SCIM architecture

The module is separated into two libraries - API and implementation. API contains useful DTOs, which can be used for client implementation:

```
<dependency>
  <groupId>eu.bcvsolutions.idm</groupId>
  <artifactId>idm-scim-api</artifactId>
  <version>2.2.0</version>
</dependency>
```

Note: The same nexus repository has to be configured and access has to be granted as above.

Security

Two authentication schemes is supported now:

- Basic authentication - specification is available here
- Token authentization - Authentication scheme using the CzechIdM authentication token in `CIDMST` header. Documentation and example is available in swagger `<server>/swagger-ui.html#!/Authentication/loginUsingPOST` (read more below).

Permissions

Module defines new permission group:

- `SCIM\_READ` - read all resources
- `SCIM\_CREATE` - create all resources
- `SCIM\_UPDATE` - update all resources
- `SCIM\_DELETE` - delete all resources

Standard CzechIdM authorization policies are not supported.

SCIM standard resources

Read more about SCIM model, operations and endpoints.

Standard scim schemas and resources rfc7643 are implemented with a few limitations:

User resource unsupported properties, which cannot be saved in CzechIdM:

- `nickName`
- `title`
- `preferredLanguage`

- `locale`
- `timeZone`
- `addresses`
- `ims`
- `photos`
- `roles`
- `entitlements`
- `x509Certificates`
- `name.middleName`
- `emails` - only one email (primary) can be given
- `phoneNumbers` - only one phone number (primary) can be given

These attributes are not implemented on the CzechIdM side by default, so when the client tries to save these attributes, an exception with filled unsupported attribute name will be thrown. Custom module extension can be created, when attributes should be supported and saved in CzechIdM (with `ScimUserService` extension - override `toDto` and `toResource` methods and save attribute e.g. to custom extended attributes or into custom entity).

Group resource property `displayName` is mapped to CzechIdM attributes `code` and `name`. If `displayName` is changed, then both properties are changed (since 1.2.0 version, previous version modified `name` attribute only, `code` was unmodifiable).

Implemented filter and sort properties on standard scim resources:

- `User` - `externalId`, `userName`, `name.familyName`
- `Group` - `externalId`, `displayName`

Filter supports equals (`eq`) operator and `AND` clause only. When other operator or clause is used, then the unsupported operation will be thrown. Pagination `startIndex` and `count` parameter can be used. `startIndex` parameter is the 1-based index of the first query result. Start index has to be the first index on the page (n * count + 1), the exception with code `FIND\_START\_INDEX\_INVALID` is thrown otherwise (CzechIdM can paginate by the whole page only). The filter parameter has to be URL encoded.

Standard CzechIdM filter properties can be used too, this is not in SCIM standard - e.g. `<server>/api/v1/scim/Users?username=testOne` is alias to SCIM standard `<server>/api/v1/scim/Users?filter%3DuserName%20eq%20%22testOne%22`.

If CzechIdM resource implements `Codeable` interface (e.g. User, Group, TreeType), then resource can be addressed additivelly by code in url and used in filter properties. E.q. `<server>/api/v1/scim/Users/testOne` will get user resource. UUID identifier can be used too.

Bulk operation is not supported.

Attribute names and values are case sensitive.

Custom schemas, resources and extensions

Module adds custom schema `urn:ietf:params:scim:schemas:CzechIdM:8.1` with resources:

- `Contract` - ~ `IdmIdentityContract`, implemented filter properties `externalId`, `user`,

`position`, `main`, `workPosition`
- `ContractGuarantee` - ~ `IdmContractGuarantee`, implemented filter properties `externalId`, `guarantee`, `contract`
- `UserGroup` - ~ `IdmIdentityRole`, implemented filter properties `externalId`, `guarantee`, `contract`
- `TreeType` - ~ `IdmTreeType`, implemented filter properties `externalId`, `code`
- `TreeNode` - ~ `IdmTreeNode`, implemented filter properties `externalId`, `code`, `parent`, `treeType`
- `Form` - extended form definitions, attributes and values - just schema without resource
- `FormDefinition` - ~ `IdmFormDefinition`, implemented filter properties `type`, `code`. Readonly form definitions.
- `FormAttribute` - ~ `IdmFormAttribute` - just schema without resource
- `FormValue` - ~ `IdmFormValue` - just schema without resource

and extensions:

- Extended form values - namespace `urn:ietf:params:scim:schemas:CzechIdM:8.1:Form`. Extended form values can be added to standard and custom resources. Resources, which support extended forms can be listed on schemes endpoint.

Available form definitions (and their attributes) for saving extended attribute values can be listed by `FormDefinition` resource type endpoint. Then extended form values can be saved together with the resource (Resource has to implement `FormableResource` interface):

```
{
   "schemas": [
     "urn:ietf:params:scim:schemas:core:2.0:User",
     "urn:ietf:params:scim:schemas:CzechIdM:1.0:Form"
    ],
   "userName": "scimOne",
   "urn:ietf:params:scim:schemas:CzechIdM:1.0:Form": {
   "forms" : [{
        "code": "default",
        "attributes": [{
                "code": "extArtrOne",
                "values": [{
                      "shortTextValue": "test"
                }]
        }
    ]
   }
   ]}
}
```

As you can see, you don't need to know exact form definition and attributes uuid identifiers - code can be given instead, but uuid identifier can be used too (alias). When request with the resource is sent (POST / PUT / PATCH), then uuid identifiers are returned for saved resources. Saved value identifier has to be used when the value has to be updated, otherwise, the value will be recreated (⇒ drop and create).

Swagger

To expose swagger documentation endpoints, application property has to be configured:

```
## Swagger config
# enable swagger endpoint (can be disabled for development etc.)
springfox.documentation.swagger.enabled=true
```

Then swagger documentation will be available at url `<server>/swagger-ui.html`.

This property is configured in the test and production profile by default.