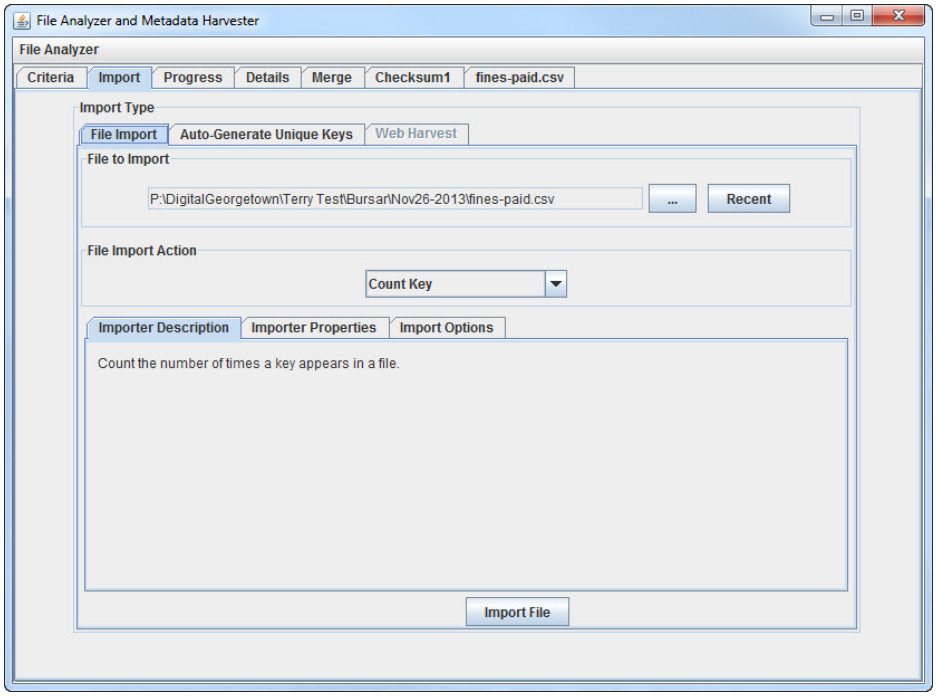
File Import rule

Terry Brady edited this page on Aug 18, 2014 · 10 revisions

File Import Rules act on a single file at one time. A File Import Rule generally reports on the status of each record that is found within a given file.

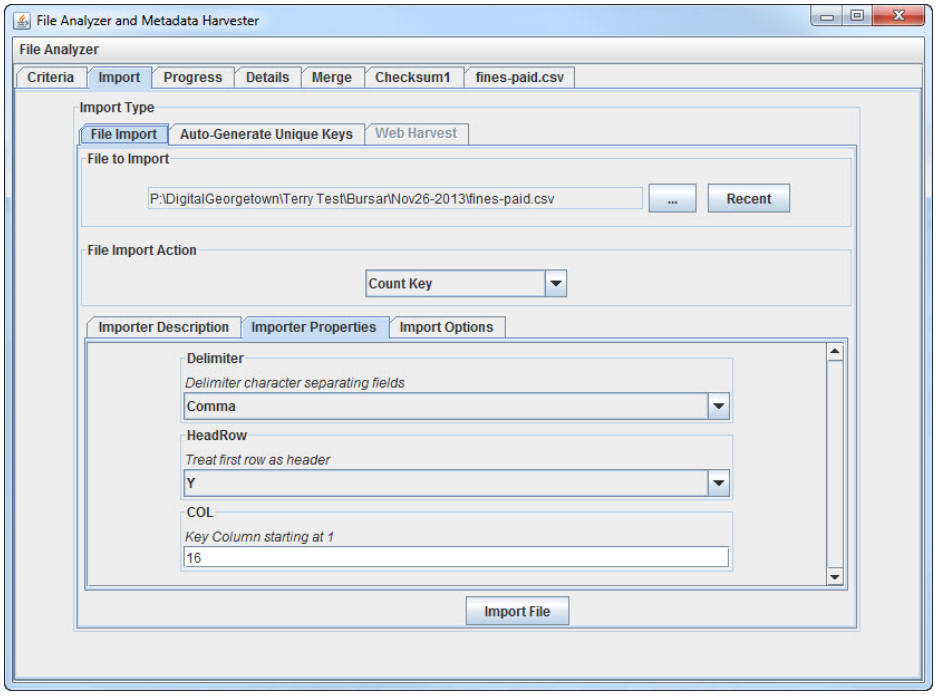
Components of a File Import Rule

Name and description: explains the File Import Rule to a user



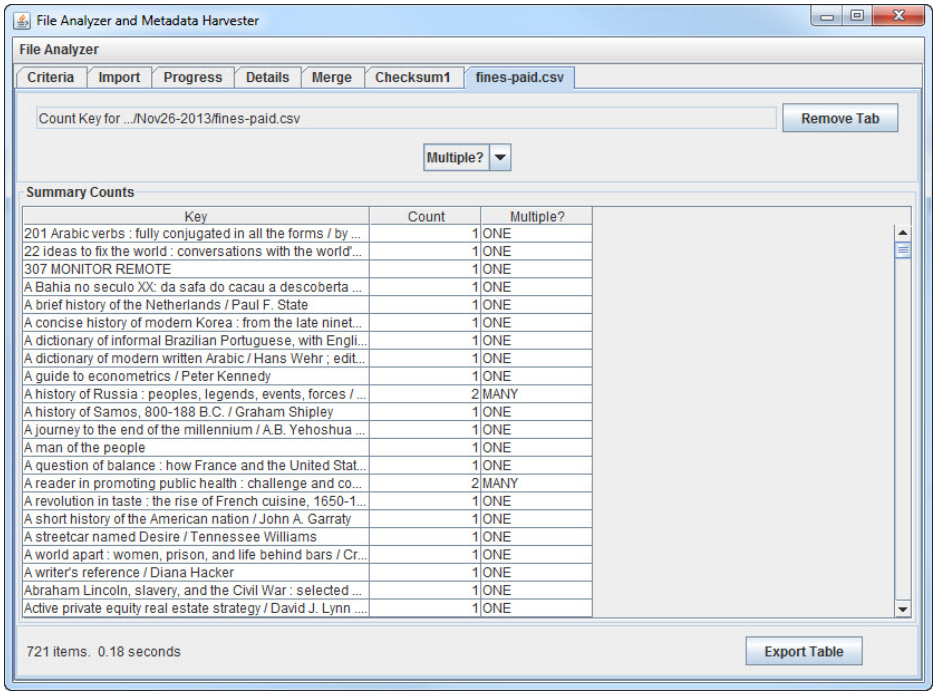

```
public classclass CountKey extends DefaultImporter {

public String toString() {
    return "Count Key";
}
public String getDescription() {
    return "Count the number of times a key appears in a file.";
}
public String getShortName() {
    return "Key";
}
```

Properties: runtime parameters that the user can pass to the File Import Rule




```
public static final String DELIM = "Delimiter";
public static final String HEADROW = "HeadRow";
public static final String COL = "COL";
public CountKey(FTDriver dt) {
    super(dt);
    this.ftprops.add(new FTPropEnum(dt, this.getClass().getName(), DELIM, "delim",
        "Delimiter character separating fields", Separator.values(), Separator.Comma
    this.ftprops.add(new FTPropEnum(dt, this.getClass().getName(), HEADROW, HEADROW,
        "Treat first row as header", YN.values(), YN.Y));
    this.ftprops.add(new FTPropString(dt, this.getClass().getName(), COL, COL,
        "Key Column starting at 1", "1"));
}
```

Result Stats: defines the resulting information that will be displayed to the user (as a table)




```
public static enum MULT {ONE, MANY;}
private static enum CountStatsItems implements StatsItemEnum {
    Key(StatsItem.makeStringStatsItem("Key", 100)),
    Count(StatsItem.makeIntStatsItem("Count")),
    Stat(StatsItem.makeEnumStatsItem(MULT.class, "Multiple?"))
    ;

    StatsItem si;
    CountStatsItems(StatsItem si) {this.si=si;}
    public StatsItem si() {return si;}
}
public static enum Generator implements StatsGenerator {
    INSTANCE;

    public Stats create(String key) {return new Stats(details, key);}
}

public static StatsItemConfig details = StatsItemConfig.create(CountStatsItems.class
```

Code the File Import Rule

In the example displayed above, a checksum is generated on the file using the algorithm provided by the user.

```
public ActionResult importFile(File selectedFile) throws IOException {

    int col = 0;
    try {
        col = Integer.parseInt(this.getProperty(COL,"").toString());
        col--;
    } catch (NumberFormatException e) {
    }

    Separator fileSeparator = (Separator)getProperty(DELIM);
    Timer timer = new Timer();

    TreeMap<String,Stats> types = new TreeMap<String,Stats>();
```

```
    DelimitedFileReader dfr = new DelimitedFileReader(selectedFile, fileSeparator.se
    boolean firstRow = (YN)getProperty(HEADROW) == YN.Y;

    firstRow = (YN)getProperty(HEADROW) == YN.Y;
    dfr = new DelimitedFileReader(selectedFile, fileSeparator.separator);
    for(Vector<String> cols = dfr.getRow(); cols != null; cols = dfr.getRow()){
        if (firstRow) {
            firstRow = false;
            continue;
        }
        String key = cols.get(col < cols.size() ? col : 0);
        Stats stats = types.get(key);
        if (stats == null) {
            stats = Generator.INSTANCE.create(key);
            stats.setVal(CountStatsItems.Count, 1);
            stats.setVal(CountStatsItems.Stat, MULT.ONE);
            types.put(key, stats);
        } else {
            stats.sumVal(CountStatsItems.Count, 1);
            stats.setVal(CountStatsItems.Stat, MULT.MANY);
        }
    }

    return new ActionResult(selectedFile, selectedFile.getName(), this.toString(), c
}
```

Register the Importer with the File Analyzer

```
public class ImporterRegistry extends Vector<Importer> {

private static final long serialVersionUID = 1L;

public ImporterRegistry(FTDriver dt) {
    ...
    add(new CountKey(dt));
}
```