Getting Started

lipi-core-toolkit 4.0

lipitk.sourceforge.net

Contents

1 Prerequisites

This section describes the prerequisites for installing and using *lipi-core-toolkit 4.0.0*.

1-1 Supported platforms and environment

lipi-core-toolkit 4.0.0 has been tested on the following platforms:

- Microsoft Windows - 7 32 bit and 64 bit
- Ubuntu Linux Version - 10.10 32 bit and 64 bit

1-2 Disk space requirements

lipi-core-toolkit 4.0.0 provides source and binary packages for Windows and Linux. Separate source packages are provided for Windows and Linux. The space required to extract the source package is 20MB and binary package is 30MB. To build the source package after extracting you need 130MB of free space on Windows and 40MB on Linux.

1-3 Software requirements

Item and Description	**Windows 7**	**Linux**
Building lipi-core-toolkit code	Microsoft Visual C++ 2008 / MsBuild for VC2008	G++ 4.4 or above
Executing scripts	Perl 5.1 or above and Archive::Zip	Perl 5.1 or above and Archive::Zip
Building lipiDesigner code	Java Development Kit (JDK) jdk1.6.0_26 or above	Openjdk-6-jdk

Table 1: Software requirements

NOTE: Archive::Zip perl module can be obtained from http://search.cpan.org/~adamk/Archive-Zip-1.30/lib/Archive/Zip.pm. This module is in tar.gz format, user can use “tar –xzvf <tar.gz file> ” to untar it on Linux and winzip on windows. Once uncompressed, user can go to “lib/Archive/” folder of Archive-Zip-1.30 and copy “Zip/ & Zip.pm” to “Archive” folder of perl installation on the current machine.

---

2 Installing Packages

lipi-core-toolkit 4.0.0 is available as binary and source packages for Windows and Linux. The Windows package is a self-extracting exe and it can be installed by double-clicking the exe. The Linux package is a tar ball and it can be installed by executing the below command:

```
tar -xzvf  <lipi-core-toolkit-tarred-compressed file>
```

Platform	**Package**
Windows 7	**Binary:** lipi-core-toolkit4.0.0-bin-x86.exe lipi-core-toolkit4.0.0-bin-x64.exe **Source:** lipi-core-toolkit4.0.0-src-win-x86.exe lipi-core-toolkit4.0.0-src-win-x64.exe
Linux	**Binary:** lipi-core-toolkit4.0.0-linux-x86_64.tar.gz lipi-core-toolkit4.0.0-linux-i686.tar.gz **Source:** lipi-core-toolkit4.0.0-src-linux.tar.gz

Table 2: lipi-core-toolkit packages

Lipi-core-toolkit uses an environment variable called LIPI_ROOT as reference for locating the dlls/shared objects, root location of data etc. This variable needs to be set for building the source code and also for executing the binaries on Linux and Windows.

On windows platform the LIPI_ROOT variable is automatically set on installation of the core tool kit. On Linux platform this variable has to be set manually. For example, if the package was extracted to the directory `/opt`, LIPI_ROOT should be set to `/opt/`<lipi-core-toolkit install directory> as shown below:

```
export LIPI_ROOT=/opt/<lipi-core-toolkit install directory>
```

If the user wants LIPI_ROOT to persist across multiple sessions of Linux then the above entry can be made in .profile or .bashrc file of user's home directory by executing the below command at the command prompt:

```
# echo "export LIPI_ROOT=/opt/<lipi-core-toolkit install directory> " >> $HOME/.profile
```

CAUTION: There should be no blank spaces in the path specified by LIPI_ROOT

NOTE: The alpha-numeric shape recognizer is bundled with all packages of the toolkit. Listed below is the accuracy in percentage observed on Windows & Linux

Recognizer	**Accuracy (Windows & Linux) (%)**
Numerals	94.34
English uppercase	94.58
English lowercase	90.59
Punctuation	95.83

3 Building Source Package

On downloading the source package, user can follow the below instructions for build on Windows and Linux. If binaries are downloaded, user can skip this section and go to the next section on Executing binaries.

3-1 Building on Windows for VC2008

a) To build the modules on Windows for VC2008, devenv must be included in system PATH variable. This can be done by executing <visual studio 2008 install dir>\Common7\Tools\ vsvars32.bat from the command prompt.

b) To build lipi-core-toolkit 4.0.0, on Windows for VC2008, execute the following command from `<lipi-core-toolkit install directory>\windows\vc2008`

```
MsBuild lipitk.targets
```

c) To build lipiDesigner, first go to $LIPI_ROOT/src/apps/lipiDesigner/src/lipijniinterface/windows and execute the below command:

```
msbuild  lipijniinterface.targets
```

d) To build javauserinterface go to $LIPI_ROOT/src/apps/lipiDesigner/src/javauserinterface/ windows and execute :

```
buildjar.bat
```

3-2 Building on Linux

To build *lipi-core-toolkit 4.0.0*, execute the following command from <lipi-core-toolkit install directory>/linux on 32 bit or <lipi-core-toolkit install directory>/linux-x64 on 64 bit machine.

```
make -f Makefile.linux
```

To build lipiDesigner, execute the following command from <lipi-core-toolkit install directory>/src/apps/lipiDesigner/src/lipijniinterface/linux on 32 bit or <lipi-core-toolkit install directory>/src/apps/lipiDesigner/src/ lipijniinterface/linux-x64 on 64 bit machine.

```
make -f Makefile.linux
```

Then user needs to go to <lipi- core-toolkit install directory>/src/apps/lipiDesigner/src/javauserinterface/linux on 32 –bit or <lipi-core-toolkit install directory>/src/apps/lipiDesigner/src/javauserinterface/linux-x64 on 64 bit machine and export JAVA_HOME environment variable. User can execute buildjar.sh script to build java code as shown below:

```
# export JAVA_HOME=/usr/lib/jvm/java-6-openjdk
# sh buildjar.sh
```

4 Executing binaries

The user can go to $LIPI_ROOT/bin and execute binaries according to their requirements. These binaries are seven in number and their usage can be obtained by executing them with no options. (Note: User can refer user manual for more details on binaries.)

There is another binary present under the directory $LIPI_ROOT/src/apps/lipiDesigner/bin. This binary belongs to lipiDesigner and it can be launched by executing a script (on Linux) or a batch file (on Windows) as shown below.

For example:

(on Linux)

```
sh lipiDesigner.sh
```

(on Windows)

```
lipiDesigner.bat
```

For more details on lipiDesigner, user can refer section Using lipiDesigner.

5 Creating your own shape recognizer

Using the toolkit users can create their own shape recognizers in addition to the built in recognizers that are shipped with the tool kit. Custom shape recognizers can be created using the following applications:

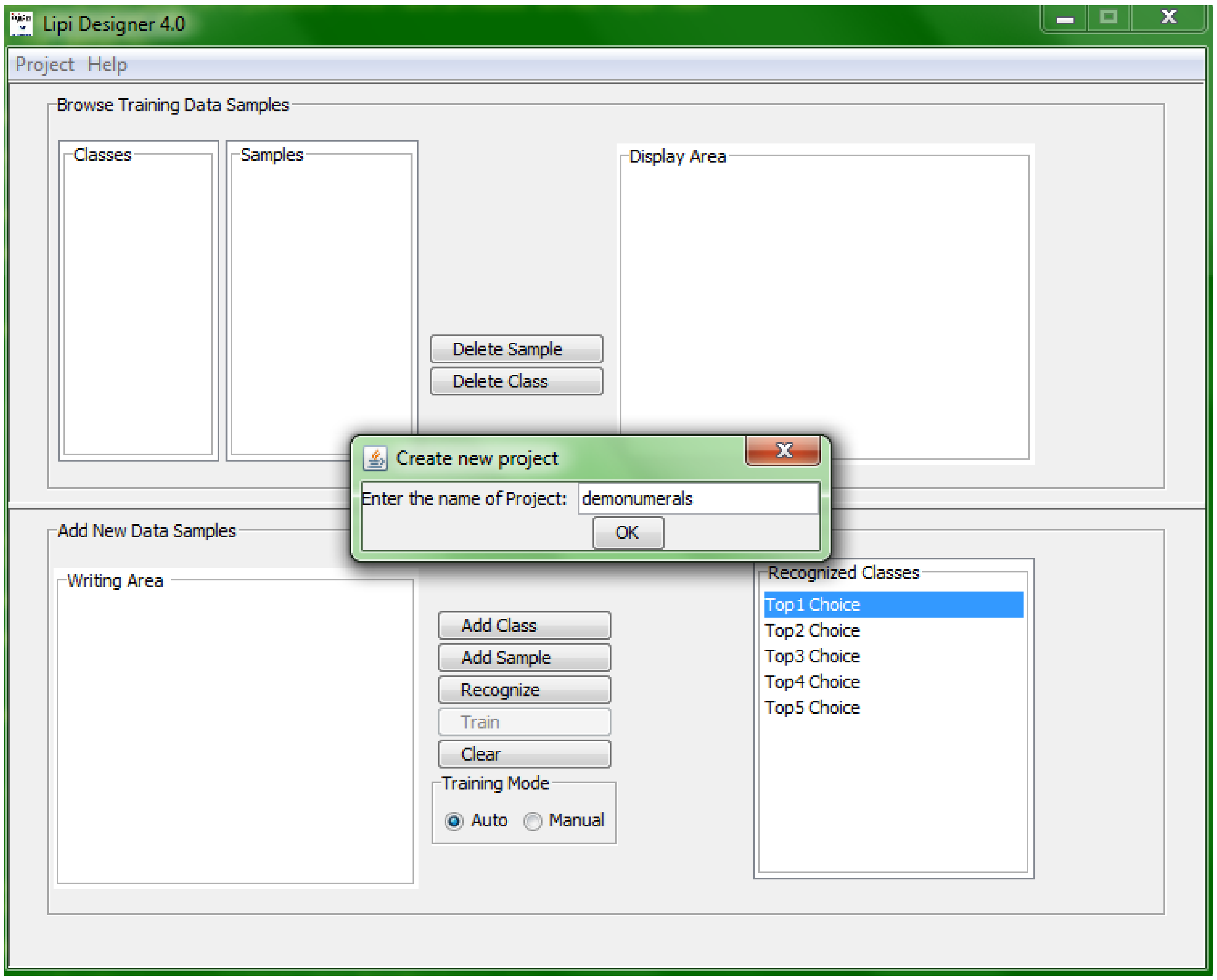
- lipiDesigner
- runshaperec

5-1 Using lipiDesigner

5-1-1 Creating a new project

Click **New** in the Project menu**,** and the **Create new project** dialog box (Figure 1) appears. The dialog box enables the user to enter the project name. For every new project created by the user, a new folder is created under `$LIPI_ROOT/projects`.

WARNING: The name of the new project must not clash with any of the already existing projects. A warning message is displayed in case of a clash.



Enter the name of Project: demonumerals

Training Mode: [✓ Auto] Manual

Figure 1: Create new project dialog box

1. To add a new class to the project, draw a shape in the Writing Area and click on **Add Class**. User can add multi-stroke shapes also.

   ⚠**WARNING:** A warning message pops up in case an empty stroke is added.

2. To delete a class, select the class and click on **Delete Class**.

3. To add samples to an existing class, click on **Add Sample**. The new samples are added to the class selected the **Classes** list box. By default, the most recently added class is selected. To add samples to a different class, select the desired class in the **Classes** list box before clicking on **Add Sample**.

4. To delete a sample from the class, select the sample and click on **Delete Sample.**

NOTE: Deleting the last sample of a class results in deletion of the class from the project.

5. To train the recognizer on all the classes in a project, click on **Train**. There are two modes for training the recognizer:

a) **Auto Mode**: If the training mode is set to Auto, the application automatically trains the recognizer after every addition/deletion of a sample.

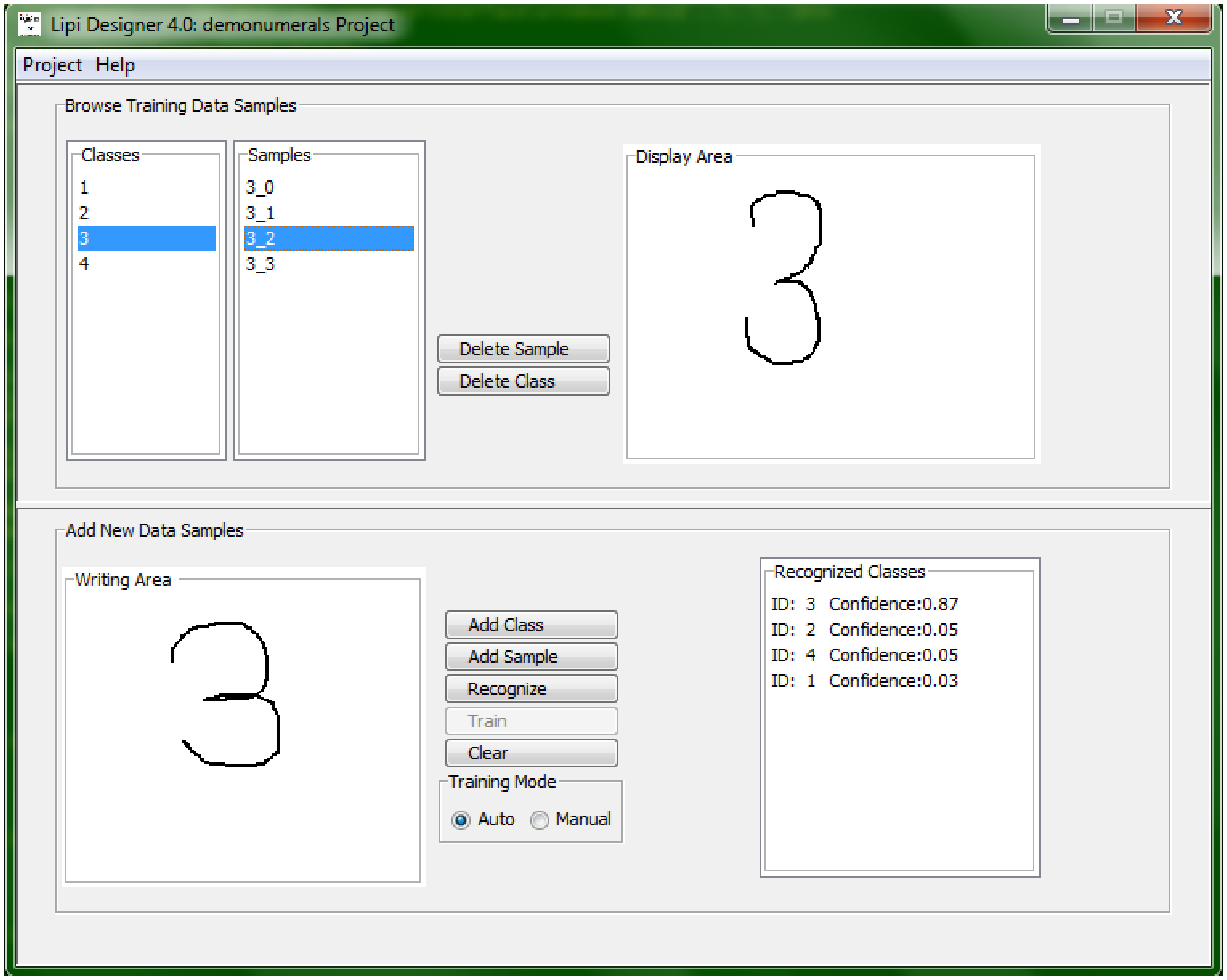
b) **Manual Mode**: In this mode, the user is required to train the recognizer explicitly, by clicking on the **Train** button. The user can choose to invoke training after adding or deleting multiple classes/several samples.

6. The performance of a trained recognizer can be tested using the Recognize functionality of *Lipi Designer*

a) Draw a shape in the writing area

b) Click the **Recognize** button, and the five best matching classes are listed in the **Recognized Classes** panel (Figure 2), with the first choice indicating the highest confidence result. If the number of matching classes is less than five, the system will display only the available classes, with top choice indicating the highest confidence.

⚠**WARNING:** Recognize cannot be called for an empty stroke.



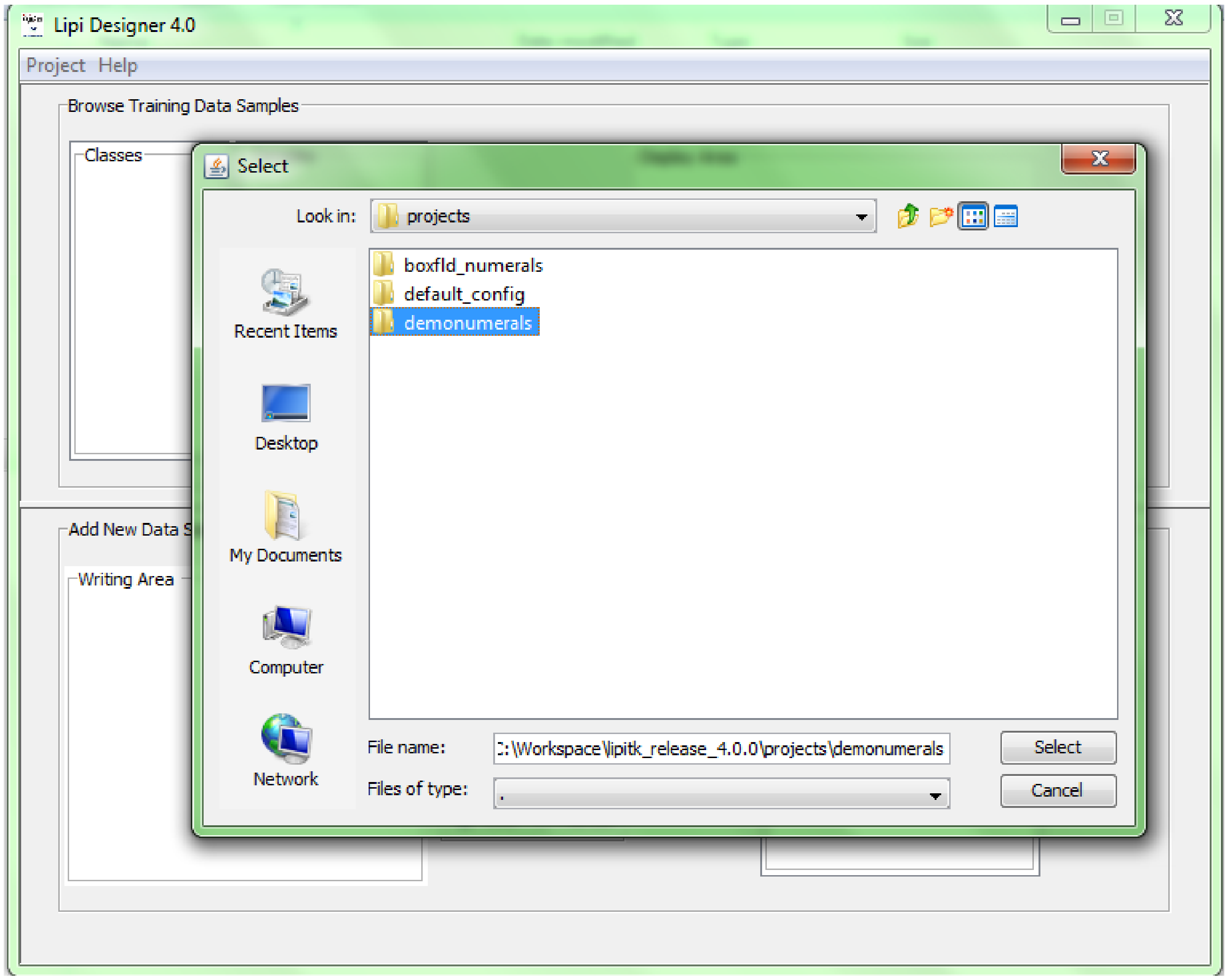
```
ID:  3  Confidence:0.87
ID:  2  Confidence:0.05
ID:  4  Confidence:0.05
ID:  1  Confidence:0.03
```

Figure 2 : Recognition results

To clear the stroke(s) written in the **Writing Area** or the result shown in the **Recognized Classes** panel, click **Clear**.

5-1-2 Loading an existing Project

To load an existing project, click **Load** in the Project menu. A window (Figure 3) displaying all the existing projects appear. The user can load a project by opening the projects directory and selecting the `<project name>`, available under `$LIPI_ROOT/projects/`



File name: :\Workspace\lipitk_release_4.0.0\projects\demonumerals

Figure 3: Load existing project

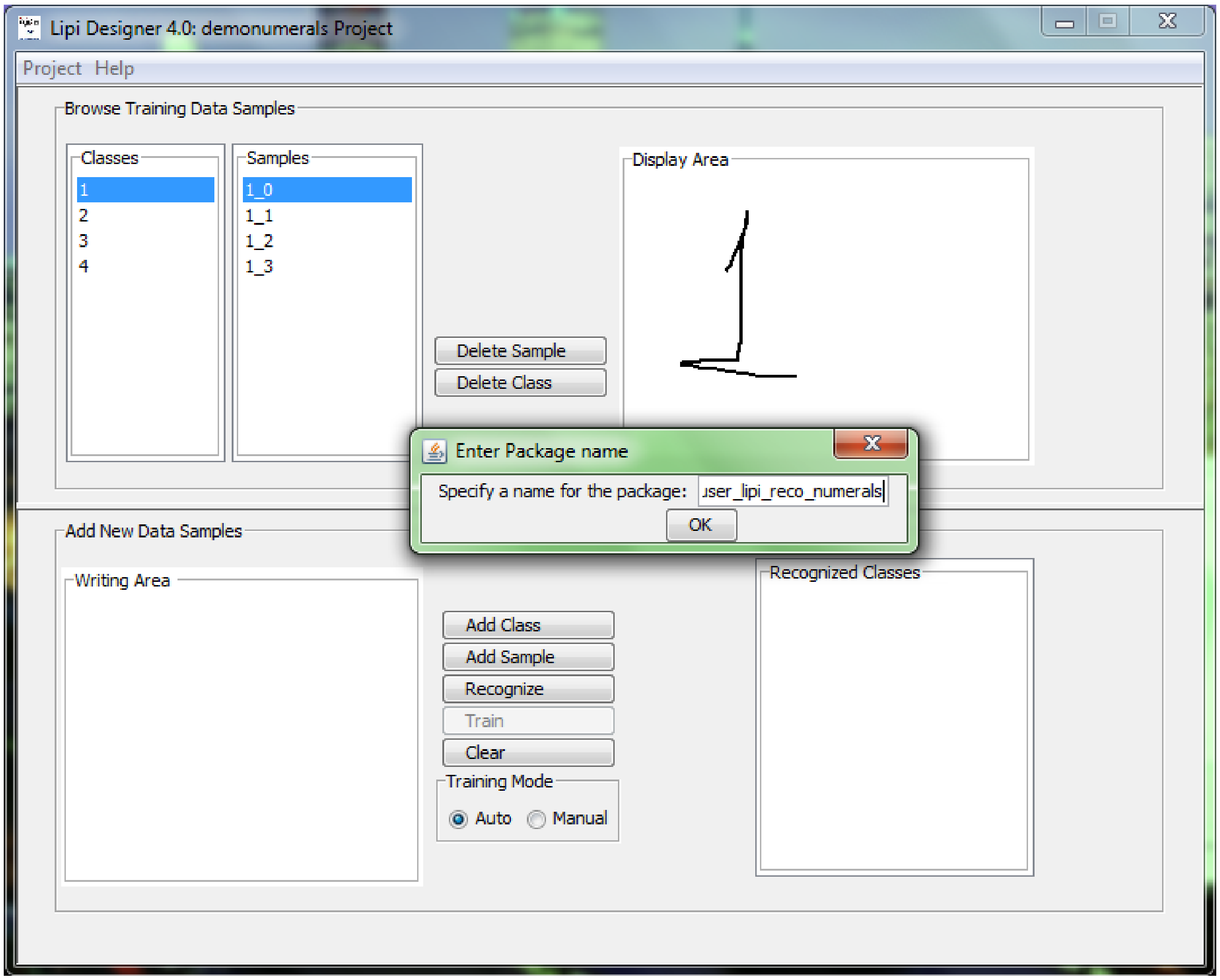
⚠**WARNING:** Before loading/opening a project, the current working project must be closed.

Note: The created shape recognizer can be integrated with sample application on same or on client machine. Refer to section 14 of user manual. For integrating on client machine, the shape recognizer has to be packaged. Refer to next section for details on packaging.

5-1-3 Packaging a project

Once the recognizer has been trained, tested, evaluated and found to be satisfactory, it may be packaged for integrating it into an application. Packaging a project results in the creation of a zip file.

To create a package, click **Package** in the Project menu, and the Enter package name dialog box (Figure 4) appears.



Specify a name for the package: user_lipi_reco_numerals

Training Mode: [✓ Auto] Manual

Figure 4: Enter package name dialog box

Enter package name and click **Ok**. Package is created under `$LIPI_ROOT/package` directory. A prefix USER_LIPI_RECO is, by default, added to every package name. However, user can remove the prefix from the final package name.

⚠**WARNING:** The package name entered must not clash with any already existing packages.

5-2 Using runshaperec

The `runshaperec` tool is an executable found under `<lipi-core-toolkit install directory>/bin`, used for training and testing the shape recognizer module. The shape recognizer module can be trained by specifying train option to runshaperec and if the user wants to test it, test option can be specified. Runshaperec execution is shown below as an example and user can refer section 8 of user manual for more details on runshaperec.

Usage:

```
runshaperec –project <project name> -train <file name>
```

e.g.

(on Linux)

```
./runshaperec –project demonumerals –train
$LIPI_ROOT/projects/demonumerals/data/trainlist.txt
```

(on Windows)

```
runshaperec –project demonumerals –train
%LIPI_ROOT%\projects\demonumerals\data\trainlist.txt
```

6 Developing an application with the built-in shape recognizer

6-1 Introduction

This section demonstrates how to use APIs exposed by recognition components and modules of lipi-core-toolkit with sample applications that require shape or word recognition.

lipi-core-toolkit 4.0.0 provides two sample applications for linux and windows.

- shaperectst: sample client application for shape recognizers.
- wordrectst: sample client application for word recognizers.

In addition to the above two the toolkit also provides another sample UI based application for windows

- shaperectsui

The following section covers all the technical details of the sample application – shaperectst provided by LipiTk. The developer can refer these details and also the source code available under LIPI_ROOT\src\apps\samples for developing their own applications with the built-in recognizer.

6-2 Sample program `shaperectst`

The sample program `shaperectst` is provided as an example of how the character recognizers can be invoked from an application program. It illustrates the steps for instantiating the recognizers using their logical names, passing digital ink to them and obtaining recognition results. The major steps in the program are illustrated below:

1) Get the Lipiroot environment variable.
2) Load the lipiengine dll/shared project.
3) Create an instance of lipiengine module
4) Load the model data into memory before starting the recognition.
5) Read the unipen ink file which has all the screen context, device context and ink information and pass it to recognize function.
6) Invoke the recognize function.

7) Display the recognized results returned by recognize function.
8) Delete the shape recognizer object.
9) Unload lipiengine module from memory.

Note: More details for developing the application can be obtained by referring 13-2 section of User Manual.

---

7 Acknowledgement

We would like to thank IRESTE, University of Nantes (France) for providing the handwriting database termed IRONOFF. The pre-built recognizers are trained using the IRONOFF data. For more details see http://www.infres.enst.fr/~elc/GRCE/news/IRONOFF.doc