Hewlett Packard
Enterprise

Eduction SDK

Software Version: 11.2.0

Windows and UNIX operating systems

Programming Guide

Document Release Date: October 2016

Software Release Date: October 2016

Legal Notices

Warranty

The only warranties for Hewlett Packard Enterprise Development LP products and services are set forth in the express warranty statements accompanying such products and services. Nothing herein should be construed as constituting an additional warranty. HPE shall not be liable for technical or editorial errors or omissions contained herein.

The information contained herein is subject to change without notice.

Restricted Rights Legend

Confidential computer software. Valid license from HPE required for possession, use or copying. Consistent with FAR 12.211 and 12.212, Commercial Computer Software, Computer Software Documentation, and Technical Data for Commercial Items are licensed to the U.S. Government under vendor's standard commercial license.

Copyright Notice

Trademark Notices

Adobe™ is a trademark of Adobe Systems Incorporated.

Microsoft® and Windows® are U.S. registered trademarks of Microsoft Corporation.

UNIX® is a registered trademark of The Open Group.

This product includes an interface of the 'zlib' general purpose compression library, which is Copyright © 1995-2002 Jean-loup Gailly and Mark Adler.

Documentation Updates

The title page of this document contains the following identifying information:

- Software Version number, which indicates the software version.
- Document Release Date, which changes each time the document is updated.
- Software Release Date, which indicates the release date of this version of the software.

To check for recent software updates, go to https://downloads.autonomy.com/productDownloads.jsp.

To verify that you are using the most recent edition of a document, go to https://softwaresupport.hpe.com/group/softwaresupport/search-result?doctype=online help.

This site requires that you register for an HPE Passport and sign in. To register for an HPE Passport ID, go to https://hpp12.passport.hpe.com/hppcf/login.do.

You will also receive updated or new editions if you subscribe to the appropriate product support service. Contact your HPE sales representative for details.

Support

Visit the HPE Software Support Online web site at https://softwaresupport.hpe.com.

This web site provides contact information and details about the products, services, and support that HPE Software offers.

HPE Software online support provides customer self-solve capabilities. It provides a fast and efficient way to access interactive technical support tools needed to manage your business. As a valued support customer, you can benefit by using the support web site to:

- Search for knowledge documents of interest
- Submit and track support cases and enhancement requests
- Access product documentation
- Manage support contracts
- Look up HPE support contacts
- Review information about available services
- Enter into discussions with other software customers
- Research and register for software training

Most of the support areas require that you register as an HPE Passport user and sign in. Many also require a support contract.

To register for an HPE Passport ID, go to https://hpp12.passport.hpe.com/hppcf/login.do.

To find more information about access levels, go to https://softwaresupport.hpe.com/web/softwaresupport/access-levels.

To check for recent software updates, go to https://downloads.autonomy.com/productDownloads.jsp.

About this PDF Version of Online Help

This document is a PDF version of the online help. This PDF file is provided so you can easily print multiple topics from the help information or read the online help in PDF format. Because this content was originally created to be viewed as online help in a web browser, some topics may not be formatted properly. Some interactive topics may not be present in this PDF version. Those topics can be successfully printed from within the online help.

Contents

Chapter 1: Introduction to Eduction

This section introduces Eduction components and concepts.

- About Eduction
- Eduction Architecture
- Eduction Components
- Common Use Scenarios
- Eduction Concepts

About Eduction

Eduction is a tool that you can use to identify and extract an *entity* (a word, phrase, or block of information) from text, based on a *pattern* you define. The entities are contained inside *grammar* files.

Eduction includes standard grammar files, which allow you to quickly and easily extract commonly sought entities, such as social security numbers, names, telephone numbers, addresses, and so on.

Eduction also contains sentiment grammar files that enable you to identify positive or negative sentiments in text.

Eduction Architecture

Figure 1 shows the basic Eduction architectural flow for extracting entities from a document when Eduction is used with IDOL. For information about the individual components for Eduction SDK, see Eduction Components, on page 15.

Figure 1: IDOL Eduction workflow

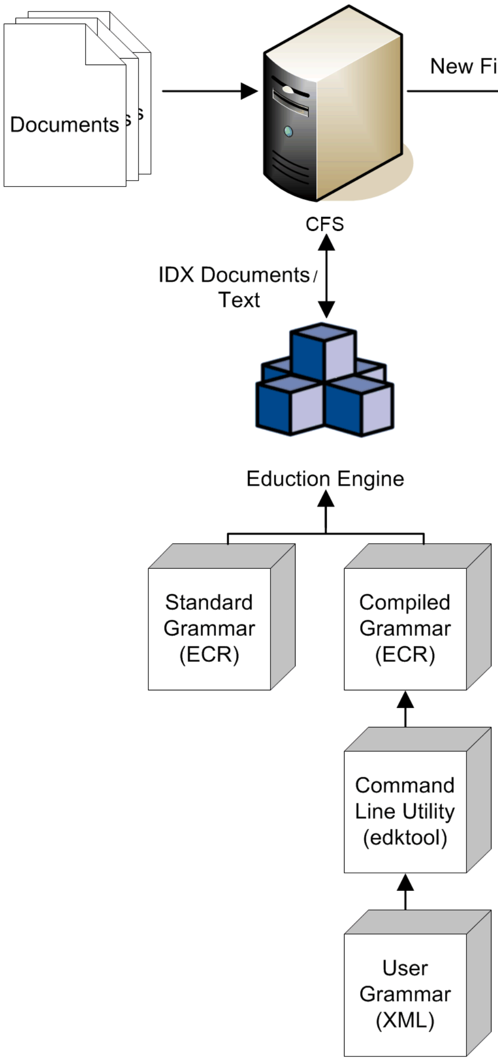


Figure 2 shows the Eduction components used when programming with the Eduction SDK. For information about the individual components for Eduction, see Eduction Components, on page 15.

Figure 2: Eduction SDK workflow

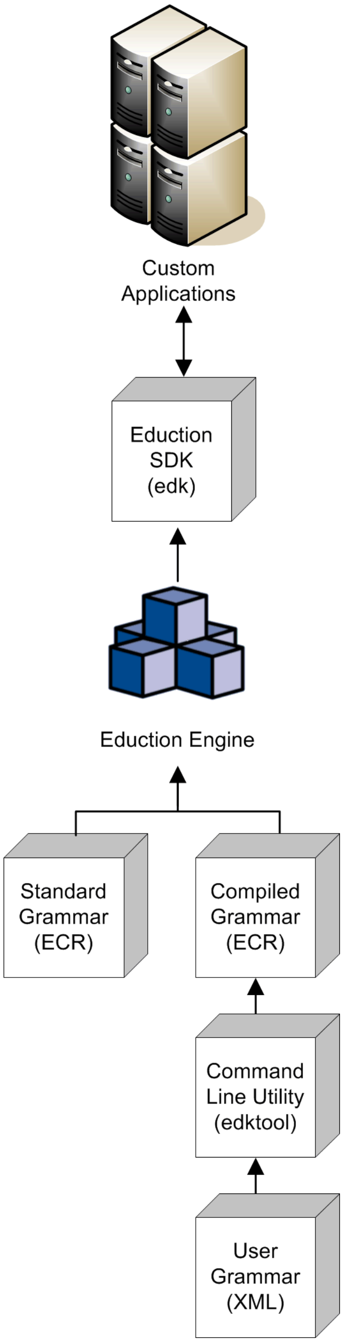


Figure 3 shows how you can perform extraction by using the Eduction grammar and the Eduction ACI server. For information about the individual components for Eduction, see Eduction Components, on page 15.

Figure 3: Extraction using the Eduction ACI Server

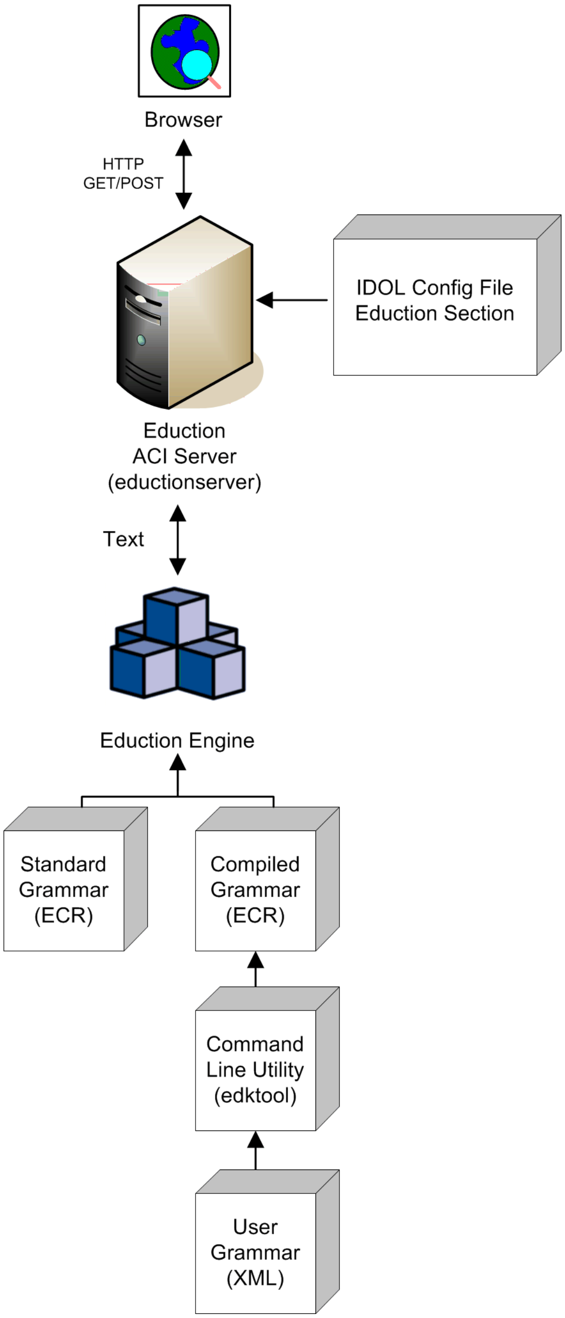
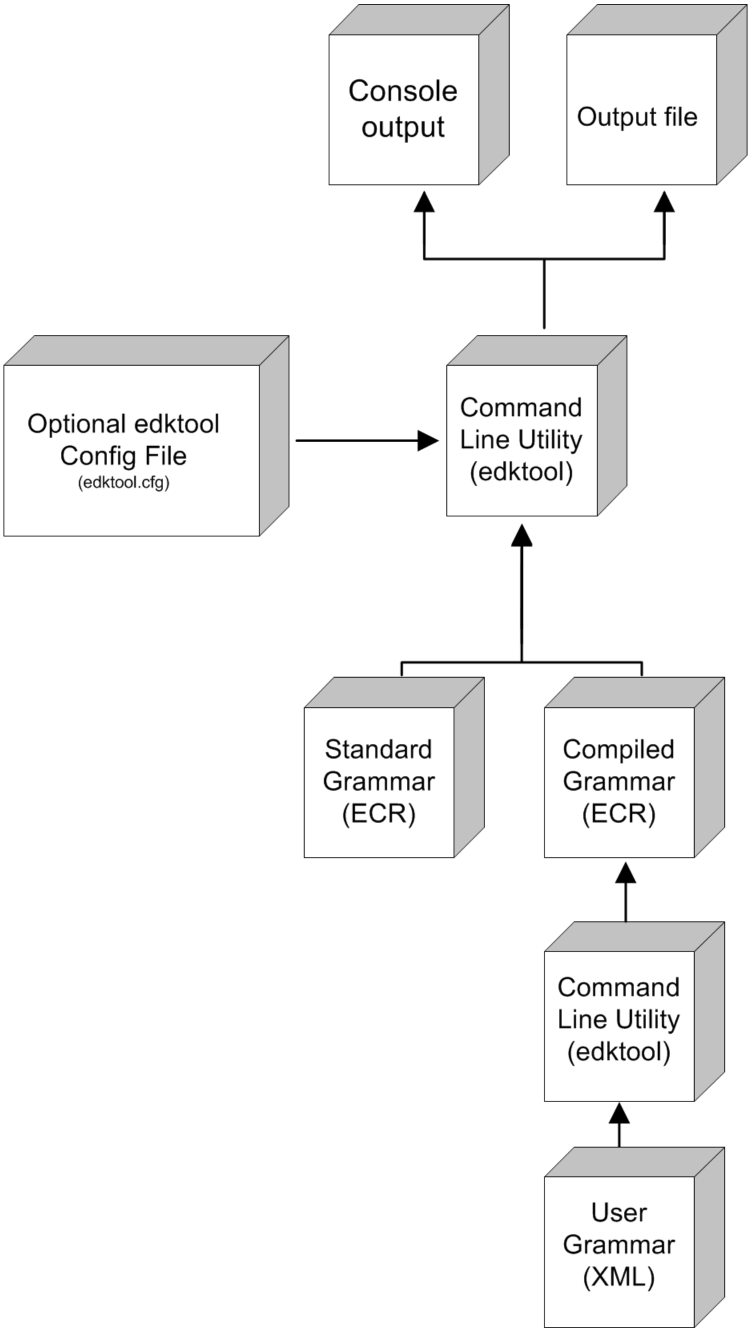


Figure 4 shows the workflow if you want to use the `edktool` command-line tool to compile and test grammar files, list entities, extract entitles, and so on. For more information see edktool Command-Line Tool, on page 87.

Figure 4: `edktool` workflow



Eduction Components

Eduction consists of the following components:

- **Eduction Engine**. The core logic in Eduction that supports the extraction of predefined entities from text, using Eduction grammar files compiled from XML source.

- **Command-line tool (`edktool`)**. `edktool` compiles grammar files, performs extraction from the command line, lists entities in a grammar file, and tests the accuracy of the extraction. See edktool Command-Line Tool, on page 87. A separate `edktool.cfg` configuration file contains `edktool` settings.
- **Standard grammars**. A collection of ECR grammar files covering commonly sought entities such as Social Security numbers, postal addresses, telephone numbers, people names, and so on. See Standard Grammars, on page 129.

  You can license standard grammar files by category and by language, so that it is possible to be licensed for any combination of category (for example, sentiment, place, or person) and language.

  The license is now required when using `edktool` to compile any grammar files that use the standard grammars as resources, as well as for extraction.
- **User grammar**. XML files created by the user that describe entities that can locate patterns in text using the Eduction grammar language. See Grammar Reference, on page 299.
- **Eduction tasks in CFS**. You can use the Connector Framework Server (CFS) to manipulate and enrich documents in a variety of ways, including performing Eduction on document fields, using Lua scripts to run Eduction processing and post-processing tasks, and redacting information in documents. For more information, see the *Connector Framework Server Administration Guide*.
- **Eduction SDK**. This is the C and Java programming interface for Eduction.

  For information on the IDOL configuration file for the Eduction module, refer to the *IDOL Eduction User Guide**.

  For information on the IDOL configuration file for the ACI server, see Eduction ACI Server, on page 69.
- **Eduction ACI server (eductionserver)**. This component is a stand-alone Eduction server, included in the SDK and typically used with the LSA grammar to identify sentiment in passages of text. The `eductionserver.cfg` configuration file contains the settings for the Eduction server.

* Not all distributions of Eduction contain this component.

Common Use Scenarios

The following scenarios describe possible use cases for Eduction:

- Use `edktool` to compile grammars, and test extraction on IDX, XML, or plain text documents.
- Use the Eduction ACI server from a browser to identify employee names in a document.
- Use Eduction to extract and compile a list of product names or company names from your data.
- Use Eduction as a preprocessing tool to automatically extract entities from your documents and add them as metadata to your documents before indexing.
- Use Eduction to extract common search phrases from documents before indexing, and tag the documents with this data.
- Set up an Eduction task to extract different parts of your documents to different fields. For example, if your documents are letters, you can extract the name, address, and date from each document into predefined fields.
- Use Eduction to extract information from the results of Optical Character Recognition.
- Use the redaction feature to conceal sensitive information in your output, so that you can conform to data protection standards and use your records for multiple purposes.

The Eduction architecture supports the extraction of:

- common entities, including place, personal names, and companies.
- any basic entities you define using dictionaries or grammars.
- complex entities you define (facts, events, relationships, and so on).

Eduction Concepts

This section introduces some of the Eduction concepts and terminology used throughout this guide.

Dictionaries

In Eduction, a *dictionary* is a file that provides a vocabulary for a simple entity, such as cities or countries. The dictionaries are organized by a list of headwords (simple or compound). Each headword can also have a set of associated words called *synonyms*. Eduction uses the dictionary to scan a document and extract the defined entities that match the search pattern.

Dictionaries are written in XML and included in a grammar file.

Grammars

A *grammar* is a file that provides rules for complicated entities such as URLs or postal addresses. The entities can be recursively defined. Rules can refer to entities in external grammars and dictionaries. Eduction uses the grammar to scan a document and extract the defined entities that match the search pattern.

The pattern can be a dictionary of names such as people or places, or the pattern can describe what the sequence of text looks like without having to list it explicitly, for example, a telephone number, or a time.

Grammars are written in XML and the regular expression format. Eduction supports context-free grammars.

Eduction also allows you to extend existing grammars, and to author new ones, either from scratch or by referencing existing entities. For more information about how and when to extend your grammars, and how to improve the recall of your grammar files, refer to *IDOL Expert*.

Linguistic Sentiment Analysis

The sentiment analysis grammar files contain dictionaries of types of word (for example, positive adjective, negative noun, neutral adverb, and so on), and patterns that describe how to combine these dictionaries to form positive and negative phrases.

For example, you could run sentiment extraction using the English sentiment grammar file (`sentiment_eng.ecr`), with the following hotel review as the input file:

The room was nice enough, with a plug in radiator, tv with an English news channel, hot shower, comfy bed. The receptionist we first dealt with was miserable and rude, and just grunted at us and rolled her eyes because we were too early for check in having just got off the morning train from Khabarovsk. Fortunately, a younger receptionist with a nice smile appeared, spoke to us helpfully suggesting a few cafes nearby to pass some time, and we tried to forget about the other woman. Breakfast is terrible. Unidentifiable cordials, gloomy porridge, bread rolls filled with things you don't expect for breakfast, like potato, egg and dill. Don't come here for the breakfast, but for the cost of the room in a city like Vladivostok, the hotel is still decent value for money.

The following is a sample of the output that this produces:

```
<?xml version="1.0" encoding="UTF-8"?>
  <MATCHLIST>
    <DOCUMENT Type="IDOL IDX" ID="Unknown">
      <FIELD Name="DRECONTENT">
        <FIELD_INSTANCE Value="1">
          <MATCH EntityName="sentiment/positive/eng" Offset="7" OffsetLength="5"
          Score="1.05" NormalizedTextSize="17" NormalizedTextLength="17"
          OriginalTextSize="17" OriginalTextLength="17">
            <ORIGINAL_TEXT>The room was nice</ORIGINAL_TEXT>
            <NORMALIZED_TEXT>The room was nice</NORMALIZED_TEXT>
            <COMPONENTS>
              <COMPONENT Name="TOPIC" Text="The room" Offset="0"
              OffsetLength="0" TextSize="8" TextLength="8"/>
              <COMPONENT Name="SENTIMENT" Text="nice" Offset="13"
              OffsetLength="13" TextSize="4" TextLength="4"/>
            </COMPONENTS>
          </MATCH>
          <MATCH EntityName="sentiment/negative/eng" Offset="494"
          OffsetLength="492" Score="1.2" NormalizedTextSize="21"
          NormalizedTextLength="21" OriginalTextSize="21"
          OriginalTextLength="21">
```

```
            <ORIGINAL_TEXT>Breakfast is terrible</ORIGINAL_TEXT>
            <NORMALIZED_TEXT>Breakfast is terrible</ NORMALIZED_TEXT>
            <COMPONENTS>
              <COMPONENT Name="TOPIC" Text="Breakfast" Offset="0"
              OffsetLength="0" TextSize="9" TextLength="9"/>
              <COMPONENT Name="SENTIMENT" Text="terrible" Offset="13"
              OffsetLength="13" TextSize="8" TextLength="8"/>
            </COMPONENTS>
          </MATCH>
        </FIELD_INSTANCE>
      </FIELD>
    </DOCUMENT>
</MATCHLIST>
```

The following example configuration shows the recommended usage:

```
[Eduction]
ResourceFiles=grammars/sentiment_eng.ecr
// Note: replace sentiment_eng.ecr by sentiment_user_eng.ecr if using user
modification

// standard entities for all sentiment analysis in English:
Entity0=sentiment/positive/eng
Entity1=sentiment/negative/eng
EntityField0=POSITIVE_VIBE
EntityField1=NEGATIVE_VIBE
EntityComponentField0=TOPIC,SENTIMENT
EntityComponentField1=TOPIC,SENTIMENT

// some invalid matches are given very low scores so that we can filter them out:
MinScore=0.1

// for extraction of Twitter handles, hashtags and emoticons:
TangibleCharacters=@#:;

// for displaying metadata:
OutputScores=True
OutputSimpleMatchInfo=False
EnableComponents=True
```

For more information on the sentiment analysis grammar files, how to adjust the sentiment analysis by extending the grammars, and the features that the sentiment grammars support, refer to *IDOL Expert*.

Perform Sentiment Analysis on Short Comments

The standard sentiment analysis grammars are designed for high precision. For some sources of short comment data, such as YouTube comments, no positive or negative matches are found in some documents despite sentiment clearly being expressed.

If recall with the full `sentiment_eng.ecr` grammar file is too low, and your documents are generally short comments, use `sentiment_basic_eng.ecr` to extract additional matches.This grammar contains carefully-selected lists of positive and negative terms that help determine the sentiment of a document in which sentiment_eng.ecr found no matches.

sentiment_basic_eng.ecr contains terms in title case, but research shows that for most data these impair recall, so these are given a lower score. HPE recommends that you set EntityMinScore*N* to 0.4 to filter out these terms unless you need them.

sentiment_basic_eng.ecr does not expose TOPIC or SENTIMENT components, and does not use scores to reflect strength or reliability of polarity. The following additional example configuration shows the recommended usage:

```
[Eduction]
ResourceFiles=grammars/sentiment_eng.ecr,grammars/sentiment_basic_eng.ecr
// optional further layer of analysis for very short documents:
Entity2=sentiment/basic_positive/eng
Entity3=sentiment/basic_negative/eng
EntityField2=BASIC_POSITIVE_VIBE
EntityField3=BASIC_NEGATIVE_VIBE
// remove this setting to include basic matches in titlecase - this is not
recommended because on most data it decreases precision:
EntityMinScore2=0.4
EntityMinScore3=0.4
```

Components

Some of the standard grammar files contain *components*, which enable you to extract attributes from matched phrases, such as topic, subject, and positive or negative sentiments. The attributes are called components because they are the components of a single match.

For example, if you used sentiment analysis to match the phrase *Their service is fantastic* as conveying positive sentiment, you can then use components to identify *service* as the subject matter, and *fantastic* as the adjective that describes the subject (note that the sentiment is not necessarily an adjective in all cases). You can also set up components when you write your own custom grammar files.

Note: sentiment_basic_eng.ecr does not support the TOPIC or SENTIMENT components.

For more information on how to configure and define components in your grammar files, and when to use them, refer to *IDOL Expert*.

Related Topics

- Configuration Parameters Used by edktool Only, on page 125
- EntityComponentFieldN, on page 111

Extraction

Eduction extracts entities from documents based on the rules you have created in your dictionaries and grammars. Eduction can output files in multiple formats using edktool.

For each nominated field in a document, Eduction identifies each instance of the requested entity. Eduction returns an XML list of matches, or adds the matches to the source document as new fields. Eduction can also identify components of an entity match, such as:

- the parts of a social security number or phone number.
- a confidence score for the accuracy of the match.

Results Relevance

Eduction returns entities based on the extraction rules from the grammars and dictionaries. Eduction provides a test mode to measure extraction relevance *precision* and *recall*. Precision and recall are based on the comparison between human-marked results and engine-marked results. The following terms describe result relevance as used in Eduction.

- **True Positives** (TP). Human-marked results that are also marked by the engine. These results specify that an entity returned by the engine has also been marked as true by the person marking the document.
- **False Positives** (FP). Engine-marked results that are not marked by a human. These results specify that an entity returned by the engine has not been marked as true by the person marking the document.
- **True Negatives** (TN). Results that are not marked either by the person marking the document, or the engine.
- **False Negatives** (FN). Human-marked results that are not marked by the engine. These results specify that an entity not returned by the engine has been marked as true by the person marking the document.

From these relevance terms, you can determine precision and recall as follows:

- Recall is the percentage of true relevant entities that are extracted by an extraction rule, that is,

  ```
  TP / (TP + FN) * 100
  ```

- Precision is the percentage of extracted entities that are true entities, that is,

  ```
  TP / (TP + FP) * 100
  ```

Case Sensitive Matches

You can configure Eduction to match characters case sensitively or case insensitively. By default, it is case sensitive, which has better performance.

The simplest way to match case insensitively is to disable the `MatchCase` configuration parameter (set the parameter to **`False`** in the configuration file). Alternatively, if you are creating your own custom XML grammar files, you can configure individual grammars, entities, and entries individually to be case sensitive or insensitive. If you configure case sensitivity at a lower level, it overrides the higher level settings. Additionally, if you reference the entity in another entity, it maintains its own case sensitivity setting.

Most entities in the standard grammars do not have case sensitivity set explicitly, giving you the flexibility to use case sensitivity as required in your grammars.

Note: If you design an entity for case-insensitive matching, it is important that entries in the entity have a consistent case style to ensure that all matches are extracted correctly. You should use all lower case, all upper case, or all initial capitals, but not a mixture.Eduction uses an optimization technique for case insensitive matching that might not extract every possible match if the entity is not defined consistently.

Case Insensitive Match Performance

Case sensitive matching generally has better performance than case insensitive matching. If you require case insensitive matching, you can use case normalization to give the same performance as case-sensitive matching.

When you want to use case normalization:

- Do not set case sensitivity explicitly in grammars and entities.
- Set the `MatchCase` configuration parameter to **`True`**.
- Create all entries in your entities in either all lower case, or all upper case.
- Set `CaseNormalization` to:
- **`LOWER`** if all your entities are lower case
- **`UPPER`** if all your entities are upper case.

Eduction normalizes the input data accordingly before the (case sensitive) matching. This process means that both your input and grammars are all in the same case, so the matching is effectively case insensitive, with the performance benefits of case sensitive matching.

When to Configure Case Sensitivity

HPE recommends that you always create and use Eduction grammars that allow you to do case sensitive matching, because it has better performance. Most of the standard grammars come with entities using common and appropriate case styles. Some also have different entities for different case styles. If your data uses a consistent case, it is unlikely that you need to use case insensitive matching.

Case Sensitivity and Configured Field Names

You can set the `CaseSensitiveFieldName` configuration parameter to **`1`** to preserve the case and case sensitivity of configured field names in your output. See CaseSensitiveFieldName , on page 107 for more information.

Related Topics

- CaseNormalization , on page 107
- MatchCase , on page 117

Licenses

To use HPE IDOL solutions, you must have a running HPE License Server, and a valid license key file for the products that you want to use. Contact HPE Big Data Support to request a license file for your installation.

License Server controls the IDOL licenses, and assigns them to running components. License Server must run on a machine with a static, known IP address, MAC address, or host name. The license key file is tied to the IP address and ACI port of your License Server and cannot be transferred between machines. For more information about installing License Server and managing licenses, see the *License Server Administration Guide*.

When you start Eduction SDK, it requests a license from the configured License Server. You must configure the host and port of your License Server in the Eduction SDK configuration file.

You can revoke the license from a product at any time, for example, if you want to change the client IP address or reallocate the license.

Caution: Taking any of the following actions causes the licensed module to become inoperable.

You **must not**:

- Change the IP address of the machine on which a licensed module runs (if you use an IP address to lock your license).
- Change the service port of a module without first revoking the license.
- Replace the network card of a client without first revoking the license.
- Remove the contents of the `license` and `uid` directories.

All modules produce a `license.log` and a `service.log` file. If a product fails to start, check the contents of these files for common license errors. See Troubleshoot License Errors, on page 25.

Display License Information

You can verify which modules you have licensed either by using the IDOL Admin interface, or by sending the `LicenseInfo` action from a web browser.

To display license information in IDOL Admin

- In the **Control** menu of the IDOL Admin interface for your License Server, click **Licenses**.

  The **Summary** tab displays summary information for each licensed component, including:

  - The component name.
  - The number of seats that the component is using.
  - The total number of available seats for the component.
  - (Content component only) The number of documents that are currently used across all instances of the component.
  - (Content component only) The maximum number of documents that you can have across all instances of the component.

  The **Seats** tab displays details of individual licensed seats, and allows you to revoke licenses.

To display license information by sending the LicenseInfo action

- Send the following action from a web browser to the running License Server.

```
http://LicenseServerHost:Port/action=LicenseInfo
```

where:

`LicenseServerHost` is the IP address of the machine where License Server resides.

`Port` is the ACI port of License Server (specified by the `Port` parameter in the `[Server]` section of the License Server configuration file).

In response, License Server returns the requested license information. This example describes a license to run four instances of IDOL Server.

```
<?xml version="1.0" encoding="UTF-8" ?>
<autnresponse xmlns:autn="http://schemas.autonomy.com/aci/">
  <action>LICENSEINFO</action>
  <response>SUCCESS</response>
  <responsedata>
    <LicenseDiSH>
      <LICENSEINFO>
        <autn:Product>
          <autn:ProductType>IDOLSERVER</autn:ProductType>
          <autn:TotalSeats>4</autn:TotalSeats>
          <autn:SeatsInUse>0</autn:SeatsInUse>
        </autn:Product>
      </LICENSEINFO>
    </LicenseDiSH>
  </responsedata>
</autnresponse>
```

Configure the License Server Host and Port

Eduction SDK is licensed through HPE License Server. In the Eduction SDK configuration file, specify the information required to connect to the License Server.

To specify the license server host and port

1. Open your configuration file in a text editor.
2. In the `[License]` section, modify the following parameters to point to your License Server.

   `LicenseServerHost` The host name or IP address of your License Server.

   `LicenseServerACIPort` The ACI port of your License Server.

   For example:

   ```
   [License]
   LicenseServerHost=licenses
   LicenseServerACIPort=20000
   ```

3. Save and close the configuration file.

Revoke a Client License

After you set up licensing, you can revoke licenses at any time, for example, if you want to change the client configuration or reallocate the license. The following procedure revokes the license from a component.

To revoke a license

1. Stop the HPE solution that uses the license.
2. At the command prompt, run the following command:

```
InstallDir/ExecutableName[.exe] –revokelicense –configfile cfgFilename
```

This command returns the license to the License Server.

You can send the `LicenseInfo` action from a web browser to the running License Server to check for free licenses. In this sample output from the action, one IDOL Server license is available for allocation to a client.

```
<autn:Product>
   <autn:ProductType>IDOLSERVER</autn:ProductType>
   <autn:Client>
      <autn:IP>192.123.51.23</autn:IP>
      <autn:ServicePort>1823</autn:ServicePort>
      <autn:IssueDate>1063192283</autn:IssueDate>
      <autn:IssueDateText>10/09/2003 12:11:23</autn:IssueDateText>
   </autn:Client>
      <autn:TotalSeats>2</autn:TotalSeats>
      <autn:SeatsInUse>1</autn:SeatsInUse>
</autn:Product>
```

Troubleshoot License Errors

The table contains explanations for typical licensing-related error messages.

License-related error messages

Error message	Explanation
Error: Failed to update license from the license server. Your license cache details do not match the current service configuration. Shutting the service down.	The configuration of the service has been altered. Verify that the service port and IP address have not changed since the service started.
Error: License for *ProductName* is invalid. Exiting.	The license returned from the License Server is invalid. Ensure that the license has not expired.
Error: Failed to connect to license server using cached licensed details.	Cannot communicate with the License Server. The product still runs for a limited period; however, you should verify whether your License Server is still available.

License-related error messages, continued

Error message	Explanation
Error: Failed to connect to license server. Error code is SERVICE: *ErrorCode*	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.
Error: Failed to decrypt license keys. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Provide HPE Big Data Support with the exact error message and your license file.
Error: Failed to update the license from the license server. Shutting down	Failed to retrieve a license from the License Server or from the backup cache. Ensure that your License Server can be contacted.
Error: Your license keys are invalid. Please contact Autonomy support. Error code is SERVICE:*ErrorCode*	Your license keys appear to be out of sync. Provide HPE Big Data Support with the exact error message and your license file.
Failed to revoke license: No license to revoke from server.	The License Server cannot find a license to revoke.
Failed to revoke license from server *LicenseServer Host*:*LicenseServerPort*. Error code is *ErrorCode*	Failed to revoke a license from the License Server. Provide HPE Big Data Support with the exact error message.
Failed to revoke license from server. An instance of this application is already running. Please stop the other instance first.	You cannot revoke a license from a running service. Stop the service and try again.
Failed to revoke license. Error code is SERVICE:*ErrorCode*	Failed to revoke a license from the License Server. Provide HPE Big Data Support with the exact error message.
Your license keys are invalid. Please contact Autonomy Support. Error code is ACISERVER:*ErrorCode*	Failed to retrieve a license from the License Server. Provide HPE Big Data Support with the exact error message and your license file.
Your product ID does not match the generated ID.	Your installation appears to be out of sync. Forcibly revoke the license from the License Server and rename the `license` and `uid` directories.
Your product ID does not match this configuration.	The service port for the module or the IP address for the machine appears to have changed. Check your configuration file.

Chapter 2: Deploy Eduction SDK

This chapter describes the files in the Eduction SDK and how to deploy the SDK. It contains the following sections:

Eduction SDK

The Eduction SDK includes the following components:

- *Eduction SDK Programming Guide*
- Standard collection of grammar files covering a range of commonly used entities
- `edktool` command-line tool used for compiling Eduction XML source grammar files into compiled run-time ECR files
- Eduction Server
- C API
- Java API

Eduction SDK Installation

Use the following procedure to install Eduction SDK.

To install Eduction SDK

1. Go to the Download Center on the Customer Support site.
2. Download the latest Eduction SDK .zip file for your platform:

   `EductionSDK_<VersionNumber>_<Platform>.zip`
3. Copy the .zip file to a location on your local machine, and unzip it.

Eduction Server

Eduction Server is an ACI server, and requires a configuration file to run. A sample configuration file is provided with Eduction SDK. To run Eduction Server, you must update this configuration file with the host and port details of your HPE License Server.

Start and Stop Eduction Server

- If you are running Eduction Server on Windows, you can install the server as a Windows service. After you have installed the server as a service, you can start and stop the service from the Windows Services manager.
- If you are running Eduction Server on Linux and UNIX, you can use the `StartEductionServer.sh` and `StopEductionServer.sh` shell scripts to start and stop Eduction Server.

C API Component

The C API component of the Eduction SDK includes:

- The Eduction header file (`edk.h`).
- The Eduction SDK library linker file (`edk.lib`), Windows only.
- The Eduction SDK library on Windows (`edk.dll`) or shared object on UNIX (`libedk.so`).
- Several sample C programs that demonstrate various SDK features.

Solution files and project files are provided for Visual Studio 2010.

To use the Eduction SDK in C, include the `edk.h` header file from your C source code and link with the SDK library. A sample makefile is provided showing how to compile and link to the SDK.

> **Note:** On Windows, you must specify the Eduction library in the `PATH` environment variable. On UNIX, the shared object must be in the library search path.

Build the Sample Programs

You can use the sample programs to validate that the Eduction EDK compiles, links, and runs correctly.

UNIX

Use the following procedure to build the sample programs on UNIX.

To build the sample programs on UNIX

1. Navigate to the `c_api/test` directory.
2. Set the `PLATFORM` environment variable (refer to the makefile for more details).
3. Run `make dir=ext`.

The resulting binaries require that the shared object be in the library search path. By default, this is not the case, so a Perl script is provided to add the shared object to the search path by setting LD_LIBRARY_PATH. The script then runs the test binary. To run the script, use the following command:

```
perl runtestedk.pl ext
```

The Perl script expects a valid Eduction OEM license key with the name licensekey.dat to be located in the test directory. If the test works correctly, you see a line of text in the output beginning with the string PASS:, otherwise you see the string FAIL:.

Windows

To build the test executable binaries, a Visual Studio 2010 solution file, test.sln, is provided in the test folder.

The resulting binaries require that the edk.dll be in the library search path. By default, this is not the case, so a Perl script is provided to add the library to the search path and then run the test binary. To run the script, use the following command:

```
perl runtestedk.pl ext
```

The Perl script expects a valid Eduction OEM license key with the name licensekey.dat to be located in the test directory. If the test works correctly, you see a line of text in the output beginning with the characters PASS:, otherwise you see FAIL:.

Java API Component

The Java API component of the Eduction SDK includes:

- The Eduction library on Windows (edkjni.dll) or shared object on UNIX (libedkjni.so).
- The Java JAR file (edk.jar).
- Java code samples that you can compile and execute by using the Eduction SDK. These samples are located in the \src\com\autonomy\eduction\test\ directory of the Eduction SDK root directory, and are named SampleN.java (Sample1.java, Sample2.java, and so on). Each SampleN.java file illustrates aspects of SDK functionality, and contains detailed descriptions of the functionality in question. For information on how to compile and run Sample1.java, see Build and Run the Sample Programs, below.

  Note: Before you follow the instructions on how to compile and run Sample1.java, you must copy the licensekey.dat file into the test subfolder of your Eduction SDK installation. You must perform all command-line operations from the Eduction SDK installation directory.

Note: On Windows, you must specify the Eduction library in the PATH environment variable. On UNIX, the shared object must be in the library search path.

Build and Run the Sample Programs

Use the following procedure to build and run the sample programs.

To build and run the sample programs

1. Navigate to the `java_api` directory.
2. Run `ant all`.

Chapter 3: API Reference

This chapter describes the C and Java APIs for Eduction SDK, and contains the following sections:

- C API Concepts
- C API Reference
- C API Examples
- Java API Concepts
- Java API Reference
- Java API Example

C API Concepts

This section describes concepts required to implement C language applications for Eduction SDK.

Include Files

The Eduction SDK C API uses the `edk.h` include file. This include file contains the core APIs for the Eduction SDK engine.

Naming Conventions

The types, functions, and macros specific to Eduction SDK are prefixed with the string `Edk`.

Concurrency Control

Concurrency in Eduction is handled using *sessions*. An instance of an `EDKEngine` is initialized with corresponding grammars for entity extraction. Each such engine can be associated with one or more sessions. All the sessions in the engine share the same grammars. The engine must be configured fully before any sessions are created.

After you create a session, an `EdkError` is thrown if you change the engine settings. However, each individual session can process many documents or streams. Each session maintains its state independent of others. Each engine must be associated with at least one session by default.

Standalone API Usage

The Eduction Software Development Kit (SDK) C API allows C developers to interact directly with the Eduction engine.

At the core of the API is an eduction engine. The first call of the API in an application should create the engine. You must then set a valid license for the engine.

After you create it, you can optionally configure an engine to determine its matching behavior. You must load one or more resource files that contain the eduction grammars, and you must add one or more entities, from the loaded resource files, to match against.

Data processing is performed in an eduction session. You can create multiple sessions for each eduction engine. All sessions use the same loaded grammars and entities. Each session maintains its own state so that the sessions can run concurrently in a multithreaded application.

You can use the session to process multiple documents. You can pull (stream) or push (add) data. You call a function to get the next available match. You can call this function repeatedly to cycle through all the matches. For each match, you can access the associated text and properties by using several function calls.

You can keep a session alive for as long as necessary. However, you must destroy it before you destroy the engine that is associated with it. The call to destroy the engine should be the last call of this API in an application.

This section describes the skeletal structure of a stand-alone application using the API. See the source code in C API Examples, on the next page. Typically, your application takes the following actions:

1. Include `edk.h`.
2. Instantiate the engine and obtain an engine handle.
3. Set the license key.
4. Configure the engine to:
    - set optional parameters.
    - load the grammar files to use for matching.
    - add specific entities from grammars to use for matching.
5. Create a session associated with the engine, and obtain a session handle. You can create and run concurrent sessions in a multithreaded application. Each session uses the same grammars, but maintains its own state.
6. Feed UTF-8 encoded text to the session.
7. Call `EdkGetNextMatch` to obtain an entity match. You can call this method repeatedly to obtain all matches.
8. If required, call `EdkGetRedactedText` to produce redacted output.
9. For each match, get details and properties of the match.
10. To process multiple documents, repeat Step 6 to Step 9.
11. Release resources when done. You must destroy all session handles before destroying the engine handle.

C API Reference

The C API Reference documentation is located in the following directory:

InstallDir/c_api/help/

where *InstallDir* is the directory where you have installed the Eduction SDK.

C API Examples

This section provides sample programs that demonstrate usage of specific API functions.

basic.c

The basic.c program demonstrates the basic usage of the C API functions, as described in Standalone API Usage, on the previous page. This program:

- creates an Eduction engine.
- configures an Eduction engine.
- associates a session with the Eduction engine.
- adds text to the session, and performs an entity extraction on the text.
- outputs the results.

To build and run the sample, navigate to the `test` directory in your Eduction installation, and open the text file `readme.txt`. This file provides specific directions on how to compile and run the sample. You must have an Eduction OEM license key to run the sample.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <malloc.h>
#include <edk.h>
#include <string.h>

#ifdef _WIN32
#define stat _stat
#define off_t _off_t
#endif // _WIN32
#define BUFLEN 5120
```

```
// Helper function
void displayusageinfo() {
   EDK_VERSION_INFO versionInfo;
   EdkGetVersion(&versionInfo);
   if (versionInfo.vChangeSet)
      printf("INFO: Eduction SDK Sample for SDK version       v%i.%i.%i.%i.%i\n",
versionInfo.vMajor,       versionInfo.vMinor, versionInfo.vServicePack,
versionInfo.vRelease, versionInfo.vChangeSet);
   else
      printf("INFO: Eduction SDK Sample for SDK version       v%i.%i.%i.%i\n",
versionInfo.vMajor,       versionInfo.vMinor, versionInfo.vServicePack,
versionInfo.vRelease);
   printf("INFO: SDK Built: %s\n", versionInfo.buildTime);
   printf("INFO: Copyright %s\n", versionInfo.copyright);
   printf("INFO: Usage: sample1 <grammarpath> <entity> <documentpath>
<licensepath>\n");
   printf("INFO: Parameters:\n");
   printf("INFO:     <grammarpath>   Path to the grammar file that defines
matchable\n");
   printf("INFO:                     entities. The grammar file can be in
uncompiled (XML)\n");
   printf("INFO:                     or compiled (ECR) format.\n");
   printf("INFO:     <entities>      Comma deliminted list of entities in the
grammar file\n");
   printf("INFO:                     to be used for matching.\n");
   printf("INFO:     <documentpath>  Path to the document to be parsed.\n");
   printf("INFO:     <licensepath>   Path to the license file to be used.\n");
}

// Helper function
int fileExists(const char * const szFileName) {
   struct stat buf;
   int exists;
   if (!szFileName)
      return 0;
   exists = stat(szFileName, &buf) == 0;
   return exists;
}

// Helper function
int checkargs(const int argc, char **argv) {
   if (argc != 5) {
      printf("FAIL: Program requires four arguments.\n");
      displayusageinfo();
      return 0;
   }
   printf("INFO: Grammar Path:  %s\n", argv[1]);
   printf("INFO: Entities:      %s\n", argv[2]);
   printf("INFO: Document Path: %s\n", argv[3]);
   printf("INFO: License Path:  %s\n", argv[4]);
   if (!fileExists(argv[1])) {
      printf("FAIL: Grammar path does not exist.\n");
      return 0;
   }
   if (!fileExists(argv[3])) {
      printf("FAIL: Document path does not exist.\n");
      return 0;
   }
   if (!fileExists(argv[4])) {
      printf("FAIL: License path does not exist.\n");
      return 0;
   }
   return 1;
}

// Helper function
char *readFile(const char * const fn) {
   struct stat fnInfo;
   off_t len;
   FILE *f;
   char *buf;
   size_t itemsRead;
   if (stat(fn, &fnInfo)) {
      printf("FAIL: Unable to get file size for \"%s\".\n", fn);
      return NULL;
   }
   len = fnInfo.st_size;
   if (!len) {
      printf("FAIL: Zero byte file size for \"%s\".\n", fn);
      return NULL;
   }
   f = fopen(fn, "rb");
   if (!f) {
      printf("FAIL: Unable to open file \"%s\".\n", fn);
      return NULL;
   }
   buf = (char*)malloc(len+1);
   itemsRead = fread(buf, 1, len, f);
   *(buf+len) = '\0';
   fclose(f);
   if (itemsRead < (size_t)len) {
      free(buf);
      printf("FAIL: Unable to read \"%s\".\n", fn);
      return NULL;
   }
   return buf;
}
```

```
// Helper function
int readFirst(const char * const fn, FILE **f, char * const buf, const size_t
bufLen, off_t * const fileSize, size_t * const bytesRead) {
   struct stat fnInfo;
   size_t itemsToRead, itemsRead;

   if (stat(fn, &fnInfo)) {
      printf("FAIL: Unable to get file size for \"%s\".\n", fn);
      return 0;
   }

   *fileSize = fnInfo.st_size;
   if (!*fileSize) {
      printf("FAIL: Zero byte file size for \"%s\".\n", fn);
      return 0;
   }

   *f = fopen(fn, "rb");
   if (!*f) {
      printf("FAIL: Unable to open file \"%s\".\n", fn);
      return 0;
   }

   itemsToRead = (off_t)bufLen < *fileSize ? bufLen :    (size_t)*fileSize;
   itemsRead = fread(buf, 1, itemsToRead, *f);
   if (itemsRead < itemsToRead) {
      fclose(*f);
      *f = NULL;
      printf("FAIL: Unable to read \"%s\".\n", fn);
      return 0;
   }
   *bytesRead = itemsRead;
   return 1;
}

// Helper function
int readNext(const char * const fn, FILE *f, char * const buf, const size_t bufLen,
const size_t bytesRemaining, size_t * const bytesRead) {
   size_t itemsToRead = bufLen < bytesRemaining ? bufLen :    bytesRemaining;
   size_t itemsRead;
   if (!itemsToRead) {
      *bytesRead = 0;
      return 1;
   }
   itemsRead = fread(buf, 1, itemsToRead, f);
   if (itemsRead < itemsToRead) {
      printf("FAIL: Unable to continue reading \"%s\".\n", fn);
      return 0;
   }
   *bytesRead = itemsRead;
   return 1;
}

// Main function
int main(int argc, char ** argv)
{
   int32_t nErrCode;
   EdkEngineHandle pEngine;
   char *license;
   const char* szErrorMsg;
   EdkSessionHandle pSession;
   size_t bytesRead, bytesRemaining;
   FILE *f;
   char buf[BUFLEN], componentText[128];
   off_t fileSize;
   const char *szEntityName, *szEntityText, *szOrigText;
   size_t textSize, textLength, origSize, origLength,    origOffset, offsetLength;
   double score;
   size_t nComponents, nComponent;

   printf("INFO: Program loaded.\n");

   if (!checkargs(argc, argv))
      return -1;
   printf("INFO: Parameters valid.\n");

   if (( nErrCode = EdkEngineCreate( &pEngine )) != EdkSuccess) {
      printf("Unable to create the EDK Engine. Error code: %d\n" , nErrCode);
      return -1;
   }
   printf("INFO: Engine created.\n");

   license = readFile(argv[4]);
   if (!license) {
      EdkEngineDestroy( pEngine );
      return -1;
   }
   printf("INFO: License read.\n");

   // Set the license to the eduction engine
   if ((EdkSetLicenseKey( pEngine, license) != EdkSuccess)) {
      nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
      printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
      free(license);
      EdkEngineDestroy(pEngine);
      return -1;
   }
   free(license);
   printf("INFO: License validated.\n");

   // Configure the eduction engine
   // Settings include:
   // EnableComponents
   // EnableUniqueMatches
   // MaxMatchLength
   // MaxMatchesPerDoc
   // MatchWholeWord
   // TokenWithPunctuation
   // AllowOverlpas
   // AllowMultipleResults
   // MatchCases
   // Locale
   EdkSetEnableComponents(pEngine, true);

   // Load resource file
   // Call this function repeatedly until all required resource    files are loaded
   if ((EdkLoadResourceFile(pEngine, argv[1]) != EdkSuccess)) {
      nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
      printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
      EdkEngineDestroy(pEngine);
      return -1;
   }
   printf("INFO: Resource file loaded.\n");

   // Add an entity to match to against
   // Call this function repeatedly to add all desired entities
   // The entities to be added must be defined in the resource    files added above
   if ((EdkAddTargetEntity(pEngine, argv[2]) != EdkSuccess)) {
      nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
      printf("FAIL: %s (%d)\n", szErrorMsg, nErrCode);
      EdkEngineDestroy(pEngine);
      return -1;
   }
   printf("INFO: Grammar(s) initialized.\n");

   // Create an eduction session associated with this eduction engine
   // Multiple sessions can be created and concurrent processing in multithreaded
applications
   if (EdkSessionCreate(pEngine, &pSession) != EdkSuccess) {
      nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
      printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
      EdkEngineDestroy(pEngine);
      return -1;
   }
   printf("INFO: Session created.\n");
   if (!readFirst(argv[3], &f, buf, BUFLEN, &fileSize, &bytesRead)) {
      EdkSessionDestroy(pSession);
      EdkEngineDestroy(pEngine);
      return -1;
   }
   bytesRemaining = (size_t)fileSize - bytesRead;
   printf("INFO: Data file opened and %d byte block reads    initiated.\n", 
BUFLEN);

   // Add input data
   // EdkAddInputText is called repeatedly for as many times as needed until all 
the input has been exhausted
   // The input data must be UTF-8 encoded.
   // Note: An alternative method of adding input data is to create a data input 
stream
   while (bytesRead) {
      printf("INFO: Adding data block to engine.\n");
      if ((EdkAddInputText( pSession, buf, bytesRead,       bytesRemaining ? false 
: true)) != EdkSuccess) {
         fclose(f);
         nErrCode = EdkGetLastSessionError(pSession, &szErrorMsg);
         printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
         EdkSessionDestroy(pSession);
         EdkEngineDestroy(pEngine);
         return -1;
      }
      printf("INFO: Data block added.\n");

      // Get a match
      // This is called repeated to get all matches
      // The test ensures that the loop is exited if no more match is required,
      //  found, or if an error occurs
      while (EdkGetNextMatch(pSession) == EdkSuccess) {
         // While we have a match, obtain all required information about the match
         EdkGetMatchEntityName(pSession, &szEntityName);
         EdkGetMatchOrigOffset(pSession, &origOffset);
         EdkGetMatchOrigOffsetLength(pSession, &offsetLength);
         EdkGetMatchScore(pSession, &score);
         EdkGetMatchTextSize(pSession, &textSize);
         EdkGetMatchTextLength(pSession, &textLength);
         EdkGetMatchOrigSize(pSession, &origSize);
         EdkGetMatchOrigLength(pSession, &origLength);
         EdkGetMatchOrigText(pSession, &szOrigText);
         EdkGetMatchText(pSession, &szEntityText);
         printf("INFO: EntityName=\"%s\" Offset=\"%u\" OffsetLength=\"%u\"\n", 
szEntityName, origOffset, offsetLength);
         printf("INFO:   Score=\"%04.2f\" NormalizedTextSize=\"%u\" 
NormalizedTextLength=\"%u\"\n", score, textSize, textLength);
         printf("INFO:   OriginalTextSize=\"%u\" OriginalTextLength=\"%u\"\n", 
origSize, origLength);
         printf("INFO:   Original Text=\"%s\"\n", szOrigText);
         printf("INFO:   Normalized Text=\"%s\"\n", szEntityText);
         EdkGetMatchComponentCount(pSession, &nComponents);
         for (nComponent = 0; nComponent < nComponents; ++nComponent) {
            EdkGetMatchComponentName(pSession, nComponent, &szEntityName);
            EdkGetMatchComponentOffset(pSession, nComponent, &origOffset);
            EdkGetMatchComponentOffsetLength(pSession, nComponent, &offsetLength);
            EdkGetMatchComponentSize(pSession, nComponent, &origSize);
            EdkGetMatchComponentLength(pSession, nComponent, &origLength);
            strncpy(componentText, szEntityText + origOffset, origSize);
            *(componentText + origSize) = '\0';
            printf( "INFO:     Component Name=\"%s\" Text=\"%s\"\n", szEntityName,
componentText);
            printf( "INFO:       Offset=\"%u\" OffsetLength=\"%u\" TextSize=\"%u\"
TextLength=\"%u\"\n", origOffset, offsetLength, origSize, origLength);
         }
      }
      printf("INFO: Matching on block complete.\n");

      nErrCode = EdkGetLastSessionError(pSession, &szErrorMsg);
      if ((nErrCode != EdkNoMatch)) {
         fclose(f);
         printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
         EdkSessionDestroy(pSession);
         EdkEngineDestroy(pEngine);
         return -1;
      }
      if(!readNext(argv[3], f, buf, BUFLEN, bytesRemaining, &bytesRead)) {
         fclose(f);
         EdkSessionDestroy(pSession);
         EdkEngineDestroy(pEngine);
         return -1;
      }
      bytesRemaining -= bytesRead;
   }
   fclose(f);

   // Destroy the session handle and release the resource
   EdkSessionDestroy(pSession);

   // Ensure that all session handles have been destroyed before calling this
   EdkEngineDestroy(pEngine);

   printf("PASS: Program completed without an error.\n");
   return 0;
}
```

cjknormalization.c

The cjknormalization.c program builds upon basic.c and shows how you can perform normalization of Chinese, Japanese, and Korean input text before you process it in an eduction session.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <malloc.h>
#include <edk.h>
#include <string.h>

#ifdef _WIN32
#define stat _stat
#define off_t _off_t
#endif // _WIN32
#define BUFLEN 5120

// Helper function
void displayusageinfo() {
    EDK_VERSION_INFO versionInfo;
    EdkGetVersion(&versionInfo);
    if (versionInfo.vChangeSet)
        printf("INFO: Eduction SDK Sample for SDK version v%s.%i\n", 
versionInfo.versionString, versionInfo.vChangeSet);
    else
        printf("INFO: Eduction SDK Sample for SDK version v%s\n", 
versionInfo.versionString);
    printf("INFO: SDK Built: %s\n", versionInfo.buildTime);
    printf("INFO: Copyright %s\n", versionInfo.copyright);
    printf("INFO: Usage: cjknormalization.exe <grammarpath> <documentpath> 
<licensepath>\n");
    printf("INFO: Parameters:\n");
    printf("INFO:     <grammarpath>   Path to the grammar file to be used.\n");
    printf("INFO:     <documentpath>  Path to the document to be parsed.\n");
    printf("INFO:     <licensepath>   Path to the license file to be used.\n");
}

// Helper function
int fileExists(const char * const szFileName) {
    struct stat buf;
    int exists;
    if (!szFileName)
        return 0;
    exists = stat(szFileName, &buf) == 0;
    return exists;
}
```

```
// Helper function
int checkargs(const int argc, char **argv) {
    if (argc != 4) {
        printf("FAIL: Program requires four arguments.\n");
        displayusageinfo();
        return 0;
    }
    printf("INFO: Grammar Path:  %s\n", argv[1]);
    printf("INFO: Document Path: %s\n", argv[2]);
    printf("INFO: License Path:  %s\n", argv[3]);
    if (!fileExists(argv[1])) {
        printf("FAIL: Grammar path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[2])) {
        printf("FAIL: Document path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[3])) {
        printf("FAIL: License path does not exist.\n");
        return 0;
    }
    return 1;
}

// Helper function
char *readFile(const char * const fn) {
    struct stat fnInfo;
    off_t len;
    FILE *f;
    char *buf;
    size_t itemsRead;
    if (stat(fn, &fnInfo)) {
        printf("FAIL: Unable to get file size for \"%s\".\n", fn);
        return NULL;
    }
    len = fnInfo.st_size;
    if (!len) {
        printf("FAIL: Zero byte file size for \"%s\".\n", fn);
        return NULL;
    }
    f = fopen(fn, "rb");
    if (!f) {
        printf("FAIL: Unable to open file \"%s\".\n", fn);
        return NULL;
    }
    buf = (char*)malloc(len+1);
    itemsRead = fread(buf, 1, len, f);
    *(buf+len) = '\0';
    fclose(f);
    if (itemsRead < (size_t)len) {
        free(buf);
        printf("FAIL: Unable to read \"%s\".\n", fn);
        return NULL;
    }
    return buf;
}

// Main function
int main(int argc, char ** argv)
{
    int32_t nErrCode;
    EdkEngineHandle pEngine;
    char *license;
    const char* szErrorMsg;
    EdkSessionHandle pSession;
    char* buf = NULL;
    off_t fileSize;
    const char *szEntityName, *szEntityText, *szOrigText;
    size_t textSize, textLength, origSize, origLength, origOffset, offsetLength;
    double score;
    const char* szCJKNormalizedText = NULL;
    const char* szCJKNormalizationOptions = "HWNum,HWAlpha";

    printf("INFO: Program loaded.\n");

    if (!checkargs(argc, argv))
        return -1;
    printf("INFO: Parameters valid.\n");

    if (( nErrCode = EdkEngineCreate( &pEngine )) != EdkSuccess) {
        printf("Unable to create the EDK Engine. Error code: %d\n" , nErrCode);
        return -1;
    }
     printf("INFO: Engine created.\n");

    license = readFile(argv[3]);
    if (!license) {
        EdkEngineDestroy( pEngine );
        return -1;
    }
    printf("INFO: License read.\n");

    // Set the license to the eduction engine
    if ((EdkSetLicenseKey( pEngine, license) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        free(license);
        EdkEngineDestroy(pEngine);
          return -1;
    }
    free(license);
    printf("INFO: License validated.\n");

    // Load resource file
    // Call this function repeatedly until all required resource files are loaded
    if ((EdkLoadResourceFile(pEngine, argv[1]) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Resource file loaded.\n");

    // Add an entity to match to against
    // Call this function repeatedly to add all desired entities
    // The entities to be added must be defined in the resource files added above
    if ((EdkAddTargetEntity(pEngine, "e/e") != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n", szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Grammar(s) initialized.\n");

    // Create an eduction session assosicated with this eduction engine
    // Multiple sessions can be created and concurrent processing in multithreaded
applications
    if (EdkSessionCreate(pEngine, &pSession) != EdkSuccess) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Session created.\n");

    buf = readFile(argv[2]);
    if (!buf) {
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Data file opened and %d byte block reads initiated.\n", BUFLEN);

    if (EdkCJKNormalizeText(pEngine, buf, &szCJKNormalizedText,
szCJKNormalizationOptions) != EdkSuccess) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Input buffer normalized.\n");

    // Add input data
    // EdkAddInputText is called repeatedly for as many times as needed until all
the input has been exhausted
    // The input data must be UTF-8 encoded.
    // Note: An alternative method of adding input data is to create a data input
stream
    printf("INFO: Adding data block to engine.\n");
    if ((EdkAddInputText( pSession, szCJKNormalizedText, strlen
(szCJKNormalizedText), true)) != EdkSuccess) {
        nErrCode = EdkGetLastSessionError(pSession, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Data block added.\n");

    // Get a match
    // This is called repeatedly to get all matches
    while (EdkGetNextMatch(pSession) == EdkSuccess) {
        // While we have a match, obtain all required information about the match
        EdkGetMatchEntityName(pSession, &szEntityName);
        EdkGetMatchOrigOffset(pSession, &origOffset);
        EdkGetMatchOrigOffsetLength(pSession, &offsetLength);
        EdkGetMatchScore(pSession, &score);
        EdkGetMatchTextSize(pSession, &textSize);
        EdkGetMatchTextLength(pSession, &textLength);
        EdkGetMatchOrigSize(pSession, &origSize);
        EdkGetMatchOrigLength(pSession, &origLength);
        EdkGetMatchOrigText(pSession, &szOrigText);
        EdkGetMatchText(pSession, &szEntityText);
        printf("INFO: EntityName=\"%s\" Offset=\"%u\" OffsetLength=\"%u\"\n",
szEntityName, origOffset, offsetLength);
        printf("INFO:   Score=\"%04.2f\" NormalizedTextSize=\"%u\"
NormalizedTextLength=\"%u\"\n", score, textSize, textLength);
        printf("INFO:   OriginalTextSize=\"%u\" OriginalTextLength=\"%u\"\n",
origSize, origLength);
        printf("INFO:   Original Text=\"%s\"\n", szOrigText);
        printf("INFO:   Normalized Text=\"%s\"\n", szEntityText);
    }
    printf("INFO: Matching on block complete.\n");
    nErrCode = EdkGetLastSessionError(pSession, &szErrorMsg);
    if ((nErrCode != EdkNoMatch)) {
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }

    // Destroy the session handle and release the resource
    EdkSessionDestroy(pSession);

    // Ensure that all session handles have been destroyed before calling this
    EdkEngineDestroy(pEngine);

    printf("PASS: Program completed without an error.\n");
    return 0;
}
```

postprocess.c

The postprocess.c program builds upon basic.c and shows the work flow needed to support post-processing in eduction. The program:

1. creates and configures an eduction engine.
2. creates post-processing tasks.
3. creates an eduction session to process the input.
4. adds input text to the session.
5. performs the extraction.
6. collects the matches from the extraction.
7. runs post-processing tasks on the matches.
8. prints the results.
9. cleans up the post-processing tasks.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <malloc.h>
#include <edk.h>
#include <string.h>

#ifdef _WIN32
#define stat _stat
#define off_t _off_t
#endif // _WIN32
#define BUFLEN 5120
```

```
// Helper function
void displayusageinfo() {
    EDK_VERSION_INFO versionInfo;
    EdkGetVersion(&versionInfo);
    if (versionInfo.vChangeSet)
        printf("INFO: Eduction SDK Sample for SDK version v%s.%i\n",
versionInfo.versionString, versionInfo.vChangeSet);
    else
        printf("INFO: Eduction SDK Sample for SDK version v%s\n",
versionInfo.versionString);
    printf("INFO: SDK Built: %s\n", versionInfo.buildTime);
    printf("INFO: Copyright %s\n", versionInfo.copyright);
    printf("INFO: Usage: sample1 <grammarpath> <entity> <documentpath>
<licensepath>\n");
    printf("INFO: Parameters:\n");
    printf("INFO:     <grammarpath>   Path to the grammar file that defines
matchable\n");
    printf("INFO:                     entities. The grammar file can be in
uncompiled (XML)\n");
    printf("INFO:                     or compiled (ECR) format.\n");
    printf("INFO:     <entities>      Comma deliminted list of entities in the
grammar file\n");
    printf("INFO                      to be used for matching.\n");
    printf("INFO:     <documentpath>  Path to the document to be parsed.\n");
    printf("INFO:     <licensepath>   Path to the license file to be used.\n");
}

// Helper function
int fileExists(const char * const szFileName) {
    struct stat buf;
    int exists;
    if (!szFileName)
        return 0;
    exists = stat(szFileName, &buf) == 0;
    return exists;
}

// Helper function
int checkargs(const int argc, char **argv) {
    if (argc != 5) {
        printf("FAIL: Program requires four arguments.\n");
        displayusageinfo();
        return 0;
    }
    printf("INFO: Grammar Path:  %s\n", argv[1]);
    printf("INFO: Entities:      %s\n", argv[2]);
    printf("INFO: Document Path: %s\n", argv[3]);
    printf("INFO: License Path:  %s\n", argv[4]);
    if (!fileExists(argv[1])) {
        printf("FAIL: Grammar path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[3])) {
        printf("FAIL: Document path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[4])) {
        printf("FAIL: License path does not exist.\n");
        return 0;
    }
    return 1;
}

// Helper function
char *readFile(const char * const fn) {
    struct stat fnInfo;
    off_t len;
    FILE *f;
    char *buf;
    size_t itemsRead;
    if (stat(fn, &fnInfo)) {
        printf("FAIL: Unable to get file size for \"%s\".\n", fn);
        return NULL;
    }
    len = fnInfo.st_size;
    if (!len) {
        printf("FAIL: Zero byte file size for \"%s\".\n", fn);
        return NULL;
    }
    f = fopen(fn, "rb");
    if (!f) {
        printf("FAIL: Unable to open file \"%s\".\n", fn);
        return NULL;
    }
    buf = (char*)malloc(len+1);
    itemsRead = fread(buf, 1, len, f);
    *(buf+len) = '\0';
    fclose(f);
    if (itemsRead < (size_t)len) {
        free(buf);
        printf("FAIL: Unable to read \"%s\".\n", fn);
        return NULL;
    }
    return buf;
}

// Helper function
```

```
int readFirst(const char * const fn, FILE **f, char * const buf, const size_t
bufLen, off_t * const fileSize, size_t * const bytesRead) {
    struct stat fnInfo;
    size_t itemsToRead, itemsRead;

    if (stat(fn, &fnInfo)) {
        printf("FAIL: Unable to get file size for \"%s\".\n", fn);
        return 0;
    }

    *fileSize = fnInfo.st_size;
    if (!*fileSize) {
        printf("FAIL: Zero byte file size for \"%s\".\n", fn);
        return 0;
    }

    *f = fopen(fn, "rb");
    if (!*f) {
        printf("FAIL: Unable to open file \"%s\".\n", fn);
        return 0;
    }

    itemsToRead = (off_t)bufLen < *fileSize ? bufLen : (size_t)*fileSize;
    itemsRead = fread(buf, 1, itemsToRead, *f);
    if (itemsRead < itemsToRead) {
        fclose(*f);
        *f = NULL;
        printf("FAIL: Unable to read \"%s\".\n", fn);
        return 0;
    }
    *bytesRead = itemsRead;
    return 1;
}

// Helper function
int readNext(const char * const fn, FILE *f, char * const buf, const size_t bufLen,
const size_t bytesRemaining, size_t * const bytesRead) {
    size_t itemsToRead = bufLen < bytesRemaining ? bufLen : bytesRemaining;
    size_t itemsRead;
    if (!itemsToRead) {
        *bytesRead = 0;
        return 1;
    }
    itemsRead = fread(buf, 1, itemsToRead, f);
    if (itemsRead < itemsToRead) {
        printf("FAIL: Unable to continue reading \"%s\".\n", fn);
        return 0;
    }
    *bytesRead = itemsRead;
    return 1;
}

// Main function
int main(int argc, char ** argv)
{
    int32_t nErrCode;
    EdkEngineHandle pEngine;
    char *license;
    const char* szErrorMsg;
    EdkSessionHandle pSession;
    size_t bytesRead, bytesRemaining;
    FILE *f;
    char buf[BUFLEN], componentText[128];
    off_t fileSize;
    const char *szEntityName, *szEntityText, *szOrigText;
    size_t textSize, textLength, origSize, origLength, origOffset, offsetLength;
    double score;
    size_t nComponents, nComponent;
    EdkPostProcessTaskHandle pTask = NULL;
    EdkPostProcessTasksCollectionHandle pTaskSet = NULL;
    EdkPostProcessorHandle pProcessor = NULL;
    EdkMatchesCollectionHandle pMatchSet = NULL;
    EdkMatchHandle pMatch = NULL;
    size_t nMatches = 0;
    size_t nIndex = 0;

    printf("INFO: Program loaded.\n");

    if (!checkargs(argc, argv))
        return -1;
    printf("INFO: Parameters valid.\n");

    if (( nErrCode = EdkEngineCreate( &pEngine )) != EdkSuccess) {
        printf("Unable to create the EDK Engine. Error code: %d\n" , nErrCode);
        return -1;
    }
    printf("INFO: Engine created.\n");

    license = readFile(argv[4]);
    if (!license) {
        EdkEngineDestroy( pEngine );
        return -1;
    }
    printf("INFO: License read.\n");

    // Set the license to the eduction engine
    if ((EdkSetLicenseKey( pEngine, license) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        free(license);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    free(license);
    printf("INFO: License validated.\n");

    // Configure the eduction engine
    // Settings include:
    //  EnableComponents
    //  EnableUniqueMatches
    //  MaxMatchLength
    //  MaxMatchesPerDoc
    //  MatchWholeWord
    //  TokenWithPunctuation
    //  AllowOverlaps
    //  AllowMultipleResults
    //  MatchCases
    //  Locale
    EdkSetEnableComponents(pEngine, true);

    // Load resource file
    // Call this function repeatedly until all required resource files are loaded
    if ((EdkLoadResourceFile(pEngine, argv[1]) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Resource file loaded.\n");

    // Add an entity to match to against
    // Call this function repeatedly to add all desired entities
    // The entities to be added must be defined in the resource files added above
    if ((EdkAddTargetEntity(pEngine, argv[2]) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n", szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Grammar(s) initialized.\n");

    /* Set up post processing stuff */
    if ((nErrCode = EdkPostProcessorTaskCreate("test_task", "scripts/turing.lua",
argv[2], false, &pTask)) != EdkSuccess)
    {
        printf("Unable to create post processing task. Error code: %d\n" ,
nErrCode);
```

```
        EdkEngineDestroy(pEngine);
        return -1;
    }

    printf("INFO: Post process task test_task, with script scripts/turing.lua,
created.\n");

    if ((nErrCode = EdkPostProcessorTasksCollectionCreate(&pTaskSet)) !=
EdkSuccess)
    {
        printf("Unable to create post processing task collection. Error code: %d\
     n" , nErrCode);
        EdkPostProcessorTaskDestroy(pTask);
        EdkEngineDestroy(pEngine);
        return -1;
    }

    if ((nErrCode = EdkPostProcessorTasksCollectionAddTask(pTaskSet, pTask)) !=
EdkSuccess)
    {
        printf("Unable to add post processing task to collection. Error code: %d\
     n" , nErrCode);
        EdkPostProcessorTaskDestroy(pTask);
        EdkPostProcessorTasksCollectionDestroy(pTaskSet);
        EdkEngineDestroy(pEngine);
        return -1;
    }

    printf("INFO: Post process task list created.\n");

    if ((nErrCode = EdkPostProcessorCreate(pTaskSet, &pProcessor)) != EdkSuccess)
    {
        printf("Unable to add post processing task to collection. Error code: %d\
     n" , nErrCode);
        EdkPostProcessorTaskDestroy(pTask);
        EdkPostProcessorTasksCollectionDestroy(pTaskSet);
        EdkEngineDestroy(pEngine);
        return -1;
    }

    printf("INFO: Post processor object initialized.\n");

    // Create an eduction session associated with this eduction engine
    // Multiple sessions can be created and concurrent processing in multithreaded
   applications
    if (EdkSessionCreate(pEngine, &pSession) != EdkSuccess) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Session created.\n");

    if (!readFirst(argv[3], &f, buf, BUFLEN, &fileSize, &bytesRead)) {
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    bytesRemaining = (size_t)fileSize - bytesRead;
    printf("INFO: Data file opened and %d byte block reads initiated.\n", BUFLEN);

    // Add input data
    // EdkAddInputText is called repeatedly for as many times as needed until all
the input has been exhausted
    // The input data must be UTF-8 encoded.
    // Note: An alternative method of adding input data is to create a data input
stream
    while (bytesRead)
    {
        printf("INFO: Adding data block to engine.\n");
        if ((EdkAddInputText( pSession, buf, bytesRead, bytesRemaining ? false :
     true)) != EdkSuccess)
        {
            fclose(f);
            nErrCode = EdkGetLastSessionError(pSession, &szErrorMsg);
            printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
            EdkSessionDestroy(pSession);
            EdkEngineDestroy(pEngine);
            return -1;
        }
        printf("INFO: Data block added.\n");

        if(!readNext(argv[3], f, buf, BUFLEN, bytesRemaining, &bytesRead))
        {
            fclose(f);
            EdkSessionDestroy(pSession);
            EdkEngineDestroy(pEngine);
            return -1;
        }
        bytesRemaining -= bytesRead;
    }

    fclose(f);

    /* Get all matches found from the input and process them */
    if ((nErrCode = EdkFillMatches(pSession, &pMatchSet)) != EdkSuccess)
    {
        printf("Unable to retrieve matches from current session. Error code: %d\
    n" , nErrCode);
        EdkPostProcessorTaskDestroy(pTask);
        EdkPostProcessorTasksCollectionDestroy(pTaskSet);
        EdkEngineDestroy(pEngine);
        return -1;
    }

    printf("INFO: Running post-processor on match set.\n");
    EdkPostProcessorRun(pProcessor, pMatchSet);
    EdkGetNumMatches(pMatchSet, &nMatches);
    printf("INFO: Post processing complete.\n");
    printf(nMatches == 1 ? "PASS: " : "FAIL: ");
    printf("Got expected number of matches.\n");

    for (nIndex = 0; nIndex < nMatches; nIndex++)
    {
        EdkRetrieveMatch(pMatchSet, nIndex, &pMatch);
        /* print out match info using match info accessors */
        EdkMatchGetEntityName(pMatch, &szEntityName);
        EdkMatchGetMatchedTextOffset(pMatch, &origOffset);
        EdkMatchGetMatchedTextOffsetLength(pMatch, &offsetLength);
        EdkMatchGetScore(pMatch, &score);
        EdkMatchGetNormalizedTextSize(pMatch, &textSize);
        EdkMatchGetNormalizedTextLength(pMatch, &textLength);
        EdkMatchGetMatchedTextSize(pMatch, &origSize);
        EdkMatchGetMatchedTextLength(pMatch, &origLength);
        EdkMatchGetMatchedText(pMatch, &szOrigText);
        EdkMatchGetNormalizedText(pMatch, &szEntityText);
        printf("INFO:   EntityName=\"%s\" Offset=\"%u\" OffsetLength=\"%u\"\n",
    szEntityName, origOffset, offsetLength);
        printf("INFO:   Score=\"%04.2f\" NormalizedTextSize=\"%u\"
NormalizedTextLength=\"%u\"\n", score, textSize, textLength);
        printf("INFO:   OriginalTextSize=\"%u\" OriginalTextLength=\"%u\"\n",
  origSize, origLength);
        printf("INFO:   Original Text=\"%s\"\n", szOrigText);
        printf("INFO:   Normalized Text=\"%s\"\n", szEntityText);
        EdkMatchGetComponentCount(pMatch, &nComponents);
        for (nComponent = 0; nComponent < nComponents; ++nComponent)
        {
            EdkMatchComponentHandle pComponent = NULL;
            EdkMatchGetComponentHandle(pMatch, nComponent, &pComponent);

            EdkMatchComponentGetName(pComponent, &szEntityName);
            EdkMatchComponentGetMatchedTextOffset(pComponent, &origOffset);
            EdkMatchComponentGetMatchedTextLength(pComponent, &offsetLength);
            EdkMatchComponentGetSize(pComponent, &origSize);
            EdkMatchComponentGetLength(pMatch, pComponent, &origLength);
            strncpy(componentText, szEntityText + origOffset, origSize);
            *(componentText + origSize) = '\0';
            printf( "INFO:     Component Name=\"%s\" Text=\"%s\"\n",
szEntityName, componentText);
            printf( "INFO:     Offset=\"%u\" OffsetLength=\"%u\" TextSize=\"%u\"
         TextLength=\"%u\"\n", origOffset, offsetLength, origSize,
origLength);
        }
        printf("INFO: Post processing complete.\n");
        printf(score == 5.00 ? "PASS: " : "FAIL: ");
        printf("Got expected score for match.\n");
        printf(!strcmp("Alan Turing", szOrigText) ? "PASS: " : "FAIL: ");
        printf("Got expected text for match.\n");
    }

    // Destroy the post-processing things, plus the match set
    EdkPostProcessorTaskDestroy(pTask);
    EdkPostProcessorTasksCollectionDestroy(pTaskSet);
    EdkPostProcessorDestroy(pProcessor);
    EdkDestroyMatches(pMatchSet);

    // Destroy the session handle and release the resource
    EdkSessionDestroy(pSession);

    // Ensure that all session handles have been destroyed before calling this
    EdkEngineDestroy(pEngine);

    printf("PASS: Program completed without an error.\n");
    return 0;
}
```

redaction.c

The redaction.c program demonstrates the use of the `EdkGetRedactedText` function. It shows how to set up an eduction engine with the specified grammars and entities, and how you can use that engine to produce redacted output when given an input buffer.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <malloc.h>
#include <edk.h>
#include <string.h>

#ifdef _WIN32
#define stat _stat
#define off_t _off_t
#endif // _WIN32
#define BUFLEN 5120
```

```
// Helper function
void displayusageinfo() {
    EDK_VERSION_INFO versionInfo;
    EdkGetVersion(&versionInfo);
    if (versionInfo.vChangeSet)
        printf("INFO: Eduction SDK Sample for SDK version v%s.%i\n",
versionInfo.versionString, versionInfo.vChangeSet);
    else
        printf("INFO: Eduction SDK Sample for SDK version v%s\n",
versionInfo.versionString);
    printf("INFO: SDK Built: %s\n", versionInfo.buildTime);
    printf("INFO: Copyright %s\n", versionInfo.copyright);
    printf("INFO: Usage: sample1 <grammarpath> <entity> <documentpath>
<licensepath>\n");
    printf("INFO: Parameters:\n");
    printf("INFO:     <grammarpath>   Path to the grammar file that defines
matchable\n");
    printf("INFO:                     entities. The grammar file can be in
uncompiled (XML)\n");
    printf("INFO:                     or compiled (ECR) format.\n");
    printf("INFO:     <entities>      Comma deliminted list of entities in the
grammar file\n");
    printf("INFO:                     to be used for matching.\n");
    printf("INFO:     <documentpath>  Path to the document to be parsed.\n");
    printf("INFO:     <licensepath>   Path to the license file to be used.\n");
}

// Helper function
int fileExists(const char * const szFileName) {
    struct stat buf;
    int exists;
    if (!szFileName)
        return 0;
    exists = stat(szFileName, &buf) == 0;
    return exists;
}

// Helper function
int checkargs(const int argc, char **argv) {
    if (argc != 5) {
        printf("FAIL: Program requires four arguments.\n");
        displayusageinfo();
        return 0;
    }
    printf("INFO: Grammar Path:  %s\n", argv[1]);
    printf("INFO: Entities:      %s\n", argv[2]);
    printf("INFO: Document Path: %s\n", argv[3]);
    printf("INFO: License Path:  %s\n", argv[4]);
    if (!fileExists(argv[1])) {
        printf("FAIL: Grammar path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[3])) {
        printf("FAIL: Document path does not exist.\n");
        return 0;
    }
    if (!fileExists(argv[4])) {
        printf("FAIL: License path does not exist.\n");
        return 0;
    }
    return 1;
}

// Helper function
char *readFile(const char * const fn) {
    struct stat fnInfo;
    off_t len;
    FILE *f;
    char *buf;
    size_t itemsRead;
    if (stat(fn, &fnInfo)) {
        printf("FAIL: Unable to get file size for \"%s\".\n", fn);
        return NULL;
    }
    len = fnInfo.st_size;
    if (!len) {
        printf("FAIL: Zero byte file size for \"%s\".\n", fn);
        return NULL;
    }
    f = fopen(fn, "rb");
    if (!f) {
        printf("FAIL: Unable to open file \"%s\".\n", fn);
        return NULL;
    }
    buf = (char*)malloc(len+1);
    itemsRead = fread(buf, 1, len, f);
    *(buf+len) = '\0';
    fclose(f);
    if (itemsRead < (size_t)len) {
        free(buf);
        printf("FAIL: Unable to read \"%s\".\n", fn);
        return NULL;
    }
    return buf;
}

// Helper function
int readFirst(const char * const fn, FILE **f, char * const buf, const size_t
bufLen, off_t * const fileSize, size_t * const bytesRead) {
    struct stat fnInfo;
    size_t itemsToRead, itemsRead;

    if (stat(fn, &fnInfo)) {
        printf("FAIL: Unable to get file size for \"%s\".\n", fn);
        return 0;
    }

    *fileSize = fnInfo.st_size;
    if (!*fileSize) {
        printf("FAIL: Zero byte file size for \"%s\".\n", fn);
        return 0;
    }

    *f = fopen(fn, "rb");
    if (!*f) {
        printf("FAIL: Unable to open file \"%s\".\n", fn);
        return 0;
    }

    itemsToRead = (off_t)bufLen < *fileSize ? bufLen : (size_t)*fileSize;
    itemsRead = fread(buf, 1, itemsToRead, *f);
    if (itemsRead < itemsToRead) {
        fclose(*f);
        *f = NULL;
        printf("FAIL: Unable to read \"%s\".\n", fn);
        return 0;
    }
    *bytesRead = itemsRead;
    return 1;
}

// Helper function
int readNext(const char * const fn, FILE *f, char * const buf, const size_t bufLen,
const size_t bytesRemaining, size_t * const bytesRead) {
    size_t itemsToRead = bufLen < bytesRemaining ? bufLen : bytesRemaining;
    size_t itemsRead;
    if (!itemsToRead) {
        *bytesRead = 0;
        return 1;
    }
    itemsRead = fread(buf, 1, itemsToRead, f);
    if (itemsRead < itemsToRead) {
        printf("FAIL: Unable to continue reading \"%s\".\n", fn);
        return 0;
    }
    *bytesRead = itemsRead;
    return 1;
```

```
}

// Main function
int main(int argc, char ** argv)
{
    int32_t nErrCode;
    EdkEngineHandle pEngine;
    char *license;
    const char* szErrorMsg;
    EdkSessionHandle pSession;
    size_t bytesRead, bytesRemaining;
    FILE *f;
    char buf[BUFLEN], componentText[128];
    off_t fileSize;
    const char *szEntityName, *szEntityText, *szOrigText;
    size_t textSize, textLength, origSize, origLength, origOffset, offsetLength;
    double score;
    size_t nComponents, nComponent;
    const char* output;

    printf("INFO: Program loaded.\n");

    if (!checkargs(argc, argv))
        return -1;
    printf("INFO: Parameters valid.\n");

    if (( nErrCode = EdkEngineCreate( &pEngine )) != EdkSuccess) {
        printf("Unable to create the EDK Engine. Error code: %d\n" , nErrCode);
        return -1;
    }
    printf("INFO: Engine created.\n");

    license = readFile(argv[4]);
    if (!license) {
        EdkEngineDestroy( pEngine );
        return -1;
    }
    printf("INFO: License read.\n");

    // Set the license to the eduction engine
    if ((EdkSetLicenseKey( pEngine, license) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        free(license);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    free(license);
    printf("INFO: License validated.\n");
    // Configure the eduction engine
    // Settings include:
    //  EnableComponents
    //  EnableUniqueMatches
    //  MaxMatchLength
    //  MaxMatchesPerDoc
    //  MatchWholeWord
    //  TokenWithPunctuation
    //  AllowOverlaps
    //  AllowMultipleResults
    //  MatchCases
    //  Locale
    EdkSetEnableComponents(pEngine, true);

    // Load resource file
    // Call this function repeatedly until all required resource files are loaded
    if ((EdkLoadResourceFile(pEngine, argv[1]) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n" , szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Resource file loaded.\n");

    // Add an entity to match to against
    // Call this function repeatedly to add all desired entities
    // The entities to be added must be defined in the resource files added above
    if ((EdkAddTargetEntity(pEngine, argv[2]) != EdkSuccess)) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n", szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    printf("INFO: Grammar(s) initialized.\n");

    if (!readFirst(argv[3], &f, buf, BUFLEN, &fileSize, &bytesRead)) {
        EdkSessionDestroy(pSession);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    bytesRemaining = (size_t)fileSize - bytesRead;
    printf("INFO: Data file opened and %d byte block reads initiated.\n", BUFLEN);

    if(EdkGetRedactedText(pEngine, buf, &output) != EdkSuccess) {
        nErrCode = EdkGetLastEngineError(pEngine, &szErrorMsg);
        printf("FAIL: %s (%d)\n", szErrorMsg, nErrCode);
        EdkEngineDestroy(pEngine);
        return -1;
    }
    fclose(f);

    printf("INFO: Redacted text: %s\n", output);

    // Ensure that all session handles have been destroyed before calling this
    EdkEngineDestroy(pEngine);

    printf("PASS: Program completed without an error.\n");
    return 0;
}
```

multithread.c

The multithread.c example program is for use on the Linux platform.

This sample program demonstrates the use of multiple eduction sessions associated with a single eduction engine running in parallel.

This program takes arguments of the names of the text files to process. The program has the following stages:

1. The program creates and configures an eduction engine.
2. The program creates a worker thread for each text file.
3. The worker thread starts with the eduction engine and the text file. This worker thread:
    - creates a session associated with the eduction engine.
    - adds text from the text file to the session.
    - performs an entity extraction of the text.
4. The main program then waits for each thread to finish extraction, exit, and join.

```
#include <stdlib.h>
#include <stdio.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <malloc.h>
#include <string.h>
#include <pthread.h>
#include <edk.h>

/*
 * Create an app that uses multiple sessions in parallel
 * Number of sessions depends on the number of input files
 */

// Data to pass to each worker thread
```

```
typedef struct thread_data
{
   char* file;
      EdkEngineHandle engine;
} t_thread_data;

// The worker thread function
// Creates a session using the engine handle
// Processes the the input file
static void* WorkProc(void* pvArgs)
{
   t_thread_data* p = (t_thread_data*)pvArgs;
   EdkEngineHandle engine = p->engine;
   char* file = p->file;
   size_t bytesRead;
   char* buf;
   struct stat fnInfo;

   if (stat(file, &fnInfo)) {
      printf("Unable to stat input file.\n");
      return NULL;
   }
   size_t fileSize = fnInfo.st_size;
   if (!fileSize)
   {
      printf("File size of input file is zero!\n");
      return NULL;
   }

   FILE *f;
   f = fopen(file, "rb");
   if (!f)
   {
      printf("Can't open input file!\n");
      return NULL;
   }

   // Read input file to buffer
   // Note that the input must be UTF-8 encoded.
   buf = (char*)malloc(fileSize * sizeof(char));
   bytesRead = fread(buf, 1, fileSize, f);
   if (bytesRead < fileSize) {
      fclose(f);
      f = NULL;
      printf("Unable to read input file!\n");
      return NULL;
   }

   // Create a session and process the input file
   EdkError error;
   EdkSessionHandle session = NULL;
   if (EdkSuccess != (error = EdkSessionCreate(engine, &session)))
   {
      printf("Error: Unable to create session, error code = %d\n", error);
      return NULL;
   }
   if (EdkSuccess != (error = EdkAddInputText(session, buf, bytesRead, true)))
   {
      printf("Unable to add input text, error code = %d\n", error);
      return NULL;
   }
   else
   {
      while(EdkSuccess == EdkGetNextMatch(session))
      {
         // For each match found, do this ...
         const char* szMatch = NULL;
         EdkGetMatchText(session, &szMatch);
         printf("Match found: %s\n", szMatch);
      }
   }

   // Destroy the session and exit when done
   EdkSessionDestroy(session);
   session = NULL;

   free(buf);
   return NULL;
}

// Main function
// Reads arguments from command line as input files
// Creates one thread to process each input file
int main(int argc, char **argv)
{
   char *license;
   struct stat fnLicense;
   off_t len;
   FILE *f;
   size_t itemsRead;

   // Input files are specified at the command line
   // One thread will be created to process every input file
   if (argc <= 1) {
      printf("No input file specified.\n");
      return -1;
   }
```

```
// Licensing
if (stat("licensekey.dat", &fnLicense)) {
   printf("Unable to stat license file.\n");
   return -1;
}
len = fnLicense.st_size;
if (!len) {
   printf("Zero byte file size for license file.\n");
   return -1;
}
f = fopen("licensekey.dat", "rb");
if (!f) {
   printf("Unable to open license file.\n");
   return -1;
}
license = (char*)malloc(len+1);
itemsRead = fread(license, 1, len, f);
*(license+len) = '\0';
fclose(f);
if (itemsRead < (size_t)len) {
   free(license);
   printf("Unable to read license.\n");
   return -1;
}
if (!license) {
   return -1;
}
printf("INFO: License read.\n");

// Create the eduction engine
EdkError error;
EdkEngineHandle engine = NULL;
if (EdkSuccess != (error = EdkEngineCreate(&engine)))
{
   printf("Error: Can't create engine, error code = %d.\n", error);
   return -1;
}

if (EdkSuccess != (error = EdkSetLicenseKey(engine, license))) {
   printf("Error: Can't set license key, error code = %d.\n", error);
   free(license);
   EdkEngineDestroy(engine);
   return -1;
}
printf("INFO: License validated.\n");
free(license);

// Load the resource file - change if necessary
if (EdkSuccess != (error = EdkLoadResourceFile(engine, "company_eng.ecr")))
```

```
{
   printf("Error: Can't load resource file, error code = %d.\n", error);
   EdkEngineDestroy(engine);
   return -1;
}
printf("INFO: Resource file loaded.\n");

// Add target entity - change if necessary
if (EdkSuccess != (error = EdkAddTargetEntity(engine, "company/name/eng")))
{
   printf("Error: Can't add entity, error cdoe = %d.\n", error);
   EdkEngineDestroy(engine);
   return -1;
}
printf("INFO: Entity added.\n");

// Prepare the worker threads
size_t         nThreads = argc - 1;
pthread_t*     threadId     = (pthread_t*)calloc(nThreads, sizeof(pthread_t));
void* tmp = calloc(nThreads, sizeof(t_thread_data));
t_thread_data* threadData = (t_thread_data*)tmp;
int ii = 0;
pthread_attr_t attr;
pthread_attr_init(&attr);
pthread_attr_setstacksize(&attr, 128000);
pthread_attr_setdetachstate(&attr, PTHREAD_CREATE_JOINABLE);

// Start the threads
// Each thread uses a session to process an input file
for(ii=0;ii<nThreads;ii++)
{
   threadData[ii].engine = engine;
   threadData[ii].file = argv[ii + 1];
   pthread_create(&threadId[ii], &attr, WorkProc, (void*)&threadData[ii]);
}

// Join the threads when done
for (ii = 0; ii < nThreads; ii++)
{
   if (threadId[ii])
   {
      void *pvres;
      pthread_join(threadId[ii], &pvres);
   }
}

free(threadId);
free(threadData);
EdkEngineDestroy(engine);
```

```
   printf("INFO: Done\n");

   return 1;
}
```

Java API Concepts

Eduction SDK provides a Java API that enables your application to create an extraction engine and perform entity extractions.

This section describes the concepts used to write Java applications with the Eduction SDK.

The Java SDK consists of a JAR file, and a DLL (Windows) or shared object (UNIX/Linux).

- The JAR file contains the Eduction Java class library, and the interface to the Eduction Java Native Interface (JNI).
- The DLL or shared object implements the Eduction JNI library, and performs the Eduction functionality.
- Java developers can use either the Eduction JNI, the class library, or both. The JNI provides functionality almost identical to that of the Eduction C API. The class library encapsulates related JNI methods, implements exception handling, and provides return values from method calls that simplify application programming.

Naming Conventions

The main JNI class that provides access to native functionality is `EDKJNI`. Support classes for the JNI are prefixed with `EDKJNI`, for example `EDKJNIVersion`.

The Eduction class library classes are prefixed with `EDK`, for example `EDKEngine`.

Concurrency Control

Concurrency in Eduction is handled using *sessions*, represented by an `EDKSession` object. An instance of an `EDKEngine` object is initialized with corresponding grammars for entity extraction. Each such engine can be associated with one or more sessions. All the sessions in the engine share the same grammars. The engine must be configured fully before any sessions are created.

After a session is created, engine settings cannot be changed, or an exception is thrown. However, each individual session can process many documents or streams. Each session maintains its state independent of others. Each engine must be associated with at least one session by default.

Standalone API Usage

This section describes the basic structure of a stand-alone application using the API. See the source referenced in Java API Example, below. Typically, your application takes the following actions:

1. Instantiate an `EDKEngine` object.
2. Set the license key.
3. Configure the engine, load the grammars and select entities.
4. Instantiate an `EDKSession` object and associate it with the engine.
5. Feed UTF-8 encoded text to the session or pass the session a stream it can read.
6. Execute a `for each` loop to obtain an `EDKMatch` object for each match in a session.
7. Get the properties of the match.
8. Release resources when done.

Java API Reference

The Java API Reference documentation is located in the following directory:

`InstallDir``/java_api/help`

where *`InstallDir`* is the directory where you have installed the Eduction SDK.

Java API Example

The Eduction Java API includes sample programs that illustrate basic Eduction Java API usage. The samples are located in the `src/com/autonomy/eduction/test` folder. The samples are documented inline and are also described in the HTML documentation.

For information on how to build and run the sample programs, see Build and Run the Sample Programs, on page 29.

Chapter 4: Eduction ACI Server

This section describes the Eduction ACI Server.

Introduction to Eduction ACI Server

The Eduction ACI Server is a standalone server based upon the HPE ACI server. The Eduction Server processes UTF-8 encoded text, matching upon entities defined in Eduction grammars. Results return as XML, with tags in the HPE ACI hierarchy. You define the grammars to load and the entities to match on using a configuration file. Every time you send a query, the server creates a new engine with the grammars and entities that you specified.

You can make requests from a browser or ACI client. Browsers can make requests to process small amounts of text using an HTTP GET request, or larger amounts using an HTTP POST request.

You can specify configuration settings as query parameters to override the settings in the configuration file for individual queries. See Select Entities at Runtime, on page 85 for more information.

Command-Line Options

Usage: `eductionserver [options]`

Options	Description
`-version`	Displays the program version. This option must be the only argument. The program ends after the version information is displayed.
`directory`	Sets the working directory. The Eduction ACI server starts from this directory. All other arguments used in the command line are relative to this directory.
`-configfile` *`file`*	Sets the configuration file name. This option overrides the default configuration file name of `eductionserver.cfg`.
`-revokelicense`	Revokes a lock on the Eduction license from the HPE License Server.
`-install`	Installs Eduction as a service (Windows only). Syntax: `-install [-start auto|manual|disabled] [-username username] [-password password]`

Options	**Description**
	The `-start` option allows you to specify the startup mode. By default, it is automatic. If you do not supply a user name and password, Eduction runs under a local account.
`-uninstall`	Uninstalls the Eduction service.

Note: Options are case sensitive.

Note: On Linux, the ACI server requires the C++ library, `libstdc++.so`. To ensure the server can locate the required library, set the Library Path:

```
    setenv LD_LIBRARY_PATH bin:$LD_LIBRARY_PATH
```

Configuration File Settings

The Eduction ACI Server requires a configuration file, which identifies settings required for the server to run. The default configuration file is `eductionserver.cfg`. You can change the configuration file by using command-line options. See Command-Line Options, on the previous page.

See Eduction Parameters, on page 102 for more information.

Note: Some configuration settings affect extraction speed, for example `MatchCase`, `MatchWholeWords`, `AllowOverlaps`, and `NonGreedyMatch`. For more information on factors that can affect Eduction performance, refer to *IDOL Expert*.

Eduction ACI Server Configuration Parameters	
Required Parameters	
[License]	
`LicenseServerHost=`*`IP address`*	The IP address where the License server is running. If you use `LicenseServerHost`, you must also specify `LicenseServerACIPort`. If you do not specify these parameters, you must use `Key` and `Holder`.
`LicenseServerACIPort=`*`port`*	The port on the machine where the IDOL License server is running. If you use `LicenseServerACIPort`, you must also specify `LicenseServerACIHost`. If you do not specify these parameters, you must use `Key` and `Holder`.
`Key=`*`LicenseKey`*	The license key string. If you use `Key`, you must also set `Holder`. If you do not specify these parameters, you must use `LicenseServerHost` and `LicenseServerACIPort`.
`Holder=`*`LicenseKeyHolder`*	The holder of the license key string. This value is a multipart

Eduction ACI Server Configuration Parameters	
	string, which describes a group, machine, and MAC address. If you use `Holder`, you must also set `Key`. If you do not specify these parameters, you must use `LicenseServerHost` and `LicenseServerACIPort`.
[Server]	
`Port=`*Port*	The port on which the Eduction ACI Server listens for requests.
[Eduction]	
`ResourceFiles=`*GrammarFile* `[,`*GrammarFile2*`]...`	The name of one or more Eduction grammar files to load. The grammar files contain the entities to use to match text. You can specify multiple grammar names as a comma-separated list, by using wildcard expressions, or by specifying a comma-separated list of multiple wildcard expressions. There must be no space before or after a comma. **Note:** Because all entities loaded to an engine must have unique names, you must not load the same grammar file more than once.
`Entity`*N*`=`*entity*	The name of an Eduction entity, contained in the loaded grammar files. When you pass text to Eduction, it matches on the entity. *N* must start at zero, and increase by one for each entity you list. For example, `Entity0`, `Entity1,` and so on. This parameter supports wildcard expressions. **Note:** All entities must have unique names.
Recommended Parameters	
[Logging]	
`LogLevel=`*LogLevel*	The level of details for logging. The options are **`Full`**, **`Normal`**, **`Warning`**, **`Error`**, and **`Always`**. The default value is **`Normal`**.
[Server]	
`Threads=`*ACIThreads*	The number of threads that can accept concurrent Eduction ACI server requests. The default value is **`10`**.
[Eduction]	
`AllowDuplicates`	A list of IDOL fields in which to allow duplicate entities. You can enter multiple fields separated by a comma.

Eduction ACI Server Configuration Parameters	
`AllowMultipleResults`	Set this parameter to **`All`** or **`True`** to allow Eduction to return multiple matches starting at the same offset. Set this parameter to **`No`** or **`False`** to allow Eduction to return only one match starting at each offset. Set this parameter to **`OnePerEntity`** to allow Eduction to return up to one match starting at the same offset per entity.
`AllowOverlaps=[False|True]`	Whether to allow Eduction to return more than one match when multiple matches involve overlapping text.
`CantHaveFieldCSVs`	Names of fields that Eduction ignores when reading an XML file. Allows you to specify the fields in documents that are discarded before the documents are stored.
`CaseNormalization=[Upper|Lower]`	Converts the case of incoming text to all uppercase or all lowercase. By default, Eduction does not perform any case conversions.
`CaseSensitiveFieldName`	Whether to preserve the case sensitivity of configured field names. By default, the Eduction module converts all field names to uppercase when it produces matches. Set this parameter to **`True`** to preserve the case of the field names. This option makes field names case sensitive.
`CJKNormalization`	Whether to normalize Chinese, Japanese, and Korean data before extraction. Set this parameter to one of the following: • `Kana` • `OldNew` • `Number` • `HWNum` • `HWAlpha` • `SimpChi` • `FWJamo` See CJKNormalization , on page 108 for more information.
`Databases`	The names of the databases to which a document belongs. Eduction runs only on documents that belong to the comma-separated list of databases. If you do not list databases, Eduction is run on documents from all databases.
`DocumentDelimiterCSVs`	File fields (tags) marking the start and end of a document.
`EnableComponents=[False|True]`	Whether to output component details in the match results, when grammars define components in an entity. This parameter is used by `edktool` only, and is ignored by Eduction.

<table>
<tr><th colspan="2">Eduction ACI Server Configuration Parameters</th></tr>
<tr><td><code>EnableUniqueMatches</code></td><td>Whether to return only unique matches in each document. If you set this parameter to <b>True</b>, Eduction returns a single occurrence of a particular value. If the same value occurs more than once, only the first instance is returned, even if the matches occur for different entities.</td></tr>
<tr><td><code>EntityAdvancedField<i>N</i></code></td><td>A comma-separated list of advanced fields to return.
To use this option you must:
• set <code>OutputSimpleMatch</code> to <b>False</b> for <code>edktool</code>.
• set <code>EnableComponents</code> to <b>True</b> for <code>edktool</code>.
• define components in the entity definition.</td></tr>
<tr><td><code>EntityComponentField<i>N</i></code></td><td>A comma-separated list of entity components that you want to return as fields.
To use this option you must:
• set <code>OutputSimpleMatch</code> to <b>False</b> for <code>edktool</code>.
• set <code>EnableComponents</code> to <b>True</b> for <code>edktool</code>.
• define components in the entity definition.</td></tr>
<tr><td><code>EntityMatchRange<i>N</i></code></td><td>A range of matching instances of the entity that are returned. The entity match range number <i>N</i> must match the corresponding <code>Entity<i>N</i></code> number. The format of the range is as follows:
<code>&lt;match&gt;\[{-\|,}&lt;match>\]\[,...\]*</code></td></tr>
<tr><td><code>EntityMinScore<i>N</i></code></td><td>Matches only items with scores equal to or exceeding the threshold. The entity minimum score number <i>N</i> must match the corresponding <code>Entity<i>N</i></code> number.</td></tr>
<tr><td><code>EntitySearchFields<i>N</i></code></td><td>Specifies the IDX fields to use for an entity. Matches for an entity are returned only if they occur in one of the specified fields.</td></tr>
<tr><td><code>MatchCase=[False|True]</code></td><td>Whether matching is case-sensitive. The default value is <b>True</b>.</td></tr>
<tr><td><code>LanguageDirectory</code></td><td>Enables tokenization of some languages using sentence breaking libraries. Set <code>LanguageDirectory</code> to the path of an IDOL Server language directory that contains the relevant sentence breaking libraries and associated data files.</td></tr>
<tr><td><code>Locale</code></td><td>Enables tokenization of some languages using sentence breaking libraries. Set <code>Locale</code> to one of <b>CHI</b>, <b>JPN</b>, <b>KOR</b>, or <b>THA</b>.
<b>Note:</b> The standard grammar files are developed without this setting; HPE recommends that you use this</td></tr>
</table>

Eduction ACI Server Configuration Parameters	
	parameter only when you are using custom grammar files that have been developed with the specific tokenization.
`MatchWholeWord=[False|True]`	Whether to allow matching on parts of a word. The default value is `False`, which means that matching is performed only on whole words, for example, *part* does not return from *partake*.
`MaxEntityLength=`*`Length`*	The maximum number of bytes that a match can have. The default value is `256`.
`MaxMatchesPerDoc`	The maximum number of matches to allow in each document.
`NonGreedyMatch`	Whether to return the shortest match. Set `NonGreedyMatch` to **`True`** to configure the Eduction module to return the shortest match. Setting this parameter to **`True`** implicitly disables the `AlowOverlaps` and `AllowMultipleResults` parameters. If you have set these parameters, `NonGreedyMatch` takes precedence.
`OutputSimpleMatchInfo`	Setting `OutputSimpleMatchInfo` to **`True`** generates basic match information only, such as document, entity, position, and original text. If `OutputSimpleMatchInfo=True`, the `EnableComponents` setting has no effect and reverts to **`False`**. This parameter is used by `edktool` only, and is ignored by Eduction.
`RedactionOutputString`	A string to include in the output in place of redacted text. If neither `RedactionOutputString` nor `RedactionReplacementCharacter` is set, Eduction uses the default value of `[redacted]`.
`RedactionReplacementCharacter`	A character to include in the output in place of each character in a passage of redacted text.
`SearchFields`	A comma-separated list of IDOL fields to search for entities. You can search the following IDOL fields: • `DREREFERENCE` • `DRETITLE` • `SUMMARY` • `DRECONTENT` You can also add any customized fields present in the IDOL database to this list. You must specify at least one field to search, or no results return.

Eduction ACI Server Configuration Parameters	
SuppressMatchLogging	Set this parameter to **True** to suppress log entries for every entity and zone pattern found in a document. When logging is set to **Full** in the IDOL configuration file, Eduction makes a log entry for every entity and zone pattern found in a document. If you set this parameter to **True**, these log entries are suppressed. This option is useful when you want to log the performance timing information, but do not want the verbose match entries. You can also set this parameter in Eduction Server. If you set logging to **Full** in the Eduction Server configuration file, the server records a log entry for every entity match found. You can set SuppressMatchLogging to **True** to suppress these log entries.
TangibleCharacters	A list of non-alphanumeric characters to make searchable.
TokenWithPunctuation	Whether to treat all punctuation characters as part of a word token, rather than treating them as word boundaries. Setting this parameter to **True** is equivalent to setting the TangibleCharacters parameter to all punctuation characters.
[PostProcessingTasks]	
NumTasks	The number of post-processing tasks that you want to configure.
Task*N*	The name of the individual post-processing task that you want to configure.
[*MyPostProcessingTask*]	
Entities	A list of the extracted entities that you want to use your post-processing script to modify. This parameter supports wildcard expressions.
ProcessEnMasse	Set ProcessEnMasse to **True** to set up an en-masse post-processing task.
Script	The path to the script to use for your pre-processing task.

Example Configuration File

```
[License]
LicenseServerHost=127.0.0.1
LicenseServerACIPort=20000

[Server]
Port=7075
```

```
Threads=1

[Eduction]
ResourceFiles=person_name_engus.ecr
Entity0=person/femalefirstname/engus
Entity1=person/malefirstname/engus
Entity2=person/lastname/engus
//MinScore=0.5
//MaxEntityLength=12
//MatchCase=0
//CaseNormalization=Lower
//AllowOverlaps=1
//EnableComponents=1
//MatchWholeWord=0
//RedactionOutputString=[censored]

[Logging]
LogLevel=full
0=ApplicationLogStream

[ApplicationLogStream]
LogFile=application.log
```

Include an External Configuration File

You can share configuration sections or parameters between ACI server configuration files. The following sections describe different ways to include content from an external configuration file.

You can include a configuration file in its entirety, specified configuration sections, or a single parameter.

When you include content from an external configuration file, the `GetConfig` and `ValidateConfig` actions operate on the combined configuration, after any external content is merged in.

In the procedures in the following sections, you can specify external configuration file locations by using absolute paths, relative paths, and network locations. For example:

```
../sharedconfig.cfg
K:\sharedconfig\sharedsettings.cfg
\\example.com\shared\idol.cfg
file://example.com/shared/idol.cfg
```

Relative paths are relative to the primary configuration file.

> **Note:** You can use nested inclusions, for example, you can refer to a shared configuration file that references a third file. However, the external configuration files must not refer back to your original configuration file. These circular references result in an error, and Eduction SDK does not start.
>
> Similarly, you cannot use any of these methods to refer to a different section in your primary configuration file.

Include the Whole External Configuration File

This method allows you to import the whole external configuration file at a specified point in your configuration file.

To include the whole external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

   ```
   < "K:\sharedconfig\sharedsettings.cfg"
   ```

4. Save and close the configuration file.

Include Sections of an External Configuration File

This method allows you to import one or more configuration sections from an external configuration file at a specified point in your configuration file. You can include a whole configuration section in this way, but the configuration section name in the external file must exactly match what you want to use in your file. If you want to use a configuration section from the external file with a different name, see Merge a Section from an External Configuration File, on the next page.

To include sections of an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the external configuration file section.
3. On a new line, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file name, add the configuration section name that you want to include. For example:

   ```
   < "K:\sharedconfig\extrasettings.cfg" [License]
   ```

   Note: You cannot include a section that already exists in your configuration file.

4. Save and close the configuration file.

Include a Parameter from an External Configuration File

This method allows you to import a parameter from an external configuration file at a specified point in your configuration file. You can include a section or a single parameter in this way, but the value in the external file must exactly match what you want to use in your file.

To include a parameter from an external configuration file

1. Open your configuration file in a text editor.
2. Find the place in the configuration file where you want to add the parameter from the external configuration file.
3. On a new line, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. After the configuration file name, add the name of the configuration section name that contains the parameter, followed by the parameter name. For example:

   ```
   < "license.cfg" [License] LicenseServerHost
   ```

   To specify a default value for the parameter, in case it does not exist in the external configuration file, specify the configuration section, parameter name, and then an equals sign (=) followed by the default value. For example:

   ```
   < "license.cfg" [License] LicenseServerHost=localhost
   ```

4. Save and close the configuration file.

Merge a Section from an External Configuration File

This method allows you to include a configuration section from an external configuration file as part of your Eduction SDK configuration file. For example, you might want to specify a standard SSL configuration section in an external file and share it between several servers. You can use this method if the configuration section that you want to import has a different name to the one you want to use.

To merge a configuration section from an external configuration file

1. Open your configuration file in a text editor.
2. Find or create the configuration section that you want to include from an external file. For example:

   ```
   [SSLOptions1]
   ```

3. After the configuration section name, type a left angle bracket (<), followed by the path to and name of the external configuration file, in quotation marks (""). You can use relative paths and network locations. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg"
   ```

   If the configuration section name in the external configuration file does not match the name that you want to use in your configuration file, specify the section to import after the configuration file name. For example:

   ```
   [SSLOptions1] < "../sharedconfig/ssloptions.cfg" [SharedSSLOptions]
   ```

   In this example, Eduction SDK uses the values in the `[SharedSSLOptions]` section of the external configuration file as the values in the `[SSLOptions1]` section of the Eduction SDK configuration file.

   Note: You can include additional configuration parameters in the section in your file. If these

> parameters also exist in the imported external configuration file, Eduction SDK uses the values in the local configuration file. For example:
>
> ```
> [SSLOptions1] < "ssloptions.cfg" [SharedSSLOptions]
> SSLCACertificatesPath=C:\IDOL\HTTPConnector\CACERTS\
> ```

4. Save and close the configuration file.

Lua Post-Processing

Eduction Server, edktool, and the Eduction SDK support Lua post-processing. You can use this to carry out internal consistency checks, or to format your output. For example, if you configured an Eduction task to extract credit card numbers, you can set up a post-processing task to discard matches that do not pass checksum validation and are therefore determined to be strings of numbers rather than correctly formatted credit card numbers.

Configure Post-Processing

To configure Lua post-processing tasks, you must specify the number of tasks and task names in the `[PostProcessingTasks]` section of the configuration file, for example:

```
[PostProcessingTasks]
numTasks=2
task0=checksum
task1=filterscore
```

Each task must also have its own section in the configuration file, for example:

```
[checksum]
type=lua
script=./scripts/checksum.lua
entities=number/creditcard
```

You must define a `processmatch` function in your script:

```
function processmatch (edkmatch)
```

where `edkmatch` is the object to match. This returns a Boolean value; set this to **`True`** to keep the match, or **`False`** to discard it.

Methods and Parameters

The following methods and parameters are available:

- `edkmatch:getEntityName(), edkmatch:setEntityName(new_name)`
- `edkmatch:getOutputText(), edkmatch:setOutputText(new_text)`
- `edkmatch:getMatchedText(), edkmatch:setMatchedText(new_text)`
- `edkmatch:getOffset(), edkmatch:setOffset(new_offset)`
- `edkmatch:getOffsetLength(), edkmatch:setOffsetLength(new_length)`
- `edkmatch:getScore(), edkmatch:setScore(new_score)`
- `edkmatch:getComponentCount()`
- `edkmatch:getComponent(index)`
- `edkcomponent:getName(), edkcomponent:setName(new_name)`
- `edkcomponent:getText(), edkcomponent:setText(new_text)`
- `edkmatch:addComponent(name, offset, offsetLength)`

For more information, see Eduction Lua Methods Reference, on page 317.

Configure En Masse Scripting

The previous section describes how to set up a post-processing task whereby the script performs the required action on each match individually. You can also set up en masse scripting tasks whereby all the matches found by an Eduction task are passed into a single post-processing task, which can use this information to modify all the matches at the same time. For example, you might want to increase the score of all matches that appear near other matches and are therefore considered more reliable as matches.

To configure an en masse scripting task, add the `ProcessEnMasse` parameter to the configuration file section for the task and set it to **`True`**.

You must define a `processmatches` function in your script:

```
function processmatches (matches)
```

where `matches` is a Lua table of all the matches extracted by your Eduction task. Each match also includes a flag to indicate if the match should display in the output.

You must set up the script so that it iterates over each match in the table, performs the post-processing task on each match, and sets `setOutput` to **`False`** for matches that should be discarded. You can use the sample script that is provided with Eduction as a reference.

Example Scripts

The following example post-processing script sets the score for all matches to **`0.5`**:

```
function processmatch (edkmatch)
if edkmatch then
```

```
    -- just overwrite the score
    edkmatch:setScore(0.5)
    end
return true
```

You can also use the sample scripts that are included with Eduction to perform the following tasks, or you can use the sample scripts as a reference if you want to write your own scripts.

Checksum Validation

The `checksum_luhn.lua` script verifies the checksum digit of each match using the *Luhn algorithm*, and reduces the score associated with the match if the checksum is wrong. The `checksum_luhn_enmasse.lua` script performs checksum validation as an en masse processing task, discards incorrect matches, and alters the score of correct matches to equal the proportion of matches that have the correct checksum digit.

You can use these scripts with the `number_cc.ecr` and `number_sin_ca.ecr` grammar files to validate most credit card numbers.

Spanish Identity Card Number Validation

You can use the `checksum_dni_es.lua` script with the `number_dni_es.ecr` grammar file to validate Spanish Documento Nacional de Identidad (national identity card) numbers.

Dutch Citizen Service Number Validation

You can use the `number_bsn_nl.lua` script with the `number_bsn_nl.ecr` grammar file to validate Dutch Citizen Service Numbers (Burgerservicenummer, or BSNs).

Geographical Co-ordinate Standardization

You can use the `lat_long.lua` script with the `place_lat_long.ecr` grammar file to convert and standardize the output of geographical co-ordinates.

Date and Time Standardization

You can use the `datetime.lua` script with the `datetime_advanced_eng.ecr` grammar file to convert and standardize the output of dates and times in English into a standardized format in cases where there are matches on several formats. For example, you can convert both *23/11/13* and *Nov 23 2013* to epoch seconds, or to *20131123*.

The `datetime_advanced_eng.ecr` grammar file can understand English natural language, and relative dates such as *last Saturday morning*. You can provide a reference date for `<today>` in the Lua script to enable normalization of relative dates into standard formats.

You can also use `datetime.lua` to convert number matches expressed in English to numeric format

Server Actions

You can run actions on the Eduction ACI server using the HTTP request:

```
http://Host:Port/?action=action[&Parameter=Value[&Parameter=Value...]]
```

For example:

```
http://localhost:13000/action=GetStatus
```

Eduction ACI server has the following valid actions (case insensitive):

GetStatus

Returns the status of the Eduction ACI Server, including version information and entities selected for matching.

Usage:

```
action=GetStatus
```

For example:

```
http://localhost:13000/action=GetStatus
```

This action returns output similar to the following XML:

```
<autnresponse>
   <action>GETSTATUS</action>
   <response>SUCCESS</response>
   <responsedata>
      <Name>EDUCTION</Name>
      <ACIPort>7075</ACIPort>
      <Version>7.5.7.0</Version>
      <entitiesenabled>
        <name>contact/phone</name>
        <name>contact/email</name>
        <name>contact/people</name>
      </entitiesenabled>
      <numentitiesloaded>3</numentitiesloaded>
   </responsedata>
</autnresponse>
```

EduceFromText

Returns matches from text that you submit in the `Text` HTTP parameter.

Example

```
http://localhost:13000/?action=EduceFromText&Text=Simon was getting ready to leave.
```

This action might result in output similar to the following XML:

```
<autnresponse>
   <action>EDUCEFROMTEXT</action>
   <response>SUCCESS</response>
   <responsedata>
      <autn:hit>
         <entity_name>contact/people</entity_name>
         <offset>0</offset>
         <offset_length>0</offset_length>
         <score>1.00000</score>
         <normalized_text_size>5</normalized_text_size>
         <normalized_text_length>5</normalized_text_length>
         <original_text_size>5</original_text_size>
         <original_text_length>5</original_text_length>
         <original_text>Simon</original_text>
         <normalized_text>Simon</normalized_text>
      </autn:hit>
      <autn:numhits>1</autn:numhits>
   </responsedata>
</autnresponse>
```

EduceFromFile

Returns matches from text that you submit in the `FileName` HTTP parameter. Set `FileName` to the name of a file on the server machine that contains the input text to process.

Example

```
http://localhost:7075/?action=EduceFromFile&FileName=input.txt
```

This action might result in output similar to the following XML:

```
<action>EDUCEFROMFILE</action>
   <response>SUCCESS</response>
   <responsedata>
```

```
      <autn:hit>
         <entity_name>contact/people</entity_name>
         <offset>0</offset>
         <offset_length>0</offset_length>
         <score>1.00000</score>
         <normalized_text_size>5</normalized_text_size>
         <normalized_text_length>5</normalized_text_length>
         <original_text_size>5</original_text_size>
         <original_text_length>5</original_text_length>
         <original_text>simon</original_text>
         <normalized_text>simon</normalized_text>
      </autn:hit>
      <autn:numhits>1</autn:numhits>
   </responsedata>
</autnresponse>
```

RedactFromText

You can use the `RedactFromText` action to submit text in the `Text` HTTP parameter, and return the input text with any matches that contain confidential or sensitive data redacted.

Example

```
http://localhost:13000/?action=RedactFromText&Text=The driver Joe Bloggs was
questioned.
```

This action might result in output similar to the following XML:

```
<autnresponse>
<action>REDACTROMTEXT</action>
<response>SUCCESS</response>
<responsedata>
<autn:redacted_output>The driver [redacted] was questioned</autn:redacted_output>
</responsedata>
</autnresponse>
```

RedactFromFile

You can use the `RedactFromFile` action to submit text in the `FileName` HTTP parameter, and return the input text with any matches that contain confidential or sensitive data redacted. Set `FileName` to the name of a file on the server machine that contains the input text to process.

Example

```
http://localhost:7075/?action=RedactFromFile&FileName=redaction.txt
```

This action might result in output similar to the following XML:

```
<autnresponse>
<action>REDACTROMTEXT</action>
<response>SUCCESS</response>
<responsedata>
<autn:redacted_output>The driver [redacted] was questioned</autn:redacted_output>
</responsedata>
</autnresponse>
```

Select Entities at Runtime

You can customize the behavior of the extraction for individual query actions by specifying the configuration settings as query parameters in the ACI request. For example:

```
http://localhost:13000/?action=EduceFromFile&MatchCase=True&Grammars=place_albal.ecr
```

If you specify a parameter as part of a query, it overrides the configuration file settings for the parameter.

You can specify a comma-separated list of grammar files to load as the value of the `Grammars` parameter. For example:

```
Grammars=GrammarFile[,GrammarFile2]
```

This corresponds to the `ResourceFiles` setting in the configuration file.

You can specify a comma-separated list of entities to use as the value of the `Entities` parameter. If no entities are specified in the configuration file, all public entities from the grammar files that you configured are available.

Alternatively,if your query uses several grammar files or entities, you can use wildcard expressions in the `Grammars` or `Entities` parameters. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example:

```
action=EduceFromText&Text=I thought it was a bad idea. Es ist nicht
gut.&Grammars=sentiment_*.ecr
```

This example uses all the available sentiment grammars for the extraction without you having to type a lengthy comma-separated list.

> **Note:** The grammar files and entities must already be specified in the configuration file. There must be no space before or after a comma.

You can specify the following parameters as part of your query:

`AllowMultipleResults`	`MatchWholeWord`
`AllowOverlaps`	`MaxEntityLength`
`CaseNormalization`	`MinScore`
`CJKNormalization`	`NonGreedyMatch`
`EnableComponents`	`RedactionOutputString`

EnableUniqueMatches	RedactionReplacementCharacter
Locale	TangibleCharacters
MatchCase	TokenWithPunctuation

These parameters behave in the same way as the parameters of the same name in the configuration file. See Configuration File Settings, on page 70 for more information.

A query parameter takes the value of the corresponding parameter in the configuration file as its default. If that parameter is not set in the configuration file, the query uses the default value for the parameter as described in the *Eduction SDK Programming Guide*. See Eduction Parameters, on page 102 for more information.

Chapter 5: edktool Command-Line Tool

This section describes `edktool` and the configuration files that specify Eduction settings that are used with it.

About edktool

`edktool` is a tool for Eduction that allows you to compile and test your grammars. `edktool` can perform the following functions:

- compile grammars
- list available entities in a grammar file
- extract entities from a file based on a grammar and select entities from the grammar for extraction
- test the accuracy of the extraction process

Note: On Linux, `edktool` requires the C++ library, `libstdc++.so`. To ensure the tool can locate the required library, set the Library Path:

```
setenv LD_LIBRARY_PATH bin:$LD_LIBRARY_PATH
```

Wildcard Expressions in edktool

The -e and -g parameters in the Generate , Compile, Assess, and Extract options in edktool support wildcard expressions. For example, if you want to use all of the available sentiment analysis files in the `grammars` directory, you can type `-e "grammars/sentiment_*.ecr"` instead of typing a lengthy comma-separated list of multiple files.

You can use the `*` wildcard to match any number of characters, or the `?` wildcard to match a single character.

Note: In some cases (for example, if you are running Linux), the command shell automatically expands wildcard expressions, which can produce unexpected results in Eduction. To avoid this, you should enclose your wildcard expression in quotation marks.

edktool Syntax

Syntax for `edktool` is as follows:

```
Usage: edktool  {
     generate    -i <inputfile> [-o <outputfile>] [-e <entityname>]
     compile     [-i] <inputfile> [-e <entity>[,<entity>...]]
                 [-l <licensefile>] [-o <outputfile>] [-p] |
     list        <grammarfile> [-a] [-q]  |
     extract     [-l <licensefile>] -i <inputfile> [-c <configfile>]
                 [-g <grammarfile>[,<grammarfile>...]]
                 [-e <entity>[,<entity>...]] [-o <outputfile>] [-m] [-q]
                 [-r <redactionfile>] [-p] |
     measure     -e <expectedfile> -a <actualfile> [-o <resultsfile>] [-q]  |
     assess      [-l <licensefile>] [-o <resultsfile>] [-c <configfile>] [-a]
                 [-m <matched entities>] [-g <grammarfile>[,<grammarfile>...]]
                 [-e <entity>[,<entity>...]] [-v <valid input>]
                 [-w <invalid input>] [-x] [-q]  |
     permissions -d <directory> [-l <licensefile>] [-a] [-q]  |
     help        [<option>] }

Options:
  generate or g          Generates an uncompiled XML (source) file
     -i inputfile        The plaintext file containing one match per line
     -o outputfile       The destination for the output XML file
     -e entityname       The name for the single entity in the XML output
  compile or c           Compiles a grammar file
     -i inputfile        Input grammar file
     -e entities         Entities to be included in the grammar
     -l licensefile      The file containing the license key
     -o outputfile       Output grammar file
     -p                  Treat the input file as plaintext rather than XML

  list or l              Lists the available entities in a grammar file
        grammarfile      The grammar file
     -a                  Show additional info - such as license requirements
     -q                  "Quiet Mode" suppresses all descriptive messages

  extract or e           Extract entities from a file
     -l licensefile      The file containing the license key
     -i inputfile        The file on which entity extraction will be performed
     -c configfile       A configuration file controlling the extraction
     -g grammarfiles     Grammar files to use, if "-c" is not used
     -e entities         The entities to extract, if "-c" is not used
     -o outputfile       The file containing the results of the extraction
     -m                  Produce match results for IDOL input files
     -q                  "Quiet Mode" suppresses all descriptive messages
     -r redactionfile    A copy of the input file, with all matches redacted
```

```
   -p                Treat the resource file as plaintext rather than XML

measure or m           Measures precision and recall between extraction runs
   -e expectedfile     The expected results file from "edktool extract"
   -a actualfile       The actual results file from a separate extraction run
   -o resultsfile      The results: precision, recall and differences
   -q                  "Quiet Mode" suppresses all descriptive messages

assess or a            Measures precision and recall, using user-defined data
   -l licensefile      The file containing the license key
   -a                  Show all results, including true matches
   -c configfile       A configuration file controlling the assessment
   -g grammarfiles     Grammar files to use, if "-c" is not used
   -e entities         The entities to extract, if "-c" is not used
   -x                  Sets comparison mode to "exact", if "-c" is not used
   -m "match entities" Entities to check results against, if "-c" is not used
   -v "valid input"    The file containing valid input, if "-c" is not used
   -w "invalid input"  The file containing invalid input, if "-c" is not used
   -o resultsfile      The results: false matches, precision and recall
   -q                  "Quiet Mode" suppresses all descriptive messages

permissions or p       Displays all licensed grammar files in a directory
   -d directory        A directory containing some eduction grammar files
   -l licensefile      The file containing the license key
   -a                  Show additional information
   -q                  "Quiet Mode" suppresses all descriptive messages

help or h              Displays edktool help
      option           More detailed help on the edktool option
```

Note: If you do not specify a license key at the command line, `edktool` assumes that the location of the license file is `licensekey.dat`. If the license is kept in this location, you do not need to specify the *-l <licensefile>* parameter.

edktool Options

This section describes the `edktool` options.

Compile

This option creates a compiled Eduction grammar file.

You can use wildcard expressions in the `-e` parameter; see Wildcard Expressions in edktool, on page 87 for more information.

`-l <licensefile>` The file containing a valid license key for Eduction.

If you do not specify a license key at the command line, `edktool` assumes that the location of the license file is `licensekey.dat`. If the license is kept in this location, you do not need to specify this parameter.

`-i <inputfile>` The grammar file to process. The input file can be an uncompiled (Source) XML Eduction grammar file or a plaintext grammar file.

`-e <entity>` A comma-separated list of entities to include in the output file. If you do not include any entities in the command line, Eduction includes all entities in the input file in the output file. If you include entities in the command line, Eduction includes only those entities specified in the output file.

`-o <outputfile>` The output file name. If you do not specify the output file name, Eduction creates an output file using the XML grammar file name with `.ecr` appended.

`-p` Set this parameter if you want to use a plaintext grammar file (containing one potential match on each line) rather than an XML grammar file as the input text to compile from.

When compiling, the XML file must follow the Eduction syntax rules for laying out grammar files. The ECR file is a proprietary format that is optimized for fast loading into the Eduction engine at run time. While the engine can load XML grammar files, as well as compiled ECR files, compiling a grammar file makes loading quicker.

Because compiled grammar files are binary files and cannot be read, the `List` option allows you to view the public entities in a compiled grammar file.

You can also specify the `-p` parameter at the command line to compile a grammar file in ECR format from a plaintext grammar file. The plaintext grammar file must be in the format described in Plaintext Grammar File Format , on page 96.

List

This option lists the entities in an uncompiled (Source) XML Eduction grammar file or a compiled ECR grammar file. Listing the contents of an XML file lists all entities in the file, both private and public. Listing the contents of a compiled ECR file lists all public entities. Private entities not referenced by the public entities are removed from the compiled ECR file.

To enable this feature, type `edktool l <grammarfile>` at the command line.

You can also include the optional `-a` parameter when using the LIST option. As well as listing the components that the entity can return, this lists the licence requirements for a particular compiled grammar file. For example, the following output:

```
category: place              languages: English or French
```

indicates that the user must be licensed for either English or French in the place category. If multiple lines appear, then the license must satisfy the conditions in every line.

If you include the optional `-q` “Quiet Mode” parameter, `edktool` removes all descriptive messages from the output and shows the entity list only. The output includes components if you also set the `-a` parameter.

Permissions

This option reads any specified directory and returns a list of all compiled grammar files inside it that you can access using the specified licence.

To enable this feature, type `edktool p -d` *`<directory>`* `-l` *`<licencefile>`* at the command line. You can also include the optional parameter `-a` to return a list of all compiled grammar files inside the directory that are **not** accessible under the specified licence.

Note: If you do not specify a license key at the command line, `edktool` assumes that the location of the license file is `licensekey.dat`. If the license is kept in this location, you do not need to specify the `-l` *`<licensefile>`* parameter.

You can include the optional `-q` parameter to enable "Quiet Mode" and remove descriptive messages from the output. If you enable “Quiet Mode”, the output consists of a list of file names only, in the format `Valid: filename.ecr` or, if you also included the `-a` parameter, `Invalid: filename.ecr`.

Generate

This option generates an uncompiled XML source file from a plaintext grammar file.

To enable this feature, type `edktool g -i` *`<inputfile>`* at the command line. You can also specify the optional `-o` and `-e` parameters.

You can use wildcard expressions in the `-e` parameter; see Wildcard Expressions in edktool, on page 87 for more information.

`-i` *`<inputfile>`* The plaintext grammar file to process. This file must contain one potential match on each line.

`-e` *`<entity>`* The resulting XML grammar file contains a single entity; you can specify the name of this entity as the value of the -e parameter. If you do not specify a name, the entity will be given a default name based on the name of the input file.

`-o` *`<outputfile>`* The output file name.

Related Topics

- Plaintext Grammar File Format , on page 96

Assess

This option enables you to assess the performance and accuracy of an Eduction grammar against a set of pre-tagged examples.

You must supply a text file with one phrase on each line; the `Assess` feature checks whether each line contains a match.

You must specify at least one input file, using the `-v` parameter or the `-w` parameter. If required, you can specify both of these parameters.

You can use wildcard expressions in the `-e` and `-g` parameters; see Wildcard Expressions in edktool, on page 87 for more information.

`-l` *`<licensefile>`*

The file containing a valid license key for Eduction.

If you do not specify a license key at the command line, `edktool` assumes that the location of the license file is `licensekey.dat`. If the license is kept in this location, you do not need to specify this parameter.

`-c` *`<configfile>`*

A configuration file controlling the assessment. The configuration file can be either an IDOL Server style .CFG configuration file or an XML configuration file. See Configuration Files for Eduction Settings, on page 97.

You can specify one or more grammar files and one or more entities in place of a configuration file. Specifying a configuration file overrides the grammar or entity parameters.

`-g` *`<grammarfile>`*

A grammar file to use when `-c` is not used.

If you provide a grammar file but you do not specify any entities with `-e`, Eduction extracts all entities in the grammar file.

`-e` *`<entity>`*

The entities to extract when `-c` is not used. Separate multiple entities with a comma.

`-x`

(Optional) Modifies the behavior so that the `Assess` feature checks for exact matches.

`-m` *`<matched entities>`*

(Optional) This parameter does not change the extraction behavior, but enables you to check which entities are producing the matches.

`-v` *`<valid_input>`*

A file of phrases where a match would be valid.

`-w` *`<invalid_input>`*

A file of phrases where a match would be invalid.

`-a`

(Optional) The output includes explanations of each failure, and statistics such as recall, precision, and F1 (depending on the type of input file you provided). Include the `-a` parameter to display additional output, including the results for every phrase in your input files.

`-o` *`<outputfile>`*

(Optional) By default, Eduction sends output to the console. To send the output to a file, use the `-o` parameter.

The output is a list of all phrases that failed. For valid input this would be a phrase that contained no match; for invalid input this would be a phrase that contained a match.

`-q` (Optional) Sets “Quiet Mode” so that descriptive messages are removed, and the output consists only of a list of examples that failed, in the form `"FAIL: "text" is matched by "entity""` or similar, depending on the test specifications. If you also set the `-a` parameter, examples that pass are also included in the output.

For more information on how to use the Assess feature to check the effectiveness and performance of your grammar files, refer to *IDOL Expert*.

Extract

This option extracts entities from a document. It can print the output to a file, or to the console. You can use this option to test your grammars.

You can use wildcard expressions in the `-e` and `-g` parameters; see Wildcard Expressions in edktool, on page 87 for more information.

Redact Extraction Results

You can enable redaction on extracted matches in `edktool` either by setting `RedactedOutput` to **`True`** in the `edktool` configuration file, or by specifying a redaction file using the `-r` parameter at the command line. Note that `edktool` only performs redaction on fields that you have configured as IDOL search fields.

If you have specified an IDX file to perform extraction on, existing fields are preserved in their unredacted form, and a redacted copy of each search field is added to the IDX file, with `_REDACTED` appended to the original field name. For example:

```
#DREREFERENCE 1
#DREFIELD DRECONTENT_REDACTED="The driver ########## was questioned."
#DRECONTENT
The driver Joe Bloggs was questioned.
#DREENDDOC
```

If you have specified a plaintext file to perform extraction on, the entities identified as matches by `edktool` are redacted from the input text to form the redacted output. For example:

Input:

```
The driver Joe Bloggs was questioned.
```

Output:

```
The driver ########## was questioned.
```

Eduction sends redacted output to the file specified in the`-r` parameter. If you do not specify this argument but you have enabled redaction in the configuration file, Eduction displays redacted output in the console after the list of matches, unless you have specified the `-q` parameter at the command line to enable Quiet mode. In Quiet mode, redacted output does not display in the console.

`-l` *`<licensefile>`* The file containing a valid license key for Eduction.

If you do not specify a license key at the command line, `edktool` assumes that the location of the license file is `licensekey.dat`. If the license is kept in this location, you do not need to specify this parameter.

`-i` *`<inputfile>`*

The file to perform entity extraction on. The input file can be either an IDOL IDX file, an IDOL XML file, or a plain text file. It must be UTF-8 encoded.

> **Note:** If the input file is an XML file, the configuration file (in either IDOL configuration file format or XML format) must contain entries for the `DocumentDelimiterCSVs` parameter. If this setting is not correct, Eduction might not find any documents in the XML file. For information on how to set this option, refer to the Eduction Parameters, on page 102.

`-c` *`<configfile>`*

A configuration file controlling the extraction. The configuration file can be either an IDOL Server style .CFG configuration file or an XML configuration file. See Configuration Files for Eduction Settings, on page 97.

You can specify one or more grammar files and one or more entities in place of a configuration file. Specifying a configuration file overrides the grammar or entity parameters.

`-g` *`<grammarfile>`*

A grammar file to use when `-c` is not used.

If you provide a grammar file but do not specify any entities with `-e`, Eduction extracts all entities in the grammar file.

`-e` *`<entity>`*

The entities to extract when `-c` is not used. Separate multiple entities with a comma.

`-o` *`<outputfile>`*

The file containing the results of the extraction. The content of the optional output file depends on the type of input file provided and whether the `-m` option is used.

If the input file type is an IDOL file and the `-m` option is *not* used, the output file is identical to the input file, except the matched entities are appended to each document as additional fields. This behavior is the same as Eduction running in IDOL.

If the input file is a plain text file or an IDOL file with the `-m` option, the output file is an XML file containing the matched entities.

If the input file is an IDOL file, the output file also contains document information.

`-m`

Produce match results for IDOL input files.

`-q`

(Optional) Sets “Quiet Mode” so that descriptive messages and redacted output are removed, and the output consists of the XML matchlist only (that is, an XML document with all the matches and any configured metadata).

`-r` *`<redaction_file>`*

A copy of the input file, with all matches redacted.For example, if you specified an IDX input file, the content is sent to the redaction file as follows, with the redactions made in place:

```
#DREREFERENCE 1
#DRECONTENT
The driver ########## was questioned.
#DREENDDOC
```

`-p`	Set this parameter if you want to use a plaintext grammar file rather than an XML grammar file as the input text to extract from.

The extract option requires an input file (either in IDOL IDX, IDOL XML, or plain text format) and either a configuration file or a grammar file. If you do not provide a configuration file, `edktool` searches the file for any specified entities in the specified grammar (or all entities, if none are specified). For example, in the simplest command line:

```
C:\>edktool e -i myData.txt -g grammar1.ecr,grammar2.ecr
```

`edktool` is invoked with no configuration file. It uses the command-line arguments to process the data file `myData.txt` with the grammar files `grammar1.ecr` and `grammar2.ecr`. Eduction identifies all the entities in the two grammar files, and matches on these. The output is sent to the console in XML format, identifying matches in the data file and using the entity names to generate field names for the matches that contain the matched data. Assuming `myData.txt` is a plain text file, the entire body of the file is matched.

You can also specify the `-p` parameter at the command line to extract matches from a plaintext grammar file.

The plaintext grammar file must be in the format described in Plaintext Grammar File Format , on the next page.

Measure

This option measures precision and recall between extraction runs by comparing the *expected results* of entity extraction with the *actual results*.

Expected results are created once and remain as a base reference for ongoing tests. Actual results are generated as required each time a grammar is modified. The two results are compared to generate precision and recall information.

To generate expected results, run `edktool -extract`, and then revise the generated output file so that it contains the correct matches. From then on, `edktool -extract` is used only to create the actual results, and the two files are compared against each other to generate precision and recall information on an ongoing basis.

`-e` *<expectedfile>*	The expected results file from `edktool -extract`.
`-a` *<actualfile>*	The actual results file from subsequent extraction runs with modified grammar files.
`-o` *<resultsfile >*	The results, including precision, recall, and differences.
`-q`	(Optional) Sets “Quiet Mode” so that descriptive messages are removed, and the output consists of only an XML document containing the differences between the expected and actual output.

For more information on how to use the Measure feature to check the effectiveness and performance of your grammar files, refer to *IDOL Expert*.

Help

This option lists the valid `edktool` options along with brief descriptions for each.

Plaintext Grammar File Format

Plaintext grammar files must have only a single entity, that consists entirely of headwords. Patterns, synonyms, scoring and so on are **not** supported.

Each line in the grammar file must consist of either a headword, a blank line, or a comment (a line beginning with `//` that is skipped when the file is read). Whitespace and blank lines are ignored when the file is read.

Command-Line Examples

This section provides some examples of the command-line tool functionality.

Compile Grammars

```
edktool c mygrammar.xml
```

Compiles `mygrammar.xml` into `mygrammar.ecr`.

```
edktool c -i mygrammar.xml -e common/* -o compiledgrammar.ecr
```

Compiles all the entities in the `common` entity type in `mygrammar.xml` into `compiledgrammar.ecr`.

List Entities

```
edktool list mygrammar.ecr
```

Lists all public entities in the compiled grammar file, `mygrammar.ecr`.

Extract Entities

```
edktool e -i myPlainTextFile.txt -g myGrammar.ecr
```

Extracts all entities in `myGrammar.ecr` from `myPlainTextFile.txt`, sending the output to the console in XML format, with the field names for the matching text automatically generated from the entity names found in `myGrammar.ecr`.

```
edktool e -i myIDOLfile.idx -c myIDOLConfigFile.cfg –o myoutputfile.idx
```

Using the configuration file `myIDOLConfigFile.cfg`, extract entities from the file `myIDOLfile.idx` and direct the output with additional Eduction fields to the file `myoutputfile.idx`.

```
edktool e -i myIDOLfile.idx -c myIDOLConfigFile.cfg –o myoutputfile.xml -m
```

The same as the previous example, except output the match results to an `edktool` XML file.

Measure Accuracy

```
edktool m -e expected.xml -a actual.xml -o difference.xml -q
```

Compare `expected.xml` with `actual.xml` and put the difference in `difference.xml`, including precision and recall. “Quiet Mode” is enabled, so all descriptive messages are removed from the output.

Run Assessments

```
edktool a -l <license> -c <configuration_file> [-a] [-o <output_file>]
```

Run several assessments from a single Eduction configuration file.

The configuration file must contain a numbered `[assessmentN]` section for each assessment you want to run. You must specify the input files, the entities to match, and whether matching should be exact. For example:

```
[assessment0]
valid=data.txt

[assessment1]
entities=entity1,entity2
valid=match.txt
invalid=should_not_match.txt
exact=true
```

You can specify multiple entities either by separating them with commas, or by using wildcard expressions. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example, set `Entities` to `org/soccer/*` to use the entities `org/soccer/us`, `org/soccer/gb`, `org/soccer/de`, and so on without having to type a lengthy comma-separated list.

Configuration Files for Eduction Settings

The Extraction option of `edktool` can take its configuration settings from one for the following file types:

- .CFG file
- XML file

You can use the same Eduction configuration settings in each file format, including wildcard expressions where applicable.

Define Eduction Settings in the .CFG Configuration File

The Eduction configuration settings that can be defined in the IDOL Server format .CFG configuration file are described in Eduction Parameters, on page 102.

The .CFG configuration file consists of several sections that are identified by a phrase in square brackets. Each section contains parameters (name/value pairs). For example:

```
[Eduction]
ResourceFiles=C:\MyGrammar\gram1.ecr
```

To define Eduction settings in the .CFG configuration file

1. Open the .CFG configuration file in a text editor.
2. Set the Eduction SDK parameters as required. The following parameters are available (see Configuration File Settings, on page 70 and Eduction Parameters, on page 102 for more information):

   AllowDuplicates

   AllowMultipleResults

   AllowOverlaps

   CaseNormalization

   CJKNormalization

   CaseSensitiveFieldName

   Databases

   EnableComponents

   Locale

   MatchCase

   MatchWholeWord

   MaxEntityLength

   MaxMatchesPerDoc

   NonGreedyMatch

   OutputScores
   (used by `edktool`)

   OutputSimpleMatchInfo

(used by `edktool` only)

EnableUniqueMatches

EntityN

EntityAdvancedFieldN

EntityComponentFieldN

EntityFieldN

EntityMatchRangeN

EntityMinScoreN

EntitySearchFieldsN

EntityZoneN

LanguageDirectory

RedactedOutput

RedactionOutputString

RedactionReplacementCharacter

ResourceFiles

SearchFields

SuppressMatchLogging

TangibleCharacters

TokenWithPunctuation

ZoneEndN

ZoneStartN

> **Note:** If you set `CaseNormalization` to **`Lower`** or **`Upper`**, set `MatchCase` to **`True`**.

3. Set the following parameters in the `[Server]` section of the configuration file. These settings are critical for the correct reading of documents.

   ```
   CantHaveFieldCSVs
   DocumentDelimiterCSVs
   ```

4. Save and close the configuration file.

Modify Configuration Parameter Values

The following section describes how to enter parameter values in the configuration file.

Enter Boolean Values

The following settings for Boolean parameters are interchangeable:

```
TRUE = true = True = ON = on = Y = y = 1
```

```
FALSE = false = False = OFF = off = N = n =0
```

Enter String Values

Some parameters require string values that contain quotation marks. Percent-encode each quotation mark by inserting a backslash before it.

For example:

```
FIELDSTART0="<font face=\"arial\"size=\"+1\"><b>"
```

Here, the beginning and end of the string are indicated by quotation marks, while all quotation marks that are contained in the string are percent-encoded.

If you want to enter a comma-separated list of strings for a parameter, and one of the strings contains a comma, you must indicate the start and the end of this string with quotation marks.

For example:

ParameterName=**cat,dog,bird,"wing,beak",turtle**

If any string in a comma-separated list contains quotation marks, you must put this string into quotation marks and percent-encode each quotation mark in the string by inserting a backslash before it.

For example:

ParameterName=**"<font face=\"arial\"size=\"+1\"><b>",dog,bird,"wing,beak",turtle**

Sample Configuration File

The following shows the configuration for a sample Eduction task:

```
[Eduction]
ResourceFiles=C:\MyGrammar\gram1.ecr,C:\MyGrammar\gram2.ecr
ZoneStart0=<TEXT>
ZoneEnd0=</TEXT>
ZoneStart1=acknowledgements
ZoneEnd1=introduction
Entity0=common/aus_holidays
EntityField0=HOLIDAYS
EntityZone0=0
Entity1=common/us_holidays
EntityField1=HOLIDAYS
EntityZone1=0
Entity2=us/social_security_number
EntityField2=SS_NUMBER
EntityZone2=1
SearchFields=DRECONTENT
AllowDuplicates=HOLIDAYS

[Logging]
LogLevel=Full
```

This sample uses two grammar files. It searches for all Australian and U.S. holidays in the `DRECONTENT` field between the text <*Text*> and </*Text*>, adding the matches as additional fields `HOLIDAYS`. It also searches for a single social security number in `DRECONTENT` between the text *acknowledgements* and *introduction* and adds the results as a new field `SS_NUMBER`.

Define Eduction Settings in the XML Configuration File

The Eduction configuration elements that you can define in the XML file are described in Eduction Parameters, on page 102.

The following XML configuration file example shows all the available XML elements:

```
<?xml version="1.0" encoding="UTF-8"?>
```

```
<!-- Sample Eduction XML configuration file for the edktool utility -->

<Eduction>
    <!-- Global Settings (Defaults shown) -->
    <MatchWholeWord>true</MatchWholeWord>
    <SuppressMatchLogging>false</SuppressMatchLogging>
    <MaxEntityLength>256</MaxEntityLength>
    <AllowOverlaps>false</AllowOverlaps>
    <EnableComponents>false</EnableComponents>
    <OutputSimpleMatchInfo>true</OutputSimpleMatchInfo>
    <MatchCase>true</MatchCase>
    <DocumentDelimiterCSVs>*/DOCUMENT</DocumentDelimiterCSVs>
    <CantHaveFields>
        <CantHaveField>*/DRESTORECONTENT</CantHaveField>
        <CantHaveField>*/CHECKSUM</CantHaveField>
        <CantHaveField>*/DREWORDCOUNT</CantHaveField>
        <CantHaveField>*/DRETYPE</CantHaveField>
        <CantHaveField>*/IMPORTBODYLEN</CantHaveField>
        <CantHaveField>*/IMPORTMETALEN</CantHaveField>
        <CantHaveField>*/IMPORTLINKLEN</CantHaveField>
        <CantHaveField>*/IMPORTTITLELEN</CantHaveField>
        <CantHaveField>*/IMPORTQUALITY</CantHaveField>
        <CantHaveField>*/DREPAGE</CantHaveField>
        <CantHaveField>*/DREFILENAME</CantHaveField>
        <CantHaveField>*/dredoctype</CantHaveField>
        </CantHaveFields>

    <!-- Eduction grammar (resource) files to load -->
    <ResourceFiles>
        <ResourceFile>phone.ecr</ResourceFile>
        <ResourceFile>jargon.ecr</ResourceFile>
    </ResourceFiles>

    <!-- IDOL databases to search. Applies only to IDOL IDX or IDOL XML input
documents -->
    <Databases>
        <Database>Contact</Database>
        <Database>Customer</Database>
    </Databases>

    <!-- Document fields to search. ignored for plain text input documents
(DRECONTENT is the default) -->
    <SearchFields>
        <SearchField>DREREFERENCE</SearchField>
        <SearchField>DRETITLE</SearchField>
        <SearchField>DRECONTENT</SearchField>
    </SearchFields>

    <!-- Definitions of search zones within a document -->
    <Zones>
        <Zone>
            <Name>Summary</Name>
            <StartPattern>Executive Summary</StartPattern>
            <EndPattern>Introduction</EndPattern>
        </Zone>
        <Zone>
            <Name>Body</Name>
            <StartPattern>Introduction</StartPattern>
        </Zone>
    </Zones>

    <!-- Fields generated from a match. Always required, but applies only to IDOL
IDX or IDOL XML input documents where the output is also a modified IDOL document -
->
    <TargetFields>
        <TargetField>
            <Name>PHONE</Name>
            <AllowDuplicates>false</AllowDuplicates>
            </TargetField>
    </TargetFields>

    <!-- Eduction grammar entities used for searching -->
    <Entities>
        <Entity>
            <Name>phone/all</Name>
            <TargetField>PHONE</TargetField>
            <MatchRange>1,2-4</MatchRange>
            <MinScore>0.5</MinScore>
            <Zone>Summary</Zone>
            <Zone>Body</Zone>
        </Entity>
    </Entities>

</Eduction>
```

If Eduction reads an IDOL XML data file, you must configure `DocumentDelimiterCSVs`, and also at least one entry for the `CantHaveFields` setting. If this is not present, Eduction defaults to `DOCUMENT` and `EDUCTION_DUMMY_FIELD` respectively.

Eduction Parameters

The following parameters can be used in the .CFG configuration file or the XML configuration file.

CantHaveFieldCSVs

Names of fields Eduction ignores when reading an XML file. Allows you to specify the fields in documents that are discarded before the documents are stored.

To specify multiple fields, separate them with commas (there must be no space before or after a comma). You can use wildcards.

Type:	String
Default:	
Required:	No
Configuration Section:	**Server**
Example:	`CantHaveFieldCSVs=*/STANDARD_HEADER` In this example, any `STANDARD_HEADER` fields that a document contains are discarded before the document is stored in IDOL server.
See Also:	

DocumentDelimiterCSVs

File fields (tags) marking start and end of document. Allows you to specify the fields in a file that indicate the beginning and end of a document, so that the documents are indexed individually. You must have only one document level for each XML schema.

If documents are indexed into IDOL server using a `DREADD` or `DREADDDATA` index action, and the index action uses a `DocumentDelimiters` parameter that conflicts with the `DocumentDelimiterCSVs` setting in the IDOL server configuration file, the index action parameter overrides the configuration parameter.

Type:	String
Default:	
Required:	No
Configuration Section:	**Server**
Example:	`DocumentDelimiterCSVs=*/DOCUMENT,*/SPEECH` In this example, the beginning and end of individual documents in a file is marked by opening and closing `DOCUMENT` and `SPEECH` tags (or fields if you are running IDOL server in DRE3 mode).

See Also:	

AllowDuplicates

A list of Eduction fields in which Eduction can write duplicate entities. If you allow duplicates, and the same result is found in more than one place in the input, Eduction can add multiple fields with the same name and value.

You can specify multiple fields by separating them with commas.

Type:	String
Default:	
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`AllowDuplicates=ANIONIC_SURFACTANTS,PERSON`
See Also:	EntityN, on page 112 EntityFieldN, on page 112

AllowMultipleResults

This parameter determines how many results to return in cases where multiple matches begin at the same offset in the input text. By default, only one result can be returned, but you can use the `AllowMultipleResults` parameter if you want to return other matches (either from the same entity or from other entities).

Set `AllowMultipleResults` to one of the following options:

- **`All`** or **`True`**. This option returns all results at a specified offset.
- **`No`** or **`False`**. Only one result at a specified offset is returned.
- **`OnePerEntity`**. Up to one result at a specified offset per entity can be returned.

Example

With the following entities, `Georgia` might return `Georgia (name)`, `Georgia (US state)` or `Georgia (country)`. By default, Eduction returns only one match. This is appropriate if it is not important to you that Georgia has multiple interpretations. Set the `AllowMultipleResults` configuration parameter to **`All`** to return all three matches. Set the `AllowMultipleResults` configuration parameter to **`OnePerEntity`** to return one match from each entity. This is appropriate if it is important to you that Georgia is a place and a name, but not that it refers to multiple places.

```
<entity name="names">
  <entry headword="Georgia (name)">
    <synonym>Georgia</synonym>
  </entry>
  <entry headword="Henry (name)">
    <synonym>Henry</synonym>
  </entry>
</entity>

<entity name="places">
  <entry headword="Delaware (US state)">
    <synonym>Delaware</synonym>
  </entry>
  <entry headword="Georgia (US state)">
    <synonym>Georgia</synonym>
  </entry>
  <entry headword="Georgia (country)">
    <synonym>Georgia</synonym>
  </entry>
  <entry headword="Mongolia (country)">
    <synonym>Mongolia</synonym>
  </entry>
</entity>
```

Type:	String
Default:	No
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`AllowMultipleResults=All`
See Also:	EntityN, on page 112 EntityFieldN, on page 112 NonGreedyMatch , on page 119

AllowOverlaps

A Boolean that specifies whether Eduction returns more than one entity from any one section of text. To return only one entity from a section of text, set this parameter to **`False`**. To return all entities in the text, set this parameter to **`True`**.

For more information on overlapping and duplicate matches, refer to *IDOL Expert*.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`AllowOverlaps=True`
See Also:	NonGreedyMatch , on page 119

CaseNormalization

The case conversion to use for all incoming text. To improve performance, use this parameter to convert all text to lowercase or uppercase before attempting to match text.

This parameter takes one of the following values:

- **None**. No case conversion.
- **Lower**. All incoming text is converted to lowercase.
- **Upper**. All incoming text is converted to uppercase.

If your grammar file consists of only lowercase or only uppercase characters but your text is mixed case, you can improve performance by setting `CaseNormalization` to **`Lower`** or **`Upper`** respectively. This provides a greater performance improvement than setting `MatchCase` to **`False`**.

If you set this parameter to `Lower` or `Upper`, set `MatchCase` to **`True`**.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`CaseNormalization=lower`
See Also:	CaseSensitiveFieldName , below MatchCase , on page 117

CaseSensitiveFieldName

A Boolean that specifies whether to preserve the case of configured field names. By default, the Eduction module converts all field names to uppercase when it produces matches. To preserve the case of the field names, set this parameter to **`True`**. This option makes field names case sensitive.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`CaseSensitiveFieldName=True`

See Also:	

CJKNormalization

This parameter allows you to specify how to normalize Chinese, Japanese, and Korean data before extraction, in all Eduction components.

You can specify the value of `CJKNormalization` as follows:

- `Kana`. Half width kana to full width kana.
- `OldNew`. Old kanji to new kanji.
- `Number`. Chinese or kanji number characters to ASCII number characters.
- `HWNum`. Full width number characters to ASCII number characters.
- `HWAlpha`. Full width alphabet characters to ASCII alphabet characters.
- `SimpChi`. Traditional Chinese to simplified Chinese.
- `FWJamo`. Half width jamo to full width jamo.

Separate multiple options with a comma.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`CJKNormalization=SimpChi,Kana`
See Also:	

Databases

The names of the databases to which a document belongs. Eduction runs only on documents that belong to the comma-separated list of databases. If you do not list databases, Eduction is run on documents from all databases.

Note: If an IDX does not have a `DREDBNAME` entry for a document, matching is not done on that document. However, if all databases are selected, matching is done.

Type:	String
Default:	
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Databases=DB1,DB2,DB3`
See Also:	EntityN, on page 112 EntityFieldN, on page 112

EnableUniqueMatches

A Boolean that specifies whether to return only unique matches in each document. To return a single occurrence of a particular value, set this parameter to **True**. When `EnableUniqueMatches=True`, two `Entity`*`N`* definitions cannot return the same value, even if they use different patterns. If the same value occurs more than once, only the first instance is returned, even if the matches occur for different entities.

Duplicates display by default unless you set `EnableUniqueMatches` to **True** to explicitly remove them.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`EnableUniqueMatches=True`
See Also:	

Entities

A list of entities that you want to modify using the post processing script. If you do not set this parameter, you can use the script to modify the matches for every entity.

You can separate multiple entities with a comma, or, you can use wildcard expressions. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example, set `Entities` to `phone/*` to apply the script to the `phone/landline/gb`, `phone/mobile/gb` entities and so on.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post processing task.

Example:	`Task0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`

	`Entities=phone/landline/gb,phone/mobile/gb`
See Also:	Script , on page 122 ProcessEnMasse , on page 120

EntityAdvancedField*N*

A comma-separated list of advanced fields to return.

To use this option you must:

- set `OutputSimpleMatch` to **`False`** for edktool.
- set `EnableComponents` to **`True`** for edktool.
- define components in the entity definition.

You configure `EntityAdvancedField`*`N`* in the same way as EntityFieldN. Specify a comma-separated list of advanced fields that you want to return. The value of the advanced field is the output of simple operations (`min`, `max`, `sum`, and `ave`) on the values of entity components.

For example, for the following configuration:

```
Entity0=testgrammar/testentity
EntityField0=FIELD0
EntityAdvancedField0=OfferPrice:max(price1 price2),BidPrice:min(price1 price2)
```

And the following data:

```
share price1 price2
Com1  165    167
Com2  1890   1880
```

An entity with the following pattern:

```
<grammar name="testgrammar">
<entity name="testentity" type="public">
<pattern>(?A=price1:\d+)\s+(?A=price2:\d+)</pattern>
</entity>
</grammar>
```

Returns the following results as fields:

```
#DREFIELD FIELD0="165 167"
#DREFIELD OfferPrice="167"
#DREFIELD BidPrice="165"
#DREFIELD FIELD0="1890 1880"
#DREFIELD OfferPrice="1890"
#DREFIELD BidPrice="1880"
```

Type:	String
Default:	None
Required:	No

Configuration Section:	`Entity0=testgrammar/testentity`
	`EntityField0=FIELD0`
	`EntityAdvancedField0=OfferPrice:max(price1 price2),BidPrice:min(price1 price2)`

Example:	`AllowOverlaps=True`
See Also:	EntityN, on the next page EntityZoneN, on page 115

EntityComponentField*N*

A comma-separated list of entity components that you want to return as fields.

To use this option you must:

- set `OutputSimpleMatch` to **`False`** for edktool.
- set `EnableComponents` to **`True`** for edktool.
- define components in the entity definition.

You configure `EntityComponentFieldN` in the same way as EntityFieldN. Specify a comma-separated list of entity components that you want to return as fields.

For example, for the following configuration:

```
Entity0=testgrammar/testentity
EntityField0=FIELD0
EntityComponentField0=Name,Age
```

And the following data:

```
name    age
geoff   45
jane    54
```

An entity with the following pattern:

```
<grammar name="testgrammar">
<entity name="testentity" type="public">
<pattern>name\s+age(\n(?A=Name:\w+)\s+(?A=Age:\d+)){1,}</pattern>
</entity>
</grammar>
```

Returns the following values as fields:

```
#DREFIELD Name="geoff"
#DREFIELD Age="45"
#DREFIELD Name="jane"
#DREFIELD Age="54"
```

Type:	String
Default:	None

Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=testgrammar/testentity`
	`EntityField0=FIELD0`
	`EntityComponentField0=Name,Age`

See Also:	EntityN, below EntityZoneN, on page 115

EntityField*N*

A comma-separated list of document fields to associate with the entities specified by the `EntityN` parameter. If entities are identified in a document, the text is saved in the fields specified by this parameter. The entity field number *N* must match the corresponding `Entity`*N* number.

A many-to-many relationship exists between the `Entity`*N* and `EntityField`*N* parameters. If an `Entity`*N* setting does not have an `EntityField`*N* setting, text matching the entity is not passed to Eduction.

If no `Entity`*N* settings are provided, `EntityField`*N* settings are ignored, because Eduction automatically generates `EntityField`*N* settings corresponding to each EntityN, below setting that exists in the selected grammars.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`

See Also:	EntityN, below EntitySearchFieldsN, on page 115 EntityZoneN, on page 115

Entity*N*

A comma-separated list of entities to extract. Entities are defined in the resource file identified in the ResourceFiles parameter. Replace *N* with the zero-based rank of the entity.

You must associate each entity with a field by using the EntityFieldN parameter.

You cannot use the entity name `entities/ZoneStartN` or `entities/ZoneEndN` (where *N* is a numeric value). These entity names are reserved for use by Eduction.

If you do not define an `EntityN` parameter, Eduction looks for all entities in all loaded grammar files. In this case, the `EntityFieldN` settings are automatically generated from the entities found in grammar files by converting the entity names to uppercase and replacing slashes with an underscore. For example, if the entity `edk_common_entities/place` is found, Eduction generates the entity field: `EDK_COMMON_ENTITIES_PLACE`.

If you want to use several entities, you can use wildcard expressions instead of typing a lengthy comma-separated list. For example:

```
Entity0=place/city1/*,place/city2/*
EntityField0=CITY
Entity1=place/*/spabo
EntityField1=BOLIVIAN_PLACE
```

You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	`EntityZone0=0`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`
	`EntityZone1=1`
	
	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityFieldN, on the previous page EntityMinScoreN, on the next page EntitySearchFieldsN, on page 115 EntityZoneN, on page 115

EntityMatchRange*N*

A range of matching instances of the entity that are returned. The entity match range number *N* must match the corresponding EntityN number. The format of the range is as follows:

```
<match>\[{-\|,}<match>\]\[,...\]*
```

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityMatchRange0=1-3,6,9-`
	This example specifies the first through third match for the `ss_number` entity, as well as the sixth match and all matches starting with the ninth.

See Also:	EntityN, on page 112

EntityMinScore*N*

Matches only items with scores equal to or exceeding the threshold. The entity minimum score number *N* must match the corresponding EntityN number. The lowest possible score is 0. The upper limit varies depending on the entity.

The *score* for an entity is defined by the author of the grammar and defaults to **1**. See the Eduction Grammar Syntax for a description of the score attribute.

The minimum score defaults to **0**, which returns all matches.

As the minimum score is increased above **0**, up to a maximum of **1**, the user of the grammar is indicating that matches must meet a higher confidence level for the match to be returned. If scoring is not used by the author of an entity, the minimum score parameter specified by the user has no effect, because by default the entity has a score of **1** and therefore will always be returned.

The entity number (*N*) in `EntityMinScoreN` must match the corresponding entity number in the `EntityN` entry. For example:

```
Entity0=person/namefirstlast/engus
EntityMinScore0=0.5
```

Type:	Long
Default:	0
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityMinScore0=3`

See Also:	EntityN, on page 112

EntitySearchFields*N*

The `EntitySearchFieldsN` parameter enables you to specify the IDX fields to use for an entity, and to set up your Eduction task so that matches for an entity are returned only if they occur in one of the specified fields. If you do not configure search fields for an entity explicitly, Eduction uses the fields specified in the `SearchFields` parameter. If you do not set a value for the `SearchFields` parameter, Eduction uses the default settings.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	In the following example, matches for the `airport/icao` entity are returned only if they occur in the `STARTAIRPORT` or `DESTAIRPORT` fields. Because the `place/state/engus` entity has no search fields configured for it, it uses the fields in the `SearchFields` parameter.
	`[Eduction]`
	`SearchFields=DRECONTENT`
	`Entity0=airport/icao`
	`EntityField0=AIRPORTCODE`
	`EntitySearchFields0=STARTAIRPORT,DESTAIRPORT`
	
	`Entity1=person/femalefirstname/engus`
	`EntityField1=FIRSTNAME`
	`EntitySearchFields1=PASSENGER_FIRSTNAME`
	
	`Entity2=person/malefirstname/engus`
	`EntityField2=FIRSTNAME`
	`EntitySearchFields2=PASSENGER_FIRSTNAME`
	
	`Entity3=person/lastname/engus`
	`EntityField3=SURNAME`
	`EntitySearchFields3=PASSENGER_SURNAME`
	
	`Entity4=place/state/engus`
	`EntityField4=STATE`

See Also:	EntityN, on page 112 EntityFieldN, on page 112 SearchFields , on page 122

EntityZone*N*

Associates an EntityN entity with one or more zones defined using the ZoneStartN and ZoneEndN parameters. Type the number of the `ZoneStartN` and `ZoneEndN` parameters to associate with the

EntityN. Eduction searches for the entity in the specified zones. The entity zone number *N* must match the corresponding EntityN number.

Type:	Long
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`Entity0=edk_common_entities/ss_number`
	`EntityField0=SOCIAL_SECURITY_NUMBER`
	`EntityZone0=0`
	
	`Entity1=edk_common_entities/postal_address`
	`EntityField1=SHIPPING_ADDRESS`
	`EntityZone1=1`
	
	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	

LanguageDirectory

Enables tokenization of Chinese, Japanese, Korean, and Thai languages. Set `LanguageDirectory` to the path of an IDOL Server language directory that contains the relevant sentence breaking libraries and associated data files.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`C:\Program Files\IDOLServer\IDOL\langfiles`
See Also:	Locale , below

Locale

Enables tokenization of Chinese, Japanese, Korean, and Thai languages. Set `Locale` to one of **`CHI`**, **`JPN`**, **`KOR`**, or **`THA`**.

Note: The standard grammar files are developed without this setting; HPE recommends that you use this parameter only when you are using custom grammar files that have been developed with the specific tokenization.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`Locale=THA`
See Also:	LanguageDirectory , on the previous page

MatchCase

By default, Eduction is case sensitive when matching characters. This default value applies only when no grammar-specific case attribute has been specified for an entity.

To ignore case when matching characters, set this parameter to **`False`**.

Type:	Boolean
Default:	True
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`MatchCase=False`
See Also:	

MatchWholeWord

To match only terms in the text that begin and end on a whole word boundary, set this parameter to **`True`**.

To match terms that start and end anywhere, including in the middle of a word in the text, set this parameter to **`False`**.

For example, if `MatchWholeWord=True`, a search for the term `80` does not find a match in the text string `80mph`. If `MatchWholeWord=False`, a search for the term `par` finds a match in the text string `separated`.

Type:	Boolean
Default:	False

Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`MatchWholeWord=False`
See Also:	TangibleCharacters , on page 123 TokenWithPunctuation , on page 124

For more information on modifying the matching behavior by using `MatchWholeWord`, refer to *IDOL Expert*.

MaxEntityLength

The maximum number of characters in a returned entry.

Reducing this number can assist performance by preventing Eduction from scanning a long string of text for an entity that is expected to be small.

Type:	Integer
Default:	256
Allowed Range:	Minimum: 1 Maximum: 1024
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`MaxEntityLength=100`
See Also:	EntityN, on page 112 ZoneEndN, on page 124 ZoneStartN, on page 125

MaxMatchesPerDoc

The maximum number of matches to allow in each document.

Type:	Integer
Default:	Unlimited
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`MaxMatchesPerDoc=15`
See Also:	

NonGreedyMatch

A Boolean that specifies whether to return the shortest match. To configure Eduction to return the shortest match, set `NonGreedyMatch` to **`True`**. If two matches from two different entities start at the same word, and `NonGreedyMatch` is set to **`True`**, Eduction returns only the shortest match.

Setting this parameter to **`True`** implicitly disables the `AllowOverlaps` and `AllowMultipleResults` parameters. If you have set these parameters, `NonGreedyMatch` takes precedence.

For more information on how to configure the Eduction matching behavior using `NonGreedyMatch`, refer to *IDOL Expert*.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`NonGreedyMatch=True`
See Also:	AllowMultipleResults , on page 105 AllowOverlaps , on page 106

OutputScores

Set this parameter to **`True`** to include the score associated with a match in the output from an extraction task. If the output is in .IDX format, the score is added as a new `DREFIELD`, with the field name `SCORE`. If the output is in XML format, the score is added as an attribute with the name `"score"`.

Note: This parameter is used by `edktool` only.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`OutputScores=True`
See Also:	

ProcessEnMasse

Configures an en masse post-processing task. If you set `ProcessEnMasse` to **`True`**, your post-processing script takes the entire set of educed matches as its input argument, rather than a single match. The script can thus look at all the matches at once and modify them accordingly.

A Boolean that specifies whether to consider and modify all of the matches at the same time. For example, to increase the score of a match if it is found near other matches, you must consider all of the matches together.

- When `ProcessEnMasse=False`, the Lua post processing script takes an individual match as its input argument.
- When `ProcessEnMasse=True`, the Lua post processing script takes the entire set of matches as its input argument.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post processing task.

Example:	`PostProcessingTask0=EductionLuaPostProcessing`
	
	`[EductionLuaPostProcessing]`
	`Script=scripts/eduction_post_process.lua`
	`ProcessEnMasse=True`

See Also:	Entities, on page 109 Script , on page 122

RedactedOutput

Set this parameter to **`True`** to enable redaction of sensitive information in the output text.

You can also set **one** of RedactionOutputString or RedactionReplacementCharacter ; if neither are set, the default behavior is to replace redacted text with `[redacted]` in the output. If both are configured, `RedactionReplacementCharacter` takes precedence.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`RedactedOutput=False`
See Also:	

RedactionOutputString

A string that replaces redacted information in the output text.

Type:	String
Default:	[redacted]
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`RedactionOutputString=[censored]`
See Also:	

RedactionReplacementCharacter

A single character that replaces each character in redacted text.

Type:	String
Default:	Use [redacted] instead.
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`RedactionReplacementCharacter=*`
See Also:	

ResourceFiles

The full path to a compiled ECR file containing Eduction grammar entries. At least one resource file is required.

You can specify multiple resource files either by separating them with commas, or by using wildcard expressions. You can use the * wildcard to match any number of characters, or the ? wildcard to match a single character. For example, set `ResourceFiles` to `<grammar_files_directory>/sentiment_*.ecr` to use all available sentiment grammars without having to type a lengthy comma-separated list.

Type:	String

Default:	None
Required:	Yes
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`ResourceFiles=C:\MyGrammar\gram1.ecr,C:\MyGrammar\gram2.ecr`
See Also:	

Script

The path to the Lua script that you want to run to process the data returned by the Eduction module.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for an Eduction post processing task.
Example:	`Script=./scripts/checksum.lua`
See Also:	Entities, on page 109 ProcessEnMasse , on page 120

SearchFields

A comma-separated list of Eduction fields to search for entities. You can search the following Eduction fields:

- `DREREFERENCE`
- `DRETITLE`
- `SUMMARY`
- `DRECONTENT`

You can also add any customized fields present in the Eduction database to this list. You must specify at least one field to search, otherwise no results return.

Type:	String
Default:	`DRETITLE,SUMMARY,DRECONTENT`
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`SearchFields=DRECONTENT,DRETITLE`
See Also:	EntityN, on page 112 EntitySearchFieldsN, on page 115

SuppressMatchLogging

Set this parameter to **`True`** to suppress log entries for every entity and zone pattern found in a document.

When logging is set to **`Full`** in the Eduction configuration file, Eduction makes a log entry for every entity and zone pattern found in a document. If you set this parameter to **`True`**, these log entries are suppressed. This option is useful when you want to log the performance timing information, but do not want the verbose match entries.

You can also set this parameter in Eduction Server. If you set logging to **`Full`** in the Eduction Server configuration file, the server records a log entry for every entity match found. You can set `SuppressMatchLogging` to **`True`** to suppress these log entries.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`SuppressMatchLogging=True`
See Also:	

TangibleCharacters

A list of punctuation characters to treat as part of the word, rather than as word boundaries. By default almost all punctuation characters are treated as word boundaries.

Note: You cannot specify spaces, returns, and tabs as `TangibleCharacters`.

This parameter has no effect when MatchWholeWord is set to **`False`**.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`TangibleCharacters=-/\@`

See Also:	MatchWholeWord , on page 117 TokenWithPunctuation , below

For more information on using `TangibleCharacters` to specify punctuation characters to match, or to match punctuation at the start of a match, refer to *IDOL Expert*.

TokenWithPunctuation

A Boolean that specifies whether to treat all punctuation characters as part of a word token, rather than treating them as word boundaries. Setting this parameter to **`True`** is equivalent to setting the `TangibleCharacters` parameter to all punctuation characters.

This parameter has no effect when `MatchWholeWord` is set to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`TokenWithPunctuation=True`
See Also:	MatchWholeWord , on page 117 TangibleCharacters , on the previous page

For more information on using `TokenWithPunctuation` to configure all punctuation marks as tangible characters, refer to *IDOL Expert*.

ZoneEnd*N*

An Eduction regular expression that defines the end point of the zone. Replace *N* with the zero-based rank order of the zone description.

A zone is a section of a field defined by a starting and ending pattern. Zones locate entities in parts of a field. If you do not add zone entries, the entire field is matched for text. If the ending pattern is absent, the search begins at a match for the starting pattern and continues until the end of the field.

Use the EntityZoneN parameter to associate an entity identified in an EntityN parameter with one or more zones defined using the ZoneStartN and `ZoneEnd`*N* parameters.

Note: You must choose start and end patterns that do not match the same text in a field.

Type:	String
Default:	None
Required:	No
Configuration	Any section that you have defined for Eduction settings.

Section:	

Example:	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityN, on page 112 EntityZoneN, on page 115 ZoneStartN, below

ZoneStart*N*

An Eduction regular expression that defines the starting point of the zone. Replace *N* with the zero-based rank of the zone.

A zone is a section of a field defined by a starting and ending pattern. Zones locate entities in parts of a field. If you do not add zone entries, Eduction searches the entire field. If a starting pattern is absent, the search begins at the start of the field and ends with a match for the ending pattern.

Use the EntityZoneN parameter to associate an entity identified in an EntityN parameter with one or more zones defined using the `ZoneStartN` and ZoneEndN parameters.

Note: You must choose start and end patterns that do not match the same text in a field.

Type:	String
Default:	None
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.

Example:	`ZoneStart0=Social Security:`
	`ZoneEnd0=Shipping Address`
	
	`ZoneStart1=Shipping Address:`
	`ZoneEnd1=Billing Address`

See Also:	EntityN, on page 112 EntityZoneN, on page 115 ZoneEndN, on the previous page

Configuration Parameters Used by edktool Only

The following parameters are used by `edktool` only, and are ignored by Eduction.

EnableComponents

Set this parameter to **`False`** to return only the entity. Set it to **`True`** to return the entity and all the components of the entity.

This parameter requires `OutputSimpleMatchInfo` to be set to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`EnableComponents=True`
See Also:	OutputSimpleMatchInfo, below

OutputSimpleMatchInfo

When `edktool` is used with both the extract option and the option to generate a list of matches, setting `OutputSimpleMatchInfo` to **`True`** generates basic match information only, such as document, entity, position, and original text.

If `OutputSimpleMatchInfo=True`, the `EnableComponents` setting has no effect and reverts to **`False`**.

Type:	Boolean
Default:	False
Required:	No
Configuration Section:	Any section that you have defined for Eduction settings.
Example:	`OutputSimpleMatchInfo=True`
See Also:	EnableComponents, above

Match Validity

The order in which a match is assessed for its validity is as follows:

- If the match is not found inside one of the required zones, discard it.
- If the match does not meet the minimum score requirement, discard it.

- If duplicates are allowed:
  - If the instance of the match is allowable, count this instance and return the match.
  - Otherwise, count this instance and discard it.
- If duplicates are not allowed for the entity field:
  - If the matched text has been found before, discard it.
  - Otherwise, if the instance of the match is allowable, count this instance and return the matched text.
  - If the instance of the match is not allowable, count this instance and discard it.

Order of Returned Matches

When multiple fields in a document are selected for parsing, the field order in which matches are returned is as follows:

- `DREREFERENCE`
- `DRETITLE`
- `DRECONTENT`
- Any remaining fields in the order in which they are specified.

Chapter 6: Standard Grammars

This chapter contains specific information concerning the standard grammars that come with Eduction.

File Names

File names consist of up to four parts:

- **Basic entity type**. For example, *place*, *number,* or *person*.
- **Further detail on the basic type**. For example, *malefirstname* or *ss* for Social Security number. This part is optional, and is preceded by an underscore.
- **Language**. The three-character ISO 639-2/B code in which the grammar was written. For example, *eng* for English. It is preceded by an underscore.
- **Country**. The two-character ISO 3166-1 code describing the country for which the grammar was written. For example: *us* for the United States. This part is optional if the grammar does not target a specific country (for example, a credit card number). It is preceded by an underscore.

> **Note:** Entity names follow the same four-part structure, except for the basic type. The further detail and language/country parts are separated by forward slashes. The language code and the optional country code are concatenated.

Sentiment Grammars

Eduction includes standard grammars designed to identify those phrases in a passage of text that indicate positive or negative sentiment. These grammars can also identify which sentiments are expressed for which topics.

The sentiment grammar files also have ‘lite’ counterparts. These can process data up to twice as fast compared to the full versions, depending on language. The ‘lite’ versions are identical to the full versions in most respects, but they do not support components or user modification. HPE recommends that you use the ‘lite’ versions except in cases where you want to enable components or modify the built-in dictionaries.

The ‘lite’ versions are distinguished from the full versions by the addition of *lite* to the file name, preceded by an underscore. For example, the file name of the Chinese sentiment grammar file is `sentiment_chi.ecr`, and the file name of the ‘lite’ version is `sentiment_chi_lite.ecr`.

All sentiment analysis grammar files except `sentiment_basic_eng.ecr` now support components. You can extract the SENTIMENT and TOPIC components in most matches.

Polarity Scoring

The Arabic, Chinese, Czech, English, French, German, Italian, Polish, Portuguese, Russian, Spanish, and Turkish sentiment grammars (`sentiment_ara.ecr`, `sentiment_chi.ecr`, `sentiment_cze.ecr`, `sentiment_eng.ecr`, `sentiment_fre.ecr`, `sentiment_ger.ecr`, `sentiment_ita.ecr`, `sentiment_pol.ecr`, `sentiment_por.ecr`, `sentiment_rus.ecr`, `sentiment_spa.ecr`, and `sentiment_tur.ecr`) support *polarity scoring*. This is a number, usually between 0.50 and 1.50, that represents the strength of the sentiment in the matched phrase. For example:

- a strongly positive or negative phrase might have a score of 1.35
- a typical phrase might have a score of 1.00
- a match where the sentiment is weak or ambiguous might have a score of 0.60

You can edit the user modification files (`sentiment_user_ara.xml`, `sentiment_user_chi.xml`, `sentiment_user_cze.xml`, `sentiment_user_eng.xml`, `sentiment_user_fre.xml`, `sentiment_user_ita.xml`, `sentiment_user_pol.xml`, `sentiment_user_por.xml`, `sentiment_user_rus.xml`, `sentiment_user_spa.xml`, and `sentiment_user_tur.xml`) to increase the scores of words in the dictionaries. For example, add the following on a new line in the user modification file to modify the existing entry `flexible` so that it has a score of `1.23`:

```
"      <entry score="1.23" headword="flexible"/>"
```

Note: `sentiment_basic_eng.ecr` does not support polarity scoring.

Verb Sentiment Transitivity

The sentiment analysis files for Arabic, Chinese, Czech, English, French, German, Italian, Polish, Portuguese, Russian, Spanish, and Turkish (`sentiment_ara_ecr`, `sentiment_chi.ecr`, `sentiment_cze.ecr`, `sentiment_eng.ecr`, `sentiment_fre.ecr`, `sentiment_ger.ecr`, `sentiment_ita.ecr`, `sentiment_pol.ecr`, `sentiment_por.ecr`, `sentiment_rus.ecr`, `sentiment_spa.ecr`, and `sentiment_tur.ecr`) support *verb sentiment transitivity*. This enables the TOPIC components of the matches to determine what the sentiment is about with more accuracy by using advanced contextual understanding of whether that sentiment is being expressed about the subject or object of the sentence. For example, given two matches, `x likes y` and `x wins at y`, the grammar files can determine that the first match is a positive statement about `y`, whereas the second match is a positive statement about `x`.

Place Name Disambiguation

Ambiguous names in all place grammars have been given a score of 0.98 so that you can filter them out by setting `EntityMinScore`*`N`* to **`0.99`**. For example, if you want to use the `place/state/engau` entity to extract Australian state names using the `place_engau.ecr` grammar file, you can set `EntityMinScore`*`N`* to **`0.99`** to filter out ambiguous names such as *Victoria*.

Standard Grammar – Compiled

The following sections list the compiled grammar files included with Eduction.

Note: All the Chinese grammar files support traditional Chinese.

A

File	Entity	Description
address_au.ecr	address/postcode/au address/state_postcode/au	Australian postal codes. For example, *2600*. Australia state or territory, and postal code. For example, *NSW 2060*.
	address/city_state_postcode/au address/au	Australian city, state or territory, and postal code. For example, *North Sydney, NSW 2060*. Any Australian address. For example: *Shop 17, Winnellie Shopping Centre, 347 Stuart Hwy, Winnellie, NT, 0820*. *P.O.Box 27, Armadale North, Victoria, 3143, AUSTRALIA*. *121 North Seal Way, Cocos Keeling Islands, WA, 6799*. Eduction supports all common delimiters, including newlines.
address_ca.ecr	address/postcode/ca address/region_postcode/ca	Canadian postal codes. For example, *T2P-0B4*, *T2P0B4*, or *T2P 0B4*.

File	Entity	Description
	address/city_region_postcode/ca	Canadian province or territory, and postal code. For example, *Alberta, T2P0B4*.
	address/ca	Canadian city, province or territory, and postal code. For example, *Calgary, Alberta, T2P 0B4*.
		Any Canadian address. For example: *240 4th Avenue S.W., Suite 600, Calgary, Alberta T2P 4H4, Canada*.
		124 Av de la Peine, Montreal QC, H3Z 2Y7.
		Suite 600, 222-3rd Ave S.W., Calgary Alberta, T2P 0B4.
		Eduction supports all common delimiters, including newlines.
address_cn.ecr	address/pc/chicn	Chinese postal code. For example, *266033*.
	address/chicn	Any Chinese address. For example, 中国，山东省，青岛市 香港东路6号，5号楼，8号室 李小方 (先生)收.
	address/engcn	A Chinese address in English. For example. *63 Renmin Lu, Qingdao Shi, 266033 Shandong, China*.
	address/cn	A Chinese address in Chinese or English.
address_de.ecr	address/postcode/de	German postal code. For example, *80639*.
	address/postcode_city/de	German postal code, and city. For example, *80639, München*.
	address/de	Any German address. For example: *Hewlett-Packard-Straße 1, 61352, Bad Homburg vor der Höhe*.
		Postfach 10 01 65, 32547, Bad Oeynhausen, GERMANY.
		Grüner Weg 6, 61169, Friedberg, GERMANY.

File	Entity	Description
		Eduction supports all common delimiters, including newlines.
address_eng.ecr	address/strnum/eng	Street numbers. For example, *12a* or *14-17B*.
	address/pobox/eng	Post office box numbers. For example, *PO Box 26*.
	address/pmb/eng	Private mail box number. For example, *Private Mail Box 26*.
	address/pmb_or_pobox/eng	Post office box or private mail box number.
	address/street_pre/eng	Special street type that prefixes street numbers. For example, *Highway Contract, HC*.
	address/street_hwy/eng	Highway. For example, *City Route*.
	address/street_grid/eng	Grid address. For example, *400W350N*.
	address/street/eng	A street. For example, *Cowley Road* or *5th Street NW*.
	address/street_corner/eng	A street corner. For example, *Corner King Street & Queen Street*.
	address/street_all/eng	Any street For example, *12a Carlisle Lane*.
	address/suite/eng	Suite number. For example, *Suite 1*.
	address/floor/eng	Floor or level number. For example, *3rd Floor*, *Second Floor*, *Level 8*.
	address/floor_or_suite/eng	A floor or suite number.
	address/unitshipmil/eng	A military address analogous to a street address.
	address/building/eng	A building. For example, *Spear Tower*.
address_es.ecr	address/postcode/es	Spanish postal code. For example, *19208*.
	address/postcode_city/es	Spanish postal code and city. For example, *19208 Guadalajara*.
	address/es	

File	Entity	Description
		Any Spanish address. For example: *Av. de las Cortes de Cádiz, s/n, C. C. El Corte Inglés, 11011, Cádiz*. *Avda. Alfonso XIII, 6, Santander, España*. *Calle de la Fundición, 3, 33206, Gijón, Spain*. Eduction supports all common delimiters, including newlines.
address_fr.ecr	address/postcode/fr address/postcode_city/fr address/fr	French postal codes. For example, *75008*. French postal code, city, and optional CEDEX. For example, *75008, Paris*. Any French address. For example: *3, Avenue Denis Semeria, Saint-Jean-Cap-Ferrat, Provence-Alpes-Côte d'Azur, 06230, France*. *950 route des Colles - BP 27, 06901 Valbonne Sophia Antipolis*. *Bât G1 147 r Oberkampf, 75011 PARIS*. Eduction supports all common delimiters, including newlines.
address_fre.ecr	address/strnum/fre address/pobox/fre address/park/fre address/building/fre address/delivery_point/fre address/street_type/fre address/street/fre	A street number. For example, *12a* or *14-17B*. Post office box number in French. For example, *Boite Postale 26*. A business park in French. For example, *Technopark de Marseille*. A building. For example, *Château de Chambord*. A delivery point in French. For example, *BÂTIMENT 15*.

File	Entity	Description
	address/street_all/fre address/house_type/fre	A street type in French. For example, *Rue*. A street in French. For example, *Rue Pierre Charron*. Any street in French. A house type in French. For example, *Residence*.
address_gb.ecr	address/postcode/gb address/city_county_postcode/gb address/gb	United Kingdom postal codes. For example, *GY9 3UX*. UK city, optional county/country name, post code, and optional place name. For example, *Cambridge, CB4 0WZ*. Any United Kingdom address. For example: *Cambridge Business Park, Cowley Road, Cambridge, CB4 0WZ*. *12-14 The Diamond, Londonderry, Northern Ireland, BT48 6HW*. *105 Piccadilly, (First Floor), London, W1J 7NJ*. *Unit D, Acorn Business Park, Ling Road, Tower Park, Poole, Dorset, BH12 4NZ*. *44 Dorset Road, Providenciales, TURKS AND CAICOS ISLANDS*. Eduction supports all common delimiters, including newlines.
address_ger.ecr	address/strnum/ger address/pobox/ger address/street/ger	A street number. For example, *12a*. A post office box number in German. For example, *Postfach 26*. A street in German. For example, *12 Romanstr*.
address_it.ecr	address/postcode/it	Italian postal code. For example, *12345* or *IT-98765*.

File	Entity	Description
	address/postcode_city/it	Italian postal code and city. For example, *52100 Arezzo*.
	address/it	Any Italian address. For example: *Strada del Masarone 67, 13900 Biella (MI)*. *Via Balbi 3 e 40 16126 Genova*. *Via Mascarella n° 21/3, 40131 Bologna, Italia*. Eduction supports all common delimiters, including newlines.
address_ita.ecr	address/strnum/ita	Italian street number. For example, *12a*.
	address/pobox/ita	A post office box number in Italian. For example, *Casella postale 26*.
	address/street_type/ita	A street type in Italian. For example, *Via* or *Lungomare*.
	address/street/ita	An entire street name in Italian. For example, *Via del Fosso de Dragoncello*.
address_jp.ecr	address/postcode/jp	Japanese postal code. For example, 青森市.
address_spa.ecr	address/strnum/spa	A street number. For example, *12a* or *14-17B.*
	address/pobox/spa	A post office box number in Spanish. For example, *Apartado de correos 26*.
	address/street_type/spa	A street type in Spanish or in another language spoken in Spain. For example, *Calle* or *Passeig*.
	address/street_name/spa	A Spanish name that may refer to a street. For example, *26 de Marzo de 1824* or *Trujillo*. These are often used for street names in South America without a street type such as *Calle*.
	address/street/spa	An entire street name in Spanish. For example, *Calle de La Habana*.
	address/business_area/spa	

File	Entity	Description
		A shopping centre or business park in Spanish. For example, *Parque Tecnológico de Andalucía*.
address_us.ecr	address/zipcode/us address/city_state_zipcode/us address/military/us address/us	U.S. ZIP codes. For example, *94070-1234.* U.S. city, state, and ZIP code. For example, *Chicago, IL 80803.* U.S. military address. For example, *Unit 45013, Box 2666, USAG J, APO AP 96338.* Any U.S. address. For example: *30 South Wacker Drive, 22nd Floor, Chicago, IL 60606.* *P.O. Box 29, Sometown, AL 12345.* *5758 West Las Positas Blvd, Suite 100, Pleasanton, CA 94588.* *1 Market Street, Spear Tower, Suite 1900, San Francisco, CA 94105.* Eduction supports all common delimiters, including newlines.
age_eng.ecr	age/all/eng	An age in English.
age_fre.ecr	age/all/fre	An age in French.

B

File	Entity	Description
bank.ecr	bank/engca bank/engb	Canadian banks. For example, *Canadian Imperial Bank of Commerce*. UK banks. For example, *HSBC*.

File	Entity	Description
	bank/engus	U.S. banks. For example, *Morgan Stanley*.

C

File	Entity	Description
company_chicn.ecr	company/all/chicn	Chinese company.
company_dutnl.ecr	company/top500/dutnl company/designator/dutnl	Top 500 Dutch companies. Dutch company identifiers.
company_engau.ecr	company/law/engau	Law firms in Australia.
company_engca.ecr	company/tsx60/engca company/TSXVenture50/engca company/all/engca	A Canadian TSX60 company. A Canadian TSX Venture 50 company. Any Canadian company.
company_enggb.ecr	company/LSE/enggb company/law/enggb company/ftse100/enggb company/all/enggb	A United Kingdom company listed on the London Stock Exchange. Law firms in the United Kingdom. A FTSE 100 United Kingdom company. Any United Kingdom company.
company_engjp.ecr	company/nikkei225/engjp company/all/engjp	A Nikkei225 Japanese company. Any Japanese company.
company_engus.ecr	company/fortune_1000_2008/engus company/sp500/engus company/major_company/engus	The 2008 list of Fortune 1000 companies. U.S. S&P 500 companies. Major U.S. companies.

File	Entity	Description
	company/law/engus	Law firms in the United States.
	company/fortune_500/engus	A company that has featured in the Fortune 500 list at any time since 2011.
	company/forbes_largest_private_companies2010/engus	The 2010 list of Forbes largest companies.
	company/all/engus	All U.S. companies.
company_frefr.ecr	company/CAC_40/frefr	A French CAC 40 company.
	company/CAC_40_stocksymbol/frefr	A French CAC 40 company stock symbol.
	company/CAC_next_20/frefr	A French CAC Next 20 company.
	company/CAC_next_20_stocksymbol/frefr	A French CAC Next 20 company stock symbol.
	company/CAC_mid_60/frefr	A French CAC Mid 60 company.
	company/CAC_small/frefr	A French CAC Small company.
	company/SBF_120/frefr	A French SBF 120 company.
	company/all/frefr	Any French company.
company_gerde.ecr	company/dax/gerde	A German DAX company.
	company/dax_stocksymbol/gerde	A German DAX company stock symbol.
	company/cdax/gerde	A German CDAX company.
	company/hdax/gerde	A German HDAX company.
	company/mdax/gerde	A German MDAX company.
	company/sdax/gerde	A German SDAX company.
	company/tecdax/gerde	A German TecDAX company.
	company/all/gerde	Any German company.

File	Entity	Description
company_jpnjp.ecr	company/nikkei225/jpnjp company/all/jpnjp	A Japanese Nikkei 225 company. Any Japanese company.
company_korkr.ecr	company/all/korkr	Any Korean company.
company_law_eng.ecr	company/law_sgl/eng company/law_multi/eng	Law firms with single-word names. Law firms with multiple-word names. When names include commas and ampersand characters, the entity includes up to three versions of the name: • full name • with commas removed • with commas and ampersand removed All suffixes are removed for data in these entities.
company_other_eng.ecr	company/designator/eng company/org_legal/eng company/common_end_word/eng company/non_name/eng company/business/eng	A company designator. For example, *Corp, Inc.* Legal practice extensions. For example, *LLC*, *PC*. A common company name end word. For example, *Partners, Bros*. A non-specific name used in a company name. For example, *American, National*. A business term in a company name. For example, *Resorts, Capital, Accountants*.
company_rusru.ecr	company/all/rusru	Any Russian company.

D

File	Entity	Description
date_chi.ecr	date/season/chi	The four seasons in Chinese.
	date/season_simplified/chi	The four seasons in simplified Chinese.
	date/solar_term/chi	The solar terms in Chinese.
	date/solar_term_simplified/chi	The solar terms in simplified Chinese.
	date/yyyy/chi	The year in Chinese.
	date/yyyy_simplified/chi	The year in simplified Chinese and ASCII numbers.
	date/mm/chi	The month in Chinese.
	date/mm_simplified/chi	The month in simplified Chinese and ASCII numbers.
	date/ddd/chi	The day of the week in Chinese.
	date/ddd_simplified/chi	The day of the week in simplified Chinese.
	date/rel_period/chi	A period relative to the current date in Chinese.
	date/rel_period_simplified/chi	A period relative to the current date in simplified Chinese.
	date/period/chi	A fixed period of time in Chinese.
	date/period_simplified/chi	A fixed period of time in simplified Chinese.
	date/rel_day/chi	A day relative to the current date in Chinese.
	date/rel_day_simplified/chi	A day relative to the current date in simplified Chinese.
	date/ddd_dd/chi	The day of the week and the day of the month in Chinese.
	date/ddd_dd_simplified/chi	The day of the week and the day of the month in simplified Chinese and ASCII numbers.
	date/ddd_mmdd/chi	The day of the week and the month and day in Chinese.

File	Entity	Description
	date/ddd_mmdd_simplified	The day of the week and the month and day in simplified Chinese and ASCII numbers.
	date/mmdd/chi	The month and day in Chinese.
	date/mmdd_simplified	The month and day in simplified Chinese and ASCII numbers.
	date/mmdd_ddd/chi	The month, day, and day of the week in Chinese.
	date/mmdd_ddd_simplified/chi	The month, day, and day of the week in simplified Chinese and ASCII numbers.
	date/yyyymmdd/chi	The year, month, and day in Chinese.
	date/yyyymmdd_simplified/chi	The year, month, and day in simplified Chinese and ASCII numbers.
	date/yyyymmdd_ddd/chi	The year, month, day, and day of the week in Chinese.
	date/yyyymmdd_ddd_simplified/chi	The year, month, day, and day of the week in simplified Chinese and ASCII numbers.
	date/lunar_mmdd/chi	The month and the day of the lunar calendar in Chinese.
	date/lunar_mmdd_simplified/chi	The month and the day of the lunar calendar in simplified Chinese and ASCII numbers.
	date/chi	A date in any format in Chinese.
	date/simplified/chi	A date in any format in simplified Chinese and ASCII numbers.
	date/day_and_time/chi	A time of day on a specific or relative date in Chinese.
	date/day_and_time_simplified/chi	A time of day on a specific or relative date in simplified Chinese and ASCII numbers.
date_eng.ecr	date/season/eng	The four seasons in English. For example, *Winter*, *Spring*.
	date/year/eng	

File	Entity	Description
	date/mmm/eng date/ddd/eng date/rel_period/eng date/rel_day/eng date/mmmdd/eng date/ddmmm/eng date/day_date/eng date/month_dd_year/eng date/dd_month_year/eng date/day_date_year/eng date/mmm_year/eng date/eng	A year in English, in any format. The month in English, written in full or in short form. For example, *September*, *Sept*. The day of the week in English. For example, *Monday*, *Tuesday*. A period relative to the current date in English. A day relative to the current date in English. The month and day in English. For example, *January 5th*, *January 5*, or *January the 5th*. The day and month in English. For example, *5th January*, *5 January*, or *5th of January*. The date preceded by the day of the week in English. For example, *Sat January 5*, *Saturday the 5th Jan*. The month, day, and year in English. For example, *January 5th, 2008*. The day, month, and year in English. For example, *5th January, 2008*. The date and year, preceded by the day of the week, in English. For example, *Saturday, January 5th, 2008*. The month and year in English. For example, *January 2008*. A date in any format in English. Supported formats include: Date and month, with optional day and optional year: • 04 Oct 2008 • 4th October 2008

File	Entity	Description
		• 4 Oct • 4th of October 2008 • October 4th 2008 • 4th Oct '08 • 04 OCTOBER '08 • Saturday, October the 4th • Sat 4th of Oct • SATURDAY 4 OCTOBER 2008 • SAT OCT 4 • Sat. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008 • 4.10.08 • 04/10/2008 • Saturday 4-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008 • 10/28/08 • OCT 28 2008
date_fre.ecr	date/season/fre date/ddd/fre date/mmm/fre date/year/fre date/ddmmm/fre	The seasons in French. For example, *l'Hiver*, *saison des pluies*. A day of the week, in French. For example, *Lundi*, *Mardi, VEN*. Month, written in full or in short form, in French. For example, *Septembre*, *Sept*.

File	Entity	Description
	date/day_date/fre date/date_year/fre date/day_date_year/fre date/mmm_year/fre date/fre	A year in any format. The day and month in French. For example, *5e Janvier*, *5 Janvier*. The day and month in French, preceded by the day of the week. For example, *Samedi, 5 Janvier*. The day, month, and year in French. For example, *5 Janvier, 2008*. The day, month, and year in French, preceded by the day of the week. For example, *Samedi, 5 Janvier, 2008*. The month and year in French. For example, *Janvier, 2008*. A date in any format in French. Supported formats include: Date and month, with optional day and optional year: • 04 OCT. 2008 • 4ième Octobre 2008 • 4 Oct • 4 10 '08 • 04 OCTOBRE '08 • Samedi, 4 Oct • SAMEDI 4 OCTOBRE 2008 • Sam. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008 • 04/10/2008

File	Entity	Description
		• Samedi 4-10-08 • 04102008 (years 1970-2029 only)
date_ger.ecr	date/ddd/ger	A day of the week in German. For example, *Montag*, *Dienstag*.
	date/mmm/ger	A month in German. For example, *März*.
	date/year/ger	A year in any format.
	date/ddmmyyyy_dotspace/ger	dd. mm. yyyy. For example, *5. 1. 2008*.
	date/ddmmm/ger	The day and month in German. For example, *5 Januar*.
	date/day_date/ger	The day and month in German, preceded by the day of the week. For example*, Samstag, 5. Januar*.
	date/date_year/ger	The day, month, and year in German. For example, *5. Januar, 2008*.
	date/day_date_year/ger	The day, month, and year in German, preceded by the day of the week. For example*, Samstag, 5. Januar, 2008*.
	date/ger	A date in any numeric format in German. Supported formats include: Date and month, with optional day and optional year: • 04 Okt 2008 • 4 OKTOBER 2008 • 4. okt • 4 Oktober '08 • 04 OCT. '08 • 04. 2. 2007 • Samstag, 03.2.2007 • SONNABEND 4 OKTOBER 2008

File	Entity	Description
		• SA 04 OKT • Sa. 4. Okt. 2008 Extra delimiter support for formats where the year is present: • 04_OKT_2008 • 04/10/2008 • SA. 04-Okt-2008 • 04102008 (years 1970-2029 only) • 28-10-2008
	date/mmm_year/ger	The month and year in German. For example, *Januar 2008*.
date_ita.ecr	date/season/ita	The seasons in Italian. For example, *la primavera*, *l’inverno*.
	date/ddd/ita	A day of the week in Italian. For example, *lunedì*, *MAR*.
	date/mmm/ita	A month in Italian. For example, *gen.*, *FEBBRAIO*.
	date/year/ita	A year in any format.
	date/ddmmm/ita	The day and month in Italian. For example, *5 di gennaio*.
	date/day_date/ita	The day and month in Italian, preceded by the day of the week. For example*, sabato 5 di gennaio.*
	date/date_year/ita	The day, month, and year in Italian. For example, *5 di gennaio del 2008*.
	date/day_date_year/ita	The day, month, and year in Italian, preceded by the day of the week. For example*, sabato 5 di gennaio del 2008*.
	date/ita	A date in any format in Italian. Supported formats include: Date and month, with optional day and optional year:

File	Entity	Description
		• 04 Ott 2008 • 4 OTTOBRE 2008 • 4 ott • 04 di Ottobre 2008 • 4 di Ott del '08 • 4 Ott. '08 • Venerdi', 03 di Ottobre • Sab 4 di Ott • VENERDÌ 03 DI OTTOBRE DEL 2008 • SAB 4 OTT • Sab. 4 Ott. 2008 Extra delimiter support for formats where the year is present: • 04_OTT_2008 • 04/10/2008 • Venerdì 3-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008
	date/mmm_year/ita	Month and year in Italian. For example, *gennaio del 2008*.
date_jpn.ecr	date/season/jpn	The seasons in Japanese.
	date/ddd/jpn	A day of the week in Japanese.
	date/mmm/jpn	A month in Japanese (Kanji, numerals and fullwidth numerals).
	date/year_gregorian/jpn	A year in the Gregorian calendar, in Japanese, in any format, with optional A.D./B.C.
	date/year_imperial/jpn	

File	Entity	Description
	date/mmmdd/jpn date/day_date/jpn date/date_year_gregorian/jpn date/date_year_imperial/jpn date/day_date_year_gregorian/jpn date/day_date_year_imperial/jpn date/jpn date/mmm_year_gregorian/jpn date/mmm_year_imperial/jpn	Japanese imperial calendar year from 1868 onwards, in any format. The month and day in Japanese. The day and month in Japanese, preceded by the day of the week. The year, month, and day in the Gregorian calendar, in Japanese. The year, month, and day in the Japanese imperial calendar, in Japanese. The day, month, and year in the Gregorian calendar, in Japanese, preceded by the day of the week. The day, month, and year in the Japanese imperial calendar, in Japanese, preceded by the day of the week. A date in any numeric format in Japanese. The month and year in the Gregorian calendar, in Japanese. The month and year in the Japanese imperial calendar, in Japanese.
date_numeric.ecr	date/dd	A day from 1 to 31.
	date/dd_fullwidth	A day from 1 to 31, in fullwidth characters.
	date/dd2	A day from 01 to 31.
	date/dd2_fullwidth	A day from 01 to 31, in fullwidth characters.
	date/mm	A month from 1 to 12.
	date/mm_fullwidth	A month from 1 to 12, in fullwidth characters.
	date/mm2	A month from 01 to 12.

File	Entity	Description
	date/mm2_fullwidth	A month from 01 to 12, in fullwidth characters.
	date/yy	The last two digits of the year. For example, *67*, *08*.
	date/yy_fullwidth	The last two digits of the year, in fullwidth characters.
	date/yyyy	A three- or four-digit year, from 100 to 2099.
	date/yyyy_fullwidth	A three- or four-digit year in fullwidth characters, from 100 to 2099.
	date/yyyy4	A four-digit year, from 1000 to 2099.
	date/yyyy4_fullwidth	A four-digit year in fullwidth characters, from 1000 to 2099.
	date/year	A year in any numerical format.
	date/year_fullwidth	A year in any numerical format in fullwidth characters.
	date/yyyymmddsep	yyyy-mm-dd. For example, *2008-10-28*.
	date/yyyymmdd	yyyymmdd. For example, *20081028*.
	date/yyyymmdd_safe	yyyymmdd for a date between 19700101 and 20291231. For example, *20081028*.
	date/yymmddsep	yy-mm-dd. For example, *08-10-28*.
	date/yymmdd	yymmdd. For example, *081028*.
	date/ddmmyyyysep	dd-mm-yyyy. For example, *28-10-2008*.
	date/ddmmyyyy	ddmmyyyy. For example, *28102008*.
	date/ddmmyyyy_safe	ddmmyyyy for a date between 01011970 and 31122029. For example, *28102008*.
	date/ddmmyysep	dd-mm-yy. For example, *28-10-08*.
	date/ddmmyy	ddmmyy. For example, *281008*.
	date/mmddyyyysep	mm-dd-yyyy. For example, *10-28-2008*.
	date/mmddyyyy	

File	Entity	Description
	date/mmddyyyy_safe date/mmddyysep date/mmddyy	mmddyyyy. For example, *10282008*. mmddyyyy for a date between 01011970 and 12312029. For example, *10282008.* mm-dd-yy. For example, *10-28-2008*. mmddyy. For example, *102808*.
date_por.ecr	date/season/por date/ddd/por date/mmm/por date/year/por date/ddmmm/por date/day_date/por date/date_year/por date/day_date_year/por date/por	The seasons in Portugese. For example, *Verão, Outono.* A day of the week in Portugese. For example, *Segunda-feira*, *Terça-feira, DOM*. A month in Portugese. For example, *Setembro*. A year in any format. The day and month in Portugese.For example, *5 de Janeiro*. The day and month in Portugese, preceded by the day of the week. For example, *Sábado 5 de Janeiro*. The day, month, and year in Portugese. For example, *5 de maio 2008*. The day, month, and year in Portugese, preceded by the day of the week. For example, *Sábado 5 de janiero de 2008*. Any date in Portugese. Supported formats include: Date and month, with optional day and optional year: • 04 Out. 2008 • 4 OUTUBRO 2008 • 04 de Outubro 2008 • 4 de Out de '08

File	Entity	Description
	date/mmm_year/por	• SÁB 04 OUT 2008 • Sábado, 04 de Outubro • Terça-feira 14 Out. 1947 • SÁBADO 04 DE OUTUBRO DE 2008 • Quinta-feira, 12 de Setembro de 2013 EC • 4 de Março de 2012 Extra delimiter support for formats where the year is present: • 04_OUT_2008 • 04/10/2008 • Quarta feira 30-12-1953 • 04102008 (years 1970-2029 only) • 28-10-2008 The month and year in Portugese. For example, *Junho de 2008.*
date_spa.ecr	date/season/spa date/ddd/spa date/mmm/spa date/year/spa date/ddmmm/spa date/date_year/spa date/day_date/spa date/day_date_year/spa	The seasons in Spanish. For example, *el invierno, la primavera.* A day of the week in Spanish. For example, *Lunes, Domingo*. A month in Spanish. For example, *Septiembre*. A year in any format. The day and month in Spanish. For example, *5 de enero*. The day, month, and year in Spanish. For example, *5 de enero 2008*. The day and month in Spanish, preceded by the day of the week. For example, *Sábado 5 de enero*.

File	Entity	Description
	date/mmm_year/spa date/spa	The day, month, and year in Spanish, preceded by the day of the week. For example, *Sábado 5 de enero de 2008*. The month and year in Spanish. For example, *Januar 2008*. Any date in Spanish. Supported formats include: Date and month, with optional day and optional year: • 04 Oct 2008 • 4 OCTUBRE 2008 • 4 OCT • 4 de Octubre 2008 • 4 de Oct de '08 • 04 OCT. '08 • Sábado, 04 de Octubre • Jueves, 12 de Septiembre de 2013 d. J.C. • SÁBADO 04 DE OCTUBRE DE 2008 • SAB 4 OCT • Sab. 4 Oct. 2008 Extra delimiter support for formats where the year is present: • 04_OCT_2008 • 04/10/2008 • Sábado 4-10-08 • 04102008 (years 1970-2029 only) • 28-10-2008

File	Entity	Description
datetime_advanced_eng.ecr	datetime/advanced_hms24/eng	Time in hh:mm:ss.ss ZZZ format (seconds, fractional seconds, and timezone are optional). For example, *04:35*, *18:56:00*, *21:42:56.45 +0100*.
	datetime/advanced_hms24_range/eng	Time range in hh:mm:ss.ss ZZZ format (seconds, fractional seconds, and timezone are optional). For example, *04:35-04:36*, *18:56:00-21:00:00*, *21:42:56.45 to 23:59:59.99 +0100*.
	datetime/advanced_hm24_dot/eng	Time in hh.mm format. For example, *04.56*.
	datetime/advanced_hm24_dot_range/eng	Time range in hh.mm format. For example, *04.56 to 12.34*.
	datetime/advanced_hm24tz_nosep/eng	Time in hhmm ZZZ format. For example, *2100 GMT*.
	datetime/advanced_hm24tz_nosep_range/eng	Time range in hhmm ZZZ format. For example, *2100-2330 GMT*.
	datetime/advanced_hm24_nosep/eng	Time in hhmm format, with higher scores if the number of minutes is a multiple of 5. For example, *2100*.
	datetime/advanced_hm24_nosep_range/eng	Time range in hhmm format, with higher scores if the number of minutes is a multiple of 5. For example, *2100-2330*.
	datetime/advanced_hms12/eng	Time in 12-hour h:mm:ss am/pm ZZZ format (seconds and timezone are optional). For example, *9:30am*, *9:30:00pm GMT*.
	datetime/advanced_hms12_range/eng	Time range in 12-hour h:mm:ss am/pm ZZZ format (seconds and timezone are optional). For example, *9:30-10:30am*, *9:30:00am to 9:30:00pm GMT*.
	datetime/advanced_hm12_noampm/eng	Time in 12-hour h:mm:ss ZZZ format without am or pm specified (seconds and timezone are optional). For example, *9:30*, *9:30:00 GMT*.
	datetime/advanced_hm12_noampm_range/eng	Time range in 12-hour h:mm:ss ZZZ format without am or
	datetime/advanced_hm12_dot/eng	
	datetime/advanced_hm12_dot_range/eng	

File	Entity	Description
	datetime/advanced_hm12_nosep/eng	pm specified (seconds and timezone are optional). For example, *9:30-10:30*, *9:30:00 to 9:30:00 GMT*.
	datetime/advanced_hm12_nosep_range/eng	Time in 12-hour h.mm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *6.30am*, *8.45 GMT*, *11.35*.
	datetime/advanced_namedtime/eng	Time range in 12-hour h.mm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *6-7.30am*, *8.45am-6.30pm GMT*, *11-12.35*.
	datetime/advanced_clocktime_loose/eng datetime/advanced_clocktime_strict/eng	Time in 12-hour hmm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *630am*, *845 GMT*, *1135*.
	datetime/advanced_clocktime/eng	Time range in 12-hour hmm am/pm ZZZ format (am, pm, and timezone are optional, but scores are lower without them, although multiples of 5 minutes are boosted). For example, *630-730am*, *845-945 GMT*, *1135 to 345*.
	datetime/nameddays_strict/eng	Times of the day with a specific name in the English language. For example, *noon*, *midnight*.
	datetime/nameddays_all/eng	Time of the day, or time range, described in English (low confidence, scores reduced). For example, *twelve*, *two to three*.
	datetime/advanced_yyyymmdd/eng	Time of the day, or time range, described in English (high confidence). For example, *twelve o'clock*, *two fifteen*, *ten past one*, *quarter to midnight*, *ten to ten forty-five*.
	datetime/advanced_yyyymmdd_nosep/eng datetime/advanced_ddmmyyyy/eng	Time of the day, or time range, described in English (high and low confidence, scored appropriately). For example, *twelve o'clock*, *two fifteen in the afternoon*, *ten past one*,

File	Entity	Description
	datetime/advanced_ddmmyy/eng datetime/advanced_ddmm/eng datetime/advanced_ddMmmyyyy/eng datetime/advanced_ddMmmyy/eng datetime/advanced_ddMmm/eng datetime/advanced_mmddyyyy/eng datetime/advanced_mmddyy/eng datetime/advanced_mmdd/eng datetime/advanced_Mmmddyyyy/eng datetime/advanced_Mmmddyy/eng datetime/advanced_Mmmdd/eng datetime/advanced_Mmmyyyy/eng datetime/advanced_textdate_withyear/eng datetime/advanced_textdate_noyear/eng datetime/advanced_textdate/eng	*quarter to midnight*, *ten to ten forty-five*, *twelve at night*, *two to three*. Specially named days (confident matched only). For example, *Christmas Day*, *Easter Monday*. Specially named days (high and low confidence matches, scored appropriately). For example, *Christmas*, *Easter*. Dates in yyyy-mm-dd, yyyy-Mmm-dd, yyyy.mm.dd, yyyy.Mmm.dd, yyyy/mm/dd, or yyyy/Mmm/dd formats. For example, *2008-10-28*, *2008.Oct.28*. Dates in yyyy-mm-dd, yyyy.mm.dd, or yyyy/mm/dd formats. For example, *20081028*. Dates in dd-m-yyyy, dd.m.yyyy, or dd/m/yyyy formats. For example, *28-10-2008*. Dates in dd-mm-yy, dd.mm.yy, or dd/mm/yy formats. For example, *28.10.08*. Dates in dd-mm, dd.mm, or dd/mm formats. For example, *28/10*. Dates in dd-Mmm-yyyy, dd.Mmm.yyyy, or dd/Mmm/yyyy formats. For example, 28-Oct-2008. Dates in dd-Mmm-yy, dd.Mmm.yy, or dd/Mmm/yy formats. For example, *28.Oct.08*. Dates in dd-Mmm, dd.Mmm, or dd/Mmm formats. For example, *28/Oct*. Dates in m-dd-yyyy, m.dd.yyyy, or m/dd/yyyy formats. For example, *10-28-2008*. Dates in mm-dd-yy, mm.dd.yy, or mm/dd/yy formats. For example, *10.28.08*.

File	Entity	Description
	datetime/advanced_reldate/eng	Dates in mm-dd, mm.dd, mm/dd formats. For example, *10/28*. Dates in Mmm-dd-yyyy, Mmm.dd.yyyy, or Mmm/dd/yyyy formats. For example, *Oct-28-2008*. Dates in Mmm-dd-yy, Mmm.dd.yy, or Mmm/dd/yy formats. For example, *Oct.28.08*. Dates in Mmm-dd, Mmm.dd, Mmm/dd formats. For example, *Oct/28*. Named month/year. For example, *Oct 2008*, *October of 2008*, *October '08*.
	datetime/advanced_date_and_time/eng	
	datetime/advanced_date_only/eng	A date, with a year, described in English. For example, *July 4th 2008*, *July the 4th 2008*, *The morning of Wednesday July fourth 2008*, *4th July 2008*, *the 4th of July 2008*, *In the morning on Wednesday fourth July 2008*, *Christmas Day 2012*, *Easter '02*.
	datetime/advanced_time_only/eng	A date, without a year, described in English. For example, *July 4th*, *July the 4th*, *The morning of Wednesday July fourth*, *4th July,the 4th of July*, *In the morning on Wednesday fourth July*, *Christmas Eve*, *Easter Day*.
	datetime/advanced/eng	A date, described in English (with or without a year). A day, or part of a day, relative to today, described in English. Less confident matches are scored lower. For example, *This morning*, *tomorrow evening*, *Today* (score 0.9), *yesterday* (score 0.9), *the day after tommorrow* [sic] (score 0.9), *Two weeks ago on Monday*, *This coming Tuesday AM*, *Two weeks on Wednesday afternoon*, *Tuesday week*, *Tomorrow fortnight*, *Two weeks ago last Monday*, *Monday last week*, *Tuesday this wk*, *Wednesday next*, *Not this Tuesday but next* (score 0.9),

File	Entity	Description
		Not last Monday but the one before (score 0.9), *4th* (score 0.5), *the fourth* (score 0.8), *Wednesday the fourth of next month*, *last month on Wednesday 4th*, *The first Sunday of next month*, *Second Monday in July*, *Last Tuesday of April but one*, *Monday morning* (score 0.9), *Tuesday* (score 0.7). Any date with time, in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date and time. Any date (without a time), in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date. Any time (without a date), in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a time. Any date, with optional time, in any recognized format. Relevant components are extracted. Score indicates the confidence that the matched text is a genuine reference to a date and time. You can use the `datetime.lua` script to standardize the output of these entities.

E

File	Entity	Description
ethnicity_eng.ecr	ethnicity/nationality/eng	A nationality. For example, *Andorran*, *Welsh*.
ethnicity_engca.ecr	ethnicity/aboriginal/engca ethnicity/population_group/engca	A Canadian aboriginal group. For example, *Inuit*. A Canadian population group. For example, *Arab*, *White*.
ethnicity_enggb.ecr	ethnicity/enggb ethnicity/identity_code/enggb	Ethnicity classification in England. For example, *Irish*, *Indian*. United Kingdom identity code. For example, *IC1*, *IC2*.
ethnicity_engus.ecr	ethnicity/races/engus ethnicity/races_lowercase/engus ethnicity/native_american/engus ethnicity/asian/engus ethnicity/pacific/engus ethnicity/hispanic/engus ethnicity/engus	A United States race. For example, *Japanese*, *White*. A U.S. race in lowercase. For example, *japanese*, *white*. A U.S. native. For example, *Cherokee*, *Lambee*. A U.S. ethnicity of Asian origin. For example, *Pakistani*, *Korean*. A U.S. ethnicity of Pacific origin. For example, *Fijian*, *Tongan*. A U.S. ethnicity of Hispanic origin. For example, *Cuban*, *Spanish*. Any U.S. ethnicity.
ethnicity_fre.ecr	ethnicity/nationality/fre ethnicity/ethnic_groups/fre	Nationality in French. For example, *Andorrane*, *Vietnamien*. Ethnic groups in the French language. For example, *Africain*, *Autres*.

G

File	Entity	Description
gender_eng.ecr	gender/gender_word/eng gender/gender_context/eng gender/all/eng	A word that describes a family relation or gender in English. For example, *lady, father*. A gender in the context of English language. A gender in the English language, either in a word or in context.
gender_fre.ecr	gender/gender_word/fre gender/gender_context/fre gender/all/fre	A word that describes a family relation or gender in French. For example, *Dame*, *voisines*. A gender in the context of French language. A gender in the French language, either in a word or in context.
gender_ger.ecr	gender/gender_word/ger gender/gender_context/ger gender/all/ger	A word that describes a family relation or gender in German. For example, *mensch*, *Frau*. A gender in the context of German language. A gender in the German language, either in a word or in context.
gov_chicn.ecr	org/gov/chicn	A Chinese government agency.
gov_engca.ecr	org/gov/engca	A Canadian government agency.

H

File	Entity	Description
holiday_ca.ecr	holiday/statutory/engca	Statutory Canadian holidays in English. For example,

File	Entity	Description
	holiday/statutory/freca holiday/statutory/ca holiday/federal/engca holiday/federal/freca holiday/federal/ca holiday/statother/engca holiday/statother/freca holiday/statother/ca	*Good Friday.* Statutory Canadian holidays in French. For example, *Le vendredi saint.* Statutory Canadian holidays, in English or French. Federal Canadian holidays in English. For example, *Victoria Day.* Federal Canadian holidays in French. For example, *La fête de la Reine.* Federal Canadian holidays, in English or French. Other statutory Canadian holidays in English. For example, *Family Day.* Other statutory Canadian holidays in French. For example, *La fête du Travail.* Other statutory Canadian holidays, in English or French.
	holiday/alberta/engca holiday/britishcolumbia/engca holiday/manitoba/engca holiday/newbrunswick/engca holiday/newfoundlandlabrador/engca holiday/northwestterritories/engca holiday/novascotia/engca holiday/nunavut/engca holiday/ontario/engca holiday/princeedwardisland/engca	Holidays for each Canadian province and territory in English.

File	Entity	Description
	holiday/quebec/engca holiday/saskatchewan/engca holiday/yukon/engca holiday/prov_terr/engca holiday/other/engca holiday/ca	Holidays for Canadian provinces and territories in English. Other Canadian holidays and observances in English. All Canadian holidays.
holiday_enggb.ecr	holiday/bank_holiday/enggb holiday/holiday/enggb	British Bank Holiday name. Traditional days celebrated. For example, *Mother's Day*.
holiday_engus.ecr	holiday/federal/engus holiday/traditional/engus holiday/engus	U.S. federal holidays. For example, *Memorial Day*. Traditional U.S. days celebrated. For example, *Mother's Day*. All U.S. holidays.

I

File	Entity	Description
internet.ecr	internet/host_domain internet/host_ip/ipv4 internet/host_ip/ipv6 internet/host_ip/ipv4mapped internet/host_ip	A host name. For example, *www.myhost.com*. An IPv4 IP address. For example, *127.0.0.1*. An IPv6 IP address. For example, *1234:5678:90AB:CDEF.* An IPv4-mapped IP address. For example, *::FFFF:129.144.52.38*.

File	Entity	Description
	internet/addr_host internet/addr_email internet/addr_email_mailto internet/addr_https internet/addr_file internet/addr_ftp internet/addr_news internet/addr_telnet internet/addr_gopher	Any IP address. Host address. For example, *www.myhost.com* or *192.231.21.2*. Email address. For example, *jsmith@mailserver.com*. Email address with *mailto:* prefix. For example, *mailto:jsmith@mailserver.com*. HTTP or HTTPS address. file:// address. FTP address. news:// address. Telnet address. Gopher address.

J

File	Entity	Description
jobtitledicts_eng.ecr	person/titleprefix_camelcase/eng person/titleprefix_lowercase/eng person/titlesuffix_camelcase/eng person/titlesuffix_lowercase/eng person/govdep/engus person/titlegeneric_camelcase/eng person/titlegeneric_lowercase/eng	Job title prefix in camel case. For example, *Acting*. Job title prefix in lowercase. For example, *acting*. Job title suffix in camel case. For example, *Associate*, *Advisor*. Job title suffix in lowercase. For example, *educator*, *trainee*. U.S. government departments and abbreviations. For example, *National Security Council*, *FBI.*

File	Entity	Description
	person/titlefull_camelcase/eng	Generic job titles in camel case. For example, *Sales Assistant*. Generic job titles in lowercase. For example, *sales assistant*. Full job title in camel case, including prefixes and suffixes. For example, *Head of Customer Communications*.
	person/titlefull_lowercase/eng person/titlecorp/eng person/titlecorpabb/eng person/titlegov/eng person/titleroyal/eng person/titlepolitical/eng person/titlereligious/eng	Full job title in lower case, including prefixes and suffixes. For example, *head of customer communications*. Corporate job titles. For example, *Chief Financial Officer.* Abbreviated version of corporate job titles. For example, *CFO.* Government and cabinet titles. For example, *President*, *Secretary of Defense.* Royal titles. For example, *King.* Political titles. For example, *Foreign Minister*, *Governor.* Religious titles. For example, *Pope*, *Father*, *Imam.*

L

File	Entity	Description
languages.ecr	language/iso_lowercase language/all language/output_iso	Three-letter ISO 639-2/B language code. For example, *fin*, *ger*. Language name in a local language, English, or other major language.

File	Entity	Description
		Language name in a local language, English, or other major language (output is normalized to the ISO 639-2/B code)
legal_engus.ecr	legal/citsupr/engus legal/citcofa/engus	Supreme Court Citations. For example, *Roe v. Wade, 410 U.S. 113 (1973)*. Federal Court Reporter Citations. For example, *Universal City Studios, Inc. v. Corley, 273 F.3d 429 (2d Cir. 2001)*.

M

File	Entity	Description
measure_eng.ecr	measure/len/met/eng measure/len/usuk/eng measure/area/met/eng measure/area/usuk/eng measure/vol/met/eng measure/vol/usuk/eng measure/mass/met/eng measure/mass/usuk/eng	Metric measures of length. For example, *mm.*, *kilometre.* U.S. and UK measures of length. For example, *foot*, *mile*, *in.* Metric measures of area. For example, *sq. m.*, *square kilometres.* U.S. and UK measures of area. For example, *sq. in.*, *acres.* Metric measures of volume. For example, *microlitres*, *cubic centimetres.* U.S. and UK measures of volume. For example, *pinches*, *cups*, *gal.* Metric measures of mass. For example, *gram*, *tonnes.* U.S. and UK measures of mass. For example, *pound*, *lb.*
medical_condition.ecr	medical/disability/social_security/engus	Impairment for the purpose of disability evaluation under social security.

File	Entity	Description
	medical/disease_condition medical/lifestyle	Disease or medical condition. Lifestyle that relates to medical condition.
medical_drug.ecr	drug/brand drug/generic drug/medication	Trade name of medical drugs. Generic name of medical drugs. Description of a medication.
medical_healthcare_engus.ecr	healthcare/provider/AK/engus healthcare/provider/AL/engus healthcare/provider/AR/engus healthcare/provider/AZ/engus healthcare/provider/CA/engus healthcare/provider/CO/engus healthcare/provider/CT/engus healthcare/provider/DC/engus healthcare/provider/DE/engus healthcare/provider/FL/engus healthcare/provider/GA/engus healthcare/provider/HI/engus healthcare/provider/IA/engus healthcare/provider/ID/engus healthcare/provider/IL/engus healthcare/provider/IN/engus healthcare/provider/KS/engus	U.S. healthcare provider in Alaska. U.S. healthcare provider in Alabama. U.S. healthcare provider in Arkansas. U.S. healthcare provider in Arizona. U.S. healthcare provider in California. U.S. healthcare provider in Colorado. U.S. healthcare provider in Connecticut. U.S. healthcare provider in Washington, D.C. U.S. healthcare provider in Delaware. U.S. healthcare provider in Florida. U.S. healthcare provider in Georgia. U.S. healthcare provider in Hawaii. U.S. healthcare provider in Iowa. U.S. healthcare provider in Idaho. U.S. healthcare provider in Illinois. U.S. healthcare provider in Indiana. U.S. healthcare provider in Kansas.

File	Entity	Description
	healthcare/provider/KY/engus	U.S. healthcare provider in Kentucky.
	healthcare/provider/LA/engus	U.S. healthcare provider in Louisiana.
	healthcare/provider/MA/engus	U.S. healthcare provider in Massachusetts.
	healthcare/provider/MD/engus	U.S. healthcare provider in Maryland.
	healthcare/provider/ME/engus	U.S. healthcare provider in Maine.
	healthcare/provider/MI/engus	U.S. healthcare provider in Michigan.
	healthcare/provider/MN/engus	U.S. healthcare provider in Minnesota.
	healthcare/provider/MO/engus	U.S. healthcare provider in Missouri.
	healthcare/provider/MS/engus	U.S. healthcare provider in Mississippi.
	healthcare/provider/MT/engus	U.S. healthcare provider in Montana.
	healthcare/provider/NC/engus	U.S. healthcare provider in North Carolina.
	healthcare/provider/ND/engus	U.S. healthcare provider in North Dakota.
	healthcare/provider/NE/engus	U.S. healthcare provider in Nebraska.
	healthcare/provider/NH/engus	U.S. healthcare provider in New Hampshire.
	healthcare/provider/NJ/engus	U.S. healthcare provider in New Jersey.
	healthcare/provider/NM/engus	U.S. healthcare provider in New Mexico.
	healthcare/provider/NV/engus	U.S. healthcare provider in Nevada.
	healthcare/provider/NY/engus	U.S. healthcare provider in New York.
	healthcare/provider/OH/engus	U.S. healthcare provider in Ohio.
	healthcare/provider/OK/engus	U.S. healthcare provider in Oklahoma.
	healthcare/provider/OR/engus	U.S. healthcare provider in Oregon.
	healthcare/provider/PA/engus	U.S. healthcare provider in Pennsylvania.
	healthcare/provider/PR/engus	U.S. healthcare provider in Puerto Rico.

File	Entity	Description
	healthcare/provider/RI/engus	U.S. healthcare provider in Rhode Island.
	healthcare/provider/SC/engus	U.S. healthcare provider in South Carolina.
	healthcare/provider/SD/engus	U.S. healthcare provider in South Dakota.
	healthcare/provider/TN/engus	U.S. healthcare provider in Tennessee.
	healthcare/provider/TX/engus	U.S. healthcare provider in Texas.
	healthcare/provider/UT/engus	U.S. healthcare provider in Utah.
	healthcare/provider/VA/engus	U.S. healthcare provider in Virginia.
	healthcare/provider/VT/engus	U.S. healthcare provider in Vermont.
	healthcare/provider/WA/engus	U.S. healthcare provider in Washington.
	healthcare/provider/WI/engus	U.S. healthcare provider in Wisconsin.
	healthcare/provider/WV/engus	U.S. healthcare provider in West Virginia.
	healthcare/provider/WY/engus	U.S. healthcare provider in Wyoming.
	healthcare/provider/all/engus	Any U.S. healthcare provider.
medical_procedure.ecr	medical/blood_test	Blood test.
	medical/lab_test	Lab test.
	medical/surgical_procedure	Surgical procedure.
	medical/specialty	Medical specialty.
money_eng.ecr	money/fracunits	Fractional units of currency such as *Cent* or *Penny*.
	money/iso4217	ISO 4217 currency codes. For example, *AUD* or *USD*.
	money/currency	Currency name. For example, *Algerian dinar*.
	money/currencyabbrev	Abbreviated currency name. For example, *dinar* or *dollar*.
	money/denom_us	U.S. denominations. For example, *penny* or *quarter*.
	money/symbol	Currency symbols. For example, *$* or *€*.

N

File	Entity	Description
number_banking_au.ecr	number/bsb/au	Australian bank state branch number. For example, *34* or *985.*
number_banking_ca.ecr	number/cpa_transit_micr/ca number/cpa_transit_eft/ca number/cpa_transit/ca number/bankaccount/ca	Canadian Payments Association MICR transit number, in the format *BBBBB-AAA*, where *BBBB* is a five-digit code that identifies the branch, and *AAA* is a three-digit code that identifies the institution. For example, *25539-001*. Canadian Payments Association EFT transit number, in the format *0AAABBBBB*, where *AAA* is a three-digit code that identifies the institution, and *BBBB* is a five-digit code that identifies the branch. The first digit is always a leading zero. For example, *000125539*. Canadian Payments Association transit number. Canadian bank account number.
number_banking_de.ecr	number/sort_code/de number/bank_number/de	8-digit German bank sort code. For example, *10019610*. German bank account number.
number_banking_fr.ecr	number/bankaccount/fr	French bank account number.
number_banking_gb.ecr	number/sortcode/gb number/bankaccount/gb	United Kingdom bank sort code. *For example, 301007, 30-10-07, or 30 10 07.* United Kingdom bank account number.
number_banking_ie.ecr	number/sortcode/ie number/bankaccount/ie	Ireland bank sort code. For example, *906005*, *90-60-05*, or *90 60 05*. Ireland bank account number.

File	Entity	Description
number_banking_us.ecr	number/aba_micr/us number/aba_fraction/us number/aba_routing/us number/bankaccount/us	American Bankers Association MICR transit number, in the format *XXXXYYYYC*, where *XXXX* is the Federal Reserve Routing Symbol, *YYYY* is the ABA Institution Identifier, and *C* is the check digit. For example, *129131673*. American Bankers Association fraction transit number, in the format *PP-YYYY/XXXX*, where *PP* is a one-digit or two-digit prefix that represents the bank's check processing center location, *YYYY* is the ABA Institution identifier, and *XXXX* is the Federal Reserve Routing Symbol. American Bankers Association transit number. United States bank account number.
number_bsn_nl.ecr	number/bsn/nl	Dutch Citizen Service Numbers (burgerservicenummer). BSNs always consist of nine digits.
number_cc.ecr	number/cc12dn number/cc12dh number/cc12ds number/cc12 number/cc13dn number/cc13dh number/cc13ds number/cc13 number/cc14dn number/cc14dh	12-digit credit card numbers with no delimiters. 12-digit credit card numbers with hyphen delimiters. 12-digit credit card numbers with space delimiters. For example, All 12-digit credit card numbers. 13-digit credit card numbers with no delimiters. 13-digit credit card numbers with hyphen delimiters. 13-digit credit card numbers with space delimiters. All 13-digit credit card numbers. 14-digit credit card numbers with no delimiters. 14-digit credit card numbers with hyphen delimiters.

File	Entity	Description
	number/cc14ds	14-digit credit card numbers with space delimiters.
	number/cc14	All 14-digit credit card numbers.
	number/cc15dn	15-digit credit card numbers with no delimiters.
	number/cc15dh	15-digit credit card numbers with hyphen delimiters.
	number/cc15ds	15-digit credit card numbers with space delimiters.
	number/cc15	All 15-digit credit card numbers.
	number/cc16dn	16-digit credit card numbers with no delimiters.
	number/cc16dh	16-digit credit card numbers with hyphen delimiters.
	number/cc16ds	16-digit credit card numbers with space delimiters.
	number/cc16	All 16-digit credit card numbers.
	number/cc17dn	17-digit credit card numbers with no delimiters.
	number/cc17dh	17-digit credit card numbers with hyphen delimiters.
	number/cc17ds	17-digit credit card numbers with space delimiters.
	number/cc17	All 17-digit credit card numbers.
	number/cc18dn	18-digit credit card numbers with no delimiters.
	number/cc18dh	18-digit credit card numbers with hyphen delimiters.
	number/cc18ds	18-digit credit card numbers with space delimiters.
	number/cc18	All 18-digit credit card numbers.
	number/cc19dn	19-digit credit card numbers with no delimiters.
	number/cc19dh	19-digit credit card numbers with hyphen delimiters.
	number/cc19ds	19-digit credit card numbers with space delimiters.
	number/cc19	All 19-digit credit card numbers.
	number/ccdn	All credit card numbers with no delimiters.

File	Entity	Description
	number/ccdh number/ccds number/cc number/cc_amex number/cc_bankcard number/cc_china_union_pay number/cc_diners_club number/cc_discover number/cc_instapayment number/cc_jcb number/cc_laser number/cc_maestro number/cc_mastercard number/cc_solo number/cc_switch number/cc_visa	All credit card numbers with hyphen delimiters. All credit card numbers with space delimiters. Any credit card number. American Express credit card number. American Express credit card account numbers are 15 digits in lengths, and generally start with either *34* or *37*. For example, *378124403602370*. Bankcard credit card number (discontinued in 2006). China UnionPay credit card number. Most China UnionPay card numbers have prefixes from *620* to *625*, and range in length from 16 to 19 characters. Diners Club credit card number. Most Diners Club credit card numbers are 16 or 14 digits long. For example, *30544726571210* (Carte Blanche), *36072371463677* (International), or *5484308289255581* (North America). Discover credit card number. Discover credit card numbers start with *6011*, *622126* to *622925*, *644*, *645*, *646*, *647*, *648*, *649*, or *65*, and are 16 digits long. For example, *6011541256841963*. InstaPayment credit card number. InstaPayment credit card numbers start with either *637*, *638*, or *639*, and are 16 digits long. For example, *6393519709142682*. JCB credit card number. JCB credit card numbers consist of 16 digits. Either the first four digits must be *3088*, *3096*, *3112*, *3158*, or *3337*, or the first eight digits must be in the range *35280000* to *35899999*. For example, *3158745776935953*. Laser credit card number (discontinued in 2014). Laser

File	Entity	Description
		credit card numbers start with *6304*, *6706*, *6771*, or *6709*, and are between 16 to 19 digits long. For example, *6709682431878947*. Maestro credit card number. Maestro credit card numbers start with *5018*, *5020*, *5038*, *5893*, *6304*, *6759*, *6761*, *6762*, or *6763*, and are between 16 to 19 digits long (although they can have as few as 12 digits). For example, *5018452935461261*. Mastercard credit card number. Mastercard credit card numbers start with *51*, *52*, *53*, *54*, or *55*, and are between 16 to 19 digits long. 16-digit, 18-digit, or 19-digit Solo credit card number (discontinued in 2011). For example,*6331101999990016*. 16-digit, 18-digit, or 19-digit Switch credit card number (rebranded as Maestro). Switch credit card numbers begin with *4903*, *4905*, *4911*, *4936*, *564182*, *633110*, *6333*, or *6759*. Visa credit card number. Most Visa credit card numbers start with *4* and are 16 digits long; however, there are a few that consist of 13 digits. The numbers are always spaced in four groups of four digits each. For example, *4929 8198 5006 5312*. HPE Eduction supports the following credit card formats: • American Express • Bankcard • China Union Pay • InstaPayment • JCB • Laser

File	Entity	Description
		- DanKort - Diners Club Carte Blanche - Diners Club International - Diners Club enRoute - Discover - Maestro - Mastercard - Solo - Switch - Visa
number_dni_es.ecr	number/dni/es	Spanish DNI.
number_driverlic_ca.ecr	number/driverlic/AB/ca number/driverlic/NB/ca number/driverlic/ON/ca number/driverlic/BC/ca number/driverlic/NS/ca number/driverlic/SK/ca number/driverlic/PE/ca number/driverlic/MB/ca number/driverlic/QC/ca number/driverlic/YT/ca number/driverlic/NT/ca number/driverlic/ca	Driver's licence number for each Canadian province and territory. For more information on license number formatting requirements for each province, see http://www.deverusdemos.com/Help/FlashHelp/Search_Types/Formats_for_MVR_license_numbers.htm. .All Canadian driver's licence numbers.
number_driverlic_de.ecr	number/driverlic/de	German driver's licence number.
number_driverlic_fr.ecr	number/driverlic/fr	French driver's licence number.

File	Entity	Description
number_driverlic_gb.ecr	number/driverlic/gb	United Kingdom driving licence number.
number_driverlic_us.ecr	number/driverlic/AL/us number/driverlic/AK/us number/driverlic/AR/us number/driverlic/AZ/us number/driverlic/CA/us number/driverlic/CO/us number/driverlic/CT/us number/driverlic/DC/us number/driverlic/DE/us number/driverlic/FL/us number/driverlic/GA/us number/driverlic/HI/us number/driverlic/IA/us number/driverlic/ID/us number/driverlic/IL/us number/driverlic/IN/us number/driverlic/KS/us number/driverlic/KY/us number/driverlic/LA/us number/driverlic/MA/us number/driverlic/MD/us	Driver's licence number for each U.S. state. For more information on license number formatting requirements for each state, see http://www.deverusdemos.com/Help/FlashHelp/Search_Types/Formats_for_MVR_license_numbers.htm.

File	Entity	Description
	number/driverlic/ME/us number/driverlic/MI/us number/driverlic/MN/us number/driverlic/MO/us number/driverlic/MS/us number/driverlic/MT/us number/driverlic/NC/us number/driverlic/ND/us number/driverlic/NE/us number/driverlic/NH/us number/driverlic/NJ/us number/driverlic/NM/us number/driverlic/NV/us number/driverlic/NY/us number/driverlic/OH/us number/driverlic/OK/us number/driverlic/OR/us number/driverlic/PA/us number/driverlic/RI/us number/driverlic/SC/us number/driverlic/SD/us number/driverlic/TN/us number/driverlic/TX/us	

File	Entity	Description
	number/driverlic/UT/us number/driverlic/VA/us number/driverlic/VT/us number/driverlic/WA/us number/driverlic/WV/us number/driverlic/WI/us number/driverlic/WY/us number/driverlic/us	All U.S. driver’s licence numbers.
number/iban.ecr	number/ibandn/albania number/ibands/albania number/ibandn/andorra number/ibands/andorra number/ibandn/austria number/ibands/austria number/ibandn/bahrain number/ibands/bahrain number/ibandn/belgium number/ibands/belgium number/ibandn/bosniaherzegovina number/ibands/bosniaherzegovina	Undelimited (dn) or space-delimited (ds) International Bank Account Number (IBAN) for each country. For more information on IBAN formatting requirements for each country, see https://www.iban.com/structure.html.

File	Entity	Description
	number/ibandn/bulgaria number/ibands/bulgaria number/ibandn/costarica number/ibands/costarica number/ibandn/croatia number/ibands/croatia number/ibandn/cyprus number/ibands/cyprus number/ibandn/czechrepublic number/ibands/czechrepublic number/ibandn/denmark number/ibands/denmark number/ibandn/dominicanrepublic number/ibands/dominicanrepublic number/ibandn/estonia number/ibands/estonia number/ibandn/finland number/ibands/finland number/ibandn/france number/ibands/france number/ibandn/georgia number/ibands/georgia number/ibandn/germany	

File	Entity	Description
	number/ibands/germany	
	number/ibandn/gibraltar	
	number/ibands/gibraltar	
	number/ibandn/greece	
	number/ibands/greece	
	number/ibandn/hungary	
	number/ibands/hungary	
	number/ibandn/iceland	
	number/ibands/iceland	
	number/ibandn/ireland	
	number/ibands/ireland	
	number/ibandn/israel	
	number/ibands/israel	
	number/ibandn/italy	
	number/ibands/italy	
	number/ibandn/kazakhstan	
	number/ibands/kazakhstan	
	number/ibandn/kuwait	
	number/ibands/kuwait	
	number/ibandn/latvia	
	number/ibands/latvia	
	number/ibandn/lebanon	
	number/ibands/lebanon	

File	Entity	Description
	number/ibandn/liechtenstein number/ibands/liechtenstein number/ibandn/lithuania number/ibands/lithuania number/ibandn/luxembourg number/ibands/luxembourg number/ibandn/macedonia number/ibands/macedonia number/ibandn/malta number/ibands/malta number/ibandn/mauritania number/ibands/mauritania number/ibandn/mauritius number/ibands/mauritius number/ibandn/monaco number/ibands/monaco number/ibandn/montenegro number/ibands/montenegro number/ibandn/netherlands number/ibands/netherlands number/ibandn/norway number/ibands/norway number/ibandn/poland	

File	Entity	Description
	number/ibands/poland number/ibandn/portugal number/ibands/portugal number/ibandn/romania number/ibands/romania number/ibandn/sanmarino number/ibands/sanmarino number/ibandn/saudiarabia number/ibands/saudiarabia number/ibandn/serbia number/ibands/serbia number/ibandn/slovakrepublic number/ibands/slovakrepublic number/ibandn/slovenia number/ibands/slovenia number/ibandn/spain number/ibands/spain number/ibandn/sweden number/ibands/sweden number/ibandn/switzerland number/ibands/switzerland number/ibandn/tunisia number/ibands/tunisia	

File	Entity	Description
	number/ibandn/turkey number/ibands/turkey number/ibandn/unitedarabemirates number/ibands/unitedarabemirates number/ibandn/unitedkingdom number/ibands/unitedkingdom number/ibandn number/ibands	All IBAN numbers without delimiters. All IBAN numbers with space delimiters.
number_insee_fr.ecr	number/insee/fr	French INSEE number. INSEE numbers are composed of 13 digits and a two-digit key. `Score="0.2"` is used for examples with unspecified months.
number_licenseplate_ca.ecr	number/licenseplate/AB/ca number/licenseplate/BC/ca number/licenseplate/MB/ca number/licenseplate/NB/ca number/licenseplate/NL/ca number/licenseplate/NT/ca number/licenseplate/NS/ca number/licenseplate/NU/ca number/licenseplate/ON/ca	Licence plate numbers for each Canadian province and territory.

File	Entity	Description
	number/licenseplate/PE/ca number/licenseplate/QC/ca number/licenseplate/SK/ca number/licenseplate/YT/ca number/licenseplate/ca	All Canadian licence plate numbers.
number_licenseplate_de.ecr	number/licenseplate/de	German vehicle number plate.
number_licenseplate_es.ecr	number/licenseplate/es	Spanish vehicle number plate.
number_licenseplate_fr.ecr	number/licenseplate/fr	French vehicle registration number.
number_licenseplate_gb.ecr	number/licenseplate/gb	United Kingdom vehicle registration number.
number_licenseplate_us.ecr	number/licenseplate/AL/us number/licenseplate/AK/us number/licenseplate/AR/us number/licenseplate/AZ/us number/licenseplate/CA/us number/licenseplate/CO/us number/licenseplate/CT/us number/licenseplate/DE/us number/licenseplate/DC/us number/licenseplate/FL/us number/licenseplate/GA/us number/licenseplate/HI/us	Licence plate numbers for each U.S. state.

File	Entity	Description
	number/licenseplate/IA/us number/licenseplate/ID/us number/licenseplate/IL/us number/licenseplate/IN/us number/licenseplate/KS/us number/licenseplate/KY/us number/licenseplate/LA/us number/licenseplate/MA/us number/licenseplate/MD/us number/licenseplate/ME/us number/licenseplate/MI/us number/licenseplate/MN/us number/licenseplate/MO/us number/licenseplate/MS/us number/licenseplate/MT/us number/licenseplate/NC/us number/licenseplate/ND/us number/licenseplate/NE/us number/licenseplate/NH/us number/licenseplate/NJ/us number/licenseplate/NM/us number/licenseplate/NV/us number/licenseplate/NY/us	

File	Entity	Description
	number/licenseplate/OH/us	
	number/licenseplate/OK/us	
	number/licenseplate/OR/us	
	number/licenseplate/PA/us	
	number/licenseplate/RI/us	
	number/licenseplate/SC/us	
	number/licenseplate/SD/us	
	number/licenseplate/TN/us	
	number/licenseplate/TX/us	
	number/licenseplate/UT/us	
	number/licenseplate/VA/us	
	number/licenseplate/VT/us	
	number/licenseplate/WA/us	
	number/licenseplate/WV/us	
	number/licenseplate/WI/us	
	number/licenseplate/WY/us	
	number/licenseplate/us	United States license plate number.
number_mac_address.ecr	number/EUI48dh number/EUI48dc number/EUI48 number/EUI64dh	MAC address in EUI-48 format (hyphen-separated). For example, *01-23-45-67-89-Ab* MAC address in EUI-48 format (colon-separated). For example, *01:23:45:67:89:Ab* MAC address in EUI-48 format. For example, *01-23-45-*

File	Entity	Description
	number/EUI64dc	*67-89-Ab*
	number/EUI64	MAC address in EUI-64 format (hyphen-separated). For example, *01-23-45-67-89-ab-CD-eF*
		MAC address in EUI-64 format (colon-separated). For example, *01:23:45:67:89:ab:CD:eF*
		MAC address in EUI-64 format. For example, *01-23-45-67-89-ab-CD-eF*
number_ni_gb.ecr	number/nids/gb	UK National Insurance number with space delimiters.
	number/nidn/gb	UK National Insurance number without delimiters.
	number/nidh/gb	UK National Insurance number with hyphen delimiters.
	number/ni/gb	Any UK National Insurance number. The format of the number is two prefix letters, six digits, and one suffix letter.
number_passport_engca.ecr	number/passport_number/engca	Canadian passport number (in any context).
	number/passport_context/engca	Canadian passport number (when found in English-language context).
number_passport_enggb.ecr	number/passport_context/enggb	UK passport number (when found in English-language context).
number_passport_engus.ecr	number/passport_context/engus	U.S. passport number (when found in English-language context).
number_passport_freca.ecr	number/passport_number/freca	French Canadian passport number (in any context).
	number/passport_context/freca	French Canadian passport number (when found in French-language context).

File	Entity	Description
number_passport_frefr.ecr	number/passport_number/frefr number/passport_context/frefr	French passport number (in any context). French passport number (when found in French-language context).
number_passport_gerde.ecr	number/passport_context/gerde	German passport number (when found in German-language context).
number_phone_au.ecr	phone/landline/au phone/mobile/au phone/other/au phone/all/au	A complete landline phone number in Australia A complete mobile phone number in Australia. A complete 08- or 09- phone number in Australia. Any complete phone number in Australia. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_be.ecr	phone/landline/be phone/mobile/be phone/other/be phone/all/be	A complete landline phone number in Belgium. A complete mobile phone number in Belgium. A complete 08- or 09- phone number in Belgium. Any complete phone number in Belgium. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_ca.ecr	phone/numds/ca phone/numdh/ca phone/numdd/ca phone/numdn/ca	A numeric-only Canadian phone number, delimited by spaces. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A numeric-only Canadian phone number, delimited by hyphens. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A numeric-only Canadian phone number, delimited by dots. To ensure that this entity performs correctly, set

File	Entity	Description
	phone/num/ca phone/alphanumds/ca phone/alphanumdh/ca phone/alphanumdd/ca phone/alphanumdn/ca phone/alphanum/ca	`TangibleCharacters` to include '+' and '('. An undelimited, numeric-only Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. Any numeric-only Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. An alphanumeric Canadian phone number, delimited by spaces. An alphanumeric Canadian phone number, delimited by hyphens. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. An alphanumeric Canadian phone number, delimited by dots. An undelimited, alphanumeric Canadian phone number. Any alphanumeric Canadian phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_cn.ecr	phone/landline/cn phone/mobile/cn phone/tollfree/cn phone/all/cn	A Chinese landline phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A Chinese mobile phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A Chinese toll free phone number. Any Chinese phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

File	Entity	Description
number_phone_de.ecr	phone/landline/de phone/mobile/de phone/other/de phone/all/de	A complete landline phone number in Germany. A complete mobile phone number in Germany. A complete freephone or premium phone number in Germany. Any complete German phone number. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_es.ecr	phone/landline/es phone/mobile/es phone/other/es phone/all/es	A complete landline phone number in Spain. A complete mobile phone number in Spain. A complete freephone or premium phone number in Spain. Any complete phone number in Spain. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_fr.ecr	phone/landline/fr phone/mobile/fr phone/other/fr phone/all/fr	A complete landline phone number in France. A complete mobile phone number in France. A complete 08- or 09- phone number in France. Any complete phone number in France. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_gb.ecr	phone/areacode/gb phone/landline/gb phone/mobile/gb	United Kingdom area code. A complete landline phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

File	Entity	Description
	phone/freephone/gb phone/business/gb phone/non_geographic/gb phone/personal/gb phone/all/gb	A complete mobile phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A complete freephone phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A complete 08- or 09- phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A complete non-geographic phone number in the United Kingdom. For example, *0345 678 579 40.* To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A complete 070- phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. Any complete phone number in the United Kingdom. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_it.ecr	phone/landline/it phone/mobile/it phone/other/it phone/all/it	A complete landline phone number in Italy. A complete mobile phone number in Italy. A premium rate, freephone, or shared-cost phone number in Italy. Any complete phone number in Italy. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_lu.ecr	phone/landline/lu	A complete landline phone number in Luxembourg.

File	Entity	Description
	phone/mobile/lu phone/all/lu	A complete mobile phone number in Luxembourg. Any complete phone number in Luxembourg. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_nl.ecr	phone/landline/nl phone/mobile/nl phone/other/nl phone/all/nl	A complete landline phone number in the Netherlands. A complete mobile phone number in the Netherlands. A complete 08- or 09- phone number in the Netherlands. Any complete phone number in the Netherlands. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_pt.ecr	phone/landline/pt phone/mobile/pt phone/other/pt phone/all/pt	A complete landline phone number in Portugal. A complete mobile phone number in Portugal. Other complete phone number in Portugal. Any complete phone number in Portugal. To ensure that these entities perform correctly, set `TangibleCharacters` to include '+' and '('.
number_phone_us.ecr	phone/numds/us phone/numdh/us phone/numdd/us phone/numdn/us	A numeric-only U.S. phone number, delimited by spaces. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A numeric-only U.S. phone number, delimited by hyphens. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. A numeric-only U.S. phone number, delimited by dots. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('.

File	Entity	Description
	phone/num/us phone/alphanumds/us phone/alphanumdh/us phone/alphanumdd/us phone/alphanumdn/us phone/alphanum/us	An undelimited, numeric-only U.S. phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. Any numeric-only U.S. phone number. To ensure that this entity performs correctly, set `TangibleCharacters` to include '+' and '('. An alphanumeric U.S. phone number, delimited by spaces. An alphanumeric U.S. phone number, delimited by hyphens. An alphanumeric U.S. phone number, delimited by dots. An undelimited, alphanumeric U.S. phone number. Any alphanumeric U.S. phone number.
number_sin_ca.ecr	number/sindh/ca	Canadian social insurance number with dash delimiters.
	number/sinds/ca	Canadian social insurance number with space delimiters.
	number/sindn/ca	Canadian social insurance number without delimiters.
	number/sin/ca	Any Canadian social security number.
number_ss_us.ecr	number/ssdh/us	Social Security number with dash delimiters.
	number/ssdsh/us	Social Security number with soft hyphen delimiters.
	number/ssds/us	Social Security number with space delimiters.
	number/ssdnbs/us	Social Security number with non-breaking space delimiters.
	number/ssdn/us	Social Security number without delimiters.
	number/ss/us	Any Social Security number.
	number/medicareid/us	Medicare ID.

File	Entity	Description
number_swiftcode.ecr	number/swiftcode	Swift code.
number_telecoms.ecr	number/imei	International Mobile Station Equipment Identity number.
	number/imeisv	International Mobile Station Equipment Identity Software Version number.
	number/meid_hex	Mobile Equipment Identifier (hexadecimal format).
	number/iccid	Integrated Circuit Card Identifier number.
	number/imsi	International Mobile Subscriber Identity number.
	number/plmn	Public Land Mobile Network number.
	number/msisdn	Mobile Subscriber Integrated Services Digital Network number.
number_types_chi.ecr	number/one_to_nine/chi	The numbers one to nine in Chinese.
	number/zero_to_nine/chi	The numbers zero to nine in Chinese.
	number/zero_to_twelve/chi	The numbers zero to twelve in Chinese.
	number/zero_to_twenty_four/chi	The numbers zero to twenty-four in Chinese.
	number/one_to_thirty_one/chi	The numbers one to thirty-one in Chinese.
	number/zero_to_fifty_five/chi	The numbers zero to fifty-five in steps of five in Chinese.
	number/zero_to_fifty_nine/chi	The numbers zero to fifty-nine in Chinese.
	number/one_to_ninety_nine/chi	The numbers one to ninety-nine in Chinese.
	number/one_to_one_hundred/chi	The numbers one to one hundred in Chinese.
	number/all/chi	Large numbers in Chinese.
	number/num/chi	A simple string of digits that does not start with a zero in Chinese.
	number/digits/chi	String of digits in Chinese.
	number/fraction/chi	

File	Entity	Description
		A simple fraction consisting of two strings of digits that do not start with a zero. For example, *-12/13, 1/5*.
number_types_eng.ecr	number/num/eng number/ncomma/eng number/comma/eng number/sign/eng number/natural/eng number/int/eng number/real/eng number/fraction/eng number/fracalpha/eng number/fracnum/eng number/fracmixed/eng number/pcnt/eng number/suff/eng number/suffalpha/eng number/doz/eng number/alpha/eng number/bigalpha/eng number/bignum/eng	A simple string of digits that does not start with a zero. For example, *123*. A number without commas. For example, *123456.* A number with commas. For example, *123,456.* A sign. For example +, -, *plus*, *minus.* A natural number. For example, *123,456.* An integer. For example, *-123,456*, *minus 2*, *+20.* A real number. For example, *123.456*, *-123.456.* A simple fraction of two unsigned strings of digits that do not start with a zero. For example, *12/13*. An alphabetical fraction. For example, *one half*, *three sixths*, *three and five ninths.* Numeric fractions. For example, *-12/13.* Mixed fractions. For example, *1 twelfth*, *8 fourteenths*, *3 and five ninths.* Percent. For example, *100%*, *100 percent*, *12.78%.* Numbers with a suffix. For example, *1st*, *3rd.* Fractions with mixed alphabetical and numeric terms. For example, *a 12th*, *three 3rds.* Number based on dozen. For example, *half a dozen*, *2 dozen*, *three and a half dozen*. Alphabetical numbers less than 100. For example, *one*, *ten*, *thirty-one.*

File	Entity	Description
	number/big/eng number/sci/eng number/fullalpha/eng number/numord/eng number/num_plurals/eng	Big alphabetical numbers. Big numeric abbreviated numbers. A big number, alphabetical or numeric abbreviated. A number in scientific notation. For example, *1.23x10^11, 1.23E+5, 6.1^-3.* A fully written out number up to 999,999,999,999,999. For example, *one thousand two hundred and thirty four*. An ordinal number up to 999. For example, *thirty fourth*. Plural numbers. For example, *dozens, millions*.
number_types_fre.ecr	number/num/fre number/num_sep/fre number/digits/fre number/natural/fre number/fraction/fre number/int/fre number/real/fre number/numalpha/fre number/numord/fre	A simple string of digits that does not start with 0. For example, *123*. A number with separators. For example, *123.456.789*. A string of digits. For example, *00123*. A natural number. For example, *123.456.789* or *123456789.* A simple fraction of two unsigned strings of digits that do not start with a zero. For example, *12/13*. An integer. For example, *-123.456.789*; *moins 2*; *20*. A real number. For example, *123.456*, *-123.456*. A fully written out number up to 999. For example, *deux cent trente-quatre*. An ordinal number up to 999. For example, *trente quatrième*.
number_vin.ecr	number/vin/wmi number/vin/vds	The world manufacturer identifier section (3 characters) of a vehicle identification number.

File	Entity	Description
	number/vin/model_year number/vin/plant_code number/vin/seq_number number/vin/vis number/vin number/vin/anonymized	The vehicle descriptor section (6 characters) of a vehicle identification number. The model year character of a vehicle identification number. The plant code character of a vehicle identification number. The vehicle identifier section sequential number (6 characters) of a vehicle identification number. The vehicle identifier section (8 characters) of a vehicle identification number. A vehicle identification number (17 characters). An anonymized vehicle identification number (first 9 or 11 characters).

O

File	Entity	Description
organization.ecr	org/organization	An organization.

P

File	Entity	Description
person_name_chicn.ecr	person/femalefirstname_s/chicn	Popular simplified Chinese female first name.
	person/malefirstname_s/chicn	Popular simplified Chinese male first name.
	person/lastname_s/chicn	Popular simplified Chinese last name.

File	Entity	Description
	person/firstname/chicn person/namelastfirst/chicn	Chinese first name. Chinese last and first name.
person_name_dutnl.ecr	person/femalefirstname/dutnl person/malefirstname/dutnl person/firstname/dutnl person/surname/dutnl person/namefirstmiddlelast/dutn	Popular Dutch female first name. Popular Dutch male first name. Dutch first name. Dutch surname. Dutch first, optional middle, and last name.
person_name_engcn.ecr	person/femalefirstname/engcn person/femalefirstname_lowercase/engcn person/malefirstname/engcn person/malefirstname_lowercase/engcn person/lastname/engcn person/firstname/engcn person/namelastfirst/engcn person/namefirstlast/engcn	Popular Chinese female first name in English. Popular Chinese female first name in lowercase English. Popular Chinese male first name in English. Popular Chinese male first name in lowercase English. Popular Chinese last name in English. Chinese first name in English. Chinese last and first name in English. Chinese first and last name in English.
person_name_enggb.ecr	person/femalefirstname/enggb person/malefirstname/enggb person/lastname/enggb person/firstname/enggb person/namefirstlast/enggb person/namefirstmiddlelast/enggb	Popular UK female first name. Popular UK male first name. Popular UK last name. UK first name. UK first and last name. UK first, optional middle, and last name.

File	Entity	Description
person_name_enggr.ecr	person/femalefirstname/enggr person/malefirstname/enggr person/lastname/enggr person/firstname/enggr person/namefirstlast/enggr	Popular Greek female first name. Popular Greek male first name. Popular Greek last name. Greek first name. Greek first and last name.
person_name_engin.ecr	person/femalefirstname/engin person/malefirstname/engin person/lastname/engin person/firstname/engin person/namefirstlast/engin person/namefirstmiddlelast/engin	Popular Indian female first name. Popular Indian male first name. Popular Indian last name. Indian first name. Indian first and last name. Indian first, optional middle, and last name.
person_name_engjp.ecr	person/femalefirstname/engjp person/malefirstname/engjp person/lastname/engjp person/firstname/engjp person/namelastfirst/engjp	Popular Japanese female first name in English. Popular Japanese male first name in romanji. Popular Japanese last name in English. Japanese first name in English. Japanese last and first name in English.
person_name_engru.ecr	person/femalefirstname/engru person/malefirstname/engru person/lastname/engru person/firstname/engru person/namefirstlast/engru person/namefirstmiddlelast/engru	Popular Russian female first name in English. Popular Russian male first name in English. Popular Russian last name in English. Russian first name in English. Russian first and last name in English. Russian first, optional middle, and last name in

File	Entity	Description
		English.
person_name_engus.ecr	person/femalefirstname/engus	Popular U.S. female first name.
	person/malefirstname/engus	Popular U.S. male first name.
	person/lastname/engus	Popular U.S. last name.
	person/firstname/engus	U.S. first name.
	person/compoundlastname/engus	U.S. last name that might be compound.
	person/namefirstlast/engus	U.S. first and last name.
	person/namefirstmiddlelast/engus	U.S. first, optional middle, and last name.
	person/nameinitial/engus	U.S. initialed name.
	person/namelastsuffix/engus	Last name and suffix.
	person/namefirstneelast/engus	First name and maiden name.
	person/namelastcommafirst/engus	Full name in address book format (*Last Name, First Name*).
person_name_frefr.ecr	person/femalefirstname/frefr	Popular French female first name.
	person/malefirstname/frefr	Popular French male first name.
	person/lastname/frefr	Popular French last name.
	person/firstname/frefr	French first name.
	person/namefirstlast/frefr	French first and last name.
	person/namefirstmiddlelast/frefr	French first, optional middle, and last name.
person_name_gerde.ecr	person/femalefirstname/gerde	Popular German female first name.
	person/malefirstname/gerde	Popular German male first name.
	person/lastname/gerde	Popular German last name.

File	Entity	Description
	person/firstname/gerde	German first name.
	person/namefirstmiddlelast/gerde	German first, optional middle, and last name.
person_name_itait.ecr	person/femalefirstname/itait	Popular Italian female first name.
	person/malefirstname/itait	Popular Italian male first name.
	person/lastname/itait	Popular Italian last name.
	person/firstname/itait	Italian first name.
	person/namefirstlast/itait	Italian first and last name.
	person/namefirstmiddlelast/itait	Italian first, optional middle, and last name.
person_name_jpnjp.ecr	person/femalefirstname/jpnjp	Popular Japanese female first name.
	person/malefirstname/jpnjp	Popular Japanese male first name in kanj.
	person/lastname/jpnjp	Popular Japanese last name in kanj.
	person/firstname/jpnjp	Japanese first name.
	person/namelastfirst/jpnjp	Japanese last and first name.
person_name_norno.ecr	person/femalefirstname/norno	Popular Norwegian female first name.
	person/malefirstname/norno	Popular Norwegian male first name.
	person/lastname/norno	Popular Norwegian last name.
	person/firstname/norno	Norwegian first name.
	person/namefirstlast/norno	Norwegian first and last name.
	person/namefirstmiddlelast/norno	Norwegian first, optional extra given name, optional middle name, and last name.
person_name_rusru.ecr	person/femalefirstname_unambiguous	Common Russian female first name in Russian that rarely has any alternative meaning.
	person/malefirstname_unambiguous	Common Russian male first name in Russian that

File	Entity	Description
		rarely has any alternative meaning.
	person/femalefirstname/rusru	Russian female first name in Russian.
	person/femalelastname/rusru	Russian female last name in Russian.
	person/malefirstname/rusru	Russian male first name in Russian.
	person/malelastname/rusru	Russian male last name in Russian.
	person/firstname/rusru	Russian first name in Russian.
	person/lastname/rusru	Russian last name in Russian.
	person/fullname/rusru	Russian full name in Russian.
person_name_spaes.ecr	person/femalefirstname/spaes	Popular Spanish female first name.
	person/malefirstname/spaes	Popular Spanish male first name.
	person/lastname/spaes	Popular Spanish last name.
	person/firstname/spaes	Spanish first name.
	person/compoundlastname/spaes	Spanish compound last name.
	person/namefirstoptionallast/spaes	Spanish first and optional last name.
	person/namefirstlast/spaes	Spanish first and last name.
	person/namelastfirst/spaes	Spanish last name, comma, and first name.
	person/fullname	Spanish full name.
person_name_swese.ecr	person/femalefirstname/swese	Popular Swedish female first name.
	person/malefirstname/swese	Popular Swedish male first name.
	person/lastname/swese	Popular Swedish last name.
	person/firstname/swese	Swedish first name.
	person/namefirstlast/swese	Swedish first and last name.

File	Entity	Description
person_politician_engus.ecr	person/poli_hor/engus	Full names of members of the U.S. House of Representatives. For example, *Robert E. Cramer*, *Robert Cramer.*
	person/poli_last_hor/engus	Last name of House of Representatives members. For example, *Cramer.*
	person/poli_sen/engus	Full name of U.S. senate members.
	person/poli_last_sen/engus	Last name of U.S. senate members.
	person/poli_gov/engus	Full name of U.S. governors.
	person/poli_last_gov/engus	Last name of U.S. governors.
	person/poli_cabinet_gw_bush/engus	Full name of a member of the George W. Bush administration.
	person/poli_last_cabinet_gw_bush/engus	Last name of a member of the George W. Bush administration.
	person/poli_cabinet_obama/engus	Full name of a member of the Barack Obama administration.
	person/poli_last_cabinet_obama/engus	Last name of a member of the Barack Obama administration.
	person/poli_other_2012/engus	Full name of other currently active politician. For example, a Presidential nominee.
	person/poli_last_other_2012/engus	Last name of other currently active politician. For example, a Presidential nominee.
	person/poli_president/engus	Past and present U.S. Presidents.
	person/poli_title_hor/engus	Formal title for legislative members. For example, *Congressman Cramer.*
	person/poli_title_sen/engus	Formal title for senate members.
	person/poli_title_gov/engus	Formal title for governors.

File	Entity	Description
		Chinese National People's Congress delegate.
person_politician_jpnjp.ecr	person/politician/jpnjp	Japanese politician.
person_public_figure_chi.ecr	person/politician/chicn person/legislativecouncil/chihk person/entertainer/chi person/npc/chicn	Chinese politician. Hong Kong legislative council member. Chinese entertainer.
person_public_figure_eng.ecr	person/public_figure/eng	A list of public figures in English.
person_public_figure_jpn.ecr	person/public_figure/jpn	A list of public figures in Japanese.
person_salutation_eng.ecr	person/salutation/common/eng person/salutation/military/eng person/salutation/political/eng person/salutation/religious/eng person/salutation/nobility/eng person/salutation/eng	Common salutation. For example, *Mr*. Military salutation. Political salutation. Religious salutation. Salutation of nobility. Any salutation in English.
person_salutation_fre.ecr	person/salutation/fre	French salutations. For example, *Madame*, *Mlle*.
person_suffix_eng.ecr	person/suffixjr/eng person/suffixrmn/eng person/suffixacab/eng person/ suffixacam/eng person/suffixacad/eng personal/suffixprof/eng	Name suffixes. For example, *Jr.* Roman suffixes. For example, *III.* Academic suffix – Bachelor's. For example, *BA.* Academic suffix – Master's. For example, *MA.* Academic suffix – Doctoral. For example, *PhD*. Professional suffix. For example, *MD.*

File	Entity	Description
place_albal.ecr	place/city1/albal	Albanian settlement with over 100,000 inhabitants.
	place/city1_uppercase/albal	Albanian settlement with over 100,000 inhabitants, in uppercase.
	place/city2/albal	Albanian settlement with between 10,000 and 100,000 inhabitants.
	place/city2_uppercase/albal	Albanian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
	place/county/albal	Albanian county.
	place/county_uppercase/albal	Albanian county in uppercase.
	place/district/albal	Albanian district.
	place/district_uppercase/albal	Albanian district in uppercase.
place_albxk.ecr	place/city1/albxk	Kosovan settlement with over 100,000 inhabitants.
	place/city1_uppercase/albxk	Kosovan settlement with over 100,000 inhabitants, in uppercase.
	place/city2/albxk	Kosovan settlement with between 10,000 and 100,000 inhabitants.
	place/city2_uppercase/albxk	Kosovan settlement with between 10,000 and 100,000 inhabitants, in uppercase.
	place/district/albxk	Kosovan district.
	place/district_uppercase/albxk	Kosovan district in uppercase.
place_bosba.ecr	place/city1/bosba	Settlement of Bosnia and Herzegovina with over 100,000 inhabitants.
	place/city1_uppercase/bosba	Settlement of Bosnia and Herzegovina with over 100,000 inhabitants, in uppercase.
	place/city2/bosba	Settlement of Bosnia and Herzegovina with between

File	Entity	Description
	place/city2_uppercase/bosba	10,000 and 100,000 inhabitants. Settlement of Bosnia and Herzegovina with between 10,000 and 100,000 inhabitants, in uppercase.
place_chicn.ecr	place/city/chicn place/province/chicn	Chinese city. Chinese province.
place_chihk.ecr	place/district/chihk place/island/chihk place/port/chihk place/hospital/chihk place/tunnel/chihk place/bridge/chihk place/hotel/chihk place/locality/chihk	District in Hong Kong. Island in Hong Kong. Port in Hong Kong. Hospital in Hong Kong. Tunnel in Hong Kong. Bridge in Hong Kong. Hotel in Hong Kong. Place in Hong Kong.
place_countries.ecr	country/iso_lowercase country/all country/output_iso	ISO 3166-1 alpha-2 country code. Country in a local or major language. Country in a local or major language (output is normalized to the ISO 3166-1 alpha-2 code).
place_czecz.ecr	place/city1/czecz place/city1_uppercase/czecz place/city2/czecz place/city2_uppercase/czecz place/region/czecz place/region_uppercase/czecz	Czech settlement with over 100,000 inhabitants. Czech settlement with over 100,000 inhabitants, in uppercase. Czech settlement with between 10,000 and 100,000 inhabitants. Czech settlement with between 10,000 and 100,000 inhabitants, in uppercase.

File	Entity	Description
		Czech region. Czech region in uppercase.
place_dandk.ecr	place/city1/dandk place/city1_uppercase/dandk place/city2/dandk place/city2_uppercase/dandk place/region/dandk place/region_uppercase/dandk place/municipality/dandk place/municipality_uppercase/dandk place/island/dandk place/island_uppercase/dandk	Danish settlement with over 100,000 inhabitants. Danish settlement with over 100,000 inhabitants, in uppercase. Danish settlement with between 10,000 and 100,000 inhabitants. Danish settlement with between 10,000 and 100,000 inhabitants, in uppercase. Danish region. Danish region in uppercase. Danish municipality. Danish municipality in uppercase. Danish island Danish island in uppercase
place_dutnl.ecr	place/city1/dutnl place/city1_uppercase/dutnl place/city2/dutnl place/city2_uppercase/dutnl place/county/dutnl place/county_uppercase/dutnl place/municipality/dutnl place/municipality_uppercase/dutnl	Dutch settlement with over 100,000 inhabitants. Dutch settlement with over 100,000 inhabitants, in uppercase. Dutch settlement with between 10,000 and 100,000 inhabitants. Dutch settlement with between 10,000 and 100,000 inhabitants, in uppercase. Dutch county. Dutch county in uppercase.

File	Entity	Description
	place/island/dutnl place/island_uppercase/dutnl	Dutch municipality. Dutch municipality in uppercase. Dutch island. Dutch island in uppercase.
place_dutsr.ecr	place/city1/dutsr place/city1_uppercase/dutsr place/city2/dutsr place/city2_uppercase/dutsr place/district/dutsr place/district_uppercase/dutsr	Surinamese settlement with over 100,000 inhabitants. Surinamese settlement with over 100,000 inhabitants, in uppercase. Surinamese settlement with under 100,000 inhabitants. Surinamese settlement with under 100,000 inhabitants, in uppercase. Surinamese district. Surinamese district in uppercase.
place_engae.ecr	place/city1/engae place/city1_uppercase/engae place/city2/engae place/city2_uppercase/engae place/emirate/engae place/emirate_uppercase/engae	Settlement of the United Arab Emirates with over 100,000 inhabitants. Settlement of the United Arab Emirates with over 100,000 inhabitants, in uppercase. Settlement of the United Arab Emirates with under 100,000 inhabitants. Settlement of the United Arab Emirates with under 100,000 inhabitants, in uppercase. Emirate of the United Arab Emirates. Emirate of the United Arab Emirates in uppercase.
place_engau.ecr	place/state/engau	Australian state or territory.

File	Entity	Description
	place/state_uppercase/engau place/state_abbrev/engau place/state_capital/engau place/state_capital_uppercase/engau place/city1/engau place/city1_uppercase/engau place/city2/engau place/city2_uppercase/engau place/city/NSW/engau place/city_uppercase/NSW/engau place/city/QLD/engau place/city_uppercase/QLD/engau place/city/SA/engau place/city_uppercase/SA/engau place/city/TAS/engau place/city_uppercase/TAS/engau place/city/VIC/engau place/city_uppercase/VIC/engau place/city/WA/engau place/city_uppercase/WA/engau place/city/NT/engau place/city_uppercase/NT/engau place/city/ACT/engau	Australian state or territory in uppercase. Australian state or territory abbreviations. Australian state or territory capitals. Australian state or territory capitals in uppercase. Australian city with population greater than 100,000. Australian city with population greater than 100,000, in uppercase. Australian city with population between 10,000 and 100,000. Australian city with population between 10,000 and 100,000, in uppercase. Settlement in New South Wales. Settlement in New South Wales, in uppercase. Settlement in Queensland, Australia. Settlement in Queensland, Australia, in uppercase. Settlement in South Australia. Settlement in South Australia, in uppercase. Settlement in Tasmania. Settlement in Tasmania, in uppercase. Settlement in Victoria, Australia. Settlement in Victoria, Australia, in uppercase. Settlement in Western Australia. Settlement in Western Australia, in uppercase. Settlement in Northern Territory, Australia.

File	Entity	Description
	place/city_uppercase/ACT/engau place/city/engau place/city_uppercase/engau	Settlement in Northern Territory, Australia, in uppercase. Settlement in Australian Capital Territory. Settlement in Australian Capital Territory, in uppercase. Australian cities. Australian cities in uppercase.
place_engbd.ecr	place/city1/engbd place/city1_uppercase/engbd place/city2/engbd place/city2_uppercase/engbd place/division/engbd place/division_uppercase/engbd place/district/engbd place/district_uppercase/engbd	Bangladeshi settlement with over 100,000 inhabitants. Bangladeshi settlement with over 100,000 inhabitants, in uppercase. Bangladeshi settlement with between 10,000 and 100,000 inhabitants. Bangladeshi settlement with between 10,000 and 100,000 inhabitants, in uppercase. Bangladeshi division. Bangladeshi division in uppercase. Bangladeshi district. Bangladeshi district in uppercase.
place_engbg.ecr	place/city1/engbg place/city1_uppercase/engbg place/city2/engbg place/city2_uppercase/engbg place/province/engbg	Bulgarian settlement with over 100,000 inhabitants. Bulgarian settlement with over 100,000 inhabitants, in uppercase. Bulgarian settlement with between 10,000 and 100,000 inhabitants. Bulgarian settlement with between 10,000 and

File	Entity	Description
	place/province_uppercase/engbg	100,000 inhabitants, in uppercase. Bulgarian province. Bulgarian province in uppercase.
place_engby.ecr	place/city1/engby place/city1_uppercase/engby place/city2/engby place/city2_uppercase/engby place/region/engby place/region_uppercase/engby	Belarusian settlement with over 100,000 inhabitants. Belarusian settlement with over 100,000 inhabitants, in uppercase. Belarusian settlement with between 10,000 and 100,000 inhabitants. Belarusian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Belarusian region. Belarusian region in uppercase.
place_engca.ecr	place/region/engca place/region_uppercase/engca place/region_abbrev/engca place/region_all/engca place/region_all_uppercase/engca place/region_capitals/engca place/region_capitals_uppercase/engca place/city1/engca place/city1_uppercase/engca place/city2/engca place/city2_uppercase/engca	Canadian province or territory. Canadian province or territory in uppercase. Canadian province or territory abbreviation. Canadian province or territory full name or abbreviation. Canadian province or territory full name or abbreviation, in uppercase. Canadian provincial or territorial capital. Canadian provincial or territorial capital in uppercase. Canadian settlement with over 100,000 inhabitants. Canadian settlement with over 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/city/AB/engca place/city_uppercase/AB/engca place/city/BC/engca place/city_uppercase/BC/engca place/city/MB/engca place/city_uppercase/MB/engca place/city/NB/engca place/city_uppercase/NB/engca place/city/NL/engca place/city_uppercase/NL/engca place/city/NS/engca place/city_uppercase/NS/engca place/city/ON/engca place/city_uppercase/ON/engca place/city/PE/engca place/city_uppercase/PE/engca place/city/QC/engca place/city_uppercase/QC/engca place/city/SK/engca place/city_uppercase/SK/engca place/city/NT/engca place/city_uppercase/NT/engca place/city/NU/engca	Canadian settlement with between 10,000 and 100,000 inhabitants. Canadian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Settlements in each Canadian province or territory, in normal or uppercase.

File	Entity	Description
	place/city_uppercase/NU/engca place/city/YT/engca place/city_uppercase/YT/engca place/city/engca place/city_uppercase/engca	Canadian settlement. Canadian settlement in uppercase.
place_engcn.ecr	place/city0/engcn place/city0_uppercase/engcn place/city1/engcn place/city1_uppercase/engcn place/city2/engcn place/city2_uppercase/engcn place/province/engcn place/province_uppercase/engcn	Chinese settlement with over 1,000,000 inhabitants. Chinese settlement with over 1,000,000 inhabitants, in uppercase. Chinese settlement with over 100,000 inhabitants. Chinese settlement with over 100,000 inhabitants, in uppercase. Chinese settlement with over 10,000 inhabitants. Chinese settlement with over 10,000 inhabitants, in uppercase. Chinese province. Chinese province in uppercase.
place_enggb.ecr	place/possession/enggb place/possession_uppercase/enggb place/country/enggb	UK crown dependencies. UK crown dependencies in uppercase. UK countries.

File	Entity	Description
	place/country_uppercase/enggb	UK countries in uppercase.
	place/country_capital/enggb	UK country capitals.
	place/country_capital_uppercase/enggb	UK country capitals in uppercase.
	place/county/england/enggb	Counties in England.
	place/county_uppercase/england/enggb	Counties in England, in uppercase.
	place/county/northern_ireland/enggb	Counties in Northern Ireland.
	place/county_uppercase/northern_ireland/enggb	Counties in Northern Ireland, in uppercase.
	place/county/scotland/enggb	Counties in Scotland.
	place/county_uppercase/scotland/enggb	Counties in Scotland, in uppercase.
	place/county/wales/enggb	Counties in Wales.
	place/county_uppercase/wales/enggb	Counties in Wales, in uppercase.
	place/county/enggb	Counties in UK.
	place/county_uppercase/enggb	Counties in UK in uppercase.
	place/city1/enggb	Settlement in the UK with over 100,000 inhabitants.
	place/city1_uppercase/enggb	Settlement in the UK with over 100,000 inhabitants, in uppercase.
	place/city/england/enggb	Settlements in each UK country, in normal or uppercase.
	place/city_uppercase/england/enggb	
	place/city/scotland/enggb	
	place/city_uppercase/scotland/enggb	
	place/city/wales/enggb	
	place/city_uppercase/wales/enggb	
	place/city/northern_ireland/enggb	
	place/city_uppercase/northern_ireland/enggb	

File	Entity	Description
	place/city/enggb place/city_uppercase/enggb place/londonborough/enggb place/island/enggb place/island_uppercase/enggb	UK settlements. UK settlements in uppercase. London borough. Major islands of the United Kingdom. Major islands of the United Kingdom in uppercase.
place_enggr.ecr	place/city1/enggr place/city1_uppercase/enggr place/city2/enggr place/city2_uppercase/enggr place/region/enggr place/region_uppercase/enggr place/prefecture/enggr place/prefecture_uppercase/enggr place/municipality/enggr place/municipality_uppercase/enggr place/island/enggr place/island_uppercase/enggr	Greek settlement with over 100,000 inhabitants. Greek settlement with over 100,000 inhabitants, in uppercase. Greek settlement with between 10,000 and 100,000 inhabitants. Greek settlement with between 10,000 and 100,000 inhabitants, in uppercase. Greek region. Greek region in uppercase. Greek prefecture (obsolete after 2010). Greek prefecture in uppercase (obsolete after 2010). Greek municipality. Greek municipality in uppercase. Greek island. Greek island in uppercase.
place_enggy.ecr	place/city1/enggy	Guyanan settlement with over 100,000 inhabitants.

File	Entity	Description
	place/city1_uppercase/enggy place/city2/enggy place/city2_uppercase/enggy place/region/enggy place/region_uppercase/enggy	Guyanan settlement with over 100,000 inhabitants, in uppercase. Guyanan settlement with between 10,000 and 100,000 inhabitants. Guyanan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Guyanan region. Guyanan region in uppercase.
place_enghk.ecr	place/district/enghk place/island/enghk place/enghk	District in Hong Kong. Island in Hong Kong. Street in Hong Kong.
place_engid.ecr	place/city1/engid place/city1_uppercase/engid place/city2/engid place/city2_uppercase/engid place/province/engid place/province_uppercase/engid place/regency/engid place/regency_uppercase/engid place/island/engid place/island_uppercase/engid	Indonesian settlement with over 100,000 inhabitants. Indonesian settlement with over 100,000 inhabitants, in uppercase. Indonesian settlement with between 10,000 and 100,000 inhabitants. Indonesian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Indonesian province. Indonesian province in uppercase. Indonesian regency. Indonesian regency in uppercase. Indonesian island. Indonesian island in uppercase.

File	Entity	Description
place_engie.ecr	place/city1/engie place/city1_uppercase/engie place/city2/engie place/city2_uppercase/engie place/county_engie place/county_uppercase/engie place/island/engie place/island_uppercase/engie	Irish settlement with over 100,000 inhabitants. Irish settlement with over 100,000 inhabitants, in uppercase. Irish settlement with between 10,000 and 100,000 inhabitants. Irish settlement with between 10,000 and 100,000 inhabitants, in uppercase. Irish county. Irish county in uppercase. Irish island. Irish island in uppercase.
place_engin.ecr	place/city1/engin place/city1_uppercase/engin place/city2/engin place/city2_uppercase/engin place/state/engin place/state_uppercase/engin place/union_territory/engin place/union_territory_uppercase/engin place/district/engin place/district_uppercase/engin place/island/engin place/island_uppercase/engin	Indian settlement with over 100,000 inhabitants. Indian settlement with over 100,000 inhabitants, in uppercase. Indian settlement with between 10,000 and 100,000 inhabitants. Indian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Indian state. Indian state in uppercase. Indian union territory. Indian union territory in uppercase. Indian district. Indian district in uppercase.

File	Entity	Description
		Indian island. Indian island in uppercase.
place_engir.ecr	place/city1/engir place/city1_uppercase/engir place/city2/engir place/city2_uppercase/engir place/province/engir place/province_uppercase/engir place/county/engir place/county_uppercase/engir	Iranian settlement with over 100,000 inhabitants. Iranian settlement with over 100,000 inhabitants, in uppercase. Iranian settlement with between 10,000 and 100,000 inhabitants. Iranian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Iranian province. Iranian province in uppercase. Iranian county. Iranian county in uppercase.
place_engjp.ecr	place_city1/engjp place_city1_uppercase/engjp place/city2/engjp place/city2_uppercase/engjp place/special_ward/engjp place/special_ward_uppercase/engjp place/island/engjp place/island_uppercase/engjp place/prefecture/engjp place/prefecture_uppercase/engjp	Japanese settlement with over 100,000 inhabitants, in English. Japanese settlement with over 100,000 inhabitants, in uppercase English. Japanese settlement with between 10,000 and 100,000 inhabitants, in English. Japanese settlement with between 10,000 and 100,000 inhabitants, in uppercase English. Special ward of Tokyo in English. Special ward of Tokyo in uppercase English. Japanese island in English. Japanese island in uppercase English.

File	Entity	Description
	place/region/engjp place/region_uppercase/engjp place/city/aichi/engjp place/city_uppercase/aichi/engjp place/city/akita/engjp place/city_uppercase/akita/engjp place/city/aomori/engjp place/city_uppercase/aomori/engjp place/city/chiba/engjp place/city_uppercase/chiba/engjp place/city/ehime/engjp place/city_uppercase/ehime/engjp place/city/fukui/engjp place/city_uppercase/fukui/engjp place/city/fukuoka/engjp place/city_uppercase/fukuoka/engjp place/city/fukushima/engjp place/city_uppercase/fukushima/engjp place/city/gifu/engjp place/city_uppercase/gifu/engjp place/city/gunma/engjp place/city_uppercase/gunma/engjp place/city/hiroshima/engjp	Japanese prefectures in English. Japanese prefectures in uppercase English. Japanese regions in English. Japanese regions in uppercase English. Cities in each Japanese prefecture in English, in normal or uppercase.

File	Entity	Description
	place/city_uppercase/hiroshima/engjp	
	place/city/hokkaido/engjp	
	place/city_uppercase/hokkaido/engjp	
	place/city/hyogo/engjp	
	place/city_uppercase/hyogo/engjp	
	place/city/ibaraki/engjp	
	place/city_uppercase/ibaraki/engjp	
	place/city/ishikawa/engjp	
	place/city_uppercase/ishikawa/engjp	
	place/city/iwate/engjp	
	place/city_uppercase/iwate/engjp	
	place/city/kagawa/engjp	
	place/city_uppercase/kagawa/engjp	
	place/city/kagoshima/engjp	
	place/city_uppercase/kagoshima/engjp	
	place/city/kanagawa/engjp	
	place/city_uppercase/kanagawa/engjp	
	place/city/kochi/engjp	
	place/city_uppercase/kochi/engjp	
	place/city/kumamoto/engjp	
	place/city_uppercase/kumamoto/engjp	
	place/city/kyoto/engjp	
	place/city_uppercase/kyoto/engjp	

File	Entity	Description
	place/city/mie/engjp place/city_uppercase/mie/engjp place/city/miyagi/engjp place/city_uppercase/miyagi/engjp place/city/miyazaki/engjp place/city_uppercase/miyazaki/engjp place/city/nagano/engjp place/city_uppercase/nagano/engjp place/city/nagasaki/engjp place/city_uppercase/nagasaki/engjp place/city/nara/engjp place/city_uppercase/nara/engjp place/city/niigata/engjp place/city_uppercase/niigata/engjp place/city/oita/engjp place/city_uppercase/oita/engjp place/city/okayama/engjp place/city_uppercase/okayama/engjp place/city/okinawa/engjp place/city_uppercase/okinawa/engjp place/city/osaka/engjp place/city_uppercase/osaka/engjp place/city/saga/engjp	

File	Entity	Description
	place/city_uppercase/saga/engjp place/city/saitama/engjp place/city_uppercase/saitama/engjp place/city/shiga/engjp place/city_uppercase/shiga/engjp place/city/shimane/engjp place/city_uppercase/shimane/engjp place/city/shizuoka/engjp place/city_uppercase/shizuoka/engjp place/city/tochigi/engjp place/city_uppercase/tochigi/engjp place/city/tokushima/engjp place/city_uppercase/tokushima/engjp place/city/tokyo/engjp place/city_uppercase/tokyo/engjp place/city/tottori/engjp place/city_uppercase/tottori/engjp place/city/toyama/engjp place/city_uppercase/toyama/engjp place/city/wakayama/engjp place/city_uppercase/wakayama/engjp place/city/yamagata/engjp place/city_uppercase/yamagata/engjp	

File	Entity	Description
	place/city/yamaguchi/engjp place/city_uppercase/yamaguchi/engjp place/city/yamanashi/engjp place/city_uppercase/yamanashi/engjp place/city/engjp	Japanese cities in English.
place_engkr.ecr	place/city1_rr/engkr	South Korean settlement with over 100,000 inhabitants, in revised romanization.
	place/city1_rr_uppercase/engkr	South Korean settlement with over 100,000 inhabitants, in uppercase revised romanization.
	place/city2_rr/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in revised romanization.
	place/city2_rr_uppercase/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in uppercase revised romanization.
	place/province_rr/engkr	South Korean province in revised romanization.
	place/province_rr_uppercase/engkr	South Korean province in uppercase revised romanization.
	place/county_rr/engkr	South Korean county in revised romanization.
	place/county_rr_uppercase/engkr	South Korean county in uppercase revised romanization.
	place/island_rr/engkr	South Korean island in revised romanization.
	place/island_rr_uppercase/engkr	South Korean island in uppercase revised
	place/city1_mcr/engkr	

File	Entity	Description
	place/city1_mcr_uppercase/engkr	romanization.
	place/city2_mcr/engkr	South Korean settlement with over 100,000 inhabitants in McCune-Reischauer romanization.
	place/city2_mcr_uppercase/engkr	South Korean settlement with over 100,000 inhabitants, in uppercase McCune-Reischauer romanization.
	place/province_mcr/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in McCune-Reischauer romanization.
	place/province_mcr_uppercase/engkr	South Korean settlement with between 10,000 and 100,000 inhabitants, in uppercase McCune-Reischauer romanization.
	place/county_mcr/engkr	South Korean province in McCune-Reischauer romanization.
	place/county_mcr_uppercase/engkr	South Korean province in uppercase McCune-Reischauer romanization.
	place/island_mcr/engkr	South Korean county in McCune-Reischauer romanization.
	place/island_mcr_uppercase/engkr	South Korean county in uppercase McCune-Reischauer romanization.
		South Korean island in McCune-Reischauer romanization.
		South Korean island in uppercase McCune-Reischauer romanization.
place_englk.ecr	place/city1/englk	Sri Lankan settlement with over 100,000 inhabitants.
	place/city1_uppercase/englk	Sri Lankan settlement with over 100,000 inhabitants, in uppercase.
	place/city2/englk	

File	Entity	Description
	place/city2_uppercase/englk place/province/englk place/province_uppercase/englk place/district/englk place/district_uppercase/englk	Sri Lankan settlement with between 10,000 and 100,000 inhabitants. Sri Lankan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Sri Lankan province. Sri Lankan province in uppercase. Sri Lankan district. Sri Lankan district in uppercase.
place_engmk.ecr	place/city1/engmk place/city1_uppercase/engmk place/city2/engmk place/city2_uppercase/engmk place/municipality/engmk place/municipality_uppercase/engmk	Macedonian settlement with over 100,000 inhabitants. Macedonian settlement with over 100,000 inhabitants, in uppercase. Macedonian settlement with between 10,000 and 100,000 inhabitants. Macedonian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Macedonian municipality. Macedonian municipality in uppercase.
place_engmn.ecr	place/city1/engmn place/city1_uppercase/engmn place/city2/engmn place/city2_uppercase/engmn place/province/engmn place/province_uppercase/engmn	Mongolian settlement with over 100,000 inhabitants. Mongolian settlement with over 100,000 inhabitants, in uppercase. Mongolian settlement with between 10,000 and 100,000 inhabitants. Mongolian settlement with between 10,000 and 100,000 inhabitants, in uppercase.

File	Entity	Description
		Mongolian province. Mongolian province in uppercase.
place_engmy.ecr	place/city1/engmy place/city1_uppercase/engmy place/city2/engmy place/city2_uppercase/engmy place/state/engmy place/state_uppercase/engmy place/district/engmy place/district_uppercase/engmy	Malaysian settlement with over 100,000 inhabitants. Malaysian settlement with over 100,000 inhabitants, in uppercase. Malaysian settlement with between 10,000 and 100,000 inhabitants. Malaysian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Malaysian state. Malaysian state in uppercase. Malaysian district. Malaysian district in uppercase.
place_engnz.ecr	place/city1/engnz place/city1_uppercase/engnz place/city2/engnz place/city2_uppercase/engnz place/region/engnz place/region_uppercase/engnz place/terr_auth/engnz place/terr_auth_uppercase/engnz place/island/engnz place/island_uppercase/engnz	New Zealand settlement with over 100,000 inhabitants. New Zealand settlement with over 100,000 inhabitants, in uppercase. New Zealand settlement with between 10,000 and 100,000 inhabitants. New Zealand settlement with between 10,000 and 100,000 inhabitants, in uppercase. New Zealand region. New Zealand region in uppercase. New Zealand territorial authority. New Zealand territorial authority in uppercase.

File	Entity	Description
		New Zealand island. New Zealand island in uppercase.
place_engph.ecr	place/city1/engph place/city1_uppercase/engph place/city2/engph place/city2_uppercase/engph place/region/engph place/region_uppercase/engph place/province/engph place/province_uppercase/engph place/island/engph place/island_uppercase/engph	Philippine settlement with over 100,000 inhabitants. Philippine settlement with over 100,000 inhabitants, in uppercase. Philippine settlement with between 10,000 and 100,000 inhabitants. Philippine settlement with between 10,000 and 100,000 inhabitants, in uppercase. Philippine region. Philippine region in uppercase. Philippine province. Philippine province in uppercase. Philippine island. Philippine island in uppercase.
place_engpk.ecr	place/city1/engpk place/city1_uppercase/engpk place/city2/engpk place/city2_uppercase/engpk place/province/engpk place/province_uppercase/engpk place/district/engpk place/district_uppercase/engpk	Pakistani settlement with over 100,000 inhabitants. Pakistani settlement with over 100,000 inhabitants, in uppercase. Pakistani settlement with between 10,000 and 100,000 inhabitants. Pakistani settlement with between 10,000 and 100,000 inhabitants, in uppercase. Pakistani province. Pakistani province in uppercase.

File	Entity	Description
		Pakistani district. Pakistani district in uppercase.
place_engqa.ecr	place/city1/engqa place/city1_uppercase/engqa place/city2/engqa place/city2_uppercase/engqa place/municipality/engqa place/municipality_uppercase/engqa	Qatari settlement with over 100,000 inhabitants. Qatari settlement with over 100,000 inhabitants, in uppercase. Qatari settlement with fewer than 100,000 inhabitants. Qatari settlement with fewer than 100,000 inhabitants, in uppercase. Qatari municipality. Qatari municipality in uppercase.
place_engru.ecr	place/city1/engru place/city1_uppercase/engru place/city2/engru place/city2_uppercase/engru place/republic/engru place/republic_uppercase/engru place/oblast/engru place/oblast_uppercase/engru place/krai/engru place/krai_uppercase/engru place/okrug/engru place/okrug_uppercase/engru	Russian settlement with over 100,000 inhabitants. Russian settlement with over 100,000 inhabitants, in uppercase. Russian settlement with between 10,000 and 100,000 inhabitants. Russian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Russian republic (type of region). Russian republic (type of region), in uppercase. Russian oblast (type of region). Russian oblast (type of region), in uppercase. Russian krai (type of region). Russian krai (type of region), in uppercase.

File	Entity	Description
	place/federal_city/engru place/federal_city_uppercase/engru place/region/engru place/region_uppercase/engru place/island/engru place/island_uppercase/engru	Russian okrug (type of region). Russian okrug (type of region), in uppercase. Russian federal city (type of region). Russian federal city (type of region), in uppercase. Russian region. Russian region, in uppercase. Russian island. Russian island, in uppercase.
place_engsa.ecr	place/city1/engsa place/city1_uppercase/engsa place/city2/engsa place/city2_uppercase/engsa place/province/engsa place/province_uppercase/engsa	Saudi Arabian settlement with over 100,000 inhabitants. Saudi Arabian settlement with over 100,000 inhabitants, in uppercase. Saudi Arabian settlement with fewer than 100,000 inhabitants. Saudi Arabian settlement with fewer than 100,000 inhabitants, in uppercase. Saudi Arabian province. Saudi Arabian province in uppercase.
place_ength.ecr	place/city1/ength place/city1_uppercase/ength place/city2/ength place/city2_uppercase/ength place/province/ength place/province_uppercase/ength	Thai settlement with over 100,000 inhabitants. Thai settlement with over 100,000 inhabitants, in uppercase. Thai settlement with between 10,000 and 100,000 inhabitants. Thai settlement with between 10,000 and 100,000 inhabitants, in uppercase.

File	Entity	Description
		Thai province. Thai province in uppercase.
place_engtw.ecr	place/city1/engtw place/city1_uppercase/engtw place/city2/engtw place/city2_uppercase/engtw place/county/engtw place/county_uppercase/engtw	Taiwanese settlement with over 100,000 inhabitants. Taiwanese settlement with over 100,000 inhabitants, in uppercase. Taiwanese settlement with between 10,000 and 100,000 inhabitants. Taiwanese settlement with between 10,000 and 100,000 inhabitants, in uppercase. Taiwanese county. Taiwanese county in uppercase.
place_engua.ecr	place/city1/engua place/city1_uppercase/engua place/city2/engua place/city2_uppercase/engua place/region/engua place/region_uppercase/engua	Ukrainian settlement with over 100,000 inhabitants. Ukrainian settlement with over 100,000 inhabitants, in uppercase. Ukrainian settlement with between 10,000 and 100,000 inhabitants. Ukrainian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Ukrainian region. Ukrainian region in uppercase.
place_engus.ecr	place/possession/engus place/possession_uppercase/engus place/possession_abbrev/engus place/state/engus	U.S. possessions in long form. For example, *American Samoa*. U.S. possessions in long form, in uppercase. U.S. possession abbreviations. For example, *GU*.

File	Entity	Description
	place/state_uppercase/engus place/state_abbrev/engus place/poss_state/engus place/poss_state_abbrev/engus place/statecapital/engus place/statecapital_uppercase/engus	U.S. states. For example, *New Hampshire*. U.S. states, in uppercase. U.S. states abbreviations. For example, *AL*. U.S. possessions and states. U.S. possession and state abbreviations. U.S. state capitals. U.S. state capitals, in uppercase.
	place/city/AL/engus place/city_uppercase/AL/engus place/city/AK/engus place/city_uppercase/AK/engus place/city/AZ/engus place/city_uppercase/AZ/engus place/city/AR/engus place/city_uppercase/AR/engus place/city/CA/engus place/city_uppercase/CA/engus place/city/CO/engus place/city_uppercase/CO/engus place/city/CT/engus place/city_uppercase/CT/engus place/city/DE/engus	Settlements in each U.S. state, in normal or uppercase.

File	Entity	Description
	place/city_uppercase/DE/engus place/city/FL/engus place/city_uppercase/FL/engus place/city/GA/engus place/city_uppercase/GA/engus place/city/HI/engus place/city_uppercase/HI/engus place/city/ID/engus place/city_uppercase/ID/engus place/city/IL/engus place/city_uppercase/IL/engus place/city/IN/engus place/city_uppercase/IN/engus place/city/IA/engus place/city_uppercase/IA/engus place/city/KS/engus place/city_uppercase/KS/engus place/city/KY/engus place/city_uppercase/KY/engus place/city/LA/engus place/city_uppercase/LA/engus place/city/ME/engus place/city_uppercase/ME/engus	

File	Entity	Description
	place/city/MD/engus place/city_uppercase/MD/engus place/city/MA/engus place/city_uppercase/MA/engus place/city/MI/engus place/city_uppercase/MI/engus place/city/MN/engus place/city_uppercase/MN/engus place/city/MS/engus place/city_uppercase/MS/engus place/city/MO/engus place/city_uppercase/MO/engus place/city/MT/engus place/city_uppercase/MT/engus place/city/NE/engus place/city_uppercase/NE/engus place/city/NV/engus place/city_uppercase/NV/engus place/city/NH/engus place/city_uppercase/NH/engus place/city/NJ/engus place/city_uppercase/NJ/engus place/city/NM/engus	

File	Entity	Description
	place/city_uppercase/NM/engus place/city/NY/engus place/city_uppercase/NY/engus place/city/NC/engus place/city_uppercase/NC/engus place/city/ND/engus place/city_uppercase/ND/engus place/city/OH/engus place/city_uppercase/OH/engus place/city/OK/engus place/city_uppercase/OK/engus place/city/OR/engus place/city_uppercase/OR/engus place/city/PA/engus place/city_uppercase/PA/engus place/city/RI/engus place/city_uppercase/RI/engus place/city/SC/engus place/city_uppercase/SC/engus place/city/SD/engus place/city_uppercase/SD/engus place/city/TN/engus place/city_uppercase/TN/engus	

File	Entity	Description
	place/city/TX/engus place/city_uppercase/TX/engus place/city/UT/engus place/city_uppercase/UT/engus place/city/VA/engus place/city_uppercase/VA/engus place/city/VT/engus place/city_uppercase/VT/engus place/city/WA/engus place/city_uppercase/WA/engus place/city/WI/engus place/city_uppercase/WI/engus place/city/WV/engus place/city_uppercase/WV/engus place/city/WY/engus place/city_uppercase/WY/engus	
	place/city1/engus	U.S. city with over 100,000 inhabitants.
	place/city1_uppercase/engus	U.S. city with over 100,000 inhabitants, in uppercase.
	place/city2/engus	U.S. city with over 10,000 inhabitants.
	place/city2_uppercase/engus	U.S. city with over 10,000 inhabitants, in uppercase.
	place/county/AL/engus	County in Alabama.
	place/county/AK/engus	County in Alaska.

File	Entity	Description
	place/county/AZ/engus	County in Arizona.
	place/county/AR/engus	County in Arkansas.
	place/county/CA/engus	County in California.
	place/county/CO/engus	County in Colorado.
	place/county/CT/engus	County in Connecticut.
	place/county/DE/engus	County in Delaware.
	place/county/FL/engus	County in Florida.
	place/county/GA/engus	County in Georgia.
	place/county/HI/engus	County in Hawaii.
	place/county/ID/engus	County in Idaho.
	place/county/IL/engus	County in Illinois.
	place/county/IN/engus	County in Indiana.
	place/county/IA/engus	County in Iowa.
	place/county/KS/engus	County in Kansas.
	place/county/KY/engus	County in Kentucky.
	place/county/LA/engus	County in Louisiana.
	place/county/ME/engus	County in Maine.
	place/county/MD/engus	County in Maryland.
	place/county/MA/engus	County in Massachusetts.
	place/county/MI/engus	County in Michigan.
	place/county/MN/engus	County in Minnesota.
	place/county/MS/engus	County in Mississippi.
	place/county/MO/engus	County in Missouri.

File	Entity	Description
	place/county/MT/engus	County in Montana.
	place/county/NE/engus	County in Nebraska.
	place/county/NV/engus	County in Nevada.
	place/county/NH/engus	County in New Hampshire.
	place/county/NJ/engus	County in New Jersey.
	place/county/NM/engus	County in New Mexico.
	place/county/NY/engus	County in New York.
	place/county/NC/engus	County in North Carolina.
	place/county/ND/engus	County in North Dakota.
	place/county/OH/engus	County in Ohio.
	place/county/OK/engus	County in Oklahoma.
	place/county/OR/engus	County in Oregon.
	place/county/PA/engus	County in Pennsylvania.
	place/county/RI/engus	County in Rhode Island.
	place/county/SC/engus	County in South Carolina.
	place/county/SD/engus	County in South Dakota.
	place/county/TN/engus	County in Tennessee.
	place/county/TX/engus	County in Texas.
	place/county/UT/engus	County in Utah.
	place/county/VT/engus	County in Vermont.
	place/county/VA/engus	County in Virginia.
	place/county/WA/engus	County in Washington.
	place/county/WV/engus	County in West Virginia.

File	Entity	Description
	place/county/WI/engus	County in Wisconsin.
	place/county/WY/engus	County in Wyoming.
	place/county_uppercase/AL/engus	County in Alabama in uppercase.
	place/county_uppercase/AK/engus	County in Alaska in uppercase.
	place/county_uppercase/AZ/engus	County in Arizona in uppercase.
	place/county_uppercase/AR/engus	County in Arkansas in uppercase.
	place/county_uppercase/CA/engus	County in California in uppercase.
	place/county_uppercase/CO/engus	County in Colorado in uppercase.
	place/county_uppercase/CT/engus	County in Connecticut in uppercase.
	place/county_uppercase/DE/engus	County in Delaware in uppercase.
	place/county_uppercase/FL/engus	County in Florida in uppercase.
	place/county_uppercase/GA/engus	County in Georgia in uppercase.
	place/county_uppercase/HI/engus	County in Hawaii in uppercase.
	place/county_uppercase/ID/engus	County in Idaho in uppercase.
	place/county_uppercase/IL/engus	County in Illinois in uppercase.
	place/county_uppercase/IN/engus	County in Indiana in uppercase.
	place/county_uppercase/IA/engus	County in Iowa in uppercase.
	place/county_uppercase/KS/engus	County in Kansas in uppercase.
	place/county_uppercase/KY/engus	County in Kentucky in uppercase.
	place/county_uppercase/LA/engus	County in Louisiana in uppercase.
	place/county_uppercase/ME/engus	County in Maine in uppercase.
	place/county_uppercase/MD/engus	County in Maryland in uppercase.
	place/county_uppercase/MA/engus	County in Massachusetts in uppercase.

File	Entity	Description
	place/county_uppercase/MI/engus	County in Michigan in uppercase.
	place/county_uppercase/MN/engus	County in Minnesota in uppercase.
	place/county_uppercase/MS/engus	County in Mississippi in uppercase.
	place/county_uppercase/MO/engus	County in Missouri in uppercase.
	place/county_uppercase/MT/engus	County in Montana in uppercase.
	place/county_uppercase/NE/engus	County in Nebraska in uppercase.
	place/county_uppercase/NV/engus	County in Nevada in uppercase.
	place/county_uppercase/NH/engus	County in New Hampshire in uppercase.
	place/county_uppercase/NJ/engus	County in New Jersey in uppercase.
	place/county_uppercase/NM/engus	County in New Mexico in uppercase.
	place/county_uppercase/NY/engus	County in New York in uppercase.
	place/county_uppercase/NC/engus	County in North Carolina in uppercase.
	place/county_uppercase/ND/engus	County in North Dakota in uppercase.
	place/county_uppercase/OH/engus	County in Ohio in uppercase.
	place/county_uppercase/OK/engus	County in Oklahoma in uppercase.
	place/county_uppercase/OR/engus	County in Oregon in uppercase.
	place/county_uppercase/PA/engus	County in Pennsylvania in uppercase.
	place/county_uppercase/RI/engus	County in Rhode Island in uppercase.
	place/county_uppercase/SC/engus	County in South Carolina in uppercase.
	place/county_uppercase/SD/engus	County in South Dakota in uppercase.
	place/county_uppercase/TN/engus	County in Tennessee in uppercase.
	place/county_uppercase/TX/engus	County in Texas in uppercase.
	place/county_uppercase/UT/engus	County in Utah in uppercase.

File	Entity	Description
	place/county_uppercase/VT/engus	County in Vermont in uppercase.
	place/county_uppercase/VA/engus	County in Virginia in uppercase.
	place/county_uppercase/WA/engus	County in Washington in uppercase.
	place/county_uppercase/WV/engus	County in West Virginia in uppercase.
	place/county_uppercase/WI/engus	County in Wisconsin in uppercase.
	place/county_uppercase/WY/engus	County in Wyoming in uppercase.
	place/county/engus	Any U.S. county.
	place/county_uppercase/engus	Any U.S. county in uppercase.
place_engvn.ecr	place/city1/engvn	Vietnamese settlement with over 100,000 inhabitants.
	place/city1_uppercase/engvn	Vietnamese settlement with over 100,000 inhabitants, in uppercase.
	place/city2/engvn	Vietnamese settlement with between 10,000 and 100,000 inhabitants.
	place/city2_uppercase/engvn	Vietnamese settlement with between 10,000 and 100,000 inhabitants, in uppercase.
	place/province/engvn	Vietnamese province.
	place/province_uppercase/engvn	Vietnamese province in uppercase.
	place/district/engvn	Vietnamese district.
	place/district_uppercase/engvn	Vietnamese district in uppercase.
place_engza.ecr	place/city1/engza	South African settlement with over 100,000 inhabitants.
	place/city1_uppercase/engza	South African settlement with over 100,000 inhabitants, in uppercase.
	place/city2/engza	South African settlement with between 10,000 and
	place/city2_uppercase/engza	

File	Entity	Description
	place/province/engza place/province_uppercase/engza place/district/engza place/district_uppercase/engza place/island/engza place/island_uppercase/engza	100,000 inhabitants. South African settlement with between 10,000 and 100,000 inhabitants, in uppercase. South African province. South African province in uppercase. South African district. South African district in uppercase. South African island. South African island in uppercase.
place_estee.ecr	place/city1/estee place/city1_uppercase/estee place/city2/estee place/city2_uppercase/estee place/county/estee place/county_uppercase/estee	Estonian settlement with over 100,000 inhabitants. Estonian settlement with over 100,000 inhabitants, in uppercase. Estonian settlement with between 10,000 and 100,000 inhabitants. Estonian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Estonian county. Estonian county in uppercase.
place_finfi.ecr	place/city1/finfi place/city1_uppercase/finfi place/city2/finfi place/city2_uppercase/finfi place/region/finfi place/region_uppercase/finfi	Finnish settlement with over 100,000 inhabitants. Finnish settlement with over 100,000 inhabitants, in uppercase. Finnish settlement with between 10,000 and 100,000 inhabitants. Finnish settlement with between 10,000 and 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/island/finfi place/island_uppercase/finfi	Finnish region. Finnish region in uppercase. Finnish island. Finnish island in uppercase.
place_frefr.ecr	place/city1/frefr place/city1_uppercase/frefr place/region_metro/frefr place/region_metro_uppercase/frefr place/department_metro/Alsace/frefr place/department_metro_uppercase/Alsace/frefr place/department_metro/Aquitaine/frefr place/department_metro_uppercase/Aquitaine/frefr place/department_metro/Auvergne/frefr place/department_metro_uppercase/Auvergne/frefr place/department_metro/BasseNormandie/frefr place/department_metro_uppercase/BasseNormandie/frefr place/department_metro/Bourgogne/frefr place/department_metro_uppercase/Bourgogne/frefr place/department_metro/Brittany/frefr place/department_metro_uppercase/Brittany/frefr place/department_metro/Centre/frefr place/department_metro_uppercase/Centre/frefr	French settlement with over 100,000 inhabitants. French settlement with over 100,000 inhabitants, in uppercase. French metropolitan regions. French metropolitan regions in uppercase. Departments of each French metropolitan region, in normal or uppercase.

File	Entity	Description
	place/department_metro/ChampagneArdenne/frefr place/department_metro_uppercase/ChampagneArdenne/frefr place/department_metro/Corsica/frefr place/department_metro_uppercase/Corsica/frefr place/department_metro/FrancheComte/frefr place/department_metro_uppercase/FrancheComte/frefr place/department_metro/HauteNormandie/frefr place/department_metro_uppercase/HauteNormandie/frefr place/department_metro/IleDeFrance/frefr place/department_metro_uppercase/IleDeFrance/frefr place/department_metro/LanguedocRoussillon/frefr place/department_metro_uppercase/LanguedocRoussillon/frefr place/department_metro/Limousin/frefr place/department_metro_uppercase/Limousin/frefr place/department_metro/Lorraine/frefr place/department_metro_uppercase/Lorraine/frefr place/department_metro/MidiPyrenees/frefr place/department_metro_uppercase/MidiPyrenees/frefr place/department_metro/NordPasDeCalais/frefr	

File	Entity	Description
	place/department_metro_uppercase/NordPasDeCalais/frefr	
	place/department_metro/PaysDeLaLoire/frefr	
	place/department_metro_uppercase/PaysDeLaLoire/frefr	
	place/department_metro/Picardie/frefr	
	place/department_metro_uppercase/Picardie/frefr	
	place/department_metro/PoitouCharentes/frefr	
	place/department_metro_uppercase/PoitouCharentes/frefr	
	place/department_metro/ProvenceAlpesCoteDAzur/frefr	
	place/department_metro_uppercase/ProvenceAlpesCoteDAzur/frefr	
	place/department_metro/RhoneAlpes/frefr	
	place/department_metro_uppercase/RhoneAlpes/frefr	
	place/department_metro/frefr	French metropolitan departments.
	place/department_metro_uppercase/frefr	French metropolitan departments in uppercase.
	place/departmentcode_metro/frefr	French metropolitan department INSEE codes.
	place/departmentcode_overseas/frefr	French overseas department INSEE codes.
	place/communecode_metro/frefr	French metropolitan commune INSEE codes.
	place/communecode_overseas/frefr	French overseas commune INSEE codes.

File	Entity	Description
	place/city/alsace/Bas_Rhin/frefr place/city_uppercase/alsace/Bas_Rhin/frefr place/city/alsace/Haut_Rhin/frefr place/city_uppercase/alsace/Haut_Rhin/frefr place/city/aquitaine/Dordogne/frefr place/city_uppercase/aquitaine/Dordogne/frefr place/city/aquitaine/Gironde/frefr place/city_uppercase/aquitaine/Gironde/frefr place/city/aquitaine/Landes/frefr place/city_uppercase/aquitaine/Landes/frefr place/city/aquitaine/Lot_et_Garonne/frefr place/city_uppercase/aquitaine/Lot_et_Garonne/frefr place/city/aquitaine/Pyrenees_Atlantiques/frefr place/city_uppercase/aquitaine/Pyrenees_Atlantiques/frefr place/city/auvergne/Allier/frefr place/city_uppercase/auvergne/Allier/frefr place/city/auvergne/Cantal/frefr place/city_uppercase/auvergne/Cantal/frefr place/city/auvergne/Haute_Loire/frefr place/city_uppercase/auvergne/Haute_Loire/frefr place/city/auvergne/Puy_de_Dome/frefr	Settlements in each French department, in normal or uppercase.

File	Entity	Description
	place/city_uppercase/auvergne/Puy_de_Dome/frefr place/city/basseNormandie/Calvados/frefr place/city_uppercase/basseNormandie/Calvados/frefr place/city/basseNormandie/Manche/frefr place/city_uppercase/basseNormandie/Manche/frefr place/city/basseNormandie/Orne/frefr place/city_uppercase/basseNormandie/Orne/frefr place/city/bourgogne/Cote_dOr/frefr place/city_uppercase/bourgogne/Cote_dOr/frefr place/city/bourgogne/Nievre/frefr place/city_uppercase/bourgogne/Nievre/frefr place/city/bourgogne/Saone_et_Loire/frefr place/city_uppercase/bourgogne/Saone_et_Loire/frefr place/city/bourgogne/Yonne/frefr place/city_uppercase/bourgogne/Yonne/frefr place/city/brittany/Cotes_dArmor/frefr place/city_uppercase/brittany/Cotes_dArmor/frefr place/city/brittany/Finistere/frefr place/city_uppercase/brittany/Finistere/frefr place/city/brittany/Ille_et_Vilaine/frefr place/city_uppercase/brittany/Ille_et_Vilaine/frefr	

File	Entity	Description
	place/city/brittany/Morbihan/frefr place/city_uppercase/brittany/Morbihan/frefr place/city/centre/Cher/frefr place/city_uppercase/centre/Cher/frefr place/city/centre/Eure_et_Loir/frefr place/city_uppercase/centre/Eure_et_Loir/frefr place/city/centre/Indre/frefr place/city_uppercase/centre/Indre/frefr place/city/centre/Indre_et_Loire/frefr place/city_uppercase/centre/Indre_et_Loire/frefr place/city/centre/Loir_et_Cher/frefr place/city_uppercase/centre/Loir_et_Cher/frefr place/city/centre/Loiret/frefr place/city_uppercase/centre/Loiret/frefr place/city/champagneArdenne/Ardennes/frefr place/city_uppercase/champagneArdenne/Ardennes/frefr place/city/champagneArdenne/Aube/frefr place/city_uppercase/champagneArdenne/Aube/frefr place/city/champagneArdenne/Marne/frefr place/city_uppercase/champagneArdenne/Marne/frefr place/city/champagneArdenne/Haute_Marne/frefr	

File	Entity	Description
	place/city_uppercase/champagneArdenne/Haute_Marne/frefr	
	place/city/corsica/Corse_du_Sud/frefr	
	place/city_uppercase/corsica/Corse_du_Sud/frefr	
	place/city/corsica/Haute_Corse/frefr	
	place/city_uppercase/corsica/Haute_Corse/frefr	
	place/city/francheComte/Doubs/frefr	
	place/city_uppercase/francheComte/Doubs/frefr	
	place/city/francheComte/Jura/frefr	
	place/city_uppercase/francheComte/Jura/frefr	
	place/city/francheComte/Haute_Saone/frefr	
	place/city_uppercase/francheComte/Haute_Saone/frefr	
	place/city/francheComte/Territoire_de_Belfort/frefr	
	place/city_uppercase/francheComte/Territoire_de_Belfort/frefr	
	place/city/hauteNormandie/Eure/frefr	
	place/city_uppercase/hauteNormandie/Eure/frefr	
	place/city/hauteNormandie/Seine_Maritime/frefr	
	place/city_uppercase/hauteNormandie/Seine_Maritime/frefr	
	place/city/ileDeFrance/Seine_et_Marne/frefr	
	place/city_uppercase/ileDeFrance/Seine_et_Marne/frefr	

File	Entity	Description
	place/city/ileDeFrance/Yvelines/frefr place/city_uppercase/ileDeFrance/Yvelines/frefr place/city/ileDeFrance/Essonne/frefr place/city_uppercase/ileDeFrance/Essonne/frefr place/city/ileDeFrance/Hauts_de_Seine/frefr place/city_uppercase/ileDeFrance/Hauts_de_Seine/frefr place/city/ileDeFrance/Seine_Saint_Denis/frefr place/city_uppercase/ileDeFrance/Seine_Saint_Denis/frefr place/city/ileDeFrance/Val_de_Marne/frefr place/city_uppercase/ileDeFrance/Val_de_Marne/frefr place/city/ileDeFrance/Val_dOise/frefr place/city_uppercase/ileDeFrance/Val_dOise/frefr place/city/ileDeFrance/Paris/frefr place/city_uppercase/ileDeFrance/Paris/frefr place/city/languedocRoussillon/Aude/frefr place/city_uppercase/languedocRoussillon/Aude/frefr place/city/languedocRoussillon/Gard/frefr place/city_uppercase/languedocRoussillon/Gard/frefr place/city/languedocRoussillon/Herault/frefr	

File	Entity	Description
	place/city_ uppercase/languedocRoussillon/Herault/frefr	
	place/city/languedocRoussillon/Lozere/frefr	
	place/city_ uppercase/languedocRoussillon/Lozere/frefr	
	place/city/languedocRoussillon/Pyrenees_ Orientales/frefr	
	place/city_ uppercase/languedocRoussillon/Pyrenees_ Orientales/frefr	
	place/city/limousin/Correze/frefr	
	place/city_uppercase/limousin/Correze/frefr	
	place/city/limousin/Creuse/frefr	
	place/city_uppercase/limousin/Creuse/frefr	
	place/city/limousin/Haute_Vienne/frefr	
	place/city_uppercase/limousin/Haute_Vienne/frefr	
	place/city/lorraine/Meurthe_et_Moselle/frefr	
	place/city_uppercase/lorraine/Meurthe_et_ Moselle/frefr	
	place/city/lorraine/Meuse/frefr	
	place/city_uppercase/lorraine/Meuse/frefr	
	place/city/lorraine/Moselle/frefr	
	place/city_uppercase/lorraine/Moselle/frefr	
	place/city/lorraine/Vosges/frefr	
	place/city_uppercase/lorraine/Vosges/frefr	

File	Entity	Description
	place/city/midiPyrenees/Ariege/frefr place/city_uppercase/midiPyrenees/Ariege/frefr place/city/midiPyrenees/Aveyron/frefr place/city_uppercase/midiPyrenees/Aveyron/frefr place/city/midiPyrenees/Haute_Garonne/frefr place/city_uppercase/midiPyrenees/Haute_Garonne/frefr place/city/midiPyrenees/Gers/frefr place/city_uppercase/midiPyrenees/Gers/frefr place/city/midiPyrenees/Lot/frefr place/city_uppercase/midiPyrenees/Lot/frefr place/city/midiPyrenees/Hautes_Pyrenees/frefr place/city_uppercase/midiPyrenees/Hautes_Pyrenees/frefr place/city/midiPyrenees/Tarn/frefr place/city_uppercase/midiPyrenees/Tarn/frefr place/city/midiPyrenees/Tarn_et_Garonne/frefr place/city_uppercase/midiPyrenees/Tarn_et_Garonne/frefr place/city/nordPasDeCalais/Nord/frefr place/city_uppercase/nordPasDeCalais/Nord/frefr place/city/nordPasDeCalais/Pas_de_Calais/frefr place/city_uppercase/nordPasDeCalais/Pas_de_Calais/frefr	

File	Entity	Description
	place/city/paysDeLaLoire/Loire_Atlantique/frefr place/city_uppercase/paysDeLaLoire/Loire_Atlantique/frefr place/city/paysDeLaLoire/Maine_et_Loire/frefr place/city_uppercase/paysDeLaLoire/Maine_et_Loire/frefr place/city/paysDeLaLoire/Mayenne/frefr place/city_uppercase/paysDeLaLoire/Mayenne/frefr place/city/paysDeLaLoire/Sarthe/frefr place/city_uppercase/paysDeLaLoire/Sarthe/frefr place/city/paysDeLaLoire/Vendee/frefr place/city_uppercase/paysDeLaLoire/Vendee/frefr place/city/picardie/Aisne/frefr place/city_uppercase/picardie/Aisne/frefr place/city/picardie/Oise/frefr place/city_uppercase/picardie/Oise/frefr place/city/picardie/Somme/frefr place/city_uppercase/picardie/Somme/frefr place/city/poitouCharentes/Charente/frefr place/city_uppercase/poitouCharentes/Charente/frefr place/city/poitouCharentes/Charente_Maritime/frefr place/city_uppercase/poitouCharentes/Charente_Maritime/frefr	

File	Entity	Description
	place/city/poitouCharentes/Deux_Sevres/frefr place/city_uppercase/poitouCharentes/Deux_Sevres/frefr place/city/poitouCharentes/Vienne/frefr place/city_uppercase/poitouCharentes/Vienne/frefr place/city/provenceAlpesCoteDAzur/Alpes_de_Haute_Provence/frefr place/city_uppercase/provenceAlpesCoteDAzur/Alpes_de_Haute_Provence/frefr place/city/provenceAlpesCoteDAzur/Hautes_Alpes/frefr place/city_uppercase/provenceAlpesCoteDAzur/Hautes_Alpes/frefr place/city/provenceAlpesCoteDAzur/Alpes_Maritimes/frefr place/city_uppercase/provenceAlpesCoteDAzur/Alpes_Maritimes/frefr place/city/provenceAlpesCoteDAzur/Bouches_du_Rhone/frefr place/city_uppercase/provenceAlpesCoteDAzur/Bouches_du_Rhone/frefr place/city/provenceAlpesCoteDAzur/Var/frefr place/city_	

File	Entity	Description
	uppercase/provenceAlpesCoteDAzur/Var/frefr place/city/provenceAlpesCoteDAzur/Vaucluse/frefr place/city_uppercase/provenceAlpesCoteDAzur/Vaucluse/frefr place/city/rhoneAlpes/Ain/frefr place/city_uppercase/rhoneAlpes/Ain/frefr place/city/rhoneAlpes/Ardeche/frefr place/city_uppercase/rhoneAlpes/Ardeche/frefr place/city/rhoneAlpes/Drome/frefr place/city_uppercase/rhoneAlpes/Drome/frefr place/city/rhoneAlpes/Isere/frefr place/city_uppercase/rhoneAlpes/Isere/frefr place/city/rhoneAlpes/Loire/frefr place/city_uppercase/rhoneAlpes/Loire/frefr place/city/rhoneAlpes/Rhone/frefr place/city_uppercase/rhoneAlpes/Rhone/frefr place/city/rhoneAlpes/Savoie/frefr place/city_uppercase/rhoneAlpes/Savoie/frefr place/city/rhoneAlpes/Haute_Savoie/frefr place/city_uppercase/rhoneAlpes/Haute_Savoie/frefr	
	place/city/frefr place/city_uppercase/frefr	French cities. French cities in uppercase.

File	Entity	Description
place_fregf.ecr	place/city2/fregf	French Guianan settlement with over 10,000 inhabitants.
	place/city2_uppercase/fregf	French Guianan settlement with over 10,000 inhabitants, in uppercase.
	place/canton/fregf	French Guianan canton.
	place/canton_uppercase/fregf	French Guianan canton in uppercase.
place_geo_dut.ecr	place/country/dut	Country in Dutch.
	place/country_capital/dut	Country capital in Dutch.
place_geo_eng.ecr	place/region/eng	Regions. For example, *Asia-Pacific*.
	place/region_uppercase/eng	Regions in uppercase.
	place/continent/eng	Continents. For example, *Africa*.
	place/continent_uppercase/eng	Continents in uppercase.
	place/ocean/eng	Oceans. For example, *Pacific*.
	place/ocean_uppercase/eng	Oceans in uppercase.
	place/country/eng	Countries. For example, *Australia*.
	place/country_uppercase/eng	Countries in uppercase.
	place/country_capital/eng	Country capitals. For example, *Canberra*.
	place/country_capital_uppercase/eng	Country capitals in uppercase.
	place/direction/eng	Directions. For example, *Southwest*.
	place/direction_uppercase/eng	Directions in uppercase.
	place/direction_abb/eng	Direction abbreviations. For example, *SW.*
	place/direction_mod/eng	Direction modifiers. For example, *Southwestern, Central, Downtown*.
	place/direction_mod_uppercase/eng	

File	Entity	Description
	place/area/eng place/area_uppercase/eng place/street_type/eng place/street_type_uppercase/eng	Direction modifiers in uppercase. Areas. For example, *Cape, Canyon, Grassland, Peninsula.* Areas in uppercase. Street types. For example, *Ave*, *Street*, *Place.* Street types in uppercase.
place_geo_fre.ecr	place/street_type/fre place/street_type_lowercase/fre place/street_type_uppercase/fre place/house_type/fre place/house_type_uppercase/fre place/direction/fre place/direction_uppercase/fre place/direction_abb/fre	Street types in French. For example, *Chauss*, *Cloitre*. Street types in lowercase French. For example, *chauss*, *cloitre*. Street types in uppercase French. For example, *CHAUSS*, *CLOITRE*. House types in French. For example, *Residence*, *Batiment*. House types in uppercase French. Directions in French. For example, *Sudouest*. Directions in uppercase French. Direction abbreviations in French. For example, *NO*.
place_gerat.ecr	place/city1/gerat place/city1_uppercase/gerat place/city2/gerat place/city2_uppercase/gerat place/state/gerat place/state_uppercase/gerat	Austrian settlement with over 100,000 inhabitants. Austrian settlement with over 100,000 inhabitants, in uppercase. Austrian settlement with between 10,000 and 100,000 inhabitants. Austrian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Austrian state.

File	Entity	Description
		Austrian state in uppercase.
place_gerde.ecr	place/state/gerde	German states.
	place/state_uppercase/gerde	German states in uppercase.
	place/state_abbrev/gerde	German state abbreviations.
	place/city/state_capital/gerde	German state capitals.
	place/city_uppercase/state_capital/gerde	German state capitals in uppercase.
	place/city/bw/gerde	Settlements in each German state, in normal or uppercase.
	place/city_uppercase/bw/gerde	
	place/city/by/gerde	
	place/city_uppercase/by/gerde	
	place/city/be/gerde	
	place/city_uppercase/be/gerde	
	place/city/bb/gerde	
	place/city_uppercase/bb/gerde	
	place/city/hb/gerde	
	place/city_uppercase/hb/gerde	
	place/city/hh/gerde	
	place/city_uppercase/hh/gerde	
	place/city/he/gerde	
	place/city_uppercase/he/gerde	
	place/city/mv/gerde	
	place/city_uppercase/mv/gerde	

File	Entity	Description
	place/city/ni/gerde	
	place/city_uppercase/ni/gerde	
	place/city/nw/gerde	
	place/city_uppercase/nw/gerde	
	place/city/rp/gerde	
	place/city_uppercase/rp/gerde	
	place/city/sl/gerde	
	place/city_uppercase/sl/gerde	
	place/city/sn/gerde	
	place/city_uppercase/sn/gerde	
	place/city/st/gerde	
	place/city_uppercase/st/gerde	
	place/city/sh/gerde	
	place/city_uppercase/sh/gerde	
	place/city/th/gerde	
	place/city_uppercase/th/gerde	
	place/city1/gerde	German settlement with more than 100,000 inhabitants.
	place/city1_uppercase/gerde	German settlement with more than 100,000 inhabitants, in uppercase.
	place/city/gerde	German cities.
	place/city_uppercase/gerde	German cities in uppercase.
place_hrvhr.ecr	place/city1/hrvhr	Croatian settlement with over 100,000 inhabitants.

File	Entity	Description
	place/city1_uppercase/hrvhr place/city2/hrvhr place/city2_uppercase/hrvhr place/county/hrvhr place/county_uppercase/hrvhr	Croatian settlement with over 100,000 inhabitants, in uppercase. Croatian settlement with between 10,000 and 100,000 inhabitants. Croatian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Croatian county. Croatian county in uppercase.
place_hunhu.ecr	place/city1/hunhu place/city1_uppercase/hunhu place/city2/hunhu place/city2_uppercase/hunhu place/county/hunhu place/county_uppercase/hunhu	Hungarian settlement with over 100,000 inhabitants. Hungarian settlement with over 100,000 inhabitants, in uppercase. Hungarian settlement with between 10,000 and 100,000 inhabitants. Hungarian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Hungarian county. Hungarian county in uppercase.
place_itait.ecr	place/city1/itait place/city1_uppercase/itait place/city2/itait place/city2_uppercase/itait place/region_abbreviation/itait place/region/itait place/region_uppercase/itait	Italian settlement with over 100,000 inhabitants. Italian settlement with over 100,000 inhabitants, in uppercase. Italian settlement with between 10,000 and 100,000 inhabitants. Italian settlement with between 10,000 and 100,000 inhabitants, in uppercase. 2-letter abbreviation for an Italian region. For example,

File	Entity	Description
	place/municipality/itait place/municipality_uppercase/itait place/island/itait place/island_uppercase/itait place/locality/itait place/locality_uppercase/itait	RM (includes SCV and RSM). Italian region. Italian region in uppercase. Italian municipality. Italian municipality in uppercase. Italian island. Italian island in uppercase. Italian place. Italian place in uppercase.
place_jpnjp.ecr	place/prefecture/jpnjp place/region/jpnjp place/city/aichi/jpnjp place/city/akita/jpnjp place/city/aomori/jpnjp place/city/chiba/jpnjp place/city/ehime/jpnjp place/city/fukui/jpnjp place/city/fukuoka/jpnjp place/city/fukushima/jpnjp place/city/gifu/jpnjp place/city/gunma/jpnjp place/city/hiroshima/jpnjp	Japanese prefectures. Japanese regions. Settlements in each Japanese prefecture.

File	Entity	Description
	place/city/hokkaido/jpnjp place/city/hyogo/jpnjp place/city/ibaraki/jpnjp place/city/ishikawa/jpnjp place/city/iwate/jpnjp place/city/kagawa/jpnjp place/city/kagoshima/jpnjp place/city/kanagawa/jpnjp place/city/kochi/jpnjp place/city/kumamoto/jpnjp place/city/kyoto/jpnjp place/city/mie/jpnjp place/city/miyagi/jpnjp place/city/miyazaki/jpnjp place/city/nagano/jpnjp place/city/nagasaki/jpnjp place/city/nara/jpnjp place/city/niigata/jpnjp place/city/oita/jpnjp place/city/okayama/jpnjp place/city/okinawa/jpnjp place/city/osaka/jpnjp place/city/saga/jpnjp	

File	Entity	Description
	place/city/saitama/jpnjp	
	place/city/shiga/jpnjp	
	place/city/shimane/jpnjp	
	place/city/shizuoka/jpnjp	
	place/city/tochigi/jpnjp	
	place/city/tokushima/jpnjp	
	place/city/tokyo/jpnjp	
	place/city/tottori/jpnjp	
	place/city/toyama/jpnjp	
	place/city/wakayama/jpnjp	
	place/city/yamagata/jpnjp	
	place/city/yamaguchi/jpnjp	
	place/city/yamanashi/jpnjp	
	place/city/jpnjp	Japanese settlements.
	place/misc/jpnjp	Japanese places.
place_kokr.ecr	place/province/korkr	Province of South Korea, in Korean language.
	place/province_DPRK/korkr	Province of North Korea (DPRK) as claimed by South Korea (Republic of Korea), in Korean language.
	place/district/korkr	District of South Korea, in Korean language.
	place/city1/korkr	Settlement in South Korea with over 100,000 inhabitants, in Korean language.
	place/city_DPRK/korkr	
	place/city2/korkr	

File	Entity	Description
		Settlement in North Korea (DPRK) as claimed by South Korea (Republic of Korea), in Korean language. Settlement in South Korea with between 10,000 and 100,000 inhabitants, in Korean language.
place_lat_long.ecr	place/lat_long place/utm	Geographical co-ordinate in any format (minimum precision is 1/10 degree or one minute of a degree). Supports the components NS, EW, LAT_DEGREES, LAT_DECIMAL, LAT_MINUTES, LAT_SECONDS, LONG_DEGREES, LONG_DECIMAL, LONG_MINUTES, and LONG_SECONDS You can use the `lat_long.lua` script to process this entity. Geographical co-ordinate written using the Universal Transverse Mercator convention. Supports no components.
place_lavlv.ecr	place/city1/lavlv place/city1_uppercase/lavlv place/city2/lavlv place/city2_uppercase/lavlv place/municipality/lavlv place/municipality_uppercase/lavlv	Latvian settlement with over 100,000 inhabitants. Latvian settlement with over 100,000 inhabitants, in uppercase. Latvian settlement with between 10,000 and 100,000 inhabitants. Latvian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Latvian municipality. Latvian municipality in uppercase.
place_litlt.ecr	place/city1/litlt place/city1_uppercase/litlt place/city2/litlt	Lithuanian settlement with over 100,000 inhabitants. Lithuanian settlement with over 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/city2_uppercase/litlt place/county/litlt place/county_uppercase/litlt	Lithuanian settlement with between 10,000 and 100,000 inhabitants. Lithuanian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Lithuanian county. Lithuanian county in uppercase.
place_mil_engus.ecr	place/mil/engus place/mil_uppercase/engus	U.S. military places. U.S. military places in uppercase.
place_mulbe.ecr	place/city1/mulbe place/city1_uppercase/mulbe place/city2/mulbe place/city2_uppercase/mulbe place/province/mulbe place/province_uppercase/mulbe place/region/mulbe place/region_uppercase/mulbe	Belgian settlement with over 100,000 inhabitants. Belgian settlement with over 100,000 inhabitants, in uppercase. Belgian settlement with between 10,000 and 100,000 inhabitants. Belgian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Belgian province. Belgian province in uppercase. Belgian region. Belgian region in uppercase.
place_mulch.ecr	place/city1/mulch place/city1_uppercase/mulch place/city2/mulch place/city2_uppercase/mulch place/canton/mulch	Swiss settlement with over 100,000 inhabitants. Swiss settlement with over 100,000 inhabitants, in uppercase. Swiss settlement with between 10,000 and 100,000 inhabitants.

File	Entity	Description
	place/canton_uppercase/mulch place/canton_abbr/mulch	Swiss settlement with between 10,000 and 100,000 inhabitants, in uppercase. Swiss canton. Swiss canton in uppercase. Two-letter abbreviation for a Swiss canton (always uppercase).
place_mullu.ecr	place/city2/mullu place/city2_uppercase/mullu place/district/mullu place/district_uppercase/mullu place/canton/mullu place/canton_uppercase/mullu	Luxembourgish city. Luxembourgish city in uppercase. Luxembourgish district. Luxembourgish district in uppercase. Luxembourgish canton. Luxembourgish canton in uppercase.
place_norno.ecr	place/city1/norno place/city1_uppercase/norno place/city2/norno place/city2_uppercase/norno place/county/norno place/county_uppercase/norno place/island/norno place/island_uppercase/norno	Norwegian settlement with over 100,000 inhabitants. Norwegian settlement with over 100,000 inhabitants, in uppercase. Norwegian settlement with between 10,000 and 100,000 inhabitants. Norwegian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Norwegian county. Norwegian county in uppercase. Norwegian island. Norwegian island in uppercase.
place_polpl.ecr	place/city1/polpl	Polish settlement with over 100,000 inhabitants.

File	Entity	Description
	place/city1_uppercase/polpl place/city2/polpl place/city2_uppercase/polpl place/province/polpl place/province_uppercase/polpl place/county/polpl place/county_uppercase/polpl place/province/polpl place/province_uppercase/polpl place/county/polpl place/county_uppercase/polpl	Polish settlement with over 100,000 inhabitants, in uppercase. Polish settlement with between 10,000 and 100,000 inhabitants. Polish settlement with between 10,000 and 100,000 inhabitants, in uppercase. Polish province. Polish province in uppercase. Polish county. Polish county in uppercase. Polish province (in English). Polish province in uppercase (in English). Polish county (in English). Polish county in uppercase (in English).
place_porbr.ecr	place/city1/porbr place/city1_uppercase/porbr place/city2/porbr place/city2_uppercase/porbr place/state/porbr place/state_uppercase/porbr place/island/porbr place/island_uppercase/porbr	Brazilian settlement with over 100,000 inhabitants. Brazilian settlement with over 100,000 inhabitants, in uppercase. Brazilian settlement with between 10,000 and 100,000 inhabitants. Brazilian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Brazilian state. Brazilian state in uppercase. Brazilian island.

File	Entity	Description
		Brazilian island in uppercase.
place_porpt.ecr	place/city1/porpt place/city1_uppercase/porpt place/city2/porpt place/city2_uppercase/porpt place/district/porpt place/district_uppercase/porpt place/island/porpt place/island_uppercase/porpt	Portuguese settlement with over 100,000 inhabitants. Portuguese settlement with over 100,000 inhabitants, in uppercase. Portuguese settlement with between 10,000 and 100,000 inhabitants. Portuguese settlement with between 10,000 and 100,000 inhabitants, in uppercase. Portuguese district. Portuguese district in uppercase. Portuguese island. Portuguese island in uppercase.
place_rummd.ecr	place/city1/rummd place/city1_uppercase/rummd place/city2/rummd place/city2_uppercase/rummd place/district/rummd place/district_uppercase/rummd	Moldovan settlement with over 100,000 inhabitants. Moldovan settlement with over 100,000 inhabitants, in uppercase. Moldovan settlement with between 10,000 and 100,000 inhabitants. Moldovan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Moldovan district. Moldovan district in uppercase.
place_rumro.ecr	place/city1/rumro place/city1_uppercase/rumro place/city2/rumro	Romanian settlement with over 100,000 inhabitants. Romanian settlement with over 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/city2_uppercase/rumro place/county/rumro place/county_uppercase/rumro	Romanian settlement with between 10,000 and 100,000 inhabitants. Romanian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Romanian county. Romanian county in uppercase.
place_slksk.ecr	place/city1/slksk place/city1_uppercase/slksk place/city2/slksk place/city2_uppercase/slksk place/region/slksk place/region_uppercase/slksk	Slovakian settlement with over 100,000 inhabitants. Slovakian settlement with over 100,000 inhabitants, in uppercase. Slovakian settlement with between 10,000 and 100,000 inhabitants. Slovakian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Slovakian region. Slovakian region in uppercase.
place_slvsi.ecr	place/city1/slvsi place/city1_uppercase/slvsi place/city2/slvsi place/city2_uppercase/slvsi	Slovenian settlement with over 100,000 inhabitants. Slovenian settlement with over 100,000 inhabitants, in uppercase. Slovenian settlement with between 10,000 and 100,000 inhabitants. Slovenian settlement with between 10,000 and 100,000 inhabitants, in uppercase.
place_spaar.ecr	place/city1/spaar place/city1_uppercase/spaar place/city2/spaar	Argentinian settlement with over 100,000 inhabitants. Argentinian settlement with over 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/city2_uppercase/spaar place/province/spaar place/province_uppercase/spaar place/island/spaar place/island_uppercase/spaar	Argentinian settlement with between 10,000 and 100,000 inhabitants. Argentinian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Argentinian province. Argentinian province in uppercase. Argentinian island. Argentinian island in uppercase.
place_spabo.ecr	place/city1/spabo place/city1_uppercase/spabo place/city2/spabo place/city2_uppercase/spabo place/department/spabo place/department_uppercase/spabo place/province/spabo place/province_uppercase/spabo	Bolivian settlement with over 100,000 inhabitants. Bolivian settlement with over 100,000 inhabitants, in uppercase. Bolivian settlement with between 10,000 and 100,000 inhabitants. Bolivian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Bolivian department. Bolivian department in uppercase. Bolivian province. Bolivian province in uppercase.
place_spacl.ecr	place/city1/spacl place/city1_uppercase/spacl place/city2/spacl place/city2_uppercase/spacl place/region/spacl	Chilean settlement with over 100,000 inhabitants. Chilean settlement with over 100,000 inhabitants, in uppercase. Chilean settlement with between 10,000 and 100,000 inhabitants.

File	Entity	Description
	place/region_uppercase/spacl place/commune/spacl place/commune_uppercase/spacl	Chilean settlement with between 10,000 and 100,000 inhabitants, in uppercase. Chilean region. Chilean region in uppercase. Chilean commune. Chilean commune in uppercase.
place_spaco.ecr	place/city1/spaco place/city1_uppercase/spaco place/city2/spaco place/city2_uppercase/spaco place/department/spaco place/department_uppercase/spaco	Colombian settlement with over 100,000 inhabitants. Colombian settlement with over 100,000 inhabitants, in uppercase. Colombian settlement with between 10,000 and 100,000 inhabitants. Colombian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Colombian department. Colombian department in uppercase.
place_spaec.ecr	place/city1/spaec place/city1_uppercase/spaec place/city2/spaec place/city2_uppercase/spaec place/province/spaec place/province_uppercase/spaec place/island/spaec place/island_uppercase/spaec	Ecuadorian settlement with over 100,000 inhabitants. Ecuadorian settlement with over 100,000 inhabitants, in uppercase. Ecuadorian settlement with between 10,000 and 100,000 inhabitants. Ecuadorian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Ecuadorian province. Ecuadorian province in uppercase. Ecuadorian island.

File	Entity	Description
		Ecuadorian island in uppercase.
place_spaes.ecr	place/city1/spaes place/city1_uppercase/spaes place/city2/spaes place/city2_uppercase/spaes place/region/spaes place/region_uppercase/spaes place/province/spaes place/province_uppercase/spaes place/island/spaes place/island_uppercase/spaes	Spanish settlements with over 100,000 inhabitants. Spanish settlements with over 100,000 inhabitants, in uppercase. Spanish settlements with between 10,000 and 100,000 inhabitants. Spanish settlements with between 10,000 and 100,000 inhabitants, in uppercase. Region in Spain. Region in Spain in uppercase. Province in Spain. Province in Spain in uppercase. Balearic and Canary Islands. Balearic and Canary Islands in uppercase.
place_spamx.ecr	place/city1/spamx place/city1_uppercase/spamx place/city2/spamx place/city2_uppercase/spamx place/state/spamx place/state_uppercase/spamx place/islands/spamx place/islands_uppercase/spamx	Mexican settlements with over 100,000 inhabitants. Mexican settlements with over 100,000 inhabitants, in uppercase. Mexican settlements with between 10,000 and 100,000 inhabitants. Mexican settlements with between 10,000 and 100,000 inhabitants, in uppercase. States in Mexico. States in Mexico in uppercase. Mexican islands.

File	Entity	Description
		Mexican islands in uppercase.
place_spape.ecr	place/city1/spape place/city1_uppercase/spape place/city2/spape place/city2_uppercase/spape place/region/spape place/region_uppercase/spape	Peruvian settlement with over 100,000 inhabitants. Peruvian settlement with over 100,000 inhabitants, in uppercase. Peruvian settlement with between 10,000 and 100,000 inhabitants. Peruvian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Peruvian region. Peruvian region in uppercase.
place_spapy.ecr	place/city1/spapy place/city1_uppercase/spapy place/city2/spapy place/city2_uppercase/spapy place/region/spapy place/region_uppercase/spapy place/commune/spapy place/commune_uppercase/spapy	Paraguayan settlement with over 100,000 inhabitants. Paraguayan settlement with over 100,000 inhabitants, in uppercase. Paraguayan settlement with between 10,000 and 100,000 inhabitants. Paraguayan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Paraguayan region. Paraguayan region in uppercase. Paraguayan commune. Paraguayan commune in uppercase.
place_spauy.ecr	place/city1/spauy place/city1_uppercase/spauy place/city2/spauy	Uruguayan settlement with over 100,000 inhabitants. Uruguayan settlement with over 100,000 inhabitants, in uppercase.

File	Entity	Description
	place/city2_uppercase/spauy place/department/spauy place/department_uppercase/spauy	Uruguayan settlement with between 10,000 and 100,000 inhabitants. Uruguayan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Uruguayan department. Uruguayan department in uppercase.
place_spave.ecr	place/city1/spave place/city1_uppercase/spave place/city2/spave place/city2_uppercase/spave place/region/spave place/region_uppercase/spave place/state/spave place/state_uppercase/spave place/island/spave place/island_uppercase/spave	Venezuelan settlement with over 100,000 inhabitants. Venezuelan settlement with over 100,000 inhabitants, in uppercase. Venezuelan settlement with between 10,000 and 100,000 inhabitants. Venezuelan settlement with between 10,000 and 100,000 inhabitants, in uppercase. Venezuelan region. Venezuelan region in uppercase. Venezuelan state. Venezuelan state in uppercase. Venezuelan island. Venezuelan island in uppercase.
place_srpme.ecr	place/city1/srpme place/city1_uppercase/srpme place/city2/srpme place/city2_uppercase/srpme	Montenegrin settlement with over 100,000 inhabitants. Montenegrin settlement with over 100,000 inhabitants, in uppercase. Montenegrin settlement with between 10,000 and 100,000 inhabitants.

File	Entity	Description
	place/municipality/srpme place/municipality_uppercase/srpme	Montenegrin settlement with between 10,000 and 100,000 inhabitants, in uppercase. Montenegrin municipality. Montenegrin municipality in uppercase.
place_srprs.ecr	place/city1/srprs place/city1_uppercase/srprs place/city2/srprs place/city2_uppercase/srprs place/district/srprs place/district_uppercase/srprs	Serbian settlement with over 100,000 inhabitants. Serbian settlement with over 100,000 inhabitants, in uppercase. Serbian settlement with between 10,000 and 100,000 inhabitants. Serbian settlement with between 10,000 and 100,000 inhabitants, in uppercase. Serbian district. Serbian district in uppercase.
place_swese.ecr	place/city1/swese place/city1_uppercase/swese place/city2/swese place/city2_uppercase/swese place/county/swese place/county_uppercase/swese place/island/swese place/island_uppercase/swese	Swedish settlement with over 100,000 inhabitants. Swedish settlement with over 100,000 inhabitants, in uppercase. Swedish settlement with between 10,000 and 100,000 inhabitants. Swedish settlement with between 10,000 and 100,000 inhabitants, in uppercase. Swedish county. Swedish county in uppercase. Swedish island. Swedish island in uppercase.

File	Entity	Description
place_turtr.ecr	place/city1/turtr	Turkish settlement with over 100,000 inhabitants.
	place/city1_uppercase/turtr	Turkish settlement with over 100,000 inhabitants, in uppercase.
	place/city2/turtr	Turkish settlement with between 10,000 and 100,000 inhabitants.
	place/city2_uppercase/turtr	Turkish settlement with between 10,000 and 100,000 inhabitants, in uppercase.
	place/region/turtr	Turkish region.
	place/region_uppercase/turtr	Turkish region in uppercase.
	place/province/turtr	Turkish province.
	place/province_uppercase/turtr	Turkish province in uppercase.
	place/district/turtr	Turkish district.
	place/district_uppercase/turtr	Turkish district in uppercase.
profanity_chi.ecr	profanity/biological/chi	Potentially offensive term in Chinese pertaining to biological processes (including obscured representations).
	profanity/sexual/chi	Potentially offensive term in Chinese pertaining to sex (including obscured representations).
	profanity/personal/chi	Directly insulting term in Chinese (including obscured representations).
	profanity/exclaim/chi	Potentially offensive term in Chinese pertaining to exclamation (including obscured representations).
	profanity/chi	Any potentially offensive Chinese term (including obscured representations). Eduction gives higher scores to matches with a greater tendency to offend.

File	Entity	Description
		The following `MinScore` parameter values are provided as a guide: • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases • `MinScore=2.5` returns no matches at all
	profanity/phrase/chi	Any potentially offensive Chinese phrase (including obscured representations). The following `MinScore` parameter values are provided as a guide: • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases • `MinScore=3.0` returns no matches at all
profanity_eng.ecr	profanity/blasphemous/eng profanity/homophobic/eng profanity/racial/eng profanity/personal/eng profanity/sexual/eng	Religious term often used for blasphemy (including obscured representations). Homophobic term (including obscured representations). Racial derogatory term (including obscured representations). Personally insulting term. Contains all racial and

File	Entity	Description
	profanity/biological/eng profanity/censored/eng profanity/eng	homophobic offensive terms (including obscured representations). Potentially offensive term pertaining to sex (including obscured representations). Potentially offensive term pertaining to biological processes (including obscured representations). Word that appears in the text in a fully-censored format. Any potentially-offensive English term (including obscured representations) Eduction gives higher scores to matches with a greater tendency to offend. The following `MinScore` parameter values are provided as a guide: • `MinScore=0.1` removes false matches, for example from URL shorteners • `MinScore=0.7` removes many weakly offensive terms and phrases • `MinScore=1.1` returns moderately-offensive terms and phrases • `MinScore=1.3` returns only strongly offensive terms and phrases • `MinScore=2.5` returns no matches at all

S

File	Entity	Description
sentiment_ara.ecr	sentiment/positive/ara	An Arabic phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/ara	An Arabic phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/ara	A positive or negative phrase in Arabic. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_chi.ecr	sentiment/positive/chi	A Chinese phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/chi	A Chinese phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/chi	A positive or negative phrase in Chinese. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_cze.ecr	sentiment/positive/cze	A Czech phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/cze	A Czech phrase that expresses a negative statement.

File	Entity	Description
		Supports the TOPIC and SENTIMENT components.
	sentiment/cze	A positive or negative phrase in Czech. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_dut.ecr	sentiment/positive/dut	A Dutch phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/dut	A Dutch phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_eng.ecr	sentiment/positive/eng	An English phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/eng	An English phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/eng	A positive or negative phrase in English. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_basic_eng.ecr	sentiment/positive/eng	If recall with `sentiment_eng.ecr` is too low, and your documents are generally short comments, use

File	Entity	Description
	sentiment/negative/eng sentiment/eng	`sentiment_basic_eng.ecr` to extract additional matches.This grammar contains carefully-selected lists of positive and negative terms that help determine the sentiment of a document in which `sentiment_eng.ecr` found no matches. TOPIC and SENTIMENT components are **not** supported. `sentiment_basic_eng.ecr` contains terms in title case, but research shows that for most data these impair recall, so these are given a lower score. HPE recommends that you set `EntityMinScore`*`N`* to **`0.4`** to filter out these terms unless you need them.
sentiment_fre.ecr	sentiment/positive/fre	A French phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/fre	A French phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/fre	A positive or negative phrase in French. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.
sentiment_ger.ecr	sentiment/positive/ger	A German phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/ger	A German phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/ger	A positive or negative phrase in German. This entity adds a POSITIVE or NEGATIVE component wrapper to an

File	Entity	Description
		empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_ita.ecr	sentiment/positive/ita	An Italian phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/ita	An Italian phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/ita	A positive or negative phrase in Italian. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_pol.ecr	sentiment/positive/pol	A Polish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/pol	A Polish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/pol	A positive or negative phrase in Polish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

File	Entity	Description
sentiment_por.ecr	sentiment/positive/por	A Portuguese phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/por	A Portuguese phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/por	A positive or negative phrase in Portuguese. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.
sentiment_rus.ecr	sentiment/positive/rus	A Russian phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/rus	A Russian phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/rus	A positive or negative phrase in Russian. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to **`:;@#`**.
sentiment_spa.ecr	sentiment/positive/spa	A Spanish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/spa	A Spanish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.

File	Entity	Description
	sentiment/spa	A positive or negative phrase in Spanish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.
sentiment_tur.ecr	sentiment/positive/tur	A Turkish phrase that expresses a positive statement. Supports the TOPIC and SENTIMENT components.
	sentiment/negative/tur	A Turkish phrase that expresses a negative statement. Supports the TOPIC and SENTIMENT components.
	sentiment/tur	A positive or negative phrase in Turkish. This entity adds a POSITIVE or NEGATIVE component wrapper to an empty string after the match. You can use this component to determine the sentiment of the phrase. Use this entity when faster performance is desirable. HPE recommends that you configure Eduction to allow all duplicates, and set `TangibleCharacters` to `:;@#`.

T

File	Entity	Description
team_american_football.ecr	org/football/us	American Football team in the U.S.
	org/football/ca	Canadian Football team in Canada. All synonyms for team names produce the same normalized text (for example, *The Bears* normalizes to *Chicago Bears*) to identify variant team names.

File	Entity	Description
team_baseball.ecr	org/baseball/mlb	Major League baseball team in the U.S. and Canada. All synonyms for team names produce the same normalized text (for example, *LA Dodgers* normalizes to *Los Angeles Dodgers*) to identify variant team names.
team_basketball.ecr	org/basketball/nba	Basketball team in the NBA. All synonyms for team names produce the same normalized text (for example, *Sixers* normalizes to *Philadelphia 76ers*) to identify variant team names.
team_hockey.ecr	org/hockey/nhl	Hockey team in the NHL. All synonyms for team names produce the same normalized text (for example, *NJ Devils* normalizes to *New Jersey Devils*) to identify variant team names.
team_soccer.ecr	org/soccer/us	Soccer team in U.S. and Canada (Major League Soccer).
	org/soccer/gb	Football (soccer) team in the United Kingdom. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Celtic*.
	org/soccer/de	Football (soccer) team in Germany (current Bundesliga teams). Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Wolfsburg*.
	org/soccer/fr	Football (soccer) team in France. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Nice*.
	org/soccer/nl	Football (soccer) team in the Netherlands. Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Ajax*.
	org/soccer/es	Football (soccer) team in Spain (current Primera & Segunda Divisiónes teams). Set `EntityMinScoreN=0.99`
	org/soccer/it	

File	Entity	Description
		to filter out ambiguous names such as *Barcelona*.
		Football (soccer) team in Italy (current teams in Serie A and Serie B). Set `EntityMinScoreN=0.99` to filter out ambiguous names such as *Inter*.
		All synonyms for team names produce the same normalized text (for example, *Man United* normalizes to *Manchester United*) to identify variant team names.
time_chi.ecr	time/time_of_day/chi	A descriptive time of day in Chinese.
	time/time_of_day_simplified/chi	A descriptive time of day in simplified Chinese.
	time/period/chi	An amount of time in Chinese.
	time/period_simplified/chi	An amount of time in simplified Chinese.
	time/alpha_time/chi	Time of the day in Chinese words.
	time/alpha_time_simplified/chi	Time of the day in simplified Chinese words and ASCII numbers.
	time/hms/chi	Time in hours and minutes with optional seconds and fractions thereof.
	time/hms_simplified/chi	Time in hours and minutes with optional seconds and fractions thereof, in simplified Chinese and ASCII numbers.
	time/chi	Any time of day in Chinese, in a variety of formats.
	time/simplified/chi	Any time of day in simplified Chinese and ASCII numbers, in a variety of formats.
time_eng.ecr	time/time_of_day/eng	A descriptive time of day in English. For example, *dawn*, *morning*, *Mid-afternoon.*
	time/period/eng	An amount of time. For example, *day*, *quarter*, *month*, *decades.*

File	Entity	Description
	time/alpha_time/eng time/hms/eng time/eng	Time of day in English words, for example, *4 o'clock*, *ten past five*. Time in hours and minutes with optional seconds and fractions thereof. Any time in English or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 a.m. • 02:20 at night • quarter past midnight • 20 to midnight • ten past six • One o'clock • 6.10pm • 1.49 in the afternoon • noon • 5:00 UTC+1 • 19:15 Hawaii-Aleutian Time
time_fre.ecr	time/time_of_day/fre time/period/fre time/alpha_time/fre	A descriptive time of day in French. For example, *l'aube*, *Matin*. An amount of time. For example, *une décennie*, *un siècle*. Time of day in French words. For example, *sept heures du matin*, *trois heures de l'après-midi*.

File	Entity	Description
	time/hms/fre time/fre	Time in hours and minutes with optional seconds and fractions thereof. Any time in French or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 du matin • 4.54 de la nuit • minuit et 15 • midi moins vingt • 6 heures 20 du soir • 1 heure 49 de l'apres midi • une heure trente cinq • six heures et dix • midi • 5:00 UTC+1 • 19:15 PDT
time_ger.ecr	time/time_of_day/ger time/period/ger time/alpha_time/ger time/hms/ger time/ger	A descriptive time of day. For example, *Nachmittag*. An amount of time in German (all declensions). For example, *Jahrzehnt*. Time of day in German words. For example, *fünf nach zehn*. Time in hours and minutes with optional seconds and fractions thereof.

File	Entity	Description
		Any time in German or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 morgens • 4.54 nachts • viertel nach mitternacht • 6.20 nachmittags • 1 Uhr 49 nachmittags • Fünf Uhr • Sechs Uhr Zehn • mittag • 5:00 UTC+1 • 19:15 PDT
time_ita.ecr	time/time_of_day/ita time/period/ita time/alpha_time/ita time/hms/ita time/ita	A descriptive time of day in Italian. For example, *pomeriggio*. An amount of time in Italian. For example, *giorno*, *Mesi*, *secolo*. Time of day in Italian words. For example, *Sono le 4*, *5 y 10 del pomeriggio*, *mezzanotte meno cinque.* Time in hours and minutes with optional seconds and fractions thereof. Any time in Italian or numeric format. Supported formats include:

File	Entity	Description
		• 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 4.54 del mattino • Tre e tre quarti di notte • un quarto alle sette • dieci all'una • sono le due meno cinque • 6 e 20 • 1 e 49 del pomeriggio • 13:35 • sei e dieci • Mezzo giorno • 5:00 UTC+1 • 19:15 PDT
time_numeric.ecr	time/hms12 time/hms24 time/tz_abbr time/tz_abbr_plus	12-hour time in hours and minutes, with optional seconds and fractions. 24-hour time in hours and minutes, with optional seconds and fractions. Standard timezone abbreviations. Standard timezone abbreviations with optional +/- hh:mm modifier.
time_por.ecr	time/time_of_day/por time/period/por	A descriptive time of day in Portuguese. For example, *manhã*, *pôr do dol.*

File	Entity	Description
	time/alpha_time/por time/hms/por time/por	An amount of time in Portuguese. For example, *dia*, *Mês*, *séculos.* Time of day in Portuguese words. For example, *São dez*, *doze e um quarto da noite*, *meia-noite menos 15*. Time in hours and minutes with optional seconds and fractions thereof. Any time in Portugese. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 4.54 da manhã • doze e quarto da noite (Brazilian Portuguese) • São vinte e cinco para as cinco da manhã • cinco e vinte da manhã • 1 e 49 da tarde • 13:35 • seis e dez • Meio-dia • 5:00 UTC+1 • 19:15 PDT • 7 em ponto
time_spa.ecr	time/time_of_day/spa	A descriptive time of day in Spanish. For example, *a la medianoche*, *al amanecer*.
	time/period/spa	An amount of time in Spanish. For example, *década*.

File	Entity	Description
	time/alpha_time/spa time/hms/spa time/spa	Time of day in Spanish words. For example, *a media mañana*. Time in hours and minutes with optional seconds and fractions thereof. Any time in Spanish or numeric format. Supported formats include: • 20:20 GMT+0100 • 00:15 • 4:54 • 20:20:20.2020202020202020 • 04:54 de la mañana • 4.54 por la noche • doce y cuarto de la noche • Son las cinco menos veinticinco de la mañana • cinco y veinte de la mañana • 1 y 49 de la tarde • 13:35 • seis y diez • mediodía • 5:00 UTC+1 • 19:15 PDT • 7 en punto
transport_airport.ecr	airport/icao airport/iata	Airport ICAO code. Airport IATA code.
transport_car.ecr	car/make_model	Make and model of car.

U

File	Entity	Description
university.ecr	org/university	A university.

Standard Grammar – Source

Eduction includes standard grammar files in source form (XML) and their compiled equivalents (ECR). The source files import compiled Eduction standard grammar files and illustrate sample usage. Customers can modify these XML source files and recompile them to customize a grammar for the needs of an Eduction application. The following table lists public entities defined in the XML source files. It excludes the public entities that are republished from the imported Eduction ECR grammar files.

File	Entity	Description
measure.xml	measure/all/eng	An editable collection of patterns that match length, area, volume, and mass.
money.xml[1]	money/all	All currency amounts.
pci_dss.xml	pci_dss/person_name/engus pci_dss/date/engus pci_dss/credit_card/engus pci_dss/bank_names/engus	Person names. Dates. Credit and debit card numbers. Bank names.
pii.xml	pii/person_name/engus pii/phone_number/engus pii/email_address/engus pii/ip_address/engus pii/social_security/engus pii/car_numberplate/engus	Personal names. Phone numbers. Email addresses. IP addresses. Social Security numbers. Car license plate numbers.

[1] When matching symbols in the money entities, the Eduction option `MatchWholeWord` must be set to `0` (false). Otherwise, when encountering a string such as $10.70, Eduction will not recognize that *$* is the start of a token. Instead, it looks only for matches starting on the *1* and on the *7,* and will not return *$10.70*.

File	Entity	Description
	pii/driver_license/engus pii/credit_card/engus pii/date/engus pii/country pii/state/engus pii/county/engus pii/city/engus pii/address/engus pii/zipcode/engus pii/age/engus pii/gender/engus pii/race/engus pii/job_title/engus pii/disease_and_condition/engus pii/account_number/engus pii/license_number/engus pii/facebook_url/engus	Driver’s license numbers. Credit and debit card numbers. Dates. Countries. U.S. states or possessions. U.S. counties. U.S. cities. Geographical addresses. U.S. zipcodes. Age. Gender. Race. Job title. Disease or medical condition. Generic account number with 6-8 digits in a predictable context. Generic license number with specific alphanumeric format. Example URL for a personal Web page (Facebook).
place_europe.xml	place/country/europe place/country_uppercase/europe place/city1/europe place/city1_	European country in English (and some local languages). European country in English and local languages (uppercase). European settlement with over 100,000 inhabitants, in local language. European settlement with over 100,000 inhabitants, in local language (uppercase). European settlement with between 10,000 and 100,000 inhabitants, in local language.

File	Entity	Description
	uppercase/europe place/city2/europe place/city2_uppercase/europe place/region/Europe place/region_uppercase/Europe	European settlement with between 10,000 and 100,000 inhabitants, in local language (uppercase). High-level administrative division, in local language. High-level administrative division, in local language (uppercase).
place_south_america.xml	place/country/south_america place/country_uppercase/south_america place/city1/south_america place/city1_uppercase/south_america place/city2/south_america place/city2_uppercase/south_america place/island/south_america place/island_uppercase/south_america place/region/south_america place/region_uppercase/south_america	South American country in English, Spanish, or Portuguese. South American country in English, Spanish, or Portuguese (uppercase). South American settlement with over 100,000 inhabitants, in local language. South American settlement with over 100,000 inhabitants, in local language (uppercase). South American settlement with between 10,000 and 100,000 inhabitants, in local language. South American settlement with between 10,000 and 100,000 inhabitants, in local language (uppercase). South American island, in local language. South American island, in local language (uppercase). High-level administrative division, in local language. High-level administrative division, in local language (uppercase).

File	Entity	Description
retention.xml	retention/admission_date retention/discharge_date retention/birth_date retention/age/eng	Admission date. Discharge date. Birth date. Age.
sample.xml	sample/solar_system	A simple entity for planets of the solar system.
sentiment_user_chi.xml	sentiment/user_client_name sentiment/user_client_brand sentiment/user_client_rv1_name sentiment/user_client_rv1_brand sentiment/user_third_party_company_name sentiment/user_third_party_company_brand sentiment/user_positive_adjective sentiment/user_negative_adjective sentiment/user_positive_noun sentiment/user_negative_noun	You can use these files to modify the sentiment analysis grammar files for the relevant languages to give access to extra domain-specific vocabulary.

File	Entity	Description
	sentiment/user_neutral_noun sentiment/user_positive_verb sentiment/user_negative_verb sentiment/user_neutral_verb sentiment/user_positive_idiom sentiment/user_negative_idiom	
sentiment_user_ara.xml sentiment_user_cze.xml sentiment_user_eng.xml sentiment_user_fre.xml sentiment_user_ger.xml sentiment_user_ita.xml sentiment_user_pol.xml sentiment_	sentiment/user_positive_adjective sentiment/user_negative_adjective sentiment/user_neutral_adjective sentiment/user_positive_adverb sentiment/user_negative_adverb sentiment/user_neutral_adverb sentiment/user_positive_noun sentiment/user_negative_	

File	Entity	Description
user_por.xml sentiment_user_rus.xml sentiment_user_spa.xml sentiment_user_tur.xml	noun sentiment/user_neutral_noun sentiment/user_positive_verb sentiment/user_negative_verb sentiment/user_neutral_verb sentiment/user_positive_match sentiment/user_negative_match sentiment/user_good_noun (English only)	

The entities above incorporate the compiled Eduction entities in combination with Eduction XML grammar to create additional entities. The XML illustrates how to use the compiled Eduction entities. You can modify these XML files and compile them into Eduction ECR files that can then be used for specific applications.

The Eduction grammar files have three advantages:

- Allows for fined-grained access to basic entities that include more complex entities. Allows you to customize the complex entities to increase the precision and recall of the matching process.
- Provides both the compiled ECR grammar files as well as source-form XML grammar files that reference them.
- Separate ECR files reduce the memory footprint and file size.

Chapter 7: Grammar Reference

Eduction uses Grammar files to identify and tag entities in documents. They are written in XML in a format specific to Eduction. They are then compiled, using the Eduction command-line tool, into ECR files that Eduction can easily read at runtime. Eduction includes a collection of standard grammar files that make it easy to identify common entities such as names and phone numbers. These are described in .

- Create and Edit Grammar Files
- Example Grammar Files
- Eduction Grammar DTD

Create and Edit Grammar Files

An Eduction grammar defines patterns for matching text in a document. A pattern is a combination of characters and operators. An operator is a sequence of special characters that match text by following the rules associated with the operator.

Pattern	Description	Matches
Smith|John	Match either *Smith* or *John*	*Smith*

Pattern	Description	Matches
		John
[0-9]{3}	Match a sequence of three characters in the range 0 through 9	*123* *456*

In the above example, the square bracket operators [] are used to match on any of the characters 0 through 9 and the curly braces {} are used to repeat the previous pattern three times.

Grammars are described using XML. The template that defines the XML that Eduction understands is contained in the file `edk.dtd`. When writing grammars for Eduction, HPE recommends that you reference `edk.dtd` at the start of the XML grammar file using the include statement, and that you use a DTD-compatible XML authoring tool to eliminate syntax errors and save time.

Here is an example of a simple Eduction grammar:

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "edk.dtd">
<grammars>
   <grammar name="mygrammar">
      <entity name="name" type="public">
         <pattern>Smith|John</pattern>
      </entity>
      <entity name="digits" type="public">
         <pattern>[0-9]{3}</pattern>
      </entity>
   </grammar>
</grammars>
```

This grammar defines two entities: `mygrammar/name` and `mygrammar/digits`.

For a more extensive set of example Eduction grammar files, see Example Grammar Files, on page 311.

Compile Grammars

After a grammar is written, compile the XML file into an ECR file using the Eduction command-line tool `edktool`. XML files are easy for people to read, but inefficient for computers to process. `edktool` transforms the XML file into an ECR file that is efficient for Eduction to use directly. An example of the `edktool` compile command is:

```
edktool c mygrammar.xml
```

This command produces the output file `mygrammar.ecr`.

Eduction Grammar Syntax

The tables in this section describe the Eduction grammar syntax defined in the `edk.dtd` (see Eduction Grammar DTD, on page 314).

In the tables, terms shown in *italics* are deprecated, but are kept for backward compatibility. The terms in angled brackets <> describe the value that must be inserted. Note that the XML elements, attributes, and values are defined in lower case. Although the Eduction compiler accepts uppercase element and attribute names, this functionality is deprecated, but retained for backward compatibility. The `edk.dtd` file represents the current definition for Eduction grammar files, and must be followed for all Eduction grammars.

Note: Two deprecated elements are missing from the tables below: `dictionary` and `entryset`. These are synonymous with `grammar` and `entity` respectively, and, although not documented, are retained for backward compatibility.

<grammars>

Element: grammars

Child Elements: include, grammar

Description: This is the top-level element in an Eduction grammar.

Example: `<grammars version="1.0" debug="true" case="sensitive">`

Attribute	Value	Default	Description
`version`	<version string>	none	An optional character string providing version information for the grammar.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` takes the value from the application level, which in the case of Eduction applications is usually `sensitive`.
`debug`	true *1/on/yes/true* false	false	Displays verbose information for the grammars element while `edktool` compiles the grammar.

Attribute	Value	Default	Description
	0/off/no/false		

<include>

Element: include

Child Elements: publish

Description: References another Eduction grammar file for inclusion.

Example: `<include path="winter_names.ecr" type="private"/>`

Attribute	Value	Default	Description
`path`	<path to the grammar file>		A value is required.
`type`	public private	public	The default setting of public allows entities in included XML grammars to retain their private/public visibility. (Included ECR grammars, by definition of a compiled grammar, only contain public entities.) Setting the type attribute to **`private`** hides the included public entities from being visible in the file that includes the grammar.

<publish>

Element: publish

Child Elements: <none>

Description: Makes a private entity public. The entity can be anywhere in an included XML file chain. Note that private entities cannot be accessed in a compiled ECR file, so that even if the name of the private entity is known, publish is not able to make it public.

Example: `<publish name="grammar2/g2e2"/>`

Attribute	Value	Default	Description
`name`	<entity name>		Makes a private entity in an included XML file public. A value is required.

<grammar>

Element: grammar

Child Elements: extern, entity

Description: Defines a grammar, which is a collection of entities. Entities are used for matching.

Example: `<grammar name="grammar1" case="inherited" extend="disallow" debug="inherited">`

Attribute	Value	Default	Description
`name`	<grammar name>		A value is required.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `grammars` parent.
`extend`	append replace disallow	disallow	Extends or replaces an existing grammar definition, or disallows this if one already exists.
`debug`	true *1/on/yes/true* false *0/off/no/false* inherited	inherited	Displays verbose information for the dictionary element during compilation. The value `inherited` accepts the debug mode of the `grammars` parent.

<extern>

Element: extern

Child Elements: <none>

Description: Identifies an external grammar by name so that the entities contained by the grammar do not have to explicitly name the grammar. For example, if another grammar is `grammar1` and an entity within it is `entity1`, then in the current grammar, the entity can be referred to as simply `entity1` rather than `grammar1/entity1`.

Example: `<extern name="grammar2"/>`

Attribute	Value	Default	Description
`name`	<grammar name>		Identifies the name of the grammar. A value is required.

<entity>

Element: entity

Child Elements: entry, pattern

Description: Defines an entity used for matching.

Example: `<entity name="entity1" type="public" case="insensitive" extend="disallow" debug="true">`

Attribute	Value	Default	Description
`name`	<grammar name>		A value is required.
`type`	public private	private	Defines the entity as public or private.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `grammars` parent.
`extend`	append replace disallow	disallow	Extends or replaces an existing entity definition.
`debug`	true *1/on/yes/true* false *0/off/no/false* inherited	inherited	Displays verbose information for the entity element during compilation. The value `inherited` accepts the debug mode of the `grammar` parent.

<entry>

Element: entry

Child Elements: headword, synonym

Description: An entry represents an individual entry that is matched in an entity. The entry has one or more attributes such as the actual phrase that is returned (the *headword*), the case, and so on.

Example:`<entry headword="mat" score=".3" case="inherited" debug="inherited">`

Attribute	Value	Default	Description
`headword`	#CDATA		The dictionary entry. Headword can be an attribute or a subelement, but it must be one or the other.
`score`	>= 0	1	Can be used to assign any weightings to the matches. • A score of `1` is the default score. • A score of `0` always excludes the matching tag from the

Attribute	Value	Default	Description
			results, and can be used to specify exceptions to grammar rules. You can use these weightings for a variety of purposes: • They can represent the confidence the grammar author has in the accuracy of the match (where a value of `1` represents certainty, and lower values represent lesser confidence). • They can represent the importance of a match - for example, in the sentiment grammars the scoring represents the strength of the sentiment in the match. Multiple scores are multiplied. For example, if a match on an entity has a score of `1.5`, and that entity is used in another entity that also has a score of `0.4`, the resulting score is `0.6`. If a minimum score is specified during extraction, only those matches with a sufficiently high score are extracted. You can also display the exact scores of any match during extraction. **Note:** Entries and patterns should be assigned a score no lower than `0.01` and no higher than `100`.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `entryset` parent.
`debug`	true *1/on/yes/true* false *0/off/no/false* inherited	inherited	Displays verbose information for the entry element during compilation. The value `inherited` accepts the debug mode of the `entryset` parent.

<headword>

Element: headword

Child Elements: <none>

Description: A headword is the sequence of characters that produce an entity match.

Example: See example in <entry>, on the previous page.

Attribute	Value	Default	Description
`<element contents>`	<the headword>		The headword value. Note that if the `entry` element contains a headword attribute, it cannot have a headword subelement.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `entry` parent.
`score`	>= 0	1	See description in <entry>, on page 304.

<synonym>

Element: synonym

Child Elements: <none>

Description: A synonym is an alternative sequence of characters to a headword. Synonym matching produces an entity match, but returns the headword in place of the matching synonym. For example, if you search for *dog* with the synonym *canine* enabled, matches for *canine* return as if they matched *dog*.

Example:

```
<entry headword="Vatican City">
   <synonym>The Vatican</synonym>
   <synonym>Holy See</synonym>
   <synonym>Città del Vaticano</synonym>
   <synonym>Citta del Vaticano</synonym>
</entry>
```

Attribute	Value	Default	Description
`<element contents>`	<the synonym>		The synonym value.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `entry` parent.

<pattern>

Element: pattern

Child Elements: <none>

Description: Defines a pattern used for matching.

Example: `<pattern score=".1" case="insensitive" replace="replacechars" insert_before="prefix_" insert_after="_suffix">cat</pattern>`

Attribute	Value	Default	Description
`pattern`	<actual pattern>		A value is required.
`score`	>= 0	1	See description in <entry>, on page 304.
`case`	sensitive *1/on/yes/true* insensitive *0/off/no/false* inherited	inherited	Determines whether a match is case sensitive. The value `inherited` accepts the case matching mode of the `grammars` parent.
`replace`	<text to replace the match>	<no default>	The matched text is replaced with the specified text.
`insert_before`	<text to insert before the match>	<no default>	Matched text is prefixed with the specified text.
`insert_after`	<text to insert after the match>	<no default>	Matched text is suffixed with the specified text.
`debug`	true *1/on/yes/true* false *0/off/no/false* inherited	inherited	Displays verbose information for the pattern element during compilation. The value `inherited` accepts the debug mode of the `entity` parent.

Regular Expressions

This section describes the regular expressions syntax that Eduction supports.

The engine’s parser interprets regular expression syntax nearly identically to the UNIX regular expression syntax. The engine’s regular expression syntax also includes some extensions for matching substrings.

Operators

The following table the base regular expression operators available in the Eduction engine and the pattern the operator matches.

Operator	Matched Pattern
\	Quote the next metacharacter.
^	Match the beginning of a line.
$	Match the end of a line.
.	Match any character (except newline).
|	Alternation.
()	Used for grouping to force operator precedence.
[xy]	The character x or y.
[x-z]	The range of characters between x and z.
[^z]	Any character except z. **Note:** For performance reasons, HPE recommends that you explicitly list all the characters that you want to match, rather than using this operator.

Quantifiers

Operator	Matched Pattern
*	Match 0 or more times.
+	Match 1 or more times.
?	Match 0 or 1 times.
{n}	Match exactly *n* times.
{n,}	Match at least *n* times.
{n,m}	Match at least *n* times, but no more than *m* times.

Metacharacters

Operator	Matched Pattern
\t	Match tab.
\n	Match newline.
\r	Match return.
\f	Match formfeed.
\a	Match alarm (bell, beep, and so on).

Operator	Matched Pattern
`\e`	Match escape.
`\v`	Match vertical tab.
`\021`	Match octal character (in this example, 21 octal).
`\xF0`	Match hex character (in this example, F0 hex).
`\x{263a}`	Match wide hex character (Unicode).
`\w`	Match word character: [A-Za-z0-9_].
`\W`	Match non-word character: [^A-Za-z0-9_].
`\s`	Match whitespace character. This metacharacter also includes `\n` and `\r`: [\t\n\r].
`\S`	Match non-whitespace character: [^ \t\n\r].
`\d`	Match digit character: [0-9].
`\D`	Match non-digit character: [^0-9].
`\b`	Match word boundary.
`\B`	Match non-word boundary.
`\A`	Match start of string (never match at line breaks).
`\Z`	Match end of string. Never match at line breaks; only match at the end of the final buffer of text submitted for matching.

Extensions

Operator	Matched Pattern

`(?A:`*`entity`*`)`	Match a previously defined entity, which is then copied into the new entity’s definition.
	For example:
	`<include path="number_types_eng.ecr"/>`
	`    <entity name="fracpos" type="private">`
	`       <pattern>(?A:number/fracalpha/eng)</pattern>`
	`    </entity>`
	Copying an entity improves pattern execution speed, but increases compilation time and memory usage. It is recommended unless the copied entry is large and is copied multiple times.

`(?A^`*`entity`*`)`	Match a previously defined entity, which is then referenced by the new entity. Referencing an entity minimizes the size and memory usage of the grammar, but decreases performance. The performance impact can vary from unnoticeable to

Operator	Matched Pattern
	significant, depending on the size and structure of the grammar.

`(?A!` *`expr`*`)`	Match the expression *`expr`* but exclude its output. Designates an expression that helps identify an entity, but is not part of it.
	For example:
	`<grammars>`
	`   <grammar name="person">`
	`      <entity name="age" type="public">`
	`         <pattern>(?A!Age:\s)[1-9][0-9]?</pattern>`
	`      </entity>`
	`   </grammar>`
	`</grammars>`
	If this grammar is used to search the text
	`   Name: Simon. Age: 32. Address. 12 Fifth Street, Las Vegas.`
	the text `32` is returned but `12` is ignored because it does not have the prefix “`Age:`”, which is matched upon but excluded from the output.

`(?A=`*`component`*`:`*`expr`*`)`	Define a component within an entity’s definition. A component is a named part of an entity.
	For example, the following grammar defines `areacode` and `main` as components:
	`<grammars>`
	`   <grammar name="number">`
	`      <entity name="phone" type="public">`
	`         <pattern>(?A=areacode:[0-9]{3})-(?A=main:[0-9]{3}-[0-9]{4})`
	`</pattern>`
	`      </entity>`
	`   </grammar>`
	`</grammars>`
	If the data is as follows
	`   The phone number is 408-555-1342.`
	and the following configuration options are set
	`   <OutputSimpleMatchInfo>false</OutputSimpleMatchInfo>`
	`   <EnableComponents>true</EnableComponents>`
	then the output displays the `areacode` value `408` and the main value `555-1342` separately.

Token Properties

Caution: Token properties will be deprecated in a future release. Users should use the equivalent explicit regular expresions instead of token properties.

Operator	Match Pattern
`(?A:{properties})`	Matches a token that satisfies the list of properties provided. The properties are specified in a comma-separated list of one or more of the following: • num, alpha_num • all_caps, mixed_case, capword Any of these properties can be prefixed with the negation operator '!' for exclusion.

Example Grammar Files

The following sample grammar files contains the `gram_edk_place.xml` grammar.

grammar.xml

```
<?xml version="1.0" encoding="UTF-8"?>

<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<!-- Sample Eduction grammar file showing all elements and attributes in the DTD --
>
<grammars debug="true" case="sensitive">
  <include path="grammar_include.xml" type="private">
    <publish name="grammar2/g2e2"/>  <!-- publish previously private entity -->
  </include>
  <grammar name="grammar1" case="inherited" extend="disallow" debug="inherited">
    <extern name="grammar2"/> <!-- removes the need to refer explicitly to grammar2
-->
    <entity name="entity1" type="public" case="insensitive" extend="disallow"
debug="true">
    <!-- the following entity definitions are not useful but are provided only to
illustrate the options and combinations of elements and attributes available -->
    <pattern score=".1" case="insensitive" replace="replacechars" insert_
before="prefix_" insert_after="_suffix">cat</pattern>
    <pattern score=".2">sat</pattern>
    <entry headword="mat" score=".3" case="inherited" debug="inherited">
      <synonym case="inherited">rug</synonym> <!-- will locate rug but return mat -
->
```

```
      <!-- will locate rug but return mat -->
      <synonym case="inherited"><![CDATA[carpet]]></synonym> <!-- illustrates
allowing CDATA in this element -->
      </entry>
      <entry headword="dog" score=".6"/>
      <entry>
        <headword score=".8"><![CDATA[rabbit<hi!>&abc&amp;]]></headword>
        <synonym>bunny</synonym>
      </entry>
    </entity>
    <entity name="entity2" type="public">
      <pattern>(?A:g2e1)</pattern>
    </entity>
  </grammar>
</grammars>
```

grammar_include.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">
<grammars>
   <grammar name="grammar2">
      <entity name="g2e1">
         <pattern>animal</pattern> <!-- default visibility -->
      </entity>
      <entity name="g2e2" type="private"> <!-- explicitly private -->
         <pattern>mineral</pattern>
      </entity>
      <entity name="g2e3" type="public"> <!-- explicitly public -->
         <pattern>vegetable</pattern>
      </entity>
   </grammar>
</grammars>
```

Example Grammar File to Match Months

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE grammars SYSTEM "../published/edk.dtd">

<grammars version="4.0">
   <include path="winter_names.ecr" type="private"/>
   <grammar name="example">

      <entity name="spring_month" type="private">
         <pattern>[Mm]ar(ch|\.)</pattern>
         <entry headword="April"/>
         <entry headword="april"/>
         <entry headword="Apr"/>
         <entry headword="apr"/>
         <entry headword="Apr."/>
         <entry headword="apr."/>
         <pattern replace="May">[Mm]ay\.?</pattern>
         <entry headword="June">
            <synonym>Jun</synonym>
            <synonym>Jun.</synonym>
            <synonym>june</synonym>
            <synonym>jun</synonym>
            <synonym>jun.</synonym>
         </entry>
      </entity>

      <entity name="summer_month" type="private" case="insensitive">
         <entry headword="June"/>
         <entry headword="July"/>
         <entry headword="August"/>
         <entry headword="September"/>
      </entity>

      <entity name="month" type="public">
         <pattern>(?A^spring_month)</pattern>
         <pattern>(?A:summer_month)</pattern>
         <entry headword="September"/>
         <entry headword="October"/>
         <entry headword="November"/>
         <entry headword="December"/>
         <pattern>(?A^winter_month)</pattern>
         <!-- spelling mistakes -->
         <entry score="0.5" headword="Febuary"/>
      </entity>

   </grammar>
</grammars>
```

Simplified Grammar File Containing a Dictionary of Place Names

Note: The following grammar file is a simplified version provided for example purposes, rather than actual source code.

```
<entity name="city/spain" type="public">
    <entry headword="Barcelona"/>
    <entry headword="Ciudad Real"/>
    <entry headword="Granada"/>
    <entry headword="Madrid"/>
</entity>
<entity name="city/germany" type="headword">
    <entry headword="Berlin"/>
    <entry headword="Frankfurt"/>
    <entry headword="München"/>
    <entry headword="Leipzig"/>
</entity>
```

Simplified Grammar File Containing Patterns to Match Times of Day

Note: The following grammar file is a simplified version provided for example purposes, rather than actual source code.

```
<entity name="time_24_hour" type="public">
   <pattern>[01][0-9]:[0-5][0-9]</pattern>
   <pattern>2[0-3]:[0-5][0-9]</pattern>
</entity>

<entity name="time_all" type="public">
   <pattern>(?A:time_24_hour)</pattern>
   <entry headword="Midnight"/>
   <entry headword="midnight"/>
   <pattern>([1-9]|10|11|12) ?[ap]\.?m\.?</pattern>
</entity>
```

Eduction Grammar DTD

The XML DTD describing the Eduction grammar (such as, `edk.dtd`) is as follows:

```
<!ELEMENT grammars (include*, grammar*)>
<!ATTLIST grammars
version CDATA #IMPLIED
case (sensitive|insensitive|inherited) "inherited"
debug (true|false) "false"
>
<!ELEMENT include (publish*)>
<!ATTLIST include
path CDATA #REQUIRED
type (private|public) "public"
>
<!ELEMENT publish EMPTY>
<!ATTLIST publish
name CDATA #IMPLIED
>
<!ELEMENT grammar (extern*,entity+)>
<!ATTLIST grammar
name CDATA #REQUIRED
case (sensitive|insensitive|inherited) "inherited"
extend (append|replace|disallow) "disallow"
```

```
debug (true|false|inherited) "inherited"
>
<!ELEMENT extern EMPTY>
<!ATTLIST extern
name CDATA #REQUIRED
>
<!ELEMENT entity (entry*,pattern*)+>
<!ATTLIST entity
name CDATA #REQUIRED
type (private|public) "private"
case (sensitive|insensitive|inherited) "inherited"
extend (append|replace|disallow) "disallow"
debug (true|false|inherited) "inherited"
>
<!ELEMENT entry (headword?,synonym*)>
<!ATTLIST entry
headword CDATA #IMPLIED
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
debug (true|false|inherited) "inherited"
>
<!ELEMENT headword (#PCDATA)>
<!ATTLIST headword
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
>
<!ELEMENT synonym (#PCDATA)>
<!ATTLIST synonym
case (sensitive|insensitive|inherited) "inherited"
>
<!ELEMENT pattern (#PCDATA)>
<!ATTLIST pattern
score CDATA "1"
case (sensitive|insensitive|inherited) "inherited"
replace CDATA #IMPLIED
insert_before CDATA #IMPLIED
insert_after CDATA #IMPLIED
>
```

Appendix 1: Eduction Lua Methods Reference

This section describes the methods and parameters that you can use in your Lua post-processing scripts.

Method	Description
addComponent	Adds a new component to the match.
getComponent	Returns a specific component.
getComponentCount	Returns the total number of components in a match.
getEntityName	Returns the name of an entity in a match.
getMatchedText	Returns the input text for a match.
getName	Returns the name of a component.
getOffset	Returns the position of a match (in bytes).
getOffsetLength	Returns the position of a match (in characters).
getOutputText	Returns the output text for a match.
getScore	Returns the score of a match.
getText	Returns the text that is matched by a component.
setEntityName	Edits an entity name in a match.
setMatchedText	Edits the input text for match.
setName	Edits the name of a component.
setOffset	Edits the position of a match (in bytes).
setOffsetLength	Edits the position of a match (in bytes).
setOutputText	Assigns a new value to the output text for a match.
setScore	Edits the score of a match.
setText	Edits the text that is matched by a component.

addComponent

Adds a new component to the match. For example, if your Eduction task returns an email address as a match, you can use `addComponent` to extract the text after the @ symbol and add it as a DOMAIN component for the match.

You can also use addComponent to add metadata from other sources. For example, if you have extracted a place name, you can add components called “LATITUDE” and “LONGITUDE”, and populate them with data from a different source, regardless of the fact that they were not components of the original text.

Syntax

```
edkmatch:addComponent(name, offset, offsetLength)
```

Arguments

Argument	Description
name	The name of the new component (for example, TOPIC, or SENTIMENT)
offset	The position of the text in the match to use as the new component (in bytes).
offsetLength	The position of the text in the match to use as the new component (in characters).

Note: f you are unsure of the correct offset or offsetLength, you can specify offset=0 or offsetLength=0.

Returns

The new empty component object.

Related Topics

- getName, on page 320
- setName, on page 323
- getText, on page 322
- setText, on page 325

getComponent

The getComponent method returns a specified component object. For example, if you have six components, you can return the fourth component by using edkmatch:getComponent(4).

Note: Lua index values start from 1, not from 0 as in most other programming languages. For example, to get the third component object in a list, you must enter edkmatch:getComponent(3).

Syntax

```
edkmatch:getComponent(index)
```

Arguments

Argument	Description
`index`	The number of the component to get.

Returns

The component object at the specified index position in the match.

Related Topics

- getName, on the next page
- setName, on page 323
- getText, on page 322
- setText, on page 325

getComponentCount

Returns the total number of components in a match.

Syntax

```
edkmatch:getComponentCount()
```

Returns

The number of components.

getEntityName

Gets an entity name from a match.

Syntax

```
edkmatch:getEntityName()
```

Returns

The name of the entity in a match. You can use setEntityName to edit the name.

getMatchedText

Returns the input text for a particular match, that is, the text **before** any normalization or modification that occurs as part of the extraction process.

Syntax

```
edkmatch:getMatchedText()
```

Returns

The input text for a match. You can use setMatchedText to edit the text.

Related Topics

- getOutputText, on the next page

getName

Retrieves the name of a component.

Syntax

```
edkcomponent:getName()
```

Returns

The component name. You can use setName to edit the component name.

getOffset

Provides information on where in a document a particular match is found.

Syntax

```
edkmatch:getOffset()
```

Returns

The position of the match, in bytes. You can use setOffset to edit this information.

getOffsetLength

Provides information on where in a document a particular match is found.

Syntax

```
edkmatch:getOffsetLength()
```

Returns

The position of the match, in characters. You can use setOffsetLength to edit this information.

getOutputText

Returns the output text for a match, that is, the text **after** any normalization or modification that takes place as part of the extraction process.

Syntax

```
edkmatch:getOutputText()
```

Returns

The output text of a match. You can use setOutputText to edit the text.

Related Topics

- getMatchedText, on the previous page

getScore

Retrieves the score for a match.

Syntax

```
edkmatch:getScore()
```

Returns

The score for the match. You can use setScore to edit the score.

getText

Returns the output text that is matched by a particular component.

Syntax

```
edkcomponent:getText()
```

Returns

The matched text for a specified component. You can use setText to edit the text.

setEntityName

Edits the name of the entity that you retrieved by using getEntityName.

Syntax

```
edkmatch:setEntityName(new_name)
```

Arguments

Argument	Description
new_name	The new name for the entity in the match.

Returns

The new entity name.

setMatchedText

Edits the input text that you retrieved by using getMatchedText.

The input text is the text **before** any normalization or modification that takes place as part of the extraction process. By contrast, setOutputText enables you to edit the output text after any changes.

Syntax

```
edkmatch:setMatchedText(new_text)
```

Arguments

Argument	Description
`new_text`	The new value that you want to assign to the input text.

Returns

The new input text.

setName

Edits the name of the component that you retrieved with getName.

Syntax

```
edkcomponent:setName(new_name)
```

Arguments

Argument	Description
`new_name`	The new name for the component.

Returns

The new component name.

setOffset

Edits the position of a match in a document.

Syntax

```
edkmatch:setOffset(new_offset)
```

Arguments

Argument	Description
`new_offset`	The new position of the match (in bytes).

Returns

The new position of the match (in bytes).

Related Topics

- getOffset, on page 320

setOffsetLength

Edits the position of a match in a document.

Syntax

```
edkmatch:setOffsetLength(new_length)
```

Arguments

Argument	Description
`new_length`	The new position of the match (in characters).

Returns

The new position of the match (in characters).

setOutputText

Edits the output text that you retrieved by using getOutputText.

The output text is the text **after** any normalization or modification that takes place as part of the extraction process. By contrast, setMatchedText enables you to edit the input text before any changes are made.

Syntax

```
edkmatch:setOutputText(new_text)
```

Arguments

Argument	Description
`new_text`	The new value that you want to assign to the output text for a match.

Returns

The new output text.

setScore

Edits the match score that you retrieved with getScore.

Syntax

```
edkmatch:setScore(new_score)
```

Arguments

Argument	Description
`new_score`	The new score for the match.

Returns

The new score for the match.

setText

Edits the matched text for a particular component that you retrieved with getText.

Syntax

```
edkcomponent:setText(new_text)
```

Arguments

Argument	Description
`new_text`	The new matched text for the component.

Returns

The new matched text for the specified component.

Glossary

A

ACI Server

A server component that runs on the Autonomy Content Infrastructure (ACI).

Autonomy Content Infrastructure (ACI)

A technology layer that automates operations on unstructured information for cross enterprise applications, which enables an automated and compatible business-to-business, peer-to-peer infrastructure. The ACI allows enterprise applications to understand and process content that exists in unstructured formats, such as e-mail, Web pages, office documents, and Lotus Notes.

C

chunking

The process of dividing a sentence into a sequence of non-overlapping text regions, or chunks. See also: shallow parsing.

compiled grammar

A grammar file that has been compiled from XML into ECR file format using the Eduction command-line tool edktool, so that Eduction can use it directly. See also: XML, ECR file, grammar, standard grammar, user grammar.

connector

An HPE fetching solution (for example HTTP Connector, Oracle Connector, File System Connector and so on) that allows you to retrieve information from any type of local or remote repository (for example, a database or a Web site). It imports the fetched documents into IDX or XML file format and indexes them into IDOL server from where you can retrieve them (for example by sending queries to IDOL server).

Connector Framework Server (CFS)

Connector Framework Server processes the information that is retrieved by connectors. Connector Framework Server uses KeyView to extract document content and metadata from over 1000 different file types. When the information has been processed, it is sent to an IDOL server or Distributed Index Handler (DIH).

D

database

An IDOL server data pool that stores indexed information. The administrator can set up one or more databases, and specifies how data is fed to the databases. By default IDOL server contains the databases Profile, Agent, Activated, Deactivated, News and Archive.

dictionary

An XML file that provides a vocabulary for an entity. Eduction uses the dictionary to scan a document and extract the defined entities that match the search pattern. See also: XML, entity, extraction.

DIH (Distributed Index Handler)

The Distributed Index Handler allows you to efficiently split and index extremely large quantities of data into multiple IDOL servers to create a completely scalable solution that delivers high performance and high availability. It provides a flexible way of transparently batching, routing, and categorizing the indexing of internal and external content into the IDOL server.

E

ECR file

ECR is a proprietary format for grammar files that Eduction can easily read at runtime. You can write grammar files in XML, then use the Eduction command-line tool edktool to compile them into ECR format. See also: XML, compiled grammar.

Eduction
The process of extracting entities (patterns of text) from documents.

entity
In Eduction, an entity is a word, phrase, or block of information that the Eduction component can match and extract from documents. An entity can be a specific text string, such as a name, or it can be a pattern of text such as an address or phone number. You define the pattern in a grammar, which Eduction uses to find the entities in documents.

extraction
Eduction extracts entities from documents based on the rules you have created in your dictionaries and grammars, and returns an XML list of matches, or adds the matches to the source document as new fields. See also: XML, grammar, dictionary.

F

fetch
The process of downloading documents from the repository in which they are stored (such as a local folder, Web site, database, Lotus Domino server, and so on), importing them to IDX format, and indexing them into an IDOL server.

fetch task
A group of settings that instruct a connector how to retrieve data from a repository. Connectors can run fetch tasks automatically, or in response to an action.

field
Fields define different parts of content in IDOL documents, such as the title, content, and metadata information.

G

grammar
In Eduction, a grammar is a pattern that defines an entity.

I

IDOL server
The HPE Intelligent Data Operating Layer (IDOL) server, which integrates unstructured, semi-structured and structured information from multiple repositories through an understanding of the content, delivering a real time environment in which operations across applications and content are automated, removing all the manual processes involved in getting the right information to the right people at the right time.

IDX
A structured file format that can be indexed into IDOL server. You can use a connector to import files into this format or you can manually create IDX files.

importing
After a document has been downloaded from the repository in which it is stored, it is imported to an IDX or XML file format. This process is called “importing”.

index
The IDOL server data index contains document content and field information for analysis and retrieval.

indexing
The process of storing data in IDOL server. IDOL server stores data in different field

types (such as, index, numeric and ordinary fields). It is important to store data in appropriate field types to ensure optimized performance.

K

KeyView

The IDOL component that extracts data, including text, metadata, and subfiles from over 1,000 different file types.

L

License Server

License Server enables you to license and run multiple HPE solutions. You must have an HPE License Server on a machine with a known, static, IP address.

Lua

An embedded scripting language that you can use to write custom scripts to expand certain IDOL functionality.

Luhn algorithm

A formula used to validate identification numbers, such as credit card numbers and social security numbers. The formula checks for errors by performing mathematical operations in the number to calculate a number that must agree with the final digit of the number.

M

metadata

Data that describes and gives information about other data. For example, the metadata for a text document might include information about the author of the document, the date it was written, or a short summary.

P

parsing

The process of analyzing text according to the rules of a formal grammar.

pattern

A pattern is a description of the entity you want to extract, that enables Eduction to produce a list of matches based on that pattern. A pattern can explicitly list what Eduction should look for (for example, a list of names), or can specify in general terms what a match should look like (for example, phone numbers). See also: entity, extraction, grammar.

polarity scoring

A number, usually between 0.50 and 1.50, that represents the strength of the sentiment in the matched phrase.

precision

Precision is the percentage of extracted entities that are true entities. See also: recall.

R

recall

The recall of an extraction is the percentage of matches that are actually returned, out of the total number of matches that should return in theory. See also: precision.

regular expressions

A string that allows you to define a particular string pattern in a concise format. In IDOL server, matching in Eduction and Connectors uses regular expressions to define what you want to match.

relevance

The similarity that a particular query result has to the initial query. IDOL server assigns results a percentage relevance score according to how closely it matches the query criteria.

S

sentiment analysis
A form of Eduction that identifies positive and negative sentiment in text.

shallow parsing
A form of sentence analysis that identifies the constituent parts of the sentence, such as noun phrases, but not their structure or their role in the sentence. See also: chunking.

standard grammar
Eduction includes a set of standard grammars that allow you to extract the most common entities, such as person, place, or company names, legal terms, addresses, dates, and times. See also: entity, compiled grammar, grammar, user grammar.

T

tagging
The process of adding extra information to documents. The tag might be a category, or entities returned from Eduction. Tagging usually adds a field to a document, which you can use to search by the name of a tag.

tokens
IDOL Server stores document text as a series of tokens. Generally, a token is a word, but it can also include other strings of characters (such as a phone number or e-mail address).

U

user grammar
XML files created by the user that describe entities that can locate patterns in text using the Eduction grammar language.

X

XML
Extensible Markup Language. XML is a language that defines the different attributes of document content in a format that can be read by humans and machines. In IDOL server, you can index documents in XML format. IDOL server also returns action responses in XML format.

Send Documentation Feedback

If you have comments about this document, you can contact the documentation team by email. If an email client is configured on this system, click the link above and an email window opens with the following information in the subject line:

Feedback on Programming Guide (Eduction SDK 11.2.0)

Add your feedback to the email and click **Send**.

If no email client is available, copy the information above to a new message in a web mail client, and send your feedback to AutonomyTPFeedback@hpe.com.

We appreciate your feedback!