SimpleMagic Package

Version 1.17
December 2021

Gray Watson

Table of Contents

SimpleMagic

Version 1.17 – December 2021

This package provides some simple magic value features that simulate the Unix file(1) command to determine the type of a file or of `bytes` from the content. It has an internal set of magic number information or it can process the magic files from local ~Unix system configuration.

To get started quickly using SimpleMagic, see Chapter 1 [Quick Start], page 2. There is also a HTML version of this documentation.

Gray Watson http://256stuff.com/gray/

1 Start Using Quickly

To use SimpleMagic you need to do the following steps. For more information, see Chapter 2 [Using], page 3.

1. Download SimpleMagic from the SimpleMagic release page. See Section 2.1 [Downloading], page 3.
2. Optionally load in the magic entries from local file(s). By default, if you construct a `ContentInfoUtil` instance with the default constructor, it will load the internal magic entries file. See Section 2.2 [Loading Magic Entries], page 3.
3. Use the `ContentInfoUtil` class to get content-types for files or `byte[]`:

```
ContentInfoUtil util = new ContentInfoUtil();
ContentInfo info = util.findMatch("/tmp/upload.tmp");
// or   ContentInfo info = util.findMatch(inputStream);
// or   ContentInfo info = util.findMatch(contentByteArray);
// display content type information
if (info == null) {
   System.out.println("Unknown content-type");
} else {
   // other information in ContentInfo type
   System.out.println("Content-type is: " + info.getName());
}
```

If the `findMatch(...)` method does not recognize the content then it will return null. If it does match one of the entries then it will return a `ContentInfo` class which provides:

- Enumerated type if the type is common otherwise set to `OTHER`
- Approximate content-name
- Full message produced by the magic file
- Mime-type string if one configured by the config file
- Associated file extensions if any associated

Here are some examples of `ContentInfo` output:

- HTML, mime 'text/html', msg 'HTML document text'
- PDF, mime 'application/pdf', msg 'PDF document, version 1.4'
- GIF, mime 'image/gif', msg 'GIF image data, version 89a, 16 x 16'
- JPEG, mime 'image/jpeg', msg 'JPEG image data, JFIF standard 1.01'
- Java, msg 'Java serialization data, version 5'

For somewhat more extensive instructions, see Chapter 2 [Using], page 3.

2 Using SimpleMagic

2.1 Downloading Jar

To get started with SimpleMagic, you will need to download the jar file. The SimpleMagic release page is the default repository but the jars are also available from the central maven repository.

The code works with Java 6 or later.

2.2 How To Load Magic Entries

The library uses various magic byte information to be able to find and determine details about random blocks of bytes. By default, SimpleMagic has a built in version of a magic file that was copied from a CentOS Linux system. It contains, ~2400 magic file entries describing a number of different file types. It also has an additional ~6600 lines which provide more details about the detected content types.

The magic entries are relatively complex but in general look something like the following. The configuration line says to look at the start of the file for the string `"GIF8"`. If it is there then the file is "GIF image data".

```
0       string          GIF8            GIF image data
```

If you do not want to use the internal magic definitions, you can also construct the `ContentInfoUtil` class with a file or directory to have it parse and use another definition file.

```
ContentInfoUtil util = new ContentInfoUtil("/etc/magic");
```

WARNING: although we've tried to support different types of magic entries, there are local per-OS variations that may not be supported.

2.3 How To Find The Content Info

Once you have loaded the magic entry information into your `ContentInfoUtil`, you can use the utility class to find the content info of files, byte arrays, or `InputStreams`. The base method gets content info information from a `byte[]`.

```
byte[] uploadedBytes = ...;
ContentInfo info = util.findMatch(uploadedBytes);
```

You can also get the content type of a file which is read with a `FileInputStream`:

```
ContentInfo info = util.findMatch("/tmp/uploadedFile.tmp");
// File uploadedFile = ...
// ContentInfo info = util.findMatch(uploadedFile);
```

If you have an `InputStream`, you can also use it directly:

```
InputStream inputStream = ...
ContentInfo info = util.findMatch(inputStream);
```

If you want to process a stream of bytes as the bytes are being read, you can use the `ContentInfoInputStreamWrapper` utility class. This takes an `InputStream` which it wraps and delegates to. After you have read the bytes through the wrapper, you can call the `findMatch()` method to get its content information.

```
HttpServletRequest request = ...
ContentInfoInputStreamWrapper inputStream
   = new ContentInfoInputStreamWrapper(request.getInputStream());
// read in the file from the http request, ...
// after we have read it in, we can get its content-info
ContentInfo info = inputStream.findMatch();
```

For the file and stream versions, the first 10 kilobytes of the data is read and processed.

There is also a long internal list of file types copied from the Apache list. Not all of the files in this list have associated magic number information. However, with the list you can look up mime-types or by file-extension and get the associated information.

You can use the internal list to lookup by file-extension:

```
// find details about files with .pdf extension
ContentInfo info =
   ContentInfoUtil.findExtensionMatch("file.pdf");
// you can even just pass in the extension name
ContentInfo info =
   ContentInfoUtil.findExtensionMatch("DOC");
```

Or you can look up by mime-type:

```
// find details about this mime-type
ContentInfo info =
   ContentInfoUtil.findMimeTypeMatch("image/vnd.dwg");
```

Some internal entries provide more information than others. This list is a work in progress. Please submit improvements and edits as necessary.

2.4 Content Information

If the `findMatch(...)` method does not recognize the content then it will return null. If it does match one of the entries then it will return a `ContentInfo` class which provides:

- Enumerated content-type if the type is common otherwise set to `OTHER`. This is determined by mapping the mime-type string to an internal enumerated type and is not determined from the magic file entries.
- Approximate content-name. If the content-type is known then this will be a constant string. If not know then this is usually the first word of the full message from the magic file.
- Details about the content produced by the magic file.
- Mime-type string if one configured by the config file.
- Associated file-extensions if in the internal list.

Here are some examples of `ContentInfo` output:

- html, mime 'text/html', msg 'HTML document text'

- java, msg 'Java serialization data, version 5'
- pdf, mime 'application/pdf', msg 'PDF document, version 1.4'
- gzip, mime 'application/x-gzip', msg 'gzip compressed data, was "", from Unix...'
- gif, mime 'image/gif', msg 'GIF image data, version 89a, 16 x 16'
- png, mime 'image/png', msg 'PNG image, 600 x 371, 8-bit/color RGB, non-interlaced'
- mp4a, mime 'audio/mp4', msg 'ISO Media, MPEG v4 system, iTunes AAC-LC'
- word, mime 'application/msword', msg 'Microsoft Word Document'
- wav, mime 'audio/x-wav', msg 'RIFF (little-endian) data, WAVE audio...'
- jpeg, mime 'image/jpeg', msg 'JPEG image data, JFIF standard 1.01'

2.5 Using With Maven

To use SimpleMagic with maven, include the following dependency in your '`pom.xml`' file:

```
<dependency>
<groupId>com.j256.simplemagic</groupId>
<artifactId>simplemagic</artifactId>
<version>1.17</version>
</dependency>
```

3 Open Source License

Index of Concepts