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "0001-01-01T00:00:00Z",
    "editorUserId": null
   }
   ```

   The directory identifier from the example above is `5fa919f9292a8300019349ba`.

2. 2. Get a list of files in the directory, to do this, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Reports/Folder/{id}/ListFiles?skip=0&take=10`, where `{id}` should be replaced with a directory identifier.

Request example.

```
curl -X GET "https://fastreport.cloud/api/rp/v1/Reports/Folder/5fa919f9292a8300019349ba/ListFiles?skip=0&take=10" -H "accept: text/plain"
```

Response example.

```
[
  {
    "templateId": "5fc9ece6b792c90001d94b13",
    "reportInfo": null,
    "name": "awesome_report.fpx",
    "parentId": "5fa919f9292a8300019349ba",
    "tags": null,
    "icon": null,
    "type": "File",
    "size": 16927,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "Success",
    "id": "5fe4614bcd7c55000148e4c6",
    "createdTime": "2020-12-24T09:37:15.716Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "2020-12-24T09:37:15.716Z",
    "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
  }
]
```

The report identifier from the example above is `5fe4614bcd7c55000148e4c6`.

3. To export a report, a directory will be needed to put the export file to.

   Get the exports root directory by making a `GET` request to `https://fastreport.cloud/api/rp/v1/Exports/Root`.

   Request example.

```
curl -X GET "https://fastreport.cloud/api/rp/v1/Exports/Root" -H "accept: text/plain"
```

   Response example.

```
{
  "name": "RootFolder",
  "parentId": null,
  "tags": null,
  "icon": null,
  "type": "Folder",
  "size": 16384,
  "subscriptionId": "5fa919fa292a8300019349bc",
  "status": "None",
  "id": "5fa919fa292a8300019349bb",
  "createdTime": "2020-11-09T10:29:14.002Z",
  "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
  "editedTime": "0001-01-01T00:00:00Z",
  "editorUserId": null
}
```

   The directory identifier from the example above is `5fa919fa292a8300019349bb`.

4. To export a report, make a `POST` request to `https://fastreport.cloud/api/rp/v1/Reports/File/{id}/Export`, where

`{id}` should be replaced with the report identifier.

In the request body, pass the JSON as shown below.

```
{
 "fileName": "awesome_result.pdf",
 "folderId": "5fa919fa292a8300019349bb",
 "locale": "en-GB",
 "format": "Pdf",
 "exportParameters": {
  "additionalProp1": {},
  "additionalProp2": {},
  "additionalProp3": {}
 }
}
```

- `folderId` — identifier of the directory where the export will be placed. If left blank, the export will be placed in the exports root folder in the workspace.
- `fileName` — the name of the resulting file. If you do not specify the extension or specify it incorrectly, the server will replace it automatically.
- `locale` — localization of the exported report. This option will change the date and number formats to match the selected ISO culture code (for example, French (Switzerland) looks like this—"fr-CH"). Leaving this field blank or specifying a culture that does not exist will substitute the default locale from the subscription or English (United States) if no default locale is specified.
- `format` — export format.
- `exportParameters` — export parameters. They are set similarly to the export parameters from the FastReport .NET library. A more detailed description is available in the User Manual in the section export parameters.

> If you do not specify folderId, then the prepared report will be saved to the root folder.

Request example.

```
curl -X POST "https://fastreport.cloud/api/rp/v1/Reports/File/5fe4614bcd7c55000148e4c6/Export" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"fileName\": \"awesome_result.pdf\", \"folderId\": \"5fa919fa292a8300019349bb\", \"locale\":\"ru-RU\", \"format\": \"Pdf\"}"
```

Response example.

```
{
 "format": "Pdf",
 "reportId": "5fe4614bcd7c55000148e4c6",
 "name": "awesome_result.pdf",
 "parentId": "5fa919fa292a8300019349bb",
 "tags": null,
 "icon": null,
 "type": "File",
 "size": 16384,
 "subscriptionId": "5fa919fa292a8300019349bc",
 "status": "InQueue",
 "id": "5fe46a33cd7c55000148e4c7",
 "createdTime": "2020-12-24T10:15:15.8039648+00:00",
 "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
 "editedTime": "2020-12-24T10:15:15.8039697+00:00",
 "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
}
```

Note the status of the `InQueue` file, it means that an export task was created and it has been placed in the builder queue. At this stage the file already received its `5fe46a33cd7c55000148e4c7` identifier for work.

You should wait some time until the export is built.

5. To get information about the file, make a `GET` request to `https://fastreport.cloud/api/rp/v1/Exports/File/{id}`, where `{id}` should be replaced with the export identifier.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/rp/v1/Exports/File/5fe46a33cd7c55000148e4c7" -H "accept: text/plain"
   ```

   Response example.

   ```
   {
    "format": "Pdf",
    "reportId": "5fe4614bcd7c55000148e4c6",
    "name": "awesome_result.pdf",
    "parentId": "5fa919fa292a8300019349bb",
    "tags": null,
    "icon": null,
    "type": "File",
    "size": 41142,
    "subscriptionId": "5fa919fa292a8300019349bc",
    "status": "Success",
    "id": "5fe46a33cd7c55000148e4c7",
    "createdTime": "2020-12-24T10:15:15.803Z",
    "creatorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "editedTime": "2020-12-24T10:15:15.803Z",
    "editorUserId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
   }
   ```

   Note the `Success` status of the file. The report has been successfully exported.

6. To download the report, make a `GET` request to `https://fastreport.cloud/download/e/{id}`, where instead of `{id}` you should pass the report identifier.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/download/e/5fe46a33cd7c55000148e4c7" -H "accept: text/plain"
   ```

   The file will be in the response.

What's next?

- Working with groups.
- Adding new users to a workspace.

Adding new users to a workspace

This article walks you through the process of adding a new user to a workspace, getting a list of users in a workspace, and removing a user from a workspace.

> Each workspace always has its subscription. They share a common identifier.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription that has at least two user slots.

4. Access to the Internet.

Instruction

There are two ways to add a user to a workspace:

- Directly by specifying the user id.
- Through the invitation system.

Adding a user via an invite

1. A workspace identifier is required to create an invitation.

   Get the workspace identifier by making a `GET` request to `https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10`.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10" -H "accept: text/plain"
   ```

   Response example.

```
{
 "subscriptions": [
  {
   "id": "604f52e1261a3c19104c0e25",
   "name": "iwantpizza",
   "locale": null,
   "current": {
    "startTime": "2021-03-15T12:28:17.773Z",
    "endTime": "2121-03-15T12:28:17.773Z",
    "plan": {
     "id": "5f89b4d722d2d823440b6d10",
     "isActive": false,
     "displayName": "unlimited",
     "timePeriodType": "Year",
     "timePeriod": 100,
     "readonlyTimeLimitType": "Second",
     "readonlyTimeLimit": 0,
     "templatesSpaceLimit": 1048576000,
     "reportsSpaceLimit": 1048576000,
     "exportsSpaceLimit": 1048576000,
     "fileUploadSizeLimit": 1073741824,
     "dataSourceLimit": 10,
     "maxUsersCount": 10,
     "groupLimit": 5,
     "onlineDesigner": true,
     "isDemo": false,
     "urlToBuy": "https://fastreport.ru",
     "unlimitedPage": true,
     "pageLimit": 0
    }
   },
   "old": null,
   "invites": null,
   "templatesFolder": {
    "folderId": "604f52e1261a3c19104c0e22",
    "bytesUsed": 241247
   },
   "reportsFolder": {
    "folderId": "604f52e1261a3c19104c0e23",
    "bytesUsed": 16384
   },
   "exportsFolder": {
    "folderId": "604f52e1261a3c19104c0e24",
    "bytesUsed": 8059419
   }
  }
 ],
 "count": 1,
 "skip": 0,
 "take": 10
}
```


The workspace (subscription) ID from the example above is `604f52e1261a3c19104c0e25`.

2. To create an invitation, make a `POST` request to `https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/invite`, where {subscriptionId} should be replaced with the workspace identifier.

   By specifying the invitation properties in the request body.

   Body example.

```
{
 "usages": 1,
 "durable": true,
 "expiredDate": "2021-03-25T11:53:29.351Z"
}
```

- `usages` mean the possible number of uses,
- `durable` set to true indicates that ***the number*** of uses is unlimited,
- `expiredDate` specifies the expiration date for the invitation (the `durable` property will not make the invitation perpetual).

  Leaving the body empty will create a one-time invite that stops working the next day.

Request example.

```
curl -X POST "https://fastreport.cloud/api/manage/v1/Subscriptions/604f52e1261a3c19104c0e25/invite" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"usages\": 1, \"durable\": true, \"expiredDate\": \"2021-03-25T11:53:29.351Z\"}"
```

Response example.

```
{
 "id": "604f52e1261a3c19104c0e25",
 "name": "iwantpizza",
 "locale": null,
 "current": {
  "startTime": "2021-03-15T12:28:17.773Z",
  "endTime": "2121-03-15T12:28:17.773Z",
  "plan": {
   "id": "5f89b4d722d2d823440b6d10",
   "isActive": false,
   "displayName": "unlimited",
   "timePeriodType": "Year",
   "timePeriod": 100,
   "readonlyTimeLimitType": "Second",
   "readonlyTimeLimit": 0,
   "templatesSpaceLimit": 1048576000,
   "reportsSpaceLimit": 1048576000,
   "exportsSpaceLimit": 1048576000,
   "fileUploadSizeLimit": 1073741824,
   "dataSourceLimit": 10,
   "maxUsersCount": 10,
   "groupLimit": 5,
   "onlineDesigner": true,
   "isDemo": false,
   "urlToBuy": "https://fast-report.com",
   "unlimitedPage": true,
   "pageLimit": 0
  }
 },
 "old": null,
 "invites": [
  {
   "usages": 1,
   "durable": true,
   "accessToken": "fj3534g341ir7dyytfiaap9z1r",
   "expiredDate": "2021-03-25T11:53:29.351Z",
   "addedUsers": [],
   "creatorUserId": "2df79f83-07f1-41ba-96b5-7757bbf377df"
  }
 ],
 "templatesFolder": {
  "folderId": "604f52e1261a3c19104c0e22",
  "bytesUsed": 241247
 },
 "reportsFolder": {
  "folderId": "604f52e1261a3c19104c0e23",
  "bytesUsed": 16384
 },
 "exportsFolder": {
  "folderId": "604f52e1261a3c19104c0e24",
  "bytesUsed": 8059419
 }
}
```


An invitation appeared in the subscription with the `fj3534g341ir7dyytfiaap9z1r` access token.

3. You need to send the link for accepting the invitation (or the access token itself) to the user you want to invite (in any way convenient for you). The final link will look like this — `https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/invite/{accessToken}/accept`, where instead of `{subscriptionId}`, you will need to specify the workspace identifier, and instead of `{accessToken}` — access token.

4. Now the invited user can follow this link directly in the browser or send a `GET` request to

`https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/invite/{accessToken}/accept`, where instead of `{subscriptionId}` you will need to specify the workspace identifier, and instead of `{accessToken}` — an access token.

Request example.

```
curl -X GET
"https://fastreport.cloud/api/manage/v1/Subscriptions/6051f2a06c07a10001737394/invite/to9kxrxdz4iwbfyz3pq4fktdcr
/accept" -H "accept: text/plain"
```

Deleting an invitation

- • When an invitation runs out of uses or expires, it is not deleted automatically. You will need to do this manually by sending a `DELETE` request to `https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/invite/{accessToken}` Request example.

```
curl -X DELETE
"https://fastreport.cloud/api/manage/v1/Subscriptions/6051f2a06c07a10001737394/invite/to9kxrxdz4iwbfyz3pq4fktdcr
" -H "accept: text/plain"
```

The response will be an empty message with the `NO CONTENT 204` code.

Adding a user directly

1. To add a new user to a workspace, a workspace identifier is required.

   Get the subscription identifier by making a `GET` request to `https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10`.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10" -H "accept: text/plain"
   ```

   Response example.

```
{
 "subscriptions": [
  {
   "id": "5fa919fa292a8300019349bc",
   "name": "Awesome Corp",
   "current": {
    "startTime": "2020-11-17T10:22:58.584Z",
    "endTime": "2025-11-17T10:22:58.584Z",
    "plan": {
     "id": "5f43924b0231500001225686",
     "isActive": false,
     "displayName": "The greatest power",
     "timePeriodType": "Year",
     "timePeriod": 5,
     "readonlyTimeLimitType": "Second",
     "readonlyTimeLimit": 0,
     "templatesSpaceLimit": 1048576000,
     "reportsSpaceLimit": 1048576000,
     "exportsSpaceLimit": 1048576000,
     "fileUploadSizeLimit": 1048576000000,
     "dataSourceLimit": 10,
     "maxUsersCount": 10,
     "groupLimit": 5,
     "onlineDesigner": true,
     "isDemo": false,
     "urlToBuy": "https://fast-report.com",
     "unlimitedPage": true,
     "pageLimit": 15
    }
   },
   "old": [],
   "templatesFolder": {
    "folderId": "5fa919f9292a8300019349b9",
    "bytesUsed": 1668491
   },
   "reportsFolder": {
    "folderId": "5fa919f9292a8300019349ba",
    "bytesUsed": 6085990
   },
   "exportsFolder": {
    "folderId": "5fa919fa292a8300019349bb",
    "bytesUsed": 8336710
   }
  }
 ],
 "count": 1,
 "skip": 0,
 "take": 10
}
```


The workspace (subscription) identifier from the example above is `5fa919fa292a8300019349bc`.

2. To add a new user, make a `PUT` request to `https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/users/{userId}`, where instead of `{subscriptionId}` you should pass the workspace identifier, and instead of `{userId}` you should pass user identifier.

   Request example.

```
curl -X PUT "https://fastreport.cloud/api/manage/v1/Subscriptions/5fa919fa292a8300019349bc/users/5af5a8dc-8cb0-40f9-ac99-ca2533fa4492" -H "accept: text/plain"
```

   In response, you will receive an empty message with the `OK 200` status code.

3. To get a list of users, make a `GET` request to `https://fastreport.cloud/api/manage/v1/Subscriptions/{id}/users?skip=0&take=10`, where `{id}` should be replaced with the workspace identifier.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/manage/v1/Subscriptions/5fa919fa292a8300019349bc/users?skip=0&take=10" -H "accept: text/plain"
   ```

   Response example.

   ```
   {
    "users": [
     {
      "userId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491"
     },
     {
      "userId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4492"
     }
    ],
    "count": 2,
    "take": 10,
    "skip": 0
   }
   ```

4. To remove a user from a workspace, make a `DELETE` request to `https://fastreport.cloud/api/manage/v1/Subscriptions/{subscriptionId}/users/{userId}`, where instead of `{subscriptionId}` you should pass the workspace identifier, and instead of `{userId}` you should pass user identifier.

   Request example.

   ```
   curl -X DELETE "https://fastreport.cloud/api/manage/v1/Subscriptions/5fa919fa292a8300019349bc/users/5af5a8dc-8cb0-40f9-ac99-ca2533fa4492" -H "accept: text/plain"
   ```

   The response will be an empty message with the `OK 200` code.

What's next?

- Working with groups
- Help and feedback

Working with groups

This article walks you through the process of creating a new group, adding a user to the group, and getting a list of the group's users.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription that has two user slots.

4. Access to the Internet.

Comment

Note! It is only possible to add a user to a group if the user exists in the workspace.

Note! It is only possible to add a user to a group by its identifier.

Instruction

1. To create a new group, you need a workspace identifier and a name for the new group.

   Get the workspace identifier by making a `GET` request to `https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10`.

   Request example.

   ```
   curl -X GET "https://fastreport.cloud/api/manage/v1/Subscriptions?skip=0&take=10" -H "accept: text/plain"
   ```

   Response example.

```
{
 "subscriptions": [
  {
   "id": "5fa919fa292a8300019349bc",
   "name": "Awesome Corp",
   "current": {
    "startTime": "2020-11-17T10:22:58.584Z",
    "endTime": "2025-11-17T10:22:58.584Z",
    "plan": {
     "id": "5f43924b0231500001225686",
     "isActive": false,
     "displayName": "The greatest power",
     "timePeriodType": "Year",
     "timePeriod": 5,
     "readonlyTimeLimitType": "Second",
     "readonlyTimeLimit": 0,
     "templatesSpaceLimit": 1048576000,
     "reportsSpaceLimit": 1048576000,
     "exportsSpaceLimit": 1048576000,
     "fileUploadSizeLimit": 1048576000000,
     "dataSourceLimit": 10,
     "maxUsersCount": 10,
     "groupLimit": 5,
     "onlineDesigner": true,
     "isDemo": false,
     "urlToBuy": "https://fast-report.com/",
     "unlimitedPage": true,
     "pageLimit": 15
    }
   },
   "old": [],
   "templatesFolder": {
    "folderId": "5fa919f9292a8300019349b9",
    "bytesUsed": 1668491
   },
   "reportsFolder": {
    "folderId": "5fa919f9292a8300019349ba",
    "bytesUsed": 6085990
   },
   "exportsFolder": {
    "folderId": "5fa919fa292a8300019349bb",
    "bytesUsed": 8336710
   }
  }
 ],
 "count": 1,
 "skip": 0,
 "take": 10
}
```


The workspace (subscription) identifier from the example above is `5fa919fa292a8300019349bc`.

2. 2. To create a new group, make a `POST` request to `https://fastreport.cloud/api/manage/v1/Groups`, pass JSON in the request body as shown below.


```
{
 "name": "string",
 "subscriptionId": "string id"
}
```


Request example.

```
curl -X POST "https://fastreport.cloud/api/manage/v1/Groups" -H "accept: text/plain" -H "Content-Type: application/json-patch+json" -d "{ \"name\": \"My first group\", \"subscriptionId\": \"5fa919fa292a8300019349bc\"}"
```

Response example.

```
{
 "id": "5fe5d7866882ca0001760fcb",
 "name": "My first group",
 "subscriptionId": "5fa919fa292a8300019349bc"
}
```

The group identifier from the example above is `5fe5d7866882ca0001760fcb`.

3. 3. To add a new user to a group, make a `PUT` request to `https://fastreport.cloud/api/manage/v1/Groups/{groupId}/Users/{userId}`, instead of `{groupId}` enter the group identifier, and instead of `{userId}` enter the user identifier.

Request example.

```
curl -X PUT "https://fastreport.cloud/api/manage/v1/Groups/5fe5d7866882ca0001760fcb/Users/5af5a8dc-8cb0-40f9-ac99-ca2533fa4492" -H "accept: text/plain"
```

The response will be an empty message with the `OK 200`.

4. 4. To get a list of users in a group, make a `GET` request to `https://fastreport.cloud/api/manage/v1/Groups/{id}/Users?skip=0&take=10`, where `{id}` should be replaced with the group identifier.

Request example.

```
curl -X GET "https://fastreport.cloud/api/manage/v1/Groups/5fe5d7866882ca0001760fcb/Users?skip=0&take=10" -H "accept: text/plain"
```

Response example.

```
{
 "users": [
   {
    "userId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4491",
    "userId": "5af5a8dc-8cb0-40f9-ac99-ca2533fa4492"
   }
 ],
 "count": 2,
 "take": 10,
 "skip": 0
}
```

What's next?

- Help and feedback

Working with tasks

Tasks in FastReport Cloud are actions for converting and delivering documents to consumers. They are described in detail in the Tasks section. Here we will look at how to work with them.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription.

4. Access to the Internet.

General information

In this article we will look at how to work with tasks.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription.

4. Access to the Internet.

About ViewModels and types

On work you will pass different ViewModels in JSON. It is important that the first field of any ViewModel must be "$t": "Name". Detailed examples will be provided later in this article.

VMs for tasks creation:

- CreatePrepareTaskVM
- CreateExportTemplateTaskVM
- CreateExportReportTaskVM
- CreateFetchTaskVM
- CreateEmailTaskVM
- CreateWebhookTaskVM
- CreateFTPUploadTaskVM

VMs for raw tasks run:

- RunPrepareTaskVM
- RunExportTemplateTaskVM
- RunExportReportTaskVM
- RunFetchTaskVM
- RunEmailTaskVM
- RunWebhookTaskVM
- RunFTPUploadTaskVM

VMs for tasks' fields update:

- UpdatePrepareTaskVM
- UpdateExportTemplateTaskVM
- UpdateExportReportTaskVM
- UpdateFetchTaskVM
- UpdateEmailTaskVM
- UpdateWebhookTaskVM
- UpdateFTPUploadTaskVM

VM for tasks' permissions update:

- UpdateTaskPermissionsVM

Creating a task

To create tasks, use the CreateTask method. It can take any VM for task creation.

Let’s look at how to create a task of exporting a report and then sending it to Webhook:

```
curl -X 'POST' \
 'https://fastreport.cloud/api/tasks/v1/tasks' \
 -H 'accept: application/json' \
 -H 'Content-Type: application/json' \
 -d '{
 "$t": "CreateExportTemplateTaskVM",
 "subscriptionId": "23e0134c816935c1e11b3737",
 "name": "SendViaWebhookExport",
 "inputFile": {
  "entityId": "61e0134c816935c1e11b3787"
 },
 "transports": [
  {
   "$t": "CreateWebhookTaskVM",
   "Endpoints": [
{
  "BearerToken": "allotoken",
       "Url": "https://localhost:7104/api",
       "Headers": {"header1":"val1", "header2":"val2"}
}
   ]
  }
 ],
 "format": "Pdf"
}'
```

The EntityId and FolderId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

> Note! If you do not specify OutputFile, it will be saved in the temporary folder. If you specify an empty OutputFile, it will be saved to the root folder. If you specify the folder ID, it will be saved to that folder.

The list of all available view models is available at the - https://fastreport.cloud/api/swagger/index.html

Getting a list of tasks

```
// Get first 10 tasks from the subscription
curl -X 'GET' \
 'https://fastreport.cloud/api/tasks/v1/tasks?skip=0&take=10' \
 -H 'accept: application/json'
```

Executing a task by the specified identifier

```
curl -X 'POST' \
 'https://fastreport.cloud/api/tasks/v1/tasks/42d134ae3130aaad37by345f/run' \
 -H 'accept: */*' \
 -d ''
```

Deleting a task from storage

```
curl -X 'DELETE' \
  'https://fastreport.cloud/api/tasks/v1/tasks/42d134ae3130aaad37by345f' \
  -H 'accept: */*'
```

> Note! The examples don't have an Authorization header because they use a cookie-based authentication model. Read more about authorization in the Authentication and authorization section.

Upload to FTP server

In this article, we will consider a way to send a report to an FTP server. Instructions on how to work with tasks are described in the General information.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription.

4. Access to the Internet.

5. Configured and accessible FTP server.

> Note! The paragraphs above describe the recommended tools.

> Note! This guide assumes that you have an FTP server configured to accept files from external sources.

Creating a task

Let’s look at how to create a FTP Upload task:

```
// Task creation
curl -X 'POST' \
 'https://fastreport.cloud/api/tasks/v1/Tasks' \
 -H 'accept: application/json' \
 -H 'Content-Type: application/json' \
 -d '{
 "$t": "CreateFTPUploadTaskVM",
 "name": "FTP upload task",
 "subscriptionId": "23e0134c816935c1e11b3737",
 "ftpHost": "ftp://localhost",
 "ftpPort": 21,
 "ftpUsername": "FtpUser",
 "ftpPassword": "password",
 "archive": false,
 "archiveName": "Archive name",
 "useSFTP": false,
 "inputFile": {
   "entityId": "61e0134c816935c1e11b3787",
   "type": "Template"
 },
 "destinationFolder": "/path/"
}'
```

```
// Run task by identifier
curl -X 'POST' \
  '{your_host_name}/api/tasks/v1/Tasks/{taskId}/run' \
  -H 'accept: */*' \
  -d ''
```

The EntityId and SubscriptionId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

Run a task from request body

```
curl -X 'POST' \
 'https://fastreport.cloud/api/tasks/v1/Tasks/run' \
 -H 'accept: application/json' \
 -H 'Content-Type: application/json' \
 -d '{
 "$t": "RunFTPUploadTaskVM",
 "name": "FTP upload task",
 "subscriptionId": "23e0134c816935c1e11b3737",
 "ftpHost": "ftp://localhost",
 "ftpPort": 21,
 "ftpUsername": "FtpUser",
 "ftpPassword": "password",
 "archive": false,
 "archiveName": "Archive name",
 "useSFTP": false,
 "inputFile": {
   "entityId": "61e0134c816935c1e11b3787",
   "type": "Template"
 },
 "destinationFolder": "/path/"
}'
```

> Note! In this case, the FTP transfer will be carried out directly by this request and the task will not be saved in the database.

> Note! The examples don't have an Authorization header because they use a cookie-based authentication model. Read more about authorization in the Authentication and authorization section.

Upload via Webhook

In this article, we will consider a way to send a report via webhook. Instructions on how to work with tasks are described in the General information.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. Curl tool.

   Any other REST client will do, but the examples will be built for curl.

3. Active FastReport Cloud subscription.

4. Access to the Internet.

5. Client application that accepts requests.

> Note! The paragraphs above describe the recommended tools.

> Note! This guide assumes you have a client application that accepts and processes incoming webhooks.

Creating a task

Let’s look at how to create a Webhook task:

```
// Task creation
curl -X 'POST' \
  'https://fastreport.cloud/api/tasks/v1/Tasks' \
  -H 'accept: application/json' \
  -H 'Content-Type: application/json' \
  -d '{
  "$t": "CreateWebhookTaskVM",
  "name": "Webhook task",
  "subscriptionId": "{subscriptionId}",
  "url": "http://example.com/",
  "headers": {
     "Authorization": "Bearer <token>",
     "Content-Type": "multipart/form-data"
  },
  "inputFile": {
    "entityId": "{templateId}",
    "type": "Template"
  }
}'
```

```
// Run task by identifier
curl -X 'POST' \
  'https://fastreport.cloud/api/tasks/v1/Tasks/{taskId}/run' \
  -H 'accept: */*' \
  -d ''
```

> Note! The EntityId and SubscriptionId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

> Note! The identifier of the template, report, and export can be passed to the EntityId.

Run a task from request body

```
curl -X 'POST' \
  'https://fastreport.cloud/api/tasks/v1/Tasks/run' \
  -H 'accept: */*' \
  -H 'Content-Type: application/json' \
  -d '{
  "$t": "RunWebhookTaskVM",
  "name": "Webhook task",
  "subscriptionId": "{subscriptionId}",
  "url": "http://example.com/",
  "headers": {
     "Authorization": "Bearer <token>",
     "Content-Type": "multipart/form-data",
     "Content-Length": "1278"
  },
  "inputFile": {
    "entityId": "{templateId}",
    "type": "Template"
  }
}'
```

> Note! In this case, the Webhook transfer will be carried out directly by this request and the task will not be saved in the database.

Example of a request that will come to the webhook endpoint:

```
{
 "startedDateTime": "2024-06-07 08:37:09",
 "request": {
  "method": "POST",
  "url": "https://example.com/6e259560-25e5-482b-a7b2-8c5267ba6ae3/",
  "headers": [
   {
    "name": "connection",
    "value": "close"
   },
   {
    "name": "content-length",
    "value": "1421"
   },
   {
    "name": "content-type",
    "value": "multipart/form-data; boundary=\"62ec6524-9360-4e91-834f-e9531e2d2c30\""
   },
   {
    "name": "host",
    "value": "example.com"
   }
  ],
  "bodySize": 0,
  "postData": {
   "mimeType": "application/json",
   "text": ""
  }
 },
 "response": {
  "status": 200,
  "httpVersion": "HTTP/1.1",
  "headers": [
   {
    "name": "Content-Type",
    "value": "text/html"
   }
  ],
  "content": {
   "size": 145,
   "text": "This URL has no default content configured.",
   "mimeType": "text/html"
  }
 }
}
```


Note! The examples don't have an Authorization header because they use a cookie-based authentication model. Read more about authorization in the Authentication and authorization section.

C# .NET

This section offers step-by-step instructions and guides for performing typical tasks using FastReport Cloud with the C#/.NET programming language and FastReport.Cloud.SDK.

Getting Started

You will need the following tools and features.

1. .NET SDK.
2. A C# code editor or a text editor such as Visual Studio Code.
3. Report designer FastReport Community Designer.

   Some guides will require a designer to create the report.
4. Active FastReport Cloud subscription.
5. Access to the Internet.

> Note! These guides assume that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Installation and setup

To connect the SDK, install `FastReport.Cloud.SDK` package of the latest stable version. You can do this with the following command.

```
dotnet add package FastReport.Cloud.SDK
```

To add ASP.NET support, install the `FastReport.Cloud.SDK.Web` package.

```
dotnet add package FastReport.Cloud.SDK.Web
```

You also need to add a string to the `Startup` class.

```
services.AddFastReportCloud();
```

What's next?

We suggest starting with the following articles:

- Status codes.
- Authentication and authorization.

Authentication and authorization

The process of authenticating user data and issuing certain rights to the user in FastReport Cloud is carried out using one of two available methods:

1. Via JWT token.

   In this case, authentication must be completed by a person, and the token will only be valid for 5 minutes, during which the user must log in to their application. When connecting to the server, the browser will redirect you to the authentication server, after which it will generate an access token. For security reasons, we limit the possibility of getting a JWT token only by a person.

   > If the user has not logged into the application within 5 minutes, authentication must be repeated. If the user has logged into the application, re-authentication is not required.

2. Via API key.

   In this case, access permissions are obtained for server applications. To get the API key, the presence of a user is required. However, the key itself can remain valid for a long time, for example, a year.

Getting the first API key

Use the user panel to get the first API key. If for some reason there is no access to the user panel, you can request a key as described below.

1. Open the link in a browser: https://id.fast-report.com/Manage/ExternalLogins

   Clicking on this link will direct you to the browser's automatic authentication process.

2. Once the authentication is passed, you need to request a new key.

   Press `F12` or `Ctrl+Shift+I` to open the developer panel. The keyboard shortcut may differ from the standard ones. In this case, open the developer panel through the browser menu.

3. 3. Copy and execute the code in the JavaScript console.

   This code will make a `POST` request to the URL `https://id.fast-report.com/api/manage/v1/ApiKeys` to generate a new access key before 2030.

4. Refresh the browser page and get the result.

```
{
  "apiKeys": [
    {
      "value": "cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey",
      "description": "Generated by js develop panel",
      "expired": "2030-01-01T07:41:23.399Z"
    }
  ],
  "count": 1
}
```

Now you can use the API key, in the case above it is `cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey`.

You don’t need to get a new API key through the browser again.

How to use the API key

The key should be passed with every request in the `Authorization: Basic` header. The username should be apikey, and the password should be the key value. For example.

```
Authorization: Basic Base64Encode(apikey:cc355oeu1z5d5wncayo33me6c1g5junqdqk4pkupid7t8ynjshey);
```

Where `Base64Encode` is the base64 string encoding function.

For `FastReport.Cloud.SDK`, there is a special class that allows you to add the key to the request header <xref:FastReport.Cloud.FastReportCloudApiKeyHeader>.

To add the required header, create a new `HttpClient`.

```
HttpClient httpClient = new HttpClient();
httpClient.DefaultRequestHeaders.Authorization = new FastReportCloudApiKeyHeader(apiKey);
```

Now this `HttpClient` can be used for all requests.

Getting a new API key

To get a new key, call the method <xref:FastReport.Cloud.IApiKeysClient.CreateApiKeyAsync(FastReport.Cloud.CreateApiKeyVM,System.Threading.CancellationToken)>

```
CreateApiKeyVM model = new CreateApiKeyVM()
{
   Description = "Created by FastReport.Cloud.SDK",
   Expired = DateTime.UtcNow.AddYears(1)
};

IApiKeysClient apiKeysClient = new ApiKeysClient(httpClient);
await apiKeysClient.CreateApiKeyAsync(model);
```

> Whenever possible, use asynchronous method analogs instead of synchronous ones.

The result of executing this function will be the <xref:FastReport.Cloud.ApiKeyVM> model.

What's next?

- Uploading a new template.
- Working with groups.
- Adding new users to a workspace.

Uploading a new template

One of the main features of FastReport Cloud is storing templates, reports, and other data in the cloud. This article will cover how to upload your prepared template into the FastReport Cloud template repository.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report template.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Instruction

1. You need to get the identifier of the workspace root folder. This identifier will never change, so you can save it and not have to request it every time before uploading a new template.

2. To request the root directory, call the method <xref:FastReport.Cloud.ITemplateFoldersClient.GetRootFolderAsync(System.String,System.Threading.CancellationToken)>.

   ```
   public async Task<string> GetTemplatesRoot(HttpClient httpClient, string subscriptionId = null)
   {
      ITemplateFoldersClient templateFoldersClient = new TemplateFoldersClient(httpClient);

      FileVM result = await templateFoldersClient.GetRootFolderAsync(subscriptionId);

      return result.Id;
   }
   ```

   In this example, the `subscriptionId` parameter specifies the identifier of the workspace (subscription). If it is not specified (equals null), the identifier of the user's default workspace will be returned.

3. To upload the required template, use the method <xref:FastReport.Cloud.ITemplatesClient.UploadFileAsync(System.String,FastReport.Cloud.TemplateCreateVM,System.Threading.CancellationToken)>.

```
public async Task<string> UploadFrx(HttpClient httpClient, string folderId, string filePath)
{
   ITemplatesClient templatesClient = new TemplatesClient(httpClient);

   byte[] bytes = File.ReadAllBytes(filePath);

   TemplateCreateVM model = new TemplateCreateVM()
   {
      Name = Path.GetFileName(filePath),
      Content = Convert.ToBase64String(bytes)
   };

   TemplateVM result = await templatesClient.UploadFileAsync(folderId, model);

   return result.Id;
}
```

In this example, the function receives a file from the disk and uploads it to the `folderId` directory. More about the parameters:

- `folderId` — template directory identifier.
- `model.Name` — file name, as it will be displayed inside FastReport Cloud.

  The file name must have the `.frx` extension, if not, then add it manually.
- `model.Content` — file content encoded in base64.

The method returns the identifier of the uploaded template.

Now this template can be used to prepare (build) a report and export.

What's next?

- Permission management on the example of a template.
- Building a report.

Permission management on the example of a template

Restricting access to private resources is a very important part of FastReport Cloud. A flexible access system allows you to set a limit or grant rights to each resource separately, specifying the circle of people who can access them.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report template.

   You can find how to upload a report template in the article Uploading a new template.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Instruction

1. To view the current permissions for a resource, use the method <xref:FastReport.Cloud.ITemplatesClient.GetPermissionsAsync(System.String,System.Threading.CancellationToken)>.

   ```
   public async Task<FilePermission> GetOwnerPermissions(HttpClient httpClient, string templateId)
   {
    ITemplatesClient templatesClient = new TemplatesClient(httpClient);

      FilePermissions permissions = await
         templatesClient.GetPermissionsAsync(templateId);

      return permissions.Owner;
   }
   ```

   In this example, the method returns the permissions for the template owner.

   > Note! The <xref:FastReport.Cloud.FilePermissions> and <xref:FastReport.Cloud.FilePermission> classes differ by the letter `s` at the end, the first contains a full description of the access permissions to the entity, and the second contains only a description for one of the categories.

2. To change permissions, use the method <xref:FastReport.Cloud.ITemplatesClient.UpdatePermissionsAsync(System.String,FastReport.Cloud.Update

FilePermissionsVM,System.Threading.CancellationToken)>.

```
public async Task<FilePermission> ShareTemplate(HttpClient httpClient, string templateId)
{
   ITemplatesClient templatesClient = new TemplatesClient(httpClient);

   UpdateFilePermissionsVM viewModel = new UpdateFilePermissionsVM()
   {
      NewPermissions = new FilePermissions() {
 Anon = new FilePermission() { Get = FilePermissionGet.Entity | FilePermissionGet.Download }
      },
      Administrate = UpdateFilePermissionsVMAdministrate.Anon
   };

   var result = await templatesClient.AddPermissionAsync(templateId, viewModel);

   return result.Other;
}
```

In this example, the function offers anonymous users to view the information about the template and to download it.

Note! In this example, we only pass the permissions that we want to edit and specify them in the `Administrate` property.

If you need to change several permission categories at once, you can list them using the bitwise OR operator (`|`).

What's next?

- Building a report.
- Working with groups.
- Adding new users to a workspace.

Building a report

This article walks you through the process of building a report from a template with the FastReport Cloud report processor.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report template.

   You can find how to upload a report template in the article Uploading a new template.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Instruction

1. You need a template identifier to build it, to get it, use the method <xref:FastReport.Cloud.ITemplatesClient.GetFilesListAsync(System.String,System.Nullable{System.Int32},System.Nullable{System.Int32},System.String,System.Nullable{FastReport.Cloud.FileSorting},System.Nullable{System.Boolean},System.Nullable{System.Boolean},System.Threading.CancellationToken)>

   ```
   public async Task<string> GetTemplateId(HttpClient httpClient)
   {
      ITemplateFoldersClient templateFoldersClient =
         new TemplateFoldersClient(httpClient);
      ITemplatesClient templatesClient = new TemplatesClient(httpClient);

      FileVM rootFolder = await templateFoldersClient.GetRootFolderAsync(null);

      IEnumerable<TemplateVM> templates =
         await templatesClient.GetFilesListAsync(rootFolder.Id, 0, 10);

      TemplateVM template = templates.First();

      return template.Id;
   }
   ```

   In this example, the function requests the user's default workspace root directory, then requests 10 templates, and returns the first one.

2. To build a report, a directory will be needed to put the report to. Request the report root directory by using the method

<xref:FastReport.Cloud.IReportFoldersClient.GetRootFolderAsync(System.String,System.Threading.Cancella
tionToken)>.

```
public async Task<string> GetReportsRoot(HttpClient httpClient,
                                string subscriptionId = null)
{
   IReportFoldersClient reportFoldersClient = new ReportFoldersClient(httpClient);

  FileVM result = await reportFoldersClient.GetRootFolderAsync(subscriptionId);

   return result.Id;
}
```

In this example, the function requests the root directory and the workspace identifier can be omitted, in which case the root directory for the user's default workspace will be returned.

3. To build a report, use the method <xref:FastReport.Cloud.ITemplatesClient.PrepareAsync(System.String,FastReport.Cloud.PrepareTemplateV
M,System.Threading.CancellationToken)>.

```
public async Task<string> BuildReport(HttpClient httpClient,
                              string folderId,
                              string templateId,
                              string fileName)
{
   ITemplatesClient templatesClient = new TemplatesClient(httpClient);

   PrepareTemplateVM task = new PrepareTemplateVM()
   {
      Name = Path.ChangeExtension(fileName, ".fpx"),
      FolderId = folderId
   };

   ReportVM result = await templatesClient.PrepareAsync(templateId, task);

   return result.Id;
}
```

In this example, the function creates a report preparation task.

Note! Although the report has not yet been built, it already has an identifier assigned to it. In a while, the builder queue will reach this task, and the report will be built.

> If FolderId is not specified, the prepared report will be saved in the root folder.

4. To get information about the report, use the method <xref:FastReport.Cloud.IReportsClient.GetFileAsync(System.String,System.Threading.CancellationToken)>.

```
public async Task<ReportVMStatus> CheckStatus(HttpClient httpClient, string reportId)
{
   IReportsClient reportsClient = new ReportsClient(httpClient);

   ReportVM result = await reportsClient.GetFileAsync(reportId);

   return result.Status.GetValueOrDefault();
}
```

In this example, the function requests the report by its identifier and returns the status. You need to wait for the <xref:FastReport.Cloud.FileStatus.Success> status, check the status every few seconds.

5. We check the status in a loop and download the file.

```
    int tries = 10;
    FileStatus status;
    do
    {
        status = await CheckStatus(httpClient, reportId);
        tries--;
    } while (status != FileStatus.Success && tries > 0);

    var report = await DownloadReport(httpClient, reportId);
```

6. To download the report, use the method <xref:FastReport.Cloud.IDownloadClient.GetReportAsync(System.String,System.Threading.CancellationToken)>.

```
public async Task<byte[]> DownloadReport(HttpClient httpClient, string reportId)
{
   IDownloadClient downloadClient = new DownloadClient(httpClient);

 FileResponse file = await downloadClient.GetReportAsync(reportId);

   using(MemoryStream ms = new MemoryStream())
   {
      file.Stream.CopyTo(ms);

      return ms.ToArray();
   }
}
```

   In this example, the function requests the file and copies it into memory.

What's next?

- Exporting a report to PDF.

Report parameters

In this article, we will look at the process of passing parameters to a report, which represent a dictionary of parameters provided when generating the report. More information on this topic can be found in the FastReport .NET manual in the section Report parameters.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.
3. A C# code editor or a text editor such as Visual Studio Code.
4. Report template with the specified report parameters.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.
6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Passing a value to a report parameter

Parameters can be passed in the following cases:

- Prepare template;
- Export template;
- Working with tasks.

Let's look at the passing of parameters using an example of preparing a template. Detailed instructions on how to prepare a report can be found in the section Building a report.

```
public async Task PassingReportParameters(HttpClient httpClient,
                        string folderId,
                        string templateId,
                        string fileName)
{
   ITemplatesClient templatesClient = new TemplatesClient(httpClient);
   PrepareTemplateVM task = new PrepareTemplateVM()
   {
    Name = Path.ChangeExtension(fileName, ".fpx"),
    FolderId = folderId,
    ReportParameters = new Dictionary<string, string>
    {
       { "Parameter1", "Value1" },
       { "Parameter2", "Value2" }
    }
};

// Add a new parameter to the dictionary
task.ReportParameters.Add("Parameter3", "Value3");

ReportVM result = await templatesClient.PrepareAsync(templateId, task);}
```

Exporting a report to PDF

This article walks you through the process of exporting a report with the FastReport Cloud report processor.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report.

   You can find how to build a report in the article Building a report.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Comment

Please note that the report can be exported directly from the template, without intermediate saving of the report. To do this, run the same commands for the report template, replacing `Report` in the methods with `Template`, also use the template identifier, not that of the report.

Instruction

1. You will need a report identifier to export to PDF, to get it use the method <xref:FastReport.Cloud.ITemplatesClient.GetFilesListAsync(System.String,System.Nullable{System.Int32},System.Nullable{System.Int32},System.String,System.Nullable{FastReport.Cloud.FileSorting},System.Nullable{System.Boolean},System.Nullable{System.Boolean},System.Threading.CancellationToken)>

   ```
   public async Task<string> GetReportId(HttpClient httpClient)
   {
      IReportFoldersClient reportFoldersClient = new ReportFoldersClient(httpClient);
      IReportsClient reportsClient = new ReportsClient(httpClient);

      FileVM rootFolder = await reportFoldersClient.GetRootFolderAsync(null);

      IEnumerable<ReportVM> reports =
         await reportsClient.GetFilesListAsync(rootFolder.Id, 0, 10);

      ReportVM report = reports.First();

      return report.Id;
   }
   ```

   In this example, the function requests the user's default workspace root directory, then requests 10

reports, and returns the first one.

2. To export the report, a directory will be needed to save the export to.

   Get the exports root directory by using the method <xref:FastReport.Cloud.IExportFoldersClient.GetRootFolderAsync(System.String,System.Threading.CancellationToken)>.

```
public async Task<string> GetExportsRoot(HttpClient httpClient,
                                string subscriptionId = null)
{
   IExportFoldersClient exportFoldersClient = new ExportFoldersClient(httpClient);

   FileVM result = await exportFoldersClient.GetRootFolderAsync(subscriptionId);

   return result.Id;
}
```

   In this example, the function requests the root directory, and the workspace identifier can be omitted, in which case the root directory for the user's default workspace will be returned.

3. To export a report, use the method <xref:FastReport.Cloud.IReportsClient.ExportAsync(System.String,FastReport.Cloud.ExportReportVM,System.Threading.CancellationToken)>.

```
public async Task<string> ExportReport(HttpClient httpClient,
                               string folderId,
                               string reportId,
                               string fileName)
{
   IReportsClient reportsClient = new ReportsClient(httpClient);

   ExportReportVM task = new ExportReportVM()
   {
      FileName = Path.ChangeExtension(fileName, ".pdf"),
      FolderId = folderId,
      Format = ExportReportTaskVMFormat.Pdf,
      ExportParameters = new Dictionary<string, string>
      {
         { "additionalProp1", ""},
         { "additionalProp2", ""},
         { "additionalProp3", ""}
      }
   };

   ExportVM result = await reportsClient.ExportAsync(reportId, task);

   return result.Id;
}
```

   - `FileName` — the name of the resulting file. If you do not specify the extension or specify it incorrectly, the server will replace it automatically.
   - `FolderId` — identifier of the directory where the export will be placed. If left blank, the export will be placed in the exports root folder in the workspace.
   - `Format` — export format.
   - `ExportParameters` — export parameters. They are set similarly to the export parameters from the FastReport .NET library. A more detailed description is available in the User Manual in the section export parameters.

   In this example, the function creates a report export task.

Note! Although the report has not yet been exported, the export already has an assigned identifier at this stage. In a while, the builder queue will reach this task, and the report will be exported.

If FolderId is not specified, then the export will be saved to the root folder.

4. To get information about the file, use the method <xref:FastReport.Cloud.IExportsClient.GetFileAsync(System.String,System.Threading.CancellationToken)>.

```
public async Task<ExportVMStatus> CheckStatus(HttpClient httpClient, string exportId)
{
   IExportsClient exportsClient = new ExportsClient(httpClient);

   ExportVM result = await exportsClient.GetFileAsync(exportId);

   return result.Status.GetValueOrDefault();
}
```

   In this example, the function requests an export by its identifier and returns the status. You need to wait for the <xref:FastReport.Cloud.FileStatus.Success> status, check the status every few seconds.

5. We check the status in a loop and download the export.

```
int tries = 10;
FileStatus status;
do
{
   status = await CheckStatus(httpClient, reportId);
   tries--;
} while (status != FileStatus.Success && tries > 0);
var report = await DownloadExport(httpClient, reportId);
```

6. To download a report, use the method <xref:FastReport.Cloud.IDownloadClient.GetExportAsync(System.String,System.Nullable{System.Boolean} System.Threading.CancellationToken)>.

```
public async Task<byte[]> DownloadExport(HttpClient httpClient, string exportId)
{
   IDownloadClient downloadClient = new DownloadClient(httpClient);

   FileResponse file = await downloadClient.GetExportAsync(exportId);

   using (MemoryStream ms = new MemoryStream())
   {
      file.Stream.CopyTo(ms);

      return ms.ToArray();
   }
}
```

   In this example, the function requests a file and copies it into memory.

What's next?

- Working with groups.
- Adding new users to a workspace.

Adding new users to a workspace

This article walks you through the process of adding a new user to a subscription, getting a list of users in a subscription, and removing a user from a subscription.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Active FastReport Cloud subscription, which has two user slots.

5. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Comment

Note! It is only possible to add a user to a workspace by user identifier.

Instruction

1. To add a new user to a workspace, the identifier of the workspace (subscription) is required.

   Get the workspace identifier using the method <xref:FastReport.Cloud.ISubscriptionsClient.GetSubscriptionsAsync(System.Nullable{System.Int32},System.Nullable{System.Int32},System.Threading.CancellationToken)>.

   ```
   public async Task<string> GetSubscriptionId(HttpClient httpClient)
   {
      ISubscriptionsClient subscriptionsClient = new SubscriptionsClient(httpClient);
    SubscriptionsVM subscriptions =
          await subscriptionsClient.GetSubscriptionsAsync(0, 10);
    SubscriptionVM subscription = subscriptions.Subscriptions.First();
      return subscription.Id;
   }
   ```

   In this example, the function requests the first 10 workspaces (subscriptions) from the user's list of workspaces, selects the first workspace, and returns its identifier.

   > A workspace always matches one subscription, so they have the same identifiers.

2. To add a new user, use the method <xref:FastReport.Cloud.ISubscriptionUsersClient.AddUserAsync(System.String,System.String,System.Threading.CancellationToken)>.

```
public async Task AddUser(HttpClient httpClient, string subscriptionId, string userId)
{
    ISubscriptionUsersClient subscriptionUsersClient =
        new SubscriptionUsersClient(httpClient);
    await subscriptionUsersClient.AddUserAsync(subscriptionId, userId);
}
```

In this example, the function adds a user with userId identifier to the workspace with `subscriptionId` identifier.

3. To get a list of workspace users, use the method <xref:FastReport.Cloud.ISubscriptionUsersClient.GetUsersAsync(System.String,System.Nullable{System.Int32},System.Nullable{System.Int32},System.Threading.CancellationToken)>.

   ```
   public async Task<IEnumerable<string>> GetUsers(HttpClient httpClient, string subscriptionId)
   {
       ISubscriptionUsersClient subscriptionUsersClient =
           new SubscriptionUsersClient(httpClient);
    SubscriptionUsersVM users =
           await subscriptionUsersClient.GetUsersAsync(subscriptionId, 0, 10);
    return users.Users.Select(m => m.UserId);
   }
   ```

   In this example, the function requests the first 10 users from the workspace with the `subscriptionId` identifier.

4. To remove a user from the workspace, use the method <xref:FastReport.Cloud.ISubscriptionUsersClient.RemoveUserAsync(System.String,System.String,System.Threading.CancellationToken)>.

   ```
   public async Task RemoveUser(HttpClient httpClient, string subscriptionId, string userId)
   {
       ISubscriptionUsersClient subscriptionUsersClient =
           new SubscriptionUsersClient(httpClient);
    await subscriptionUsersClient.RemoveUserAsync(subscriptionId, userId);
   }
   ```

   In this method, the function removes the user with `userId` identifier from the workspace with subscriptionId `identifier`.

What's next?

- Working with groups
- Help and feedback

Working with groups

This article walks you through the process of creating a new group, adding a user to the group, and getting a list of the group's users.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Active FastReport Cloud subscription, which has two user slots.

5. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

Comment

Note! It is only possible to add a user to a group if the user exists in the workspace.

Note! It is only possible to add a user to a group by its identifier.

Instruction

1. To create a new group, you need a workspace identifier and a name for the new group.

   To get the workspace identifier, use the method <xref:FastReport.Cloud.ISubscriptionsClient.GetSubscriptionsAsync(System.Nullable{System.Int32},System.Nullable{System.Int32},System.Threading.CancellationToken)>.

   ```
   public async Task<string> GetSubscriptionId(HttpClient httpClient)
   {
      ISubscriptionsClient subscriptionsClient = new SubscriptionsClient(httpClient);

      SubscriptionsVM subscriptions =
       await subscriptionsClient.GetSubscriptionsAsync(0, 10);

      SubscriptionVM subscription = subscriptions.Subscriptions.First();

      return subscription.Id;
   }
   ```

   In this example, the function requests the first 10 workspaces from the user's list of workspaces, selects the first workspace, and returns its identifier.

   > The workspace and the subscription identifiers are the same because one subscription is associated with each workspace.

2. To create a new group, use the method <xref:FastReport.Cloud.IGroupsClient.CreateGroupAsync(FastReport.Cloud.CreateGroupVM,System.Threading.CancellationToken)>.

```
public async Task<string> CreateGroup(HttpClient httpClient,
                                      string subscriptionId,
                                      string groupName)
{
   IGroupsClient groupsClient = new GroupsClient(httpClient);

   CreateGroupVM viewModel = new CreateGroupVM()
   {
      Name = groupName,
      SubscriptionId = subscriptionId
   };

   GroupVM group = await groupsClient.CreateGroupAsync(viewModel);

   return group.Id;
}
```

   In this example, the function creates a new group named `groupName` for the workspace with the subscriptionId identifier, as a `result`, the function will return the identifier of the created group.

3. To add a new user to the group, use the method <xref:FastReport.Cloud.IGroupUsersClient.AddUserToGroupAsync(System.String,System.String,System.Threading.CancellationToken)>.

```
public async Task AddUser(HttpClient httpClient, string groupId, string userId)
{
   IGroupUsersClient groupUsersClient = new GroupUsersClient(httpClient);

   await groupUsersClient.AddUserToGroupAsync(groupId, userId);
}
```

   In this example, the function adds the user with the `userId` identifier to the group with the `groupId` identifier.

4. To get a list of users in the group, use the method <xref:FastReport.Cloud.IGroupUsersClient.GetUsersInGroupAsync(System.String,System.Nullable{System.Int32},System.Nullable{System.Int32},System.Threading.CancellationToken)>.

```
public async Task<IEnumerable<string>> GetUsers(HttpClient httpClient, string groupId)
{
   IGroupUsersClient groupUsersClient = new GroupUsersClient(httpClient);

   GroupUsersVM users =
    await groupUsersClient.GetUsersInGroupAsync(groupId, 0, 10);

 return users.Users.Select(m => m.UserId);
}
```

   In this example, the function requests the first 10 users from the group with the `groupId` identifier.

What's next?

- Help and feedback

Working with tasks

Tasks in FastReport Cloud are actions for converting and delivering documents to consumers. They are described in detail in the Tasks section.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report template.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

General information

In this article we will look at how to work with tasks.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.

3. A C# code editor or a text editor such as Visual Studio Code.

4. Report template.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.

6. Access to the Internet.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

First, we need to create an HttpClient, which we will use next:

```
var httpClient = new HttpClient();
httpClient.BaseAddress = new Uri("https://fastreport.cloud/");
httpClient.DefaultRequestHeaders.Authorization = new FastReportCloudApiKeyHeader(ApiKey);
```

Next, we get a subscription to work with tasks:

```
var subscriptions = new SubscriptionsClient(httpClient);
var subscription = (await subscriptions.GetSubscriptionsAsync(0, 10)).Subscriptions.FirstOrDefault();
```

Now we need to create a client to work with tasks:

```
TasksClient tasksClient = new TasksClient(httpClient);
```

Now we can start working directly with tasks.

Creating a task

To create tasks, use the CreateTask or CreateTaskAsync method. It can take any TaskBaseVM derived class:

- CreatePrepareTaskVM
- CreateExportTemplateTaskVM
- CreateExportReportTaskVM
- CreateFetchTaskVM
- CreateEmailTaskVM
- CreateWebhookTaskVM
- CreateFTPUploadTaskVM.

Let’s look at how to create a task of preparing a report, saving it to a folder and then exporting it to PDF with saving to a folder:

```
await tasksClient.CreateTaskAsync(new CreatePrepareTaskVM
        {
           Name = "My first task",
           Type = TaskType.Prepare,
           InputFile = new InputFileVM
           {
              EntityId = "{templateId}"
           },
           OutputFile = new OutputFileVM
           {
              FileName = "My first report generated by .fpx",
              FolderId = "{report folder identifier}"
           },
           Exports = new List<CreateExportReportTaskVM>
           {
              new CreateExportReportTaskVM
              {
                 Format = ExportFormat.Pdf,
                 OutputFile = new OutputFileVM
                 {
                    FileName = "pdfИзFpxИзFrx.pdf",
                    FolderId = "{export folder identifier}"
                 }
              }
           }
        });
```

The EntityId and FolderId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

> Note! If you do not specify OutputFile, it will be saved in the temporary folder. If you specify an empty OutputFile, it will be saved to the root folder. If you specify the folder ID, it will be saved to that folder.

Getting a list of tasks

```
// Get first 100 tasks from workspace
var tasks = await tasksClient.GetListAsync(0, 100, subscription.Id);
```

Executing a task by the specified identifier

```
// Start task by identifier
await tasksClient.RunTaskByIdAsync(id);
```

Deleting tasks from storage

```
// Delete all tasks
foreach(var t in tasks.Tasks)
{
   await tasksClient.DeleteTaskAsync(t.Id);
}
```

Upload to FTP server

In this article, we will consider a way to send a report to an FTP server. Instructions on how to work with tasks are described in the General information.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.
3. A C# code editor or a text editor such as Visual Studio Code.
4. Report template.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.
6. Access to the Internet.
7. Configured and accessible FTP server.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

> Note! This guide assumes that you have experience in setting up and configuring an FTP server to receive files from external sources.

Creating a task

Let’s look at how to create a FTP Upload task:

```
// Object initialization
 CreateFTPUploadTaskVM ftpUploadTaskVM = new CreateFTPUploadTaskVM
 {
    Name = "FTP upload task",
    InputFile = new InputFileVM
    {
       EntityId = "{templateId}",
       Type = FileKind.Template
    },
    FtpHost = "{FTP server address}",
    FtpPort = 21,
    FtpUsername = "{FTP server username}",
    FtpPassword = "{password}",
    UseSFTP = false,
    DestinationFolder = "{/path_to_folder/}",
    Archive = false,
    ArchiveName = "Archive name",
    SubscriptionId = "{subscriptionId}"
 };

 // Task creation
 TaskBaseVM ftpUploadTask = await tasksClient.CreateTaskAsync(ftpUploadTaskVM);

 // Run task by identifier
 await tasksClient.RunTaskByIdAsync(ftpUploadTask.Id);
```

The EntityId and SubscriptionId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

Executing a task from request body

```
// Run task from request body
 await tasksClient.RunTaskAsync(new RunFTPUploadTaskVM
 {
    InputFile = new RunInputFileVM
    {
       EntityId = "{templateId}",
       Type = FileKind.Template
    },
    FtpHost = "{FTP server address}",
    FtpPort = 21,
    FtpUsername = "{FTP server username}",
    FtpPassword = "{password}",
    UseSFTP = false,
    DestinationFolder = "{/path_to_folder/}",
    Archive = false,
    ArchiveName = "Archive name",
    SubscriptionId = "{subscriptionId}"
 });
```

> Note! In this case, the FTP transfer will be carried out directly by this request and the task will not be saved in the database.

Updating a task by the specified identifier

```
await tasksClient.UpdateTaskAsync("{oldTaskId}", new UpdateFTPUploadTaskVM()
{
    InputFile = new RunInputFileVM
    {
        EntityId = "{Updated templateId}",
        Type = FileKind.Template
    },
    FtpHost = "{Updated FTP server address}",
    FtpPort = 21,
    FtpUsername = "{Updated FTP server username}",
    FtpPassword = "{password}",
    UseSFTP = false,
    DestinationFolder = "/updated_path_to_folder/",
    Archive = false,
    ArchiveName = "Updated archive name"
});
```

Upload via Webhook

In this article, we will consider a way to send a report via webhook. Instructions on how to work with tasks are described in the General information.

Getting Started

You will need the following tools and features:

1. Knowledge of using API key in FastReport Cloud.

   This article will skip additional information on authentication and authorization.

2. .NET SDK.
3. A C# code editor or a text editor such as Visual Studio Code.
4. Report template.

   It can be created using the free program FastReport Community Designer.

5. Active FastReport Cloud subscription.
6. Access to the Internet.
7. Client application that accepts requests.

> Note! This guide assumes that you already know how to develop your application in the C# programming language.

> Note! The paragraphs above describe the recommended tools.

> Note! This guide assumes you have a client application that accepts and processes incoming webhooks.

Creating a task

Let’s look at how to create a Webhook task:

```
// Object initialization
CreateWebhookTaskVM webhookTaskVM = new CreateWebhookTaskVM
{
   Name = "Webhook task",
   InputFile = new InputFileVM
   {
      EntityId = "{templateId}",
      Type = FileKind.Template
   },
   Url = new Uri("https://example.com/"),
   Headers = new Dictionary<string, string> {
      { "Authorization", "Bearer <token>" },
      { "Content-Length", "1238" }
   },
   SubscriptionId = "{subscriptionId}"
};

// Task creation
 TaskBaseVM webhookTask = await tasksClient.CreateTaskAsync(webhookTaskVM);

// Run task by identifier
 await tasksClient.RunTaskByIdAsync(webhookTask.Id);
```

> Note! The EntityId and SubscriptionId fields should be filled with real object identifiers. Otherwise, the task will be interrupted with an error.

> Note! The identifier of the template, report, and export can be passed to the EntityId.

Run a task from request body

```
// Run task from request body
 await tasksClient.RunTaskAsync(new RunWebhookTaskVM
 {
    InputFile = new RunInputFileVM
    {
       EntityId = "{templateId}",
       Type = FileKind.Template
    },
    Url = new Uri("https://example.com/"),
    Headers = new Dictionary<string, string> {
       { "Authorization", "Bearer <token>" },
       { "Content-Type", "multipart/form-data" }
    },
    SubscriptionId = "{subscriptionId}"
 });
```

> Note! In this case, the Webhook transfer will be carried out directly by this request and the task will not be saved in the database.

Updating a task by the specified identifier

```
await tasksClient.UpdateTaskAsync("{oldTaskId}", new UpdateWebhookTaskVM()
 {
    InputFile = new RunInputFileVM
    {
       EntityId = "{Updated templateId}",
       Type = FileKind.Template
    },
    Url = new Uri("{Updated Url}"),
    Headers = new Dictionary<string, string> {
       { "{Updated header}", "{Updated value}" },
    }
 });
```

Example of a request that will come to the webhook endpoint:


```
{
  "startedDateTime": "2024-06-07 08:37:09",
  "request": {
    "method": "POST",
    "url": "https://example.com/6e259560-25e5-482b-a7b2-8c5267ba6ae3/",
    "headers": [
      {
        "name": "connection",
        "value": "close"
      },
      {
        "name": "content-length",
        "value": "1421"
      },
      {
        "name": "content-type",
        "value": "multipart/form-data; boundary=\"62ec6524-9360-4e91-834f-e9531e2d2c30\""
      },
      {
        "name": "host",
        "value": "example.com"
      }
    ],
    "bodySize": 0,
    "postData": {
      "mimeType": "application/json",
      "text": ""
    }
  },
  "response": {
    "status": 200,
    "httpVersion": "HTTP/1.1",
    "headers": [
      {
        "name": "Content-Type",
        "value": "text/html"
      }
    ],
    "content": {
      "size": 145,
      "text": "This URL has no default content configured.",
      "mimeType": "text/html"
    }
  }
}
```