File analyzer component packages

Edit New Page

Terry Brady edited this page 4 days ago · 25 revisions

The File Analyzer code base is broken into multiple modules containing different levels of functionality.

Core Package - requires only Java

The Core package contains File Test Rules and File Import Rules

- with general applicability to multiple institutions
- that do not depend on libraries other than the core java libraries

Core Package Contents

Demo Package - integrated with Apache Tika (for metadata extraction) and BagIt

The Demo package contains File Test Rules and File Import Rules

- that depend on external libraries
- may not have applicability to multiple institutions
- demonstrate how to customize Core code to implement institution-specific business logic

Demo Package Contents

DSpace Package - automation of DSpace ingestion tasks

Georgetown University has successfully automated a number of DSpace ingest tasks using the File Analyzer.

See DSpace Institutional Repository Ingest for an illustration of these tools

DSpace Tools Overview Presentations

- Why Bulk Ingest Automation is Needed (Prezi)
- Running the Bulk Ingest Process (Prezi)
- Open Repositories Presentation Describing this Process

MARC File Analyzer

Georgetown University has created a version of the File Analyzer that reads and analyzes MARC records: MARC File Analyzer. This build of this component is not fully automated with Maven. The build requires a manual download of the Marc4j library.

MARC File Analyzer Package Contents

Pages 46

- Home
- File Analyzer Component Packages
- Installation instructions
- File Analyzer Stories
- File Analyzer Use Cases at Georgetown University
- Latest Features
- User Interface Overview
- Command Line Interface
- Batch Processing
- Coding new File Test Rules and new File Import Rules
- File-Analyzer-Training-Code4Lib-2015

Clone this wiki locally

https://github.com/Georgetown

Clone in Desktop

Custom Packages

An institution can create their own module containing highly customized rules.

Create Your Own File Analyzer

```
public class GUFileAnalyzer extends DirectoryTable {

public GUFileAnalyzer(File f, boolean modifyAllowed) {
    super(f, modifyAllowed);
    this.title = "Georgetown University Libraries File Analyzer";
    this.message = "File Analyzer customized for use by the Georgetown University Li
    this.refreshTitle();
}

protected ActionRegistry getActionRegistry() {
    return new GUActionRegistry(this, modifyAllowed);
}

protected ImporterRegistry getImporterRegistry() {
    return new GUImporterRegistry(this);
}
public static void main(String[] args) {
    if (args.length > 0)
        new GUFileAnalyzer(new File(args[0]), false);
    else
        new GUFileAnalyzer(null, false);
}

}
```

Create an ActionRegistry to register your custom File Test Rule classes (and to remove default ones)

```
public class GUActionRegistry extends DemoActionRegistry {

private static final long serialVersionUID = 1L;

public GUActionRegistry(FTDriver dt, boolean modifyAllowed) {
    super(dt, modifyAllowed);

    removeFT(IngestInventory.class);
    removeFT(IngestValidate.class);
    add(new GUIngestInventory(dt));
    add(new GUIngestValidate(dt));
}

}
```

Create an ImporterRegistry to register your Custom File Import Rules

```
public class GUImporterRegistry extends DemoImporterRegistry {

private static final long serialVersionUID = 1L;

public GUImporterRegistry(FTDriver dt) {
    super(dt);
    add(new OutputToBursar(dt));
}
}
```

+ Add a custom footer