

Full-text searching with Marjory

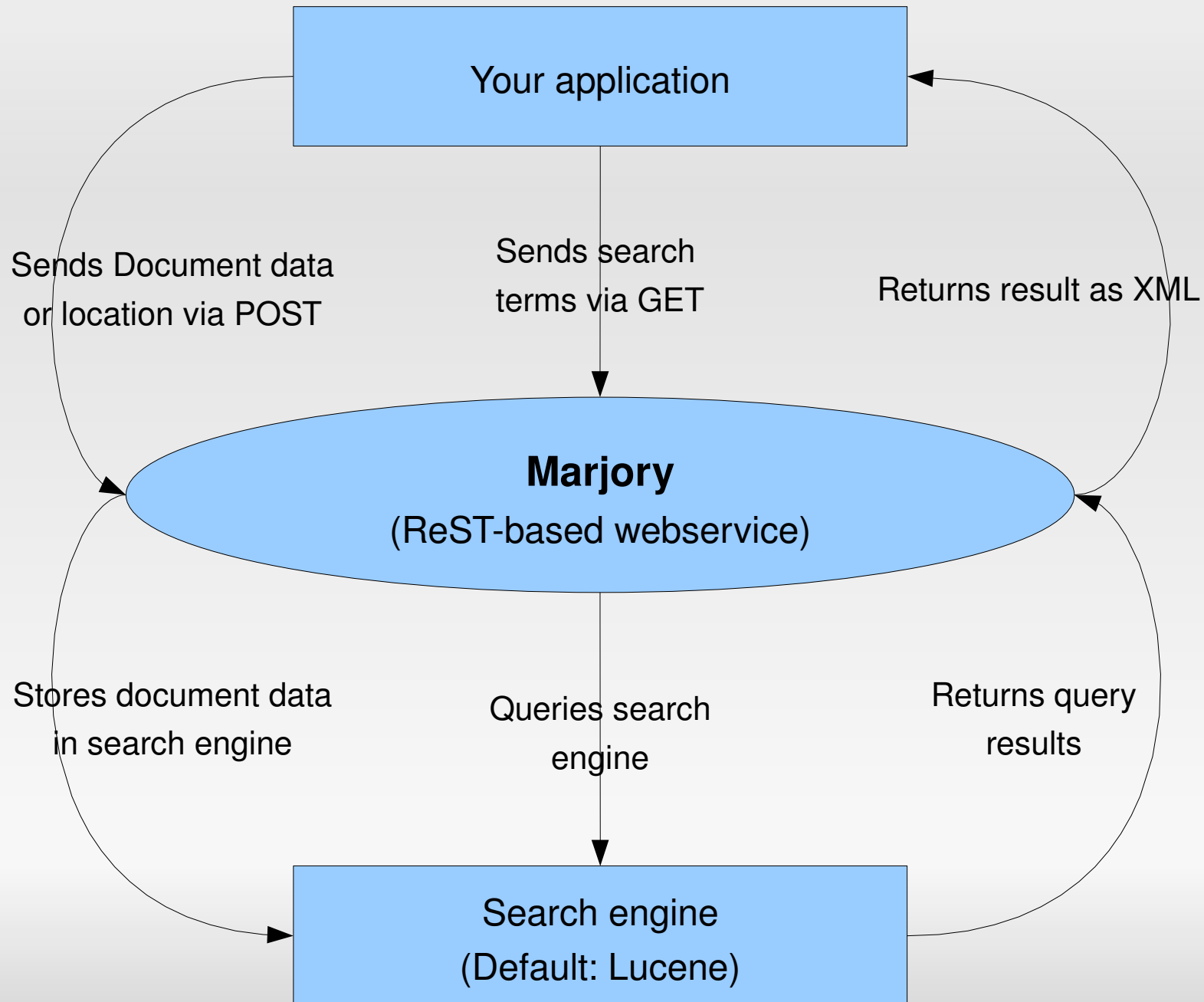


Markus Wolff

What's Marjory?

- A webservice for full-text indexing and searching of documents
- Written in PHP
- Based on Zend Framework
- (Very) Roughly comparable to Solr
- BSD-licensed, available on Google Code

How does Marjory work?



Features

- Search engine abstraction
 - use the engine that suits your needs, just write a small adaptor class
 - Zend_Search_Lucene built-in by default
- Multiple search catalogs
 - Index many sites with one dedicated search server
 - Put all documents matching any criteria into separate search indexes to speed up search

More features

- Two ways to index documents:
 - submit an XML snippet containing any content you want to index
 - **or**, just submit an URI (valid PHP stream resource) and let Marjory extract the content from the document
 - HTML supported by default (for now)
 - add your own document parser class to extract plain text from any other document format (or special markup structures)

Even more features

- Index documents asynchronously using Dropr as a messaging service
 - Dropr: PHP-based durable messaging service
 - Example webservice and Dropr client included with Marjory
 - Application does not need to wait for document retrieval, parsing and adding to the index
 - More info: www.dropr.org

And even more feat...

- ...not really, that's it for now
- It's a very young project :-)

How to add a catalog

- Send a POST request to:
`http://marjory.example.com/rest/catalog/`
- Containing this XML snippet:
`<add catalog="MyGloriousCatalog" />`
- Et voilà, you got yourself a new search index

Adding a document

- Make a POST request to:
<http://marjory.example.com/rest/add/>
- Send the document content as XML like this:

```
<add catalog="default">
  <doc uri="MyUniqueDocumentId">
    <field name="title">Marjory: Search as a service</field>
    <field name="abstract">
      An epic novel about full-text indexing in an SOA environment
    </field>
    <field name="content">Lorem ipsum dolor sit amet... (to be continued)</field>
  </doc>
</add>
```

Adding a document, the easy way

- Or, if Marjory should retrieve and parse the document:

```
<add catalog="default">
```

```
  <doc src="http://my.website.tld/my/document.html" />
```

```
</add>
```

- If you have many and/or complex documents, better use Dropr to send messages to Marjory

Searching for documents

- Make a GET request including the query terms:
`http://marjory.example.com/rest/select?q=Marjory`
- Additional parameters to...
 - Limit number of results
 - Include only specific fields in response
 - Specify a search catalog
 - Default catalog name: „default“ - who would have guessed?

Search response format

```
<?xml version="1.0" encoding="UTF-8"?>
<response>
  <responseHeader>
    <status>0</status><QTime>1</QTime>
  </responseHeader>

  <result numFound="2" start="0">
    <doc>
      <str name="id">MA147LL/A</str>
      <str name="name">Apple 60 GB iPod Black</str>
    </doc>
    <doc>
      <str name="id">EN7800GTX/2DHTV/256M</str>
      <str name="name">ASUS Extreme N7800GTX</str>
    </doc>
  </result>
</response>
```

Looks familiar?

- Blatantly stolen from Solr :-)
- Why reinvent the wheel?
- Makes switching between the two projects easy if need be

Access control

- No access control provided by Marjory
- Use your webserver's authentication and ACL capabilities
- There are currently no plans to add anything built-in, unless someone convinces me otherwise :-)

Things to do

- Fully unit-test the beast
- Add a nice admin GUI (there is a small GUI, but more for testing purposes)
- Add other engines
- Support more document formats out of the box
- Fine-tuning (how about renaming or removing catalogs, for example?)

Is it production-ready?

- Yes, and it's already being used on production websites

That's all, folks!

- More information:
 - <http://code.google.com/p/marjory/>
 - <http://www.dropr.org/>
 - <http://blog.wolff-hamburg.de/>