application-only authentication (oauth2)

this implementation is based on the client credentials grant flow of the oauth 2 specification

issuing application-only requests

step 1: encode consumer key and secret

the steps to encode an application's client id and client secret into a set of credentials to obtain a bearer token are:

1. URL encode the client id and the client secret according to RFC 1738 .
2. concatenate the encoded client id, a colon character ":", and the client secret into a single string
3. base64 encode the string from the previous step

below are example values showing the result of this algorithm. note that the consumer secret used in this page has been disabled and will not work for real requests.

client id	my_client_id
client secret	my_secret
Bearer token credentials	my_client_id:my_secret
Base64 encoded bearer token credentials	`bXlfY2xpZW50X2lkOm15X3NlY3JldA==`

step 2: obtain a bearer token

the value calculated in step 1 must be exchanged for a bearer token by issuing a request to post oauth/token

- the request must be a HTTP POST request.
- the request must include an **Authorization header with the value of Basic <base64 encoded value from step 1>**.
- the request must include a **Content-Type header with the value of application/x-www-form-urlencoded;charset=UTF-8**.
- the body of the request must have **grant_type**=client_credentials.
- (optinal) the body can have **scope**=SCOPE. scopes: read or write.

example request : https://api.idealista.com/oauth/token

```
Authorization: Basic bXlfY2xpZW50X2lkOm15X3NlY3JldA==

Content-Type: application/x-www-form-urlencoded;charset=UTF-8

grant_type=client_credentials&scope=write
```

if the request was formatted correctly, the server will respond with a JSON-encoded payload:

example response:

```
{
    "access_token": "8b9027fa-ce6c-4c16-9c0b-7d2805328139",
    "token_type": "bearer",
    "expires_in": 39235,
    "scope": "read write"
}
```

the value associated with the **access_token** key is the bearer token

note that one bearer token is valid for an application at a time. Issuing another request with the same credentials to/oauth/token will return the same token until it is invalidated

expires_in: time (in seconds) to expiration

scope: scope(s) which this token is valid

step 3: authenticate api requests with the bearer token

the bearer token may be used to issue requests to api endpoints which support application-only auth. to use the bearer token, construct a normal HTTPS request and include an Authorization header with the value of Bearer <base64 bearer token value from step 2>. signing is not required.

example request:

```
Authorization: Bearer 8b9027fa-ce6c-4c16-9c0b-7d2805328139
```

http://api.idealista.com/3.5/es/search?center=40.42938099999995,-3.7097526269835726&country=es&maxItems=50&numPage=1&distance=452&propertyType=bedrooms&operation=rent

common error cases

this section describes some common mistakes involved in the negotiation and use of bearer tokens. be aware that not all possible error responses are covered here - be observant of unhandled error codes and responses!

invalid requests to obtain bearer tokens

attempts to obtain a bearer token with incorrect app credentials will return "HTTP/1.1 401 Unauthorized" response