Table Of Content

Package com.ashley.migration

Class Summary

MigrationTool

com.ashley.migration

Class MigrationTool

```
java.lang.Object
    |
    +--javafx.application.Application
        |
        +--com.ashley.migration.MigrationTool
```

< Constructors > < Methods >

public class **MigrationTool**
extends javafx.application.Application

Constructors

MigrationTool

```
public  MigrationTool()
```

Methods

main

```
public static void main(java.lang.String[] args)
```

start

```
public void start(javafx.stage.Stage stage)
```

Overrides:

start in class javafx.application.Application

Package com.ashley.migration.controller

Class Summary

EloLogin

Elo Login Class,

This class is how the program interacts with ELO Professional It contains methods to Login, create a folder structure, load documents and correct the folder names (if they contain a '/' character originally)

IniReader

IniReader Class In this class, all document and folder information will be read and added to the ExportDocument or Mask Class objects.

InputController

This Class is will handle the information taken from the user and then send it to the relevant models and controllers.

---

com.ashley.migration.controller

Class EloLogin

```
java.lang.Object
    |
    +--com.ashley.migration.controller.EloLogin
```

< Constructors > < Methods >

---

public class **EloLogin**
extends java.lang.Object

Elo Login Class,

This class is how the program interacts with ELO Professional It contains methods to Login, create a folder structure, load documents and correct the folder names (if they contain a '/' character originally)

Author:

Ashley Stonehall

Version:

1.00

Constructors

EloLogin

```
public  EloLogin()
```

Methods

createDocumentsArchive

```
public void createDocumentsArchive(de.elo.ix.client.IXConnection conn,
                                   java.lang.String fileLocation,
                                   java.lang.String archiveDocPath,
                                   java.lang.String docName,
                                   java.lang.String extraText,
                                   java.lang.String maskId,
                                   java.lang.String oldMask,
                                   ExportDocument docObj)
```

Upload documents to the created paths in the ELO Professional Archive

This method will find the correct folder location in ELO and then upload the relevant documents to it with its new Keywording mask and information from the old one.

Parameters:

conn - IXConection, variable
fileLocation - String, pointing to the file location on the local PC
archiveDocPath - String, the complete ELO Professional archive path where the document should be uploaded to.
docName - String, the name of the document
extraText - String, the old Keywording information is contained in this String. It will be added to the "Extra Text" section.
maskId - String, the Keywording mask id the document should recieve
oldMask - String, the name of the original Keywording Mask. This will be added to the first field of the new mask for reference
docObj - ExportDocument, is an object created from the ExportDocument Class

Throws - RemoteException

---

createFolderArchive

```
public void createFolderArchive(de.elo.ix.client.IXConnection conn,
                                 java.lang.String archivePath,
                                 ExportDocument docObj)
```

Create a folder structure in the ELO Professional archive

This method creates the folder structure within ELO professional. It takes a String such as "Root FolderÂ¶Sub FolderÂ¶Another Folder" and splits it where the "Â¶" character is found.

== Root Folder

|

=== Sub Folder

|

==== Another Folder

It automatically ignores folders that already exist to ignore conflicts or errors.

Parameters:

conn - IXConection, variable
archivePath - String, is the complete path to be created in ELO using the following format - "FolderÂ¶Child FolderÂ¶Another Folder". ELO will split the string into seperate folders using the "Â¶" character.
docObj - ExportDocument, is an object created from the ExportDocument Class

Throws - RemoteException

eloLogin

```
public de.elo.ix.client.IXConnection eloLogin(java.lang.String url,
                                              java.lang.String user,
                                              java.lang.String password)
```

Login to ELO Professional

This method logs in with a user name and password that was given at the start of the program

Parameters:

url - String is the url of the ELO Professional Repository
user - String is the ELO Professional user name
password - String is the ELO Professional users password

Returns:

Returns IXConnection variable which contains ELO Professional login information

Throws - RemoteException

---

folderNameCorrection

```
public void folderNameCorrection(de.elo.ix.client.IXConnection conn)
```

This method is to be used when all folders and documents have been created in ELO Professional. During the creation of Folder paths or documents, any name with a "/" character will be changed to "###", otherwise the name itself gets split into folders which results in errors. This method changes the name back to the original form

Parameters:

conn - IXConection, variable

Throws - RemoteException

---

com.ashley.migration.controller

Class IniReader

```
java.lang.Object
    |
    +--com.ashley.migration.controller.IniReader
```

< Constructors > < Methods >

---

public class **IniReader**
extends java.lang.Object

IniReader Class In this class, all document and folder information will be read and added to the ExportDocument or Mask Class objects. An ELO export contains two types of ini files, the initial "ExpInfo.ini" file which contains useful mask information and the many es8 files, which contain all the folder or document information. There is a also a third file type, ESW, which is not relevant to this program.

Author:

Ashley Stonehall

Version:

1.0

Constructors

IniReader

```
public  IniReader()
```

Methods

eloArchivePath

```
public void eloArchivePath(java.util.List exportFileNames,
                           ExportDocument docObj,
                           java.util.Map exportDocumentObjs,
                           java.util.List eloFileNames)
```

eloArchivePath, this method corrects the path folder or document names contained in the export path, to the ELO Archive path using the correct ELO names instead of export id's

Parameters:

exportFileNames - List <String>, a list of the export paths
docObj - ExportDocument, will be the object retrieved from the exportDocumentObjs Map
exportDocumentObjs - Map <String, ExportDocument>, a Map of the ExportDocument objects
eloFileNames - List <String>, a list of all the correct names to be used

---

eloFileReader

```
public void eloFileReader(java.lang.String directoryName,
                          java.util.Map exportDocumentObjs,
                          java.util.List exportFileNames,
                          java.util.List eloFileNames)
```

eloFileReader, this method reads the es8 files and gathers the important information such as its name, location path, whether it is a folder or document etc. and adds this information the the correct ExportDocument object

Parameters:

directoryName - String, the path to be used by the File api
exportDocumentObjs - Map <String, ExportDocument>, a map of all the created ExportDocument objects
exportFileNames - List <String>, a list of every es8 file name. This will be used to find the correct object in the exportDocumentObjs Map
eloFileNames - List <String>, is a list of the correct folder and document names

Throws - InvalidFileFormatException

Throws - IOException

---

expInfoMaskReader

```
public void expInfoMaskReader(Mask maskObj,
                              java.util.Map maskObjs,
                              java.util.List maskNames,
                              org.ini4j.Wini ini,
                              java.lang.String expInfo)
```

expInfoMaskReader, in this method the initial ExpInfo.ini is read to get the exisiting mask information (including the name and any information that has been added to the fields)

Parameters:

maskObj - Mask, an object that contains Keywording Mask information
maskObjs - Map <String, Mask> a map for the created Mask objects
maskNames - List <String>, a temporary list with all the old Keywording names
ini - Wini, instance of the ini file reader api
expInfo - String, is the file location of the ExpInfo.ini file on the local PC

Throws - InvalidFileFormatException

Throws - IOException

---

extraText

```
public void extraText(java.util.List exportFileNames,
                      ExportDocument docObj,
                      java.util.Map exportDocumentObjs,
                      Mask maskObj,
                      java.util.Map maskObjs)
```

extraText, this method corrects all of the existing Keywording information taken form the es8 files to a String that will later be added to the "Extra text" section of the new mask. It replaces Field names to something that makes more sense and splits the String up with new lines to be more readable

Parameters:

exportFileNames - List <String>, a list of every es8 file name. This will be used to find the correct object
docObj - ExportDocument, will be the object retrieved from the exportDocumentObjs Map
exportDocumentObjs - Map <String, ExportDocument>, is a Map of all the document and folder objects
maskObj - Mask, will be the object retrieved from the maskObjs Map
maskObjs - Map <String, Mask>, a map with all of the Mask objects

---

com.ashley.migration.controller

Class InputController

```
java.lang.Object
    |
    +--com.ashley.migration.controller.InputController
```

< Constructors > < Methods >

---

public class **InputController**
extends java.lang.Object

This Class is will handle the information taken from the user and then send it to the relevant models and controllers. It will also manipulate strings that need to be handled in different ways by different methods.

example - to create a path in ELO the String needs to be, "rootÂ¶Sub FolderÂ¶Another Folder" before it is used by the function.

example - to create a document in ELO the String needs to be, "root/Sub Folder/Another Folder" before it is used by the function.

Author:

Ashley

Version:

1.0

Constructors

InputController

```
public  InputController()
```

Methods

userInput

```
public void userInput(javafx.event.ActionEvent event)
```

Package com.ashley.migration.model

Class Summary

ExportDocument

ExportDocument Class

Objects that will contain all of the information about folders or documents contained in the exported folder location

Uses Simple getter and setter methods to assign or retrieve the information

Mask

Objects that will contain all of the Keywording Mask information used by the folders or documents contained in the export location

Uses Simple getter and setter methods to assign or retrieve the information

---

com.ashley.migration.model

Class ExportDocument

```
java.lang.Object
    |
    +--com.ashley.migration.model.ExportDocument
```

< Constructors > < Methods >

---

public class **ExportDocument**
extends java.lang.Object

ExportDocument Class

Objects that will contain all of the information about folders or documents contained in the exported folder location

Uses Simple getter and setter methods to assign or retrieve the information

Author:
Ashley Stonehall

Version:
1.0

Constructors

ExportDocument

public **ExportDocument**(java.lang.String name)

Methods

getArchivePath

public java.lang.String **getArchivePath**()

---

getDate

public java.lang.String **getDate**()

---

getDocType

public java.lang.String **getDocType**()

---

getEloName

public java.lang.String **getEloName**()

---

getExportName

public java.lang.String **getExportName**()

---

getExportPath

public java.lang.String **getExportPath**()

---

getExt

public java.lang.String **getExt**()

---

getExtraText

public java.lang.String **getExtraText**()

---

setArchivePath

public void **setArchivePath**(java.lang.String archivePath)

---

setDate

public void **setDate**(java.lang.String date)

---

setDocType

public void **setDocType**(java.lang.String docType)

---

setEloName

public void **setEloName**(java.lang.String eloName)

---

setExportName

public void **setExportName**(java.lang.String exportName)

---

setExportPath

public void **setExportPath**(java.lang.String exportPath)

setExt

```
public void setExt(java.lang.String ext)
```

setExtraText

```
public void setExtraText(java.lang.String extraText)
```

com.ashley.migration.model

Class Mask

```
java.lang.Object
    |
    +--com.ashley.migration.model.Mask
```

< Constructors > < Methods >

public class **Mask**
extends java.lang.Object

Objects that will contain all of the Keywording Mask information used by the folders or documents contained in the export location

Uses Simple getter and setter methods to assign or retrieve the information

Author:
Ashley Stonehall

Version:
1.0

Constructors

Mask

```
public  Mask(java.lang.String name)
```

Methods

getMaskFields

public java.util.List **getMaskFields**()

---

getMaskName

public java.lang.String **getMaskName**()

---

setMaskFields

public void **setMaskFields**(java.util.List maskFields)

---

setMaskName

public void **setMaskName**(java.lang.String maskName)

INDEX