NEUBIAS Academy
Online Course

ImageJ/Fiji Macro Language

Anna Klemm - anna.klemm@it.uu.se
BioImage Informatics Facility, SciLifeLab, Sweden

SciLifeLab

With material from:
Fabrice Cordelières, Bordeaux Imaging Center, Bordeaux, France
Giovanni Cardone, MPI Biochemistry, Martinsried, Germany
Ofra Golani, Weizmann Institute of Science, Israel
NEUBIAS material

neubias network of european bioimage analysts

Outline of This Session

- Biological Data Set and Image Analysis Problem
- How can we ”talk” to Fiji? - Macro Recorder, Built-in Macro Function
- Step-by-Step Workflow

The Cell Atlas/Human Protein Atlas

Cell Atlas Aim:

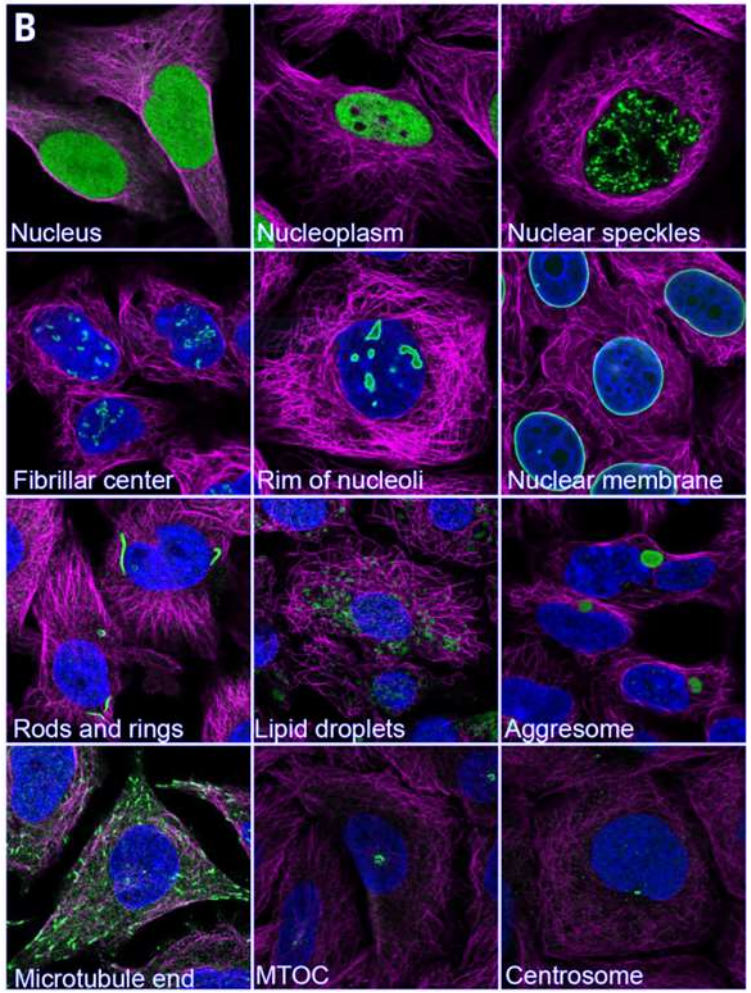
Determine the subcelluar location of all cellular proteins.

Experimental Methods:

- Antibody generation against 12.000 human proteins
- Immunostaining, 22 cell lines
- Automated confocal microscopy
- → 82.152 images

Image Analysis Aim:

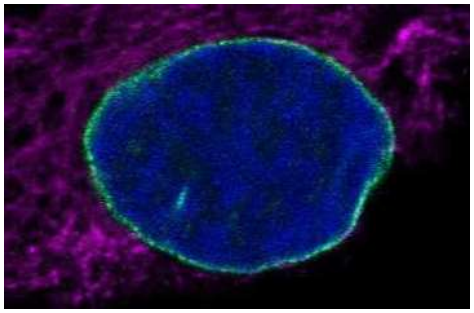
- Mapping 12.000 human proteins to 30 subcellular structures

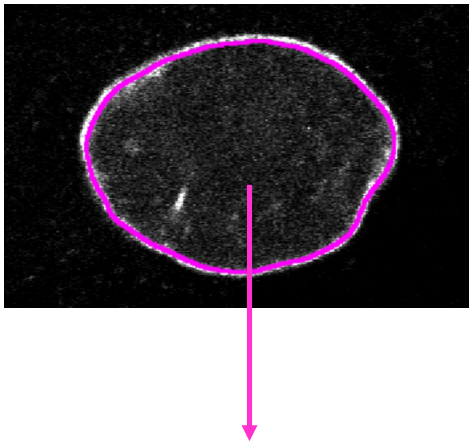


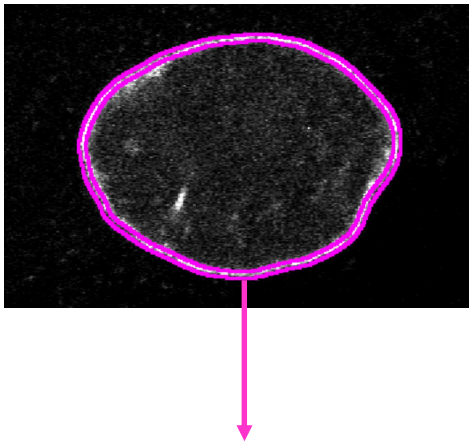
Adapted from Thul, P.J. et al. (2017). A subcellular map of the human proteome. Science *356*.

The Aim: Quantify Signal Accumulation within the Nuclear Membrane

Image source: Human Protein Atlas
v19.proteinatlas.org/ENSG00000113368-LMNB1







$$Mean\ Intensity\ Nucleus$$

$$Mean\ Intensity\ NucMembrane$$

Dataset:

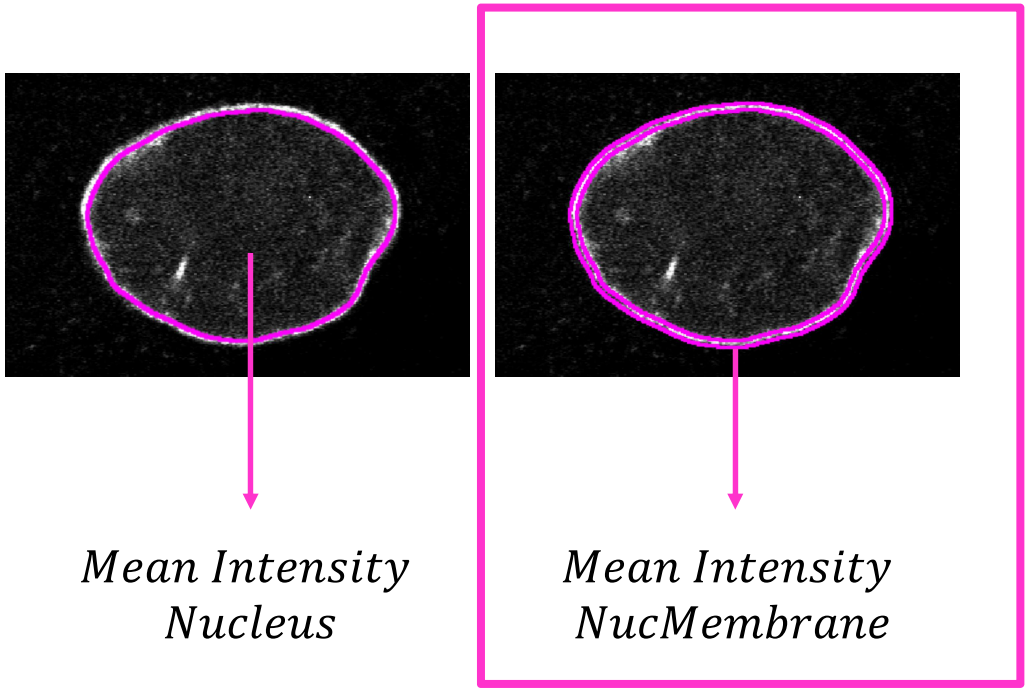
- Subset of The Cell Atlas (Human Protein Atlas)
- 3 color stack: microtubules (magenta), protein detected by antibody (green), nuclei (blue)

$$Ratio = \frac{Mean\ Intensity\ NucMembrane}{Mean\ Intensity\ Nucleus}$$

The Aim: Quantify Signal Accumulation within the Nuclear Membrane

Image source: Human Protein Atlas

v19.proteinatlas.org/ENSG00000113368-LMNB1



$$Mean\ Intensity\ Nucleus$$

$$Mean\ Intensity\ NucMembrane$$

$$Ratio = \frac{Mean\ Intensity\ NucMembrane}{Mean\ Intensity\ Nucleus}$$

Results

File Edit Font Results

	Label	Mean
1	signal:0001-0041	26957.406
2	signal:0002-0291	20013.618
3	signal:0003-0320	38092.890
4	signal:0004-0670	18741.716
5	signal:0005-0696	19940.679
6	signal:0006-0677	16445.010
7	signal:0007-1198	20677.366
8	signal:0008-1168	20005.914
9	signal:0009-1250	24444.675
10	signal:0010-1457	20037.129
11	signal:0011-1651	26454.839
12	signal:0012-1788	5380.207
13	signal:0013-1832	24655.655
14	signal:0014-1945	15790.756
15	signal:0015-1935	18773.019
16	signal:0016-2019	17444.867
17	signal:0017-2021	9947.993

Exercise: Find a Workflow

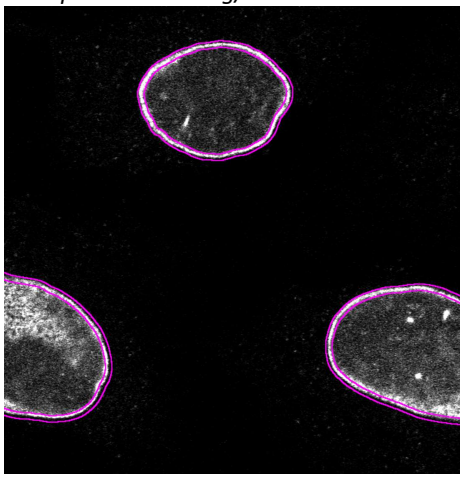
Task: Create selections (ROIs) around nuclei and the nuclear membrane

- Create selections for each nucleus: Threshold, Connected Component Analysis
- Try for one nucleus: create a selection around the nuclear membrane.

Image:

- CellAtlas_Subset/711_D6_1.tif

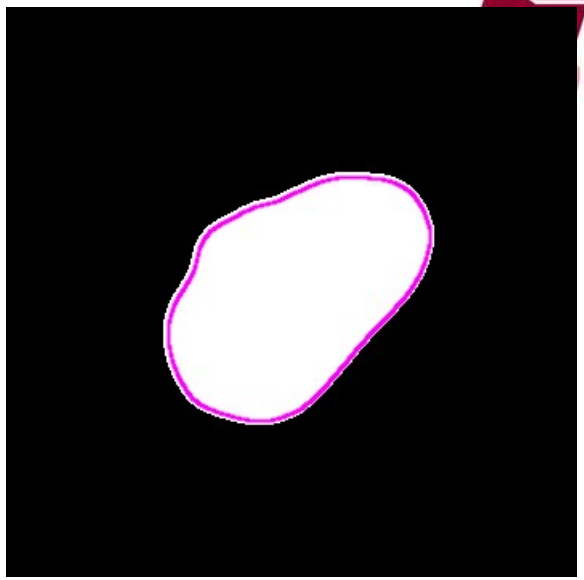
Image source: Human Protein Atlas
v19.proteinatlas.org/ENSG00000113368-LMNB1



What would be your workflow?

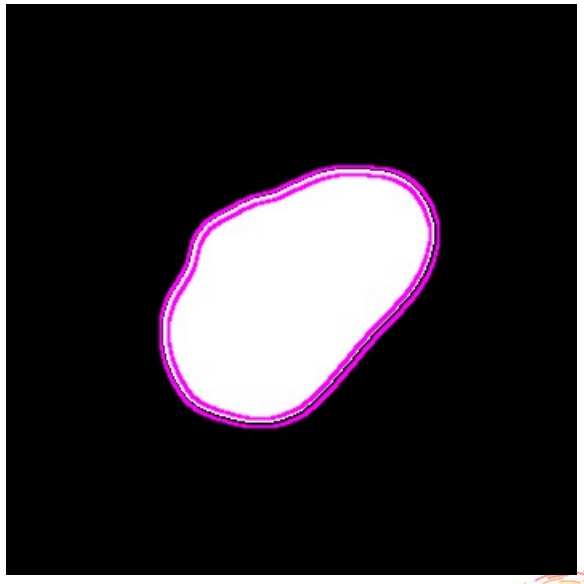
Create a selection for each nucleus:

1) Split channels
2) Prefilter and segment nuclei
3) Run „Analyze Particles“ to get ROIs of single Nuclei



Try for one nucleus: ROI of the nuclear membrane

1) Select the ROI of one nucleus
2) Decrease the size of the ROI by *Edit>Selection>Enlarge*
3) *Edit>Selection>Make Band…* to create ROI of the envelope



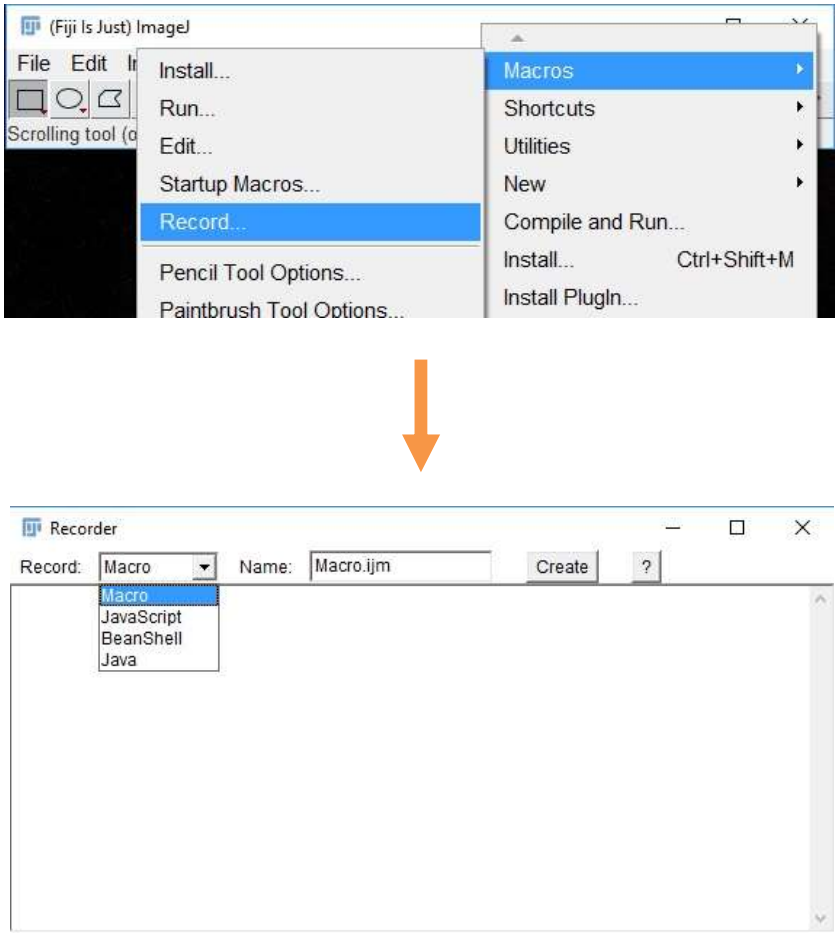
We now want to automize this process

IMAGEJ MACRO LANGUAGE

Recorder

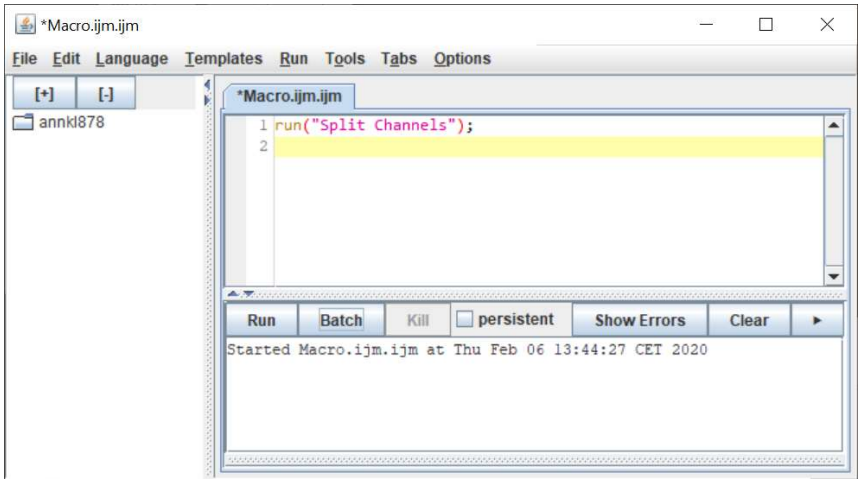
Open record window:

- Plugins > Macros > Record...
- Choose language – if needed



Let's try with only one command: Split Channels

Recorder



Script Editor

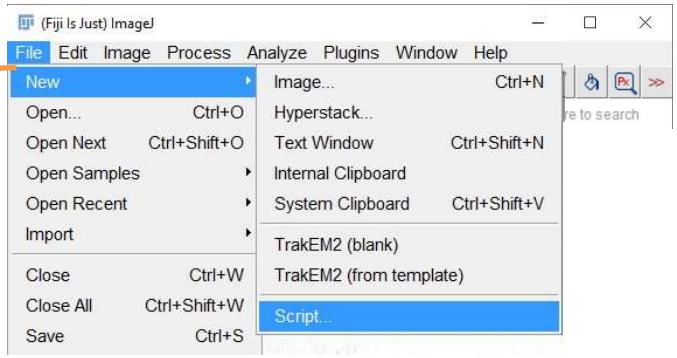
Recorder:

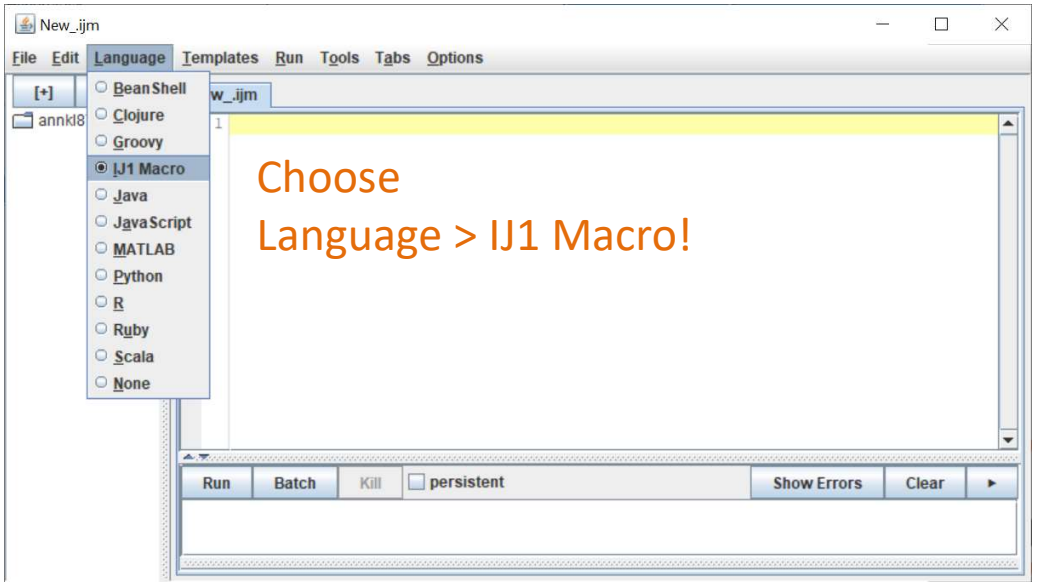
- Discover commands
- Window can be edited, copied, pasted, cut etc.

Script Editor:

- For building a script
- Has color-coding, code-completion, run-option etc.

How to start a new Script





Choose
Language > IJ1 Macro!

Importance of recorder

- Discover commands
- Get arguments for specific functions

But also:

- Record and save your workflow – for documentation and reproducibility!

Exercise: Record the Workflow

Task:

- Open the recorder and record the discussed workflow.
- Discuss open questions within the group.

Image:

- Any image from CellAtlas_Subset

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Filter nuclear image and make binary image

Retrieve nuclei boundaries

Retrieve the nuclear envelope boundary

Measure

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

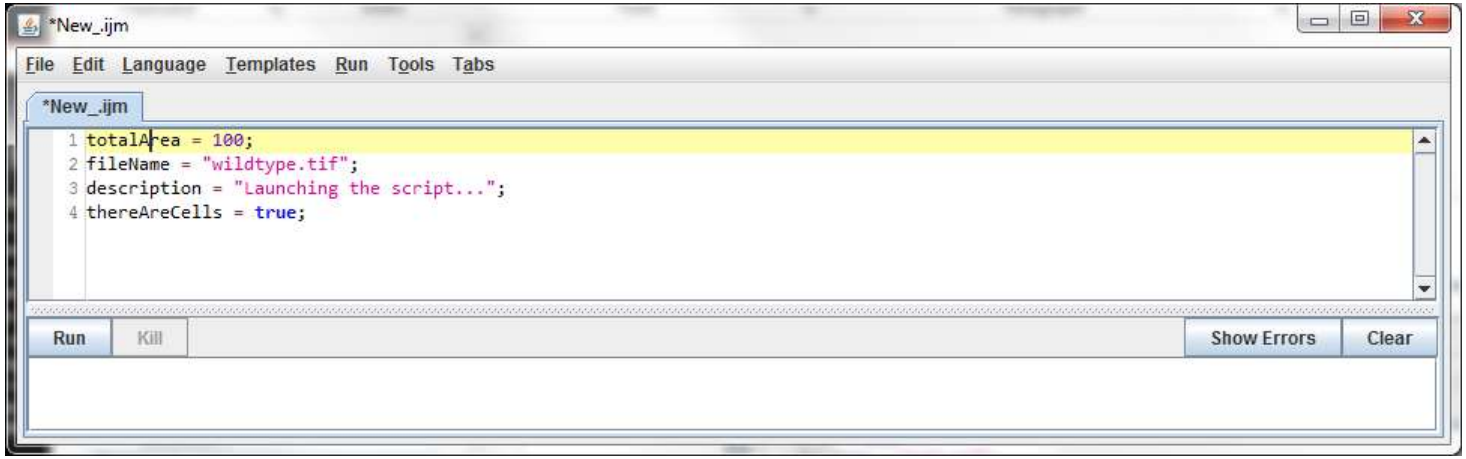
```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Programming Basics I

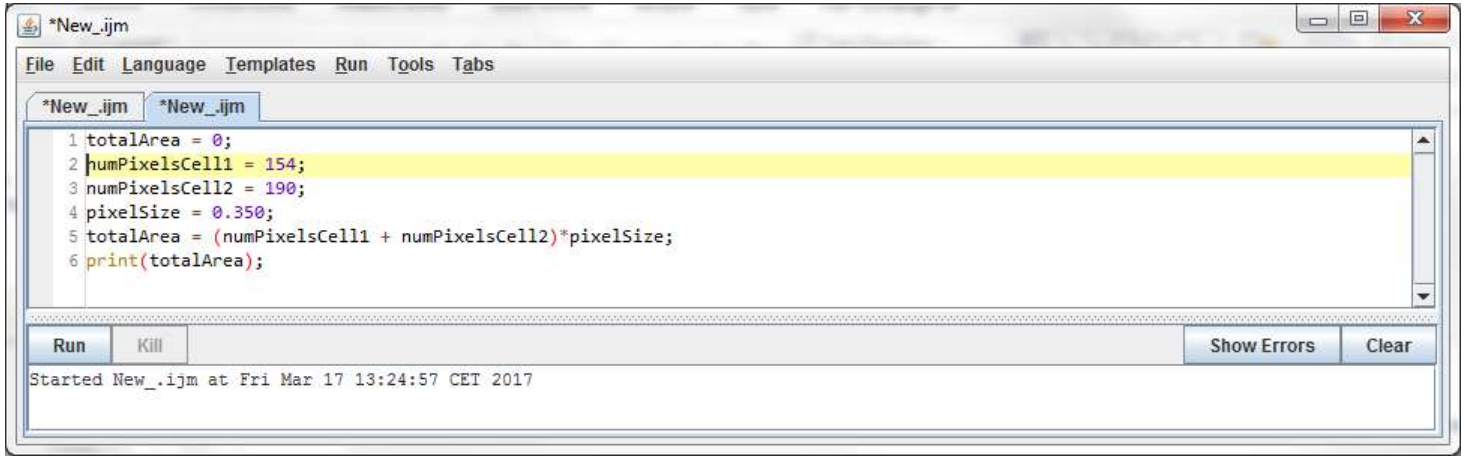
VARIABLES

Variables: definition



- Can hold numbers or phrases/strings, but only one at a time
- Used whenever a value is used many times inside the script
- You define a variable by assigning it some content
- Variable name is on the left followed by an equal sign followed by the item (or items) being assigned
- Variable names can only start with characters

Numeric Variables: assignment



- Content of numeric variables can be modified using mathematical operations
- After an assignment, the previous content (if any) is forgotten
- Good practice is to use ***d2s*** (decimal to string) when printing numbers

  e.g. print(d2s(totalArea));

Catch-up/variables_assignment.ijm

String Variables: concatenation

string_concatenation.ijm

```
1 text1 = "a";
2 text2 = "Hello";
3 text3 = "Hello everybody!";
4 text4 = " ";
5
6 text5 = text1 + text2;
7 print(text5);
8 print(text1 + text2);
9 print(text2+ "world!");
10 print(text2 + text4 + "world!");
```

string_numbers_concatenation.ijm

```
1 number1 = "2";
2 number2 = "3";
3 print(number1+number2);
4
5 number3 = 2;
6 number4 = 3;
7 print(number3+number4);
8
9 text = "image";
10 print(text + number3);
```

What about this?

string_concatenation.ijm

string_numbers_concatenation.ijm

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Introduction of the next Exercise

- Define a string variable with the name of the image
- Build selectWindow using the variables

```
run("Split Channels");

selectWindow("C1-711_D6_1.tif");
rename("nuclei");

selectWindow("C2-711_D6_1.tif");
rename("signal");
```

Exercise: Usage of Variables

- Replace the highlighted text using the variable *title.*
- Check out slide 19 (String Variables: concatenation) for help.

```
title = "711_D6_1.tif";

run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");

selectWindow("C2-711_D6_1.tif");
rename("signal");
```

Technical point:

Structuring the code using comments

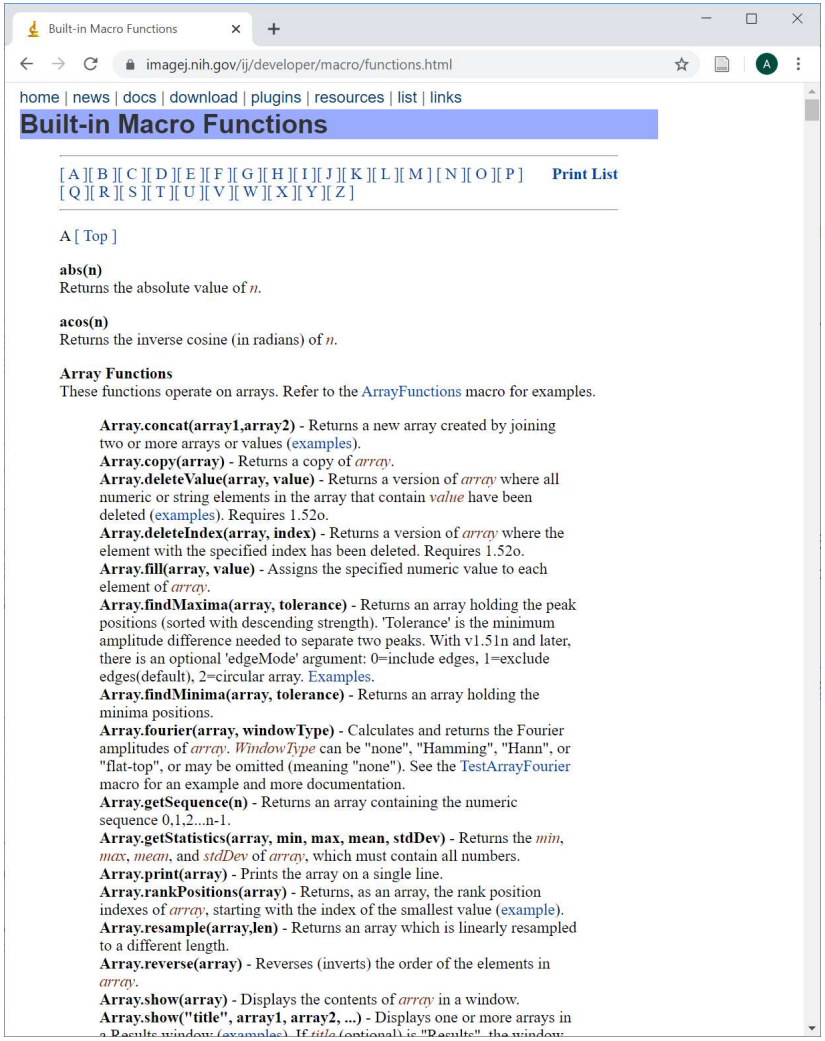
- Comments are non-interpreted elements of code
- They help structure the code
- They help collaborators interpret the original analyst's intentions
- Comments are introduced either by // of surrounded by /* */:

```
//This is a short comment

/*
        This is a very long comment, spanning over
        multiple lines, allowing line breaks
*/
```

We now know what variables are but..

HOW TO GET THE NAME OF AN IMAGE AUTOMATICALLY?

Built-in Macro Functions



- Not everything is recorded. Much more functions can be found at: https://imagej.nih.gov/ij/developer/macro/functions.html
- Tip: do a page-search (CTRL+F)

Step 1:
Normalise the data name

Solution

Step_01_SplitAndRename.ijm

File Edit Language Templates Run Tools Tabs Options

Step_05_Measure.ijm | Step_05_Measure.ijm | Step_01_SplitAndRename.ijm

```
8  //Step1: Getting image information + Normalise the data name
9  //get general information
10 title = getTitle();
11
12
13 //split channels and rename them
14 run("Split Channels");
15 selectWindow("C1-" + title);
16 rename("nuclei");
17 selectWindow("C2-" + title);
18 rename("signal");
19
20
21
22
```

Catch-up/Step_01_SplitAndRename.ijm

More about the Built-in Macro Functions

Stack.setChannel(1); Function with input

getTitle(); Function with output;

nameOfMyImage = getTitle(); output is assigned to a variable

getDimensions(width, height, channels, slices, frames);

Output is assigned to variables within the brackets

Exercise: Built-in Macro Functions

Task 1: Catch-up with the script

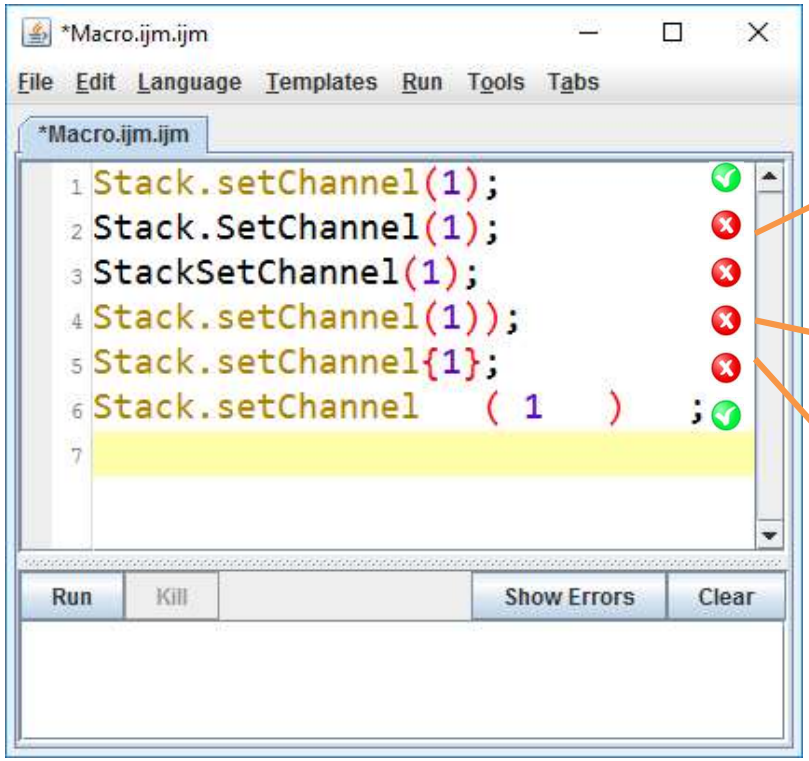
- include the getTitle() function.

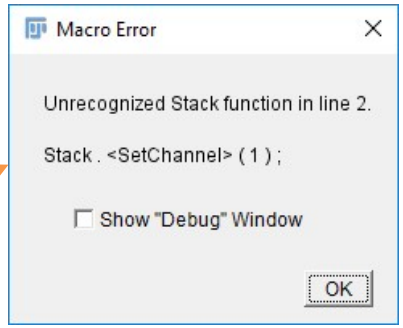
Task 2: explore the built-in macro functions.

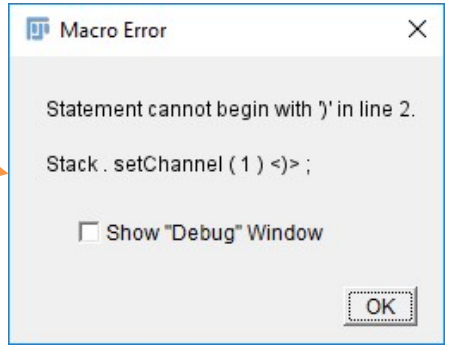
- open a new script, set language to IJ1 macro
- What happens when you run `getDimensions(channels, height, width, slices, frames)`?
- Use the `print()` function to explore the content of the variables channels and width.
- Check the usage of the getDimensions function either using code autocompletion or on the “built-in macro function” website.

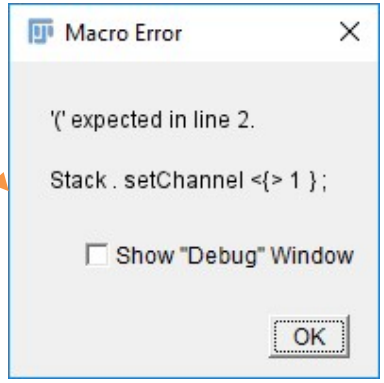
The script editor supports you with colors and error messages.

Color-coding in Script-Editor



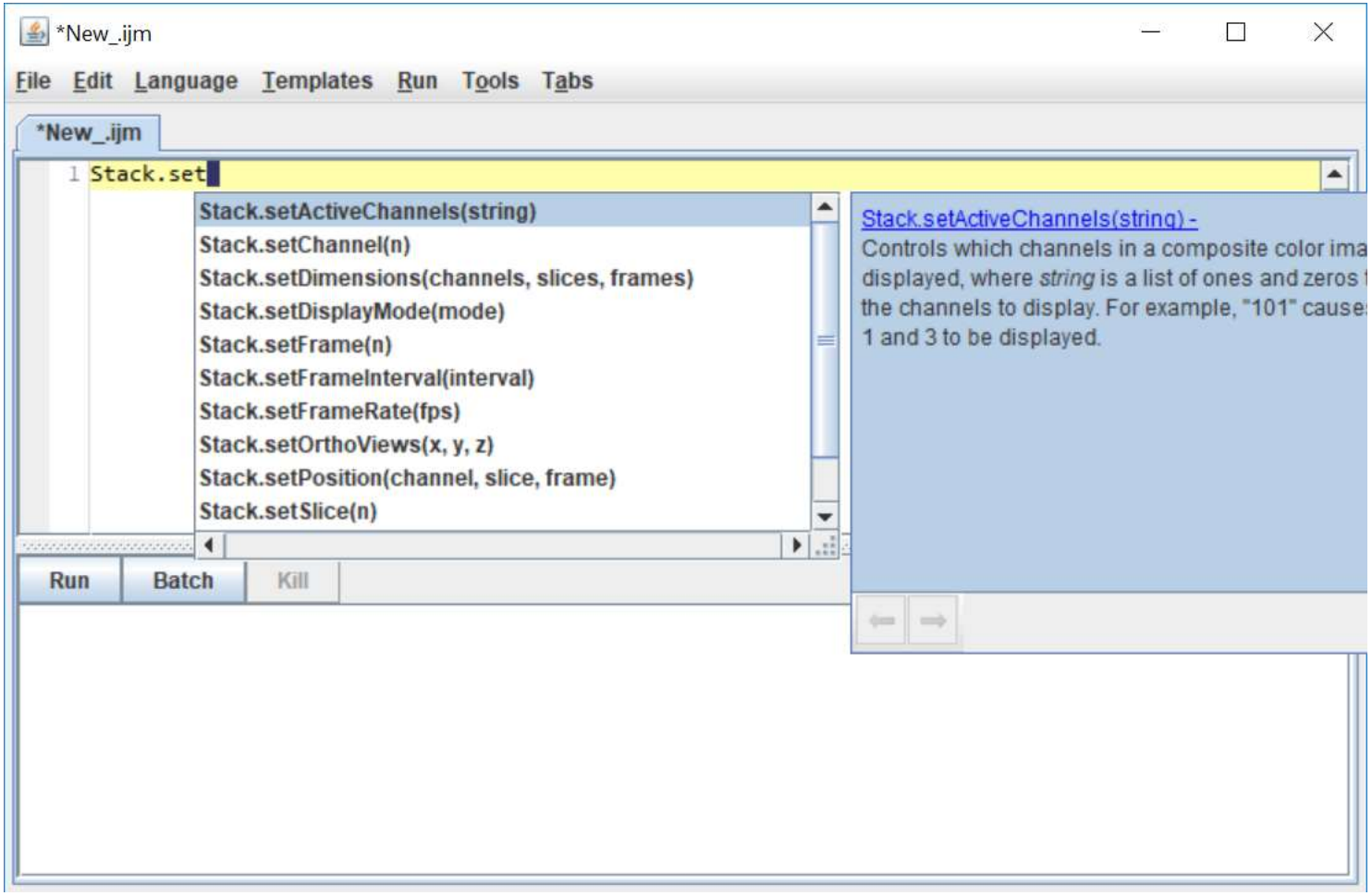






Read the error messages!

Auto-completion helps to avoid common mistakes



Workflow: How will we tackle the problem ?

Step 1	**Getting image information + Normalise the data name**	*Structuring the code, Recording basic operations, IJ macro functions' structure, Using simple variables*
Step 2	**Prefilter nuclear image and make binary image**	*Some useful shortcuts*
Step 3	**Retrieve the nuclei's boundaries**	*Using Analyze Particles*
Step 4	**Retrieve the nuclear envelope's boundaries**	*For-Loops, Manipulating ROIs*
Step 5	**Measure mean intensity and save the result**	*Saving of data; Extracting paths*

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

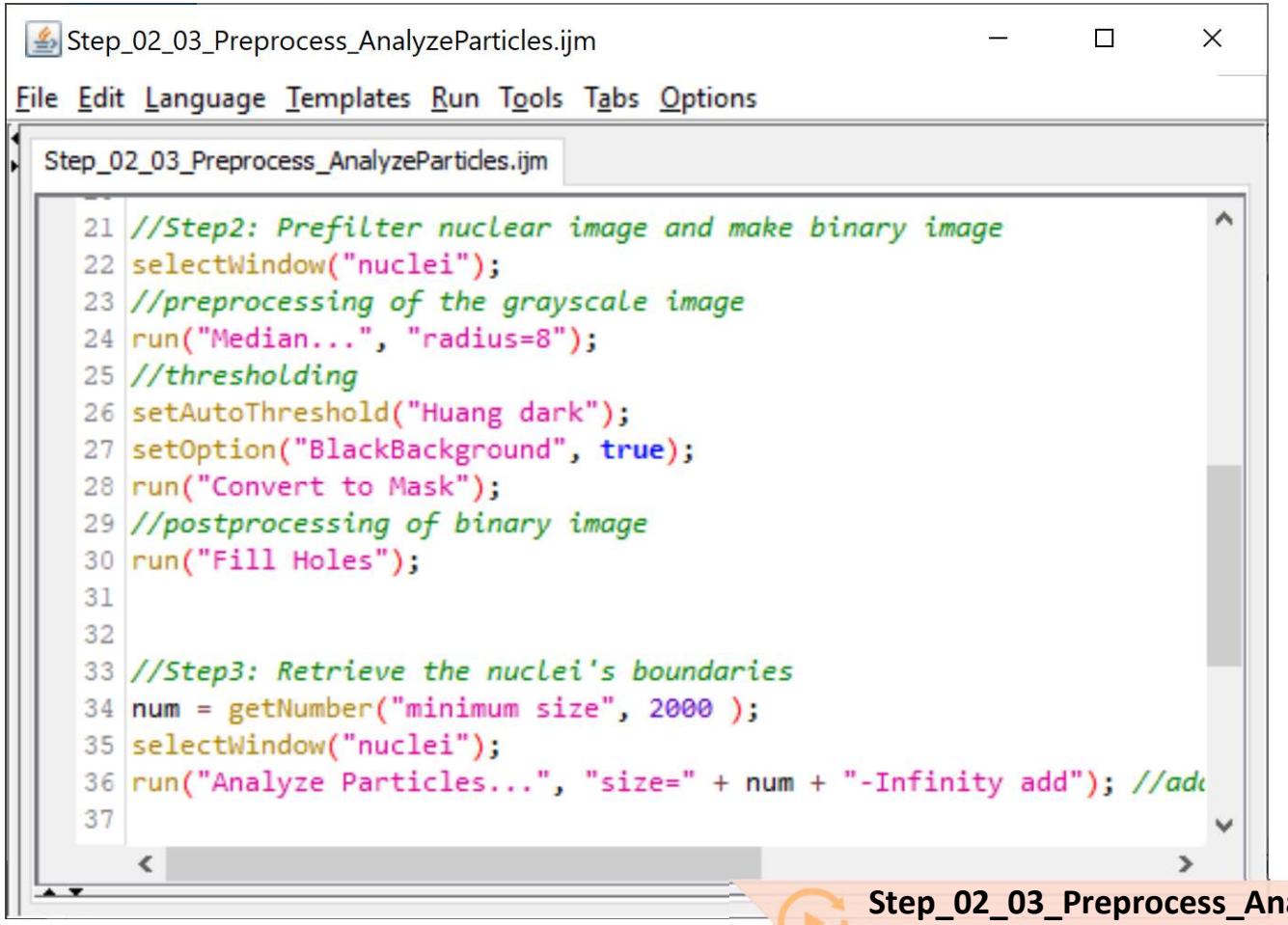
Record: Macro Name: Macro.ijm Create ?

```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Filter nuclear image and make binary image

Retrieve nuclei boundaries

Step 2 + 3:
Pre-process the nuclear image



Step_02_03_Preprocess_AnalyzeParticles.ijm

Exercise: Preprocessing, Asking for User-Input

Task 1: Catch-up with the script

As shown in the slide before:

- Insert the preprocessing and segmentation steps to your code (median filter, thresholding, fill holes)
- Use the getNumber function to ask the user for a minimum size of the nuclei in pixel

Workflow: How will we tackle the problem ?

Step 1	**Getting image information + Normalise the data name**	*Structuring the code, Recording basic operations, IJ macro functions' structure, Using simple variables*
Step 2	**Prefilter nuclear image and make binary image**	*Some useful shortcuts*
Step 3	**Retrieve the nuclei's boundaries**	*Using Analyze Particles*
Step 4	**Retrieve the nuclear envelope's boundaries**	*For-Loops, Manipulating ROIs*
Step 5	**Measure mean intensity and save the result**	*Saving of data; Extracting pathes*

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

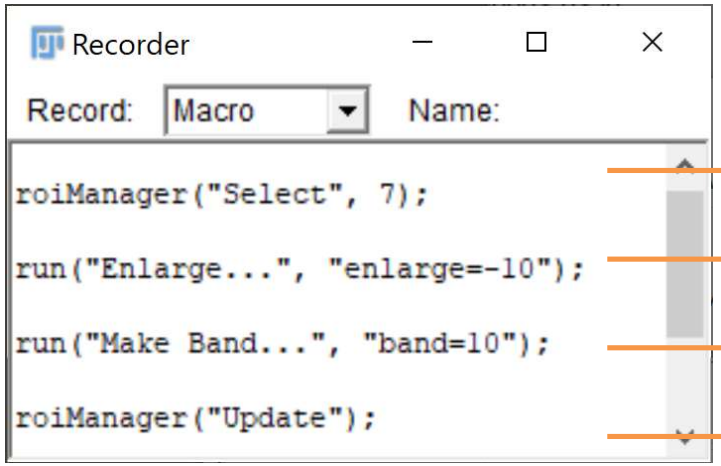
```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Filter nuclear image and make binary image

Retrieve nuclei boundaries

Step 4:
Retrieve the nuclear envelope's boundaries



Selects only one specific ROI

Shrinks the ROI

Creates donut-Shape ROI

Replaces the active ROI with the donut

We need to loop this code over all ROIs!

Programming Basics

CONTROL STRUCTURES FOR-LOOPS

Technical point

Loops

Definite loop

- Known number of iterations
- 3 arguments:
  - Initialisation
  - Condition for loop entry, as a boolean
  - Iteration

Indefinite loops

A priori

- Test performed **BEFORE** instructions are executed

A posteriori

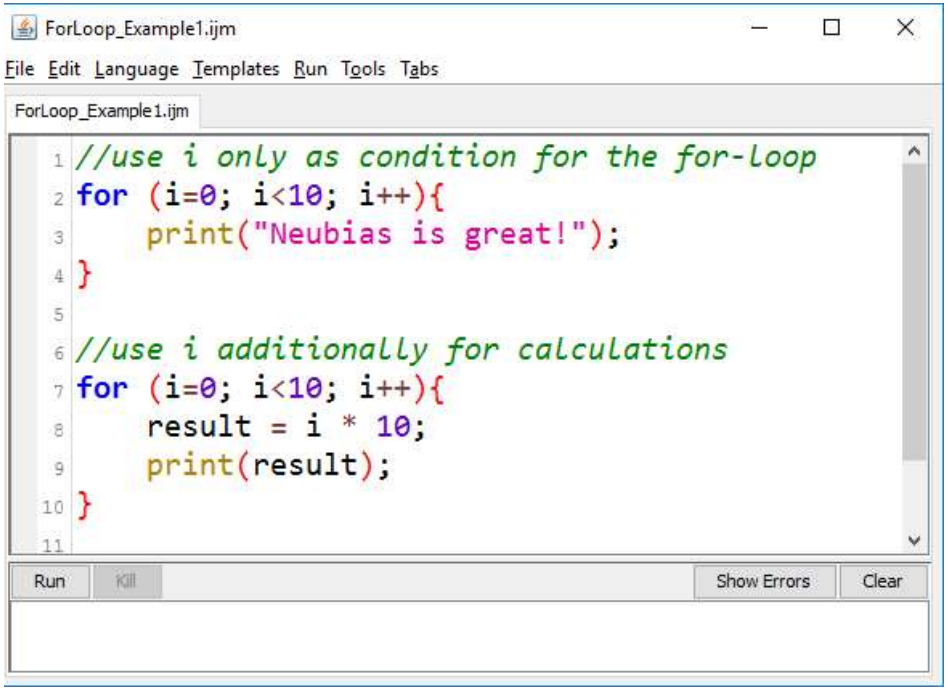
- Instructions always executed at least once
- Test performed **AFTER** instructions have been executed

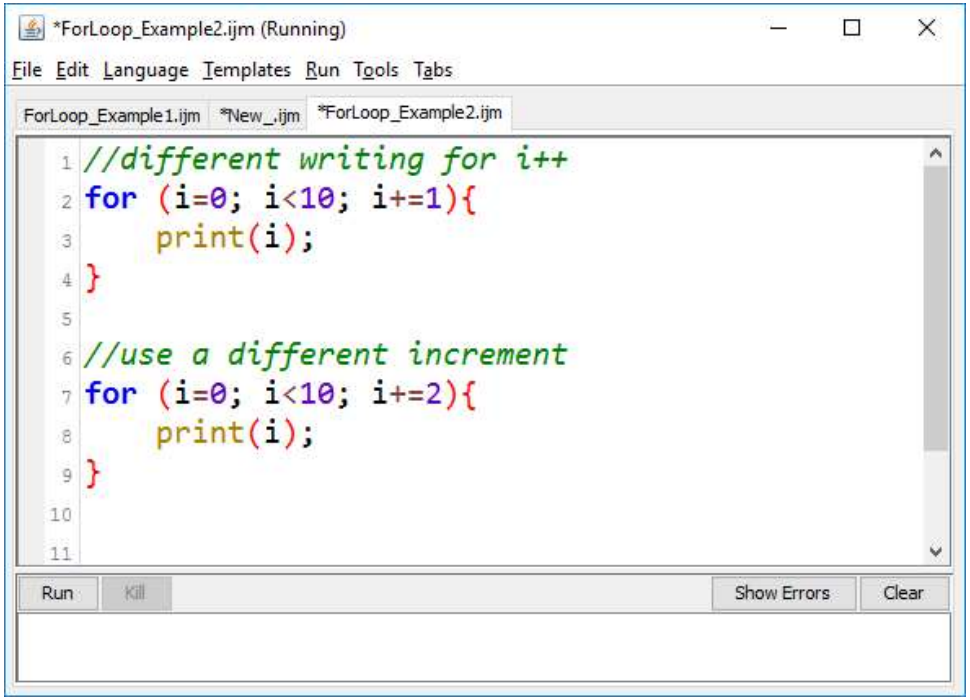
```
for(i=0; i<10; i++){
        //Instruction 1

        //Instruction 2

        //Instruction 3

}
```

```
i=0;
while(i<10){
        //Instruction 1
        //Instruction 2
        //Instruction 3

        i++;
}
```

```
i=0;
do{
        //Instruction 1
        //Instruction 2
        //Instruction 3

        i++;
} while(i<10)
```

Programming Basics: For-Loops





Catch-Up/ForLoop_Example1.ijm

Catch-Up/ForLoop_Example2.ijm

For-Loops Exercise

Find **four different ways** to modify the code below to print „Hello!“ **only once** instead of 10 times.

```
for (i=0; i<10; i++){

   print("Hello!");

}
```

Exercise: Looping

Task:

Modify the recorded code to apply it

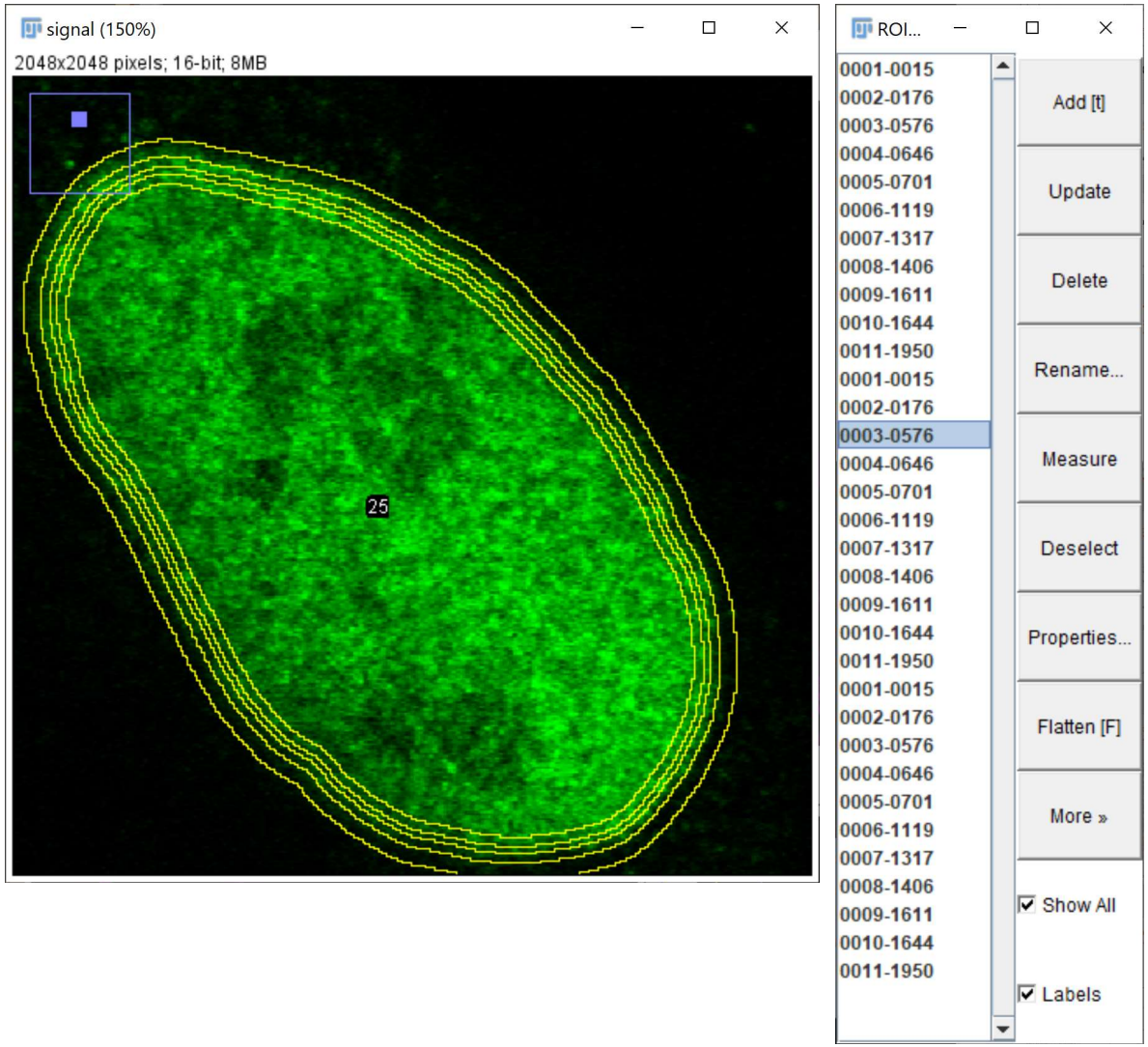
to all ROIs of the ROI-Manager.

```
roiManager("Select", 7);
run("Enlarge...", "enlarge=-4");
run("Make Band...", "band=7");
roiManager("Update");
```

Hints:

- Search for an built-in function giving the **total number of ROIs.**
- Index of the first ROI is **0**!
- **Close the ROI-manager** each time before re-running your code again and again.

Close the ROI-manager before Running again



For-Loop run over all ROIs, also “old” ones → Close the ROI-manager!

We will **automatically** close the ROI-manager soon…

Step 4:
Retrieve the nuclear envelope's boundaries

Solution

*Step_04_ForLoop.ijm

File Edit Language Templates Run Tools Tabs Options

*Step_04_ForLoop.ijm

```
41
42 //Step4: Retrieve the nuclear envelope's boundaries and save them in the ROI-Manager
43 numberOfNuclei = roiManager("count");
44 for(i=0; i<numberOfNuclei; i++){
45     roiManager("Select", i);
46     run("Enlarge...", "enlarge=-4");
47     run("Make Band...", "band=7");
48     roiManager("Update"); //original nucleus-ROI is replaced by nuclear envelope ROI
49 }
50
51
```

Catch-up/Step_04_ForLoop.ijm

Workflow: How will we tackle the problem ?

Step 1	**Getting image information + Normalise the data name**	*Structuring the code, Recording basic operations, IJ macro functions' structure, Using simple variables*
Step 2	**Prefilter nuclear image and make binary image**	*Some useful shortcuts*
Step 3	**Retrieve the nuclei's boundaries**	*Using Analyze Particles*
Step 4	**Retrieve the nuclear envelope's boundaries**	*For-Loops, Manipulating ROIs*
Step 5	**Measure mean intensity and save the result**	*Saving of data; Extracting paths*

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Filter nuclear image and make binary image

Retrieve nuclei boundaries

Retrieve the nuclear envelope boundary

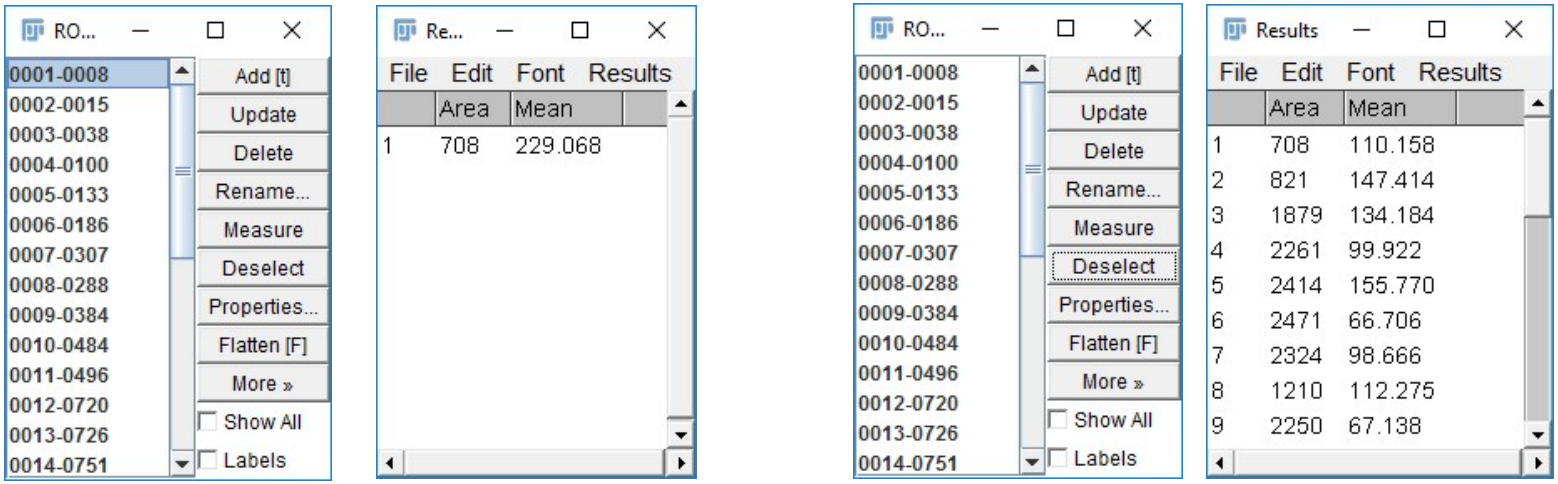
Measure

Step 5:

Make the measurement and save the result

Hint:

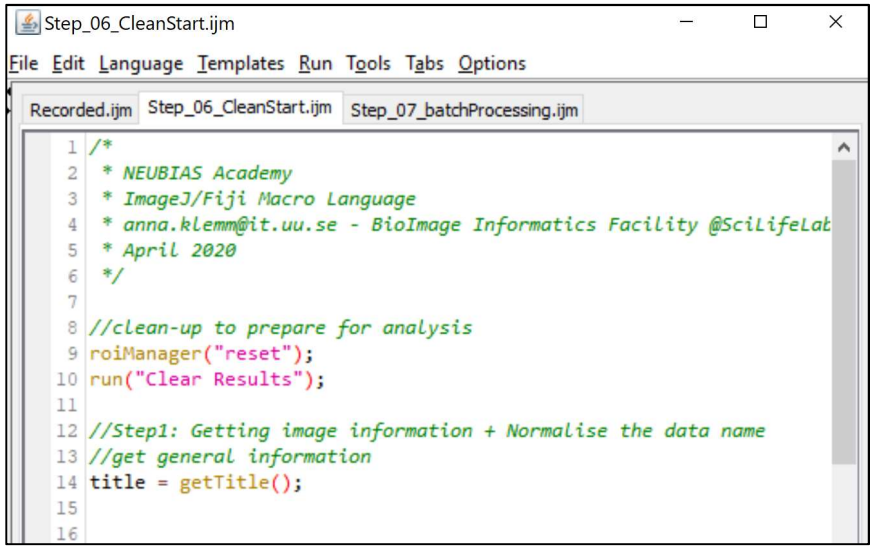
If no ROI is selected in the ROI Manager, `roiManager("Measure");` will measure **all ROIs**.



Empty ROI Manager and Empty Results window

Make sure to start with an empty ROI-manager and an empty Results table!

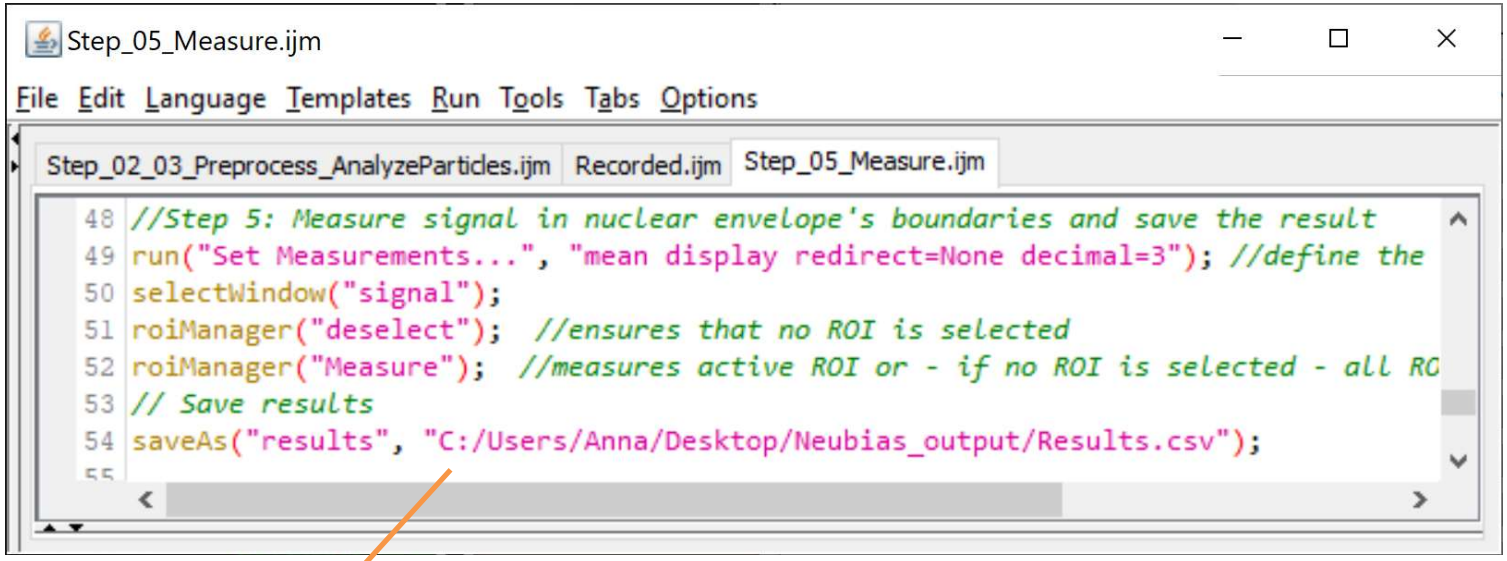
```
roiManager("reset");
run("Clear Results");
```



Exercise: Catch-up

Task:

- Insert the missing steps to your code
- For saving: Insert a path fitting to you local computer.



Attention: Use / or \\

Catch-Up/Step_05_Measure.ijm

EXTRA-STEPS

Extra I: Batch Processing

For-Loop over all images

Step 1	**Getting image information + Normalise the data name**	*Structuring the code, Recording basic operations, IJ macro functions' structure, Using simple variables*
Step 2	**Prefilter nuclear image and make binary image**	*Some useful shortcuts*
Step 3	**Retrieve the nuclei's boundaries**	*Using Analyze Particles*
Step 4	**Retrieve the nuclear envelope's boundaries**	*For-Loops, Manipulating ROIs*
Step 5	**Measure mean intensity and save the result**	*Saving of data; Extracting pathes*

Extra I: Batch processing – getFileList

- Example script:

fileList is an **array**

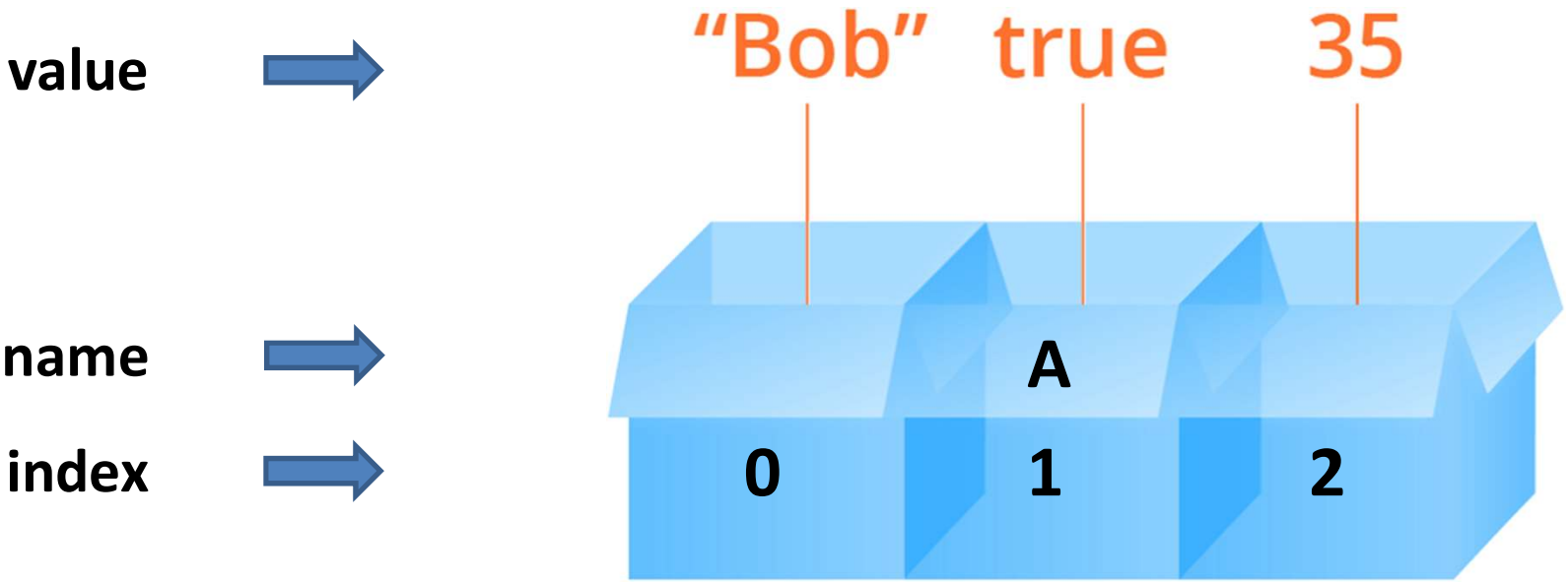
```
input_path = getDirectory("input files");

fileList = getFileList(input_path);

for (i=0; i<fileList.length; i++){
    print(fileList[i]);
}
```

number of images in the array (=directory)

Extra_02_getFileList_Example.ijm

Prints filename on position i in fileList.
Attention: 1st position has index 0!

Arrays are multi-compartment boxes with a label



- Can hold numbers or phrases/strings, different type for each compartment/cell
- Can only be 1D
- Used to keep similar data together (e.g. points in plot, measures at different slices, etc.)
- Can be defined everywhere inside a script
- Index starts at 0 and ends at N-1, where N is the size of the array

Arrays: definition

Two ways to initialize an array

```
fruit = newArray(4);
fruit[0] = "apple";
fruit[1] = "orange";
fruit[2] = "apricot";
fruit[3] = "pear";
```

```
fruit = newArray("apple", "orange",
        "apricot", "pear");
```

newArray(size) returns a new array containing *size* elements, all with value 0

How to access and print the content of an array

```
for (i =0; i<fruit.length;
i++){
    print(fruit[i]);
}
```

```
Array.print(fruit);
```

fruit.length returns the number of elements composing the array

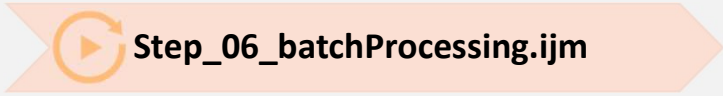
Array.print is one of the functions available for arrays: see the documentation for a list of them

Break-out Exercise: Batch-processing

Task:

- Use the template below to loop your code over all files in the input folder.

```
input_path = getDirectory("input files");
fileList = getFileList(input_path);

for (f=0; f<fileList.length; f++){

    //Clean-up to prepare for next image
    roiManager("reset");
    run("Close All");
    run("Clear Results");

    open(input_path + fileList[f]);
    print(input_path + fileList[f]); //displays file that is processed
    // rest of the code
    // (…)

    saveAs("results", "C:/Users/Anna/Desktop/"+title+"_results.xls");


}
```



Programming Basics

CONTROL STRUCTURES IF-STATEMENTS

Extra II: conditional statements – IF/ELSE IF/ELSE

- Aim:
  - To ask for the “minimum size” for *Analyze Particles..* only for the first image of a set.

```
//Step3: Retrieve the nuclei's boundaries
    num = getNumber("minimum size", 2000);
    run("Analyze Particles...", "size=" +num+ "-Infinity add");
```

Technical point

Conditional execution

Check one condition

- Only performs the operation if a condition is fulfilled
- The condition might result from a test:
  - Regular: <, >, <=, >=
  - Equality: ==
  - Difference: !=
- The result of the test could be stored in a variable (boolean)

```
i = 0;
if(i > 10){
        //Instruction 1
        //Instruction 2
        //Instruction 3
}
```

…one alternative

- Branch on 2 sets of instructions

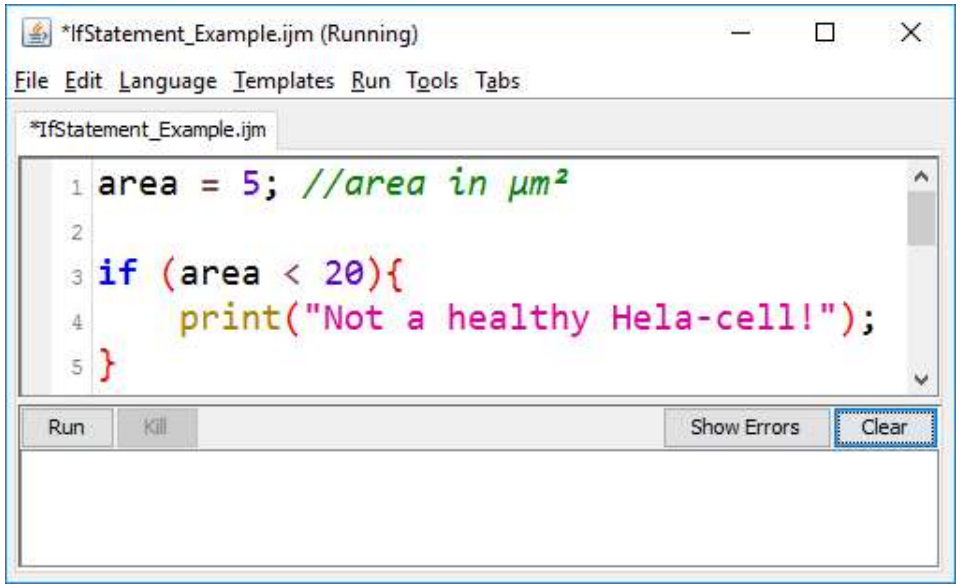
```
i = 0;
if(i > 10){
        //Instruction 1
        //Instruction 2
}else{
        //Instruction 3
        //Instruction 4
}
```

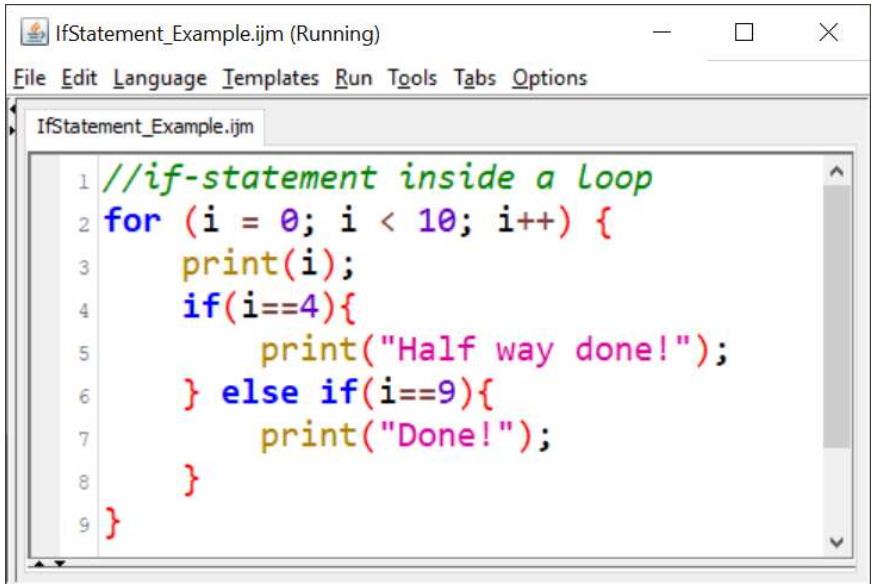
…multiple alternatives

- Branch or more than 2 alternatives

```
i = 0;
if(i > 10){
        //Instruction 1
}else if(i < 8) //Instruction 2
}else if(i <6 ) //Instruction 3
}else //Instruction 4
```

Programming Basics: If-statements





IfStatement_Example.ijm

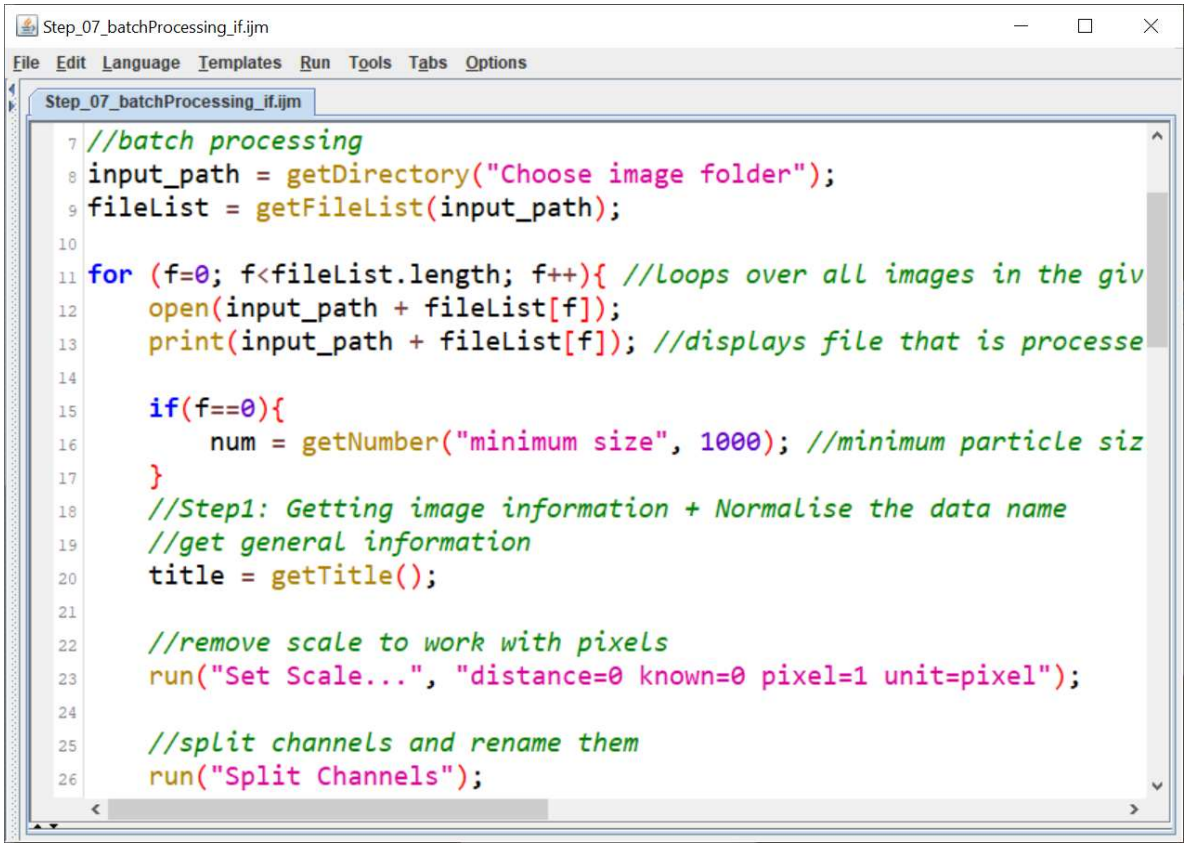
Break-out Exercise: If-statement

Task:

To ask for the “minimum size” for *Analyze Particles..* only for the first image of a set.

```
//Step3: Retrieve the nuclei's boundaries
    num = getNumber("minimum size", 1000);
    run("Analyze Particles...", "size=" +num+ "-Infinity add");
```

Extra II: conditional statements – IF/ELSE




```
7  //batch processing
8  input_path = getDirectory("Choose image folder");
9  fileList = getFileList(input_path);
10
11 for (f=0; f<fileList.length; f++){ //loops over all images in the giv
12     open(input_path + fileList[f]);
13     print(input_path + fileList[f]); //displays file that is processe
14
15     if(f==0){
16         num = getNumber("minimum size", 1000); //minimum particle siz
17     }
18     //Step1: Getting image information + Normalise the data name
19     //get general information
20     title = getTitle();
21
22     //remove scale to work with pixels
23     run("Set Scale...", "distance=0 known=0 pixel=1 unit=pixel");
24
25     //split channels and rename them
26     run("Split Channels");
```

Step_07_batchProcessing_if.ijm

The recorded workflow

Cleaned-up recorder:

Catch-up/Recorded.ijm

Recorder

Record: Macro Name: Macro.ijm Create ?

```
run("Split Channels");
selectWindow("C1-711_D6_1.tif");
rename("nuclei");
selectWindow("C2-711_D6_1.tif");
rename("signal");

selectWindow("nuclei");
run("Median...", "radius=8");
setAutoThreshold("Huang dark");
setOption("BlackBackground", false);
run("Convert to Mask");
run("Fill Holes");

run("Analyze Particles...", "size=2000-Infinity add");
roiManager("Show None");

roiManager("Select", 7);
run("Enlarge...", "enlarge=-10");
run("Make Band...", "band=10");
roiManager("Update");

run("Set Measurements...", "mean display redirect=None decimal=3");
selectWindow("signal");
roiManager("Select", 7);
run("Measure");
```

Normalise the data name

Filter nuclear image and make binary image

Retrieve nuclei boundaries

Retrieve the nuclear envelope boundary

Measure

General Good Practice

Readability:

- Use meaningful variable names
- Assign variable at the top of a script if the variable is used widely, or as close to where it is used as possible
- Comment your code: for you and others

Reproducibility:

- Add Initialization code: close windows, reset roiManager, reset Results table …
- Save quality control files, e.g. save the ROI manager
- Use file names that refer to the original files
- Save the parameters used with the other results.
- Save the macro itself or document its version
- Consider sharing your macro and parameters as Supplementary Information

Where to continue (I).

There are many more very useful built-in macro functions.

Check out SneakPreview_calculateRatio.ijm to

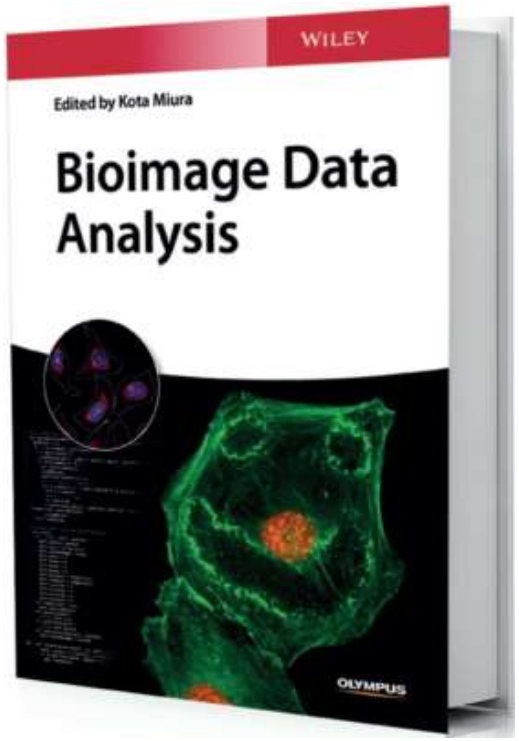
- Get all measurements: $Mean\ Intensity\ NucMembrane$, $Mean\ Intensity\ Nucleus$ and $Ratio$
- See how to customize the result window
- Get basic statistics of ROIs using getStatistics
- waitForUser: easy method to check intermediate results, debug, or wait for user interaction.

Results

File Edit Font Results

	image	mean Int nuclei	mean Int nuclear membrane	ratio	minimum size nuclei
1	931_H7_2.tif	34896.065	45857.361	1.314	700
2	931_H7_2.tif	28762.512	37526.908	1.305	700
3	931_H7_2.tif	27779.874	49714.626	1.790	700
4	931_H7_2.tif	31307.901	19444.399	0.621	700

Where to continue (II).



Other resources:

imagej.net/Introduction_into_Macro_Programming

forum.image.sc: Forum thread linked to this video

Chapter 3, ImageJ Macro Language
(free download)

Data resource

Raw images (tif) were provided by The Human Protein Atlas.

https://www.proteinatlas.org/humanproteome/cell

Thul, P.J. et al. (2017). A subcellular map of the human proteome. Science *356*.