WikipediaBase Architecture

Chris Perivolaropoulos

Sunday 21 February 2016

Contents

1 Infobox

Infoboxes are tables that are commonly used in wikipedia to provide an overview of the information in an article in a semi structured way. Infoboxes are the main source of information for WikipediaBase.

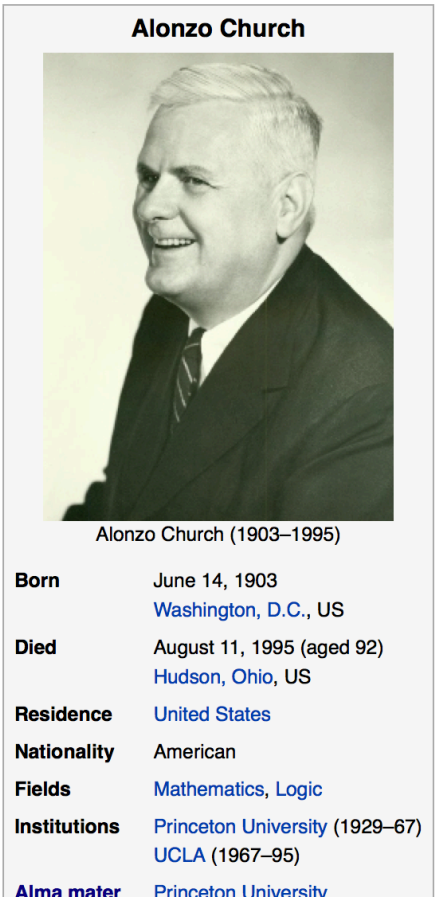

Figure 1: An example of an infobox

In mediawiki markup terms an infobox is a markup template with a type that gets rendered into html so that the provided information makes sense in the context that it is provided. For example:

```
{{Infobox scientist
| name              = Gerhard Gentzen
| image             = Gerhard Gentzen.jpg
| image_size        =
| alt               =
| caption           = Gerhard Gentzen in Prague,
   1945.
| birth_date        = {{Birth date|1909|11|24}}
| birth_place       = [[Greifswald]], [[Germany]]
```

```
| death_date         = {{Death date and age
   |1945|8|4|1909|11|24}}
| death_place        = [[Prague]], [[Czechoslovakia]]
| nationality        = [[Germany|German]]
| fields             = [[Mathematics]]
| workplaces         =
| alma_mater         = [[University of Gottingen]]
| doctoral_advisor   = [[Paul Bernays]]
| doctoral_students  =
| known_for          =
| awards             =
}}
```

will yield:

Figure 2: An example of an infobox

Infobox types are organized into a fairly wide hierarchy. For example Template:Infobox Austrian district is a special case of a Template:Infobox settlement and each is rendered differently. For our purposes, and to mirror the markup definition of infoboxes, an infobox I with attributes a_i and values v_i is a set of pairs (a_i, v_i) together with a infobox type t. Each attribute a_i and value v_i have two forms:

- a rendered form, a_i^r and v_i^r respectively, which is the rendered HTML representation and
- a markup form, a_i^m and v_i^m which is the mediawiki markup code that corresponds to them.

An article may have more than one infoboxes, for example Bill Clinton article has both Infobox Officeholder and Infobox President infoboxes.

The `Infobox` class is the basic data type for accessing information from the infobox of an article. `Infobox`, as well as `Article`, are what one would use were they to use wikipediabase as a python library. The methods provided by an infobox are:

types Because we retrieve an infobox based on a symbol name (ie page name), a single `Infobox` may actually be an interface for multiple infoboxes. There is a separate method, based on this one, for getting types in a format suitable for START.

Value access is possible provided either a_i^r or a_i^m.

Rendered keys are provided using the `MetaInfobox` (see below).

Infobox export to python types, namely:

- dict for $a_i^r \rightarrow v_i^r$ or $a_i^m \rightarrow v_i^m$
- the entire infobox rendered, or in markup form.

Infoboxes are organized in a wide hierarchy that in the WikiepdiaBase codebase is refered to as infobox tree. The infobox tree is retrieved from the list of infoboxes wikipedia page and is used to deduce the ontology of wikipedia terms.

2 MetaInfobox

The `MetaInfobox` is implemented as a subclass of the `Infobox` that provides information about the infobox, most importantly a map between markup attributes and the rendered counterparts. Say we have an infobox of type I which has attributes $a_1, ..., a_n$. Each attribute has two representations:

- a markup representation that is the key used in the infobox template.
- The HTML rendered representation, that is the left-column text of the rendered infobox table.

For example in the `officeholder` infobox there is an attribute that has markup representation `predecessor` and rendered representation `Preceded by`.

To do this the `MetaInfobox` uses the template documentation page to find the markup representation of all valid attributes of an infobox type. It then creates an infobox where each attribute has value its markup attribute name wrapped int `!!!`. (for example `predecessor = !!!predecessor!!!`). It then renders the created infobox and looks for `!!!predecessor!!!` in the rendered values. The corresponding rendered attribute names also correspond to the markup attribute names. Note that the correspondence of rendered - markup attributes is not bijective, that is to say each markup attribute may correspond to zero or more rendered attributes and vice versa.

For example, an infobox of type `Foo` has valid attributes A, B, C and D. The generated infobox markup would be:

```
{{Infobox Foo
| A = !!!A!!!
| B = !!!B!!!
| C = !!!C!!!
| D = !!!D!!!
}}
```

And the rendered version could be, depending on the implementation of the `Foo` infobox.

Attribute	Value
A	!!!A!!! !!!B!!! !!!C!!!
B	!!!A!!! !!!B!!! !!!C!!!
C	!!!A!!! !!!B!!! !!!C!!!
D	!!!D!!!

Which makes the mapping fairly obvious.

3 Article

The `Article` data structure is responsible for accessing any resource relevant to the article at large. This includes paragraphs, headings, markup source and the mediawiki categores.

4 Fetcher

The fetcher is an abstraction over the communication of WikipediaBase with the outside world. It is a singleton object that implements a specific interface.

Fetchers are organized in an inheriting hierarchiy

BaseFetcher The baseclass for fetchers, it will return the symbol instead of trying to resolve it in any way

Fetcher contains the core functionality of a a fetcher. It will fetch articles from *wikipedia.org*. It is possible to direct it to a mirror but wikipedia-mirror's runtime performance turned out to be prohibitive.

CachingFetcher inherits fetcher and retains its functionality, only it uses Redis to cache the fetched symbols. It is the default fetcher for wikipediabase.

StaticFetcher is a class that implements the `BaseFetcher` interface but instead of reaching out to some data source for the data the return values are statically defined. It is used most notably by `MetaInfobox` to use the `Infobox` functionality to convey arbitrary information.

By default, markup is fetched from the backend. If $\text{force}_{\text{live}}$ is set to True, the markup will be fetched from live wikipedia.org

When tests are ran on TravisCI, we always want to use live data. We check if Travis is running tests by looking at the $\text{WIKIPEDIABASE}_{\text{FORCELIVE}}$ env variable.

5 Renderer

Renderers are singleton classes that are useful for rendering mediawiki markup into HTML. Originally the wikiepedia sandbox was used by wikipediabase for rendering pages because it is slightly faster than the API, but the wikipedia-mirror was really slow at this and wikipedia.org would consider it an abuse of the service and block our IP. For that reason we eventually switched to the API with Redis caching, which works out pretty well because `Renderer` objects end up being used only by `MetaInfobox` which has quite a limited scope, making thus cache misses rarely.

An interesting anecdote about the `Renderer` class was that it was the reason for a couple of CSAIL IPs to get temporarily banned from editing wikipedia. While wikipedia.org has a very lenient policy when it comes to banning people who are spamming their servers, repeated testing of the `Renderer` class targeting wikipedia's sandbox caused the testing machine's ip to be temporarily banned on the grounds that "its activity does not promote the imporovement of wikipedia". We reimplemented the `Renderer` to use the wikipedia API and we never had a problem with wikipedia moderation again.

6 Pipeline

When resolving a query WikipediaBase employs a pipeline of modules to figure out what the best way to respond would be.

6.1 Frontend

WikipediaBase can be used as a library but its primary function is as a backend to START. The communication between START and WikipediaBase is carried out over a plaintext telnet connection on port 8023 using s-expressions. The frontend handles the network connection with START, translates the received queries into calls to knowledgebase and then translates the knowledgebase response into properly formulated s-expressions that it sends back over the telnet connection.

6.2 Knowledgebase

The knowledgebase is the entry point to the rest of wikipediabase. It uses the Provider/Acquirer pattern to transparently provide the frontend with arbitrary methods. Those methods are responsible for chosing whether we are to resort to classifiers or resolvers (or any other mechanism) for answering the query. Available classifiers and resolvers become accessible to the knowledgebase automatically using their base class.

6.3 Classifiers

Each classifier is a singleton that implements a heuristic for deducing a set of classes of an object. An object may inhibit zero or more classes. There are a couple classifiers available at the moment. Typically a classifier will only deduce whether an object actually inhibits a specific class or not but that is not necessary.

1. Term

   The `TermClassifier` simply assigns the `wikipedia-term` class. Wikipediabase only deals with wikipedia related information.

2. Infobox

   The `InfoboxClassifier` assigns to a term the classes of the infobox. For example Bill Clinton's page contains the infobox:

```
{{Infobox president
|name          = Bill Clinton
|image         = 44 Bill Clinton 3x4.jpg{{!}}border
[...]
}}
```

And therefore gets the class `wikipedia-president`.

3. Person

   `PersonClassifier` assigns the class `wikibase-person` using a few heuristics in the order they are described:

   (a) Category regexes

   Use the following regular expressions to match categories of an article.

   - `.* person`
   - `^\d+ deaths.*`
   - `^\d+ births.*`
   - `.* actors`
   - `.* deities`
   - `.* gods`
   - `.* goddesses`
   - `.* musicians`
   - `.* players`
   - `.* singers`

   (b) Category exclude regexes

   Exclude categories matching the following regexes.

   - `\sbased on\s`
   - `\sabout\s`
   - `lists of\s`
   - `animal\`

   (c) Category matches

   We know an article refers to a person if the page is in one or more of the following mediawiki categories :

   - `american actors`
   - `american television actor stubs`

- `american television actors`
- `architects`
- `british mps`
- `character actors`
- `computer scientist`
- `dead people rumoured to be living`
- `deities`
- `disappeared people`
- `fictional characters`
- `film actors`
- `living people`
- `musician stubs`
- `singer stubs`
- `star stubs`
- `united kingdom writer stubs`
- `united states singer stubs`
- `writer stubs`
- `year of birth missing`
- `year of death missing`

For an example of how this works see the appendix.

As it is obvious the list of categories is arbitrary and very far from complete. Multiple methods have been considered for fixing this. Some of them are:

- Supervised machine learning methods like SVM using other methods of determining person-ness to create training sets.
- Hand-pick common categories for person articles determined again with the other criteria

6.4 Resolvers

Resolvers are also singletons but their purpose is to find the value of the requested property. All resolvers descend from `BaseResolver` and should implement the following methods:

- `resolve(class, symbol, attribute)`: get the value of the `attribute` of symbol `symbol` as `class`

- `attributes(class, symbol)`: get a list of the attributes this resolver can resolve.

The implemented resolvers are the following:

Error the minimum priority resolver, it will always resolve to an error.

Infobox Resolve attributes found on Infoboxes of a symbol.

Person resolve the following specific attributes of symbols referring to people:

- `birth-date`
- `death-date`
- `gender`

Sections resolve the content of sections in an article.

Term Can resolve a fixed set of ad-hoc attributes:

- `coordinates` *The coordinates of a geographical location*
- `image` *The image in the infobox*
- `number` *True if the symbol is plural (eg The Beatles)*
- `proper` *True if it refers to a unique entity.*
- `short-article` *A summary of the article. Typically the first paragraph*
- `url` *The article url*
- `word-cout` *The size of the article*

6.5 Lisp types

Lisp type instances are wrappers for python objects or values that are presentable in s-expression form that START can understand. They are created either from the raw received query and unwrapped to be useful to the pipeline, or by the answer WikipediaBase comes up with and then encoded into a string sent over telnet to START.