Getting Started with Apache Tika

This document describes how to build Apache Tika from sources and how to start using Tika in an application.

Getting and building the sources

To build Tika from sources you first need to either download a source release or checkout the latest sources from version control.

Once you have the sources, you can build them using the Maven 2 build system. Executing the following command in the base directory will build the sources and install the resulting artifacts in your local Maven repository.

```
mvn install
```

See the Maven documentation for more information about the available build options.

Note that you need Java 7 or higher to build Tika.

Build artifacts

The Tika build consists of a number of components and produces the following main binaries:

Apache Tika

Introduction
Download
Contribute
Mailing Lists
Tika Wiki
Issue Tracker

Documentation

- Apache Tika 1.16
  - Getting Started
  - Supported Formats
  - Parser API
  - Parser 5min Quick Start Guide
  - Content and Language Detection
  - Configuring Tika
  - Usage Examples
  - API Documentation
  - REST API Documentation (Miredot)
- Apache Tika 1.15
- Apache Tika 1.14
- Apache Tika 1.13
- Apache Tika 1.12
- Apache Tika 1.11
- Apache Tika 1.10

The Apache Software Foundation

About

tika-core/target/tika-core-*.jar

Tika core library. Contains the core interfaces and classes of Tika, but none of the parser implementations. Depends only on Java 6.

tika-parsers/target/tika-parsers-*.jar

Tika parsers. Collection of classes that implement the Tika Parser interface based on various external parser libraries.

tika-app/target/tika-app-*.jar

Tika application. Combines the above components and all the external parser libraries into a single runnable jar with a GUI and a command line interface.

tika-server/target/tika-server-*.jar

Tika JAX-RS REST application. This is a Jetty web server running Tika REST services as described in this page .

tika-bundle/target/tika-bundle-*.jar

Tika bundle. An OSGi bundle that combines tika-parsers with non-OSGified parser libraries to make them easy to deploy in an OSGi environment.

Using Tika as a Maven dependency

The core library, `tika-core` , contains the key interfaces and classes of Tika and can be used by itself if you don't need the full set of parsers from the `tika-parsers` component. The tika-core dependency looks like this:

```
<dependency>
  <groupId>org.apache.tika</groupId>
  <artifactId>tika-core</artifactId>
  <version>1.16</version>
</dependency>
```

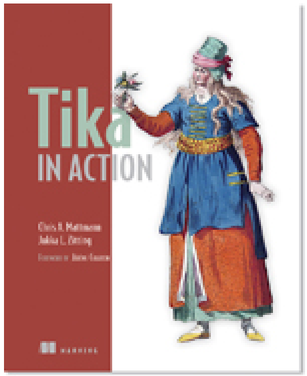
If you want to use Tika to parse documents (instead of simply detecting document types, etc.), you'll want to depend on `tika-parsers` instead:

License
Security
Sponsorship
Thanks

Search with Apache Solr

provide ▼ Search

Books about Tika



```
  <dependency>
    <groupId>org.apache.tika</groupId>
    <artifactId>tika-parsers</artifactId>
    <version>1.16</version>
  </dependency>
```

Note that adding this dependency will introduce a number of transitive dependencies to your project, including one on tika-core. You need to make sure that these dependencies won't conflict with your existing project dependencies. You can use the following command in the tika-parsers directory to get a full listing of all the dependencies.

```
$ mvn dependency:tree | grep :compile
```

Using Tika in a Gradle-built project

To add a dependency on Apache Tika to your Gradle built project, including the full set of parsers, you should depend on the `tika-parsers` artifact:

```
dependencies {
    runtime 'org.apache.tika:tika-parsers:1.16'
}
```

Using Tika in an Ant project

If you are using [Apache Ivy](#) as your dependency manager tool with Ant, then to include Tika with the full set of parsers, you should depend on the `tika-parsers` artifact like this:

```
    <dependencies>
        <dependency org="org.apache.tika" name="tika-parsers" rev
    </dependencies>
```

Otherwise, probably the easiest way to use Tika is to include the full `tika-app` jar on your classpath. For just core functionality, you can add the `tika-core` jar, but be aware that the full set of parsers have a large number of dependencies which must be included which is very fiddly to do by hand with Ant! To include Tika in your Ant project, you should do something like:

```
<classpath>
  ... <!-- your other classpath entries -->

  <!-- either: Tika Core only, no parsers -->
  <pathelement location="path/to/tika-core-${tika.version}.jar"/>
  <!-- or: Tika with all Parsers-->
  <pathelement location="path/to/tika-app-${tika.version}.jar"/>

</classpath>
```

Using Tika as a command line utility

The Tika application jar (tika-app-*.jar) can be used as a command line utility for extracting text content and metadata from all sorts of files. This runnable jar contains all the dependencies it needs, so you don't need to worry about classpath settings to run it.

The usage instructions are shown below.

```
usage: java -jar tika-app.jar [option...] [file|port...]

Options:
    -?  or --help          Print this usage message
    -v  or --verbose       Print debug level messages
    -V  or --version       Print the Apache Tika version number
    -g  or --gui           Start the Apache Tika GUI
    -s  or --server        Start the Apache Tika server
    -f  or --fork          Use Fork Mode for out-of-process extra

    -x  or --xml           Output XHTML content (default)
    -h  or --html          Output HTML content
    -t  or --text          Output plain text content
    -T  or --text-main     Output plain text content (main conter
    -m  or --metadata      Output only metadata
    -j  or --json          Output metadata in JSON
    -y  or --xmp           Output metadata in XMP
    -l  or --language      Output only language
    -d  or --detect        Detect document type
    -eX or --encoding=X    Use output encoding X
    -pX or --password=X    Use document password X
    -z  or --extract       Extract all attachements into current
    --extract-dir=<dir>    Specify target directory for -z
    -r  or --pretty-print  For XML and XHTML outputs, adds newlir
                           whitespace, for better readability

    --create-profile=X
         Create NGram profile, where X is a profile name
    --list-parsers
         List the available document parsers
    --list-parser-details
         List the available document parsers, and their supportec
    --list-detectors
         List the available document detectors
    --list-met-models
         List the available metadata models, and their supported
    --list-supported-types
         List all known media types and related information
```

```
Description:
    Apache Tika will parse the file(s) specified on the
    command line and output the extracted text content
    or metadata to standard output.

    Instead of a file name you can also specify the URL
    of a document to be parsed.

    If no file name or URL is specified (or the special
    name "-" is used), then the standard input stream
    is parsed. If no arguments were given and no input
    data is available, the GUI is started instead.

- GUI mode

    Use the "--gui" (or "-g") option to start the
    Apache Tika GUI. You can drag and drop files from
    a normal file explorer to the GUI window to extract
    text content and metadata from the files.

- Server mode

    Use the "--server" (or "-s") option to start the
    Apache Tika server. The server will listen to the
    ports you specify as one or more arguments.
```

You can also use the jar as a component in a Unix pipeline or as an external tool in many scripting languages.

```
# Check if an Internet resource contains a specific keyword
curl http://.../document.doc \
  | java -jar tika-app.jar --text \
  | grep -q keyword
```

Wrappers

Several wrappers are available to use Tika in another programming language, such as Julia or Python .