**Collecting Locality Service**

**Service description**

This service is published for general use in Biodiversity Catalogue (<https://www.biodiversitycatalogue.org/services/62>). It was developed in BioVeL project.

The idea of this service is to provide a simple way to access and reuse this existing locality data, which also can have georeference information. This can be used when labels attached to museum collection specimens are being digitised and entered to database. The digitisation process can be made more efficient by automating this data entry as much as possible. If a locality does not exist it can be added. Existing localities can be updated with additional information. Algorithms are used to automatically find the most probable collecting locality by giving collector’s name and additional information e.g. collecting date, country or coordinates. Also data cleaning is done for the results before giving it back to user.

Source code is available in GitHub (<https://github.com/mjtahtin/BioVeL/tree/master/biovel-locality-ws>) and service can installed and used with local database with Darwin Core standard compatible occurrence data.

**Collecting Locality Service API**

| **Resource URL** | **Method** | **Response** | **Description** | **Parameters** |
| --- | --- | --- | --- | --- |
| /locality | GET | Localities list | Lists localities | mandatory recordedBy, all other optional. See below. |
| /locality | POST | UUID | Creates a new collecting locality. |  |
| /locality/{OccurrenceID} | PUT |  | Updates the locality |  |
| /locality/{OccurrenceID} | DELETE |  | Deletes the collecting locality |  |

**The following parameters are for use exclusively with the Collecting locality API described above.**

| **Parameter** | **Description** |
| --- | --- |
| recordedBy | Mandatory parameter. String. |
| continent | Optional String. SQL wildcards accepted |
| country | Optional String. SQL wildcards accepted |
| stateProvince | Optional String. SQL wildcards accepted |
| county | Optional String. SQL wildcards accepted |
| locality | Optional String. SQL wildcards accepted |
| decimalLatitude | Optional String. SQL wildcards accepted |
| decimalLongitude | Optional String. SQL wildcards accepted |
| year | Optional integer. |
| month | Optional integer. |
| day | Optional integer. |
| numRecords | Output parameter for GET. Number of similar records in database. |
| occurrenceID | Unique id for record. Mandatory for POST/PUT/DELETE commands. |

**Examples of HTTP requests**

GET  [http://apps.digitarium.fi/locality?recordedBy=Ecklon%&continent=Africa](%20http://apps.digitarium.fi/locality?recordedBy=Ecklon%25&continent=Africa)

“GET http://apps.digitarium.fi/locality?recordedBy=%weber%&year=1970”

GET result format (JSON)

[{"continent":"EUROPE","country":"France","year":1844,"month":0,"day":0},

{"continent":"EUROPE","locality":"Avignon","country":"France","year":1844,"month":0,"day":0},

{"continent":"EUROPE","locality":"Avignon.","country":"France","year":1844,"month":0,"day":0}]

POST “POST http://apps.digitarium.fi /locality?recordedBy=Korte,%20Kari&country=Finland&occurrenceID=xx112&locality=Espoo&continent=Europe&stateProvince=EK&county=Uusimaa&decimalLatitude=60.40&decimalLongitude=24.40&year=2014&month=1&day=13”

DELETE “ DELETE  [http://apps.digitarium.fi/locality?occurrenceID=id112](%20http://apps.digitarium.fi/locality?occurrenceID=id112)”

PUT “PUT  [http://apps.digitarium.fi /locality?recordedBy=Korte,%20Kari&country=Finland&locality=Espoo](http://localhost:8080/locality?recordedBy=Korte,%20Kari&country=Finland&locality=Espoo)”

**Installation**

1. Install SVN for version control

Download from: http://subversion.apache.org/packages.html

1. Install Maven (Apache) for build environment

Download from: http://maven.apache.org/download.cgi

1. Check out code from GitHub <https://github.com/mjtahtin/BioVeL/tree/master/biovel-locality-ws>
2. Run using Maven

>mvn clean verify jetty:run –Plocal

**Used database and changing it**

Service uses MySQL database which needs to be installed to server. Default database name is “Locality”. Create it in SQL by command “create database Locality”. Used database table is “location”. It is created automatically if it does not exist.

Database’s name is defined in several files (localityTestModules.java, locality-test.properties

localityMyBatisModule.java, persistence/locality.properties, persistence/uio\_assistant.properties, pom.xml) which need to be changed to change database name. Creating database view might be easier.

Used Database’s username and password are defined in pom.xml.

To change database fields create new databaseChangeLog to liquibase directory and name of new log to master.xml. Also change locationMapper.xml and Location.Java (and possibly LocationList.java).