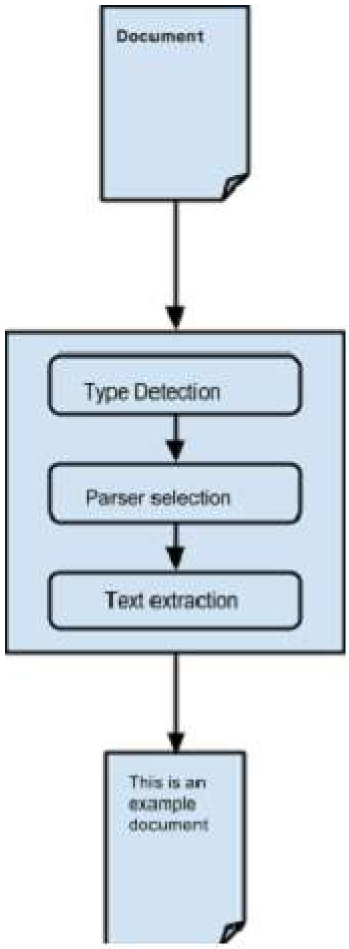
TIKA - CONTENT EXTRACTION

Tika uses various parser libraries to extract content from given parsers. It chooses the right parser for extracting the given document type.

For parsing documents, the parseToString method of Tika facade class is generally used. Shown below are the steps involved in the parsing process and these are abstracted by the Tika ParsertoString method.



Abstracting the parsing process:

- Initially when we pass a document to Tika, it uses a suitable type detection mechanism available with it and detects the document type.
- Once the document type is known, it chooses a suitable parser from its parser repository. The parser repository contains classes that make use of external libraries.
- Then the document is passed to choose the parser which will parse the content, extract the text, and also throw exceptions for unreadable formats.

Content Extraction using Tika

Given below is the program for extracting text from a file using Tika facade class:

```
import java.io.File;
import java.io.IOException;

import org.apache.tika.Tika;
import org.apache.tika.exception.TikaException;

import org.xml.sax.SAXException;
```

```
public class TikaExtraction {

   public static void main(final String[] args) throws IOException, TikaException {

      //Assume sample.txt is in your current directory
      File file = new File("sample.txt");

      //Instantiating Tika facade class
      Tika tika = new Tika();
      String filecontent = tika.parseToString(file);
      System.out.println("Extracted Content: " + filecontent);
   }
}
```

Save the above code as TikaExtraction.java and run it from the command prompt:

```
javac TikaExtraction.java
java TikaExtraction
```

Note : Assume sample.txt is having the following content.

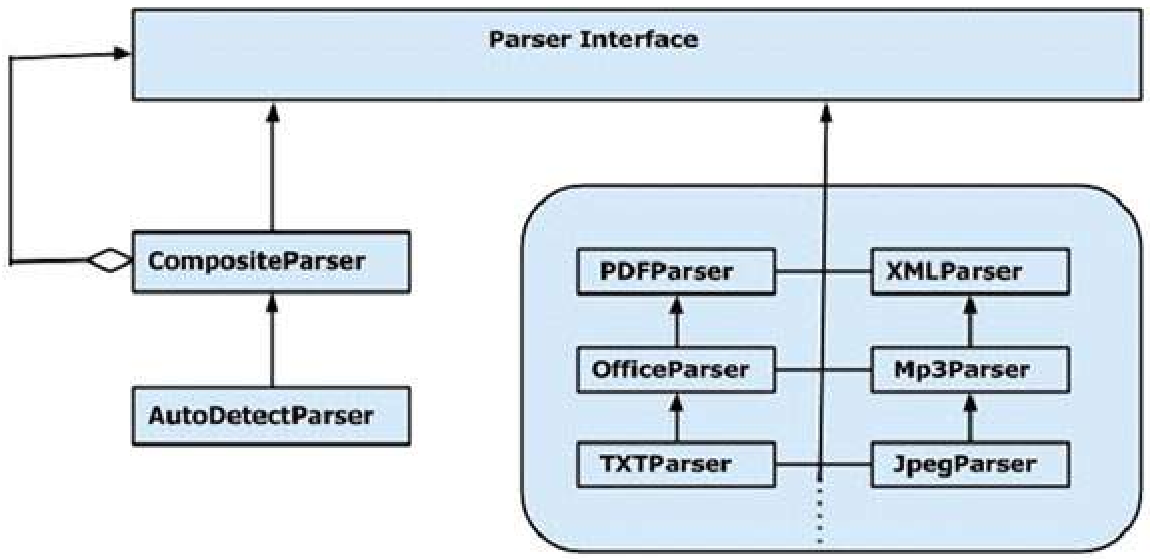
```
Hi students welcome to tutorialspoint
```

It gives you the following output:

```
Extracted Content: Hi students welcome to tutorialspoint
```

Content Extraction using Parser Interface

The parser package of Tika provides several interfaces and classes using which we can parse a text document. Given below is the block diagram of the org.apache.tika.parser package.



There are several parser classes available, e.g., pdf parser, Mp3Passer, OfficeParser, etc., to parse respective documents individually. All these classes implement the parser interface.

CompositeParser

The given diagram shows Tika’s general-purpose parser classes **CompositeParser** and **AutoDetectParser**. Since the CompositeParser class follows composite design pattern, you can use a group of parser instances as a single parser. The CompositeParser class also allows access to all the classes that implement the parser interface.

AutoDetectParser

This is a subclass of CompositeParser and it provides automatic type detection. Using this functionality, the AutoDetectParser automatically sends the incoming documents to the appropriate parser classes using the composite methodology.

parse method

Along with parseToString, you can also use the parse method of the parser Interface. The prototype of this method is shown below.

```
parse(InputStream stream, ContentHandler handler, Metadata metadata,
ParseContext context)
```

The following table lists the four objects it accepts as parameters.

S.No.	Object and Description
1	**InputStream stream** Any Inputstream object that contains the content of the file
2	**ContentHandler handler** Tika passes the document as XHTML content to this handler, thereafter the document is processed using SAX API. It provides efficient post-processing of the contents in a document.
3	**Metadata metadata** The metadata object is used both as a source and a target of document metadata.
4	**ParseContext context** This object is used in cases where the client application wants to customize the parsing process.

Example:

Given below is an example that shows how the parse method is used.

Step 1:

To use the parse method of the parser interface, instantiate any of the classes providing the implementation for this interface.

There are individual parser classes such as PDFParser, OfficeParser, XMLParser, etc. You can use any of these individual document parsers. Alternatively, you can use either CompositeParser or AutoDetectParser that uses all the parser classes internally and extracts the contents of a document using a suitable parser.

```
Parser parser = new AutoDetectParser();
   (or)
Parser parser = new CompositeParser();
   (or)
object of any individual parsers given in Tika Library
```

Step 2:

Create a handler class object. Given below are the three content handlers:

S.No.	Class and Description
1	**BodyContentHandler** This class picks the body part of the XHTML output and writes that content to the output writer or output stream. Then it redirects the XHTML content to another content handler instance.
2	**LinkContentHandler** This class detects and picks all the H-Ref tags of the XHTML document and forwards those for the use of tools like web crawlers.
3	**TeeContentHandler** This class helps in using multiple tools simultaneously.

Since our target is to extract the text contents from a document, instantiate BodyContentHandler as shown below:

```
BodyContentHandler handler = new BodyContentHandler( );
```

Step 3:

Create the Metadata object as shown below:

```
Metadata metadata = new Metadata();
```

Step 4:

Create any of the input stream objects, and pass your file that should be extracted to it.

FileInputstream

Instantiate a file object by passing the file path as parameter and pass this object to the FileInputStream class constructor.

Note : The path passed to the file object should not contain spaces.

The problem with these input stream classes is that they don't support random access reads, which is required to process some file formats efficiently. To resolve this problem, Tika provides TikaInputStream.

```
File file=new File(filepath)
FileInputStream inputstream=new FileInputStream(file);
   (or)
InputStream stream = TikaInputStream.get(new File(filename));
```

Step 5:

Create a parse context object as shown below:

```
ParseContext context =new ParseContext();
```

Step 6:

Instantiate the parser object, invoke the parse method, and pass all the objects required, as shown in the prototype below:

```
parser.parse(inputstream, handler, metadata, context);
```

Given below is the program for content extraction using the parser interface:

```
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;

import org.apache.tika.exception.TikaException;
import org.apache.tika.metadata.Metadata;
import org.apache.tika.parser.AutoDetectParser;
import org.apache.tika.parser.ParseContext;
import org.apache.tika.parser.Parser;
import org.apache.tika.sax.BodyContentHandler;

import org.xml.sax.SAXException;

public class ParserExtraction {

public static void main(final String[] args) throws IOException,SAXException,
TikaException {

   //Assume sample.txt is in your current directory
   File file = new File("sample.txt");

   //parse method parameters
   Parser parser = new AutoDetectParser();
   BodyContentHandler handler = new BodyContentHandler();
   Metadata metadata = new Metadata();
   FileInputStream inputstream = new FileInputStream(file);
   ParseContext context = new ParseContext();

   //parsing the file
   parser.parse(inputstream, handler, metadata, context);
   System.out.println("File content : " + Handler.toString());
   }
}
```

Save the above code as ParserExtraction.java and run it from the command prompt:

```
javac  ParserExtraction.java
java  ParserExtraction
```

Assumre sample.txt contains the following content.

```
Hi students welcome to tutorialspoint
```

It gives the following output:

```
File content : Hi students welcome to tutorialspoint
```