**Essential Tasks for Parsing and Inserting Script Images**

Just to demo minimum works needed when everything is hard coded.

1. Create a database (on SQL Server) and create three tables with the following SQL

CREATE DATABASE OBScript;

-- go & use are TSQL commands

go

use OBScript

go

-- create tables

CREATE TABLE SectionHeader (

RadicalID int PRIMARY KEY,

Script varbinary(max),

ChnChar nchar(1)

);

CREATE TABLE Character (

CharNo int PRIMARY KEY,

DefaultScript varbinary(max),

ChnChar nchar(1),

PhoneticNotation nchar(6),

EnglishTranslation varchar(100),

LeiCuanNo int,

Notes nvarchar(100),

RadicalID int FOREIGN KEY

REFERENCES SectionHeader

);

CREATE TABLE Variation (

SeqNo int PRIMARY KEY,

Script varbinary(max),

SpecimentNo int,

CharNo int FOREIGN KEY

REFERENCES Character

);

-- nchar=native character type

1. Develop (Java) application to read PDF file and parse the cells to harvest information. Everything is hard-coded by now.

Compile the code with the last two statements commented to just show the sample page displayed with a scrollpane.

//test.browseSripts(test);

//test.enterScripts(test);

The script images are enclosed with red squares from which image data can be retrieved and entered into database tables.

1. Uncomment the line to browse the page with the browseScripts() method. Dialog boxes will be popped up to show “Section Header” and “Character”s in order. Harvested images are shown as the Image icon on dialog box created by the JOptionPane class.

BufferedImage objects are needed to hold the page image and store smaller portions clipped from the big picture.

1. Next, comment the first line and uncomment the second to demonstrate how to insert each harvested image and other data that can be parsed from corresponding cells in the same row as the script into database.

//test.browseSripts(test);

test.enterScripts(test);

Also comment out the line in the browseScripts() method to show dialog boxes as in step 3 to avoid having to see all the scripts again.

//JOptionPane.showConfirmDialog(null, "Script #" + (i+1),

// "Harvested Script Images", JOptionPane.OK\_CANCEL\_OPTION,

// JOptionPane.INFORMATION\_MESSAGE, new ImageIcon(bim));

All the data except the script image are hard-coded in arrays as used in the enterScript() method.

The SectionHeader, which shares the same image with script #1, has been inserted into the SectionHeader table. You will see an exception when testing the program (referring a line in the insertSectHdr() method).

* A DAO object is created with the SqlDbAccessor class and connected to the SQL Server instance on AWS in the constructor.
* It is used in the enterScript() method and its insertIntoSectionHeader() method invoked.
* An insertIntoCharacterTable() method can be added in the SqlDbAccessor class and invoked in ParserTest’s insertChar() method to run the query as generated and insert records into the Character table.