Martin Scherpinski
15.11.2023

Providing Fine Grained Access Control to OGC API Features

Utilizing SOIs to extend Enterprise' newest service type based on user identities

con terra
locate the future

OGC API - Features

" OGC API - Features is a multi-part **standard** that offers the capability to create, modify, and query spatial data on the **Web** and specifies requirements and recommendations for APIs that want to follow a standard way of **sharing feature data**. " (https://ogcapi.ogc.org/features/)

- Previously dubbed as "WFS 3.0"
- No more XML, but JSON
- Endpoints defined by OpenAPI

- **From OGC Web Services to OGC APIs** – Thursday, Noon, Gerhard Trichtl

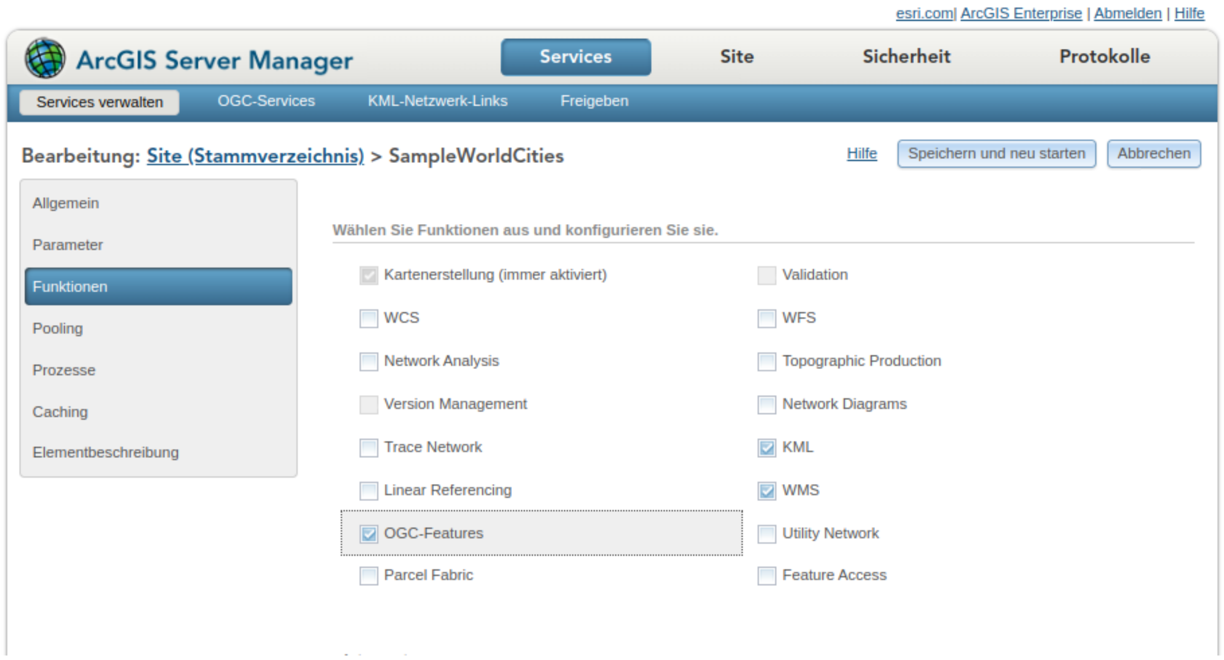
OGC API – Features on ArcGIS Enterprise

- With ArcGIS Enterprise 11: new service capability on MapServer
  - Can just be activated in ArcGIS Server Manager

- Different endpoints are provided

```
/rest/services/SampleWorldCities/OGCFeatureServer
/rest/services/SampleWorldCities/OGCFeatureServer/collections
/rest/services/SampleWorldCities/OGCFeatureServer/collections/0
/rest/services/SampleWorldCities/OGCFeatureServer/collections/0/items
/rest/services/SampleWorldCities/OGCFeatureServer/collections/0/items/1
```

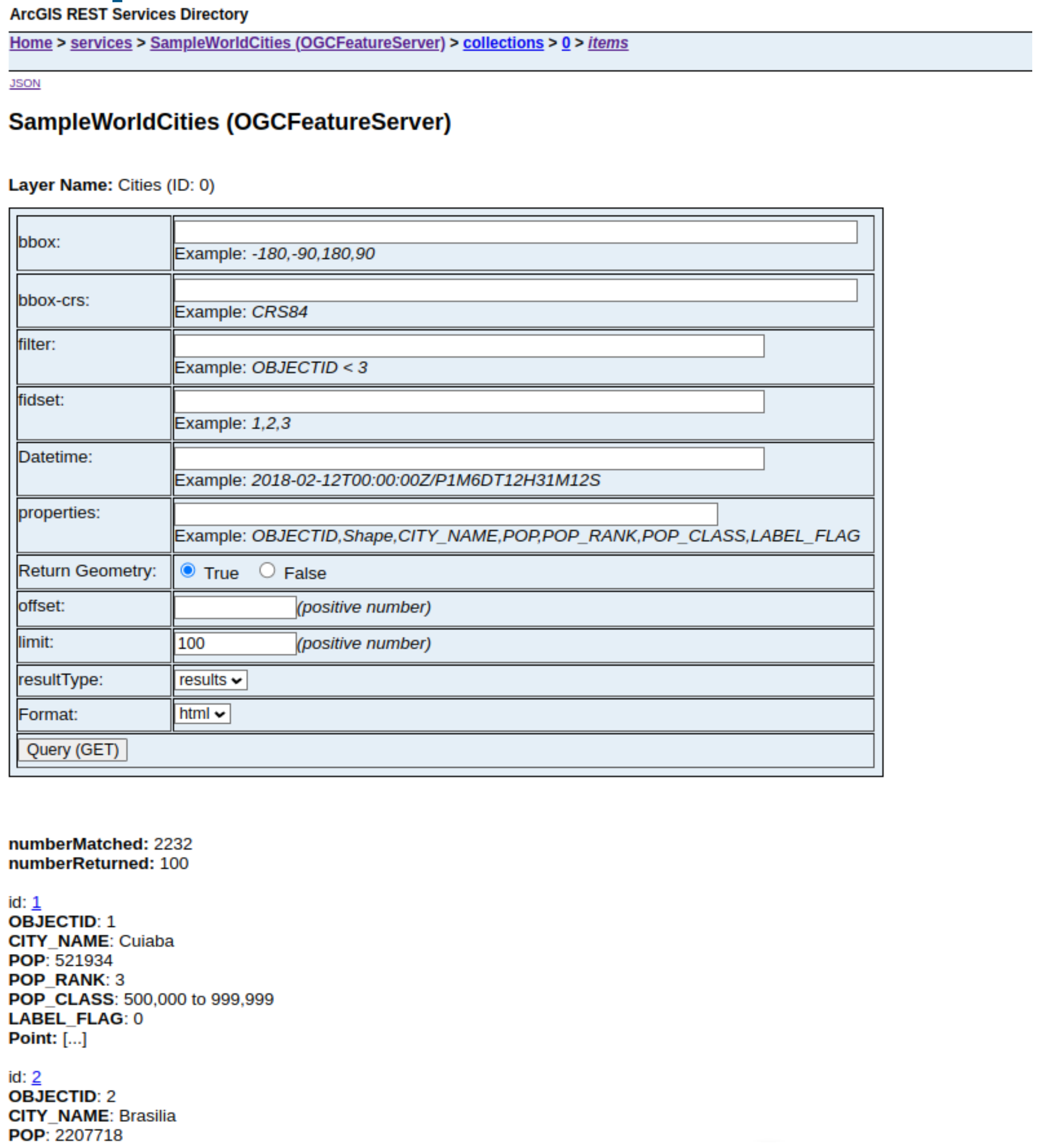
OGC API – Features on ArcGIS Enterprise



Bearbeitung: Site (Stammverzeichnis) > SampleWorldCities

Wählen Sie Funktionen aus und konfigurieren Sie sie.

✓ Kartenerstellung (immer aktiviert)	☐ Validation
☐ WCS	☐ WFS
☐ Network Analysis	☐ Topographic Production
☐ Version Management	☐ Network Diagrams
☐ Trace Network	✓ KML
☐ Linear Referencing	✓ WMS
✓ OGC-Features	☐ Utility Network
☐ Parcel Fabric	☐ Feature Access


ArcGIS REST Services Directory
Home > services > SampleWorldCities (OGCFeatureServer) > collections > 0 > items
JSON
Query (GET)

SampleWorldCities (OGCFeatureServer)

Layer Name: Cities (ID: 0)

bbox:	Example: *-180,-90,180,90*
bbox-crs:	Example: *CRS84*
filter:	Example: *OBJECTID < 3*
fidset:	Example: *1,2,3*
Datetime:	Example: *2018-02-12T00:00:00Z/P1M6DT12H31M12S*
properties:	Example: *OBJECTID,Shape,CITY_NAME,POP,POP_RANK,POP_CLASS,LABEL_FLAG*
Return Geometry:	[✓ True] False
offset:	*(positive number)*
limit:	100 *(positive number)*
resultType:	results
Format:	html

numberMatched: 2232
numberReturned: 100

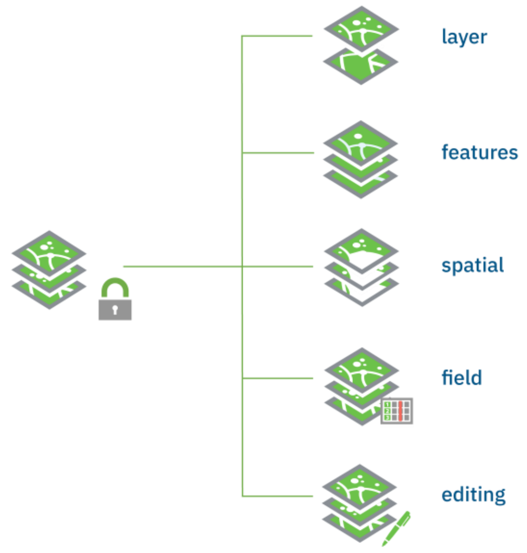
id: 1
OBJECTID: 1
CITY_NAME: Cuiaba
POP: 521934
POP_RANK: 3
POP_CLASS: 500,000 to 999,999
LABEL_FLAG: 0
Point: [...]

id: 2
OBJECTID: 2
CITY_NAME: Brasilia
POP: 2207718

OGC API – Features - Access Control

- No specification on authentication
  - Public services are recommended
  - Not a very common use case …

- Who can access the …
  - Service? ✓
  - Layer?
  - Feature?
  - Features with a specific attribute value?
  - Features with hidden fields?
  - Features within certain area?



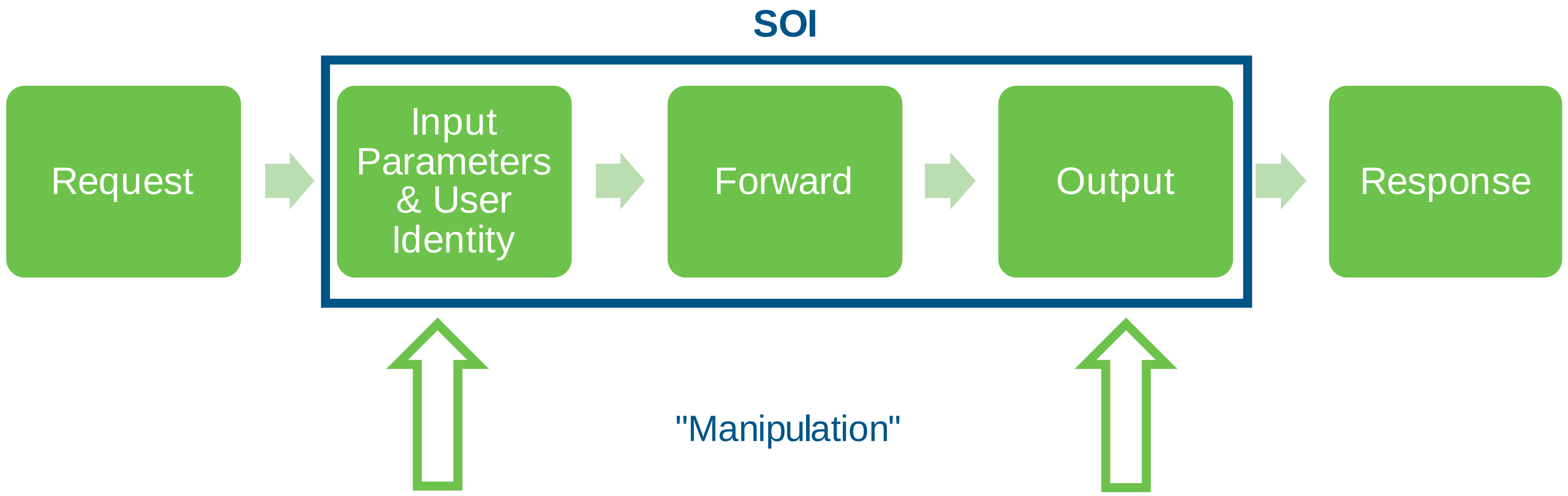
security.manager NEXT capabilites to extend ArcGIS Server security

- Likely to WMS, OGCFeatureServer accepts an **Esri token**!

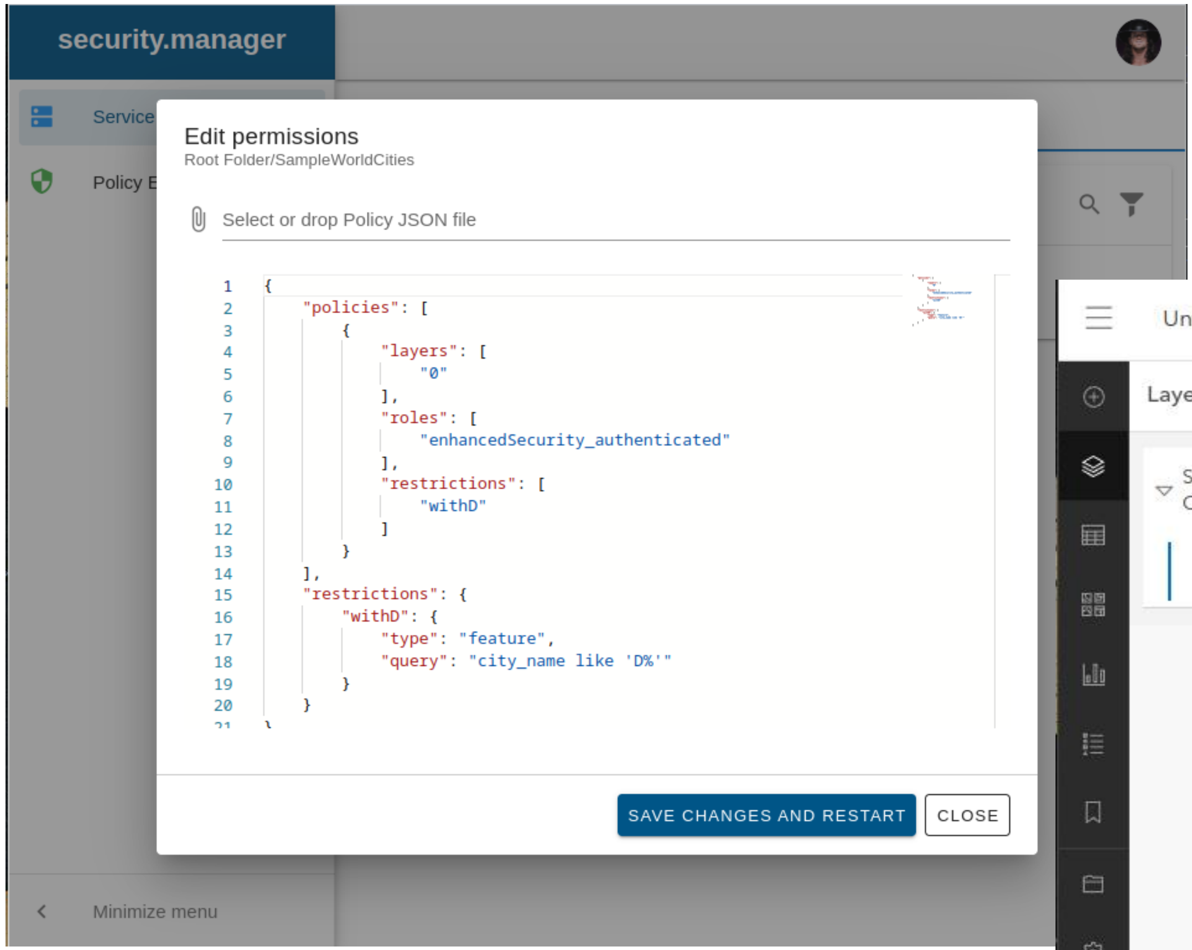
Server Object Interceptors (SOI)

- Tuesday: **Extending ArcGIS Enterprise with the ArcGIS Enterprise SDK -** Cédric Despierre Corporon

- Enterprise SDK based proxy component within the ArcGIS Server runtime
- Intercept every requests to a service



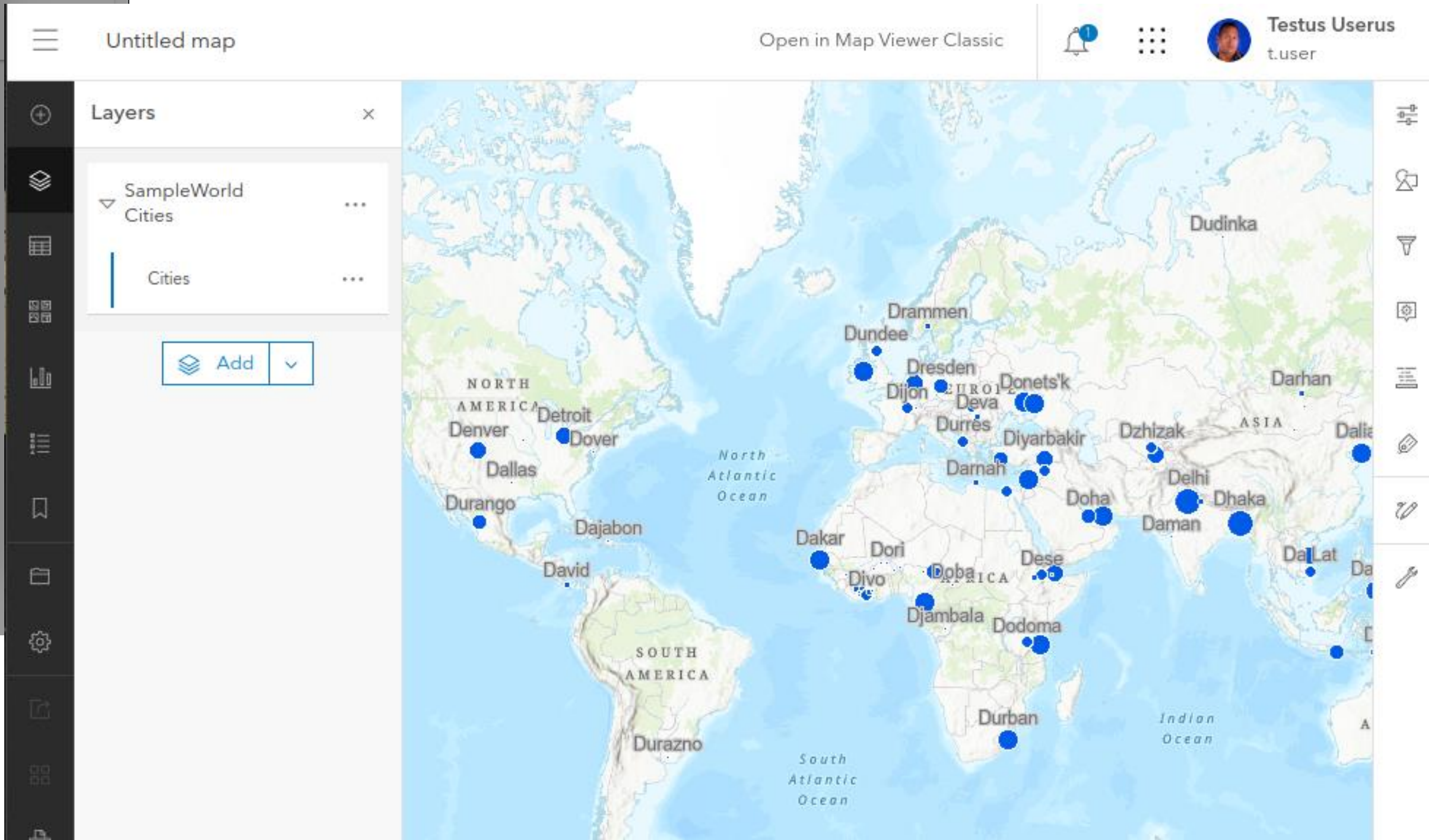
security.manager NEXT – MapServer authorization



Edit permissions
Root Folder/SampleWorldCities

Select or drop Policy JSON file

```
{
    "policies": [
        {
            "layers": [
                "0"
            ],
            "roles": [
                "enhancedSecurity_authenticated"
            ],
            "restrictions": [
                "withD"
            ]
        }
    ],
    "restrictions": {
        "withD": {
            "type": "feature",
            "query": "city_name like 'D%'"
        }
    }
}
```

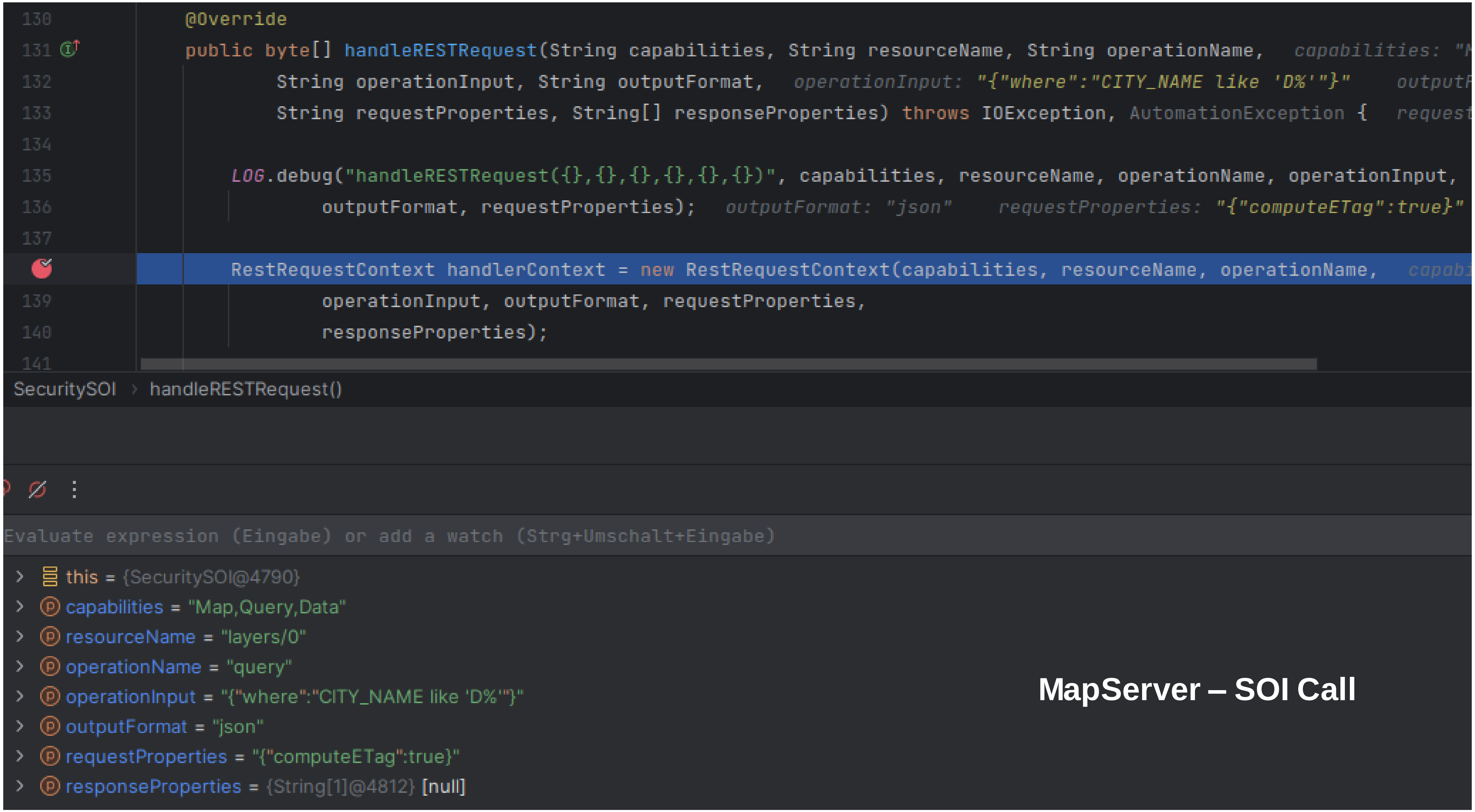


© con terra

OGCFeatureServer - SOI Support

- MapServer / FeatureServer / OGCFeatureServer -> Rest Requests

```
/**
 * Called to handle REST requests.
 * @return the response as byte array.
 */
@Override
public byte[] handleRESTRequest(
        String capabilities, String resourceName, String operationName,
        String operationInput, String outputFormat,
        String requestProperties, String[] responseProperties
) throws IOException, AutomationException {

  ServerUtilities.getServerUserInfo();
}
```



```
    @Override
    public byte[] handleRESTRequest(String capabilities, String resourceName, String operationName,   capabilities: "M
            String operationInput, String outputFormat,   operationInput: "{"where":"CITY_NAME like 'D%'"}"    outputF
            String requestProperties, String[] responseProperties) throws IOException, AutomationException {   request

        LOG.debug("handleRESTRequest({},{},{},{},{},{})", capabilities, resourceName, operationName, operationInput,
                outputFormat, requestProperties);   outputFormat: "json"    requestProperties: "{"computeETag":true}"

        RestRequestContext handlerContext = new RestRequestContext(capabilities, resourceName, operationName,   capabi
                operationInput, outputFormat, requestProperties,
                responseProperties);
```

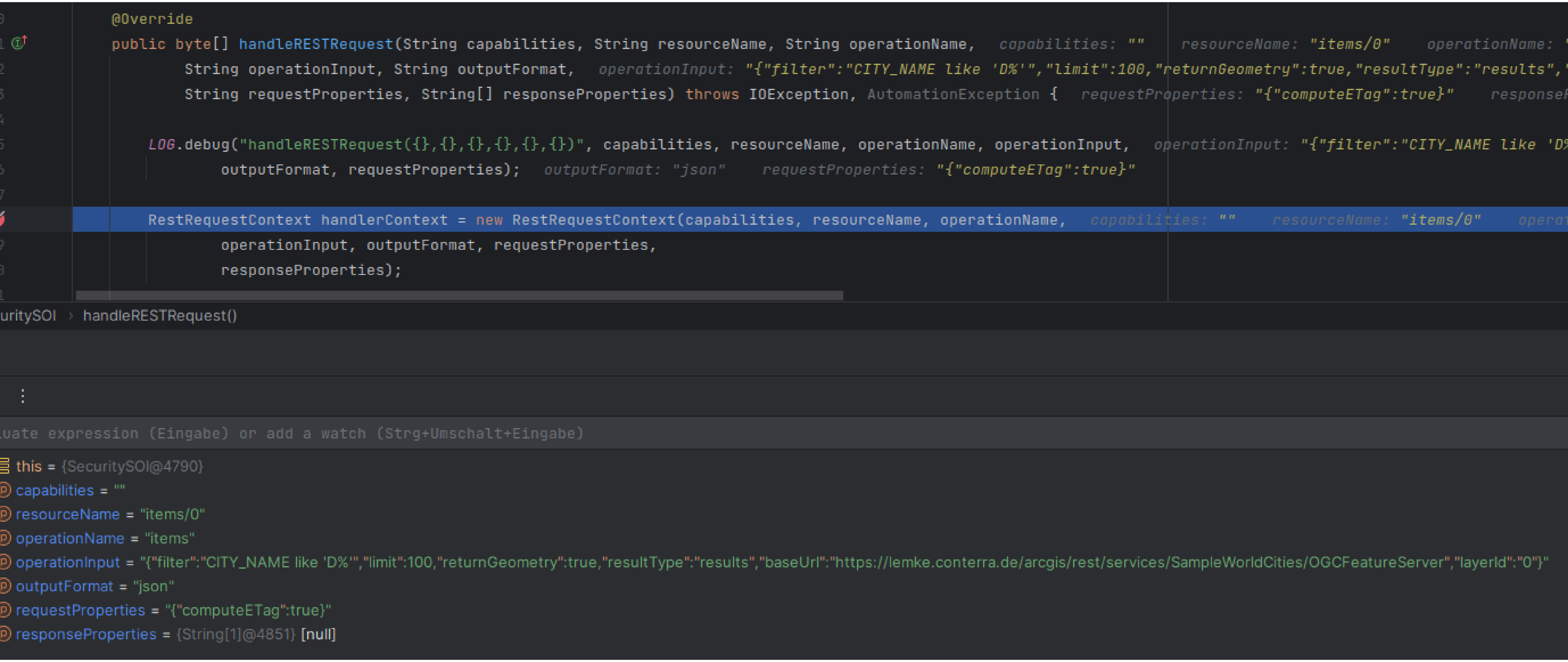
SecuritySOI › handleRESTRequest()

Evaluate expression (Eingabe) or add a watch (Strg+Umschalt+Eingabe)

```
> this = {SecuritySOI@4790}
> capabilities = "Map,Query,Data"
> resourceName = "layers/0"
> operationName = "query"
> operationInput = "{"where":"CITY_NAME like 'D%'"}"
> outputFormat = "json"
> requestProperties = "{"computeETag":true}"
> responseProperties = {String[1]@4812} [null]
```

MapServer – SOI Call

OGCFeatureServer - SOI Call

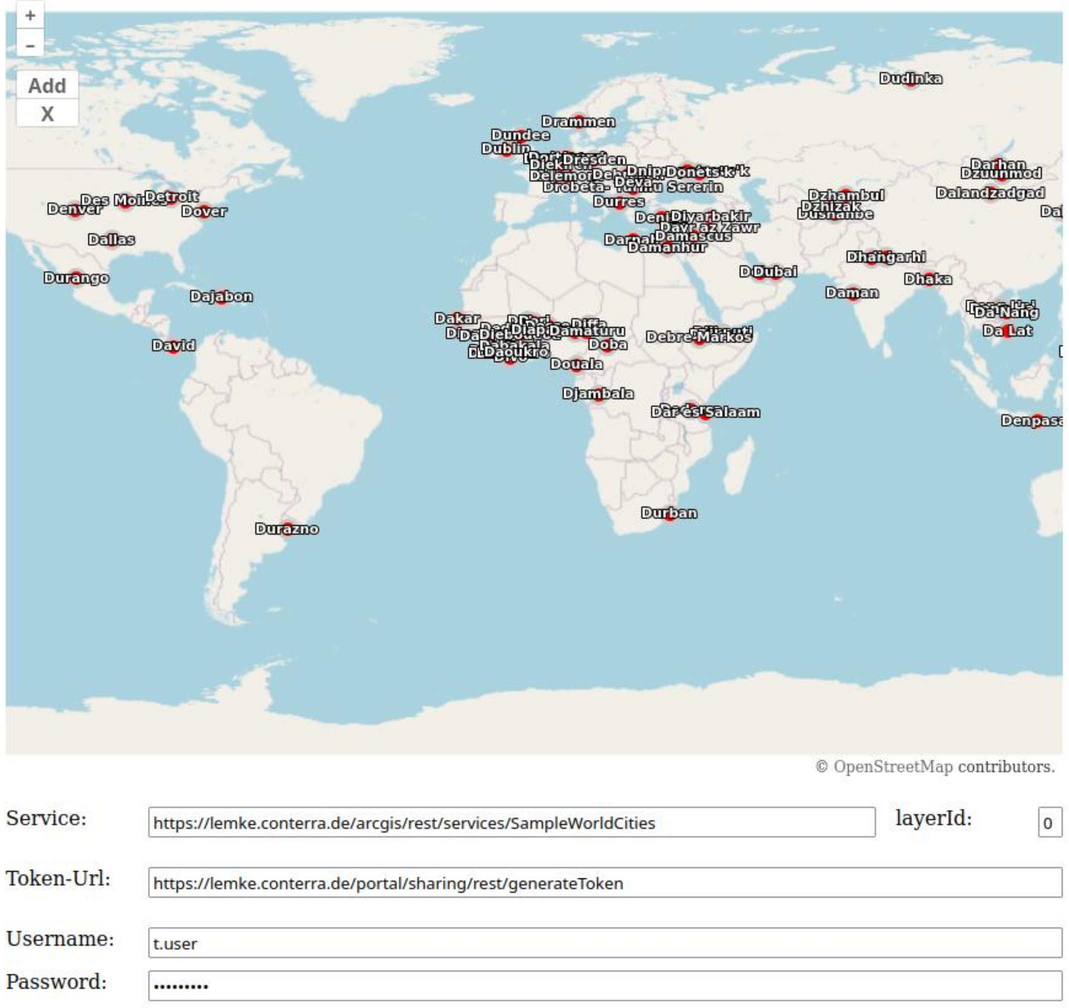


```
    @Override
    public byte[] handleRESTRequest(String capabilities, String resourceName, String operationName,   capabilities: ""    resourceName: "items/0"    operationName: '
            String operationInput, String outputFormat,   operationInput: "{"filter":"CITY_NAME like 'D%'","limit":100,"returnGeometry":true,"resultType":"results",'
            String requestProperties, String[] responseProperties) throws IOException, AutomationException {   requestProperties: "{"computeETag":true}"    responseI

        LOG.debug("handleRESTRequest({},{},{},{},{},{})", capabilities, resourceName, operationName, operationInput,   operationInput: "{"filter":"CITY_NAME like 'D%
                outputFormat, requestProperties);   outputFormat: "json"    requestProperties: "{"computeETag":true}"

        RestRequestContext handlerContext = new RestRequestContext(capabilities, resourceName, operationName,   capabilities: ""    resourceName: "items/0"    operat
                operationInput, outputFormat, requestProperties,
                responseProperties);
```

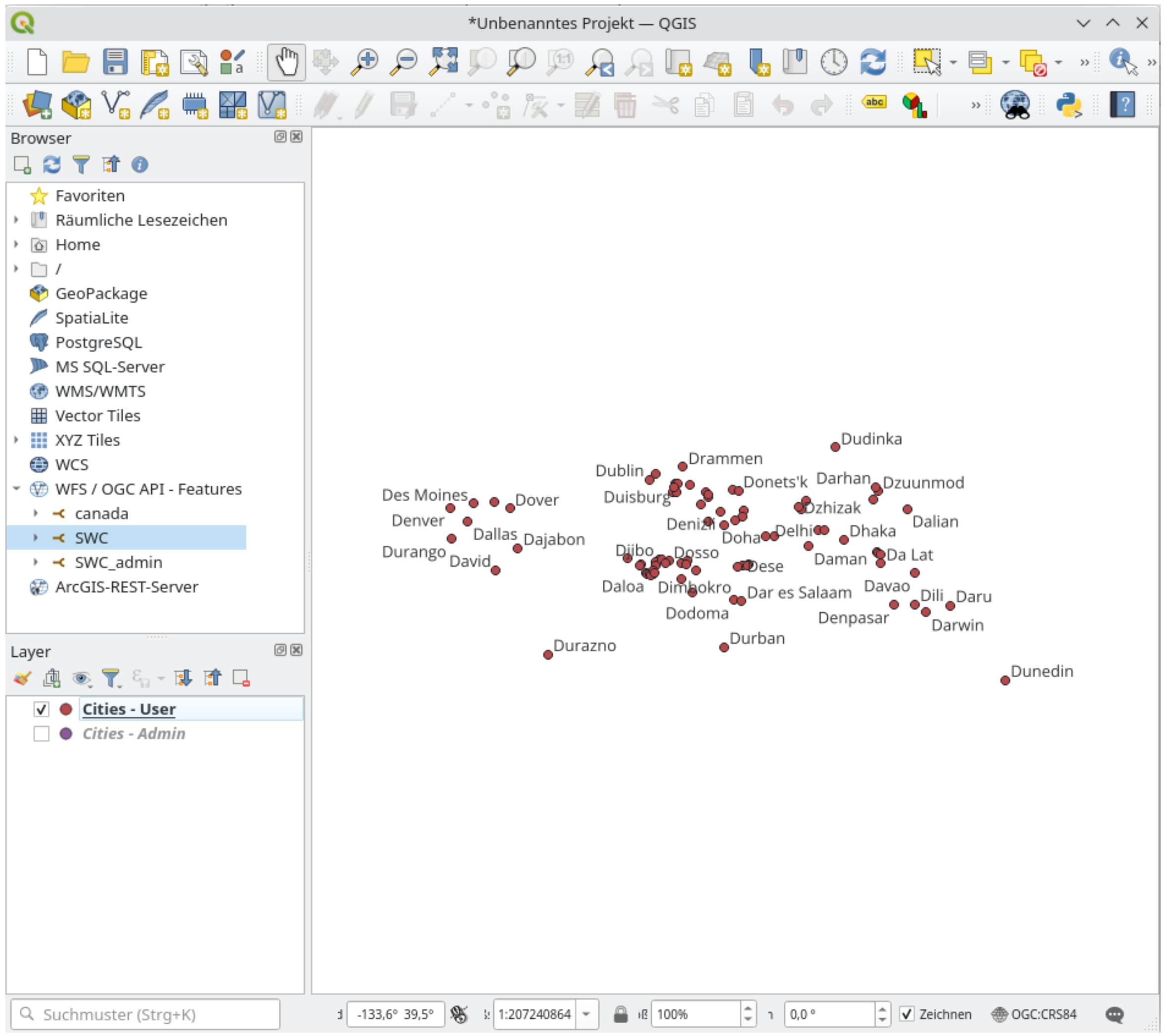
uritySOI › handleRESTRequest()

```
uate expression (Eingabe) or add a watch (Strg+Umschalt+Eingabe)
this = {SecuritySOI@4790}
capabilities = ""
resourceName = "items/0"
operationName = "items"
operationInput = "{"filter":"CITY_NAME like 'D%'","limit":100,"returnGeometry":true,"resultType":"results","baseUrl":"https://lemke.conterra.de/arcgis/rest/services/SampleWorldCities/OGCFeatureServer","layerId":"0"}"
outputFormat = "json"
requestProperties = "{"computeETag":true}"
responseProperties = {String[1]@4851} [null]
```

OGCFeatureServer – OSS Scenarios



Service:	https://lemke.conterra.de/arcgis/rest/services/SampleWorldCities	layerId:	0
Token-Url:	https://lemke.conterra.de/portal/sharing/rest/generateToken		
Username:	t.user		
Password:	••••••••••		



User Identity on OGCFeatureServer - Conclusion

- Easy to set up
  - same data sources as existing MapServer, just activate the service capability
- User Identity is based on an Esri Token
  - Service Access handled by Enterprise
  - OSS Clients must provide a token as well!
- In development, it can be treated likely to MapServer:
  - with SOI's, user specific behaviour can be added
- Caveats:
  - quiet new feature
  - some challenges in the implementation
    - exchange and good cooperation with Esri

Thank You!

Martin Scherpinski
Software Engineer

con terra
Martin-Luther-King-Weg 20
48155 Münster | Germany

T +49 251 59689 353
m.scherpinski@conterra.de
con-terra.com

https://github.com/m-scherpi/ogc_security