Documentation of the LLaMaPUn C library

Jan Frederik Schaefer

August 8, 2014

Abstract

The LLaMaPUn C library contains some NLP tools for (mainly) mathematical documents. This document is supposed to give an overview about these tools - both for users and future developers of the library.

Contents

1 Installation (on Linux)

You can clone the source from github:

```
git clone https://github.com/KWARC/LLaMaPUn.git
```

We will need a couple of libraries:

- Libxml2
- LibJSON
- uthash
- libiconv

The `.travis.yml` file in the repository shows an example installation of the LLaMaPUn C library, including the installation of these libraries with `apt-get`.

Since there are some issues with older versions of the JSON library, you might want to run a small script to fix the includes, if necessary:

```
sh jsonincludecheck.sh > /dev/null
```

Now we can compile, test, and install the library, using `cmake`:

```
mkdir -p build
cd build
cmake ..
make
make test
sudo make install
sudo ldconfig
```

2 Including and linking the LLaMaPUn C library

After installing the library, you can include e.g. the DNM library writing

```
#include <llamapun/dnmlib.h>
```

If you use `gcc` or `clang` you can link the LLaMaPUn library by setting the `-lllamapun` flag.

3 DNM library

For the natural language processing of the XHTML documents created with LaTeXML, we need to switch between two different views on the data back and forth: On one hand, most NLP tools only work with plain text, on the other hand, the XML structure contains useful information about the structure of the document etc. The DNM library is supposed to provide an easy interface for doing NLP on these documents. The central idea is creating a DNM (document narrative model) out of the DOM, which could be perceived as a layer between the DOM and the plain text.

3.1 The document narrative model (DNM)

The DNM contains the plaintext as a simple string. Additionally, it contains three lists - a list of the paragraphs of the document, a list of the sentences, and a list of the words. These levels are identified by the annotations `ltx_para`, `ltx_sentence`, and `ltx_word`. One paragraph, sentence, or word is represented as a `dnm_chunk`. A `dnm_chunk` contains

- The offsets to the corresponding part of the plain text
- If possible, he offsets to the corresponding part of the lower level list (i.e. for example to the words of a sentence)
- If possible, the offset to the parent chunk (for example to the paragraph that contains the sentence)
- The annotations of that chunk, given by the the `class` attribute of the corresponding XML tag
- The annotations of the parent tags, which will be called "inherited annotations"
- The `id` of the corresponding XML tag
- A pointer to the corresponding node in the DOM

3.2 Creating and freeing the DNM

- `dnmPtr createDNM(xmlDocPtr doc, long parameters)`
- `freeDNM(dnmPtr dnm)`

`createDNM` creates the DNM of `doc` and returns a pointer to it. So far the only defined parameters are

- `DNM_NORMALIZE_MATH`
- `DNM_SKIP_MATH`
- `DNM_SKIP_CITE`

Some tags, e.g. `math` tags can be ignored, or normalized to a string, e.g. `[MathFormula]`. Note that the DNM has to be freed by calling `freeDNM`.

```
//Example:
dnmPtr dnm = createDNM(myDoc,
                 DNM_NORMALIZE_MATH | DNM_SKIP_CITE);
freeDNM(dnm);
```