



2BIO702 - Notes for the Java Image Enhancer (MSc Biometrics, Jean-Michel Besnard)

1 Content and Usage

1.1 Content

The root directory contains:

- source code (.java files)
- compiled classes (.class)
- some images that can be loaded in the application (.jpg)

In addition, root directory contains four sub-directories in which the following content can be found:

- **Filters:** the filters available to the application
- **Graphics:** the icons and images used for the design of the Graphical User Interface
- **Javadoc:** the javadoc generated from the documentation embedded in the source code
- **Documentation:** this documentation

1.2 Usage

To launch the application, in a shell, make sure that you are in the root directory of the archive and type the following command:

```
java Enhancer
```

2 Java classes

2.1 The four classes

- **Enhancer:** the frame of the application
- **ImageDisplay:** the panel where images are being show

- **BlurFilter**: the implementation of a blur filter
- **ImageProcessor**: the class that processes the data to generate the filtered images

NOTE: an external class *ExampleFilter* from Sun Microsystem has been imported into the directory because this class was not available in the Java API

2.2 Signatures of the classes

2.2.1 Enhancer

```
public void setFilterable(boolean enable)
public void setMenus(boolean undo, boolean filters)
public boolean isSelected(char color)
public static void main( String args[] )
private Image getWelcomeImage(String imagePath)
private void resetModeHistory()
private void addGrayscaleToHistory(boolean modeIsGrayscale)
private void removeGrayscaleFromHistory()
private boolean hasAnyGrayscaleInHistory()
private String[] getFilters(String root, String extension)
private void setUndoable(boolean undoable)
private void setRGBMenu(boolean enable)
private void setModesMenu(boolean enable)
private void imageFrame_quit()
private void imageFrame_open()
private void imageFrame_blur(String filter)
private void imageFrame_switchMode(String mode)
private void imageFrame_undo()
private void imageFrame_selectRGB()
```

2.2.2 ImageDisplay

```
public ImageDisplayPanel(Enhancer enhancer )
public void paintComponent(Graphics g)
public void setWelcomeScreen(Image i)
public void setImage(Image i)
public void blur(boolean modeIsGrayscale,String filter)
public void updateRGBDisplay()
public int[] getRelevantPixels(boolean r, boolean g, boolean b)
public void undo()
public boolean isUndoable()
public boolean isFilterable()
```

2.2.3 BlurFilter

```
public BlurFilter(String filePath)
public double[] getFilterMatrix()
public int getCentralPixel()
private Vector getFilterFromFile(String filePath)
private double[] createFilterMatrix(Vector list)
```

2.2.4 ImageProcessor

```
public ImageProcessor(int[] pixels,int width,int height)
public int[] getBlurredPixels()
public void blurFilter(boolean modeIsGrayscale, double[] filterMatrix,
                       int centralPixel)
private int[][] getMatrixCoordinates(int position, int matrixSize)
private int getValue(int x, int y)
private void convertImage()
private void rgbBlurFilter(double[] filterMatrix, int centralPixel)
private void grayscaleBlurFilter(double[] filterMatrix, int centralPixel)
```

3 The application

3.1 Grayscale and RGB

For the coursework, it was asked to convert the *RGB* data (ie, into *grayscale*) to process them. The application also supports *RGB* mode but the default (that can be changed in the *mode* menu) is *grayscale*.

Once a filter has been applied in *grayscale* mode, the *mode* menu does no longer allow to switch since using a filters in *RGB* mode on an image that has been previously converted into *grayscale* would produce the same result than using a filter in *grayscale* mode.

However, with the use of *undo* in the *Modify* menu, if you get back to a stage where no filters had been applied in *grayscale* mode, the *mode* will then offer you again to switch between *grayscale* and *RGB*.

For example:

stage 0: image loaded
stage 1: filter applied in RGB mode (=> *mode* menu is enabled)
stage 2: filter applied in RGB mode (=> *mode* menu is enabled)
stage 3: filter applied in grayscale mode (=> *mode* menu is disabled)
stage 4: filter applied in grayscale mode (=> *mode* menu is disabled)
stage 5: undo filter (=> *mode* menu is disabled)
stage 6: undo filter (=> ***mode* menu is enabled**)

3.2 Disabled and enabled menus

When the application is launched, only the *File* menu is enabled, offering to either open a file or to quit.

Once an image has been successfully loaded, it is then possible to switch to the *RGB* mode (*grayscale* being the default) and to select a filter from the *Modify* menu.

After at least one filter has been applied to the loaded image, the *undo* becomes available to remove one by one all the filters that you may have applied.

Thanks to the *Colors* menu, you select which primary colors to show on the panel. This works both with *RGB* and *grayscale* mode.

If you remove all the filters previously applied by using *undo*, you will restore the original image. At this stage, both filters and the *Colors* menu become disabled.

3.3 Filters format

The filters are stored in file in the *Filters* directory and have the extension *.filter*. In the *Modify* menu, they are given for name, the name of the file without its extension. A format is defined for the filters:

- In the first line, the coordinates of the central point must two integers separated by a comma (“,”)
- Then all following lines represent the matrix of the filter with each element separated by a “:”

For example, a [3,3] matrix :

```
2,2
0.10:0.13:0.16
0.11:0.14:0.17
0.12:0.15:0.18
```

The matrix size must be an odd bigger than 1 ([3,3] [5,5] [7,7] ...).

NOTE: the application has been developed to support filter of any size (or at most, the size of the picture)

When a filter is loaded from a file, the following is checked:

- The file can be read (permissions, exists,...)
- The coordinates of the central point can be read, are integers and not out of the bound of the filter
- The number of values found match the length of the matrix
- The size of the matrix is an odd bigger than 1

If any of this seven exceptions occurs, it will be caught to not crash the program and the user will be told with a specific and detailed message in a pop up what the problem is.