Apache Tika
What’s new with 2.0?
“small, yellow and leech-like, and probably the oddest thing in the Universe”

• Like a Babel Fish for content!
• Helps you work out what sort of thing your content (1s & 0s) is
• Helps you extract the metadata from it, in a consistent way
• Lets you get a plain text version of your content, eg for full text indexing
• Provides a rich (XHTML) version too
Tika in the news

- Panama Papers – Tika used to extract content from most of the files before indexing in Apache SOLR

- MEMEX – DARPA funded project
  https://nakedsecurity.sophos.com/2015/02/16/memex-darpas-search-engine-for-the-dark-web/

Tika at ApacheCon

• Tim Allison, tomorrow (Thursday), 2.40pm
  Evaluating Text Extraction: Apache Tika's™ New Tika-Eval Module

• Also related: David North (same time...)
  Apache POI: The Challenges and Rewards of a 15 Year Old Codebase

• Several Committers around, come find us!
A bit of history
Before Tika

• In the early 2000s, everyone was building a search engine / search system for their CMS / web spider / etc
• Lucene mailing list and wiki had lots of code snippets for using libraries to extract text
• Lots of bugs, people using old versions, people missing out on useful formats, confusion abounded
• Handful of commercial libraries, generally expensive and aimed at large companies and/or computer forensics
• Everyone was re-inventing the wheel, and doing it badly....
Tika's History (in brief)

• The idea from Tika first came from the Apache Nutch project, who wanted to get useful things out of all the content they were spidering and indexing
• The Apache Lucene project (which Nutch used) were also interested, as lots of people there had the same problems
• Ideas and discussions started in 2006
• Project founded in 2007, in the Apache Incubator
• Initial contributions from Nutch, Lucene and Lius
• Graduated in 2008, v1.0 in 2011
Tika Releases

![Tika Releases Chart]

Chart showing the release timeline of Tika from 0.1 to 1.14, with dates ranging from 01/07 to 12/17.
A (brief) introduction to Tika
(Some) Supported Formats

- HTML, XHTML, XML
- Microsoft Office – Word, Excel, PowerPoint, Works, Publisher, Visio – Binary and OOXML formats
- OpenDocument (OpenOffice)
- iWorks – Keynote, Pages, Numbers
- PDF, RTF, Plain Text, CHM Help
- Compression / Archive – Zip, Tar, Ar, 7z, bz2, gz etc
- Atom, RSS, ePub
- Audio – MP3, MP4, Vorbis, Opus, Speex, MIDI, Way
- Image – JPEG, TIFF, PNG, BMP, GIF, ICO
Detection

- Work out what kind of file something is
- Based on a mixture of things
  - Filename
  - Mime magic (first few hundred bytes)
  - Dedicated code (eg containers)
  - Some combination of all of these
- Can be used as a standalone – what is this thing?
- Can be combined with parsers – figure out what this is, then find a parser to work on it
Metadata

• Describes a file
  • eg Title, Author, Creation Date, Location
• Tika provides a way to extract this (where present)
• However, each file format tends to have its own kind of metadata, which can vary a lot
  • eg Author, Creator, Created By, First Author, Creator[0]
• Tika tries to map file format specific metadata onto common, consistent metadata keys
• “Give me the thing that closest represents what Dublin Core defines as Creator”
Plain Text

- Most file formats include at least some text
- For a plain text file, that's everything in it!
- For others, it's only part
- Lots of libraries out there which can extract text, but how you call them varies a lot
- Tika wraps all that up for you, and gives consistency
- Plain Text is ideal for things like Full Text Indexing, eg to feed into SOLR, Lucene or ElasticSearch
• Structured Text extraction
• Outputs SAX events for the tags and text of a file
• This is actually the Tika default, Plain Text is implemented by only catching the Text parts of the SAX output
• Isn't supposed to be the “exact representation”
• Aims to give meaningful, semantic but simple output
• Can be used for basic previews
• Can be used to filter, eg ignore header + footer then give remainder as plain text
Tika “Architecture”, in brief

- Hide complexity
- Hide differences
- Identify, pick and use the “best” libraries and tools
- Work with all the upstreams for you
- Come “Batteries Included” where possible / not too big, “Batteries Nearby” otherwise
- Try to avoid surprises
- Support JVM + Non-JVM users as equals
- Work to fix any of the above that we happen to miss!
What's New?
Formats and Parsers
Supported Formats

- HTML
- XML
- Microsoft Office
  - Word
  - PowerPoint
  - Excel (2,3,4,5,97+)
  - Visio
  - Outlook
  - Pre-OOXML XML formats, Lock Files etc!
Supported Formats

- Open Document Format (ODF)
- iWorks, Word Perfect
- PDF, RTF
- ePUB
- Fonts + Font Metrics
- Tar, RAR, AR, CPIO, Zip, 7Zip, Gzip, BZip2, XZ and Pack200
- Plain Text
- RSS and Atom
Supported Formats

- IPTC ANPA Newswire
- CHM Help
- Wav, MIDI
- MP3, MP4 Audio
- Ogg Vorbis, Speex, FLAC, Opus, Theora
- PNG, JPG, JP2, JPX, BMP, TIFF, BPG, ICNS, PSD, PPM, WebP
- FLV, MP4 Video – Metadata and video histograms
- Java classes
Supported Formats

- Source Code
- Mbox, RFC822, Outlook PST, Outlook MSG, TNEF
- DWG CAD
- DIF, GDAL, ISO-19139, Grib, HDF, ISA-Tab, NetCDF, Matlab
- Executables (Windows, Linux, Mac)
- Pkcs7, Time Stamp Data Envelope TSD
- SQLite, dBase DBF
- Microsoft Access
What if you don't have a text file, but instead a photo of some text? Or a scan of some text?
OCR (Optical Character Recognition) to the rescue!

- Tesseract is an Open Source OCR tool
- Tika has a parser which can use Tesseract for found images
- Tesseract is detected, and used if found on your path
- Explicit path can be given, or can be disabled
- TODO: Better combining of OCR + normal, or eg PDF only
Container Formats
Databases
Databases

• A surprising number of Database and “database” systems have a single-file mode
• If there's a single file, and a suitable library or program, then Tika can get the data out!
• Main ones so far are MS Access & SQLite
• Panama Papers dump may inspire some more!

• How best to represent the contents in XHTML?
• One HTML table per Database Table best we have, so far...
Tika Config XML
Tika Config XML

• Using Config, you can specify what to use for:
  Parsers, Detectors, Translator, Service Loader + Warnings / Errors, Encoding Detectors, Mime Types
• You can do it explicitly
• You can do it implicitly (with defaults)
• You can do “default except”

• Tools available to dump out a running config as XML
• Use the Tika App to see what you have + save it
Tika Config XML example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<properties>
  <parsers>
    <parser class="org.apache.tika.parser.DefaultParser">
      <mime-exclude>image/jpeg</mime-exclude>
      <mime-exclude>application/pdf</mime-exclude>
    </parser>
    <parser class="org.apache.tika.parser.EmptyParser">
      <mime>application/pdf</mime>
    </parser>
  </parsers>
</properties>
```
Embedded Resources
Tika App
Tika Server
OSGi
Tika Batch
Tika Batch

- Easy way to run Tika against a very large number of documents, for testing and for bulk ingestion
- Multi-threaded, but not yet Hadoop enabled, see https://wiki.apache.org/tika/TikaInHadoop for more there
- Output Text or XHTML, metadata, optionally embedded
- Records failures too, so you know where things go wrong
- Sets up parent/child processes to robustly handle permanent hangs/OOMs
- Optionally restart child every x mins to mitigate memory leaks.
Tika Batch

- Runs local directory to local directory, system agnostic
- Output can be then imported into other systems
- For ingesting, record common failures, import from directory
- Or... For testing, import into Tika Eval

```
java -jar tika-app.jar -i <input_directory> -o <output_directory>
```

https://wiki.apache.org/tika/TikaBatchUsage
Named Entity Recognition
Grobid – Scientific Papers
Grobid - Scientific Papers

- Grobid - GeneRation Of Bibliographic Data
- NLP + NER + Machine Learning
- Tool to identify metadata from scientific / technical papers, based on the textual content contained within
- Works out what sections of text are, then maps to metadata
- Grobid dataset a little big, so Tika doesn’t include as standard, instead calls out to it via REST if configured
Geo Entity Lookup
Geo Entity Lookup

- Augmenting “This was written in Seville, Spain in November” with details of where that is (lat, long, country etc)

- Apache Lucene Gazetter provides fast lookup of place names to geographic details
- Geonames.org dataset used to feed Gazetter
- Apache OpenNLP identifies places in text to lookup
- Needs custom NLP model for place name identification
- GeoTopicParser saves results as metadata, best & alternate
Image Object Recognition
“Text Searchable Video”
Text Searchable Video

- Pooled Time Series Analysis
- Allows you to find “similar” videos
- Search for videos based on features of stills
Apache cTAKES
Apache Camel Integration

- Allows Parsing and Detection, from 2.19.0 onwards

```java
// Parsing a directory
from("file:C:\docs\test")
  .to("tika:parse")
  .to("log:org.apache.tika?showHeaders=true");

// Detection on a directory
from("file:C:\docs\test")
  .to("tika:detect")
  .to("log:org.apache.tika?showHeaders=true");
```
Translation
Language Detection
Troubleshooting
Troubleshooting

- Finally, we have a troubleshooting guide!
  
  http://wiki.apache.org/tika/Troubleshooting%20Tika

- Covers most of the major queries
- Why wasn’t the right parser used
- Why didn’t detection work
- What parsers do I really have etc!
Parser Errors

- As well as the troubleshooting guide, for users...
  http://wiki.apache.org/tika/Troubleshooting%20Tika

- We also have the “Errors and Exceptions” page, aimed more at people writing parsers

- Tries to explain what a parser should be doing in various problem situations, what exceptions to give etc
  http://wiki.apache.org/tika/ErrorsAndExceptions
What's New & Coming Soon?
Tika 1.12

- More consistent and better HTML between PPT and PPTX
- NamedEntity Parser, using both OpenNLP and Stanford NER, outputting text and metadata
- GeoTopic Parser speedup via using new Lucene Geo Gazetteer REST server
- Pooled Time Series parser for video - motion properties from videos to text to allow comparisons
- Bug fixes
• Lots of library upgrades – Apache POI, Apache PDFBox 2.0, Apache SIS and half a dozen others!
• Lots of new mimetypes and magic patterns, especially for scientific-related formats
• NamedEntity Parser add support for Python NLTK and MIT-NLP (MITRE)
• Tika Config XML dumping moved to core, and the app can now dump your running config for you
• Language Detectors more easily pluggable
• Bug fixes
Tika 1.14

- Embedded Document improvements and Macro extraction for MS Office formats
- Tensorflow integration for image object identification
- Tesseract OCR improvements (hOCR, full-page PDF)
- Quite a few more mime types and magics
- More library upgrades
- Re-enable fileUrl feature for Tika Server, has to be turned on manually, gives warnings about security effects!
Apache Tika 1.15+
Tika 1.15+

- Additional JPEG formats support (JPX, JP2)
- PDFBox 2.0 further updates
- Several new older MS Office format variants supported
- Word Perfect, WMF, EMF
- Language Detector improvements – N-Gram, Optimaize Lang Detector, MIT Text.jl, pluggable and pickable
- More NLP enhancement / augmentation
- Metadata aliasing
- Plus preparations for Tika 2
Image, Video, NER

- Image recognition using Tensorflow:
  https://wiki.apache.org/tika/TikaAndVision / Paper:
- Image Recognition using Deeplearning4j:
  https://wiki.apache.org/tika/TikaAndVisionDL4J
- Sentiment Analysis using OpenNLP:
  https://github.com/apache/tika/pull/169
- Video labeling using tensorflow image rec:
  https://wiki.apache.org/tika/TikaAndVisionVideo
- Named Entity Extraction using OpenNLP and CoreNLP:
  https://wiki.apache.org/tika/TikaAndNER
- Image Captioning (Image-to-Text)
  https://github.com/apache/tika/pull/180
Tika 2.0
Why no Tika v2 yet?

- Apache Tika 0.1 – December 2007
- Apache Tika 1.0 – November 2011
- Shouldn't we have had a v2 by now?

- Discussions started several years ago, on the list
- Plans for what we need on the wiki for ~1 year

- Largely though, every time someone came up with a breaking feature for 2.0, a compatible way to do it was found!
Deprecated Parts

- Various parts of Tika have been deprecated over the years
- All of those will go!

- Main ones that might bite you:
  - Parser parse with no ParseContext
  - Old style Metadata keys
Currently, Metadata in Tika is String Key/Value Lists
Many Metadata types have Properties, which provide typing, conversions, sanity checks etc
But all still stored as String Key + Value(s)

Some people think we need a richer storage model
Others want to keep it simple!
JSON, XML DOM, XMP being debated
Richer string keys also proposed
Metadata for Video etc

• Video file might have 2 video streams, 4 audio streams, a metadata stream and some subtitles
• Some of those you want to treat as embedded resources
• Some of those “belong” together

• How should we return the number of channels for the 1st audio stream in a video?
• Should it change if there’s one or many?
Java Packaging of Tika

- Maven Packages of Tika are
  - Tika Core
  - Tika Parsers
  - Tika Bundle
  - Tika XMP
  - Tika Java 7
- For just some parsers, in Tika 1.x, you need to exclude maven dependencies + re-test
- In Tika 2, more fine-grained parser collections
Tika 2.x Parser Sets

- Available today in Git on the 2.x branch

- Advanced
- CAD
- Code
- Crypto

- Database
- eBook
- Journal

- Mail
- Multimedia
- Office

- Package
- PDF
- Scientific

- Text
- Web
- XMP-Commons

- May change some more, but broadly in place now
Logging, Config, Defaults

- Logging – Moving to SLF4J
- Aim is to have all of Tika use that, and parsers configure that in for the libraries they call

- Config – ensure everything can be configured, and configured easily
- Consistent Configuration – all in one place, common format

- Defaults – Sensible, Documented, No Surprises
Fallback/Preference Parsers

- If we have several parsers that can handle a format
- Preferences?
- If one fails, how about trying others?
Multiple Parsers

- If we have several parsers that can handle a format
- What about running all of them?
  - eg extract image metadata
  - then OCR it
  - then try the regular image parser for more metadata
- Or maybe for calling multiple different NER parsers
• Currently, Tika uses a Service Loader mechanism to find and load available Parsers (and Detectors+Translators)
• This allows you to drop a new Tika parser jar onto the classpath, and have it automatically used
• Also allows you to miss one or two jars out, and not get any content back with no warnings / errors...
• You can set the Service Loader to Warn, or even Error
• But most people don't, and it bites them!
• Change the default in 2? Or change entirely how we do it?
What we still need help with...
Tika uses the SAX Content Handler interface for supplying plain text along with semantically meaningful XHTML

Streaming, write once

How does that work with multiple parsers?
  • How about if one parser fails and we want to try parsing with a different one?
  • How about if one parser works, then you want to run a second?

Language Detection / NER – how to mark up previous text?
Content Enhancement

• How can we post-process the content to “enhance” it in various ways?
• For example, how can we mark up parts of speech?
• Pull out information into the Metadata?
• Translate it, retaining the original positions?
• For just some formats, or for all?
• For just some documents in some formats?
• While still keeping the Streaming SAX-like contract?
Metadata Standards

• Currently, Tika works hard to map file-format-specific metadata onto general metadata standards.
• Means you don’t have to know each standard in depth, can just say “give me the closest to dc:subject you have, no matter what file format or library it comes from.”

• What about non-File-format metadata, such as content metadata (Table of Contents, Author information etc)?
• What about combining things?
Richer Metadata

• See Metadata Storage slides!
Bonus! Apache Tika at Scale
Lots of Data is Junk

- At scale, you're going to hit lots of edge cases
- At scale, you're going to come across lots of junk or corrupted documents
- 1% of a lot is still a lot...
- 1% of the internet is a huge amount!
- Bound to find files which are unusual or corrupted enough to be mis-identified
- You need to plan for failures!
Unusual Types

- If you're working on a big data scale, you're bound to come across lots of valid but unusual + unknown files.
- You're never going to be able to add support for all of them!
- May be worth adding support for the more common “uncommon” unsupported types.
- Which means you'll need to track something about the files you couldn't understand.
- If Tika knows the mimetype but has no parser, just log the mimetype.
- If mimetype unknown, maybe log first few bytes.
Failure at Scale

• Tika will sometimes mis-identify something, so sometimes the wrong parser will run and object
• Some files will cause parsers or their underlying libraries to do something silly, such as use lots of memory or get into loops with lots to do
• Some files will cause parsers or their underlying libraries to OOM, or infinite loop, or something else bad
• If a file fails once, will probably fail again, so blindly just re-running that task again won't help
Failure at Scale, continued

- You'll need approaches that plan for failure
- Consider what will happen if a file locks up your JVM, or kills it with an OOM
- Forked Parser may be worth using
- Running a separate Tika Server could be good
- Depending on work needed, could have a smaller pool of Tika Server instances for big data code to call
- Think about failure modes, then think about retries (or not)
- Track common problems, report and fix them!
Tika Batch, Eval & Hadoop
Tomorrow – 2.40pm, Brickell!
Any Questions?