



National Resource Centre
for EHR Standards
India

Guide to Setup Snowstorm: An open-source FHIR Terminology Server

Created by:
National Resource Centre for EHR Standards (NRCeS),
Centre for Development of Advanced Computing (C-DAC), Pune, India

Published: December 2024

Table of Contents

Introduction	3
Intended Audience	4
Prerequisite	4
Hardware Requirements	4
Software	4
Code system files	4
Setup of Snowstorm	4
Getting Started (Plain Installation)	4
Snowstorm	5
Import CodeSystems and ValueSets into Snowstorm	5
Import SNOMED CT	5
Import SNOMED International Edition – Snapshot version	5
Import SNOMED International Edition – Full version	8
Import National Extension	9
Updating SNOMED International Edition	12
Updating local Edition or National Extension	13
Import LOINC Code System	13
Import ICD-10 Code System	14
Import FHIR (CodeSystem and ValueSets) NPM Package	14
HL7 CodeSystem and ValueSets	14
ABDM CodeSystem and ValueSets	15
Known Issues	15
LOINC Codes Not Found	15
Note	16
Reference	16

Introduction

In the evolving landscape of healthcare data interoperability, the use of standardized medical terminologies is crucial for ensuring accurate and consistent information exchange across different systems. A FHIR Terminology Server is a specialized service that follows HL7 Fast Healthcare Interoperability Resource (FHIR) terminology specifications to manage and provide access to standardized terminologies used in digital health systems. These terminologies include coding systems such as SNOMED CT, Logical Observation Identifiers Names and Codes (LOINC), International Classification of Diseases (ICD), as well as HL7 FHIR value sets, code systems, and other essential resources that support interoperability in healthcare systems.

The open-source FHIR Terminology Servers offer several advantages in terms of cost-effectiveness, flexibility in customization to fit specific organizational needs, community support, and access to the source code ensuring that you can understand and modify the system as needed.

Snowstorm is an open-source terminology server developed by SNOMED International. It has been specifically designed to offer specialized assistance for SNOMED CT. This has further enhanced supporting different code systems and value sets defined in the FHIR Implementation Guide for Ayushman Bharat Digital Mission (ABDM)¹. The server has been constructed on top of Elasticsearch, prioritizing efficiency and scalability for enterprise-level applications.

This guide will walk you through the steps required to set up Snowstorm an open-source Terminology Server from SNOMED International. To support the HL7 FHIR R4 Terminology module, Snowstorm has a dedicated HL7 FHIR API. This API uses the HAPI-FHIR library, which is an exceptional open-source resource. All data resources are stored on Elasticsearch, a powerful and horizontally scalable indexing system known for its speed.

Snowstorm has two APIs:

- **SNOMED CT API**
 - Supports the management of SNOMED CT code systems
 - Supports the SNOMED CT Browser
 - Supports authoring SNOMED CT editions
- **HL7 FHIR API**
 - Implements the Terminology Module
 - Recommended for implementers
 - Supports SNOMED CT, LOINC, ICD-10, ICD-10-CM and other code systems

The guide will provide overall installation and configuration steps focusing on the HL7 FHIR APIs.

¹ FHIR Implementation Guide for ABDM – Code Systems and Value Sets;
<https://nrcea.in/ndhm/fhir/r4/terminology.html>

Intended Audience

- Health IT professionals, vendors, health informaticians, and healthcare providers exploring integration, access, and validation of terminology and code systems
- ABDM implementors for FHIR specification-based integration of code systems, terminologies, value sets, etc.
- Regulatory bodies, and standards organizations, for exploring, validation, and compliance checks, against terminology integration
- Researchers and academicians to understand set-up, requirements, and future enhancements in FHIR terminology servers

Prerequisite

This guide refers to **Snowstorm v10.2.1**.

Hardware Requirements

	Minimum	Recommended
CPU Cores	4	8
Memory	8 GB	12 GB
Storage Type	Hard Disk Drive	Solid State Drive

Software

- Java 17
- Elastic Search Version 8.11
- HAPI FHIR CLI Tool

Code system files

- SNOMED International Edition release files
- NRCeS National Extension release files
- ICD 10 (version 2019)
- LOINC v2.77
- HL7 FHIR R4 Value Sets
- ABDM FHIR CodeSystem and ValueSets Files

Setup of Snowstorm

Getting Started (Plain Installation)

- Download and install Elasticsearch v8.11.1
- Update the Elasticsearch configuration file with the memory options `-Xms4g` and `-Xmx4g`.
- When running Snowstorm on the same machine along with Elasticsearch, Elasticsearch security can be disabled which requires an SSL certificate. To disable Elasticsearch Security, change the setting `xpack.security.enabled` to **false** in the `config/elasticsearch.yml` file (This file is located in the downloaded Elasticsearch Archive).
- Download the [Snowstorm v10.2.1](https://github.com/IHTSDO/snowstorm/releases) release jar file at - <https://github.com/IHTSDO/snowstorm/releases>

Snowstorm

- Start Elasticsearch from Elasticsearch directory
 - For the Archive installer, Run the command `$/bin/elasticsearch`
 - For the system installer, Run the command `systemctl / service`
- On the first run of Snowstorm the SNOMED CT data need to be loaded.

Import CodeSystems and ValueSets into Snowstorm

Import SNOMED CT

Snowstorm has special features to support SNOMED CT terminology such as URI, filters, properties, implicit value sets, and implicit concept maps.

Note: The native Snowstorm API must be used to create code systems and import content for SNOMED CT.

Import SNOMED International Edition – Snapshot version

SNOMED International Edition – The snapshot version can be imported using either the command line or using REST APIs.

1. Via Command line

Run the below command to start the Snowstorm. This will first delete any existing Snowstorm Elasticsearch indices and import SNOMED CT data from the SNOMED International Edition – Snapshot version

(‘**--import**’ flag here tells Snowstorm to choose the snapshot version and not the full version from the zip archive)

```
$ java -Xms2g -Xmx4g -jar snowstorm*.jar --delete-indices  
--import = <Absolute-path-of-SNOMED-CT-RF2-zip>
```

Note: For minimum hardware configuration setup (8 Gb Memory), set **-Xmx2g** as max. heap size instead of **-Xmx4g** in above command (**-Xms** indicate minimum heap memory used by snowstorm and **-Xmx** indicate max heap memory that snowstorm can utilize).

2. Via REST

Run below command to start the Snowstorm application for REST API endpoints

```
$ java -Xms2g -Xmx4g -jar snowstorm*.jar
```

Follow below steps to start import process:

1. Open <http://localhost:8080/> in the browser. This will redirect to swagger API documentation page and list of available APIs will be shown as follows.

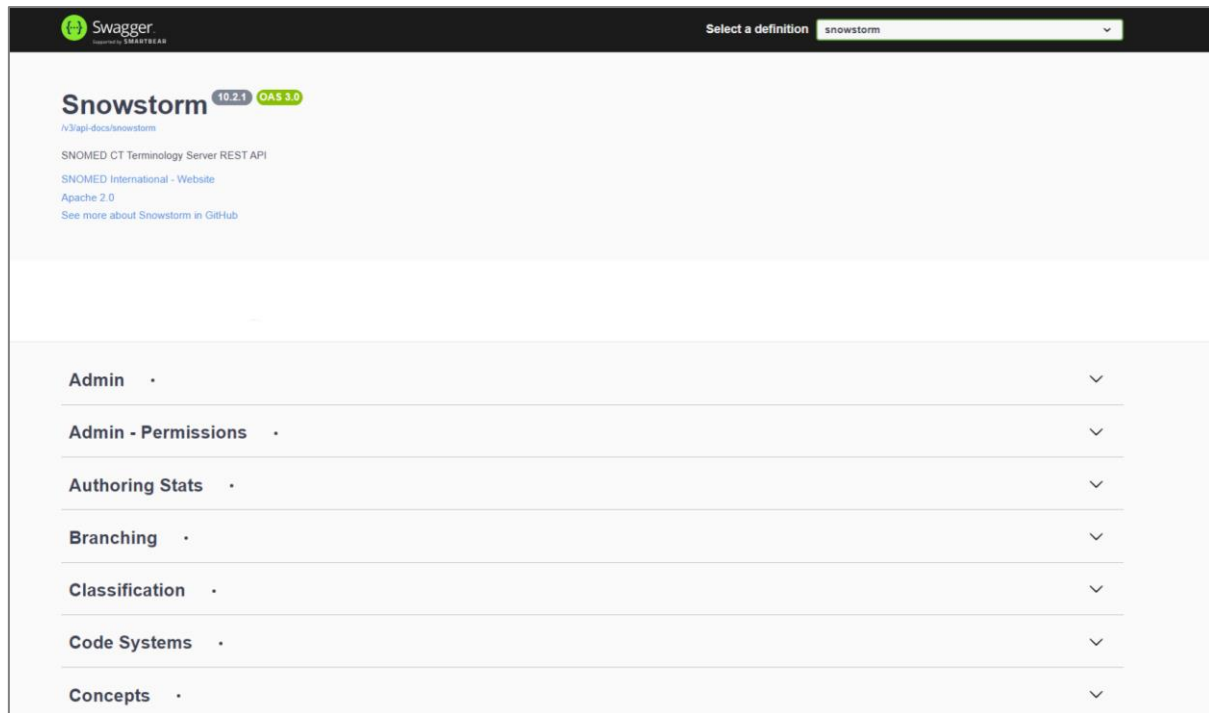


Figure 1: Swagger API documentation Page

- From List of APIs, '**Import**' section contains import related APIs. From collection of import APIs, locate **/import** with **POST** method to create new import job.

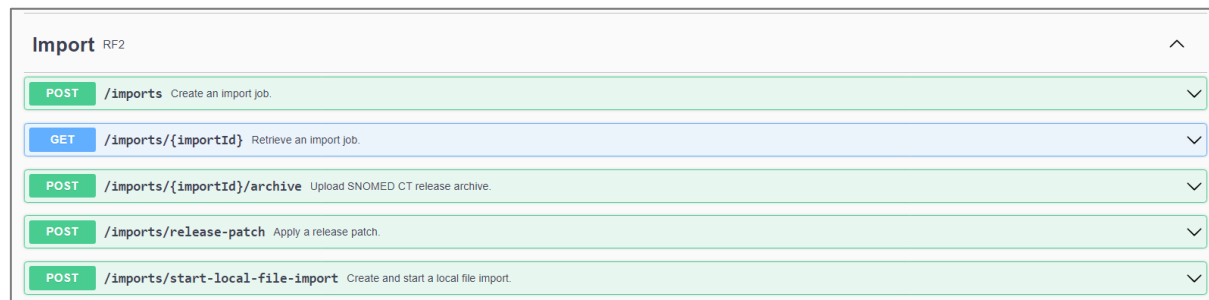


Figure 2: Collection of Import related APIs

- Open **/imports** API and add following details for creation of new import job. Here, type **SNAPSHOT** is explicitly passed to indicate snapshot type import.

```
{
  "branchPath": "MAIN",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

POST /imports Create an import job.

Creates an import job ready for an archive to be uploaded. The 'internalRelease' flag is optional, it can be used to hide a version from the code system versions listing and prevent it being chosen as the code system 'latestRelease'. The 'location' response header contains the URL, including the identifier, of the new resource. Use the upload archive function next. An optional list of module IDs can be provided like ["731000124108", "900000000000012004"] to import only those modules. Leave empty or omit argument for all modules.

Parameters Cancel Reset

No parameters

Request body required application/json

```
{
  "branchPath": "MAIN",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

Execute

Figure 3: Create an Import Job

After entering details, click on '**Execute**' and note the id of the import from response received. This ID is needed for the next step (it will look something like - **d0b30d96-3714-443e-99a5-2f282b1f1b0**).

- To upload the RF2 file, **/imports/archive** endpoint will be used from collection. Follow steps below to upload file:

POST /imports/{importId}/archive Upload SNOMED CT release archive.

Uploads a SNOMED CT RF2 release archive for an import job. The import job must already exist and have a status of WAITING_FOR_FILE. PLEASE NOTE this is an asynchronous call, this function starts the import but does not wait for it to complete. Retrieve the import to check the status until it is COMPLETED or FAILED.

Parameters Cancel Reset

Name	Description
importId required string (path)	d0b30d96-3714-443e-99a5-2f282b1f1b0

Request body multipart/form-data

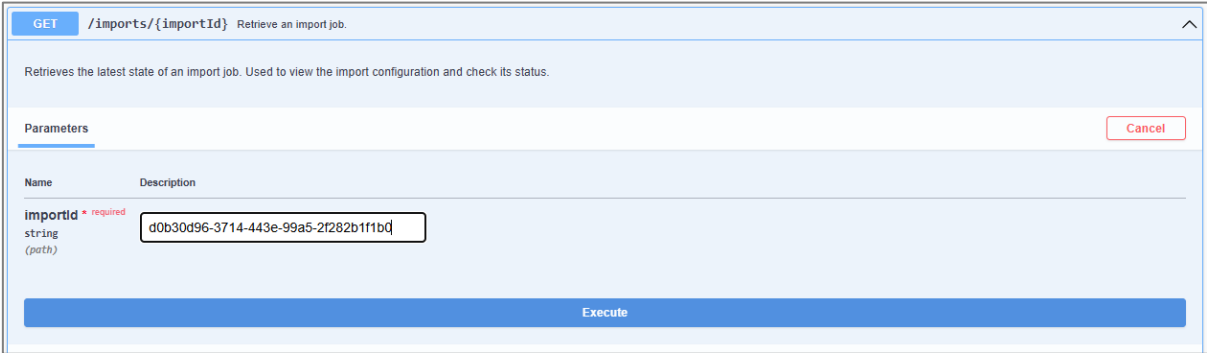
file required
string(binary) Choose File No file chosen

Execute

Figure 4 : Upload Snomed CT Release Archive

- Enter import id
- Select path for the RF2 archive
- Click on Execute

- To check File upload status, ***/imports/{importId}*** with **GET** method is used. This step also requires importId. Enter ImportId and click Execute to get Current Status of upload Process as response. Complete Upload process can take between 20-60 minutes depending on the performance of machine used.



GET ***/imports/{importId}*** Retrieve an import job.

Retrieves the latest state of an import job. Used to view the import configuration and check its status.

Parameters

Name	Description
importId * required string (path)	d0b30d96-3714-443e-99a5-2f282b1f1bd

Execute

Figure 5: Retrieve an import Job

Import SNOMED International Edition – Full version

It is possible to load the RF2 **Full** files which gives you access to all previous releases of SNOMED CT in addition to the current content. However, this will take longer, but will not have an impact to the performance.

When uploading RF2 file using command line or by using REST APIs, make following changes to steps used for Snapshot upload.

Via Command Line

Replace '**-import**' flag with '**-import-full**' flag. Updated Command is as below.

```
$ java -Xms2g -Xmx4g -jar snowstorm*.jar --delete-indices
--import-full=<Absolute-path-of-SNOMED-CT-RF2-zip>
```

Remaining steps will be same for snapshot and Full RF2 upload.

Via REST APIs

In this method, change upload '**type**' from '**SNAPSHOT**' to '**FULL**' in request body of ***/import*** API when creating import job. Updated request body will look like below.

```
{
  "branchPath": "MAIN",
  "createCodeSystemVersion": true,
  "type": "FULL"
}
```

Other steps for uploading RF2 file will remain same as Snapshot upload.

Import National Extension

To load an extension, it is necessary to first load an edition. If an edition has not been loaded, upload Edition first and then proceed with National Extension Import.

To add each extension, a corresponding CodeSystem must be created.

Follow this example to upload a Snomed CT extension. (replace Extension name with name of specific extension you are uploading).

1. Create Code System

On the Swagger interface, locate the "**create a code system**" POST method under the code systems endpoint (refer figure 6 below)

(http://localhost:8080/swagger-ui.html#!/Code_Systems/createCodeSystemUsingPOST).

Use the provided request details below to create the branch:

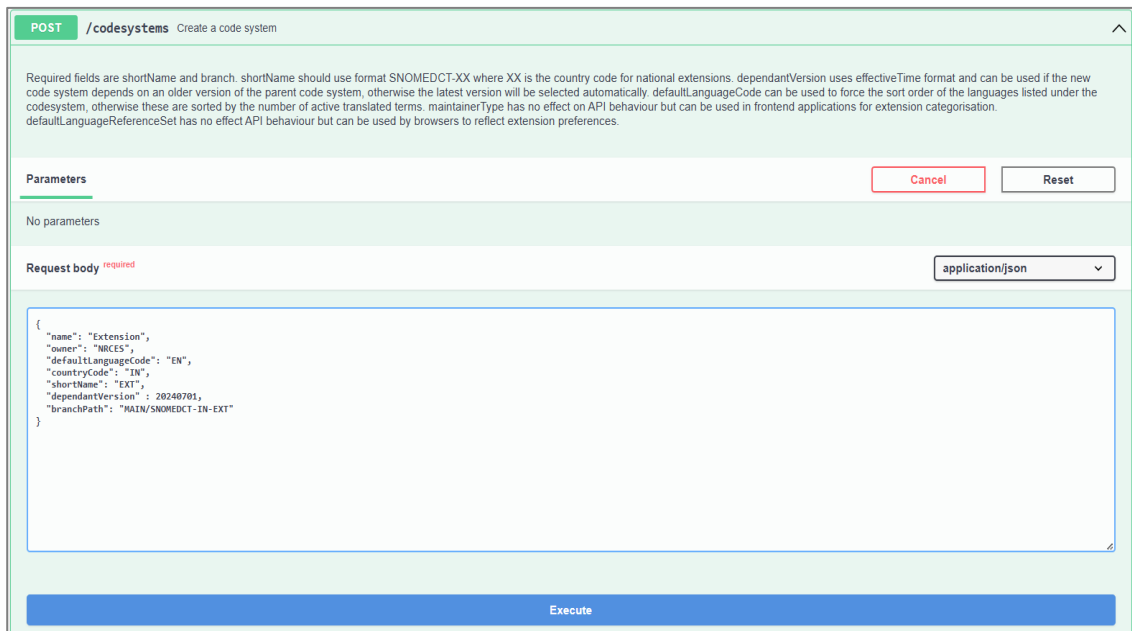
```
{
  "name": "Extension",
  "owner": "NRCES",
  "defaultLanguageCode": "EN",
  "countryCode": "IN",
  "shortName": "EXT",
  "dependantVersion": 20240701,
  "branchPath": "MAIN/SNOMEDCT-IN-EXT"
}
```

The dependantVersion is the version of the Edition which the extension being imported is dependent on. For example, an extension with an effective date of 20240430 might be dependent on the International Edition 20240131. This field is used when creating the extension branch so that the new branch can see content from the desired release in the parent branch. This **dependantVersion** will be changed when upgrading the extension

There are many optional fields available for that request that can be used to provide additional information about the code system. Below code block lists available optional fields.

```
{
  "name": "string",
  "owner": "string",
  "defaultLanguageCode": "string",
  "dependantVersionEffectiveTime": 0,
  "dependantVersion": number,
  "countryCode": "string",
  "maintainerType": "string",
  "shortName": "string",
  "defaultLanguageReferenceSets": [
    "string"
  ],
  "branchPath": "string"
}
```

These are used by the [SNOMED Browser project](#).



POST /codesystems Create a code system

Required fields are shortName and branch. shortName should use format SNOMEDCT-XX where XX is the country code for national extensions. dependantVersion uses effectiveTime format and can be used if the new code system depends on an older version of the parent code system, otherwise the latest version will be selected automatically. defaultLanguageCode can be used to force the sort order of the languages listed under the codesystem, otherwise these are sorted by the number of active translated terms. maintainerType has no effect on API behaviour but can be used in frontend applications for extension categorisation. defaultLanguageReferenceSet has no effect API behaviour but can be used by browsers to reflect extension preferences.

Parameters Cancel Reset

No parameters

Request body required application/json

```
{
  "name": "Extension",
  "owner": "NRCES",
  "defaultLanguageCode": "EN",
  "countryCode": "IN",
  "shortName": "EXT",
  "dependantVersion": 20240701,
  "branchPath": "MAIN/SNOMEDCT-IN-EXT"
}
```

Execute

Figure 6: Create a Code System for Extension Upload

Click “**EXECUTE**” to create a code system.

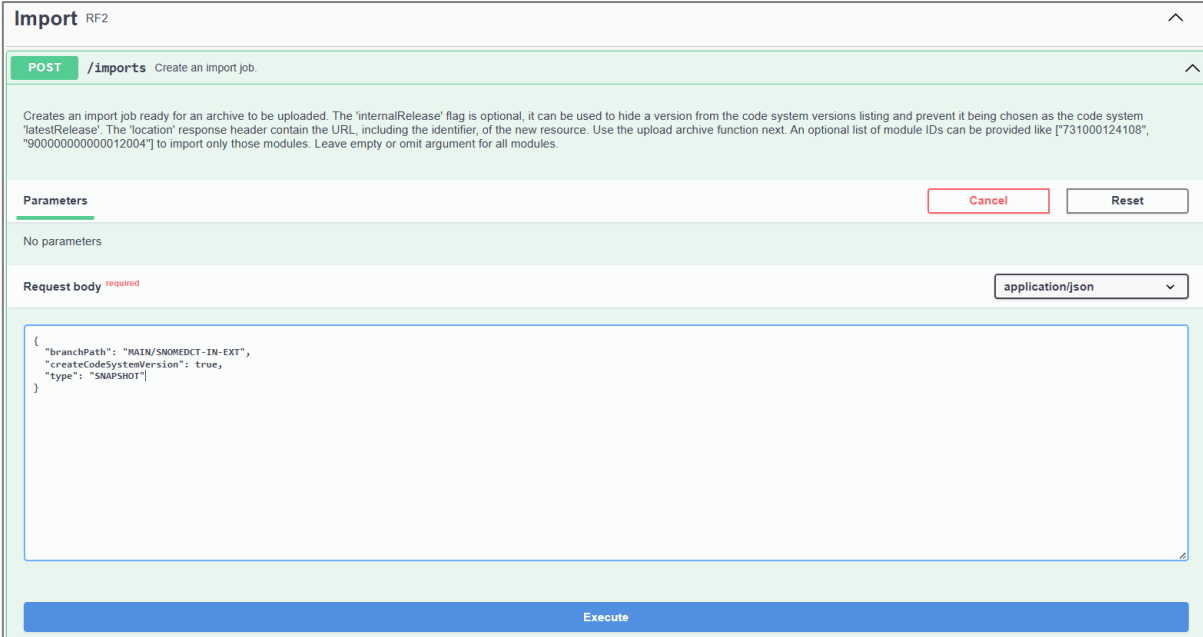
2. Create import Job

The Extension can now be imported.

- Look for the **/imports** Endpoint with POST method to Create new import job. (figure 7)

```
{
  "branchPath": "MAIN/SNOMEDCT-IN-EXT",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

- Create a new import using following request body:



Import RF2

POST /imports Create an import job.

Creates an import job ready for an archive to be uploaded. The 'internalRelease' flag is optional, it can be used to hide a version from the code system versions listing and prevent it being chosen as the code system 'latestRelease'. The 'location' response header contain the URL, including the identifier, of the new resource. Use the upload archive function next. An optional list of module IDs can be provided like ["731000124108", "900000000000012004"] to import only those modules. Leave empty or omit argument for all modules.

Parameters Cancel Reset

No parameters

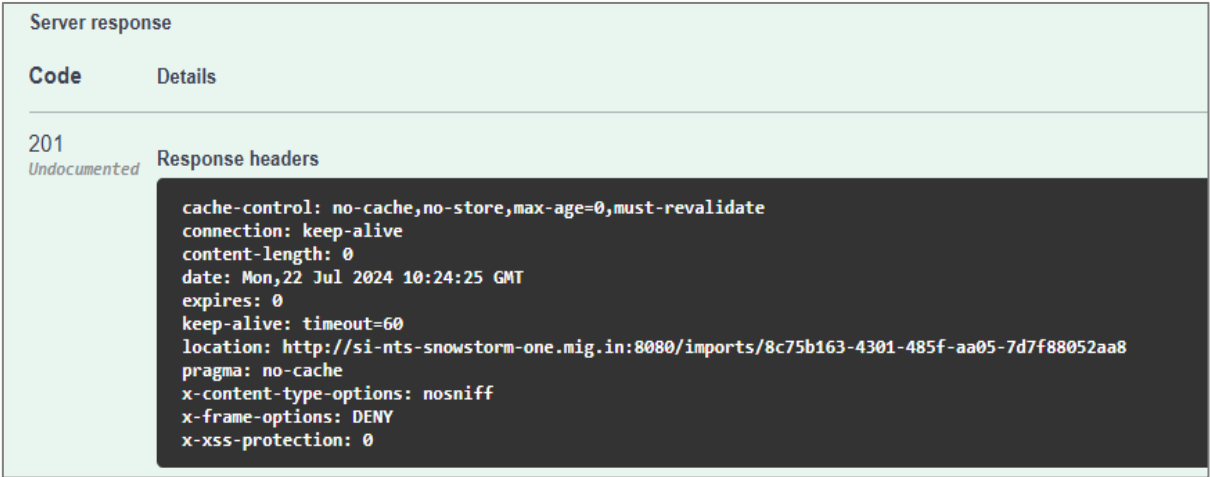
Request body *required* application/json

```
{
  "branchPath": "MAIN/SNOMEDCT-IN-EXT",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

Execute

Figure 7: Create an Import Job for Extension Upload

- Click **Execute** to create new import job.



Server response

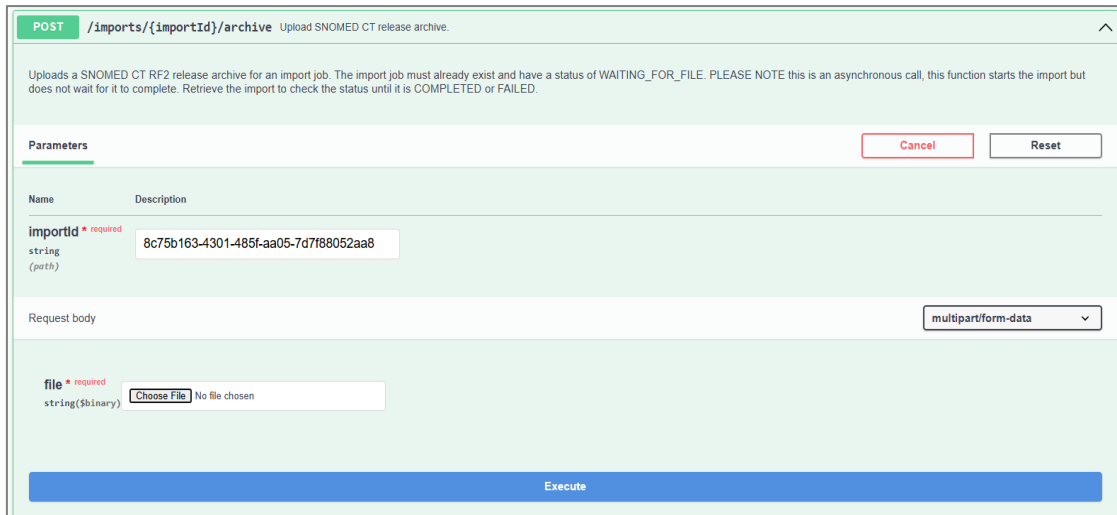
Code	Details
201 <i>Undocumented</i>	<p>Response headers</p> <pre>cache-control: no-cache,no-store,max-age=0,must-revalidate connection: keep-alive content-length: 0 date: Mon,22 Jul 2024 10:24:25 GMT expires: 0 keep-alive: timeout=60 location: http://si-nts-snowstorm-one.mig.in:8080/imports/8c75b163-4301-485f-aa05-7d7f88052aa8 pragma: no-cache x-content-type-options: nosniff x-frame-options: DENY x-xss-protection: 0</pre>

Figure 8: Response Header for Create Import Job Request

- Note the **importId** of the import as it is needed for the next step (look for a UUID like 8c75b163-4301-485f-aa05-7d7f88052aa8 in the Location response header as shown in figure 8).

3. Upload RF2 Release File

RF2 file can be uploaded through Swagger using the `/imports/{importId}/archive` endpoint.



POST `/imports/{importId}/archive` Upload SNOMED CT release archive.

Uploads a SNOMED CT RF2 release archive for an import job. The import job must already exist and have a status of WAITING_FOR_FILE. PLEASE NOTE this is an asynchronous call, this function starts the import but does not wait for it to complete. Retrieve the import to check the status until it is COMPLETED or FAILED.

Parameters

Name	Description
importId * required string (path)	8c75b163-4301-485f-aa05-7d7f88052aa8

Request body

multipart/form-data

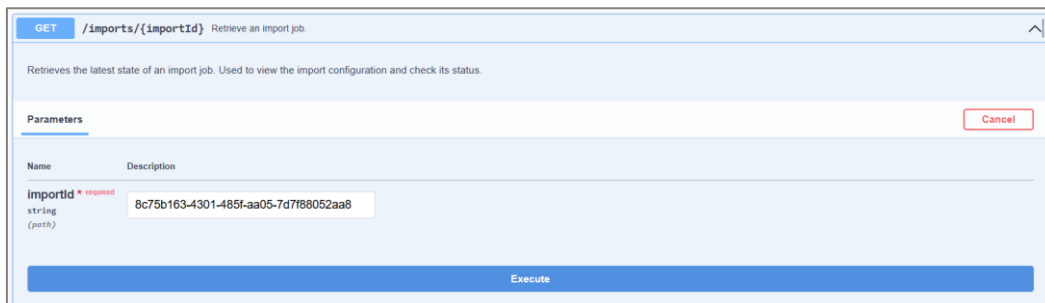
file * required
string(binary)

Choose File No file chosen

Execute

Figure 9: Upload Snomed CT Extension Archive

- Put previous noted **importId** into the parameter and attach a RF2 file of Extension to upload.
 - Click **Execute** to start upload.
4. To get status of upload process, use `/imports/{importId}` API (figure 10). Complete Upload will take around 5-10 minutes depending on size of extension RF2 File and performance of machine used.



GET `/imports/{importId}` Retrieve an import job.

Retrieves the latest state of an import job. Used to view the import configuration and check its status.

Parameters

Name	Description
importId * required string (path)	8c75b163-4301-485f-aa05-7d7f88052aa8

Execute

Figure 10: Retrieve Status of Import Job

Updating SNOMED International Edition

Since January 2022, a new SNOMED CT International Edition release is published every month. **Please do not import delta archives created with the Delta Generator Tool into Snowstorm because this will make the content inconsistent.** New releases should be imported onto the MAIN branch using the **SNAPSHOT** import type.

Follow Steps Below:

- Create an import job as Above
- click on '**Execute**'

3. note the id of the import as before
4. Upload the New International release file

Use following request body while creating import job.

```
{
  "branchPath": "MAIN",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

Updating local Edition or National Extension

The edition or extension upgrade is an import job again. The **SNAPSHOT** import type can always be used for upgrades onto the relevant code system branch. If the extension archive contains delta RF2 files then the **DELTA** import type could also be used for a slightly faster import.

First, import job must be created as before. Following request body can be used. (Figure 7)

```
{
  "branchPath": "MAIN/SNOMEDCT-IN-EXT",
  "createCodeSystemVersion": true,
  "type": "SNAPSHOT"
}
```

- Click '**Execute**' to create import job.
- Note that the **importId** from response.
- Start RF2 upload process using **/imports/{importId}/archive** (figure 9).
- To get status of RF2 upload **/imports/{importId}** endpoint can be used (figure 10).

Import LOINC Code System

The LOINC code system is supported by Snowstorm Terminology Server.

To Upload LOINC package, following steps can be followed:

- Run Following command to import Loinc using hapi fhir cli tool (Linux and Mac users can use '**hapi-fhir-cli**' script from package. Windows users can use '**hapi-fhir-cli.bat**' script file).

```
$. /hapi-fhir-cli upload-terminology -d Loinc_2.77.zip -v r4 -t
http://localhost:8080/fhir -u http://loinc.org
```

hapi-fhir-cli is the HAPI FHIR Command Line tool. It features several HAPI's built-in features as easy to use command line options

- Upload will take about 5-6 min for minimal hardware configuration setup. (wait till upload completes and prompt returns)

Note:

1. The old file naming before LOINC v2.73 is not working with snowstorm 10.2.1, use Version **2.73 onwards**.
2. While uploading, if you face error '**Limit of total fields [1000] has been exceeded**' then increase total field limit to 2000 using following command. (Make sure **curl** utility is installed if using windows. standard Linux and mac installation come with curl installed.)

```
$ curl -XPUT http://localhost:9200/fhir-concept/_settings -H 'Content-Type:application/json' -d '{ "index.mapping.total_fields.limit": 2000 }'
```

Import ICD-10 Code System

The international version of ICD-10 is supported. An ICD-10 package can be imported using the **HAPI-FHIR CLI tool** with the following command:

```
$ ./hapi-fhir-cli upload-terminology -d icd10ML2019ens.zip -v r4 -t http://localhost:8080/fhir -u http://hl7.org/fhir/sid/icd-10
```

- Upload will take about 1 min for minimal hardware configuration setup. (wait till upload completes and prompt returns)

Import FHIR (CodeSystem and ValueSets) NPM Package

HL7 CodeSystem and ValueSets

To load CodeSystems and ValueSets from a HL7 NPM FHIR package first download the package from a package registry. For example:

```
npm --registry https://packages.simplifier.net pack hl7.terminology.r4@3.1.0
```

Then upload the package to Snowstorm including a list of resource URLs to load:

```
curl --form file=@hl7.terminology.r4-3.1.0.tgz
--form resourceUrls="http://terminology.hl7.org/CodeSystem/v3-ActPriority"
--form resourceUrls="http://terminology.hl7.org/CodeSystem/appointment-cancellation-reason" http://localhost:8080/fhir-admin/load-package
```

To load all resources within a package, use **--form resourceUrls=" "*"** as shown in below command.

```
$ curl --form file=@hl7.terminology.r4-3.1.0.tgz --form resourceUrls="*"
http://localhost:8080/fhir-admin/load-package
```

Notes on import behaviour:

- Existing resources will be replaced if any imported resource has the same URL and version.
- The package index file is used as the and version of CodeSystems imported. This avoids some source of truth for the URL duplicates in the HL7 terminology package.
- Duplicate CodeSystem versions within the package with "content: not-present" are skipped. This is logged at INFO level.

ABDM CodeSystem and ValueSets

To load CodeSystems and ValueSets for the **AYUSHMAN BHARAT DIGITAL MISSION (ABDM)**, download the ABDM FHIR NPM package from the ABDM FHIR Implementation Guide from the given below link:

ABDM FHIR NPM package: <https://nrces.in/ndhm/fhir/r4/package.tgz>

ABDM FHIR Implementation Guide: <https://nrces.in/ndhm/fhir/r4/index.html>

To Upload ABDM Bundle, curl utility is being used in following command. This command shows how to upload selected resources from package.tgz archive file.

```
$ curl --form file=@package.tgz
--form resourceUrls="https://nrces.in/ndhm/fhir/r4/StructureDefinition/Patient"
--form resourceUrls="https://nrces.in/ndhm/fhir/r4/StructureDefinition/Condition"
http://localhost:8080/fhir-admin/load-package
```

To load all resources within a package, use **--form resourceUrls=" "*"** as shown in below command.

Known Issues

LOINC Codes Not Found

When uploading HL7 and LOINC packages on same Snowstorm instance, this issue is faced. To resolve this, following command can be used which removes empty LOINC CodeSystem uploaded with HL7 package.

```
$ curl -X POST "localhost:9200/fhir-codesystem-version/_delete_by_query?pretty" -H  
'Content-Type: application/json' -d '{"query": {"match": {"_id": "v3-loinc"}}}'{"match":  
{"_id": "v3-loinc"}}}'
```

This command will remove CodeSystem with id **v3-loinc** from Elasticsearch Directly. After removing duplicate CodeSystems, original Loinc codes will be available.

Note

- The guide is constructed assuming the users are using the same version/edition of software/code systems mentioned in the prerequisite section of this guide.
- This guide provides setup guide for Snowstorm Terminology Server developed by SNOMED International and not to be referred for any other terminology servers.

In evidence of any material error, change, correction, concerns or assistance regarding this document, you are requested to immediately report it at: nrc-help@cdac.in

Reference

- Snowstorm GitHub repository and documentation: <https://github.com/IHTSDO/snowstorm>
- FHIR Terminology Service Specifications: <https://build.fhir.org/terminology-service.html>
- HAPI FHIR CLI Tool: https://hapifhir.io/hapi-fhir/docs/tools/hapi_fhir_cli.html
- HAPI FHIR CLI Tool – GitHub repository: <https://github.com/hapifhir/hapi-fhir/releases>