

Thea Open Calais wrapper module

1 Introduction

[Open Calais](#) is a Web service provided by Thomson Reuters. Open Calais analyses content using NLP and semantic techniques and returns an RDF based response containing entities and relationships identified in the source and associated metadata.

thea_opencalais is a Prolog wrapper for accessing Open Calais and process the results. It is an application of [Thea](#) prolog library for OWL2 Ontologies. thea_opencalais uses the OpenCalais ontology and parses the service response first into Ontology axioms and finally into prolog terms and predicates, accessible from within a prolog program.

2 Example

1. Get your Open Calais license key and assert it as a prolog fact:

```
:- assert(thea_opencalais:open_calais(license('Your open calais license
key'))).
```

2. Load the Open Calais Ontology into Prolog using Thea.

```
:- owl_parse_rdf('owl.opencalais-
4.3.xml.owl',[imports(false),clear(complete)]).
```

The Ontology in this example has been already downloaded in the working directory. Originally it can be found [here](#)

3. Post the content of a Wikipedia page about papal visits to the Open Calais REST service

```
:-
oc_rest(http('http://en.wikipedia.org/wiki/List_of_journeys_of_Pope_Benedict_X
VI'),'','_X').
```

4. Use Prolog provided prolog predicates to examine Markup Elements (Entities and Relationships) in the result

```
:-oc_entity(A,B,C,E,D).
   A = 'http://d.opencalais.com/genericHasher-1/f545c2a6-ccd3-3095-adb0-
c1c8dda96624',
   B = 'http://s.opencalais.com/1/type/em/e/Anniversary',
   C = ['http://s.opencalais.com/1/pred/name'=literal('the 500th
anniversary of Catholic presence')],
   E = [instance_info('http://d.opencalais.com/dochash-1/2d1827f8-c251-
36b1-bfe6-043a4a6cab1/Instance/268',[.....
   D = []
```

5. You can also write custom predicates to query the resulted database of Markup Elements e.g.

```
quotation(Person,Quotation) :-
    oc_relation(_I,'http://s.opencalais.com/1/type/em/r/Quotation',PVL
ist),
    pv_attr('http://s.opencalais.com/1/pred/person',PVList,Person),
```

3 Predicate reference

oc_rest(+Request, +Params:string, -Result) is det

Calls Open Calais Rest service and posts the *Request*. Requires a valid Open_Calais license key in a dynamic open_calais(license(License)) to be asserted.

Parameters:

- can be any of the following:

Request **http**(URL)
file(Filename)
text(Atom)

Params - is a string encoded XML see description in <http://www.opencalais.com/APICalls>

Result - is the RDF response of Open_Calais see
<http://www.opencalais.com/documentation/calais-web-service-api/interpreting-api-response/rdf>

oc_entity(?I, ?C, -PV:list, +Instance:list, +Resolutions:list) is nondet

Returns the markup element entities in the response to an Open Calais request

Parameters:

I - is the returned markup entity ID (IRI)

C - is the entitie's markup type

PV - list of property value pairs for the entity

Instance - List of instance_info terms for the specific entity

Resolutions - List of resolution terms for the specific entity

oc_relation(?I, ?C, -PV:list) is nondet

Returns the markup element relations in the response to an Open Calais request

Parameters:

I - is the returned markup relation ID (IRI)

C - is the relation's markup type

PV - list of property value pairs for the relation

oc_resolution(?R, ?C, -PV:list) is nondet

Returns the resolutions in the response to an Open Calais request

Parameters:

R - is the returned resolution ID (IRI)

C - is the resolution's type

PV - list of property value pairs for the

resolution

4 Download and use

Thea and thea_opencalais are open source licensed under GPL and are hosted on [github](#). thea_opencalais module and examples can be found in apps/opencalais folder