



# LIFEx v7.8.n

## Labeling application

— LIFEx —

C. Nioche, I. Buvat

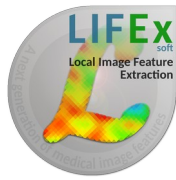


# How to use the Labeling application



LIFEx version 7.8.n

Last update of document: 2025/04/17

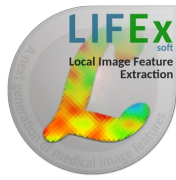


# User guide

Labeling application  
— LIFEx —

## CONTENTS

- Introduction
- Labeling protocol
  - Patient selector
  - Properties form
- Step by step
  - Building the script
  - Executing a script
  - Reading the results file
- Available Field Types
  - Textfield, Checkbox, Dropdown...



# Introduction

The labeling protocol allows users to easily **annotate images and associated regions**. It is a generic module that makes it possible to define pre-established questions / answers for application-specific annotation and to fill in a database. This database can then be exploited for **machine learning** or **deep learning** purposes.

The questions (and possible answers) are pre-defined in a script built beforehand by the user. In the script all information necessary for the annotation task is specified. The execution of the script is then a repetition of the tasks in an automatic way with automated loading of the images and of all menus needed for the annotation.

The filling in of the annotation tasks is followed by a controller that monitors and displays the progress of the script. The user can thus know at any time the progress of his work.



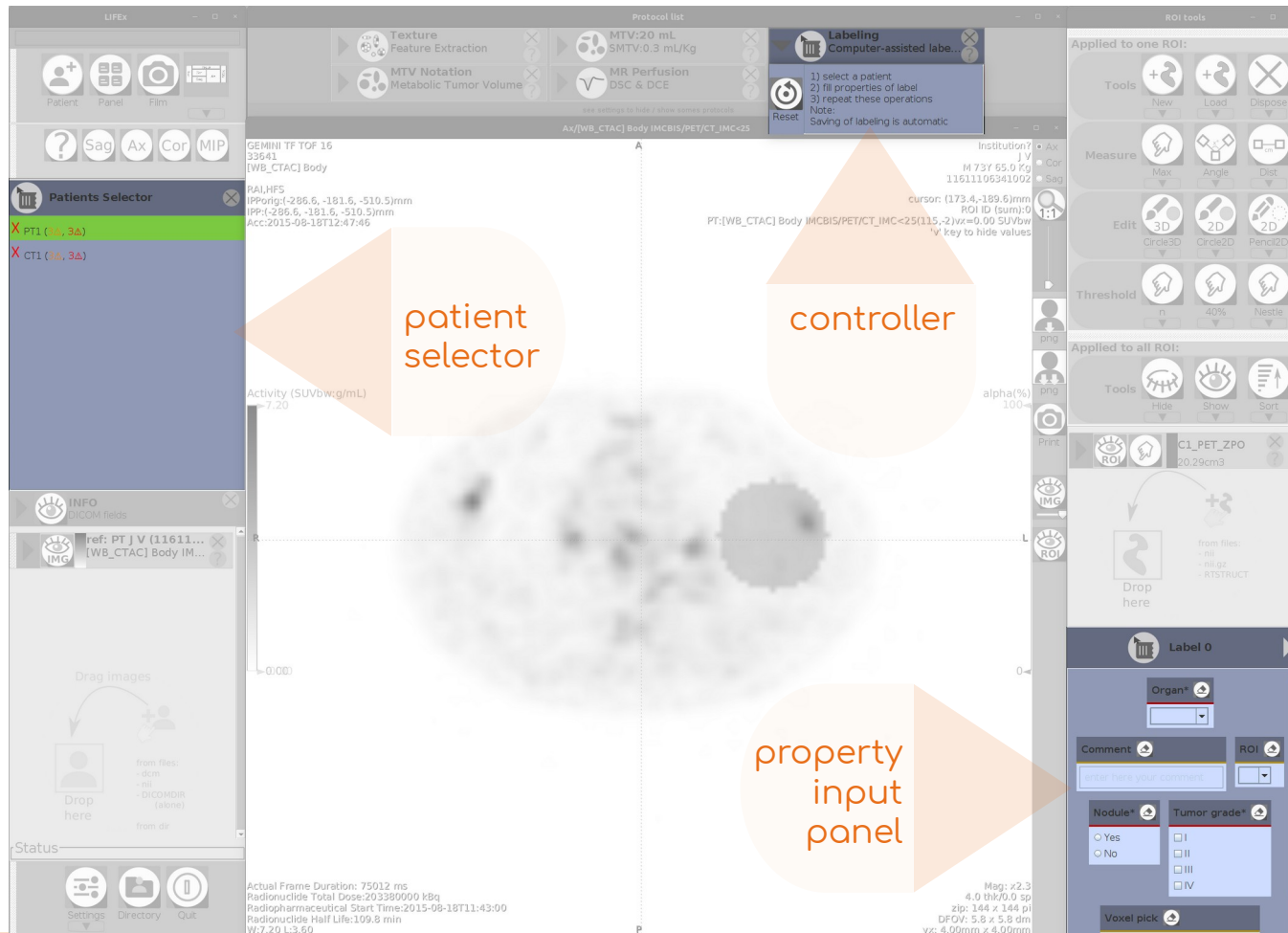
# User guide

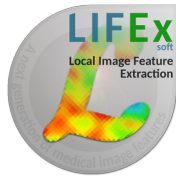
## Labeling application

— LIFEx —

# Introduction

The protocol consists of 3 panels that are open in the LIFEx application:





# User guide

Labeling application  
— LIFEx —

## CONTENTS

- Introduction
- Labeling protocol
  - Patient selector
  - Properties form
- Step by step
  - Building the script
  - Executing a script
  - Reading the results file
- Available Field Types
  - Textfield, Checkbox, Dropdown...



# User guide

Labeling application  
— LIFEx —

## Main protocol panel

Controller

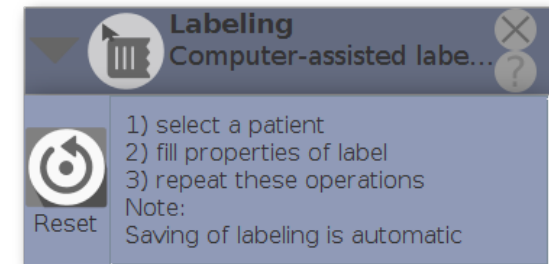
### Main Protocol Window

- This panel displays the method to be followed for the annotation process.

### Reset button

This reset deletes the annotation result file.

This deletion does NOT apply to the registered ROI/Series files. These are not deleted and all files remain on the hard disk.





# User guide

## Labeling application

— LIFEx —

# Patient selector

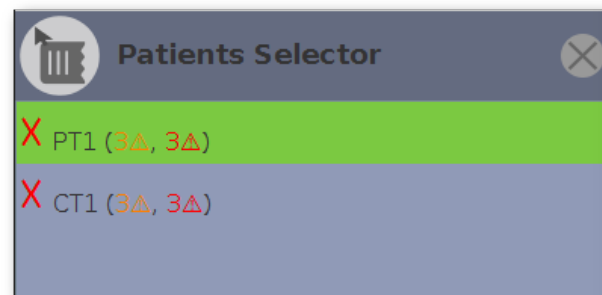
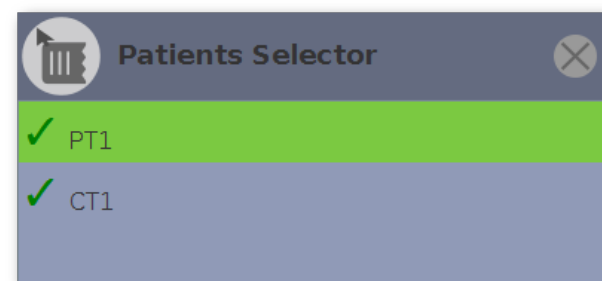
The list of patients defined in the script are indexed and displayed in the patient selector list. These are the file names that will be retrieved and displayed for the annotation process.

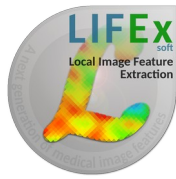
This list includes additional information, such as the progress of the scoring as indicated by red crosses and green checkmarks:

- Each patient whose annotation is not completed is shown with a red cross (X).
- All patients for whom the annotation is fully completed are shown with a green tick (✓).

To the right of the patient's name in this list are the missing data and warnings associated with the completeness of the annotation:

- The "warnings" (3⚠) are displayed in orange color and correspond to the number of fields not filled in but that are not mandatory.
- The "missing data" (3⚠) are displayed in red color and correspond to the mandatory fields that are not filled in.





# User guide

Labeling application  
— LIFEx —

## Properties form

### Properties form

The property entry form is built from the definitions given in the script. Each definition will be associated with a property and thus a dialog box.

The properties are grouped under a label. The user can add as many labels as he wants.

Navigation buttons (right and left arrows) are set on both sides of the label title. This will display the 1st label "label 0", then the 2nd label "label 1" and so on.

Note: Each property associated with a label will have to be filled in to validate the complete annotation of the patient (green tick displayed)







# Properties form

## Three verification levels

The annotation form includes 3 visual levels (three colors) of field verification

Comment 


no empty

Comment\* 

enter here your comment




Mandatory field with missing data (red line):  
The field is mandatory (title with \*) and is still empty

Comment\* 

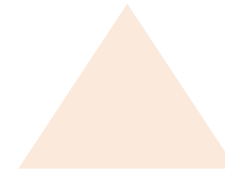
no empty



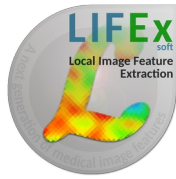
Fields properly filled in (green line):  
The fields are properly filled in (mandatory or not)

Comment 

enter here your comment



Field with warning (orange line):  
The warning that the field is not mandatory but is empty



# User guide

Labeling application  
— LIFEx —

## CONTENTS

- Introduction
- Labeling protocol
  - Patient selector
  - Properties form
- Step by step
  - Building the script
  - Executing a script
  - Reading the results file
- Available Field Types
  - Textfield, Checkbox, Dropdown...



# User guide

Labeling application

— LIFEx —

## Building the script

### What is a script?

The script file is a text file that can be easily read and edited.

It has the following file syntax: `key=value`

In this line: `script = labeling`

→ the key is 'script' and its value is 'labeling'. The set of properties that makes up the script is written using the same syntax.

### Understanding the different sections of the script.

The script is composed of several sections. The first Common section is fixed and must not be altered. It informs the application about the type of script that follows.

#### Fixed and common sections

`script = labeling`

→ Script is for labeling

The common and fixed sections of the script file are to be copied as is. Please, do not modify them. It makes it possible for the application to understand the rest of the script

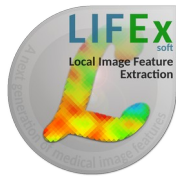
#### Result section

`output.props.file=.../LabelingResults`

→ this key indicates the output file in which the annotation results will be stored.

The path and filename must respect the syntax of the system on which you are running the Application.

In Windows system the syntax `C:/directory/home` should be used (the \ are translated into /).



# Building the script

## Properties section

`property#.title`

→ is the property title

(add \* when an answer is absolutely required for that property).

Example for an mandatory answer to a Tumor location question :

`property0.title=Tumor location*` for `property0`.

`property#.title.tooltiptext`

→ optional, is a tool tip text added to the property name

`property#.type`

→ mandatory [ radiobutton | checkbox | textfield | dropdown | voxelPick | roiPick ]

Example for checkbox of `property0`: `property0.type=checkbox` . See below « Available Field types » for more explanations regarding the different field types.

`property#.value`

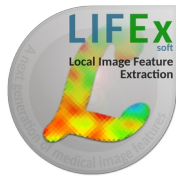
→ mandatory, is the list of property values to be selected from, several values are available using "|" as a character separator. Example of I,II,III,IV tumor grade for `property0`: `property0.value=I|II|III|IV`

`property#.value.tooltiptext`

→ optional, is a tool tip text added to value property, several texts can be included using "|" as a character separator.

Example: `property0.value.tooltiptext=Select this checkbox for I tumor grade|Select this checkbox for II tumor grade|Select this checkbox for III tumor grade|Select this checkbox for IV tumor grade`

'#' indicates the property number, it should start from 0, and should be incremented by 1 each time a new property is added.



# User guide

Labeling application

— LIFEx —

## Building the script

'#' indicates the property number, it should start from 0, and should be incremented by 1 each time a new property is added.

### Patient / Image section

#### From nifti files:

patient#.series#=.../file

→ mandatory, is the path to the image file to be read. It may be a nifti file.  
For instance, patient0.series0=/home/user/l1.nii to select the first patient (patient0) and read the first series (series0) for that patient from the /home/user\_directory.

#### From directory with dicom files:

patient#.series#=.../directory

→ mandatory, is the path to the image file to be read. It may be a directory that contains files. For instance: patient0.series0=/home/user/P1 to select the first patient (patient0) and read the first series (series0) in the directory P1 of /home/user

#### From DICOMDIR file:

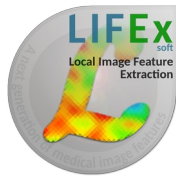
patient#.series#=.../DICOMDIR

→ mandatory, is the path to the DICOMDIR file to be read.

#### Example:

patient0.series0=/home/user/DICOMDIR

→ to select DICOMDIR file



## Optional section

directory of ROI files -> optional

`output.roi.file=.../roi`

→ is the directory where ROI files will be saved.

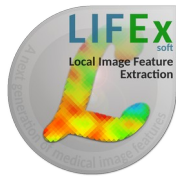
This is optional because if it is not defined, the directory set in the application will be reused (see Settings \ FrameGui \ DataDir). The path and filename must respect the syntax of the system on which you are running the application. With Windows system, the syntax `C:/directory/home` should be used for which the `\` are translated into `/`.

Window Leveling -> optional

`all.patient.series0.window.width=10`

`all.patient.series0.window.level=5`

→ It is possible to voluntarily set the windows/Leveling at some values. To do this, fill in these values in the corresponding fields



## Color map ; available options :

→ It is possible to set the color palettes on playback according to each frame (frame0, frame1 & frame2). The name of the color palette must be defined. The possible choices are listed above :

```
MONOCHROME1 || MONOCHROME1_PVALUES || MONOCHROME1_PVALUES0  
MONOCHROME2 || MONOCHROME2_PVALUES || MONOCHROME2_PVALUES0  
RAINBOW || RAINBOW_PVALUES || RAINBOW_PVALUES0  
RAINBOW_INVERSE || RAINBOW_INVERSE_PVALUES || RAINBOW_INVERSE_PVALUES0  
RAINBOW_NEW || RAINBOW_NEW_PVALUES || RAINBOW_NEW_PVALUES0  
RAINBOW_NEW_INVERSE || RAINBOW_NEW_INVERSE_PVALUES || RAINBOW_NEW_INVERSE_PVALUES0  
HEAT || HEAT_PVALUES || HEAT_PVALUES0  
HEAT_INVERSE || HEAT_INVERSE_PVALUES || HEAT_INVERSE_PVALUES0  
color map in first 3 frames -> optional, default: MONOCHROME2
```

```
frame0.series.color=MONOCHROME2  
frame1.series.color=MONOCHROME2  
frame2.series.color=MONOCHROME2
```

See LIFEx-shortcuts (color palettes) for more explanations



# User guide

## Labeling application

— LIFEx —

# Example of script

Here is a minimal script :

### LabelingScript.txt

```
# Common
script = labeling
output.props.file=.../labelingResults

# properties
property0.title = Organ*
property0.type = dropdown
property0.value = 1|2|3|4

property1.title = Comment
property1.type = textfield

property2.title = ROI
property2.type = roiPick

property3.title = Nodule*
property3.type = radiobutton
property3.value = Yes|No

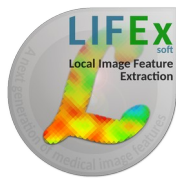
property4.title = Tumor grade*
property4.type = checkbox
property4.value = I|II|III|IV

property5.title = Voxel pick
property5.type = voxelPick

# image or directory of patients
patient0.series0=.../img/PT1
patient1.series0=.../img/CT1
```







# User guide

## Labeling application

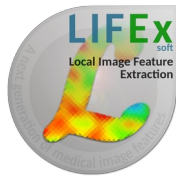
— LIFEx —

# Executing a script

The screenshot displays the LIFEx application interface. On the left, there is a 'Patients Selector' panel with a 'Drop here' area for loading images or scripts. A script file is being dragged and dropped into this area. The main window shows a patient's data, including 'GEMINI TF TQP 16', '33641', and 'WB\_CTAC Body'. A PET scan image is visible in the center. On the right, there is a 'ROI tools' panel with various tools for measuring, editing, and thresholding ROIs. The bottom status bar shows technical details like 'Actual Frame Duration: 75012 ms' and 'Radiopharmaceutical Start Time: 2015-09-18T11:43:00'.

The script is executed by drag and drop the script file in the patient loading area.

The protocol then starts executing automatically.



# User guide

## Labeling application

— LIFEx —

# Reading the results file

The results are stored in the file specified by the key 'output.props.file' : example `labelingResults.props`  
This file has the same syntax as that of the script, i.e. a key=value pair repeated on each line.

Example of results file:

```
#LIFEx6.00 Labeling properties
#Thu Apr 16 19:40:08 CEST 2020
patient0.label0.property0.value=Stomach
patient0.label0.property1.value=comment
patient0.label0.property2.value=
patient0.label0.property3.value=Yes
patient0.label0.property4.value=III
patient0.label0.property5.value=t0 z0 y61 x62
patient1.label0.property0.value=Lung
patient1.label0.property1.value=another comment
patient1.label0.property2.value=
patient1.label0.property3.value=No
patient1.label0.property4.value=IV
patient1.label0.property5.value=t0 z0 y66 x23
```

Explanation of the example file:  
This result file contains the  
annotations  
for two patients (`patient0` & `patient1`).

Each patient has only one defined label (`label0`),  
and the values of the 6 properties (`property0...5`)  
are recorded without their headings (key).



The opening of this file in an excel file is possible,  
by giving the symbol '=' and ';' as delimiter



# Reading the results file

The results are stored in a second csv file : example `labelingResults.csv`

Example of results file:

PatientIndex	labelIndex	Organ*	Comment*	ROI	Nodule*	Tumor grade*	Voxel pick
0	0	2	comment of label 0		Yes	I	t0 z105 y74 x88
1	0	3	new comment		Yes	II	

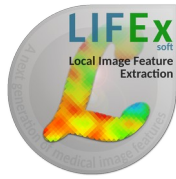
Explanation of the example file:

This result file contains the  
annotations

for two patients (**patient0** & **patient1**).



The opening of this file in an excel file is possible  
by giving the symbol ',' as delimiter

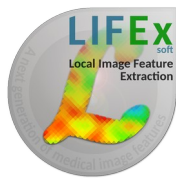


# User guide

Labeling application  
— LIFEx —

## CONTENTS

- Introduction
- Labeling protocol
  - Patient selector
  - Properties form
- Step by step
  - Building the script
  - Executing a script
  - Reading the results file
- Available Field Types
  - Textfield, Checkbox, Dropdown...



# User guide

Labeling application  
— LIFEx —

## Available Field Types

Several types of fields are available to define a complete form. These fields are taken from the main generic fields of programming languages and are:

### Checkbox

The checkbox is shown as a square box that is ticked (checked) when activated. Checkboxes are used to let a user select one or more options among a limited number of choices

property#.type=checkbox

### Voxel pick

The Voxel pick component is a complete form control including a label, input and help text. Voxel pick lets users enter coordinate voxel by mouse click

property#.type=voxelpick

### Textfield

The TextField component is a complete form control including a label, input and help text. Text fields let users enter and edit text

property#.type=textfield

### ROI pick

Dropdown for ROI is a list of ROI that is hidden until you choose to look at it

property#.type=roipick

### Radiobutton

Radio buttons are normally presented in radio groups (a collection of radio buttons describing a set of related options). Only one radio button in a group can be selected at the same time

property#.type=radiobutton

### Dropdown

Dropdown is a list of choices appearing below a menu title when it is selected, and remaining until used or dismissed

property#.type=dropdown