ELI ‘Pillar IV’ helper documentation

Version: 1

Date: 18.7.2022

Contents

Introduction

European legislation identifier (ELI) 'Pillar IV' helper is a tool that helps ELI publishers who are implementing the 'Pillar iv' specification. The fourth pillar defines a protocol to publish the exhaustive list of ELI URIs (uniform resource identifiers) in a Sitemap file and provide the latest updates on ELI URIs in an Atom file.

The 'Pillar IV' helper application allows ELI publishers to either:

- generate a Pillar-4-conformant Atom feed from a Sitemap file (this will read the most recent Sitemap entries to insert them into the Atom feed);
- generate both a Sitemap and an Atom feed from a tabular CSV (comma-separated values) file (the CSV file contains the ELI URI, its update date and, optionally, a title); or
- generate both a Sitemap and an Atom feed from a SPARQL query executed against a SPARQL endpoint (the query must return the ELI URI, its update date and, optionally, a title).

The intended usage of the tool is through automatically scheduled tasks.

Prerequisites

ELI 'Pillar IV' helper is a command-line Java application. It requires a Java runtime environment, version 8 or above.

Commands documentation

Atom generation from Sitemap

Command

The Atom generation process is run the following way:

java jar pillar4helper-app.jar sitemap2atom [options],

with the following possible options.

Mandatory parameters:

- --sitemapInput or -i:
  - path to the input Sitemap file, from which the most recent entries will be read to be inserted into the Atom feed;
- --sitemapBaseUrl or -su:
  - base URL (uniform resource locator) of every ELI listed in the Sitemap file – all ELIs in the Sitemap MUST begin with this base URL;
- --atomOutput or -ao:
  - path to the file where the output Atom feed will be written;
- --atomHeader or -ah:
  - path to the input Atom 'skeleton' file containing the Atom header information – this skeleton can be generated by another command.

Other optional parameters:

- --atomDays or -ad:
  - the number of days considered for including Sitemap entries in the Atom feed – entries updated since the specified number of days will be included (number of days defaults to 60).

Output

The Sitemap content is parsed, entries of less than 60 days (or another number of days if specified) are extracted and inserted into the Atom feed with the provided header information. The ELI URI is used as the title in the feed.

The Atom feed output file can then be copied to the web server to make it accessible at its final URL. The 'Pillar IV' specification recommends that '../eli/eli-update-feed.atom' be used.

Sitemap + Atom generation from CSV

Command

The sitemap+Atom generation process from CSV shall be run the following way:

java jar pillar4helper-app.jar csv2pillar4 [options],

with the following possible options.

Mandatory parameters:

- --input or -i:
  - path to the input CSV file,
  - the CSV file MUST contain two columns: the ELI URI and the update date of this ELI,
  - the CSV file MUST have a first line containing the headers, which will be ignored during parsing,
  - the date MUST use the format ‘yyyy-MM-dd’ or ‘yyyy-MM-dd'T'hh:mm:ss’,
  - an optional third column can contain a title to be included in the Atom feed;
- --sitemapOutput or -so:
  - path to the output directory where Sitemap files will be generated;
- --atomOutput or -ao:
  - path to the output file where the Atom feed will be generated;
- --sitemapBaseUrl or -su:
  - base URL of ELIs that will be listed in the Sitemap file – all ELIs in the Sitemap MUST begin with this base URL;
- --atomHeader or -ah:
  - path to the input Atom ‘skeleton’ file containing the Atom header information – this skeleton can be generated by another command.

Other optional parameters:

- --atomUrl or -au:
  - target Atom feed URL – if set, a ‘dct:relation’ attribute will be inserted into the Sitemap file to this URL;
- --atomDays or -ad:
  - the number of days considered for including Sitemap entries in the Atom feed – entries updated since the specified number of days will be included (number of days defaults to 60).

Output

The command will:

1. parse the input CSV file;
2. generate the Sitemap in the provided output directory, and handle the splitting of Sitemap files with the 50 000 limit:
   (a) if there are less than 50 000 entries, a single output file will be generated in the output directory with the name ‘sitemap.xml’,
   (b) if there are more than 50 000 entries, multiple output files will be generated with the name ‘sitemapX.xml’ (sitemap1.xml, sitemap2.xml, etc.), and a single Sitemap index will be generated with the name ‘sitemap.xml’;
3. generate the Atom feed at the provided location by inserting entries of less than the provided number of days to include – if provided, titles will be inserted into the Atom feed, otherwise, the ELI URI will be used as the title.

The Sitemap and Atom feed can be copied to the web server at their final target location. The ‘Pillar IV’ specification recommends that ‘../eli/sitemap.xml’ be used.

CSV format

The input CSV file must have the following structure.

```
"ELI","date"
"http://data.europa.eu/eli/dec/1998/538/oj",2016-05-04T11:32:59
"http://data.europa.eu/eli/reg/1976/2948/oj",2017-03-13T19:37:16
"http://data.europa.eu/eli/dec/2001/588/oj",2020-09-23T04:33:36
```

- The first line of the file will contain column names and will be ignored.
- The rest of the file must contain two columns:
  - the first column is the ELI URI;
  - the second column is the update date of the ELI:
    - it can either have the format ‘yyyy-MM-dd’ or the format ‘yyyy-MM-ddThh:mm:ss’.
- An optional third column can contain the title of the corresponding ELI.
- Column content may use quotes as delimiters.

The goal is that this CSV file can be easily generated from a query in a database.

Sitemap + Atom generation from SPARQL

Command

The sitemap+Atom generation from SPARQL process must be run the following way:

java jar pillar4helper-app.jar sparql2pillar4 [options].

Parameters

The parameters are the same as for the 'csv2pillar4' command, with the exception of '–input', which is replaced by the following two parameters.

- --query or -q:
  - path to the file containing a SPARQL query to execute against the specified SPARQL endpoint;
  - the file must be a valid SPARQL query, with an extension typically ending in '.rq';
  - the query MUST return two columns with the following name: '?eli', which must contain the IRI (internationalized resource identifier) of a legal resource, and '?updateDate', which must contain the update date of the resource in 'xsd:date' or 'xsd:dateTime' data type;
  - an optional column '?title' can contain a title to be included in the Atom feed.
- --endpoint or -e:
  - URL of the SPARQL endpoint to execute the query.

SPARQL query

The SPARQL query MUST return variables named '?eli' and '?updateDate', and optionally '?title'. See the corresponding annexes for examples of SPARQL queries.

Atom header generation

Command

Both the Atom feed generation command and the sitemap+Atom generation from CSV command can take a base Atom file as an input. This base Atom file is an Atom header with no entries, containing only the header information, in which the entries will be inserted.

This third command allows a basic conformant Atom header file to be generated, which can then be manually enhanced with extra information and can be passed as a parameter to the other command.

The Atom header generation command must be run the following way:

```
java jar pillar4helper-app.jar atomheader [options].
```

Mandatory parameters:

- --output or -o: path to the Atom output file.

Other optional parameters:

- --title or -t: the title to be inserted into the Atom header, if not provided, a placeholder value will be used;
- --id or -i: the ID of the feed to be inserted into the Atom header, if not provided, a placeholder value will be used;
- --link or -l: the link URL of the feed to be inserted into the Atom header, if not provided, a placeholder value will be used;
- --author or -a: the name of the author of the feed to be inserted into the Atom header, if not provided, a placeholder value will be used.

Output

The command will output a basic but conformant Atom header, like the following, with placeholder values.

```
<?xml version="1.0" encoding="UTF-8"?>
<feed xmlns="http://www.w3.org/2005/Atom">
 <title>**Put title here**</title>
 <link rel="self" type="application/atom+xml" href="**Put link here**" />
 <author>
  <name>**Put author name here**</name>
 </author>
 <id>**Put Atom ID here**</id>
</feed>
```

Annex 1: query to generate 'Pillar IV' from Cellar

```
# To be executed against http://publications.europa.eu/webapi/rdf/sparql
prefix cdm: <http://publications.europa.eu/ontology/cdm#>
select (IRI(STR(?eliString)) AS ?eli) ?updateDate where {
 ?x cdm:resource_legal_eli ?eliString .
 ?x <http://publications.europa.eu/ontology/cdm/cmr#lastModificationDate> ?updateDate .
}
```

Annex 2: query to generate ‘Pillar IV’ from Legilux

```
# To be executed against https://data.legilux.public.lu/sparqlendpoint
PREFIX jolux: <http://data.legilux.public.lu/resource/ontology/jolux#>
SELECT ?eli (STR(?publicationDate) AS ?updateDate) (STR(?titleLang) AS ?title)
WHERE {
 ?eli a jolux:Act .
 ?eli jolux:publicationDate ?publicationDate .
 OPTIONAL { ?eli jolux:isRealizedBy/jolux:title ?titleLang . }
}
ORDER BY DESC(?publicationDate)
```