

Article

[Dmitry Zasytkin](#) · Oct 7, 2020 8m read

[Open Exchange](#)

Exposing Plain ObjectScript Persistent Classes as FHIR Code Systems and Value Sets

[FHIR Terminology Service](#) specification describes a set of operations on [CodeSystem](#), [ValueSet](#) and [ConceptMap](#) resources. Among those operations, the following four operations appear to be the most widely adopted ones:

CodeSystem

ValueSet

[\\$lookup](#)

[\\$expand](#)

[\\$validate-code](#)

[\\$validate-code](#)

Developing a partial implementation of the specification has been an effective way to explore the [new FHIR framework](#) introduced in IRIS for Health 2020.1. The [implementation](#) includes four operations listed above, and supports [read](#) and [search](#) interactions for [CodeSystem](#) and [ValueSet](#) resources.

It's important to note that the implementation uses plain ObjectScript persistent classes as source terminology tables.

Installation and Testing with a Sample Strategy

The following list outlines installation and basic testing steps:

1. Install IRIS for Health 2020.1 or newer.
2. Set up a new namespace using System Administration > Configuration > System Configuration > Namespaces menu of the Portal, or by running the command `do ##class(HS.HC.Util.Installer).InstallFoundation("<name for the new namespace>")` in HSLIB namespace. Then import classes from [src/cls](#) and [samples/cls](#) folders of [intersystems-ru/fhir-terminology-service](#) GitHub repository.
3. Create a custom FHIR metadata set based on R4 set with additional search parameters defined in [dummy-search-parameters.json](#). This can be done either by using `##class(HS.FHIRServer.ConsoleSetup).Setup()` interactive utility or by running the command:

```
do ##class(HS.FHIRServer.Installer).InstallMetadataSet("<metadataSetKey>", "<metadata set description>", "HL7v40", $lb("<directory with dummy-search-parameters.json>"), 1)
```

- This step is required in order for `$expand` and `$validate-code` operations to support HTTP GET requests.
- Note that FHIR metadata set files packaged with InterSystems IRIS reside in `<installation directory>/dev/fhir/fhir-metadata` directory.

4. Create a new FHIR endpoint based on the new metadata set and on [Sample.iscru.fhir.fts.SimpleStrategy](#) class. Again, this can be achieved either by using the interactive utility or by running the command:

```
do ##class(HS.FHIRServer.Installer).InstallInstance("<web app URI, e.g. /csp/terminology>", "Sample.iscru.fhir.fts.SimpleStrategy", "<metadataSetKey>")
```

5. Allow unauthenticated access to the new endpoint: use the interactive utility or run the following [commands](#):

```
set strategy = ##class(HS.FHIRServer.API.InteractionsStrategy).GetStrategyForEndpoint
```

```
("<web app URI>")  
set config = strategy.GetServiceConfigData()  
set config.DebugMode = 4  
do strategy.SaveServiceConfigData(config)
```

6. Populate [Sample.iscru.fhir.fts.model.CodeTable](#):

```
do ##class(Sample.iscru.fhir.fts.model.CodeTable).Populate(10)
```

7. Import [fhir-terminology-service.postmancollection.json](#) file into Postman, adjust url variable defined within the collection, and test the service against `Sample.iscru.fhir.fts.model.CodeTable` which is a simple persistent class.

Supported FHIR Interactions

Currently the only supported search parameter for both `CodeSystem` and `ValueSet` is `url`.

Both HTTP GET and HTTP POST methods are supported for the four operations listed above.

The table below lists some of the possible HTTP GET requests against [Sample.iscru.fhir.fts.model.CodeTable](#) class.

URI (to be prepended with <code>http://<server>:<port><web app URI></code>)	Description
<code>/metadata</code>	Get endpoint's Capability Statement resource.
<code>/CodeSystem/Sample.iscru.fhir.fts.model.CodeTable</code>	Read <code>CodeSystem</code> resource corresponding to <code>Sample.iscru.fhir.fts.model.CodeTable</code> class.
<code>/ValueSet/Sample.iscru.fhir.fts.model.CodeTable</code>	Read <code>ValueSet</code> resource corresponding to <code>Sample.iscru.fhir.fts.model.CodeTable</code> class.
<code>/CodeSystem?url=urn:CodeSystem:CodeTable</code>	Search <code>CodeSystem</code> resource by url.
<code>/CodeSystem</code>	Output all available <code>CodeSystem</code> resources.
<code>/ValueSet?url=urn:ValueSet:CodeTable</code>	Search <code>ValueSet</code> resource by url.
<code>/ValueSet</code>	Output all available <code>ValueSet</code> resources.
<code>/CodeSystem/\$lookup?system=urn:CodeSystem:CodeTable&code=TEST</code>	Given system and code, get all the details about the concept.
<code>/ValueSet/\$expand?url=urn:ValueSet:CodeTable</code>	Expand the <code>ValueSet</code> .
<code>/CodeSystem/Sample.iscru.fhir.fts.model.CodeTable/\$validate-code?code=TEST</code>	Validate that a code is in the code system.

Creating a Custom Strategy

In order to expose your own persistent classes as FHIR code systems/value sets, you would need to create your custom strategy class by subclassing [iscru.fhir.fts.FTSStrategy](#), and then create FHIR endpoint based on the new custom strategy (see above installation step #4).

One class parameter and at least three methods must be overridden by your strategy class:

- `StrategyKey` class parameter should be assigned some unique value. Name of the current class seems to be a good option.
- `getCodeTablePackage()` class method should return package name for a given code system (or value set) identified by its [canonical URL](#). Typically all terminology classes belong to one package, so this method would usually return one and the same package name regardless of argument values.
- `getCodePropertyName()` and `getDisplayPropertyName()` class methods should return names of class properties that correspond to code and display concept elements. Different classes may have different properties mapped to terminology code/display elements.

Other methods and parameters of [iscru.fhir.fts.FTSStrategy](#) that you might find appropriate to override are as follows:

- `listCodeTableClasses()` class method needs to be overridden in order to support search requests that result in returning all available code systems (or value sets). This method is supposed to return a list of class names of all available terminology classes. [Sample.iscru.fhir.fts.SimpleStrategy](#) contains the following basic implementation of this method:

```
/// Returns a list of all available code table classes.
ClassMethod listCodeTableClasses() As %List
{
    #dim sql As %String = "SELECT name FROM %Dictionary.ClassDefinition WHERE name LI
    KE ' " _ ..#codeTablePACKAGE _ ".%" ORDER BY name"
    #dim resultSet As %SQL.StatementResult = ##class(%SQL.Statement).%ExecDirect(, sq
    l)
    if (resultSet.%SQLCODE '= 0) && (resultSet.%SQLCODE '= 100) $$$ThrowStatus($$$ERR
    OR($$$SQLError, resultSet.%SQLCODE, resultSet.%Message))

    #dim return As %List = ""
    while resultSet.%Next()
    {
        set return = return _ $lb(resultSet.name)
    }

    quit return
}
```

- `isExcludedProperty()` class method has to be overridden if any particular properties of your persistent classes should not show up in CodeSystem resources. By default, this method filters Collection, Identity, Internal, MultiDimensional and Private properties out. Note that object reference and stream properties are currently not supported and ignored by the framework.
- `codeSystemUriPREFIX` and `valueSetUriPREFIX` class parameters and `getCodeSystemForClassname()`, `getValueSetForClassname()`, `determineCodeTableClassname()` and `determineCodeSystemForValueSet()` methods control how class names are mapped to [canonical URLs](#) and vice versa. By default, the following naming scheme is used for canonical URLs:

CodeSystem	ValueSet
urn:CodeSystem:<short class name>	urn:ValueSet:<short class name>

Note that [logical id](#) (aka server id) of a CodeSystem/ValueSet resource equals full name of its corresponding class.

TO DO

Currently lacking is support for versioning of code systems, concept hierarchies and \$subsumes operation, ConceptMap resource and plenty of other stuff. Ideas and pull requests are very welcome!

[#FHIR #InterSystems IRIS for Health](#)
[Check the related application on InterSystems Open Exchange](#)

[value-sets](#)