User Guide: LIF Viewer

Vladyslav Furda

May 5, 2025

1 Features of LIF Viewer application

1.1 LIF File Support and Metadata

LIF Viewer enables seamless interaction with proprietary *Leica LIF* microscopy files. These files can contain multiple series, channels, and Z-planes, and the application extracts and organizes them in a tree-based structure. Detailed and relevant metadata such as image dimensions and physical pixel sizes are retrieved using `Bio-Formats' OMEXML` service.

- Series, channel, and plane (Z-stack) hierarchy visualization
- Metadata display with series dimensions, Z/C/T counts, and physical sizes
- MIP rendering and Z-stack inspection

1.2 Multi-Channel Image Blending

Users can blend multiple grayscale image channels into a composite multi-channel image. This is useful for visualizing co-localization across channels in a single image.

- Select channels that you want to include
- Adjust brightness in the preview for better vision
- View and interact with the result immediately

1.3 Standard Image Support and Batch Loading

In addition to LIF files, LIF Viewer supports standard image formats (PNG, JPEG, BMP, GIF) and can batch-load entire directories as well. It enables to check everything from one application, especially if you need to double-check image that was imported in standard image format.

- Open individual images
- Load folders
- Navigate and view directly in the image tree

1.4 User Interface Features

The application is built using Java Swing with FlatLaf for a modern look. It supports intuitive interaction through:

- Tree navigation with selection handlers
- Sliders for brightness, contrast, zoom, and Z-slice control
- Toolbar and menu bar for quick access to common actions

- Keyboard shortcuts: Ctrl+O (Open), Ctrl+S (Save), Ctrl+Q (Quit)

1.5 Saving and Exporting

GalleryApp supports exporting the currently displayed image (including all user adjustments) to disk in PNG or TIFF format.

2 How to Use

2.1 Opening Files

- Use **File → Open** or the toolbar button to open a file or folder.
- Supported types: `.lif`, `.png`, `.jpg`, `.jpeg`, `.bmp`, `.gif`

2.2 Navigating the Image Tree

- LIF files are expanded into a tree: Series → Channels → Planes
- Click a channel to load the full stack, or a single plane to load that slice

2.3 Adjusting Image Display

- Use the sliders on the right to control brightness, contrast, zoom, and Z-slice
- Z-slider is only enabled if the image has multiple Z-planes

2.4 Using Blend Feature

- Select a Series from the tree
- Click **Blend** in the toolbar
- Choose channels to blend and adjust brightness
- Click OK to view the blended image in the main panel

2.5 Viewing Metadata

- Click **Metadata** in the toolbar when a LIF file is loaded
- A scrollable dialog shows metadata for each series

2.6 Saving Images

- Use **File → Save As** or Ctrl+S
- Choose PNG or TIFF format
- The currently viewed image (with adjustments) will be saved

2.7 Enable MIP View

- Use **View → Show MIP** to toggle maximum intensity projection
- Re-opens the current LIF file and adds MIP entries to the tree

Setup and Build Instructions

Prerequisites

- Java Development Kit (JDK) 21
- Apache Maven 3.6+
- Internet connection (for Maven dependencies, except where stated)

Project Structure

- `src/main/java` – main application source
- `src/test/java` – unit tests
- `src/main/javadoc/overview.html` – overview page for generated Javadoc
- `lib/bioformats_package.jar` – Bio-Formats library (installed manually)

Installing the Bio-Formats JAR Manually

Due to compatibility issues, the Bio-Formats library is not pulled via a Maven repository but installed locally:

```
mvn install:install-file \
  -Dfile=lib/bioformats_package.jar \
  -DgroupId=org.openmicroscopy \
  -DartifactId=bioformats_package \
  -Dversion=6.14.1 \
  -Dpackaging=jar
```

Building the Project

Compile the application using:

```
mvn clean compile
```

Running the Application

To run the main class (`org.example.Main`) via Maven:

```
mvn exec:java
```

Running Unit Tests

Unit tests are written using JUnit 5. To execute them:

```
mvn clean test
```

Results will be displayed in the terminal and written to `target/surefire-reports`.

Generating Javadoc

To generate the HTML documentation:

```
mvn javadoc:javadoc
```

Open `target/site/apidocs/index.html` in your browser.

Java Version Configuration

The project targets Java 21:

```
<maven.compiler.source>21</maven.compiler.source>
<maven.compiler.target>21</maven.compiler.target>
```

Ensure `java -version` points to JDK 21 before building.

Contact

For issues or suggestions, refer to the README file or contact the developer.