Leerlingen WEB API

Implementation guide

Webservice version	1
Document version	V 1.19
Release date	January 20, 2025
Last modified	January 14, 2025
Last modified by	Pierre-Marie Mahieu

Contents

Document description

This document describes the “Leerlingen” Web API and provides details required for a technical implementation. It defines the communication model to use when talking to the Web API, including the required security algorithms. It describes all of the available methods in detail and is illustrated with examples.

Ownership

This document is written and maintained by Informat.

API Basics

The provided webservice is implemented as a RESTful service, based on the principles defined by Roy Fielding in his doctoral dissertation at UC Irvine in 2000.

Representational state transfer (REST) is an abstraction of the architecture of the World Wide Web; more precisely, REST is an architectural style consisting of a coordinated set of architectural constraints applied to components, connectors, and data elements, within a distributed hypermedia system. REST ignores the details of component implementation and protocol syntax in order to focus on the roles of components, the constraints upon their interaction with other components, and their interpretation of significant data elements.

This chapter describes the basic information you need to interact with the services provided by Informat within the Leerlingen context.

Endpoint

The **production environment** is available via:

`endpoint = leerlingenapi.informatsoftware.be`

Versioning

The Leerlingen WEB API uses url embedded versioning to maintain backwards compatibility during its lifecycle. This means that we inject the version number in the request URL like this:

```
https://<endpoint>/<version>/<resource>
```

Example: `https://leerlingenapi.informatsoftware.be/1/{name}`

The version number will increase if a release introduces a breaking change or a significant change in the WEB API implementation. The version number is noted on the first page of this documentation and in the footer of each page.

The current API version number is **1**

Using this allows users to upgrade their implementations at their own pace. If a release creates a change that will break the workings of previous versions, e.g. a change in the underlying data structure, or if a significant change in the internal logic is added, this will be announced at least 1 month in advance, to allow time for client implementations to upgrade. These changes will be kept to a minimum, and where possible bundled, to guarantee consistent availability.

Character sets

All service response objects will be formatted with UTF-8 encoding to ensure compatibility with most languages.

Request Format and Responses

Request verbs

All request data should be specified in the JSON format.

As per the definition of RESTful services, you can use HTTP verbs to issue different commands to the service.

Normally, you would use `GET` requests to read data from the service, but in order to allow future expansion of methods, all `GET` requests are required to be sent with the `POST` verb.

If you use an unsupported HTTP request type with a URL that does not support the specified verb, a `405` HTTP error will be returned, listing the supported HTTP methods for that URL.

Example:

```
HTTP/1.1 405 Method Not Allowed
Content-Length: 94
Content-Type: application/json
Date: Fri, 18 Oct 2019 10:48:36 GMT
{
}
```

JSON Basics

The majority of requests and responses to the WEB API use the JavaScript Object Notation (JSON) for formatting the content and structure of the data and responses.

JSON is used because it is the simplest and easiest solution for working with data within a web browser, as JSON structures can be evaluated and used as JavaScript objects within the web browser environment.

JSON supports the same basic types as supported by JavaScript, these are:

- **Number** (either integer or floating-point).
- **String**; this should be enclosed by double-quotes and supports Unicode characters and backslash escaping.

  For example: `"A String"`
- **Boolean** - a true or false value. You can use these strings directly. For example: `{ "value": true }`
- **Array** - a list of values enclosed in square brackets. For example: `["one", "two", "three"]`
- **Object** - a set of key/value pairs (i.e. an associative array, or hash). The key must be a string, but the value can be any of the supported JSON values. For example:

```
{
    "servings" : 4,
    "subtitle" : "Easy to make in advance, and then cook when ready",
    "cooktime" : 60,
    "title" : "Chicken Coriander"
}
```

Handling dates (and time)

Dates should always be sent to the server in either UTC format or using the yyyy-MM-dd (hh:mm:ss) notation.

Example:

```
2019-09-30T15:30:22.000Z            // Valid UTC Format
2019-09-30T15:30:22+0100            // Valid UTC Format
2019-09-30
2019-09-30 15:30:22
```

If you're using javascript Date() objects in your request body, your browser will automatically convert these to the UTZ format.

Example:

```
// javascript
var someDate = new Date(2019,05,21)    // will be sent as 2019-05-21T00:00:00.000Z
```

If you need to send a date and time object, use the UTC format or use yyyy-MM-dd hh:mm:ss

Example:

```
2019-09-30T22:30:00.000Z            // UTC Format
2019-09-30 22:30:00
```

Boolean

Booleans should always be formatted as true/false (lower case, no quotation marks).

Example:

```
{
   "isAdmin": true,
   "canAccess": false
}
```

Response format

The API can return data in both JSON and XML format. To change the return type of the data, add the "`Accept`" header to your request with the correct value corresponding to the desired response.

`application/json` for a JSON formatted response

`application/xml` for an XML formatted response

The response object will always contain the following fields:

Field	Type	Availability	Description
resultCode	integer	always	Contains a code that describes the status of the request. This code is an addition to the HTTP status code. A list of possible result codes can be found at the end of this document.
resultMessage	string	always	This field contains a short user friendly description of the returned result code.
resultDetails	object	Optional	This field sometimes contains extra details about the request, e.g. specific details regarding a server side error, or a problem with the request body. The content of this field is method/action specific and more details can be found in the resources chapter and in the list of result codes.

HTTP Headers

Because the WEB API uses HTTP for all communication, you need to ensure that the correct HTTP headers are supplied (and processed on retrieval) so that you get the right format and encoding. Different environments and

clients will be more or less strict on the effect of these HTTP headers (especially when not present). Where possible you should be as specific as possible.

Request Headers

`Content-type`

Specifies the content type of the information being supplied within the request. The specification uses MIME type specifications. For the majority of requests this will be JSON (`application/json`). For some settings the MIME type will be plain text. When uploading attachments it should be the corresponding MIME type for the attachment or binary (`application/octet-stream`).

The use of the correct `Content-type` header on a request is required, unless the body is empty.

`Accept`

Specifies the list of accepted data types to be returned by the server (i.e. that are accepted/understandable by the client). The format should be a list of one or more MIME types, separated by colons.

For the majority of requests the definition should be for JSON data (`application/json`). For attachments you can either specify the MIME type explicitly, or use `*/*` to specify that all file types are supported. If the `Accept` header is not supplied, then the `*/*` MIME type is assumed (i.e. client accepts all formats).

The use of `Accept` in queries for Leerlingen is not required, but is highly recommended as it helps to ensure that the data returned can be processed by the client.

If you specify a data type using the `Accept` header, Leerlingen will honor the specified type in the `Content-type` header field returned. For example, if you explicitly request `application/json` in the `Accept` of a request, the returned HTTP headers will use the value in the returned `Content-type` field.

For example, when sending a request without an explicit `Accept` header, or when specifying `*/*`:

```
POST  /1/department HTTP/1.1
Host: leerlingenapi.informatsoftware.be:443
Accept: */*
```

The returned headers are:

```
Server: Microsoft IIS/8.8
Date: Fri, 19 Oct 2019 14:28:53 GMT
Content-Type: text/plain;charset=utf-8
Content-Length: 227
Cache-Control: no-cache
```

Note that the returned content type is `text/plain` even though the information returned by the request is in JSON format.

Explicitly specifying the `Accept` header:

```
POST  /1/department HTTP/1.1
Host: leerlingenapi.informatsoftware.be:443
Accept: application/json
```

The headers returned include the `application/json` content type:

```
Date: Fri, 19 Oct 2019 14:28:53 GMT
Content-Type: application/json
Content-Length: 227
Cache-Control: no-cache
```

Response Headers

Response headers are returned by the server when sending back content and include a number of different header fields, many of which are standard HTTP response header and have no significance to Leerlingen operation. The list of response headers important to Leerlingen are listed below.

HTTP Status Codes

With the interface to Leerlingen working through HTTP, error codes and statuses are reported using a combination of the HTTP status code number, and corresponding data in the body of the response data.

A list of the error codes returned by the WEB API, and generic descriptions of the related errors are provided below. The meaning of status codes for specific request types is provided in the corresponding API call reference.

Code	Text	Description
200	OK	Request completed successfully.
400	Bad Request	Bad request structure. The error can indicate an error with the request URL, path or headers. Differences in the supplied MD5 hash and content also trigger this error, as this may indicate message corruption.
401	Unauthorized	The item requested was not available using the supplied authorization, or authorization was not supplied.
403	Forbidden	The requested item or operation is forbidden.
404	Not Found	The requested content could not be found.
405	Resource Not Allowed	A request was made using an invalid HTTP request type for the URL requested. For example, you have requested a PUT when a POST is required. Errors of this type can also triggered by invalid URL strings.
406	Not Acceptable	The requested content type is not supported by the server.
409	Conflict	Request resulted in an update conflict.
415	Bad Content Type	The content types supported, and the content type of the information being requested or submitted indicate that the content type is not supported.
416	Requested Range Not Satisfiable	The range specified in the request header cannot be satisfied by the server.
500	Internal Server Error	The request was invalid, either because the supplied JSON was invalid, or invalid information was supplied as part of the request.

Usage restrictions

To be able to balance the load on our servers and the impact on our overall performance, this service requires client implementations to comply with some basic rules. In case a specific resource or method requires additional rules, these will be added to the corresponding chapters.

Ground rules

- Cache retrieved data as much as possible, use the provided ID's and references to maintain data consistency.
- The privacy and security of the private key is the responsibility of the end user. In case the private key is compromised, the end user is required to inform Informat as soon as possible. We will then take the necessary steps to ensure the safety of the data. A new private key will then be issued.

Privacy concerns

The user is responsible for the safe and correct processing of all personal data, conform privacy regulations, as is stipulated in the contract between the user and the supplier, Informat.

Disclaimer

All requests to the service are logged and monitored to aid in the improvement of the service, to help troubleshoot issues and to avoid abuse.

If Informat finds that the user is found guilty of abuse, Informat reserves the right to terminate the users' access to the service if the abuse continues after multiple warnings.

Security

Transport security

Secure Socket Layer

To ensure the safety of the transferred data and to prevent data-theft, this service operates only via HTTPS.

Transport Layer Security (TLS) and its predecessor, Secure Sockets Layer (SSL), are cryptographic protocols designed to provide communication security over the Internet. They use X.509 certificates and hence asymmetric cryptography to authenticate the counterparty with whom they are communicating, and to exchange a symmetric key. This session key is then used to encrypt data flowing between the parties. This allows for data/message confidentiality, and message authentication codes for message integrity and as a by-product, message authentication.

Authentication endpoint

The **production environment** is available via:

```
identityEndpoint = https://www.identityserver.be
```

Request authentication

Modern secure applications often use access tokens to ensure a user has access to the appropriate resources, and these access tokens typically have a limited lifetime. This is done for various security reasons: for one, limiting the lifetime of the access token limits the amount of time an attacker can use a stolen token. Also, the information contained in or referenced by the access token could become stale.

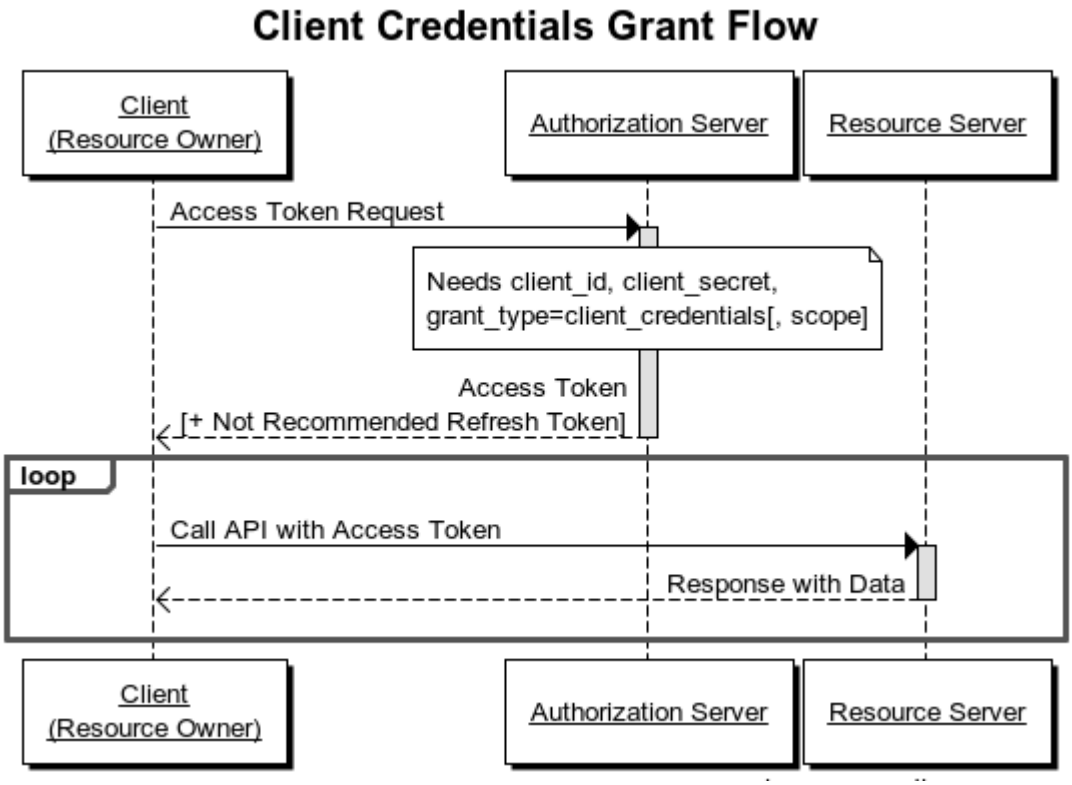
Calling resources without Access Token

When trying to access the API without Access Token, an Unauthorized response will be returned.

```
HTTP/1.1 401 Unauthorized
WWW-Authenticate: Bearer
```

Getting an Access Token by client credentials

The Client Credentials grant type is used by clients to obtain an access token by using a clientId and clientSecret.



Getting access for 1 or multiple scopes

A scope is a permission that is set on a token, a context in which that token may act.

For example, a token with the **api_informat_sas_leerlingen.voorinschrijvingen.123456** scope is permitted to read or write data to the leerlingenapi for the specified institute number, 123456 in this example. Otherwise you would be denied access.

You can request access for 1 scope but it's also possible to request a token for multiple scopes. See the following examples.

Note: The scope has two flavors:

- The pre-registrations calls use the pattern **api_informat_sas_leerlingen.voorinschrijvingen.{instituteNo}**;
- The pupil related calls use the pattern **api_informat_sas_leerlingen.leerlingen.{instituteNo}**.

The pattern that applies, is listed in each main section (green colored).

Request token for **pre-registrations** calls - 1 scope

Request

```
POST https://<identityEndpoint>/connect/token     HTTP/1.1
...
Content-Type: application/x-www-form-urlencoded

grant_type: client_credentials
client_id: <received client id>
client_secret: <received client secret>
scope: api_informat_sas_leerlingen.voorinschrijvingen.123456
```

Response

```
HTTP/1.1 200 OK
...

{
"access_token":"eyJhbGciOiJSUzI1NiIsImtpZCI6IjYyOEM3NzFEM0FCNEI0Rjk4MzRFODkyMTcyMTc2OTk5NjkyNTZFOEQiLCJ
0eXAiOiJKV1QiLCJ4NXQiOiJZb3gzSFRxMHRQbURUb2toY2hkcG1Xa2xibzAifQ.eyJuYmYiOjE1OTM2NzM1MDcsImV4cCI6MTU5MzY
3NzEwNywiaXNzIjoiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUiLCJhdWQiOlsiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZX
IuYmUvcmVzb3VyY2VzIiwiYXBpX2luZm9ybWF0X3Nhc19sZWVybGluZ2VuIl0sImNsaWVudF9pZCI6ImluZm9ybWF0X2N1c3RvbWVyX
2hoc2Nob2xlbiIsInNjb3BlIjpbImFwaV9pbmZvcm1hdF9zYXNfbGVlcmxpbmdlbi52b29yaW5zY2hyaWp2aW5nZW4uMDMyODQ3Il19
.vfTPZYPGGsbbl4Vy1S-FJdvPQTDlcT5m4E2mWFPqPMRBIQINqQBiNfxpH8DUu5cMRvcTFJxCeAsm33Rf-
my0jR1qVBYIlrKgmbIth_gyCNAycHUBAUsd-nYIJIv3Jz20_zM1L3gklUL48fCYc-
hwT1rbS4uRIpHQxDd_sdsNyvUjB9DgMCdxuoqBm0cVzmVvHlH6Fpez-
ttQ5Aymsj0dx0JvAIItVjxVHPqRwA33Sl1iSVogOjfjHP437d4ztFOjzAGUHglwO4eF10ALe_d8D1w3CqjAJGOaqP_W9ttPTATuPvwK
Q2xYaGiL1kPsejTpoRCd176vjv9i3b1dxPiw6FcVffeMUguetub8l8mHVEd0eYEDKUoTmZtp_bAhu5sUq-
eOC3GcJtGwC9TfjTQLtTdgniEfqhX81TZI3Kh2th4z5SOROKt5Sq5RtnN9U9WDrci2DQrfkt5pNg7leMIMqF8AsvuYp5rXGR-
HYWN2YB1uf2YTwIa-wOCshixjdzV4",
"expires_in": 3600,
"token_type": "Bearer",
"scope": "api_informat_sas_leerlingen.voorinschrijvingen.123456"
}
```

Request token for **pre-registrations** calls - multiple scopes

Doing a request for multiple scopes can be done by defining all the scopes, separated by a single space.

Request

```
POST https://<identityEndpoint>/connect/token     HTTP/1.1
...
Content-Type: application/x-www-form-urlencoded

grant_type: client_credentials
client_id: <received client id>
client_secret: <received client secret>
scope: api_informat_sas_leerlingen.voorinschrijvingen.123456
api_informat_sas_leerlingen.voorinschrijvingen.456789
```

Response

```
HTTP/1.1 200 OK
...

{
"access_token":"eyJhbGciOiJSUzI1NiIsImtpZCI6IjYyOEM3NzFEM0FCNEI0Rjk4MzRFODkyMTcyMTc2OTk5NjkyNTZFOEQiLCJ
0eXAiOiJKV1QiLCJ4NXQiOiJZb3gzSFRxMHRQbURUb2toY2hkcG1Xa2xibzAifQ.eyJuYmYiOjE1OTM2NzM1MDcsImV4cCI6MTU5MzY
3NzEwNywiaXNzIjoiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUiLCJhdWQiOlsiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZX
IuYmUvcmVzb3VyY2VzIiwiYXBpX2luZm9ybWF0X3Nhc19sZWVybGluZ2VuIl0sImNsaWVudF9pZCI6ImluZm9ybWF0X2N1c3RvbWVyX
2hoc2Nob2xlbiIsInNjb3BlIjpbImFwaV9pbmZvcm1hdF9zYXNfbGVlcmxpbmdlbi52b29yaW5zY2hyaWp2aW5nZW4uMDMyODQ3Il19
.vfTPZYPGGsbbl4Vy1S-FJdvPQTDlcT5m4E2mWFPqPMRBIQINqQBiNfxpH8DUu5cMRvcTFJxCeAsm33Rf-
my0jR1qVBYIlrKgmbIth_gyCNAycHUBAUsd-nYIJIv3Jz20_zM1L3gklUL48fCYc-
hwT1rbS4uRIpHQxDd_sdsNyvUjB9DgMCdxuoqBm0cVzmVvHlH6Fpez-
ttQ5Aymsj0dx0JvAIItVjxVHPqRwA33Sl1iSVogOjfjHP437d4ztFOjzAGUHglwO4eF10ALe_d8D1w3CqjAJGOaqP_W9ttPTATuPvwK
Q2xYaGiL1kPsejTpoRCd176vjv9i3b1dxPiw6FcVffeMUguetub8l8mHVEd0eYEDKUoTmZtp_bAhu5sUq-
eOC3GcJtGwC9TfjTQLtTdgniEfqhX81TZI3Kh2th4z5SOROKt5Sq5RtnN9U9WDrci2DQrfkt5pNg7leMIMqF8AsvuYp5rXGR-
HYWN2YB1uf2YTwIa-wOCshixjdzV4",
"expires_in": 3600,
"token_type": "Bearer",
"scope": "api_informat_sas_leerlingen.voorinschrijvingen.123456
api_informat_sas_leerlingen.voorinschrijvingen.456789"
}
```

Request token for **pupil related** calls - 1 scope

Request

```
POST https://<identityEndpoint>/connect/token     HTTP/1.1
...
Content-Type: application/x-www-form-urlencoded

grant_type: client_credentials
client_id: <received client id>
client_secret: <received client secret>
scope: api_informat_sas_leerlingen.leerlingen.123456
```

Response

```
HTTP/1.1 200 OK
...

{
"access_token":"eyJhbGciOiJSUzI1NiIsImtpZCI6IjYyOEM3NzFEM0FCNEI0Rjk4MzRFODkyMTcyMTc2OTk5NjkyNTZFOEQiLCJ
0eXAiOiJKV1QiLCJ4NXQiOiJZb3gzSFRxMHRQbURUb2toY2hkcG1Xa2xibzAifQ.eyJuYmYiOjE1OTM2NzM1MDcsImV4cCI6MTU5MzY
3NzEwNywiaXNzIjoiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUiLCJhdWQiOlsiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZX
IuYmUvcmVzb3VyY2VzIiwiYXBpX2luZm9ybWF0X3Nhc19sZWVybGluZ2VuIl0sImNsaWVudF9pZCI6ImluZm9ybWF0X2N1c3RvbWVyX
2hoc2Nob2xlbiIsInNjb3BlIjpbImFwaV9pbmZvcm1hdF9zYXNfbGV1cmxpbmdlbi52b29yaW5zY2hyaWp2aW5nZW4uMDMyODQ3Il19
.vfTPZYPGGsbbl4Vy1S-FJdvPQTDlcT5m4E2mWFPqPMRBIQINqQBiNfxpH8DUu5cMRvcTFJxCeAsm33Rf-
my0jR1qVBYIlrKgmbIth_gyCNAycHUBAUsd-nYIJIv3Jz20_zM1L3gklUL48fCYc-
hwT1rbS4uRIpHQxDd_sdsNyvUjB9DgMCdxuoqBm0cVzmVvHlH6Fpez-
ttQ5Aymsj0dx0JvAIItVjxVHPqRwA33Sl1iSVogOjfjHP437d4ztFOjzAGUHglwO4eF10ALe_d8D1w3CqjAJGOaqP_W9ttPTATuPvwK
Q2xYaGiL1kPsejTpoRCd176vjv9i3b1dxPiw6FcVffeMUguetub8l8mHVEd0eYEDKUoTmZtp_bAhu5sUq-
eOC3GcJtGwC9TfjTQLtTdgniEfqhX81TZI3Kh2th4z5SOROKt5Sq5RtnN9U9WDrci2DQrfkt5pNg7leMIMqF8AsvuYp5rXGR-
HYWN2YB1uf2YTwIa-wOCshixjdzV4",
"expires_in": 3600,
"token_type": "Bearer",
"scope": "api_informat_sas_leerlingen.leerlingen.123456"
}
```

Request token for **pupil related** calls - multiple scopes

Doing a request for multiple scopes can be done by defining all the scopes, separated by a single space.

Request

```
POST https://<identityEndpoint>/connect/token     HTTP/1.1
...
Content-Type: application/x-www-form-urlencoded

grant_type: client_credentials
client_id: <received client id>
client_secret: <received client secret>
scope: api_informat_sas_leerlingen.leerlingen.123456 api_informat_sas_leerlingen.leerlingen.456789
```

Response

```
HTTP/1.1 200 OK
...

{
"access_token":"eyJhbGciOiJSUzI1NiIsImtpZCI6IjYyOEM3NzFEM0FCNEI0Rjk4MzRFODkyMTcyMTc2OTk5NjkyNTZFOEQiLCJ
0eXAiOiJKV1QiLCJ4NXQiOiJZb3gzSFRxMHRQbURUb2toY2hkcG1Xa2xibzAifQ.eyJuYmYiOjE1OTM2NzM1MDcsImV4cCI6MTU5MzY
3NzEwNywiaXNzIjoiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUiLCJhdWQiOlsiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZX
IuYmUvcmVzb3VyY2VzIiwiYXBpX2luZm9ybWF0X3Nhc19sZWVybGluZ2VuIl0sImNsaWVudF9pZCI6ImluZm9ybWF0X2N1c3RvbWVyX
2hoc2Nob2xlbiIsInNjb3BlIjpbImFwaV9pbmZvcm1hdF9zYXNfbGVlcmxpbmdlbi52b29yaW5zY2hyaWp2aW5nZW4uMDMyODQ3Il19
.vfTPZYPGGsbbl4Vy1S-FJdvPQTDlcT5m4E2mWFPqPMRBIQINqQBiNfxpH8DUu5cMRvcTFJxCeAsm33Rf-
my0jR1qVBYIlrKgmbIth_gyCNAycHUBAUsd-nYIJIv3Jz20_zM1L3gklUL48fCYc-
hwT1rbS4uRIpHQxDd_sdsNyvUjB9DgMCdxuoqBm0cVzmVvHlH6Fpez-
ttQ5Aymsj0dx0JvAIItVjxVHPqRwA33Sl1iSVogOjfjHP437d4ztFOjzAGUHglwO4eF10ALe_d8D1w3CqjAJGOaqP_W9ttPTATuPvwK
Q2xYaGiL1kPsejTpoRCd176vjv9i3b1dxPiw6FcVffeMUguetub8l8mHVEd0eYEDKUoTmZtp_bAhu5sUq-
eOC3GcJtGwC9TfjTQLtTdgniEfqhX81TZI3Kh2th4z5SOROKt5Sq5RtnN9U9WDrci2DQrfkt5pNg7leMIMqF8AsvuYp5rXGR-
HYWN2YB1uf2YTwIa-wOCshixjdzV4",
"expires_in": 3600,
"token_type": "Bearer",
"scope": "api_informat_sas_leerlingen.leerlingen.123456 api_informat_sas_leerlingen.leerlingen.456789"
}
```

Use of access token

Each API Request must send with the access token as follows

Authorization: BEARER <token>

Example

```
GET https://leerlingenapi.informatsoftware.be/... HTTP/1.1
Authorization: BEARER
eyJhbGciOiJSUzI1NiIsImtpZCI6IjYyOEM3NzFEM0FCNEI0Rjk4MzRFODkyMTcyMTc2OTk5NjkyNTZFOEQiLCJ0eXAiOiJKV1QiLCJ
4NXQiOiJZb3gzSFRxMHRQbURUb2toY2hkcG1Xa2xibzAifQ.eyJuYmYiOjE1OTM2NzM1MDcsImV4cCI6MTU5MzY3NzEwNywiaXNzIjo
iaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUiLCJhdWQiOlsiaHR0cHM6Ly93d3cuaWRlbnRpdHlzZXJ2ZXIuYmUvcmVzb3VyY2
VzIiwiYXBpX2luZm9ybWF0X3Nhc19sZWVybGluZ2VuIl0sImNsaWVudF9pZCI6ImluZm9ybWF0X2N1c3RvbWVyX2hoc2Nob2xlbiIsI
nNjb3BlIjpbImFwaV9pbmZvcm1hdF9zYXNfbGVlcmxpbmdlbi52b29yaW5zY2hyaWp2aW5nZW4uMDMyODQ3Il19.vfTPZYPGGsbbl4V
y1S-FJdvPQTDlcT5m4E2mWFPqPMRBIQINqQBiNfxpH8DUu5cMRvcTFJxCeAsm33Rf-my0jR1qVBYIlrKgmbIth_gyCNAycHUBAUsd-
nYIJIv3Jz20_zM1L3gklUL48fCYc-hwT1rbS4uRIpHQxDd_sdsNyvUjB9DgMCdxuoqBm0cVzmVvHlH6Fpez-
ttQ5Aymsj0dx0JvAIItVjxVHPqRwA33Sl1iSVogOjfjHP437d4ztFOjzAGUHglwO4eF10ALe_d8D1w3CqjAJGOaqP_W9ttPTATuPvwK
Q2xYaGiL1kPsejTpoRCd176vjv9i3b1dxPiw6FcVffeMUguetub8l8mHVEd0eYEDKUoTmZtp_bAhu5sUq-
eOC3GcJtGwC9TfjTQLtTdgniEfqhX81TZI3Kh2th4z5SOROKt5Sq5RtnN9U9WDrci2DQrfkt5pNg7leMIMqF8AsvuYp5rXGR-
HYWN2YB1uf2YTwIa-wOCshixjdzV4
```

Request

It is mandatory that you provide some information with every request:

- A timestamp
- Your institute number

Institute number

Your institute is added as a header to **every** request. This header is called “InstituteNo”.

Note that header names are case-insensitive.

C# coding example

```
Request.Headers.Add("InstituteNo", "999999");
```

Please make sure that you only add this header once to the request, if you try to add it twice (or with multiple values), the header will be sent as a comma-separated list of values.

Timestamp

You need to add a timestamp header to every request, containing the current date and time, in UTC format.

The timestamp header looks like this:

```
Timestamp: 2019-10-15T15:30:22.000
```

The following format is also acceptable:

```
Timestamp: 2019-10-15T15:30:22+0100
```

Resources

This chapter lists all the resources available on the webservice. A resource can be thought of as an entity which you can get data for.

By default, calling a resource directly will provide you with a list of that resource, while adding an identifier to the URL will retrieve data for a unique resource.

For example: calling `/<endpoint>/students` will get you a list of students, while calling `/<endpoint>/students/123` will get you the data for the specific student with ID 123.

On a unique resource, like student 123, you can call other linked resources, to get specific data related to that student. For example, calling `/<endpoint>/students/123/history` will get you a dataset that contains the students' history.

Pre-registrations

This chapter describes how to:

- Add your reservation data into Informat as a Pre-registration
- Update an existing pre-registration based on the preRegistrationId
- Delete an existing pre-registration by its preRegistrationId
- Get the status of a pre-registration/registration by its preRegistrationId

It describes all the available methods, their request/response format and provides input/output examples.

Scope pattern: api_informat_sas_leerlingen.voorinschrijvingen.{instituteNo}

POST /preregistrations/save

This call can be used to:

- Add your new online reservation into Informat as a Pre-registration;
- Update an existing pre-registration based on the preRegistrationId;
- Add an "update personal details"-registration by using "000000" as admgrpId (means no admgrp), used when the original pre-registration is already accepted and can no longer be changed;
- Update an existing "update personal details"-registration based on the preRegistrationId provided during addition.

When updating, several MNA (Method Not Allowed) validation-errors can be thrown. An overview can be found over here.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

No url parameters required.

Body

Field	Type	Required/optional	Description
lastName	string	required	Pupil's last name **Validation**: `required and max. 50 characters`
firstName	string	required	Pupil's first name **Validation:** `required and max. 50 characters`
additionalNames	string	optional	Additional names of the pupil **Validation**: `max. 160 characters`
dateOfBirth	date	required	Date of birth **Format**: valid UTC Date or `yyyy-MM-dd` **Validation:** `required and valid`
countryOfBirthCode	string	required	Country of birth code must be one of the official country code. **Validation:** `length must be 5 characters and must be an official country code.`
placeOfBirthCode	string	optional	Place of birth code is a postal code. If the country of birth is Belgium then it must be an official BE postal code.

Field	Type	Required/optional	Description
			Validation: `max. 10 characters`
placeOfBirth	string	optional	Place of birth. **Validation**: `max. 50 characters`
nationalityCode	string	required	Nationality code must be one of the official nationality codes. **Validation**: `length must be 5 characters and must be an official code.`
sex	integer	required	Possible values for the sex are: 1 or 2 1 = male 2 = female **Validation**: `must have a value of 1 or 2`
insz	string	optional	The pupil's identification number. This is either the Bisnummer, for foreign pupil, or Rijksregisternummer for Belgium residents. **Format**: "yymmddxxxxx" **Example**: `"85073115025"` **Validation:** `Validates the number, no check on date of birth or gender.`
eIdNo	string	optional	Some (electronic) residence documents provide/contain an Id number. This number can be stored by filling up this field. **Validation**: `max. 12 characters`
eIdPhoto	string	optional	Some electronic residence documents provide a photo of the owner. The data must be offered as a delimited string.
mobilePhone	string	optional	Own mobile number **Validation**: `max. 20 characters`
email	string	optional	Private email address **Validation**: `Validation on email-address format`
nameOfDoctor	String	optional	Name of the pupil's doctor **Validation**: `max. 100 characters`
phoneOfDoctor	String	optional	Phone of the pupil's doctor **Validation**: `max. 20 characters`
firstLanguage	string	optional	**Validation**: `max. 100 characters`
isHomeless	boolean	required	Indicates whether the pupil is someone who changes shelter or place of residence frequently.
migrating	boolean	required	Indicates whether the pupil belongs to the migratory (trekkende) population.
isIndicatorPupil	boolean	required	Indicates whether the pupil is an indicator pupil. A pupil who meets at least one of the official indicators, will be assigned as an indicator pupil.
religion	integer	optional	Also known as levensbeschouwing. **Example:** `63`

Field	Type	Required/optional	Description

			Possible values:
			`52 = Cultuurbeschouwing`
			`63 = Eigen cultuur en religie`
			`135 = Islamitische godsdienst`
			`136 = Israëlische godsdienst`
			`140 = Katholieke godsdienst`
			`187 = Niet-confessionele zedenleer`
			`194 = Orthodoxe godsdienst`
			`225 = Protestants-evangelische godsdienst`
			`418 = Anglicaanse godsdienst`
			`9999 = Vrijstelling of niet van toepassing`
			
			Validation: `must be one of the possible values`

priorityGroup	integer	optional	Priority group, also known as voorrangsgroep
			Example: 1
			Possible values:
			`1 = Dezelfde leefentiteit`
			`2 = Kind van personeelslid`
			`3 = Kind met minstens 1 ouder die het Nederlands machtig is`
			`4 = Campusleerling`
			`5 = Indicator- of niet-indicatorleerling`
			`6 = Geen`
			Validation: `must be one of the possible values`

reasonForRefusal	integer	optional	**Example:** 19 **Possible values:** `19 = capaciteit` **Validation**: `must be one of the possible values`
Relationships	**array**	optional	Validation: MAX(2)

type	integer	required	Type of relationship.
			Example: 13
			Possible values:
			`0 = leerling`
			`13 = vader`
			`14 = moeder`
			`5 = plusvader`
			`6 = plusmoeder`
			`2 = voogd`
			`9 = grootvader`
			`10 = grootmoeder`
			`7 = pleegvader`
			`8 = pleegmoeder`
			Note: Type 0 (leerling) isn't an official relationship type. The purpose of this type is to be able to send the address of an adult pupil without creating a relationship. Can, for example, be applied to nursing. Consequently, it doesn't really make sense to define lastName, firstName, insz, phone, … in this case.
			Validation: `must be one of the possible values`

Field	Type	Required/optional	Description
lastName	string	**required** / optional	Relation's last name **Note:** This field is only optional in combination with type 0 (leerling). **Validation:** `required and max. 50 characters`
firstName	string	**required** / optional	Relation's first name **Note:** This field is only optional in combination with type 0 (leerling). **Validation:** `required and max. 30 characters`
Address	object	optional	
streetName	string	required	Street name **Validation**: `max. 50 characters`
houseNo	string	required	Includes the House number & Alpha number **Validation**: `max. 10 characters`
houseBusNo	string	optional	Bus number **Validation**: `max. 6 characters`
postalCode	string	required	Postal code must an official one. **Validation**: `max. 8 characters`
city	string	required	City. **Validation**: `max. 30 characters`
countryCode	string	required	Country code must be one of the official country codes. **Validation:** `length must be 5 characters and must appear in the list of "countries".`
isDomicileAddress	boolean	required	Indicates whether the defined address is a domicile address
isWritingAddress	boolean	required	Indicates whether the defined address is a writing address
isResidenceAddress	boolean	required	Indicates whether the defined address is residence address
phone	string	optional	Domicile phone number **Validation**: `max. 20 characters`
mobilePhone	string	optional	Own mobile number **Validation**: `max. 20 characters`
email	string	optional	Private email address **Validation**: `Validation on email-address format`
insz	string	optional	The relation's national registration number. This is either the Bisnummer, for foreign pupil, or Rijksregisternummer for Belgium residents. **Format**: "yymmddxxxxx" **Example**: `"85073115025"` **Validation:** `Validates the number`

Field	Type	Required/optional	Description
preRegistrationId	GUID	required	A mandatory unique identifier which will be used to identify the pre-registration. Other calls like the update, consult, … are based on this Id. **Validation:** `Id Not empty`
schoolyear	string	Required	The school year in which the pupil's registration takes place. **Validation**: `format: "2019-20"`
institute	string	required	Official institute number **Validation**: `6 characters (digits)`
structure	string	required	'Hoofdstructuur' of the institute (111, 311,…) **Validation**: `3 characters`
locationId	string	required	Official Discimus location number **Validation**: `3 characters (digits) & if formatted correctly, a check is done to verify whether this id exists within the students admin module for the provided instituteNo and school year. See error code "BR037C" for the error description.`
admgrpId	string	required	Official education number (Administratieve groep) **Validation**: `6 characters (digits) and must be an existing admgrp`
admgrpDetail	string	optional	Additional information about the education (1ste jaar kleuter, Latijn,…) **Validation**: `max 200 characters`
preRegistrationDate	date	optional/ required	Pre-registration date and time of the online reservation/pre-registration. **Format**: valid UTC Date **Validation**: `Pre-Registrationdate is only required if admgrpId is not equal to "000000" (no admgrp). And if provided, this date must be on or before the Start Date.`
startDate	date	required	The date the pupil starts the lessons. Startdate shouldn't be Saturday or Sunday, except on September 1st. **Format**: valid UTC Date or `yyyy-MM-dd` **Validation**: `Start Date is required and must be between beginning and end of defined school year.`
registrationStatus	integer	required	Possible values for the registration status: 0 or 1 0 = Realized (gerealiseerd) 1 = Not realized (niet-gerealiseerd) **Validation**: `Must have a value of 0 or 1.`
remark	string	optional	Remark on registration level **Validation**: `no length validation.`

Field	Type	Required/optional	Description
assignedViaRegistrationSystem	boolean	optional	Indicates whether this pre-registration is assigned via a registration system (aanmeldingssysteem). The (default) value is false, if not provided. **Format**: true or false

Responses

A response without validation errors, contains a "preRegistrationStatus"-object.

Field	Type	Description
preRegistrationId	GUID	Unique identifier which identifies the pre-registration. Determined during the creation of the pre-registration.
status	string	Status of the pre-registration. The status will always be "open" after a creation. Possible values: `open, accepted, refused`

Example request

```
POST https://<endpoint>/<version>/preregistrations/save HTTP/1.1
InstituteNo: 999999
Timestamp: 2020-02-29T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
Content-Length: 1044
{
   "lastName": "Eastwood",
   "firstName": "Scott",
   "additionalNames": null,
   "dateOfBirth": "2010-04-01",
   "countryOfBirthCode": "00150",
   "placeOfBirthCode": "8600",
   "placeOfBirth": null,
   "nationalityCode": "00150",
   "sex": 1,
   "insz": "10040115670",
   "eIdNo": "1234",
   "eIdPhoto": "",
   "mobilePhone": "",
   "email": "Scott.Eastwood@telenet.be",
   "nameOfDoctor": "Ever young",
   "phoneOfDoctor": null,
   "firstLanguage": "Vlaams",
   "isHomeless": false,
   "migrating": false,
   "isIndicatorPupil": true,
   "religion": null,
   "priorityGroup": null,
   "reasonForRefusal": null,
   "Relationships":
   [
    {
       "type": 13,
       "lastName": "Eastwood",
       "firstName": "Clint",
       "phone": null,
       "mobilePhone": null,
       "email": "TisTeZot@telenet.be",
       "insz": "80031516717",
       "Address":
       {
          "streetName": "Nijverheidslaan",
          "houseNo": "120A",
          "houseBusNo": null,
          "postalCode": "8600",
          "city": "Diksmuide",
          "countryCode": "00150",
          "isDomicileAddress": true,
          "isWritingAddress": false,
          "isResidenceAddress": true
       }
    }
   ],

   "preRegistrationId": "EA840317-856F-4280-BE8F-DE0A46E2935F",
   "schoolyear": "2020-21",
   "institute": "123456",
   "structure": "211",
   "locationId": "001",
   "admgrpId": "040102",
   "admgrpDetail": "Test jaar",
```

```
    "preRegistrationDate": "2020-06-05T10:15",
    "startDate": "2020-09-01",
    "registrationStatus": 1,
    "remark": "Mijn eerste test registratie"
    "assignedViaRegistrationSystem": true
}
```

Example response

Response without validation errors

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Date: Sat, 29 Feb 2020 09:10:46 GMT
Content-Length: 212
{
    "preRegistrationStatus": {
         "preRegistrationId": "EA840317-856F-4280-BE8F-DE0A46E2935F",
         "status": "open"
     },
     "status": 200,
     "message": null
}
```

Response with validation errors

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Date: Thu, 07 May 2020 09:10:46 GMT
Content-Length: 212
{
   "errors": [
         {
             "code": "BR010A",
             "message": "Invalid Insz"
         },
         {
             "code": "BR035B",
             "message": "Institute must have a lenght of 6 characters"
         },
          {…}
     ],
     "status": 400,
     "message": "One or more validation errors occurred. See errors for more details."
}
```

DELETE /preregistrations/{preRegistrationId}

Delete your online Pre-registration data.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
preRegistrationId	GUID	required	The unique identifier of the pre-registration. **Format**: `D919481D-6AFD-402C-B97E-795A8075503B`

Body

This request has an empty body.

Response

The response contains the “preRegistrationStatus”-object, but will always be null since the pre-registration was removed.

Example request

Request

```
DELETE https://<endpoint>/<version>/preregistrations/D919481D-6AFD-402C-B97E-795A8075503B HTTP/1.1
InstituteNo: 999999
Timestamp: 2019-11-01T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
Content-Length: 0
```

Example response

Response without errors

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Thu, 07 May 2020 09:10:46 GMT
Content-Length: 212
{
    "preRegistrationStatus": null,
    "status": 200,
    "message": null
}
```

Response with error

```
HTTP/1.1 405 Method Not Allowed
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Thu, 07 May 2020 09:10:46 GMT
Content-Length: 212
{
   "errors": [
        {
            "code": "MNA004",
            "message": "This Pre-Registration has been refused, and can no longer be deleted"
        }
    ],
    "status": 405,
    "message": "Not allowed exception"
}
```

GET /preregistrations/{preregistrationId}/status

Gets the status of a pre-registration/registration by its preRegistrationId. This is the same response as you should get after a successful save operation.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
preRegistrationId	GUID	required	The unique identifier of the pre-registration/registration. **Format**: `D919481D-6AFD-402C-B97E-795A8075503B`

Body

This request has an empty body.

Response

The response contains the "preRegistrationStatus"-object, mentioned earlier.

Field	Type	Description
preRegistrationId	GUID	Unique identifier which identifies the pre-registration. Determined during the creation of the pre-registration.
status	string	Status of the pre-registration/registration. Possible values are: `open, accepted, refused`

Example request

Request

```
GET https://<endpoint>/<version>/preregistrations/D919481D-6AFD-402C-B97E-795A8075503B/status HTTP/1.1
InstituteNo: 999999
Timestamp: 2019-11-01T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
Content-Length: 0
```

Example response

Response for an existing <preregistrationId> of a pre-registration/registration

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Thu, 01 Nov 2019 09:10:46 GMT
Content-Length: 212
{
   "preRegistrationStatus":    {
      "preRegistrationId": "d919481d-6afd-402c-b97e-795a8075503b",
      "status": "open"
   },
   "status": 200,
   "message": null
}
```

Response for a non-existing <preregistrationId>

```
HTTP/1.1 404 Not Found
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Thu, 07 May 2020 09:10:46 GMT
Content-Length: 114
{
   "errors": [
        {
            "code": "DNF001",
            "message": "No registration found for Pre-Registration Id 564dc62a-e531-48b0-bcdc-
93cb834cb5e8"
        }
    ],
    "status": 404,
    "message": "Data not found exception"
}
```

Registrations

This chapter describes how to:

- Get all registrations for an institute number and school year.

It describes all the available methods, their request/response format and provides input/output examples.

Scope pattern: api_informat_sas_leerlingen.leerlingen.{instituteNo}

GET /registrations?schoolYear={schoolYear}&changedSince={changedSince}

Gets all the registrations for the combination institute number and school year. The institute number is defined via the mandatory header "InstituteNo" and the school year must be send via the URL.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
schoolYear	string	required	Limits the output results to registrations within the given schoolyear. **Format:** "2022-23" If not provided a bad request with error code "BR001" and message "Invalid school year" will be returned.
changedSince	date	optional	Limits the output results to those registrations whose data has been changed since a certain date. Our internal "changedSince" date is determined based on a collection of change dates at different levels (depending on the content of the response). No results will return in case: • nothing has been changed; • the "changedSince"-date sent, lays too far in the past.

Body

This request has an empty body.

Response

The response contains a list of registrations.

Field	Type	Description
inschrijvingsId	GUID	Unique identifier for the registration **Format**: {42CD5A20-E22F-45D7-973C-FAB8734DEC50}
pInschr	integer	Unique integer-value for the registration within the database **Note:** This value may have changed after data migration. Therefore, the inschrijvingsId is more suitable as a reference value.
pPersoon	integer	Unique integer-value for the student within the database
persoonId	GUID	Unique identifier for the student **Format**: {42CD5A20-E22F-45D7-973C-FAB8734DEC50}
instelnr	string	Official institute number of the school

		Format: `6 characters (digits)`
hfdstructuur	string	Structure of the institute (111, 311,…)
school	string	Descriptive name of the school
stamnr	string	Students' stamnummer **Format**: `9 characters (digits) or null if not available`
vestCode	string	Own defined location code
vestiging	string	Own descriptive name of the location
begindatum	date	Date the registration starts **Format**: : yyyy-MM-dd
einddatum	date	Date the registration ends **Format**: : yyyy-MM-dd or null if empty
afdCode	string	Own defined department code
nrAdmgrp	string	Official education number (Administratieve groep) **Format**: `6 characters (digits)`
afdelingsjaar	string	Department year

status	integer	Registration status
		Possible values:
		`0 = Gerealiseerd`
		`1 = Niet-gerealiseerd`
		`2 = Uitgesteld`
		`3 = Parallel`
		`4 = Aanmelding`

graad	string	Degree
leerjaar	int	Grade
taalkeuze	string	Choice of language-options

finCode	string	Finance ability code
		Possible values secundary education:
		`01 = Vrije leerling`
		`02 = Regelmatig / financierbaar`
		`03 = Regelmatig / niet financierbaar`
		`11 = Vrije leerling in erkende privéschool`
		`13 = Regelmatige leerling in erkende privéschool`
		`20 = Financierbare GON-leerling`
		`22 = Niet-financierbare GON-leerling`
		`99 = Onder voorbehoud aanvaarde leerling`
		Possible values primary education:
		`01 = Coëfficiënt 1 - 100% financierbaar`
		`02 = Coëfficiënt 1,5 - 150% financierbaar`
		`03 = Niet financierbaar, maar telt wel mee voor de leerplichtcontrole en de financieringswet`
		`11 = Vrije leerling in erkende privéschool`
		`13 = Regelmatige leerling in erkende privéschool`
		`20 = Financierbare GON-leerling`
		`22 = Niet-financierbare GON-leerling`

levensbeschouwingCode	string	Religion code
		Possible values:
		`0000 = Blanco`
		`52 = Cultuurbeschouwing`
		`63 = Eigen cultuur en religie`
		`135 = Islamitische godsdienst`
		`136 = Israëlische godsdienst`
		`140 = Katholieke godsdienst`

		187 = Niet-confessionele zedenleer 194 = Orthodoxe godsdienst 225 = Protestants-evangelische godsdienst 418 = Anglicaanse godsdienst 9999 = Vrijstelling of niet van toepassing
isOkan	bool	Indicates whether it is a registration for a foreign-speaking new student
preRegistrationId	GUID	Unique identifier for the pre-registration, linked to a registration. This Id is provided by you when adding a new pre-registration via the "/preregistrations/save"-call. The preRegistrationId can only be returned when the pre-registration is registered via the API and accepted via the student administration module. **Format**: `{42CD5A20-E22F-45D7-973C-FAB8734DEC50} or null if not available`
inschrklassen	array	List of class registrations
inschrKlasId	GUID	Unique identifier for the class registration **Format**: `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
pInschrKlas	integer	Unique integer-value for the class registration within the database. **Note:** This value may have changed after data migration. Therefore, the inschrKlasId is more suitable as a reference value.
klasId	GUID	Unique identifier for the class **Format**: `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
pklas	integer	Unique integer-value for the class within the database. **Note:** This value may have changed after data migration. Therefore, the klasId is more suitable as a reference value.
klasCode	string	Own class code
klas	string	Own class name
groepType	integer	Class type **Possible values:** 0 = Official/main class 1 = Sub class 3 = OLOD group
klasnummer	integer	Student's class number
begindatum	date	Date the class registration starts **Format**: : yyyy-MM-dd
einddatum	date	Date the class registration ends **Format**: : yyyy-MM-dd or null if empty

Example request

Request

```
GET https://<endpoint>/<version>/registrations?schoolYear=2022-23 HTTP/1.1
InstituteNo: 999999
Timestamp: 2023-01-09T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Mon, 09 Jan 2023 16:06:30 GMT

{
   "registrations": [
        {
            "inschrijvingsId": "bd26bb65-f54a-488d-866b-e1f9927d6be5",
            "pInschr": 175926,
            "pPersoon": 69780,
            "persoonId": "5b8f64c5-3968-4b13-b962-e4acda9601bc",
            "instelnr": "999999",
            "hfdstructuur": "311",
            "school": "TEST GO! Technisch Atheneum",
            "stamnr": "20200688",
            "vestCode": "molenst2",
            "vestiging": "Molenstraat 2",
            "begindatum": "2021-09-01T00:00:00",
            "einddatum": null,
            "afdCode": "Grieks",
            "nrAdmgrp": "040092",
            "afdelingsjaar": "2° jaar Grieks",
            "status": 1,
            "graad": "1",
            "leerjaar": 2,
            "taalkeuze": "Frans - Engels",
            "finCode": "02",
            "levensbeschouwingCode": "0000",
            "isOkan": false,
            "preRegistrationId": "53a3bb93-d269-46b5-86fa-08193bd4ba30",
            "inschrKlassen": [
                {
                    "inschrKlasId": "a6a278e3-52c1-42c0-8e03-85331a5e683f",
                    "pInschrKlas": 310301,
                    "klasId": "82c5ad0d-8f0c-4b9d-b2de-752c7e5a3bd4",
                    "pKlas": 26888,
                    "klasCode": "2B HO",
                    "klas": "2B HO2",
                    "groepType": 0,
                    "klasnummer": 0,
                    "begindatum": "2021-09-01T00:00:00",
                    "einddatum": "2022-01-18T00:00:00"
                }
            ]
        },
        {…}
   ]
}
```

Response for missing school year:

```
HTTP/1.1 400 Bad Request
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Thu, 07 May 2020 09:10:46 GMT

{
    "errors": {
        "code": "BR001",
        "message": "Invalid school year ''"
    },
    "status": 400,
    "message": "Invalid school year"
}
```

Students

This chapter describes how to:

- Get all students with a registration for an institute number, school year and reference date;
- Get a student by studentId.

It describes all the available methods, their request/response format and provides input/output examples.

Scope pattern: api_informat_sas_leerlingen.leerlingen.{instituteNo}

GET /students?schoolYear={schoolYear}&refdate={referenceDate}&changedSince={changedSince}

Gets all students with a registration for a given institute number, school year and reference date falling between registration's begin & end date.
The institute number is defined via the mandatory header "InstituteNo" and the school year must be send via the URL.
The ReferenceDate is optional, but will be calculated automatically if not provided (see request for more information).

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
schoolYear	string	required	Limits the output results to students which have a registration within the given schoolyear. **Format:** "2022-23" If not provided a bad request with error code "BR001" and message "Invalid school year" will be returned.
referenceDate	date	optional	Limits the output results to students which have a registration where the reference date falls between registration's begin & end date. If the reference date is not provided, today's date will be taken. **In any case the reference date will be overridden as follows if it falls outside the boundaries of the provided school year:** - Reference date < beginning of school year => reference date will be overridden with begin date of school year; - Reference date > end of school year => reference date will be overridden with end date of school year.
changedSince	date	optional	Limits the output results to those students whose data has been changed since a certain date. Our internal "changedSince" date is determined based on a collection of change dates at different levels (depending on the content of the response).

Body

This request has an empty body.

Response

The response contains a list of students.

Field	Type	Description
pPersoon	integer	Unique integer-value for the student within the database.
persoonId	GUID	Unique identifier for the student. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
naam	string	Student's last name.
voornaam	string	Student's first name.
geboortedatum	date	Student's date of birth. **Format:** yyyy-MM-dd or null if empty
nickname	string	Student's nickname.
voornaam2	string	Student's additional names.
initialen	string	Student's initials.
geboorteland	string	Student's country of birth.
geboorteplaats	string	Student's place of birth.
nationaliteitCode	string	Student's official nationality codes.
rijksregisternr	string	Student's national registration number for Belgium residents. **Format:** `"yymmddxxxxx" if provided`
bisnr	string	Students national registration number for a foreign person. **Format:** `11 characters (digits) if provided`
geslacht	string	Student's sex. **Possible values:** "M" or "V"
huisdokter	string	Name of the student's doctor.
telefoonHuisdokter	string	Phone number of the student's doctor.
llOpSchool	integer	Student's place in line at school.
inschrijvingsId	GUID	Unique identifier of the actual registration. The **actual** registration is determined by the provided InstituteNo, school year, reference date (between registration's begin & end date) and registration status = 0 (gerealiseerd). If no such registration is found, inschrijvingsId will be null.
leerlingenkaartNummer	string	Studentcard number
fietsNummer	string	Bike number
adressen	array	List of domicile addresses (usually one).
pAdres	integer	Unique integer-value for the address within the database.
adresId	GUID	Unique identifier for the address. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
aanspreekTitel	string	Addressable title.
aanspreekNaam	string	Addressable name.
straat	string	Street name.
nr	string	House number & Alpha number.
bus	string	Bus number

postcode	string	Postal code (main).
gemeente	string	Town (main).
landcode	string	Country code
isFacturatie	bool	Indicates whether this address is a invoice address.
isAanschrijf	bool	Indicates whether this address is a writing address.
isVerblijf	bool	Indicates whether this address is a residence address.
isOverige	bool	Indicates whether this address is of another type then the provided ones.
percentage	decimal	Cost allocation percentage.
relaties	array	List of relationships.
pRelatie	integer	Unique integer-value for the relationship within the database.
relatieId	GUID	Unique identifier for the relationship. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
type	string	Relationship type **Examples:** Vader, Moeder, Plusvader, Plusmoeder, …
naam	string	Relation's last name
voornaam	string	Relation's first name
insz	string	Relation's national registration number. This is either the Bisnummer, for foreign pupil, or Rijksregisternummer for Belgium residents. **Format**: "yymmddxxxxx" **Example**: `"85073115025"`
geboortedatum	date	Relation's date of birth. **Format:** yyyy-MM-dd or null if empty
geslacht	string	Relation's sex. **Possible values:** "M" or "V"
nationaliteitCode	string	Relation's official nationality codes.
beroep	string	Relation's profession.
burgerlijkeStand	string	Relation's civil status.
lpv	integer	Indicates if the relation is marked as a school attendance officer and defines if it's the first or second one.
ophalen	bool	Indicates if the relation picks up the student from school.
isOverleden	bool	Indicates if the relation is deceased.
adressen	array	List of addresses linked to the relationship (usually one, and most likely a domicile address).
pAdres	integer	Unique integer-value for the address within the database.
adresId	GUID	Unique identifier for the address. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
aanspreekTitel	string	Addressable title.
aanspreekNaam	string	Addressable name.
straat	string	Street name.
nr	string	House number & Alpha number.
bus	string	Bus number
postcode	string	Postal code (main).

gemeente	string	Town (main).
landcode	string	Country code
isFacturatie	bool	Indicates whether this address is a invoice address.
isAanschrijf	bool	Indicates whether this address is a writing address.
isVerblijf	bool	Indicates whether this address is a residence address.
isOverige	bool	Indicates whether this address is of another type then the provided ones.
percentage	decimal	Cost allocation percentage.
comnrs	array	List of communication numbers linked to the relationship.
pComnr	integer	Unique integer-value for the communication number within the database.
nr	string	Communication number.
type	string	Communication number type.
soort	string	Communication number sort. **Possible values:** Telefoon, Fax or Gsm
emails	array	List of email addresses linked to the relationship.
pEmail	integer	Unique integer-value for the email address within the database.
email	string	Email address.
type	string	Email address type. **Examples:** Eigen, School, Privé, Ouders, …
schoolcom	bool	Indicates if this email address can be used for school communication purposes.
factuurMailen	bool	Indicates if this email address can be used to mail invoices.
comnrs	array	List of communication numbers **not** linked to a relationship.
pComnr	integer	Unique integer-value for the communication number within the database.
nr	string	Communication number.
type	string	Communication number type. **Examples:** Eigen, Privé nummer, Ouders, Vader, Moeder, …
soort	string	Communication number sort. **Possible values:** Telefoon, Fax or Gsm
emails	array	List of email addresses **not** linked to a relationship
pEmail	integer	Unique integer-value for the email address within the database.
email	string	Email address.
type	string	Email address type. **Examples:** Eigen, School, Privé, Ouders, …
schoolcom	bool	Indicates if this email address can be used for school communication purposes.
factuurMailen	bool	Indicates if this email address can be used to mail invoices.
bankrek	array	List of bank account linked to the student. Returns a “null” or an empty list, if not provided.

type	int	Bank account type.
		Possible values:
		`1 = Persoonlijk`
		`2 = Ouders`
		`3 = Vader`
		`4 = Moeder`
		`6 = Voogd`

		8 = Andere
iban	string	International bank account number.
bic	string	Bank Identification Code. **Format:** 8 to 11 characters

Example request

Request

```
GET https://<endpoint>/<version>/students?schoolYear=2022-23 HTTP/1.1
InstituteNo: 999999
Timestamp: 2023-01-09T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Tue, 10 Jan 2023 09:36:29 GMT

{
   "students": [
        {
            "pPersoon": 69780,
            "persoonId": "5b8f64c5-3968-4b13-b962-e4acda9601bc",
            "naam": "Van Damme",
            "voornaam": "Thomas",
            "geboortedatum": "2001-12-20T00:00:00",
            "nickname": " Van Damme Tom",
            "voornaam2": "",
            "initialen": "",
            "geboorteland": "België",
            "geboorteplaats": "KORTRIJK",
            "nationaliteitCode": "00150",
            "rijksregisternr": "12345678910",
            "bisnr": null,
            "geslacht": "M",
            "huisdokter": "Dr. Ever Young",
            "telefoonHuisdokter": "",
            "llOpSchool": 0,
            "inschrijvingsId": "724a9169-7e92-4f8e-a875-1d8dff1d6662",
            "leerlingenkaartNummer": "LK:012",
            "fietsnummer": "B012",
```

```
        "adressen": [
            {
                "pAdres": 80000,
                "adresId": "67c7633a-f3d9-4f82-b600-8bc617cf851c",
                "aanspreekTitel": "Aan de vader van",
                "aanspreekNaam": "Van Damme Thomas",
                "straat": "Stationsstraat",
                "nr": "100",
                 "bus": "2",
                "postcode": "8600",
                "gemeente": "DIKSMUIDE",
                "landcode": "00150",
                "isFacturatie": true,
                "isAanschrijf": true,
                "isVerblijf": true,
                "isOverige": false,
                "percentage": 100
            }
        ],
        "relaties": [
            {
                "pRelatie": 72936,
                "relatieId": "8a002367-18ac-45cd-a91c-493b2f317a69",
                "type": "Vader",
                "naam": "Van Damme",
                "voornaam": "Jean",
                "insz": null,
                "geboortedatum": "1976-03-06T00:00:00",
                "geslacht": "M",
                "nationaliteitCode": "00150",
                "beroep": "Zelfstandige",
                "burgerlijkeStand": "Gescheiden",
                "lpv": 1,
                "ophalen": false,
                "isOverleden": false,
                "adressen": [
                    {
                        "pAdres": 80000,
                        "adresId": "67c7633a-f3d9-4f82-b600-8bc617cf851c",
                        "aanspreekTitel": "Aan de vader van",
                        "aanspreekNaam": "Van Damme Thomas",
                        "straat": "Stationsstraat",
                        "nr": "100",
                        "bus": "2",
                        "postcode": "8600",
                        "gemeente": " DIKSMUIDE ",
                        "landcode": "00150",
                        "isFacturatie": true,
                        "isAanschrijf": true,
                        "isVerblijf": true,
                        "isOverige": false,
                        "percentage": 100
                    }
                ],
                "comnrs": [
                    {
                        "pComnr": 156687,
                        "nr": "0404 95 22 52",
                        "type": "Vader",
                        "soort": "Gsm"
                    }
                ],
```

```
            "emails": [
                {
                    "pEmail": 59294,
                    "email": "Jean.VanDamme@informat.be",
                    "type": "Vader",
                    "schoolcom": true,
                    "factuurMailen": false
                }
            ]
        },
        {
            "pRelatie": 72937,
            "relatieId": "06124ed9-bec9-45e7-b5d3-0f553ca77d66",
            "type": "Moeder",
            "naam": "Somers",
            "voornaam": "Melissa",
            "insz": null,
            "geboortedatum": "1981-06-08T00:00:00",
            "geslacht": "V",
            "nationaliteitCode": "00150",
            "beroep": "Bediende",
            "burgerlijkeStand": "Gescheiden",
            "lpv": 2,
            "ophalen": false,
            "isOverleden": false,
            "adressen": [
                {
                    "pAdres": 80000,
                    "adresId": "67c7633a-f3d9-4f82-b600-8bc617cf851c",
                    "aanspreekTitel": "Aan de moeder van",
                    "aanspreekNaam": "Van Damme Thomas",
                    "straat": "Grote markt",
                    "nr": "45",
                    "bus": "",
                    "postcode": "8600",
                    "gemeente": "DIKSMUIDE",
                    "isFacturatie": true,
                    "isAanschrijf": true,
                    "isVerblijf": true,
                    "isOverige": false,
                    "percentage": 100
                }
            ],
            "comnrs": [
                {
                    "pComnr": 156688,
                    "nr": "0492 09 52 25",
                    "type": "Moeder",
                    "soort": "Gsm"
                }
            ],
            "emails": [
                {
                    "pEmail": 22720,
                    "email": "Melissa.Somers@informat.be",
                    "type": "Moeder",
                    "schoolcom": true,
                    "factuurMailen": true
                }
            ]
        }
    ],
```

```
            "comnrs": [
                {
                    "pComnr": 156689,
                    "nr": "0492 25 55 29",
                    "type": "Noodnummer",
                    "soort": "Gsm"
                },
                {
                    "pComnr": 188384,
                    "nr": "025 04 09 52",
                    "type": "Domicilie",
                    "soort": "Telefoon"
                },
                {
                    "pComnr": 188385,
                    "nr": "0402 45 02 25",
                    "type": "Eigen",
                    "soort": "Gsm"
                },
                {
                    "pComnr": 234237,
                    "nr": "025 22 94 20",
                    "type": "Alternatief nummer",
                    "soort": "Telefoon"
                }
            ],
            "emails": [
                {
                    "pEmail": 58608,
                    "email": "Thomas.Vandamme@informat.be",
                    "type": "Eigen",
                    "schoolcom": true,
                    "factuurMailen": false
                }
            ],
            "bankrek": [
                {
                    "type": 2,
                    "iban": "BE12378901230178",
                    "bic": "BNAGBEBB"
                }
            ]
        },
        {…}
    ]
}
```


GET /students/{studentId}?schoolYear={schoolYear}&refdate={referenceDate}

Gets a student by it's studentId. The institute number, school year and reference date are only used to determine the actual registration (inschrijvingsId property).
The institute number is defined via the mandatory header "InstituteNo" and the school year must be send via the URL. The ReferenceDate is optional, but will be calculated automatically if not provided (see request for more information).

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
studentId	GUID	required	Limits the output results to a student with the provided studentId. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}` If not provided a bad request with error code "BR002" and message "Invalid student id" will be returned.
schoolYear	string	required	Is only used to determine the actual registration (inschrijvingsId property). So this parameter has no limiting effect on the output result. **Format:** `"2022-23"` If not provided a bad request with error code "BR001" and message "Invalid school year" will be returned.
referenceDate	date	optional	Is also only used to determine the actual registration (inschrijvingsId property). So this parameter has no limiting effect on the output result. If the reference date is not provided, today's date will be taken. **In any case the reference date will be overridden as follows if it falls outside the boundaries of the provided school year:** - Reference date < beginning of school year => reference date will be overridden with begin date of school year; - Reference date > end of school year => reference date will be overridden with end date of school year.

Body

This request has an empty body.

Response

The response contains one student instead of a list of students. This makes the response similar to the call "GET /students?schoolYear={schoolYear}&refdate={referenceDate}&changedSince={changedSince}".

Example request

Request

```
GET https://<endpoint>/<version>/students/c5806777-4bce-4d50-afc5-9cb47f1a5f14?schoolYear=2022-23
HTTP/1.1
InstituteNo: 999999
Timestamp: 2023-01-10T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Tue, 10 Jan 2023 09:36:29 GMT

{
   "student": {
            "pPersoon": 31236,
            "persoonId": "c5806777-4bce-4d50-afc5-9cb47f1a5f14",
            "naam": "Woodpecker",
            "voornaam": "Winnie",
            "geboortedatum": "2005-03-20T00:00:00",
            …
            SIMULAR TO THE GLOBAL STUDENTS CALL
            …
   }
}
```

Photos

This chapter describes how to:

- Get a student's photo by studentId.

It describes all the available methods, their request/response format and provides input/output examples.

Scope pattern: api_informat_sas_leerlingen.leerlingen.{instituteNo}

GET /students/{studentId}/photo

Gets a student's photo by studentId. The institute number is defined via the mandatory header "InstituteNo".

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
studentId	GUID	required	Limits the output results to a student with the provided studentId. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}` If not provided a bad request with error code "BR002" and message "Invalid student id" will be returned.

Body

This request has an empty body.

Response

The response contains one photo object.

Field	Type	Description
Id	Guid	Unique identifier which identifies the photo.
persoonId	Guid	Unique identifier for the student. Same as the studentId provided via the url. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`
foto	string	Base64-encoded string representation of the photo.

Example request

Request

```
GET https://<endpoint>/<version>/students/c5806777-4bce-4d50-afc5-9cb47f1a5f14/photo
HTTP/1.1
InstituteNo: 999999
Timestamp: 2023-01-10T00:00:00.000Z
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Tue, 16 May 2024 11:36:29 GMT

{
   "photo": {
        "id": "88927dc9-c707-4400-b4eb-0b74bd390578",
        "persoonId": "c5806777-4bce-4d50-afc5-9cb47f1a5f14",
        "foto":
"/9j/4AAQSkZJRgABAQEBLAEsAAD/2wBDAAgGBgcGBQgHBwcJCQgKDBQNDAsLDBkSEw8UHRofHh0aHBwgJC4nICIsIxwcKDcpLDAxND
Q0Hyc5PTgyPC4zNDL/2wBDAQkJCQwLDBgNDRgyIRwhMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyMjIyM
jIyMjL/wAARCAABAAEDASIAAhEBAxEB/8QAHwAAAQUBAQEBAQEAAAAAAAAAAAECAwQFBgcICQoL/8QAtRAAAgEDAwIEAwUFBAQAAAF9
AQIDAAQRBRIhMUEGE1FhByJxFDKBkaEII0KxwRVS0fAkM2JyggkKFhcYGRolJicoKSo0NTY3ODk6Q0RFRkdISUpTVFVWV1hZWmNkZWZ
naGlqc3R1dnd4eXqDhIWGh4iJipKTlJWWl5iZmqKjpKWmp6ipqrKztLW2t7i5usLDxMXGx8jJytLT1NXW19jZ2uHi4+Tl5ufo6erx8v
P09fb3+Pn6/8QAHwEAAwEBAQEBAQEBAQAAAAAAAAECAwQFBgcICQoL/8QAtREAAgECBAQDBAcFBAQAAQJ3AAECAxEEBSExBhJBUQdhc
RMiMoEIFEKRobHBCSMzUvAVYnLRChYkNOEl8RcYGRomJygpKjU2Nzg5OkNERUZHSElKU1RVVldYWVpjZGVmZ2hpanN0dXZ3eHl6goOE
hYaHiImKkpOUlZaXmJmaoqOkpaanqKmqsrO0tba3uLm6wsPExcbHyMnK0tPU1dbX2Nna4uPk5ebn6Onq8vP09fb3+Pn6/9oADAMBAAI
RAxEAPwD3+iigD//2QA="
    }
}
```

School history

This chapter describes how to:

- Get an overview of the school history for all students who have a registration with status “Gerealiseerd” within the provided institute number and school year.
- Get an overview of the school history for a student by the provided studentId.

It describes all the available methods, their request/response format and provides input/output examples.

Scope pattern: api_informat_sas_leerlingen.leerlingen.{instituteNo}

GET /schoolHistory?schoolYear={schoolYear}&certificateSource={certificateSource}

Gets a “school history” overview for all students who have a registrations with status “Gerealiseerd” within the provided institute number and school year. The institute number is defined via the mandatory header “InstituteNo” and the school year must be send via the URL.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
schoolYear	string	required	Limits the output results to registrations within the given schoolyear. **Format:** “2022-23” If not provided a bad request with error code “BR001” and message “Invalid school year” will be returned.
certificateSource	integer	optional	Is an optional parameter, used to filter the obtained certificates by source. **Possible values** are: 0 (Informat) or 1 (Discimus). So, value 0 means that certificates registered in informat will appear in the list. While value 1 means that the certificates read in from Discimus will appear in the list. If not provided via the URL, value 0 will be taken.

Body

This request has an empty body.

Response

The response contains a list of “school history” object.

Field	Type	Description
pPersoon	integer	Unique integer-value for the student within the database.
persoonId	Guid	Unique identifier for the student. **Format:** {42CD5A20-E22F-45D7-973C-FAB8734DEC50}
schooljaar	string	Schoolyear in which the registration falls. **Format:** “2022-23”
instelnr	string	Official institute number of the school **Format:** 6 characters (digits)

school	string	Descriptive name of the school.
vanDatum	date	Date the registration starts **Format**: : yyyy-MM-dd
totDatum	date	Date the registration ends **Format**: : yyyy-MM-dd or null if empty
nrAdmgrp	string	Official education number (Nr administratieve groep). **Format**: `6 characters (digits)`
admgrp	string	Official education name.
studiebewijzen	array	List of certificates linked to the registration.
studiebewijs	string	Name of the obtained certificate. For example, if it's a "Certificate of Professional Qualification" then certificate name can/will be supplemented with the name of the professional qualification. For example: "Bewijs van beroepskwalificatie: Medewerker groen- en tuinbeheer".
uitreiking	date	Issue date of the certificate. **Format**: : yyyy-MM-dd
clausulering	string	When a student has passed with a B certificate, the clauses will be shown for which the student can't choose 1 or more finalities, forms of education and/or fields of study in a subsequent year.

Example request

Request

```
GET https://<endpoint>/<version>/SchoolHistory?schoolYear=2023-24&certificateSource=0
HTTP/1.1
InstituteNo: 999999
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Tue, 02 July 2024 11:36:29 GMT
[
    {
        "pPersoon": 27396,
        "persoonId": "4515e23f-4314-4447-9b49-61403535b75d",
        "schooljaar": "2020-21",
        "instelnr": "023456",
        "school": "TEST Atheneum",
        "vanDatum": "2020-09-01T00:00:00",
        "totDatum": "2021-06-30T00:00:00",
        "nrAdmgrp": "038500",
        "admgrp": "2e jaar kwalificatiefase Medewerker groen- en tuinbeheer duaal",
        "studiebewijzen": [
            {
                "studiebewijs": "ARL Duaal",
                "uitreiking": "2021-06-30T00:00:00",
                "clausulering": null
            },
            {
                "studiebewijs": "Attest verworven bekwaamheden OV3",
                "uitreiking": "2021-06-30T00:00:00",
                "clausulering": null
            },
            {
                "studiebewijs": "Bewijs van beroepskwalificatie: Medewerker groen- en tuinbeheer",
                "uitreiking": "2021-06-30T00:00:00",
                "clausulering": null
            }
        ]
    },
    {
        "pPersoon": 134345,
        "persoonId": "35b0a17e-bcd4-4b20-a473-cd87a8403edd",
        "schooljaar": "2021-22",
        "instelnr": "032145",
        "school": "TEST GO! Technisch Atheneum",
        "vanDatum": "2021-12-21T00:00:00",
        "totDatum": null,
        "nrAdmgrp": "040092",
        "admgrp": "2e leerjaar A Klassieke talen (Grieks en Latijn)",
        "studiebewijzen": [
            {
                "studiebewijs": "Oriënteringsattest B",
                "uitreiking": "2022-06-30T00:00:00",
                "clausulering": "finaliteiten: Doorstroom, Dubbel"
            },
            {
                "studiebewijs": "Getuigschrift 1e graad",
                "uitreiking": "2022-06-30T00:00:00",
                "clausulering": null
            }
        ]
    }
    {…}
]
```

GET /schoolHistory/{studentId}?certificateSource={certificateSource}

Gets a “school history” overview for a student by studentId.

Request

Headers

Besides the mandatory headers, no other custom headers are required.

URL

Field	Type	Required/optional	Description
studentId	GUID	required	Limits the output results to a student with the provided studentId. **Format:** `{42CD5A20-E22F-45D7-973C-FAB8734DEC50}`

Body

This request has an empty body.

Response

The response contains the school history for one student instead of a list of students. This makes the response similar to the call “GET /schoolHistory?schoolYear={schoolYear}&certificateSource={certificateSource}”.

Example request

Request

```
GET https://<endpoint>/<version>/SchoolHistory/c5806777-4bce-4d50-afc5-9cb47f1a5f14&certificateSource=0
HTTP/1.1
InstituteNo: 999999
Authorization: BEARER
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJ1bmlxdWVfbmFtZSI6IkluZm9ybWF0IiwiaXNzIjoiaHR0cDovL2xvY2FsaG9zdC
IsImF1ZCI6ImFlZjAwMjdlNzNmOTU0MzVmNzc4MWEwY2ZhZDU5ZTVkIiwiZXhwIjoxNTc0NTIxODM3LCJuYmYiOjE1NzQzNDkwMzd9.
WOXKKj_a1-BaS3syFf-3k-ZT9JGS1_MLdHo4PqnE8eY
Accept: application/json
Content-Type: application/json
```

Example response

Response

```
HTTP/1.1 200 OK
Content-Type: application/json; charset=utf-8
Access-Control-Allow-Origin: *
Date: Tue, 02 July 2023 09:36:29 GMT
[
    {
       SIMULAR TO THE GLOBAL SCHOOLHISTORY CALL
    },
    {…}
]
```

APPENDIX Error-codes and descriptions

Bad Request errors

Code	Description
BR000	The JSON value for {field} could not be converted to applicable type
BR001A	Last Name is required
BR001B	Last Name doesn't comply with the validationrule: max. length 50
BR002A	First Name is required
BR002B	First Name doesn't comply with the validationrule: max. length 50
BR003	Additional Names doesn't comply with the validationrule: max. length 160
BR004A	Invalid Date Of Birth
BR005A	Country Of Birth Code is required
BR005B	Country Of Birth Code must have a lenght of 5 characters
BR006A	Place Of Birth Code doesn't comply with the validationrule: max. length 10
BR007A	Place Of Birth doesn't comply with the validationrule: max. length 50
BR008A	Nationality Code is required
BR008B	Nationality Code must have a length of 5 characters
BR009	Sex must have a valid value (1 or 2)
BR010A	Invalid Insz
BR011	Eid No doesn't comply with the validationrule: max. length 12
BR012	Mobile Phone doesn't comply with the validationrule: max. length 20
BR013	Invalid Email
BR014	Name Of Doctor doesn't comply with the validationrule: max. length 100
BR015	Phone Of Doctor doesn't comply with the validationrule: max. length 20
BR016	First Language doesn't comply with the validationrule: max. length 100
BR017	Invalid Religion
BR018	Invalid Priority Group
BR019	Invalid Reason For Refusal
BR020A	Max 2 Relationships are allowed
BR021	Relationship: Invalid Type
BR022A	Relationship: Last Name is required
BR022B	Relationship: Last Name doesn't comply with the validationrule: max. length 50
BR023A	Relationship: First Name is required
BR023B	Relationship: First Name doesn't comply with the validationrule: max. length 30
BR024A	Relationship: Street Name is required
BR024B	Relationship: Street Name doesn't comply with the validationrule: max. length 50
BR025A	Relationship: House No is required
BR025B	Relationship: House No doesn't comply with the validationrule: max. length 10
BR026	Relationship: House Bus No doesn't comply with the validationrule: max. length 6

BR027A	Relationship: Postal Code is required
BR027B	Relationship: Postal Code doesn't comply with the validationrule: max. length 8
BR028A	Relationship: City is required
BR028B	Relationship: City doesn't comply with the validationrule: max. length 30
BR029A	Relationship: Country Code is required
BR029B	Relationship: Country Code must have a lenght of 5 characters
BR030	Relationship: Phone doesn't comply with the validationrule: max. length 20
BR031	Relationship: Mobile Phone doesn't comply with the validationrule: max. length 20
BR032	Relationship: Invalid Email
BR043A	Relationship: Invalid Insz
BR033	Invalid Pre Registration Id
BR034A	Schoolyear is required
BR034B	Invalid Schoolyear
BR035A	Institute is required
BR035B	Institute must have a lenght of 6 characters
BR036A	Structure is required
BR036B	Structure must have a lenght of 3 characters
BR037A	Location Id is required
BR037B	Location Id must have a lenght of 3 characters
BR037C	Invalid LocationId {locationId} for this institute
BR038A	Admgrp Id is required
BR038B	Admgrp Id must have a lenght of 6 characters
BR039	Admgrp Detail doesn't comply with the validationrule: max. length 200
BR040A	Pre Registration Date is required
BR040B	Pre Registration Date must be on or before the Start date
BR041A	Start Date is required
BR041B	Start Date must be between beginning and end of defined school year
BR042	Invalid Registration Status

Data Not Found errors

Code	Description
DNF001	No registration found for Pre Registration Id {preRegistrationId}.

Method Not Allowed errors

Code	Description
MNA001	This Pre-Registration is already accepted, and can no longer be changed
MNA002	This Pre-Registration has been refused, and can no longer be changed
MNA003	This Pre-Registration is already accepted, and can no longer be deleted
MNA004	This Pre-Registration has been refused, and can no longer be deleted
MNA005	Changing this Pre-Registration to an update of personal details isn't allowed

API version history

The table below tracks the history of the different versions of the WEB API and their respective release dates. The version number corresponds to the version number used in the URL. The type can be initial / minor / major.

Api Version	Release Date	Type	Changelog
1	June 23, 2020	Initial release	- *Version 1*
1.5	July 1, 2021	Minor	- Request validation: valid JSON type
1.6	October 18, 2022	Minor	- New property “Insz” added at relationship level. When an invalid Insz number is provided, a BR043A error will be returned.
1.7	November 25, 2022	Minor	- Changes made regarding “update personal details”-registration: o preRegistrationDate is only required if admgrpId is not equal to “000000” (no admgrp) o MNA005 validation error added: Changing this Pre-Registration to an “update of personal details” isn't allowed
1.8	January 10, 2023	Major	- The following 3 new calls are implemented: o GET /preregistrations/{preregistrationId}/status o GET /students?schoolYear={schoolYear}&refdate={referenceDate} o GET students/{studentId}?schoolYear={schoolYear}&refdate={referenceDate}
1.9	February 7, 2023	Minor	- New property “assignedViaRegistrationSystem” added to the “/preregistrations/save” call. As optional and with default value false.
1.10	April 25, 2023	Minor	- The following 2 calls are extended with parameter “changedSince”: o GET /registrations?schoolYear={schoolYear}&changedSince={changedSince} o GET /students?schoolYear={schoolYear}&refdate={referenceDate}& changedSince={changedSince}
1.11	June 6, 2023	Minor	- New property “preRegistrationId” added to the response of the “GET /registrations” call.
1.12	June 20, 2023	Minor	- Additional LocationId validation added to the “/preregistrations/save” call. If the LocationId is formatted correctly, a check is done to verify whether this id exists within the students admin module for the provided instituteNo and school year. (“BR037C”). - Validationrule “BR023B” (Relationship: First Name) changed: max. length 50 => max. length 30.
1.13	March 5, 2024	Minor	- Relationship type 0 (leerling) added to the “/preregistrations/save” call. The purpose of this type is to be able to send the address of an adult pupil without creating a relationship.
1.14	May 21, 2024	Minor	- The following new call is implemented: o GET /students/{studentId}/photo
1.15	July 3, 2024	Minor	- The response of the following calls is extended with Bank accounts: o GET /students?schoolYear={schoolYear}&refdate={referenceDate} o GET /students/{studentId}?schoolYear={schoolYear}&refdate={referenceDate} See next page

Api Version	Release Date	Type	Changelog
			- The following new calls are implemented: o GET /schoolHistory?schoolYear={schoolYear}&certificateSource={ certificateSource} o GET /schoolHistory/{studentId}?certificateSource ={certificateSource}
1.16	September 25, 2024	Minor	- Relationship type 7 (pleegvader) and type 8 (pleegmoeder) added to the "/preregistrations/save" call. - The response of the following calls is extended with a studentcard number (leerlingenkaartNummer) and a bike number (fietsnummer); a new property Country Code (landcode) was added to the addresses. o GET /students?schoolYear={schoolYear}&refdate={referenceDate} o GET /students/{studentId}?schoolYear={schoolYear}&refdate={referenceDate}
1.17	November 12, 2024	Minor	- PInschr, inschrKlasId, klasId & pKlas are added to the response of GET /registrations?schoolYear={schoolYear}&changedSince={changedSince} call.
1.18	November 27, 2024	Minor	- PInschrKlas is added to the response of GET /registrations?schoolYear={schoolYear}&changedSince={changedSince} call.
1.19	January 20, 2025	Minor	- In addition to the official/main classes and sub classes, OLOD groups will also be returned as "GroepType" in the response of GET /registrations?schoolYear={schoolYear}&changedSince={changedSince}.

References

- Wikipedia on RESTful services, http://en.wikipedia.org/wiki/Representational_state_transfer
- MSDN, http://msdn.microsoft.com

List of attachments

Attachment 1 – List of result codes:

Result code	Resources	Description
200	All	The request was received and processed correctly. The requested data has been attached.
400	All	The server was unable to parse the request because of a problem with the content of the request, this can be a problem with the headers / URL / body data that was received.
401	All	The request was sent without a valid signature. This could mean that the signature header was not present, that the signature value wasn't present or that the provided signature is not correct.
404	All	The request was received and processed correctly, but there is no data in the response set.
405	All	The request was sent using an invalid HttpMethod. **Example**: Sending a request as **GET** while only **POST** is allowed.
500	All	The request triggered a handled exception on the service. The exception has been logged. Please try again and/or contact Informat about this problem.
999	All	The request triggered an unexpected exception on the service. The exception has been logged. Please try again and/or contact Informat about this problem.

Sample code (.NET)

```
// ClientId & ClientSecret received from Informat
private const string ClientId = "informat_customer_{clientName}";
private const string ClientSecret = "mySecretCode";
// address to retrieve the access token
private const string IdentityServerBaseAddress = "https://www.identityserver.be";
// address leerlingenapi
private const string ApiBaseAddress = "https://leerlingenapi.informatsoftware.be";

private static async Task<string> CallApi() {
   // the scopes for which the api can be called (multiple scopes possible)
    var scopes = new List<string>
    {
        "api_informat_sas_leerlingen.voorinschrijvingen.{123456}"
    };

    // get token from identityserver
    var token = await GetToken(scopes);

    // create HttpClient and use received Bearer token
    using (var apiClient = new HttpClient
    {
        BaseAddress = new Uri(ApiBaseAddress),
        DefaultRequestHeaders = { Authorization = new AuthenticationHeaderValue("Bearer", token) }
    })
    {
        // set InstituteNo in header for which the request is done
        apiClient.DefaultRequestHeaders.Add("InstituteNo", "123456");
        // call leerlingenapi
        var result = await apiClient.GetStringAsync("/{api method name}");
        return result;
    }
}
private static async Task<string> GetToken(IEnumerable<string> scopes) {
    using var tokenClient = new HttpClient { BaseAddress = new Uri(IdentityServerBaseAddress) };
    var response = await tokenClient.PostAsync("/connect/token", new FormUrlEncodedContent(
        new Dictionary<string, string> {
            { "client_id", ClientId },
            { "client_secret", ClientSecret },
            { "grant_type", "client_credentials" },
            { "scope", string.Join(" ", scopes) }
        }));
    response.EnsureSuccessStatusCode();
    var content = await response.Content.ReadAsStringAsync();
    var json = JsonDocument.Parse(content);
    var token = json.RootElement.GetProperty("access_token").GetString();
    return token;
}
```