The Luxembourg BabelNet Workshop

2 March 2016: Session 2

Tech session

...

Downloading and installing BabelNet
The BabelNet API

Claudio Delli Bovi

About me

Claudio Delli Bovi

dellibovi@di.uniroma1.it

bn:17381128n

Second-year PhD student
LCL group @ Sapienza
Advisor: prof. Roberto Navigli

About ~~me~~ us

Claudio Delli Bovi

dellibovi@di.uniroma1.it

bn:17381128n

Daniele Vannella

vannella@di.uniroma1.it

bn:09353189n

Francesco Cecconi

cecconi@di.uniroma1.it

bn:17381129n

Outline

- Using BabelNet
- Babelcoins, key and limits
- How to query BabelNet programmatically:
  - HTTP API
  - Java API
- The Java API: Download and set up
- The Java API: Main classes
- Usage examples

Outline

Using BabelNet

Babelcoins, key and limits

How to query BabelNet programmatically:

- HTTP API
- Java API

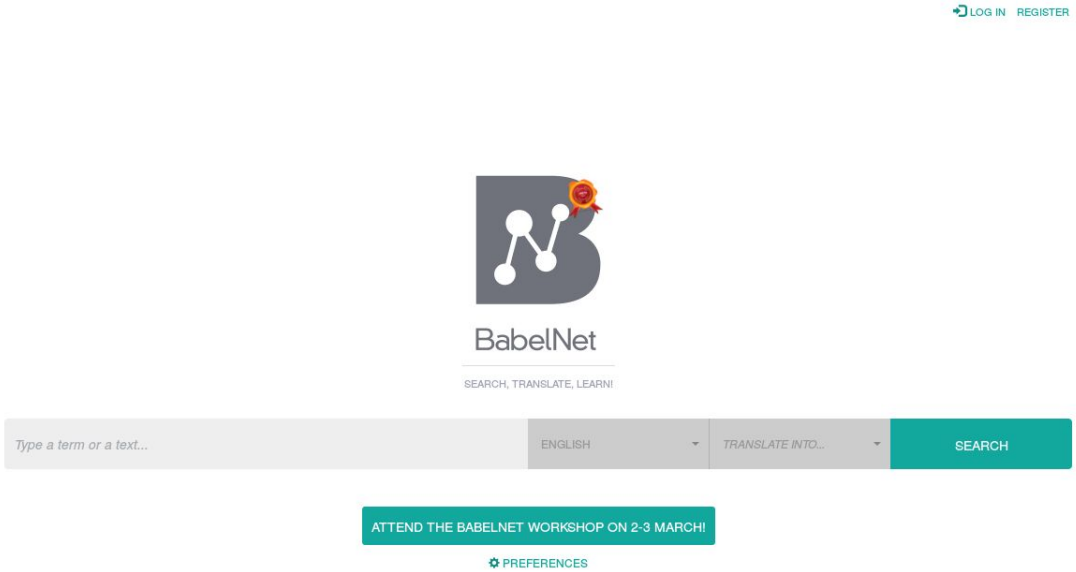
The Java API: Download and set up

The Java API: Main classes

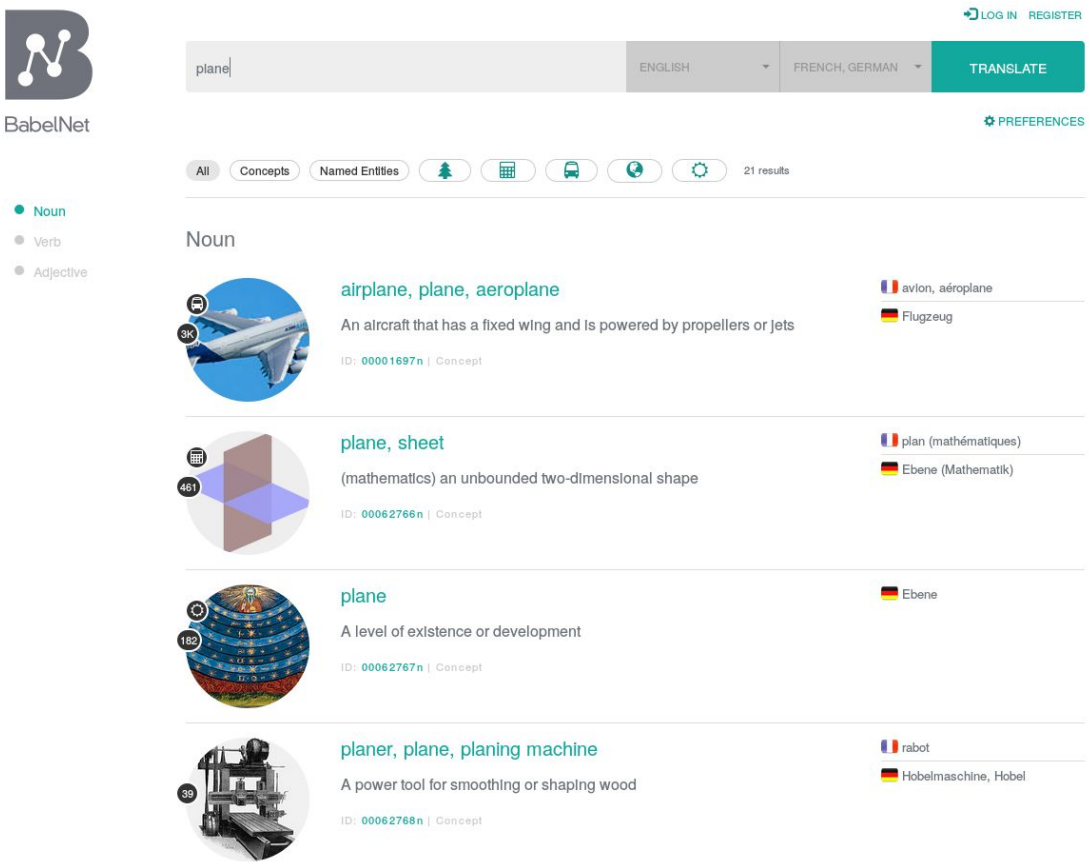
Usage examples

Technical part!

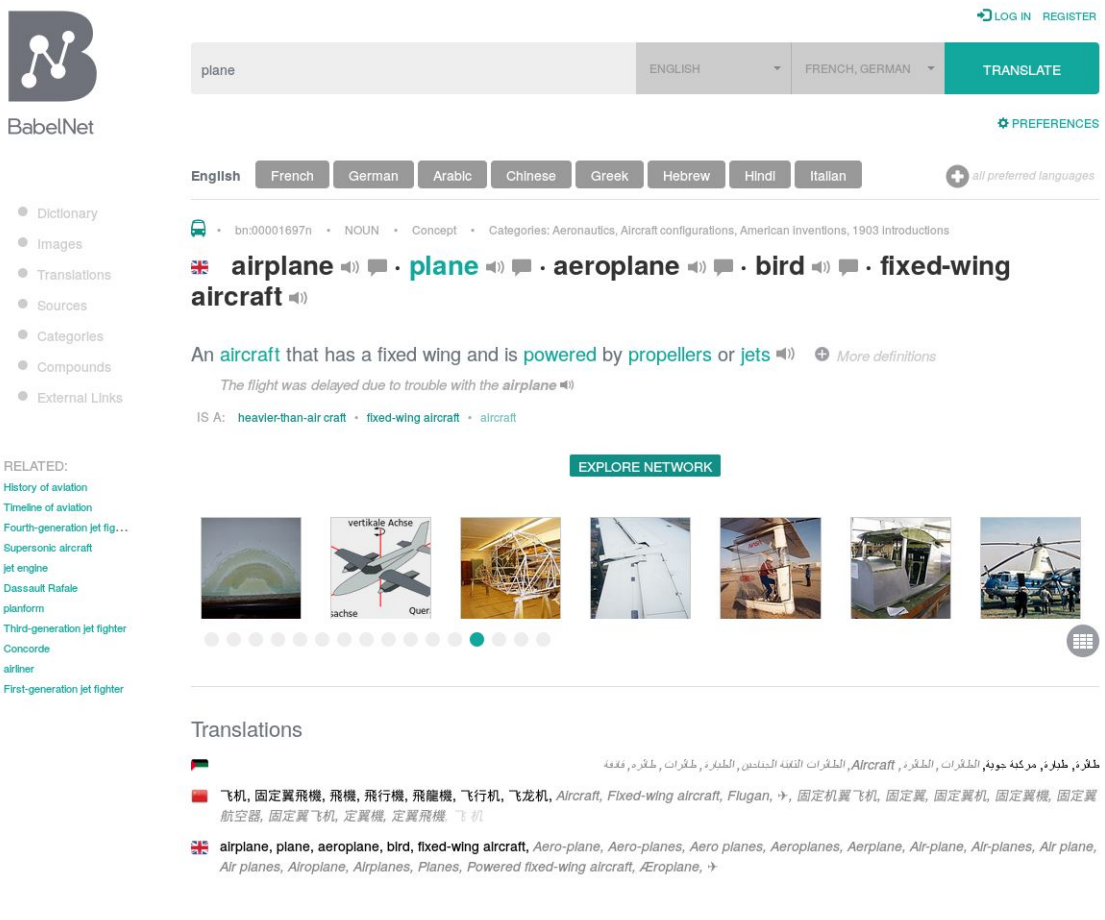
Using BabelNet



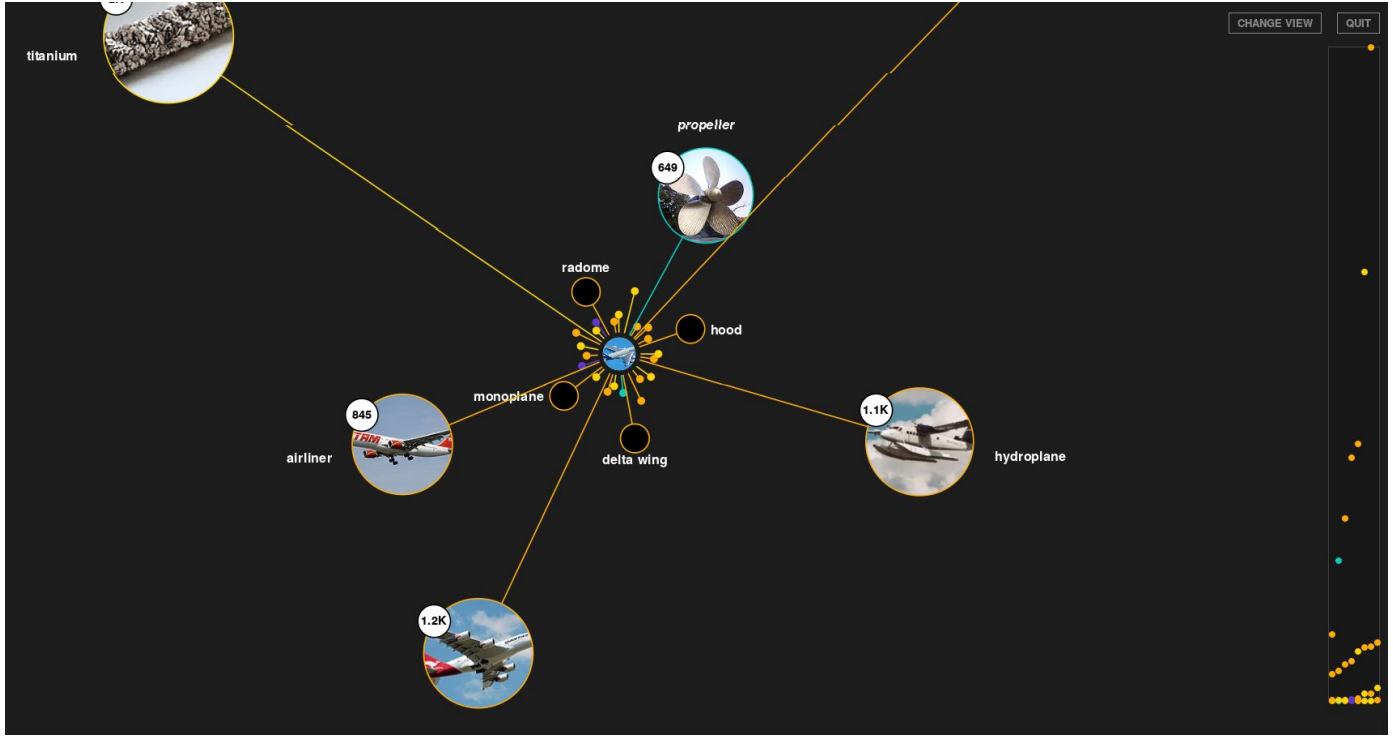
Using BabelNet



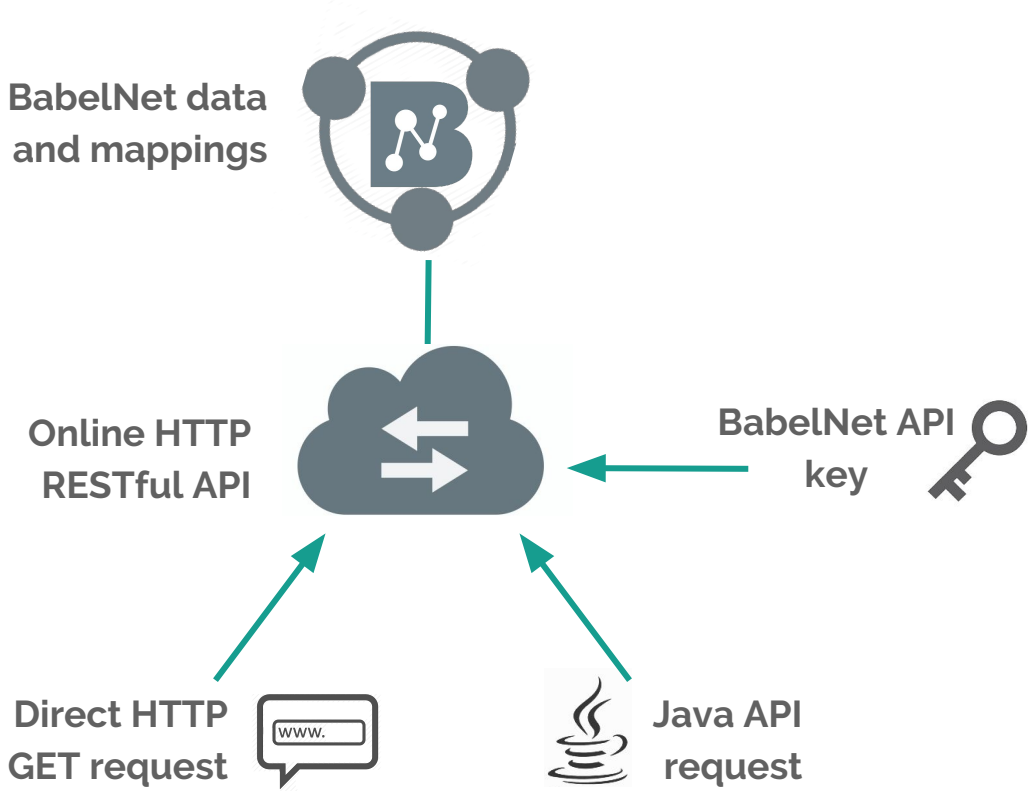
Using BabelNet



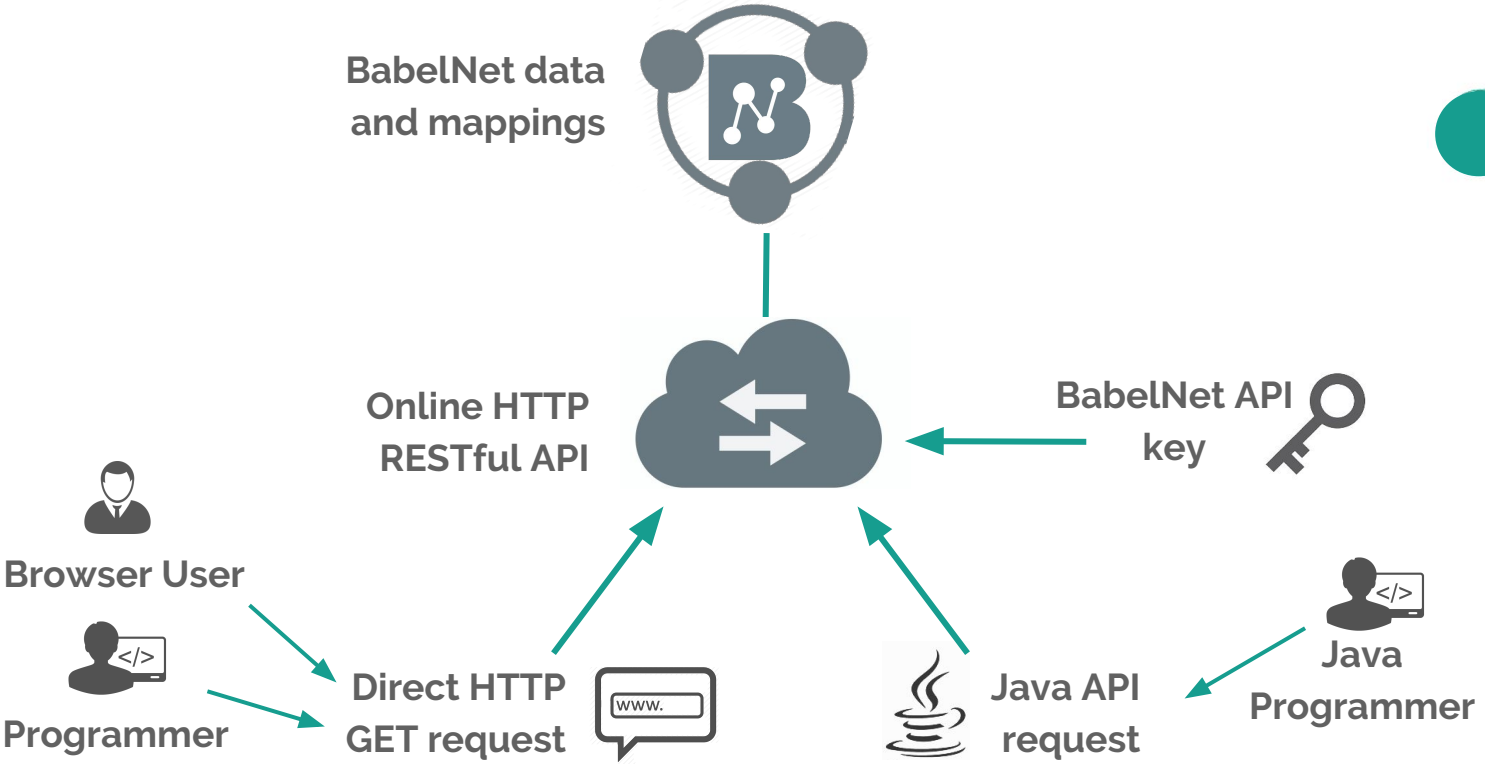
Using BabelNet



Using BabelNet… programmatically



Using BabelNet… programmatically



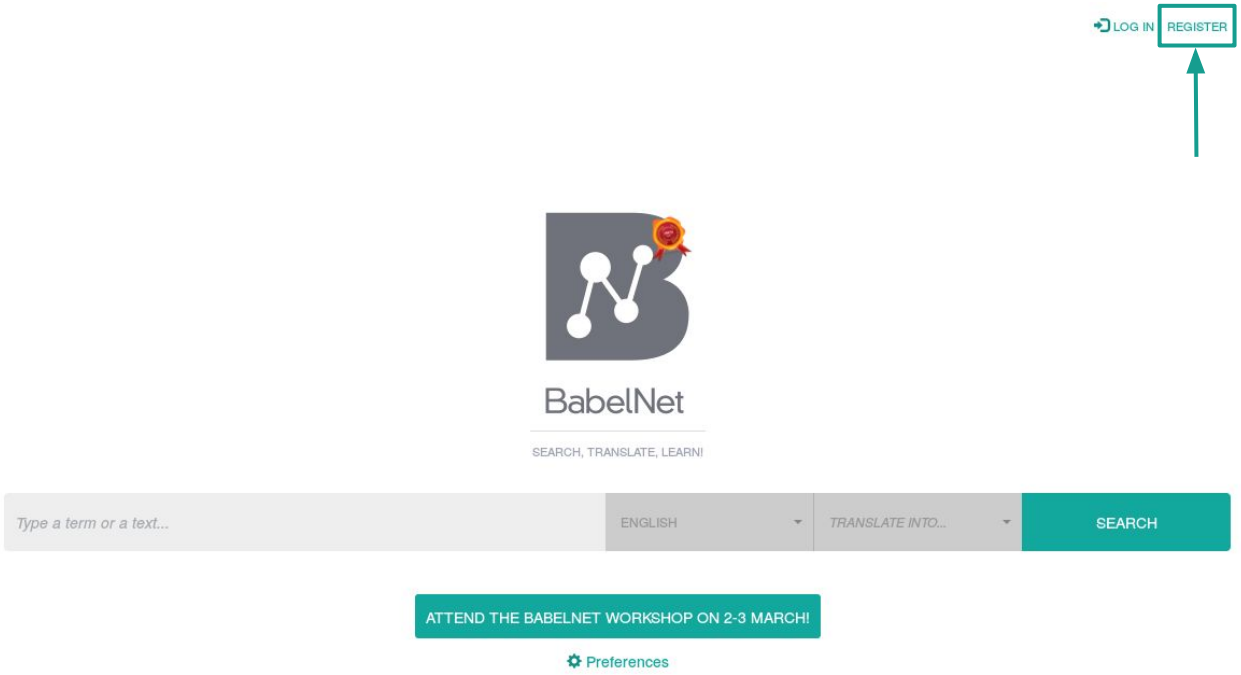
Babelcoins, key and limits

To obtain an API key you just have to register an account on BabelNet:

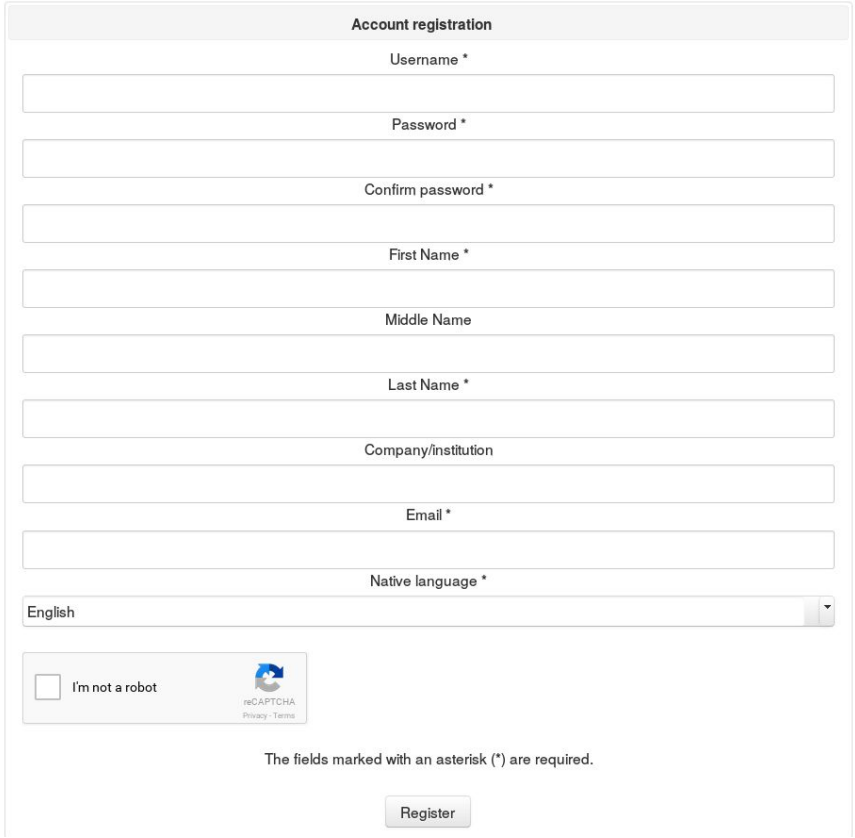
babelnet.org/register

Babelcoins, key and limits

babelnet.org/register



Babelcoins, key and limits



Babelcoins, key and limits

RESTful key information | Account details | Last searches

Key	Key limit (Requests/Day - per Service)	Remaining Babelcoins	Services
	1000	1000	BabelNet

Key information

Are you interested in increasing the RESTful key limit? Read how.

BabelNet indices download

Are you interested to download the BabelNet indices? If you have the requirements listed in the guide, you can make a request completing this form.

Log out

Babelcoins, key and limits

To obtain an API key you just have to register an account on BabelNet:

babelnet.org/register

A key enables programmatic access to the BabelNet RESTful service (including Babelfy -more on this after the lunch break!) using both the Java API and the HTTP API.

Babelcoins, key and limits

RESTful key information | Account details | Last searches

Key	Key limit (Requests/Day - per Service)	Remaining Babelcoins	Services
	1000	1000	BabelNet

What is a Babelcoin?

Key information

Are you interested in increasing the RESTful key limit? Read how.

BabelNet indices download

Are you interested to download the BabelNet indices? If you have the requirements listed in the guide, you can make a request completing this form.

Log out

Babelcoins, key and limits

To obtain an API key you just have to register an account on BabelNet:

babelnet.org/register

A key enables programmatic access to the BabelNet RESTful service (including Babelfy -more on this after the lunch break!) using both the Java API and the HTTP API.

What is a Babelcoin?

Babelcoins are used as an internal credit system to keep track of the requests made against the API.

1 Babelcoin = **1** query to BabelNet/Babelfy

Base account: **1000** Babelcoins per day

Working offline with BabelNet

If you are a researcher affiliated with a **research institution** and you need to use BabelNet for your **non-commercial research project**, you can make a request for downloading the indices using the form in your private area of the website:

babelnet.org/login

Working offline with BabelNet

RESTful key information | Account details | Last searches

Key	Key limit (Requests/Day - per Service)	Remaining Babelcoins	Services
	1000	1000	BabelNet

Key information

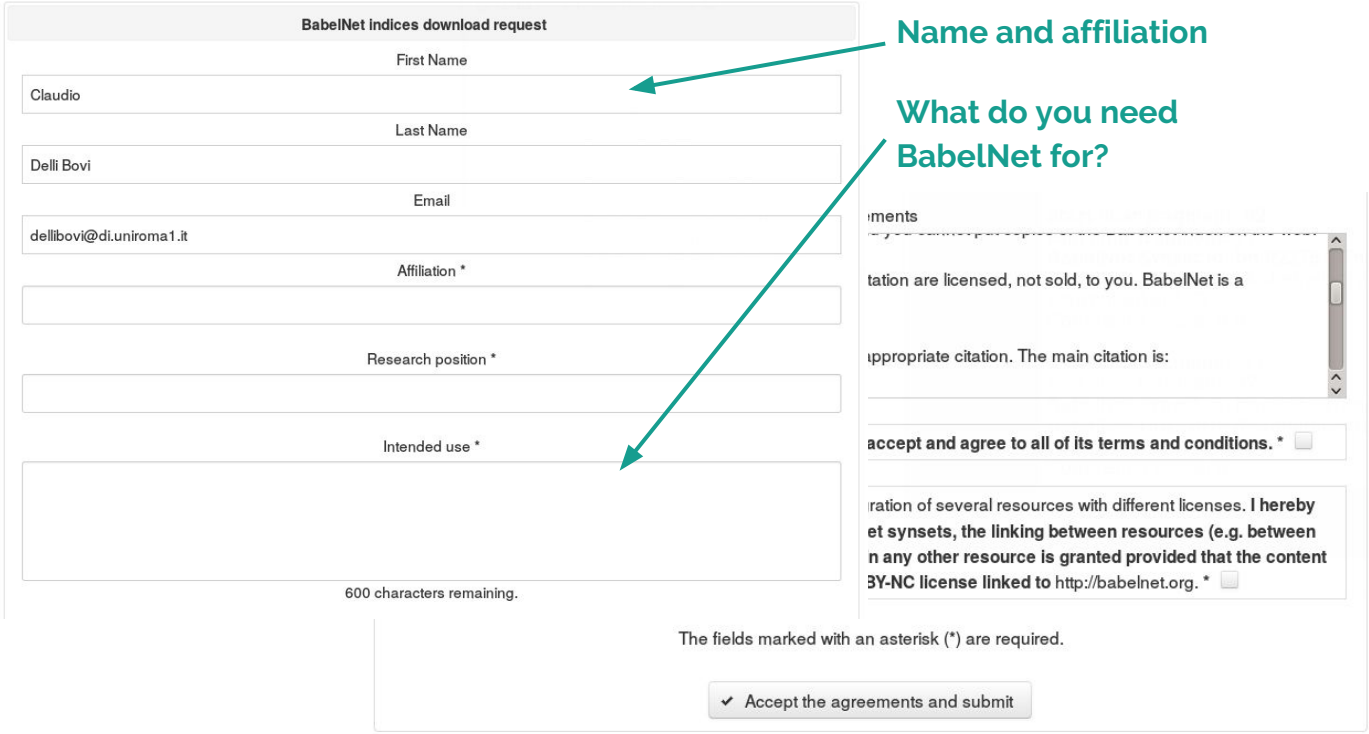
Are you interested in increasing the RESTful key limit? Read how.

BabelNet indices download

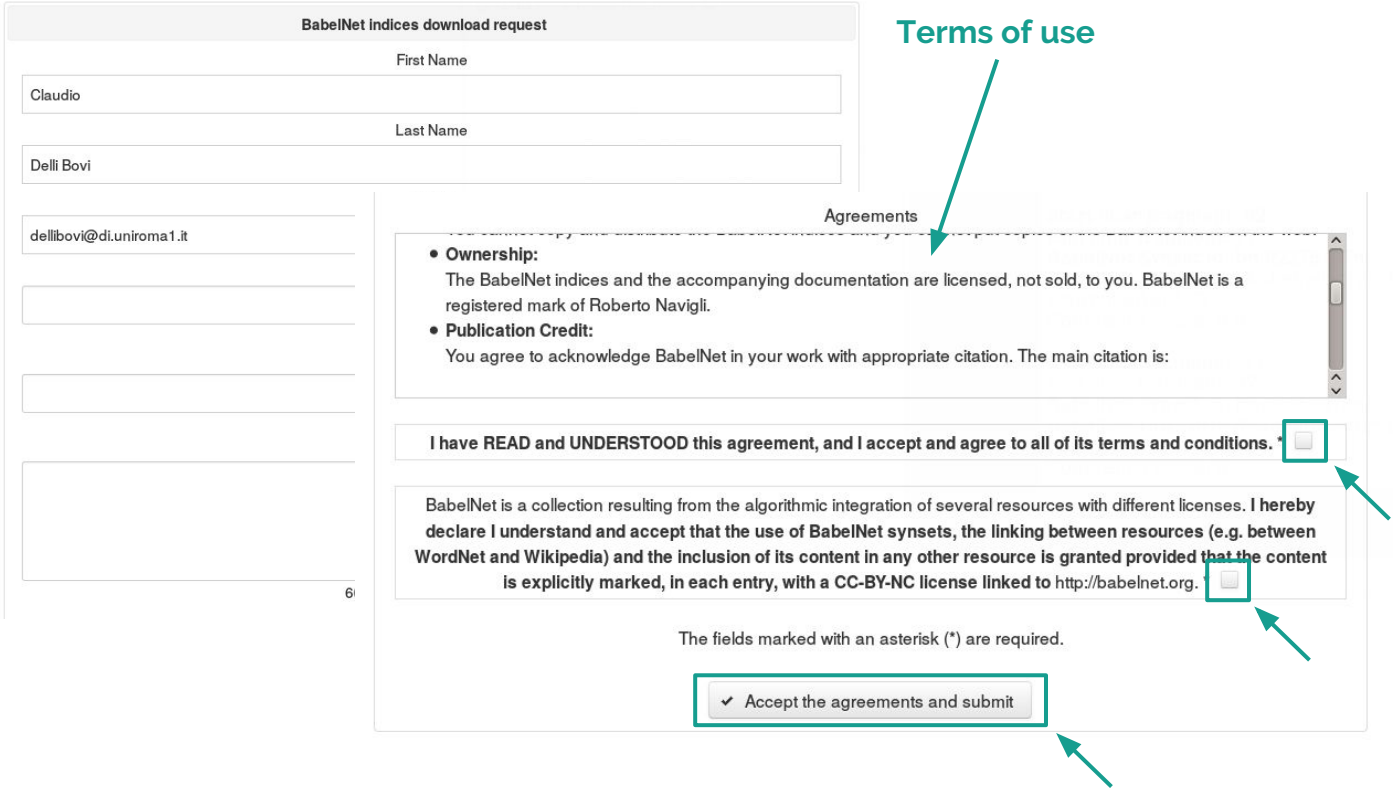
Are you interested to download the BabelNet indices? If you have the requirements listed in the guide, you can make a request completing this form.

Log out

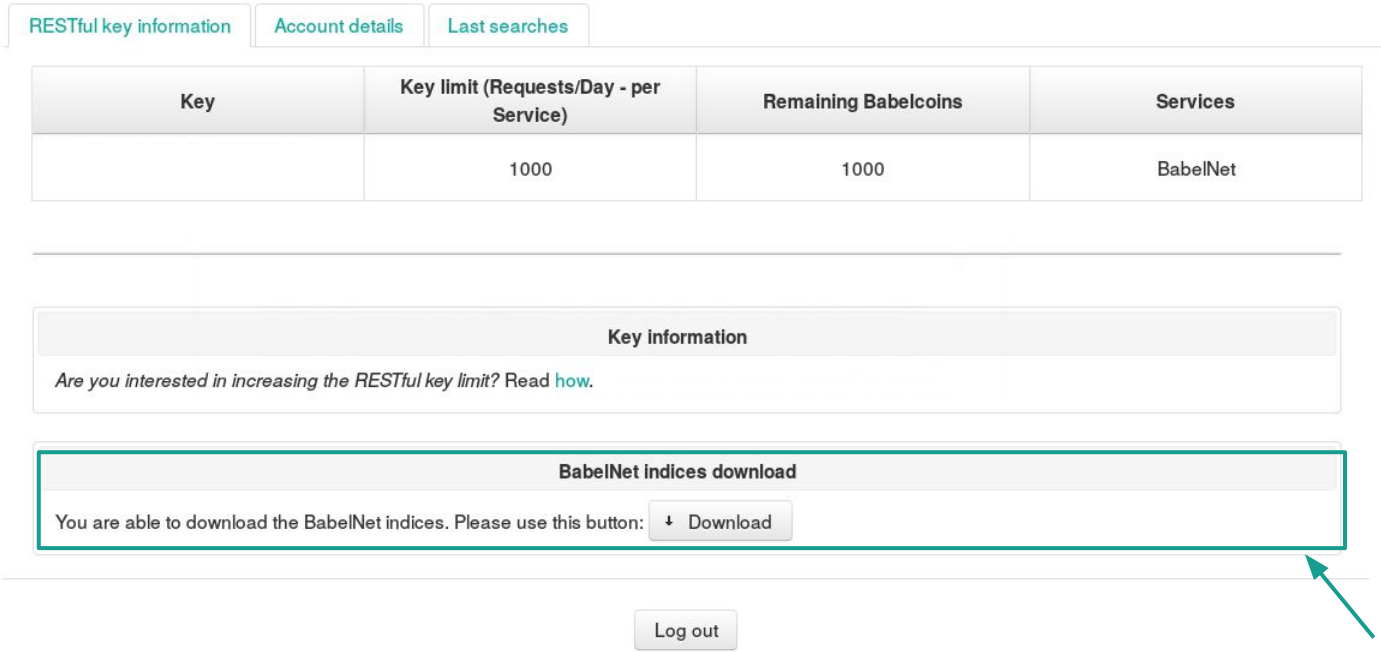
Working offline with BabelNet



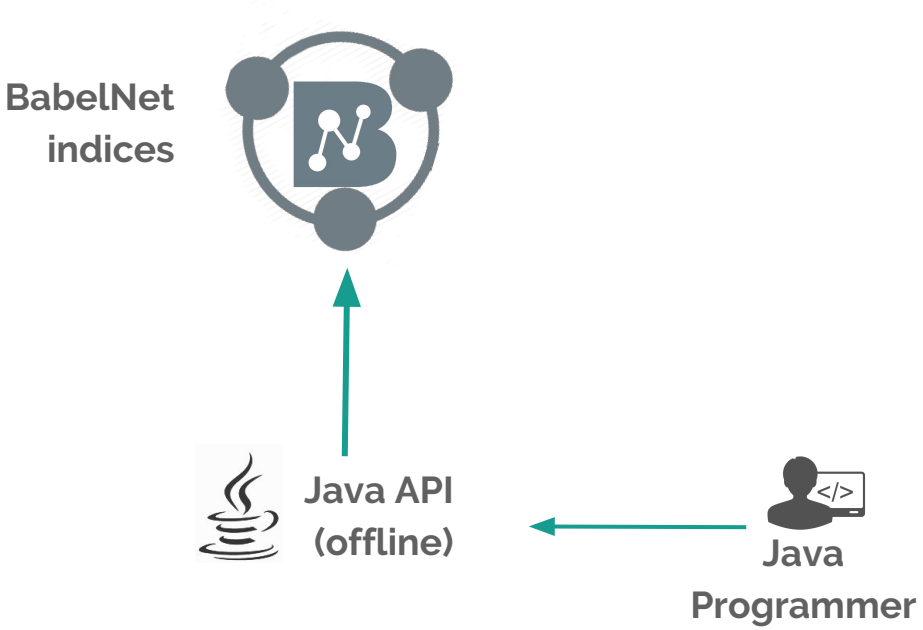
Working offline with BabelNet



Working offline with BabelNet



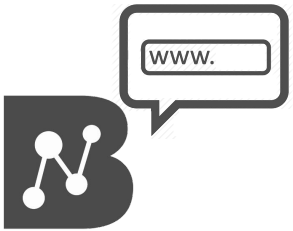
Using BabelNet… programmatically (with indices)



The HTTP API

BabelNet can be queried programmatically via an HTTP RESTful interface that returns JSON. The HTTP service uses the registration key you obtain after registering.

You just have to append a **`key`** parameter to the HTTP request.



The HTTP API

BabelNet can be queried programmatically via an HTTP RESTful interface that returns JSON. The HTTP service uses the registration key you obtain after registering.

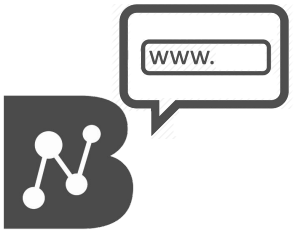
You just have to append a **`key`** parameter to the HTTP request.
For instance:

Retrieve current BabelNet version

```
https://babelnet.io/v3/getVersion?key={...}
```

Response:

```
{
  "version" : "V3_6"
}
```

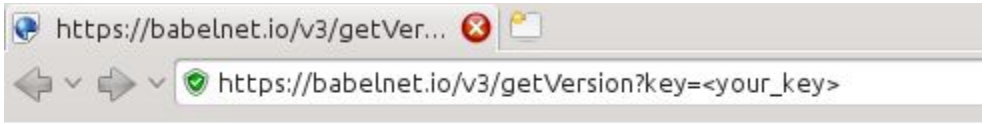


The HTTP API

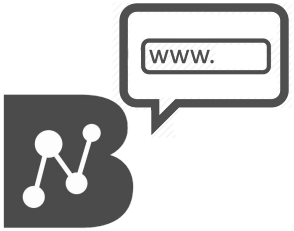
BabelNet can be queried programmatically via an HTTP RESTful interface that returns JSON. The HTTP service uses the registration key you obtain after registering.

You just have to append a **`key`** parameter to the HTTP request.
For instance:

Retrieve current BabelNet version




```
{
  "version": "V3_6"
}
```



The Java API

The BabelNet Java API is nothing more than a **Java binding** to the online HTTP RESTful service.

Once configured with a valid BabelNet key, the API provides **classes**, **types** and **methods** to query BabelNet and work with BabelNet data (senses, synsets, translations, etc.) from inside a Java program.

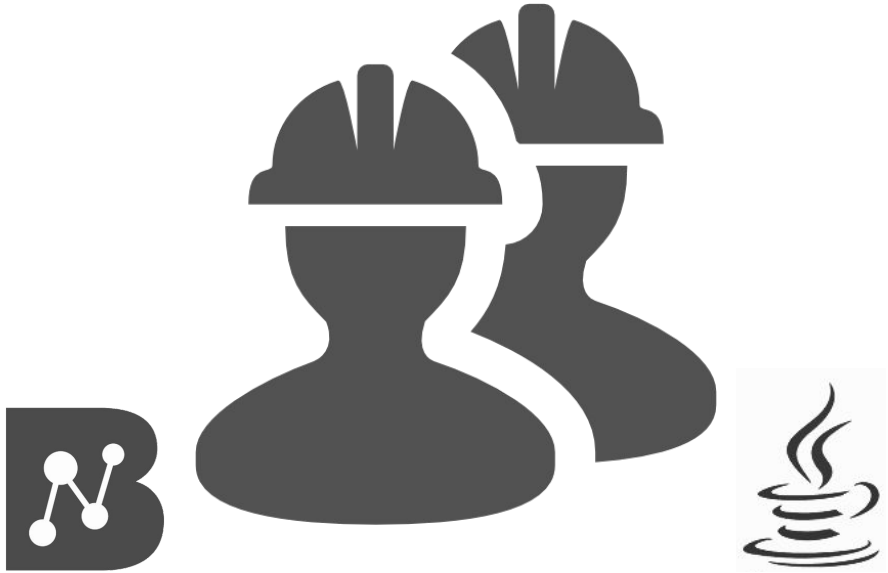
Only requirement:

Standard installation of **Java JDK** (version ≥ 1.7)

Detailed Javadoc:

babelnet.org/javadoc

Technical part ahead!



Downloading and installing instructions

Downloading and installing instructions

babelnet.org/download

BabelNet 3.6 API *(for programmatic access to BabelNet)*

- BabelNet Java API version 3.6 (February 2016 - Size: 36M) MD5
  The BabelNet Java API is a Java binding to our online HTTP RESTful service. It provides classes, types and methods to work with BabelNet data.
  If you would rather use the raw HTTP API, please read the HTTP guide.

RDF access for the Linguistic Linked Open Data cloud *(for Semantic Web fans)*

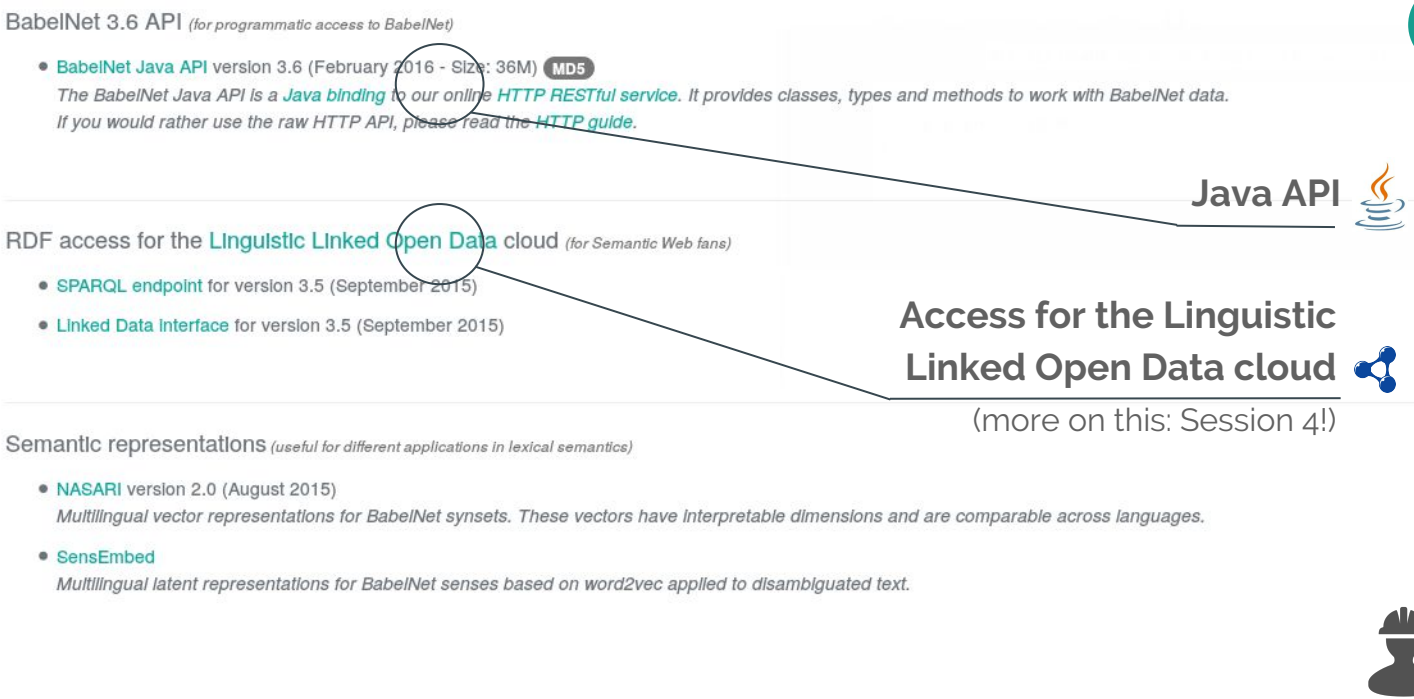
- SPARQL endpoint for version 3.5 (September 2015)
- Linked Data interface for version 3.5 (September 2015)

Semantic representations *(useful for different applications in lexical semantics)*

- NASARI version 2.0 (August 2015)
  Multilingual vector representations for BabelNet synsets. These vectors have interpretable dimensions and are comparable across languages.
- SensEmbed
  Multilingual latent representations for BabelNet senses based on word2vec applied to disambiguated text.

Downloading and installing instructions

babelnet.org/download



babelnet.org/download

Download and **unpack** the package: **`BabelNet-API-3.6.zip`**

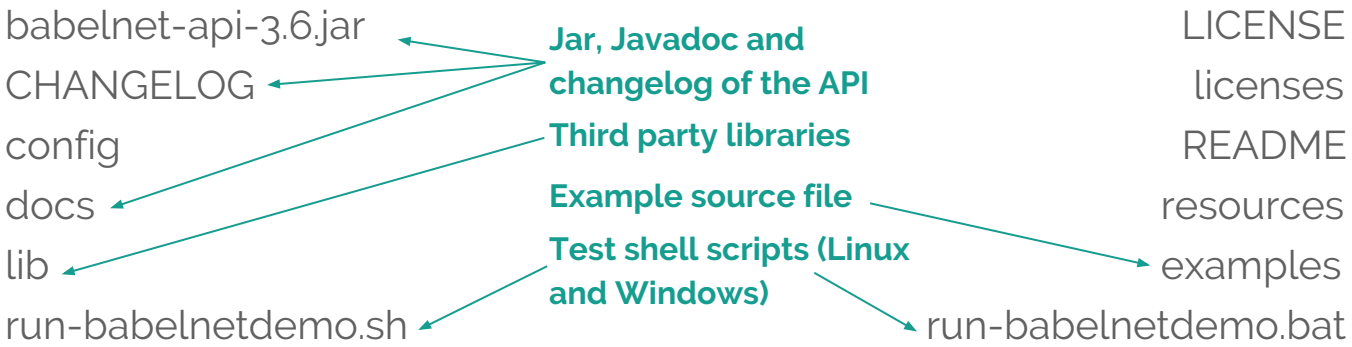
You will find the following:

babelnet-api-3.6.jar
CHANGELOG
config
docs
lib
run-babelnetdemo.sh
LICENSE
licenses
README
resources
examples
run-babelnetdemo.bat

babelnet.org/download

Download and **unpack** the package: **`BabelNet-API-3.6.zip`**

You will find the following:



babelnet.org/download

Download and **unpack** the package: **`BabelNet-API-3.6.zip`**

You will find the following:

babelnet-api-3.6.jar
CHANGELOG
config
docs
lib
run-babelnetdemo.sh

Licenses of the API and third party libraries → LICENSE, licenses

configuration files → config

README file → README

resource files → resources

examples

run-babelnetdemo.bat

Downloading and installing instructions

Two easy steps to set up and test the API:

Downloading and installing instructions
(Online API)

Two easy steps to set up and test the API:

1. Specify a **valid key** in the "`babelnet.key`" property inside the configuration file `config/babelnet.var.properties`
2. Test the API with the corresponding shell script:

   `run-babelnetdemo.sh` Linux

   `run-babelnetdemo.bat` Windows

   or with the Java example class:

```
java -classpath   lib/*:babelnet-api-3.6.jar:config
it.uniroma1.lcl.babelnet.demo.BabelNetDemo
```

Downloading and installing instructions
(Offline API: indices)

Two easy steps to set up and test the API:

1. Specify the **local path to the indices** (`<your_home>/BabelNet-3.6`) in the "`babelnet.dir`" property inside the same file `babelnet.var.properties`

2. Test the API with the corresponding shell script:

   `run-babelnetdemo.sh` Linux

   `run-babelnetdemo.bat` Windows

   or with the Java example class:

```
java -classpath   lib/*:babelnet-api-3.6.jar:config
it.uniroma1.lcl.babelnet.demo.BabelNetDemo
```

Configuring the API on Eclipse/Netbeans

Assuming you have your Java (or Scala) project in the workspace of your favourite IDE under **`projectDir/`**:

1. Copy (or link) the **`config/`** and **`resources/`** directories from the API folder into **`projectDir/`**;

Configuring the API on Eclipse/Netbeans

Assuming you have your Java (or Scala) project in the workspace of your favourite IDE under **`projectDir/`**:

1. Copy (or link) the **`config/`** and **`resources/`** directories from the API folder into **`projectDir/`**;
2. Include the third-party libraries (**`lib/*.jar`**) and the API itself (**`babelnet-api-3.6.jar`**) in the project build classpath;

Configuring the API on Eclipse/Netbeans

Assuming you have your Java (or Scala) project in the workspace of your favourite IDE under **`projectDir/`**:

1. Copy (or link) the **`config/`** and **`resources/`** directories from the API folder into **`projectDir/`**;
2. Include the third-party libraries (**`lib/*.jar`**) and the API itself (**`babelnet-api-3.6.jar`**) in the project build classpath;

Find the project in the package explorer view → Project → Properties → Java build path → Libraries → Add external JARs

Find the project in the left tree view → Properties → Categories → Libraries → compile → Add JAR/Folder

Configuring the API on Eclipse/Netbeans

Assuming you have your Java (or Scala) project in the workspace of your favourite IDE under **`projectDir/`**:

1. Copy (or link) the **`config/`** and **`resources/`** directories from the API folder into **`projectDir/`**;
2. Include the third-party libraries (**`lib/*.jar`**) and the API itself (**`babelnet-api-3.6.jar`**) in the project build classpath;
3. Include the **`config/`** directory in the project build classpath;

Configuring the API on Eclipse/Netbeans

Assuming you have your Java (or Scala) project in the workspace of your favourite IDE under **`projectDir/`**:

1. Copy (or link) the **`config/`** and **`resources/`** directories from the API folder into **`projectDir/`**;
2. Include the third-party libraries (**`lib/*.jar`**) and the API itself (**`babelnet-api-3.6.jar`**) in the project build classpath;
3. Include the **`config/`** directory in the project build classpath;

Find the project in the package explorer view → Project → Properties → Java build path → Source → Add Folder

Find the project in the left tree view → Properties → Categories → Libraries → compile → Add JAR/Folder (same as before)

The Java API: main classes

The Java API: main classes

`BabelNet`

The `BabelNet` class is used as the singleton entry point to access all the content available in BabelNet.

The Java API: main classes

`BabelNet`

The `BabelNet` class is used as the singleton entry point to access all the content available in BabelNet.

`BabelSynset`

A `BabelSynset` is a set of multilingual lexicalizations (`BabelSense`s) that are synonymous expressions of a given concept or named entity. Each `BabelSynset` has its unique ID.

The Java API: main classes

`BabelNet`

The `BabelNet` class is used as the singleton entry point to access all the content available in BabelNet.

`BabelSynset`

A `BabelSynset` is a set of multilingual lexicalizations (`BabelSense`s) that are synonymous expressions of a given concept or named entity. Each `BabelSynset` has its unique ID.

`BabelSense`

A `BabelSense` is a particular, language-specific lexicalization occurring in a given `BabelSynset`. Each `BabelSense` is tied to a particular source (WordNet, Wikipedia, Wiktionary, automatic translations, etc.).

The Java API: BabelNet

The `BabelNet` class is used as the singleton entry point to access all the content available in BabelNet. You can obtain a reference to it in your code with the following line:

```
BabelNet bn = BabelNet.getInstance();
```

The Java API: BabelNet

The `BabelNet` class is used as the singleton entry point to access all the content available in BabelNet. You can obtain a reference to it in your code with the following line:

```
BabelNet bn = BabelNet.getInstance();
```

Most of the times, you will be using the `BabelNet` class to obtain a list of `BabelSynset`s (or `BabelSense`s) given a certain lemma and language(s):

```
List<BabelSynset> synsets = bn.getSynsets(String, Language);

List<BabelSense> senses = bn.getSenses(String, Language);
```

The Java API: BabelNet

You can also specify additional constraints in your query, like the part of speech (**BabelPOS**) or the sense source (**BabelSenseSource**), using the many overloads of `getSynsets` and `getSenses`:

```
List<BabelSynset> synsets = bn.getSynsets(String,
                Language,
                BabelPOS,
                BabelSenseSource... );

List<BabelSense> senses = bn.getSenses(String,
                Language,
                BabelPOS,
                BabelSenseSource... );
```

The Java API: BabelNet

An example from the API guide:

```
// Given a word in a certain language,
// returns the concepts (`BabelSynsets`) denoted by the word.
List<BabelSynset> byl = bn.getSynsets("car", Language.EN);

// Given a word in a certain language and pos (part of speech),
// returns the concepts denoted by the word.
List<BabelSynset> byl = bn.getSynsets("run", Language.EN, BabelPOS.VERB);

// Given a word in a certain language, returns the concepts
// for the word available in the given sense sources.
List<BabelSynset> byl = bn.getSynsets("run", Language.EN, BabelPOS.NOUN,
                BabelSenseSource.WIKI, BabelSenseSource.OMWIKI);
```

Same story for `BabelNet#getSenses`

(see the example on **babelnet.org/guide#BabelSense**)

The Java API: BabelSynset

Each `BabelSynset` has an **ID** that **univocally** identifies the synset. You can obtain the ID of a `BabelSynset` via the **`BabelSynset#getId`** method.

The Java API: BabelSynset

Each `BabelSynset` has an **ID** that **univocally** identifies the synset. You can obtain the ID of a `BabelSynset` via the **`BabelSynset#getId`** method.

Of course, you can go the other way round:
if you have a specific ID (as a String object) you can easily retrieve the corresponding synset using again the `BabelNet` class:

```
// Gets a BabelSynset from a concept identifier (Babel synset ID).
BabelSynset by = bn.getSynset(new BabelSynsetID("bn:03083790n"));
```

The Java API: BabelSynset

The API contains various overloads of the method **`BabelNet#getSynset`** that allow you to retrieve a specific synset from different identifiers.

Some examples from the API guide:

```
// Gets the BabelSynsets corresponding to an input WordNet offset.
BabelSynset by = bn.getSynset(new WordNetSynsetID("wn:06879521n"));

// Gets the BabelSynsets corresponding to an input Wikidata page ID.
BabelSynset by = bn.getSynset(new WikidataID("Q4837690"));

// Given a Wikipedia title, returns the BabelSynsets which contain it.
List<BabelSynset> byl = bn.getSynsets(new WikipediaID("Men in Black (film 1997)",
                                Language.IT, BabelPOS.NOUN));
```

The Java API: BabelSynset

A `BabelSynset` is a quite structured object containing various components (senses, glosses, images, categories, etc.). The public interface of the class provides convenience classes and methods to access easily all this information:

The Java API: BabelSynset

A `BabelSynset` is a quite structured object containing various components (senses, glosses, images, categories, etc.). The public interface of the class provides convenience classes and methods to access easily all this information:

- **`BabelSense`** (we have seen this already)
- **`BabelPOS`**: the synset's part of speech
- **`BabelGloss`**: a definition of the concept in a given language
- **`BabelExample`**: an example sentence of the meaning expressed by the synset
- **`BabelImage`**: an image depicting the concept
- **`BabelSynsetIDRelation`**: a semantic connection of the synset to another synset

...

The Java API: BabelSynset

```java
// Gets a BabelSynset from a concept identifier (Babel synset ID).
BabelSynset by = bn.getSynset(new BabelSynsetID("bn:03083790n"));
// Most relevant BabelSense to this BabelSynset for a given language.
BabelSense bs = by.getMainSense(Language.EN);

// Gets the part of speech of this BabelSynset.
BabelPOS pos = by.getPOS();

// Gets the senses contained in this BabelSynset.
List<BabelSense> senses = by.getSenses();

// Collects all BabelGlosses in the given source for this BabelSynset.
List<BabelGloss> glosses = by.getGlosses();

// Collects all BabelExamples for this BabelSynset.
List<BabelExample> examples = by.getExamples();

// Gets the images (BabelImages) of this BabelSynset.
List<BabelImage> images = by.getImages();

// Collects all the edges incident on this BabelSynset.
List<BabelSynsetIDRelation> edges = by.getEdges();

// Gets the BabelCategory objects of this BabelSynset.
List<BabelCategory> cats = by.getCategories();
```

The Java API: BabelSense

A **`BabelSense`** is a particular, language-specific element inside a `BabelSynset`. As such, when we have a `BabelSense` we can always go back to the synset it belongs to using the **`BabelSense#getSynset`** method:

```
BabelSense sense = ...

BabelSynset by = sense.getSynset();
```

The Java API: BabelSense

A **`BabelSense`** is a particular, language-specific element inside a `BabelSynset`. As such, when we have a `BabelSense` we can always go back to the synset it belongs to using the **`BabelSense#getSynset`** method:

```
BabelSense sense = ...

BabelSynset by = sense.getSynset();
```

Unlike a `BabelSynset`, a `BabelSense` has a specific *lexicalization*, a specific *language* and a specific *source* (`BabelSenseSource`) it can be traced back to.

The Java API: BabelSense

```java
BabelSense bs = by.getMainSense(Language.EN);

// Gets the language of this BabelSense
Language lang = bs.getLanguage();

// Gets the part-of-speech tag of this BabelSense
BabelPOS pos = bs.getPOS();

// Gets the lemma of this BabelSense
String lemma = bs.getLemma();

// Gets the simple lemma of this sense (i.e., without parentheses, etc.)
String simpleLemma = bs.getSimpleLemma();

// Gets the pronunciations of this sense
BabelSensePhonetics pronunciations = bs.getPronunciations();

// Collects all the sources of the sense; ex: Wikipedia, WordNet, etc.
BabelSenseSource source = bs.getSource();
```

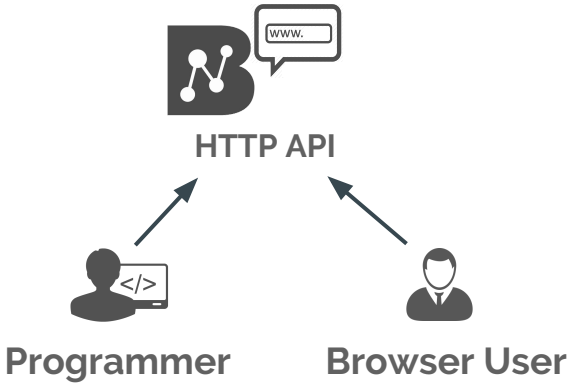
Usage examples

We will now explore some usage examples of the API and see how basic operations with BabelNet can be carried out.

Usage examples

We will now explore some usage examples of the API and see how basic operations with BabelNet can be carried out.

We will first see each example directly from the point of view of the **HTTP API**
(you can try it directly in your browser!)



Usage examples

We will now explore some usage examples of the API and see how basic operations with BabelNet can be carried out.

We will first see each example directly from the point of view of the **HTTP API**
(you can try it directly in your browser!)

HTTP API

We will then see each the corresponding Java code using the classes and methods of the **Java API**

Java API

Java Programmer

Usage examples

- Retrieving BabelSynsets of a specific word (in multiple languages) and printing their synset IDs
- Retrieving all BabelSenses of a specific BabelSynset and printing their information
- Retrieving all neighbors/hypernyms of a specific BabelSynset

Usage examples

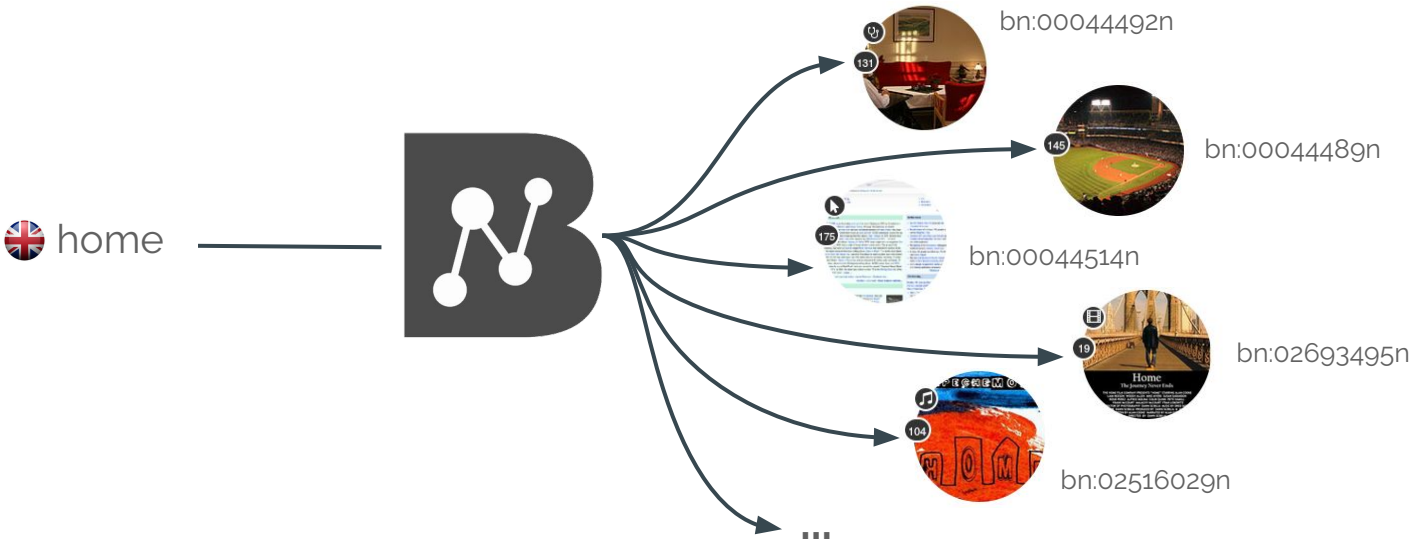
- Retrieving BabelSynsets of a specific word (in multiple languages) and printing their synset IDs

- Retrieving all BabelSenses of a specific BabelSynset and printing their information

- Retrieving all neighbors/hypernyms of a specific BabelSynset

Retrieve `BabelSynsets` of a specific word and print their synset IDs

Given the English lemma *home*, the objective is to find all possible `BabelSynset`s (i.e. semantic interpretations) associated to it.



Retrieve `BabelSynsets` of a specific word and print their synset IDs

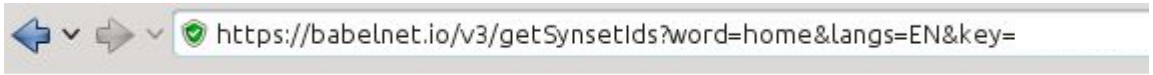
URL: `https://babelnet.io/v3/getSynsetIds?word=home & langs=EN & key=key`

- word we are interested in
- language of the word
- BabelNet API key
- Call to the RESTful service `getSynsetIDs`

HTTP API

Retrieve BabelSynsets of a specific word and print their synset IDs

URL: `https://babelnet.io/v3/getSynsetIds?word=home & langs=EN & key=key`



[{"id":"bn:10478007n","pos":"NOUN","source":"BABELNET"},{"id":"bn:14051481n","pos":"NOUN","source":"BABELNET"},

{"id":"bn:14629701n","pos":"NOUN","source":"BABELNET"}]

Browser User

HTTP API

Retrieve BabelSynsets of a specific word and print their synset IDs

URL: https://babelnet.io/v3/getSynsetIds?word=**home** & langs=**EN** & key=**key**

```
<html>
<head>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
</head>
<body>
<script>
    var service_url = 'https://babelnet.io/v2/getSynsetIds';
    var word = 'home'
    var lang = 'EN'
    var key  =

    var params = {
        'word': word,
    'langs': lang,
    'key' : key
    };

    $.getJSON(service_url + "?", params, function(response) {
        $.each(response, function(key, val) {
            $('<div>', {text:val['id']}).appendTo(document.body);
        });
    });
</script>
</body>
</html>
```

Programmer

HTTP API

Retrieve BabelSynsets of a specific word and print their synset IDs

URL: `https://babelnet.io/v3/getSynsetIds?word=home & langs=EN & key=key`

```
bn:10478007n
bn:14051481n
bn:02442174n
bn:03008481n
bn:03199617n
bn:03891547n
bn:02092291n
bn:00008792n
bn:02910619n
bn:14326182n
bn:17340893n
bn:14590626n
bn:01039750n
bn:02170958n
bn:15820329n
bn:01669387n
bn:11209713n
bn:14133760n
```

Programmer

HTTP API

Retrieve BabelSynsets of a specific word (in multiple languages) and print their synset IDs

URL:

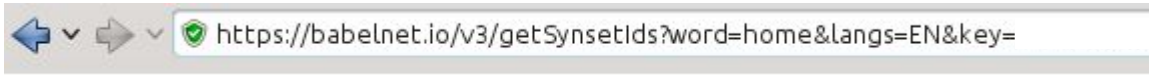
```
https://babelnet.io/v3/getSynsetIds?
 word=home & langs=EN & filterLangs=DE &
filterLangs=FR & key=key
```

language filters
(DE and FR)

HTTP API

Retrieve BabelSynsets of a specific word (in multiple languages) and print their synset IDs

URL: `https://babelnet.io/v3/getSynsetIds?word=home & langs=EN & filterLangs=DE & filterLangs=FR & key=key`



[{"id":"bn:10478007n","pos":"NOUN","source":"BABELNET"},{"id":"bn:14051481n","pos":"NOUN","source":"BABELNET"},

{"id":"bn:14629701n","pos":"NOUN","source":"BABELNET"}]

Browser User

HTTP API

Retrieve BabelSynsets of a specific word (in multiple languages) and print their synset IDs

URL: https://babelnet.io/v3/getSynsetIds?
word=**home** & langs=**EN** & filterLangs=**DE** &
filterLangs=**FR** & key=**key**

```
<html>
<head>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
</head>
<body>
<script>
    var service_url = 'https://babelnet.io/v2/getSynsetIds';
    var word = 'home'
    var lang = 'EN'
    var filterLangs = ['FR', 'DE']
    var key  =

    var params = {
        'word': word,
    'langs': lang,
    'filterLangs': filterLangs,
    'key' : key
    };

    $.getJSON(service_url + "?", params, function(response) {
        $.each(response, function(key, val) {
            $('<div>', {text:val['id']}).appendTo(document.body);
        });
    });
</script>
</body>
</html>
```

Programmer

HTTP API

Retrieve `BabelSynsets` of a specific word (in multiple languages) and print their synset IDs

All accepted parameters:

Name	Type	Description
word	string	**Required.** The word you want to search for
langs	Language	**Required.** The language of the word. Accepts multiple values. Example: https://babelnet.io/v2/getSynsetIds?word=apple&langs=EN&langs=IT&pos=NOUN&key=<your_key>
filterLangs	Language	The languages in which the data are to be retrieved. **Default** value is the **search language** and accepts not more than 3 languages except the search language. Example: https://babelnet.io/v2/getSynsetIds?word=apple&langs=EN&langs=IT&filterLangs=DE&filterLangs=IT&pos=NOUN&key=<your_key>
pos	POS	Returns only the synsets containing this part of speech (NOUN, VERB, etc). Accepts only a single value. Example: https://babelnet.io/v2/getSynsetIds?word=apple&lang=EN&pos=NOUN&key=<your_key>
source	Source	Returns only the synsets containing these sources (WIKT, WIKIDATA, etc). Accepts multiple values. Example: https://babelnet.io/v2/getSynsetIds?word=apple&lang=EN&source=WIKT&source=WIKIDATA&key=<your_key>
key	string	**Required.** API key obtained after signing up to BabelNet (see **key & limits**)

HTTP API

Retrieve BabelSynsets of a specific word and print their synset IDs

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class Example {

    public static void main(String[] args) throws IOException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSynset synset : bn.getSynsets("home", Language.EN)) {
            System.out.println("Synset ID: " + synset.getId());
        }
    }

}
```

Retrieve `BabelSynsets` of a specific word and print their synset IDs

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class Example {

    public static void main(String[] args) throws IOException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSynset synset : bn.getSynsets("home", Language.EN)) {
            System.out.println("Synset ID: " + synset.getId());
        }
    }

}
```

Reference to the `BabelNet` object

Print the ID for each `BabelSynset` retrieved

Use `BabelNet#getSynsets` to get a list of synsets for the word "*home*"

Programmer

Java API

Retrieve `BabelSynsets` of a specific word (in multiple languages) and print their synset IDs

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;
import java.util.Arrays;

public class Example {

    public static void main(String[] args) throws IOException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSynset synset : bn.getSynsets("home", Language.EN, Arrays.asList(Language.DE, Language.FR))) {
            System.out.println("Synset ID: " + synset.getId());
        }
    }
}
```

Specify here the languages in which you want the information about "*home*" to be retrieved

Default: search language (EN in this case)

Programmer

Java API

Retrieve `BabelSynsets` of a specific word (in multiple languages) and print their synset IDs

Output:

```
Synset ID: bn:00044503n
Synset ID: bn:00287069n
Synset ID: bn:01062453n
Synset ID: bn:00000356n
Synset ID: bn:00115677r
Synset ID: bn:00196994n
Synset ID: bn:02699448n
Synset ID: bn:02427132n
Synset ID: bn:00502400n
Synset ID: bn:17341158n
Synset ID: bn:01359805n
Synset ID: bn:01095781n
Synset ID: bn:03280967n
Synset ID: bn:00104253a
Synset ID: bn:16517891n
```

Usage examples

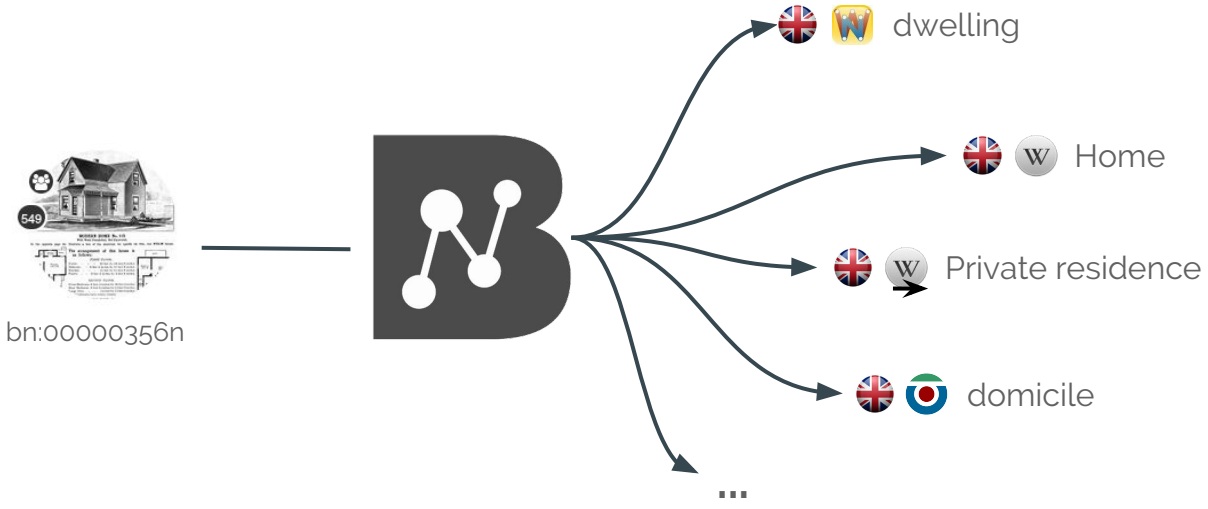
- Retrieving BabelSynsets of a specific word (in multiple languages) and printing their synset IDs

- Retrieving all BabelSenses of a specific BabelSynset and printing their information

- Retrieving all neighbors/hypernyms of a specific BabelSynset

Retrieve all `BabelSenses` of a specific `BabelSynset` and print their information

Given a specific `BabelSynset` of *home* (**bn:00000356n**), the objective is to retrieve all specific `BabelSense`s it includes.



Retrieve all BabelSenses of a specific BabelSynset and print their information

URL: `https://babelnet.io/v3/getSynset?id=bn:00000356n & key=key`

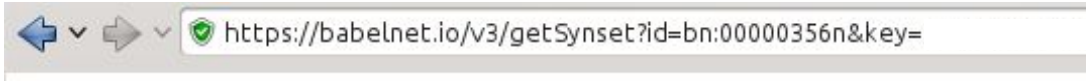
ID of synset we are interested in (BabelNet or WordNet)

BabelNet API key

HTTP API

Retrieve all BabelSenses of a specific BabelSynset and print their information

URL: https://babelnet.io/v3/getSynset?id=**bn:00000356n** & key=**key**




```
{"senses":
[{"lemma":"dwelling","simpleLemma":"dwelling","source":"WN","sensekey":"dwelling%1:06:00::","sensenumbe
r":1,"wordNetOffset":"03259505n","frequency":0,"position":1,"language":"EN","pos":"NOUN","synsetID":
{"id":"bn:00000356n","pos":"NOUN","source":"BABELNET"},"translationInfo":"","pronunciations":{"audios":
[{"lemma":"dwelling","language":"EN","filename":"en-us-dwelling.ogg"}],"transcriptions":["[/
'dwɛl.ɪŋ/]"]},"freebaseId":"025tg6m"},
```


•
•
•


```
["private_home","far_from_home","Home_construction","home_team","home_prices","home_match","traditional
_home","falling_home"]},"lnToOtherForm":{"EN":
["domestic","accommodation","residence","Residence","Housing","housing","household"]},"filterLangs":
["EN"]}
```


Browser User

HTTP API

Retrieve all BabelSenses of a specific BabelSynset and print their information

URL: `https://babelnet.io/v3/getSynset?id=bn:00000356n & key=key`

```
<html>
<head>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
</head>
<body>
<script>
    var service_url = 'https://babelnet.io/v2/getSynset';
    var id = 'bn:00000356n'
    var key =

    var params = {
        'id'  : id,
        'key' : key
    };

    $.getJSON(service_url + "?", params, function(response) {

        $.each(response['senses'], function(key, val) {
            var entry = "Sense: " + val['lemma']
                + "<br/>Language: " + val['language']
                + "<br/>Source: " + val['source'] + "<br/><br/>";

            $('<div>', {html:entry}).appendTo(document.body);
        });

    });
</script>
</body>
</html>
```

Programmer

HTTP API

Retrieve all BabelSenses of a specific BabelSynset and print their information

URL: `https://babelnet.io/v3/getSynset?id=bn:00000356n & key=key`

Sense: dwelling
Language: EN
Source: WN

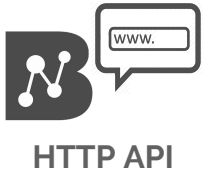
Sense: home
Language: EN
Source: WN

Sense: domicile
Language: EN
Source: WN

Sense: abode
Language: EN
Source: WN

Sense: habitation
Language: EN
Source: WN

Sense: dwelling_house
Language: EN
Source: WN



Retrieve all BabelSenses of a specific BabelSynset and print their information

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSense;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.babelnet.data.BabelAudio;
import it.uniroma1.lcl.babelnet.data.BabelSensePhonetics;

import java.io.IOException;

public class ExampleSense {

    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSense sense : bn.getSynset(new BabelSynsetID("bn:00000356n"))) {
            System.out.println("Sense: " + sense.getLemma()
                               + "\tLanguage: " + sense.getLanguage().toString()
                               + "\tSource: " + sense.getSource().toString());
            BabelSensePhonetics phonetic = sense.getPronunciations();
            for (BabelAudio audio : phonetic.getAudioItems()) {
                System.out.println("Audio URL " + audio.getValidatedUrl());
            }
        }
    }
}
```

Programmer

Java API

Retrieve all `BabelSenses` of a specific `BabelSynset` and print their information

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSense;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.babelnet.data.BabelAudio;
import it.uniroma1.lcl.babelnet.data.BabelSensePhonetics;

import java.io.IOException;

public class ExampleSense {

    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSense sense : bn.getSynset(new BabelSynsetID("bn:00000356n"))) {
            System.out.println("Sense: " + sense.getLemma()
                               + "\tLanguage: " + sense.getLanguage().toString()
                               + "\tSource: " + sense.getSource().toString());
            BabelSensePhonetics phonetic = sense.getPronunciations();
            for (BabelAudio audio : phonetic.getAudioItems()) {
                System.out.println("Audio URL " + audio.getValidatedUrl());
            }
        }
    }
}
```

Reference to the `BabelNet` object

Call to `BabelNet#getSynset` with a given synset ID

Print information for each `BabelSense` retrieved

Programmer

Java API

Retrieve all `BabelSenses` of a specific `BabelSynset` and print their information

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSense;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.babelnet.data.BabelAudio;
import it.uniroma1.lcl.babelnet.data.BabelSensePhonetics;
import it.uniroma1.lcl.babelnet.resources.WikidataID;

import java.io.IOException;

public class ExampleSense {

    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        for (BabelSense sense : bn.getSynset(new WikidataID("Q699405"))) {
            System.out.println("Sense: " + sense.getLemma()
                               + "\tLanguage: " + sense.getLanguage().toString()
                               + "\tSource: " + sense.getSource().toString());
            BabelSensePhonetics phonetic = sense.getPronunciations();
            for (BabelAudio audio : phonetic.getAudioItems()) {
                System.out.println("Audio URL " + audio.getValidatedUrl());
            }
        }
    }
}
```

`BabelNet#getSynset` can also be used with other resource ID (e.g. Wikidata)

Programmer

Java API

Retrieve all BabelSenses of a specific BabelSynset and print their information

Output:

```
Sense: dwelling Language: EN    Source: WN
Audio URL //upload.wikimedia.org/wikipedia/commons/3/30/En-us-dwelling.ogg
Sense: home     Language: EN    Source: WN
Audio URL //upload.wikimedia.org/wikipedia/commons/3/37/En-us-home.ogg
Sense: home     Language: EN    Source: WN
Audio URL //upload.wikimedia.org/wikipedia/commons/3/37/En-us-home.ogg
Sense: domicile Language: EN    Source: WN
Sense: abode    Language: EN    Source: WN
Audio URL //upload.wikimedia.org/wikipedia/commons/f/f6/En-us-abode.ogg
Sense: habitation       Language: EN    Source: WN
Sense: dwelling_house   Language: EN    Source: WN
Sense: place    Language: EN    Source: WN
Audio URL //upload.wikimedia.org/wikipedia/commons/a/a0/En-us-place.ogg
Sense: home     Language: EN    Source: WIKI
Audio URL //upload.wikimedia.org/wikipedia/commons/3/37/En-us-home.ogg
```

Programmer

Java API

Usage examples

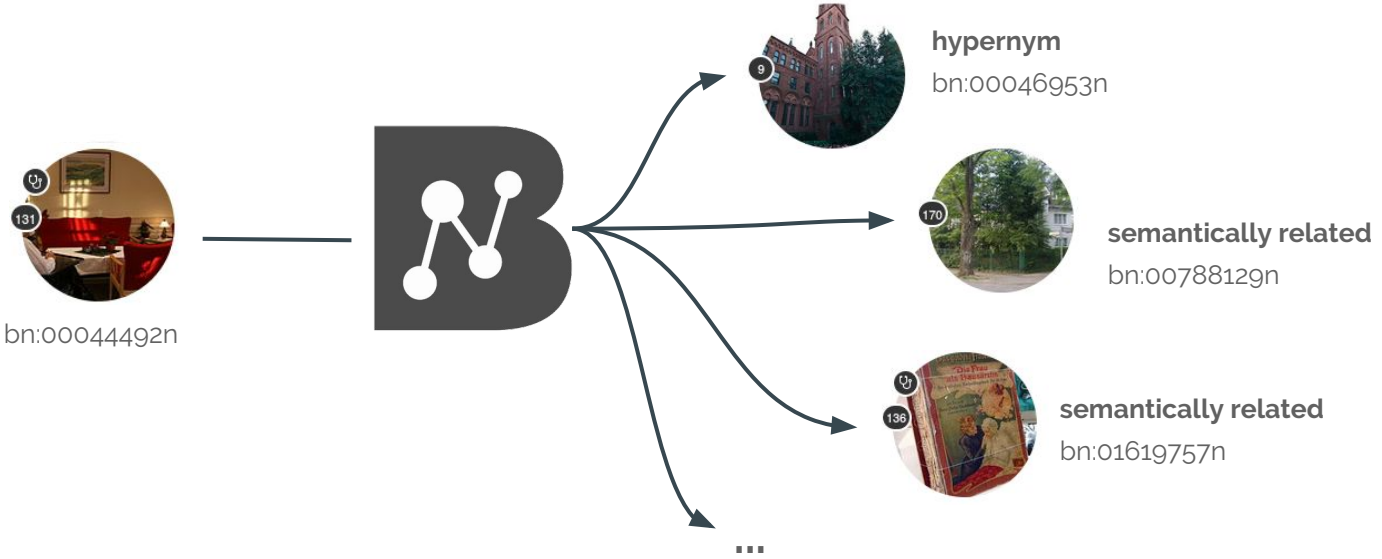
- Retrieving BabelSynsets of a specific word (in multiple languages) and printing their synset IDs

- Retrieving all BabelSenses of a specific BabelSynset and printing their information

- Retrieving all neighbors/hypernyms of a specific BabelSynset

Retrieve all neighbors of a specific `BabelSynset`

Given a specific `BabelSynset` of *home* (**bn:00044492n**), the objective is to retrieve all neighboring `BabelSynsets` in the semantic network.



Retrieve all neighbors of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00044492n & key=key`

ID of synset we are interested in

BabelNet API key

HTTP API

Retrieve all neighbors of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00044492n & key=key`


```
[{"language":"EN","pointer":{"fSymbol":"gdis","name":"Gloss related form (disambiguated)","shortName":"gloss-
related","relationGroup":"OTHER","isAutomatic":false},"target":"bn:00084525v","weight":0.12766,"normalizedWeight":0
.08317},{"language":"EN","pointer":{"fSymbol":"@","name":"Hypernym","shortName":"is-
a","relationGroup":"HYPERNYM","isAutomatic":false},"target":"bn:00046953n","weight":0.5,"normalizedWeight":0.32576}
,{"language":"EN","pointer":{"fSymbol":"+","name":"Derivationally related
form","shortName":"deriv","relationGroup":"OTHER","isAutomatic":false},"target":"bn:00089431v","weight":0.39024,"no
rmalizedWeight":0.25425},
```


•
•
•


```
{"language":"SV","pointer":{"fSymbol":"r","name":"Semantically related
form","shortName":"related","relationGroup":"OTHER","isAutomatic":false},"target":"bn:00027546n","weight":0.0,"norm
alizedWeight":0.0},{"language":"EN","pointer":{"fSymbol":"@w","name":"Hypernym","shortName":"is-
a","relationGroup":"HYPERNYM","isAutomatic":true},"target":"bn:00058336n","weight":0.0,"normalizedWeight":0.0}]
```


Browser User

HTTP API

Retrieve all neighbors of a specific BabelSynset

URL: https://babelnet.io/v3/getEdges?id=**bn:00044492n** & key=**key**

```
<html>
<head>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
</head>
<body>
<script>
    var service_url = 'https://babelnet.io/v2/getEdges';
    var id = 'bn:00044492n'
    var key =

    var params = {
        'id': id,
        'key' : key
    };

    $.getJSON(service_url + "?", params, function(response) {

        $.each(response, function(key, val) {
            var pointer = val['pointer'];
            var entry = "Source: " + id
                + "<br/>Target: " + val['target']
                + "<br/>Edge: " + pointer['name'] + "<br/><br/>";
            $('<div>', {html:entry}).appendTo(document.body);
        });

    });
</script>
</body>
</html>
```

Retrieve all neighbors of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00044492n & key=key`

Source: bn:00044492n
Target: bn:00084525v
Edge: Gloss related form (disambiguated)

Source: bn:00044492n
Target: bn:00046953n
Edge: Hypernym

Source: bn:00044492n
Target: bn:00089431v
Edge: Derivationally related form

Source: bn:00044492n
Target: bn:00046953n
Edge: Gloss related form (disambiguated)

Source: bn:00044492n
Target: bn:00061450n
Edge: Gloss related form (disambiguated)

Source: bn:00044492n
Target: bn:00788129n
Edge: Semantically related form

Source: bn:00044492n
Target: bn:03335997n
Edge: Semantically related form

-
-
-

Retrieve all neighbors of a specific BabelSynset

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.BabelSynsetIDRelation;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class ExampleNeighbors {
    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        BabelSynset by = bn.getSynset(new BabelSynsetID("bn:00044492n"));
        for(BabelSynsetIDRelation edge : by.getEdges()) {
            System.out.println(by.getId()+" "+by.getMainSense(Language.EN).getLemma()+"\t-- "
                + edge.getPointer()+" --\t"
                + edge.getBabelSynsetIDTarget()+" "+
                    edge.getBabelSynsetIDTarget().toBabelSynset().getMainSense(Language.EN).getLemma());
        }
    }
}
```

Retrieve all neighbors of a specific `BabelSynset`

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.BabelSynsetIDRelation;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class ExampleNeighbors {
    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        BabelSynset by = bn.getSynset(new BabelSynsetID("bn:00044492n"));
        for(BabelSynsetIDRelation edge : by.getEdges()) {
            System.out.println(by.getId()+" "+by.getMainSense(Language.EN).getLemma()+"\t-- "
                + edge.getPointer()+" --\t"
                + edge.getBabelSynsetIDTarget()+" "+
                    edge.getBabelSynsetIDTarget().toBabelSynset().getMainSense(Language.EN).getLemma());
        }
    }
}
```

From a `BabelSynset`, retrieve all the edges with `BabelSynset#getEdges`

`BabelSynsetIDRelation#getPointer` encodes information about the type of edge (e.g. hypernym, semantically related from)

Print ID and main sense of the connected `BabelSynset` (edge target)

Retrieve all neighbors of a specific BabelSynset

Output:

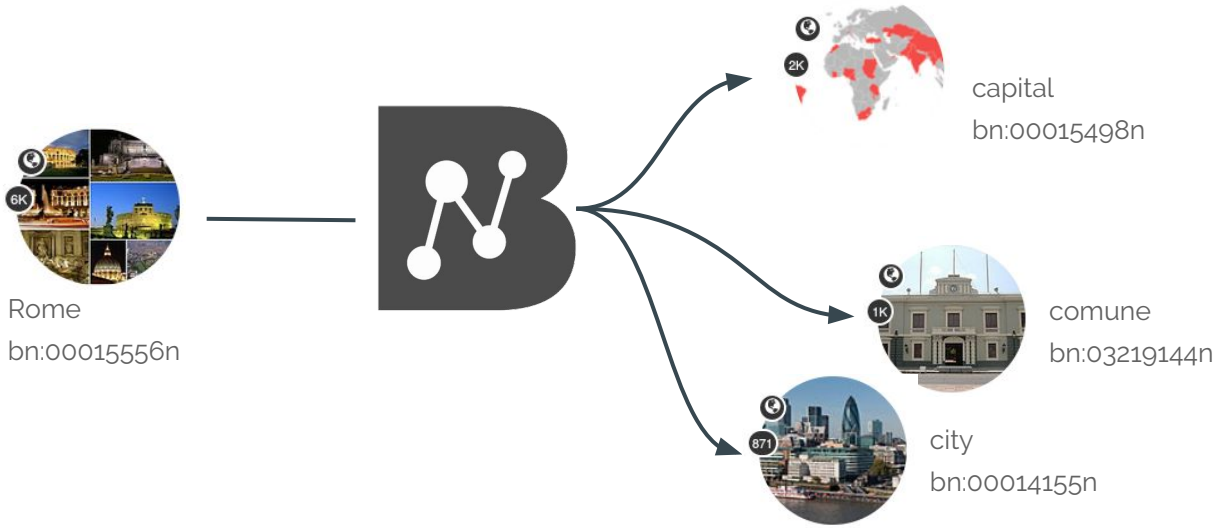
```
bn:00044492n home       -- gloss_related_form_(disambiguated) --        bn:00084525v care
bn:00044492n home       -- hypernym --  bn:00046953n institution
bn:00044492n home       -- derivationally_related_form --       bn:00089431v home
bn:00044492n home       -- gloss_related_form_(disambiguated) --        bn:00046953n institution
bn:00044492n home       -- gloss_related_form_(disambiguated) --        bn:00061450n people
bn:00044492n home       -- semantically_related_form -- bn:00788129n retirement_home
bn:00044492n home       -- semantically_related_form -- bn:03335997n City
bn:00044492n home       -- semantically_related_form -- bn:00027976n physician
bn:00044492n home       -- semantically_related_form -- bn:00243111n physical_medicine_and_rehabilitation
bn:00044492n home       -- semantically_related_form -- bn:00813668n Elderly_care
bn:00044492n home       -- semantically_related_form -- bn:01694386n Clémence_Ross-van_Dorp
```

Programmer

Java API

Retrieve all hypernyms of a specific `BabelSynset`

Given a specific `BabelSynset` of *Rome* (**bn:00015556n**), the objective is to retrieve all `BabelSynset`s in the semantic network that have an hypernymy relation with it.



Retrieve all hypernyms of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00015556n & key=key`

Exact same call of the previous example!

HTTP API

Retrieve all hypernyms of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00015556n & key=key`


```
[{"language":"EN","pointer":{"fSymbol":"gmono","name":"Gloss related form (monosemous)","shortName":"gloss-
related","relationGroup":"OTHER","isAutomatic":false},"target":"bn:00016776n","weight":0.07568,"normalizedWei
ght":0.01191},{"language":"EN","pointer":{"fSymbol":"~@w","name":"Hyponym","shortName":"has-
kind","relationGroup":"HYPONYM","isAutomatic":true},"target":"bn:00068190n","weight":0.0,"normalizedWeight":0
.0},{"language":"EN","pointer":{"fSymbol":"gmono","name":"Gloss related form
(monosemous)","shortName":"gloss-
related","relationGroup":"OTHER","isAutomatic":false},"target":"bn:00068190n","weight":0.07821,"normalizedWei
ght":0.01231},
```


•
•
•


```
,{"language":"EN","pointer":
{"fSymbol":"wd2","name":"stated_in","shortName":"stated_in","relationGroup":"OTHER","isAutomatic":false},"tar
get":"bn:01351508n","weight":0.0,"normalizedWeight":0.0},{"language":"EN","pointer":
{"fSymbol":"wd53","name":"located_in_time_zone","shortName":"located_in_time_zone","relationGroup":"OTHER","i
sAutomatic":false},"target":"bn:01623792n","weight":0.0,"normalizedWeight":0.0},{"language":"EN","pointer":
{"fSymbol":"wd33","name":"determination_method","shortName":"determination_method","relationGroup":"OTHER","i
sAutomatic":false},"target":"bn:00017097n","weight":0.0,"normalizedWeight":0.0},{"language":"EN","pointer":
{"fSymbol":"wd33","name":"determination_method","shortName":"determination_method","relationGroup":"OTHER","i
sAutomatic":false},"target":"bn:00065677n","weight":0.0,"normalizedWeight":0.0}]
```


Browser User

HTTP API

Retrieve all hypernyms of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00015556n & key=key`

```
<html>
<head>
    <script src="http://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js"></script>
</head>
<body>
<script>
    var service_url = 'https://babelnet.io/v2/getEdges';
    var id = 'bn:00015556n'
    var key =

    var params = {
        'id': id,
    'key' : key
    };

    $.getJSON(service_url + "?", params, function(response) {

        $.each(response, function(key, val) {

        var pointer = val['pointer'];
        var edge = pointer['name'];
        var group = pointer['relationGroup'];

        if (group.toLowerCase().indexOf("hypernym") > -1) {
        var entry = "Source: " + id
            + "<br/>Target: " + val['target']
            + "<br/>Edge: " + edge
            + "<br/>Relation group: " + group + "<br/><br/>";
        $('<div>', {html:entry}).appendTo(document.body);
        }
        });

    });
</script>
</body>
</html>
```

Edge type information is encoded in the field '`relationGroup`' in '`pointer`'

Retrieve all hypernyms of a specific BabelSynset

URL: `https://babelnet.io/v3/getEdges?id=bn:00015556n & key=key`

Source: bn:00015556n
Target: bn:00056922n
Edge: Hypernym
Relation group: HYPERNYM

Source: bn:00015556n
Target: bn:00056922n
Edge: Instance hypernym
Relation group: HYPERNYM

Source: bn:00015556n
Target: bn:03335997n
Edge: Hypernym
Relation group: HYPERNYM

Source: bn:00015556n
Target: bn:03335997n
Edge: Hypernym
Relation group: HYPERNYM

Retrieve all hypernyms of a specific BabelSynset

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.BabelSynsetIDRelation;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.babelnet.data.BabelPointer;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class ExampleHypernyms {
    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        BabelSynset by = bn.getSynset(new BabelSynsetID("bn:00015556n"));
        for(BabelSynsetIDRelation edge : by.getEdges(BabelPointer.ANY_HYPERNYM)) {
            System.out.println(by.getId()+" "+by.getMainSense(Language.EN).getLemma()+"\t-- "
                    + edge.getPointer()+" --\t"
                    + edge.getBabelSynsetIDTarget()+" "+
                        edge.getBabelSynsetIDTarget().toBabelSynset().getMainSense(Language.EN).getLemma());
        }
    }
}
```

Retrieve all hypernyms of a specific `BabelSynset`

```java
import it.uniroma1.lcl.babelnet.BabelNet;
import it.uniroma1.lcl.babelnet.BabelSynset;
import it.uniroma1.lcl.babelnet.BabelSynsetID;
import it.uniroma1.lcl.babelnet.BabelSynsetIDRelation;
import it.uniroma1.lcl.babelnet.InvalidBabelSynsetIDException;
import it.uniroma1.lcl.babelnet.data.BabelPointer;
import it.uniroma1.lcl.jlt.util.Language;

import java.io.IOException;

public class ExampleHypernyms {
    public static void main(String[] args) throws IOException, InvalidBabelSynsetIDException {
        BabelNet bn = BabelNet.getInstance();
        BabelSynset by = bn.getSynset(new BabelSynsetID("bn:00015556n"));
        for(BabelSynsetIDRelation edge : by.getEdges(BabelPointer.ANY_HYPERNYM)) {
            System.out.println(by.getId()+" "+by.getMainSense(Language.EN).getLemma()+"\t-- "
                    + edge.getPointer()+" --\t"
                    + edge.getBabelSynsetIDTarget()+" "+
                        edge.getBabelSynsetIDTarget().toBabelSynset().getMainSense(Language.EN).getLemma());
        }
    }
}
```

Retrieve all the hypernym edges by specifying a constraint to `BabelSynset#getEdges`

`BabelSynsetIDRelation#getPointer` encodes information about the type of edge (e.g. hypernym, semantically related from)

Print ID and main sense of the connected `BabelSynset` (edge target)

Retrieve all hypernyms of a specific `BabelSynset`

Output:

```
bn:00015556n Rome       -- hypernym --  bn:00056922n national_capital
bn:00015556n Rome       -- instance_hypernym -- bn:00056922n national_capital
bn:00015556n Rome       -- hypernym --  bn:03335997n City
bn:00015556n Rome       -- hypernym --  bn:00015498n capital
bn:00015556n Rome       -- hypernym --  bn:00064917n provincial_capital
bn:00015556n Rome       -- hypernym --  bn:03335997n City
bn:00015556n Rome       -- hypernym --  bn:03219144n comune
bn:00015556n Rome       -- hypernym --  bn:03219144n comune
```

•
•
•

Wrap-up exercise

Wrap-up exercise

Given a pair of related words, e.g. *Apple* and *Microsoft*, find the pair of corresponding `BabelSynset`s that share the largest number of neighbors in the network.

Wrap-up exercise

Given a pair of related words, e.g. *Apple* and *Microsoft*, find the pair of corresponding `BabelSynset`s that share the largest number of neighbors in the network.

Three steps:

1. Find a set of associated `BabelSynset` for each word;
2. Find the set of neighbors of each `BabelSynset`;
3. Compare each possible pair of such sets and select the the pair with the **largest** intersection.

Wrap-up exercise: structure

```
public class CompareNeighbors {

        BabelNet bn = BabelNet.getInstance();

        /**
         * Given two words, select the pair of {@link BabelSynset}s that share the largest number of neighbors.
         *
         * @param word1
         * @param word2
         * @return
         */
        public SynsetPair selectClosestSynsetPair(String word1,String word2)

        /**
         * Given a list of {@link BabelSynset}s, retrieve the set of connected {@link BabelSynsetID}s.
         *
         * @param synsetList
         */
        protected Map<BabelSynsetID,Set<BabelSynsetID>> retrieveNeighbors(List<BabelSynset> synsetList)

        public class SynsetPair
        {
            protected BabelSynset first;
            protected BabelSynset second;

            public SynsetPair(BabelSynset first, BabelSynset second) {
                this.first = first;
                this.second = second;
            }

            public BabelSynset getFirst() { return this.first; }

            public BabelSynset getSecond() { return this.second; }
        }

        public static void main(String[] args)
}
```

Wrap-up exercise: structure

```
public class CompareNeighbors {

        BabelNet bn = BabelNet.getInstance();

        /**
         * Given two words, select the pair of {@link BabelSynset}s that share the largest number of neighbors.
         *
         * @param word1
         * @param word2
         * @return
         */
        public SynsetPair selectClosestSynsetPair(String word1,String word2)

        /**
         * Given a list of {@link BabelSynset}s, retrieve the set of connected {@link BabelSynsetID}s.
         *
         * @param synsetList
         */
        protected Map<BabelSynsetID,Set<BabelSynsetID>> retrieveNeighbors(List<BabelSynset> synsetList)

        public class SynsetPair
        {
            protected BabelSynset first;
            protected BabelSynset second;

            public SynsetPair(BabelSynset first, BabelSynset second) {
                this.first = first;
                this.second = second;
            }

            public BabelSynset getFirst() { return this.first; }

            public BabelSynset getSecond() { return this.second; }
        }

        public static void main(String[] args)
}
```

Instance of `BabelNet`

Main function that does the job

Auxiliary function that retrieves neighbor `BabelSynset`s

Auxiliary nested class for encoding pairs of `BabelSynset`s

Wrap-up exercise: retrieve neighbors

We need to compute the set of neighbors for each `BabelSynset` inside a list given as input:

```
protected Map<BabelSynsetID,Set<BabelSynsetID>> retrieveNeighbors(List<BabelSynset> synsetList)
{
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap = new HashMap<>();
    for(BabelSynset synset : synsetList) {
        BabelSynsetID id = synset.getId();
        Set<BabelSynsetID> neighbors = synset.getEdges().stream()
                .map(BabelSynsetIDRelation::getBabelSynsetIDTarget).collect(Collectors.toSet());
        neighborsMap.put(id, neighbors);
    }
    return neighborsMap;
}
```

Wrap-up exercise: retrieve neighbors

We need to compute the set of neighbors for each `BabelSynset` inside a list given as input:

```
protected Map<BabelSynsetID,Set<BabelSynsetID>> retrieveNeighbors(List<BabelSynset> synsetList)
{
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap = new HashMap<>();
    for(BabelSynset synset : synsetList) {
        BabelSynsetID id = synset.getId();
        Set<BabelSynsetID> neighbors = synset.getEdges().stream()
                .map(BabelSynsetIDRelation::getBabelSynsetIDTarget).collect(Collectors.toSet());
        neighborsMap.put(id, neighbors);
    }
    return neighborsMap;
}
```

Generate a map indexed by `BabelSynsetID` that contains, for each synset, a set of `BabelSynsetID`s

First call `BabelSynset#getEdges` and then pick the target `BabelSynsetID` from each edge

Wrap-up exercise

```
public SynsetPair selectClosestSynsetPair(String word1,String word2)
{
    // 1. Find list of synsets for each word
    List<BabelSynset> synsetList1 = bn.getSynsets(word1, Language.EN);
    List<BabelSynset> synsetList2 = bn.getSynsets(word2, Language.EN);

    // 2. Find neighbors for each synset
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap1 = retrieveNeighbors(synsetList1);
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap2 = retrieveNeighbors(synsetList2);

    // 3. Compare synset pairs and select the maximum
    Map<SynsetPair,List<BabelSynsetID>> intersectionMap = new HashMap<>();
    for(BabelSynset synset1 : synsetList1) {
        for(BabelSynset synset2 : synsetList2) {
            List<BabelSynsetID> intersection = neighborsMap1.get(synset1.getId()).stream()
                    .filter(neighborsMap2.get(synset2.getId())::contains).collect(Collectors.toList());
            intersectionMap.put(new SynsetPair(synset1,synset2), intersection);
        }
    }
    return intersectionMap.keySet().stream()
            .max((s1,s2) -> Integer.compare(intersectionMap.get(s1).size(), intersectionMap.get(s2).size()))
            .orElseThrow(NullPointerException::new);
}
```

Wrap-up exercise

```
public SynsetPair selectClosestSynsetPair(String word1,String word2)
{
    // 1. Find list of synsets for each word
    List<BabelSynset> synsetList1 = bn.getSynsets(word1, Language.EN);
    List<BabelSynset> synsetList2 = bn.getSynsets(word2, Language.EN);

    // 2. Find neighbors for each synset
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap1 = retrieveNeighbors(synsetList1);
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap2 = retrieveNeighbors(synsetList2);


}
```

Use `BabelNet#getSynsets` to retrieve all the `BabelSynset`s associated with the words

Use the function we just wrote to associate each `BabelSynset` to a list of neighbors

Wrap-up exercise

```
public SynsetPair selectClosestSynsetPair(String word1,String word2)
{
    // 1. Find list of synsets for each word
    List<BabelSynset> synsetList1 = bn.getSynsets(word1, Language.EN);
    List<BabelSynset> synsetList2 = bn.getSynsets(word2, Language.EN);

    // 2. Find neighbors for each synset
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap1 = retrieveNeighbors(synsetList1);
    Map<BabelSynsetID,Set<BabelSynsetID>> neighborsMap2 = retrieveNeighbors(synsetList2);

    // 3. Compare synset pairs and select the maximum
    Map<SynsetPair,List<BabelSynsetID>> intersectionMap = new HashMap<>();
    for(BabelSynset synset1 : synsetList1) {
        for(BabelSynset synset2 : synsetList2) {
            List<BabelSynsetID> intersection = neighborsMap1.get(synset1.getId()).stream()
                    .filter(neighborsMap2.get(synset2.getId())::contains).collect(Collectors.toList());
            intersectionMap.put(new SynsetPair(synset1,synset2), intersection);
        }
    }
    return intersectionMap.keySet().stream()
            .max((s1,s2) -> Integer.compare(intersectionMap.get(s1).size(), intersectionMap.get(s2).size()))
            .orElseThrow(NullPointerException::new);
}
```

Compare all possible pairs of `BabelSynset`s by looping over the two lists

Compute the `SynsetPair` that maximizes the number of shared neighbor `BabelSynsets`

Wrap-up exercise

Main class:

```
public static void main(String[] args)
{
    SynsetPair closestPair = new CompareNeighbors().selectClosestSynsetPair("Apple", "Microsoft");

    // Print closest pair
    System.out.println(closestPair.getFirst().getMainSense(Language.EN)+"\t"
            +closestPair.getSecond().getMainSense(Language.EN));

}
```

Wrap-up exercise

Main class:

```
public static void main(String[] args)
{
    SynsetPair closestPair = new CompareNeighbors().selectClosestSynsetPair("Apple", "Microsoft");

    // Print closest pair
    System.out.println(closestPair.getFirst().getMainSense(Language.EN)+"\t"
            +closestPair.getSecond().getMainSense(Language.EN));

}
```

Output:

WIKI:EN:Apple_Inc.

bn:03739345n

WIKI:EN:Microsoft

bn:01165400n

Wrap-up exercise

Let's try instead with another word pair:

```
public static void main(String[] args)
{
    SynsetPair closestPair = new CompareNeighbors().selectClosestSynsetPair("Apple", "Pear");

    // Print closest pair
    System.out.println(closestPair.getFirst().getMainSense(Language.EN)+"\t"
            +closestPair.getSecond().getMainSense(Language.EN));

}
```

Wrap-up exercise

Let's try instead with another word pair:

```
public static void main(String[] args)
{
    SynsetPair closestPair = new CompareNeighbors().selectClosestSynsetPair("Apple", "Pear");

    // Print closest pair
    System.out.println(closestPair.getFirst().getMainSense(Language.EN)+"\t"
            +closestPair.getSecond().getMainSense(Language.EN));

}
```

Output:

WN:EN:apple

bn:00005054n

WN:EN:pear

bn:00061187n

Wrapping up:

Wrapping up:

- You can access BabelNet data programmatically using a **dedicated easy-to-use API** with convenient methods and classes to query the knowledge base and work with the data;

Wrapping up:

- You can access BabelNet data programmatically using a **dedicated easy-to-use API** with convenient methods and classes to query the knowledge base and work with the data;

- The API is based on a **HTTP RESTful service** that relies on an internal credit mechanism (**Babelcoins**) for users registered to the BabelNet service;

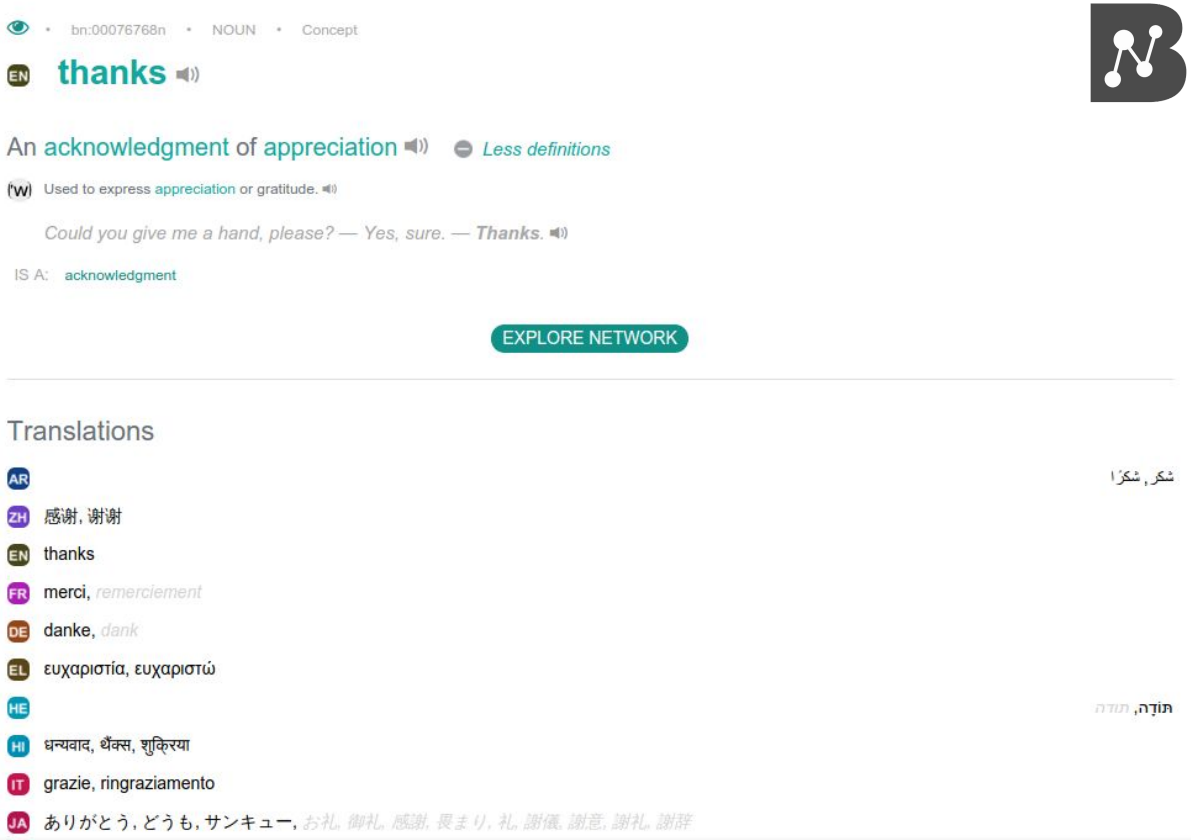
Wrapping up:

- You can access BabelNet data programmatically using a **dedicated easy-to-use API** with convenient methods and classes to query the knowledge base and work with the data;

- The API is based on a **HTTP RESTful service** that relies on an internal credit mechanism (**Babelcoins**) for users registered to the BabelNet service;

- The API also comes with a powerful **Java binding** to the very same service: hence you are free to use directly the RESTful API (using your only favourite language) or the Java API (if you are a Java/Scala/Groovy programmer).

Thank you!


bn:00076768n • NOUN • Concept
EN thanks
An acknowledgment of appreciation
Less definitions
(W) Used to express appreciation or gratitude.
Could you give me a hand, please? — Yes, sure. — Thanks.
IS A: acknowledgment
EXPLORE NETWORK
Translations
AR شكر, شكرًا
ZH 感谢, 谢谢
EN thanks
FR merci, remerciement
DE danke, dank
EL ευχαριστία, ευχαριστώ
HE תּוֹדָה, תודה
HI धन्यवाद, थैंक्स, शुक्रिया
IT grazie, ringraziamento
JA ありがとう, どうも, サンキュー, お礼, 御礼, 感謝, 畏まり, 礼, 謝儀, 謝意, 謝礼, 謝辞