

ImageJ Macro Programming

Robert Haase

With material from

Anne Esslinger, Alberti Lab, Biotec, MPI CBG

Martin Weigert, EPFL, Lausanne

Benoit Lombardot, Scientific Computing Facility, MPI CBG

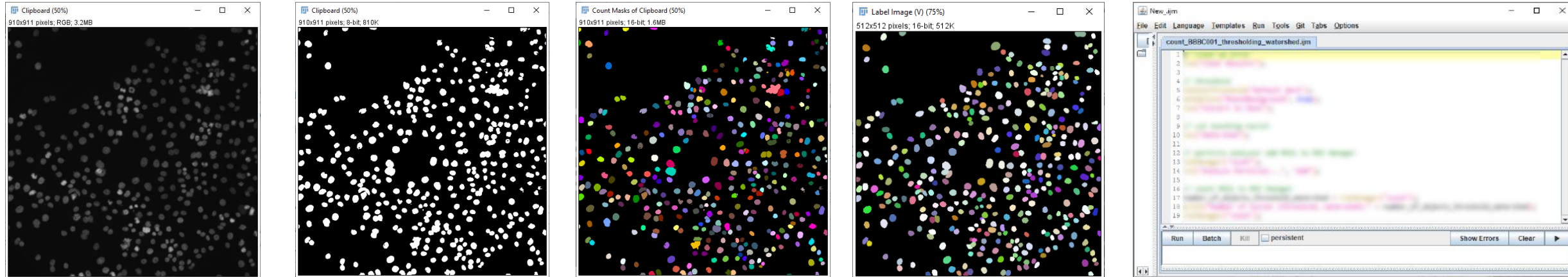
Gayathri Nadar, Scientific Computing Facility, MPI CBG

Jens Ehrig, CMCB, TU Dresden

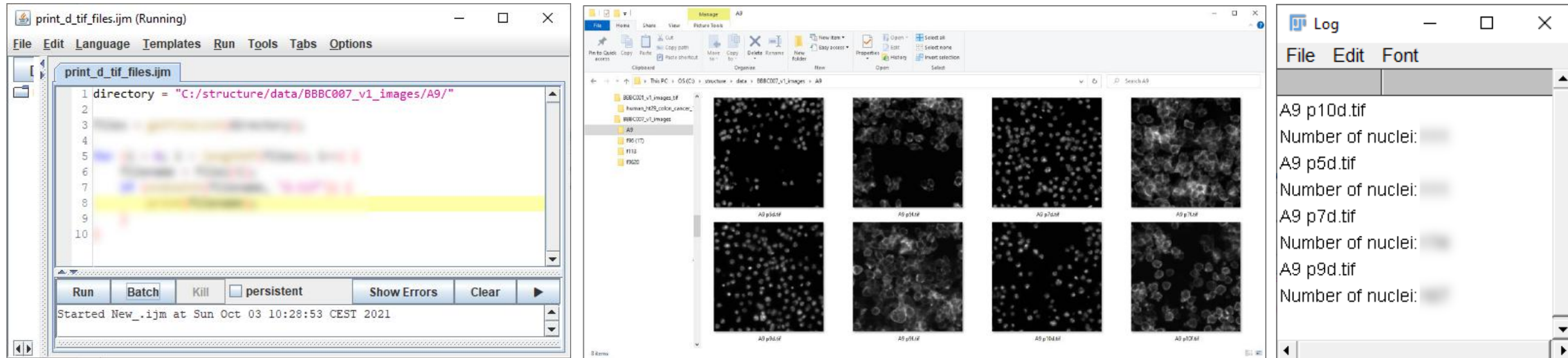
Noreen Walker, Scientific Computing Facility, MPI CBG

Virtually at CCI Gothenburg, October 2021

- Day 1: Scripting in ImageJ / Fiji allows reproducible workflows.



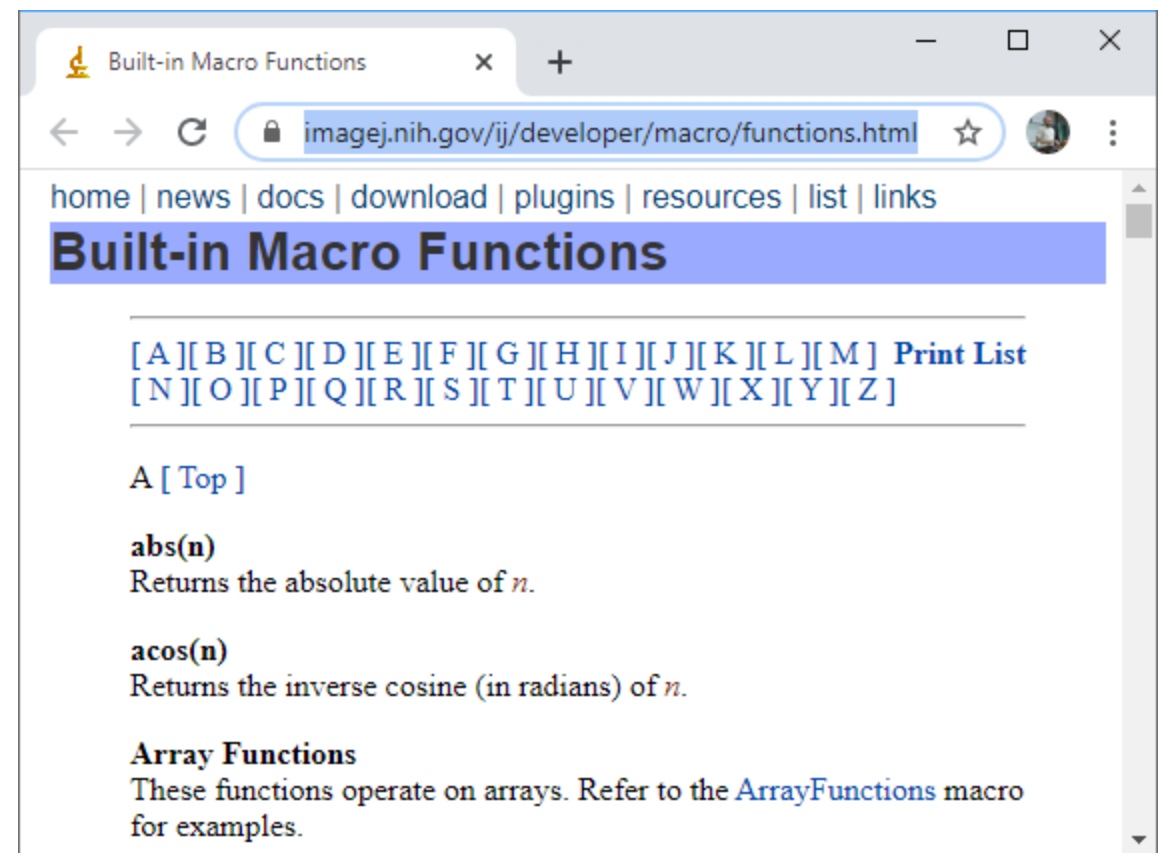
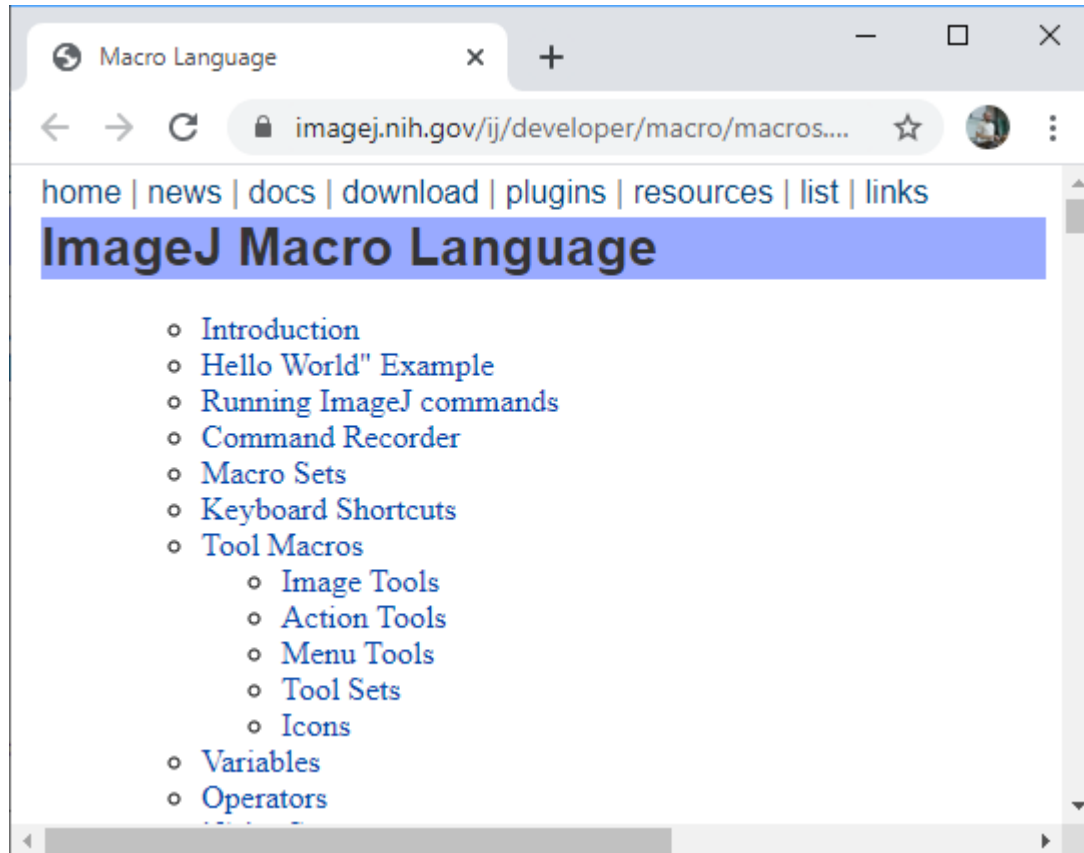
- Day 2: Run workflows on hundreds over folders of images!



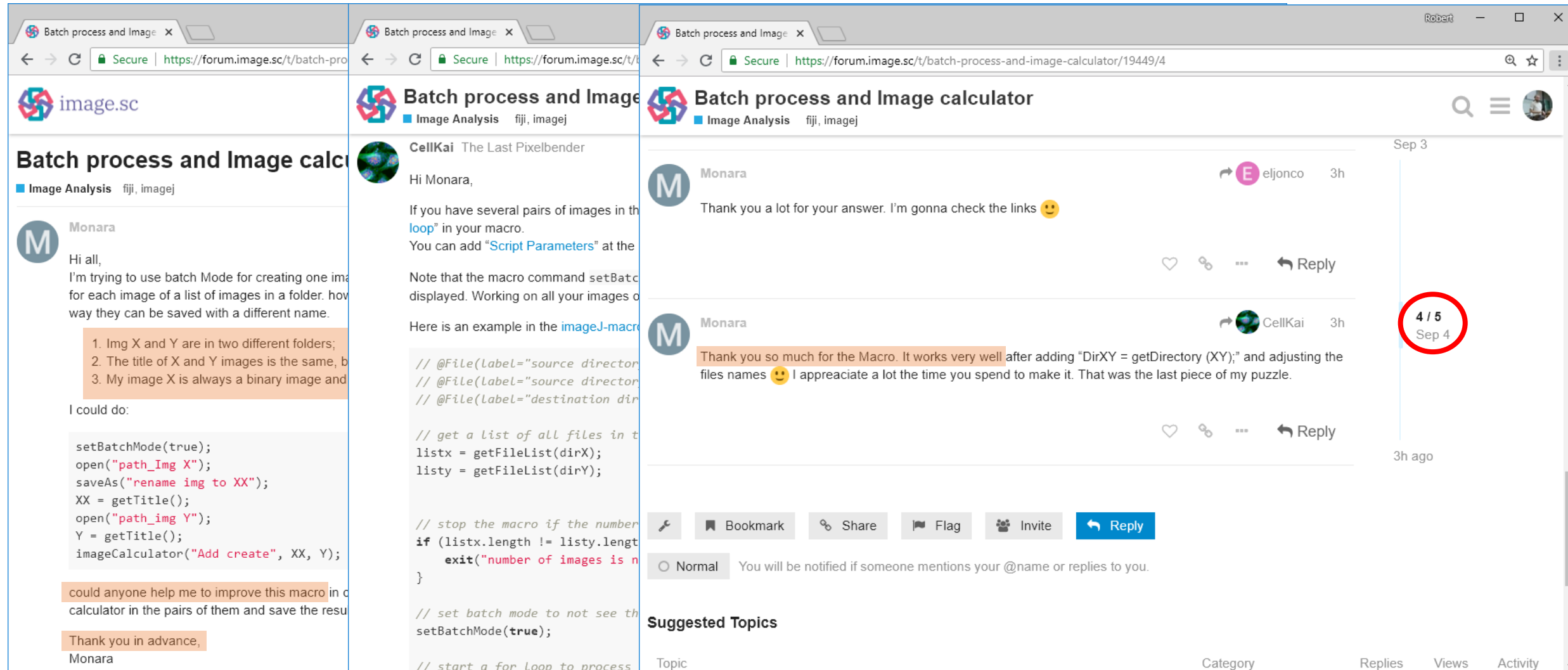
- Important ImageJ macro websites

<https://imagej.nih.gov/ij/developer/macro/macros.html>

<https://imagej.nih.gov/ij/developer/macro/functions.html>



- Visit <http://forum.image.sc> !



The screenshot displays a forum thread on the ImageJ forum (forum.image.sc) titled "Batch process and Image calculator". The thread is part of the "Image Analysis" category and is tagged with "fiji, imagej".

The thread starts with a post by user **Monara** (posted 3h ago). Monara asks for help in using batch mode to create one image for each image of a list of images in a folder, where they can be saved with a different name. Monara lists three requirements:

1. Img X and Y are in two different folders;
2. The title of X and Y images is the same, but
3. My image X is always a binary image and

Monara then provides a macro code snippet:

```
setBatchMode(true);
open("path_img X");
saveAs("rename img to XX");
XX = getTitle();
open("path_img Y");
Y = getTitle();
imageCalculator("Add create", XX, Y);
```

Monara asks if anyone can help improve this macro in the calculator in the pairs of them and save the results. Monara thanks the community in advance.

The thread continues with a reply from user **CellKai** (The Last Pixelbender, posted 3h ago). CellKai responds to Monara, stating that if you have several pairs of images in the "loop" in your macro, you can add "Script Parameters" at the beginning. CellKai notes that the macro command setBatchMode is displayed. Working on all your images. CellKai provides an example in the imageJ-macro:

```
// @File(label="source director
// @File(label="source director
// @File(label="destination dir

// get a list of all files in t
listx = getFileList(dirX);
listy = getFileList(dirY);

// stop the macro if the number
if (listx.length != listy.length)
    exit("number of images is n
}
```

CellKai then provides a macro code snippet:

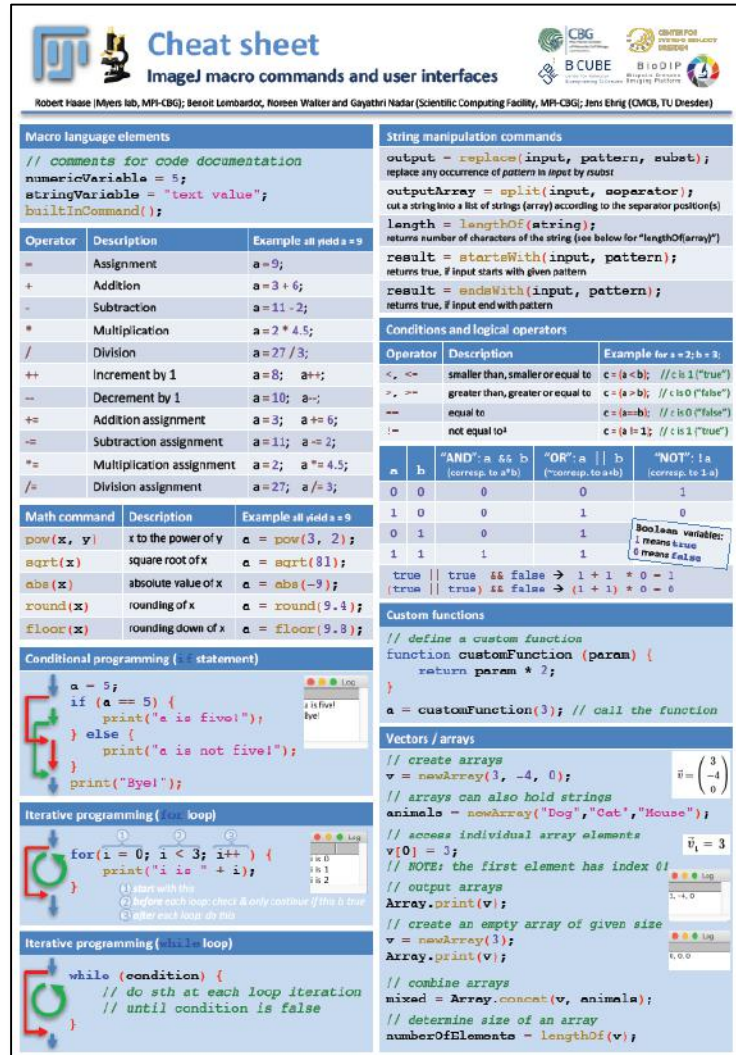
```
// set batch mode to not see th
setBatchMode(true);

// start a for loop to process
```

The thread concludes with a reply from user **Monara** (posted 3h ago). Monara thanks CellKai for the macro, stating that it works very well after adding "DirXY = getDirectory(XY);" and adjusting the file names. Monara expresses appreciation for the time spent making it, stating that it was the last piece of the puzzle.

The forum interface shows a "4 / 5" rating for the thread, indicating it is the 4th of 5 posts in the thread. The thread is also marked as "Suggested Topics".

- https://github.com/BiAPoL/imagej-macro-cheat-sheet/blob/master/ImageJ_macro_cheatsheet.pdf



Cheat sheet
ImageJ macro commands and user interfaces

Robert Haase (Myers lab, MPI-CBG); Benoit Lombardot, Noreen Walker and Gayathri Nadar (Scientific Computing Facility, MPI-CBG); Jens Ehrig (CMCB, TU Dresden)

Macro language elements

```
// comments for code documentation
numericVariable = 5;
stringValue = "text value";
buildInCommand();
```

Operator	Description	Example all yield a = 9
=	Assignment	a = 9;
+	Addition	a = 3 + 6;
-	Subtraction	a = 11 - 2;
*	Multiplication	a = 2 * 4.5;
/	Division	a = 27 / 3;
++	Increment by 1	a = 8; a++;
--	Decrement by 1	a = 10; a--;
+=	Addition assignment	a = 3; a += 6;
-=	Subtraction assignment	a = 11; a -= 2;
*=	Multiplication assignment	a = 2; a *= 4.5;
/=	Division assignment	a = 27; a /= 3;

Math command

Command	Description	Example all yield a = 9
pow(x, y)	x to the power of y	a = pow(3, 2);
sqrt(x)	square root of x	a = sqrt(81);
abs(x)	absolute value of x	a = abs(-9);
round(x)	rounding of x	a = round(9.4);
floor(x)	rounding down of x	a = floor(9.8);

Conditional programming (if... statement)

```
a = 5;
if (a == 5) {
    print("a is five!");
} else {
    print("a is not five!");
}
print("Bye!");
```

Iterative programming (for loop)

```
for(i = 0; i < 3; i++) {
    print("i is " + i);
}
```

Iterative programming (while loop)

```
while (condition) {
    // do sth at each loop iteration
    // until condition is false
}
```

String manipulation commands

```
output = replace(input, pattern, subset);
// replace any occurrence of pattern in input by subset

outputArray = split(input, separator);
// cut a string into a list of strings (array) according to the separator position(s)

length = lengthOf(string);
// returns number of characters of the string (see below for "lengthOf(array)")

result = startsWith(input, pattern);
// returns true, if input starts with given pattern

result = endsWith(input, pattern);
// returns true, if input end with pattern
```

Conditions and logical operators

Operator	Description	Example for a = 2; b = 8
<, <=	smaller than, smaller or equal to	c = (a < b); // c is 1 ("true")
>, >=	greater than, greater or equal to	c = (a > b); // c is 0 ("false")
==	equal to	c = (a == b); // c is 0 ("false")
!=	not equal to	c = (a != b); // c is 1 ("true")

AND, OR, NOT

a	b	"AND": a & b (corresp. to a*b)	"OR": a b (corresp. to a+b)	"NOT": !a (corresp. to 1-a)
0	0	0	0	1
1	0	0	1	0
0	1	0	1	1
1	1	1	1	0

Boolean variables

```
Boolean variable:
1 means true
0 means false
```

Custom functions

```
// define a custom function
function customFunction (param) {
    return param * 2;
}

a = customFunction(3); // call the function
```

Vectors / arrays

```
// create arrays
v = newArray(3, -4, 0);
// arrays can also hold strings
animals = newArray("Dog", "Cat", "Mouse");

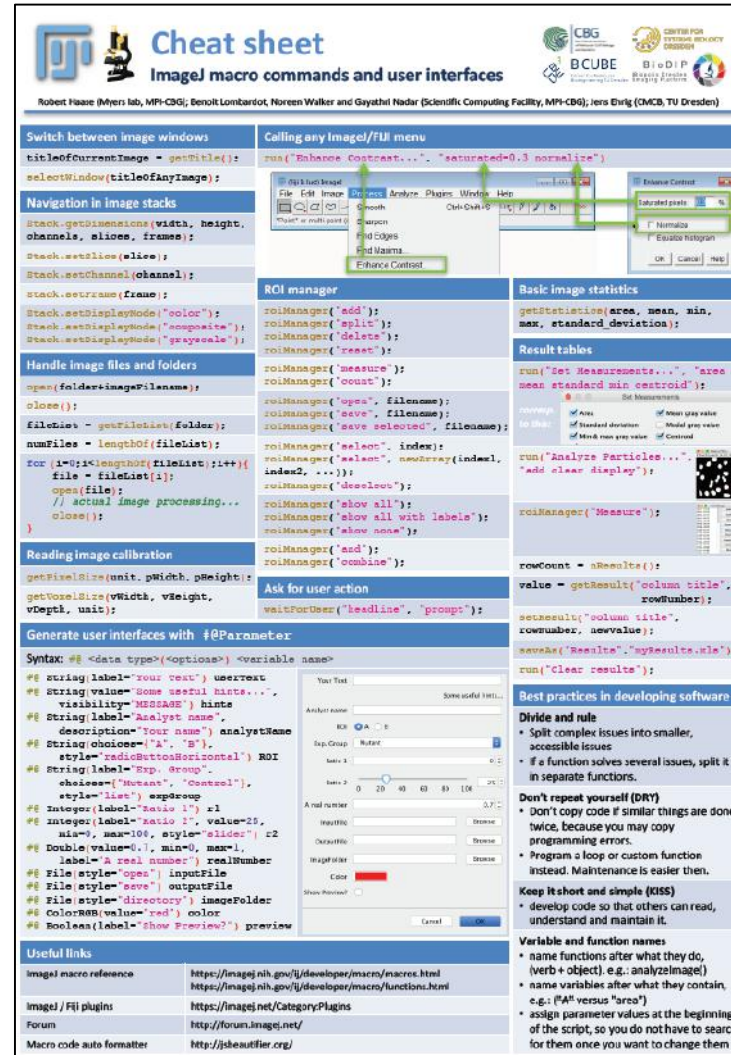
// access individual array elements
v[0] = 3;
// NOTE: the first element has index 0!

// output arrays
Array.print(v);

// create an empty array of given size
v = newArray(3);
Array.print(v);

// combine arrays
mixed = Array.concat(v, animals);

// determine size of an array
numberOfElements = lengthOf(v);
```



Cheat sheet
ImageJ macro commands and user interfaces

Robert Haase (Myers lab, MPI-CBG); Benoit Lombardot, Noreen Walker and Gayathri Nadar (Scientific Computing Facility, MPI-CBG); Jens Ehrig (CMCB, TU Dresden)

Switch between image windows

```
titleOfCurrentImage = getTitle();
selectWindow(titleOfAnyImage);
```

Navigation in image stacks

```
stack.getDimensions(width, height,
channels, slices, frames);
stack.setSlice(slice);
stack.setChannel(channel);
stack.setCurrentFrame(frame);
stack.setDisplayMode("color");
stack.setDisplayMode("composite");
stack.setDisplayMode("grayscale");
```

Handle image files and folders

```
open(folder+imageFilename);
close();
filename = getFileNames(folder);
numberOfFiles = lengthOf(filename);
for(i=0; i<numberOfFiles; i++){
    file = filename[i];
    open(file);
    // actual image processing...
    close();
}
```

Reading image calibration

```
getPixelSize(unit, pWidth, pHeight);
getVoxelSize(vWidth, vHeight,
vDepth, unit);
```

Generate user interfaces with @Parameter

```
Syntax: @<data type>(<options>) <variable name>
@String label="Your text" userText
@String value="Some useful hint..."
visibility="MESSAGE" hints
@String label="Analyst name",
description="Your name" analystName
@String choices={"A", "B"},
@String label="Exp. group",
choices={"Mutant", "Control"},
style="list" expGroup
@Integer label="Ratio 1" r1
@Integer label="Ratio 2" value=20,
min=0, max=100, style="slider" r2
@Double value=0.1, min=0, max=1,
label="A real number" realNumber
@File style="open", inputFile
@File style="directory" imageFolder
@ColorRGB(value="red") color
@Boolean label="Show Preview?" preview
```

ROI manager

```
roiManager("add");
roiManager("split");
roiManager("delete");
roiManager("reset");
roiManager("measure");
roiManager("copy", filename);
roiManager("paste", filename);
roiManager("save selected", filename);
roiManager("select", index);
roiManager("select", newArray(index1,
index2, ...));
roiManager("deselect");
roiManager("show all");
roiManager("show all with labels");
roiManager("show none");
roiManager("add");
roiManager("combine");
```

Ask for user action

```
waitForUser("headline", "prompt");
```

Basic image statistics

```
getStatistics(area, mean, min,
max, standard_deviation);
run("Set Measurements...", "area
mean standard min centroid");
run("Analyze Particles...",
"add clear display");
roiManager("Measure");
```

Result tables

```
rowCount = nResults();
value = getResult("column title",
rowNumber);
setResult("column title",
rowNumber, newValue);
saveAs("Results", "myResults.kile");
run("Clear results");
```

Best practices in developing software

Divide and rule

- Split complex issues into smaller, accessible issues
- If a function solves several issues, split it in separate functions.

Don't repeat yourself (DRY)

- Don't copy code if similar things are done twice, because you may copy programming errors.
- Program a loop or custom function instead. Maintenance is easier then.

Keep it short and simple (KISS)

- develop code so that others can read, understand and maintain it.

Variable and function names

- name functions after what they do, (verb + object) e.g. analyzeImage()
- name variables after what they contain, e.g. ("A" versus "area")
- assign parameter values at the beginning of the script, so you do not have to search for them once you want to change them

Useful links

ImageJ macro reference: <https://imagej.nih.gov/ij/developer/macros/macros.html>
<https://imagej.nih.gov/ij/developer/macros/functions.html>

ImageJ / FIJi plugins: <https://imagej.net/Category/Plugins>

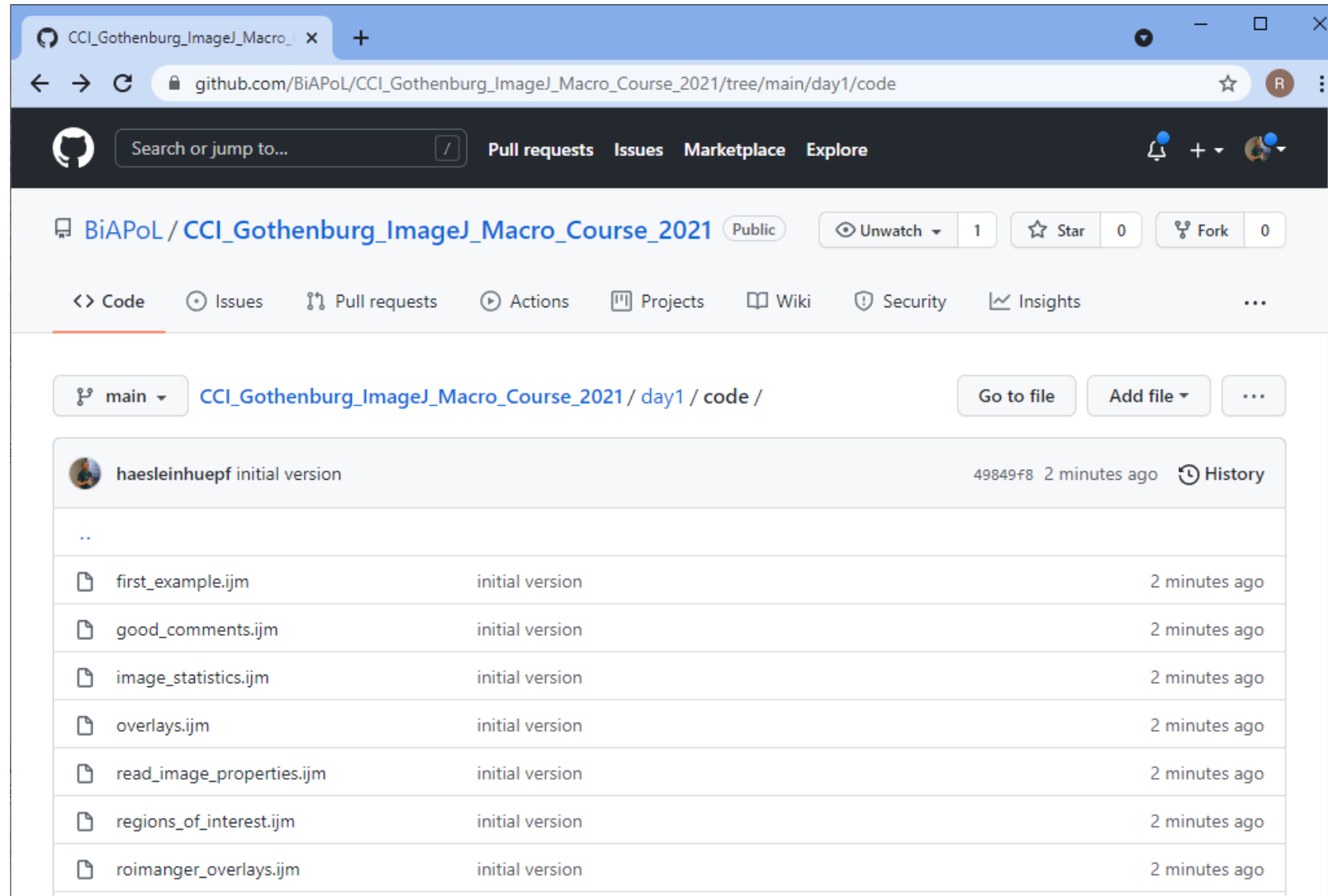
Forum: <http://forum.imagej.net/>

Macro code auto formatter: <http://jsbeautifier.org/>

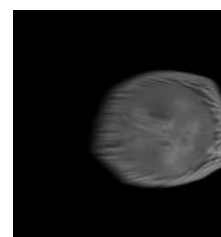
Thanks to
Benoit Lombardot,
Gayathri Nadar,
Jens Ehrig and
Noreen Walker for
contributing to and
maintaining this document!

Example code for this course

- https://github.com/BiAPoL/CCI_Gothenburg_ImageJ_Macro_Course_2021



banana0008.tif
banana0009.tif
banana0010.tif
banana0011.tif
banana0012.tif



- Remove shell
- Repeat until nothing left:

- Take a bite

- Chew

- Swallow

- Digest

- Access folder
- Repeat for all images:

- Open an image file

- Segment the banana slice

- Analyse it

- Save measurements

```
folder = "C:/structure/teaching/lecture_applied_bioimage_an
filelist = getFileList(folder);

for (i = 0; i < lengthOf(filelist); i++) {
    // get the nth entry from the filelist array
    file = filelist[i];

    if (endsWith(file, ".tif")) {

        // open the image
        open(folder + file);

        // segment the object in the image
        setAutoThreshold("Default dark");
        setOption("BlackBackground", true);
        run("Convert to Mask");
        run("Fill Holes");

        // measure the position of this one single object
        run("Set Measurements...", "centroid redirect=None");
        run("Create Selection");
        run("Measure");
        setResult("Slice", nResults() - 1, i);

        // close the image
        close();

    }
}

// save the positions as CSV to disc
result_filename = "Position.csv"
saveAs("Results", folder + result_filename);
```

ImageJ macro programming: Variables and operations

Robert Haase

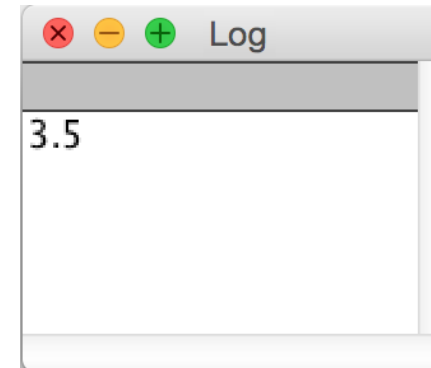
With material from
Benoit Lombardot, Scientific Computing Facility, MPI CBG

Virtually at CCI Gothenburg, October 2021

- Variables hold **values** and can be connected using **operators**

```
// initialise program  
a = 1;  
b = 2.5;  
  
// run complicated algorithm  
final_result = a + b;  
  
// print the result  
print( final_result );
```

variables.ijm



- Math commands supplement operators to be able to implement any form of calculations

Command	Description	Example
<code>pow(x, y)</code>	x to the power of y	<code>a = pow(3, 2);</code>
<code>sqrt(x)</code>	square root of x	<code>a = sqrt(81);</code>
<code>abs(x)</code>	absolute value of x	<code>a = abs(-9);</code>
<code>round(x)</code>	rounding of x	<code>a = round(9.4);</code>
<code>sin(x)</code>	sinus of x given in radians	<code>b = sin(PI);</code>
<code>cos(x)</code>	cosinus of x given in radians	<code>b = cos(PI);</code>
<code>tan(x)</code>	tangens of x given in radians	<code>b = tan(PI);</code>

Comments should contain additional information such as

- User documentation
 - What does the program do?
 - How can this program be used?
- Your name / institute in case a reader has a question
- Comment why things are done.
- Do not comment what is written in the code already!

```
//  
// This program sums up two numbers.  
//  
// Usage:  
// * Run it in FIJI (www.fiji.sc)  
//  
// Author: Robert Haase, MPI CBG,  
//         rhaase@mpi-cbg.de  
// July 2016  
  
// initialise program  
a = 1;  
b = 2.5;  
  
// run complicated algorithm  
final_result = a + b;  
  
// print the final result  
print( final_result );
```

good_comments.ijm

/* Comments and markdown */

- Write your own ImageJ macro notebooks!

- Use:

/*

Headlines

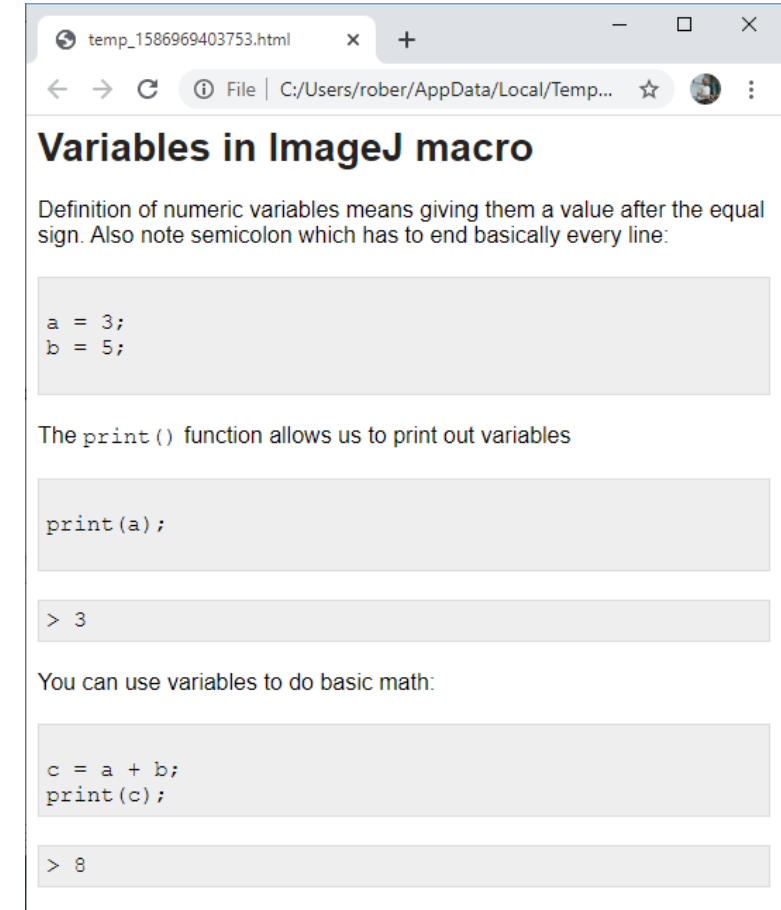
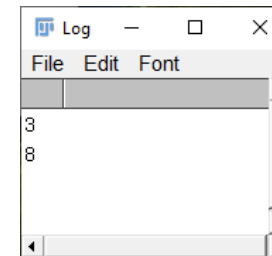
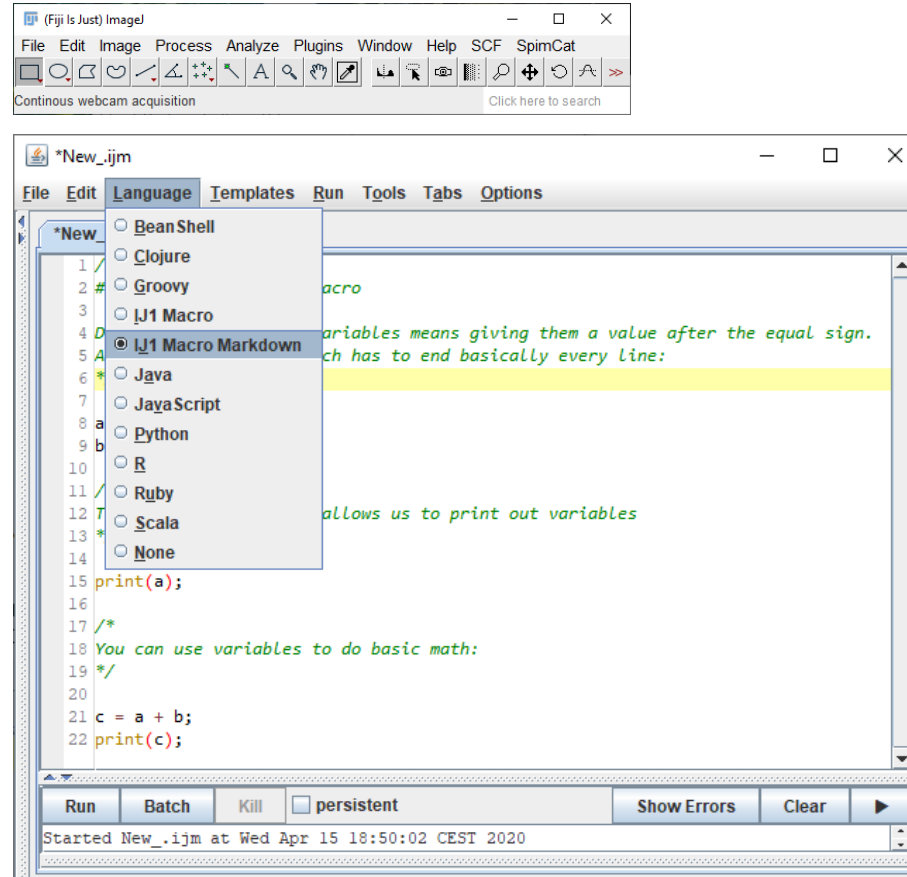
* Bullet points

[Links] (http://fiji.sc)

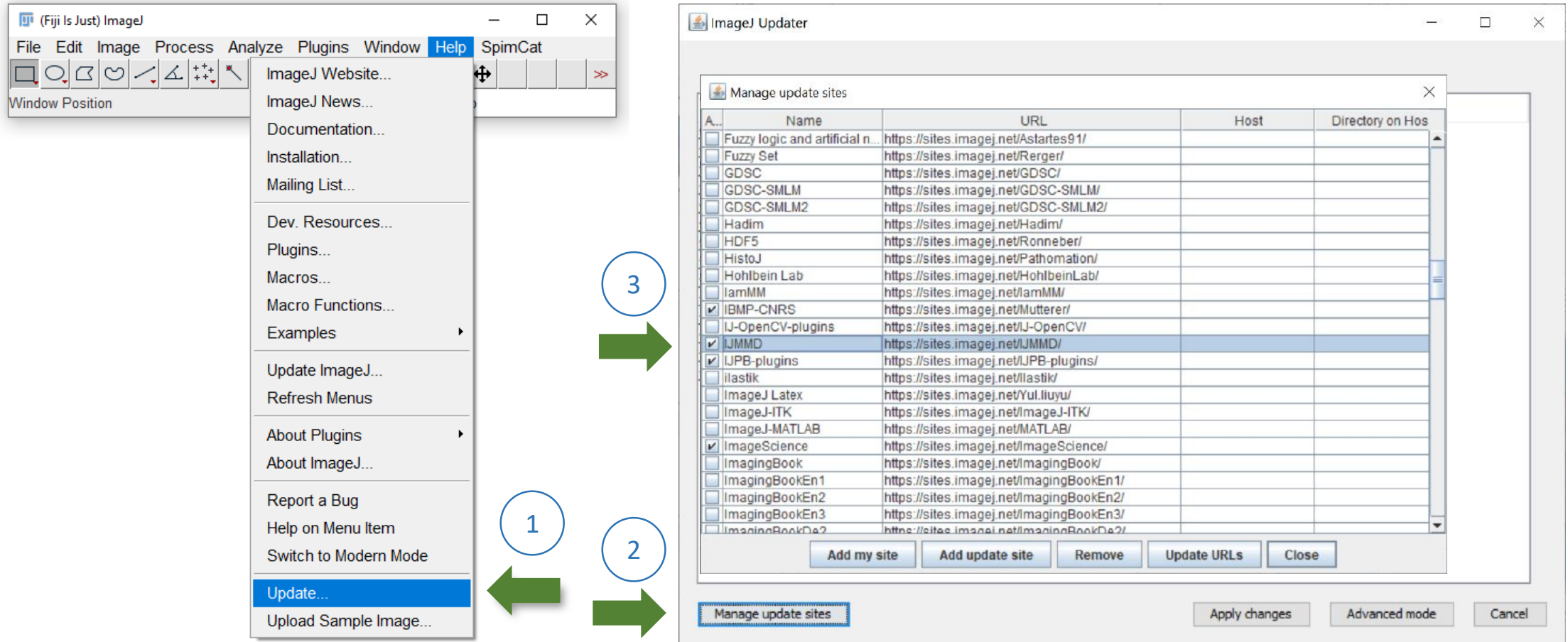
*/

Installation

- <https://github.com/haesleinhuepf/imagejmacromarkdown>



- Available via the IJMMD update site



The screenshot illustrates the installation process for ImageJ Macro Markdown (IJMMD) using the ImageJ Updater. It is divided into three numbered steps:

- Step 1:** Open the **Help** menu in the ImageJ application.
- Step 2:** Select **Update...** from the Help menu.
- Step 3:** In the **ImageJ Updater** window, check the **IJMMD** checkbox and click the **Add update site** button.

The **ImageJ Updater** window displays a table of available update sites:

A...	Name	URL	Host	Directory on Hos
<input type="checkbox"/>	Fuzzy logic and artificial n...	https://sites.imagej.net/Astartes91/		
<input type="checkbox"/>	Fuzzy Set	https://sites.imagej.net/Rerger/		
<input type="checkbox"/>	GDSC	https://sites.imagej.net/GDSC/		
<input type="checkbox"/>	GDSC-SMLM	https://sites.imagej.net/GDSC-SMLM/		
<input type="checkbox"/>	GDSC-SMLM2	https://sites.imagej.net/GDSC-SMLM2/		
<input type="checkbox"/>	Hadim	https://sites.imagej.net/Hadim/		
<input type="checkbox"/>	HDF5	https://sites.imagej.net/Ronneber/		
<input type="checkbox"/>	HistoJ	https://sites.imagej.net/Pathomation/		
<input type="checkbox"/>	Hohlbein Lab	https://sites.imagej.net/HohlbeinLab/		
<input type="checkbox"/>	IamMM	https://sites.imagej.net/IamMM/		
<input checked="" type="checkbox"/>	IBMP-CNRS	https://sites.imagej.net/Mutterer/		
<input type="checkbox"/>	IJ-OpenCV-plugins	https://sites.imagej.net/IJ-OpenCV/		
<input checked="" type="checkbox"/>	IJMMD	https://sites.imagej.net/IJMMD/		
<input checked="" type="checkbox"/>	IJPB-plugins	https://sites.imagej.net/IJPB-plugins/		
<input type="checkbox"/>	ilastik	https://sites.imagej.net/ilastik/		
<input type="checkbox"/>	ImageJ Latex	https://sites.imagej.net/Yul.liuyu/		
<input type="checkbox"/>	ImageJ-ITK	https://sites.imagej.net/ImageJ-ITK/		
<input type="checkbox"/>	ImageJ-MATLAB	https://sites.imagej.net/MATLAB/		
<input checked="" type="checkbox"/>	ImageScience	https://sites.imagej.net/ImageScience/		
<input type="checkbox"/>	ImagingBook	https://sites.imagej.net/ImagingBook/		
<input type="checkbox"/>	ImagingBookEn1	https://sites.imagej.net/ImagingBookEn1/		
<input type="checkbox"/>	ImagingBookEn2	https://sites.imagej.net/ImagingBookEn2/		
<input type="checkbox"/>	ImagingBookEn3	https://sites.imagej.net/ImagingBookEn3/		
<input type="checkbox"/>	ImagingBookDe2	https://sites.imagej.net/ImagingBookDe2/		

Buttons at the bottom of the ImageJ Updater window include: **Add my site**, **Add update site**, **Remove**, **Update URLs**, **Close**, **Manage update sites**, **Apply changes**, **Advanced mode**, and **Cancel**.

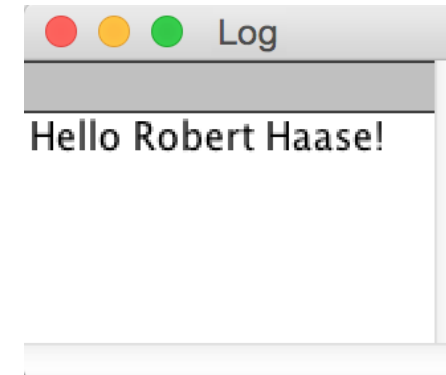
- Texts are called “strings” in the world of programming
- They represent a chain of characters

```
// initialise program
firstname = "Robert";
lastname = "Haase";

// run complicated algorithm
name = firstname + " " + lastname;

print("Hello " + name + "!!");
```

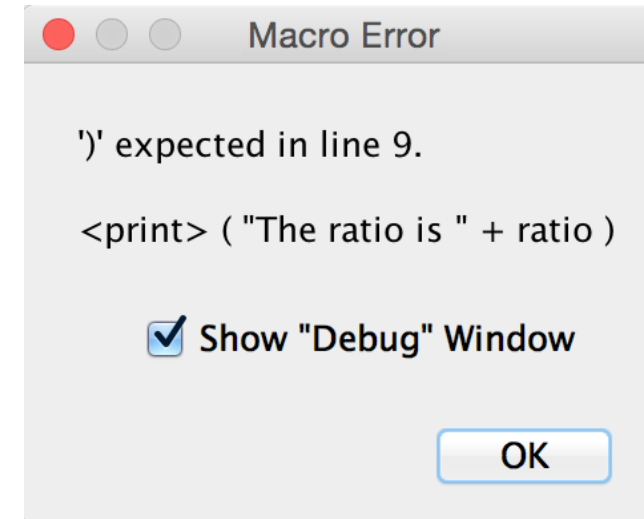
string_variables.ijm



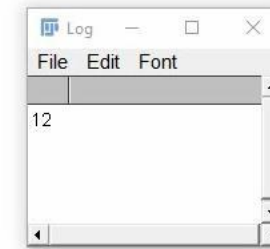
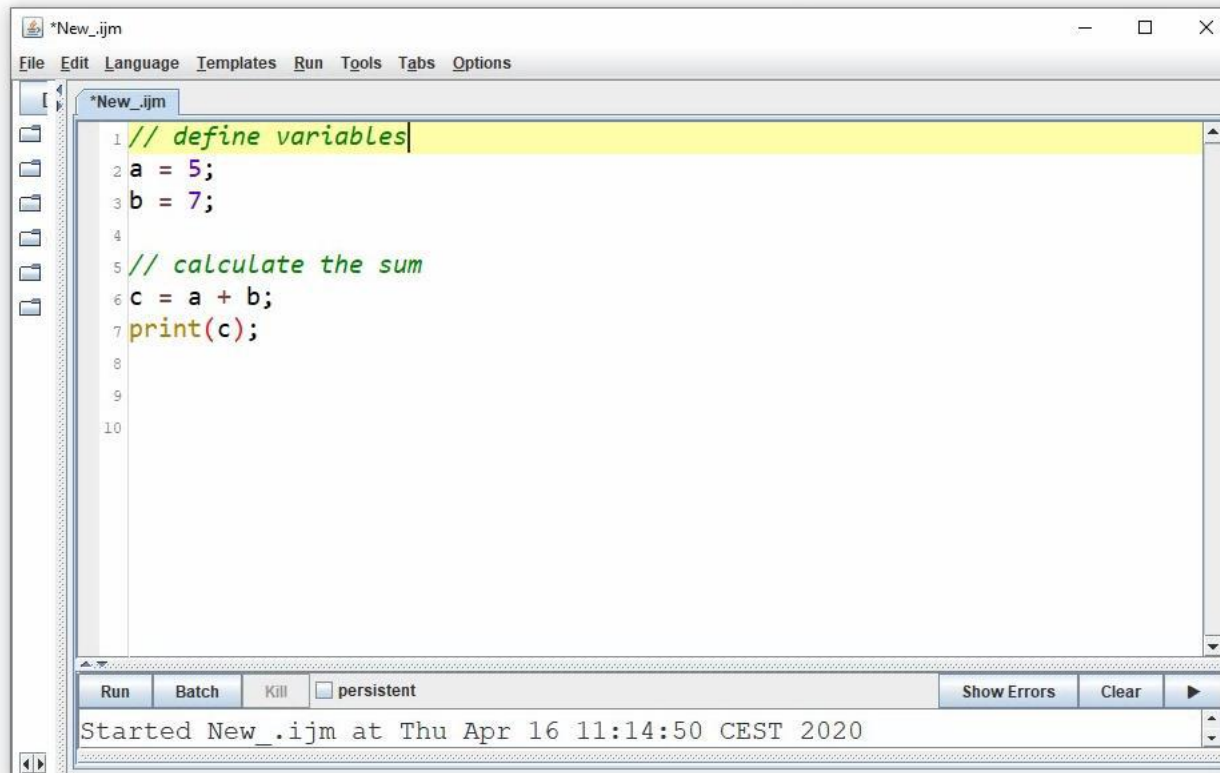
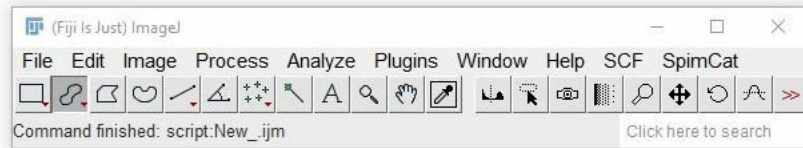
- Colours help us to read and understand what we see:
 - comments / documentation
 - commands / action
 - values (numbers, vectors, texts)
- If something is wrong with the colour, likely something is wrong with the program.

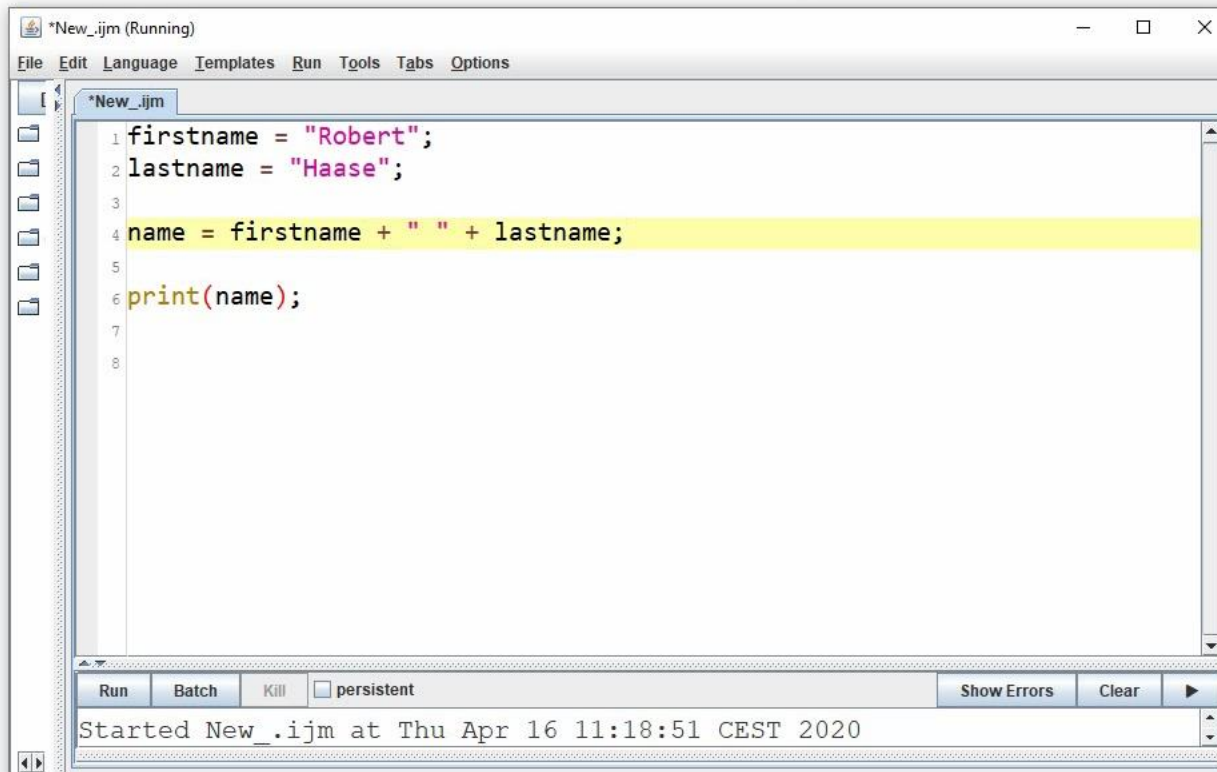
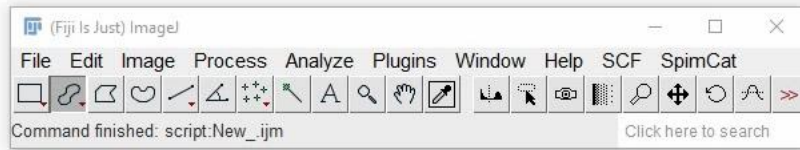
```
1 a = 5;  
2 b = 7;  
3  
4 sum = a + b;  
5 ratio = a / b;  
6  
7 print("The sum is + sum);  
8 print("The ratio is " + ratio);  
9  
10
```

code_highlights.ijm



Learning by doing!





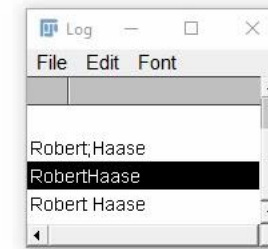
*New_ijm (Running)

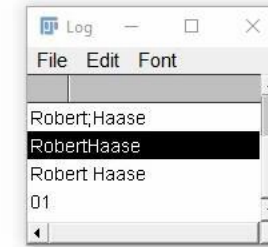
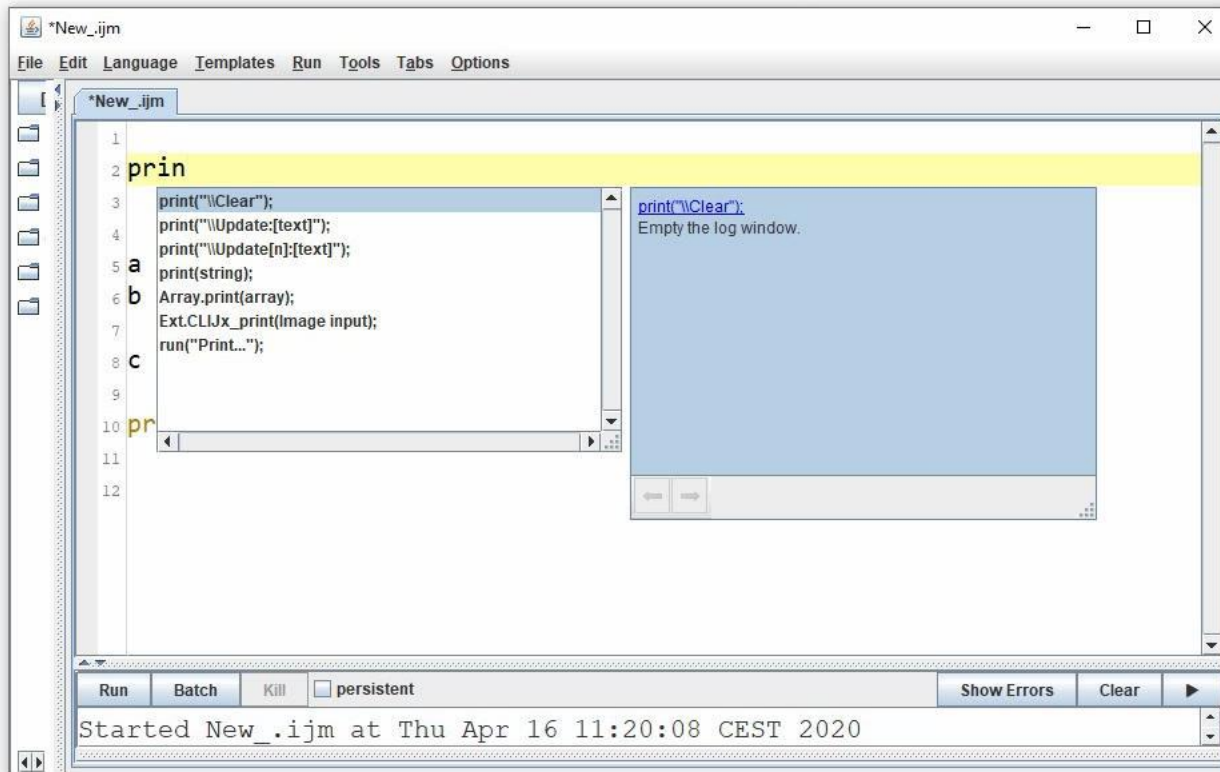
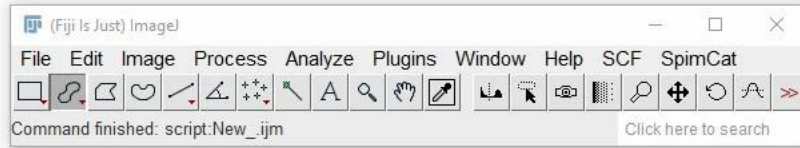
File Edit Language Templates Run Tools Tabs Options

```
1 firstname = "Robert";  
2 lastname = "Haase";  
3  
4 name = firstname + " " + lastname;  
5  
6 print(name);  
7  
8
```

Run Batch Kill ☐ persistent Show Errors Clear ▶

Started New_.ijm at Thu Apr 16 11:18:51 CEST 2020





ImageJ macro programming: Recording macros & working with images

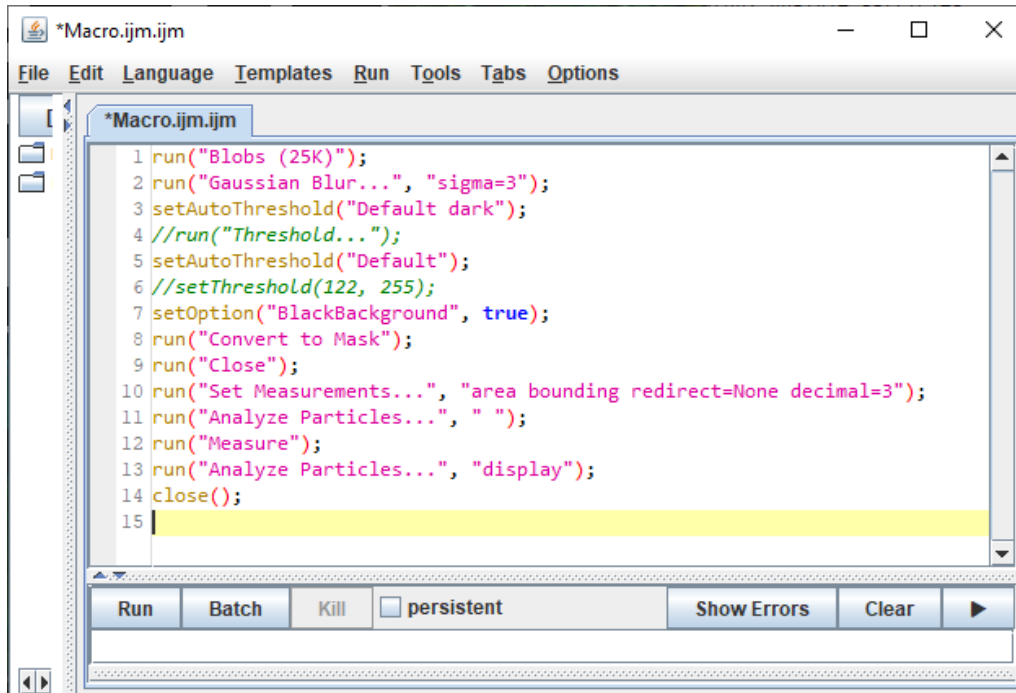
Robert Haase

With material from

Benoit Lombardot, Scientific Computing Facility, MPI CBG

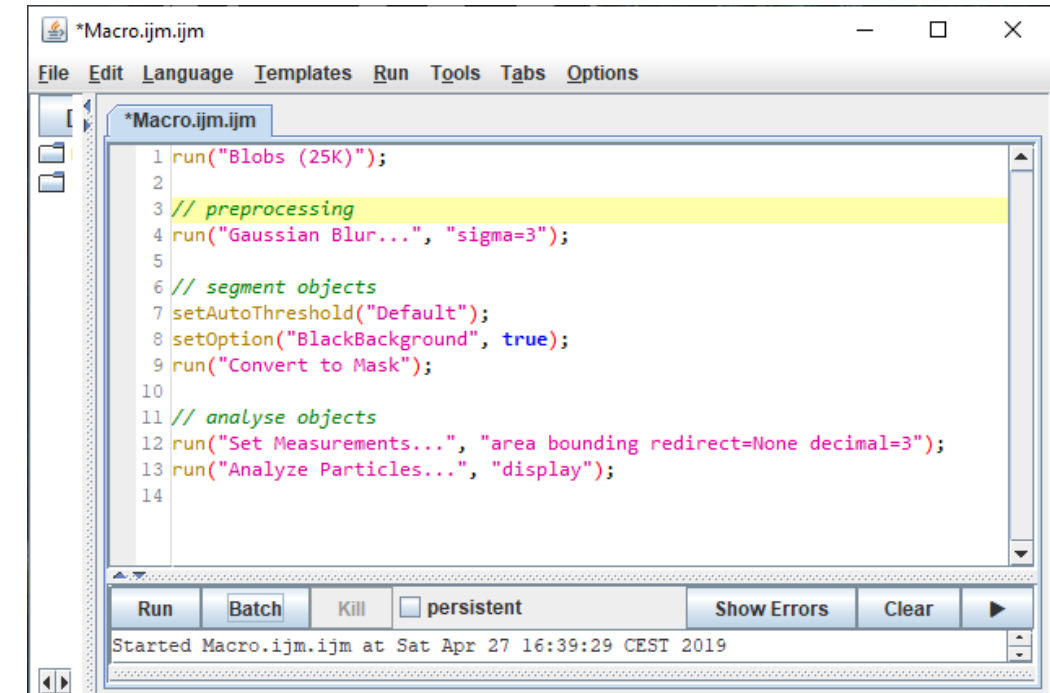
Virtually at CCI Gothenburg, October 2021

- Editing recorded macros needs to be trained. It's 80% reading and 20% writing
- Hints:
 - Put comments first. Try to understand what was recorded and why.
 - Do it in tiny steps. As soon as you have a working workflow consisting of 4-5 steps, create a macro.
 - Collect macros. When you do something new, do cherry picking from the old macros.



```
*Macro.ijm.ijm
File Edit Language Templates Run Tools Tabs Options

*Macro.ijm.ijm
1 run("Blobs (25K)");
2 run("Gaussian Blur...", "sigma=3");
3 setAutoThreshold("Default dark");
4 //run("Threshold...");
5 setAutoThreshold("Default");
6 //setThreshold(122, 255);
7 setOption("BlackBackground", true);
8 run("Convert to Mask");
9 run("Close");
10 run("Set Measurements...", "area bounding redirect=None decimal=3");
11 run("Analyze Particles...", " ");
12 run("Measure");
13 run("Analyze Particles...", "display");
14 close();
15
```



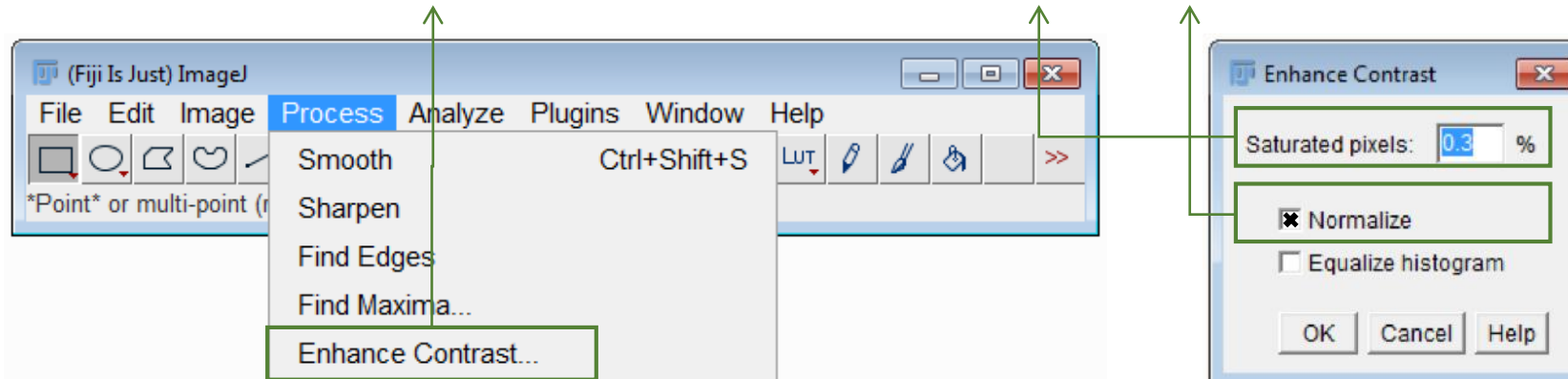
```
*Macro.ijm.ijm
File Edit Language Templates Run Tools Tabs Options

*Macro.ijm.ijm
1 run("Blobs (25K)");
2
3 // preprocessing
4 run("Gaussian Blur...", "sigma=3");
5
6 // segment objects
7 setAutoThreshold("Default");
8 setOption("BlackBackground", true);
9 run("Convert to Mask");
10
11 // analyse objects
12 run("Set Measurements...", "area bounding redirect=None decimal=3");
13 run("Analyze Particles...", "display");
14
15
```

- The first parameter of the run command is the menu you want to call
- The following parameter contains the entries of the dialog in case the menu has one.
 - Numeric values: "label=value"
 - Checkboxes: "label" in case it is activated.

Calling any ImageJ/FIJI menus

```
run("Enhance Contrast...", "saturated=0.3 normalize")
```



- Open and close commands allow handling image files.

```
// initialise program
imageFilename = "/Users/rhaase/images/blobs.gif";

open( imageFilename );

// process the image
// ...

close();
```

Built-in command	Description	Parameters
<code>open(filename);</code>	open an image	filename of the image
<code>close();</code>	close current image	

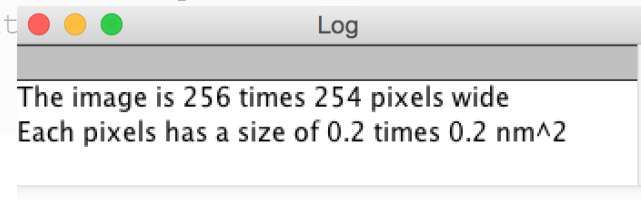
- Firstly, lets read image properties

```
// initialise program
imageFilename = "blobs.gif";
open( imageFilename );

// read image properties
width = getWidth();
height = getHeight();
getPixelSize(unit, pixelWidth, pixelHeight);

// show results
print("The image is " + width + " times " + height + " pixels wide");
print("Each pixels has a size of " + pixelWidth +
      " times " + pixelHeight);

close();
```



read_image_properties.ijm

Built-in command	Description	Parameters
width = <code>getWidth()</code> ; height = <code>getHeight()</code> ;	return the width/height of the current image	
<code>getPixelSize</code> (unit, pWidth, pHeight)	get size of a pixel	variables where physical unit, width and height will be stored
<code>getVoxelSize</code> (vWidth, vHeight, vDepth, unit)	get size of a voxel	variables where width, height, depth and physical unit will be stored

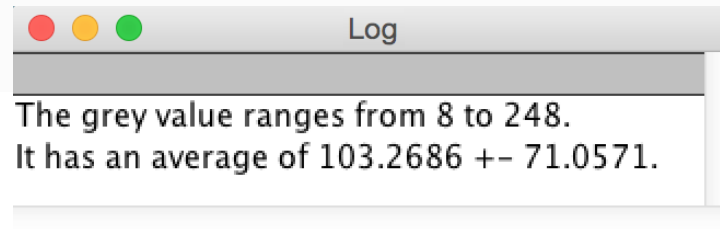
- Let's do some pixel value statistics

```
// initialise program
imageFilename = "blobs.gif";
open( imageFilename );

// get image statistics
getStatistics(area, mean, min, max, std);

// show results
print("The grey value ranges from " + min + " to " + max + ". ");
print("It has an average of " + mean + " +- " + std + ". ");

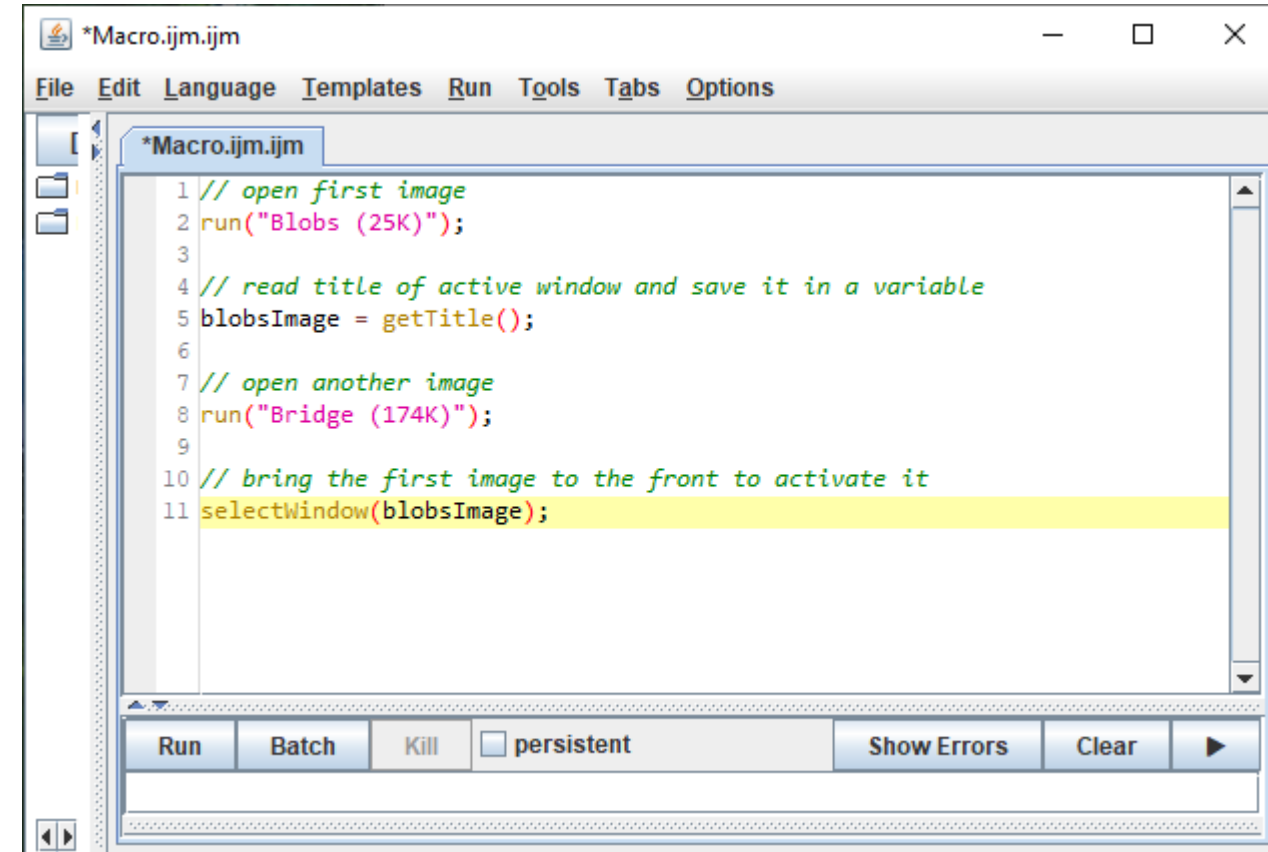
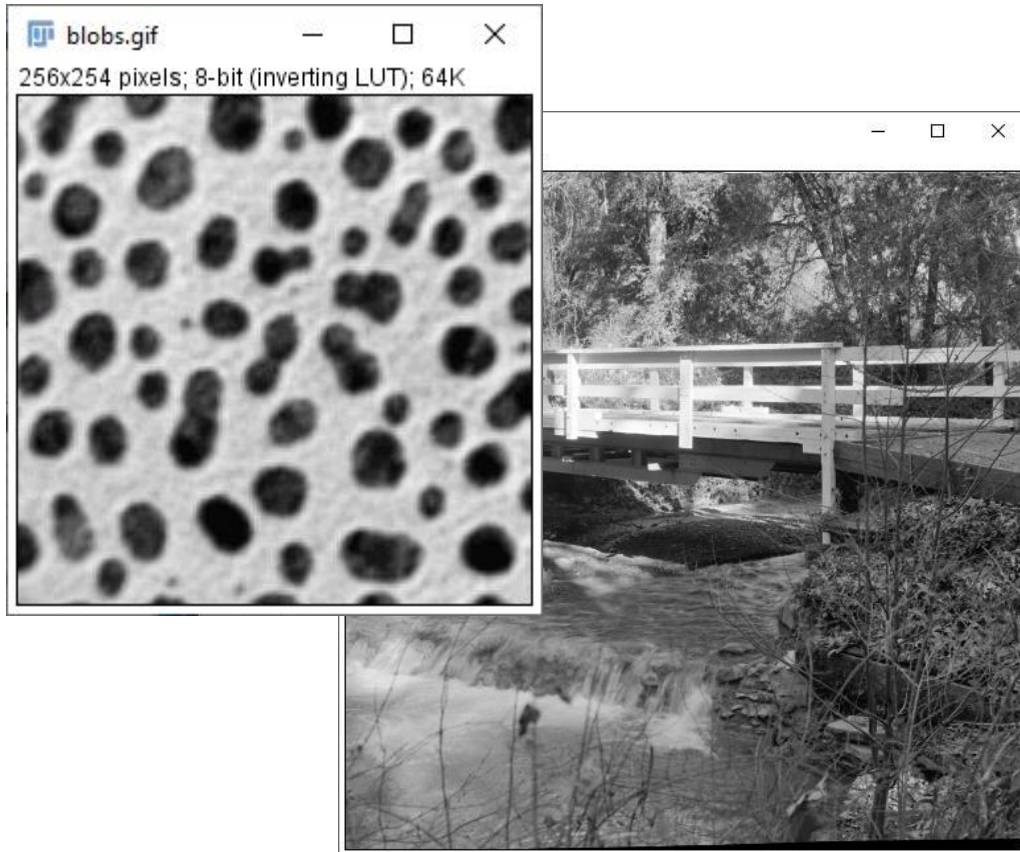
close();
```



image_statistics.ijm

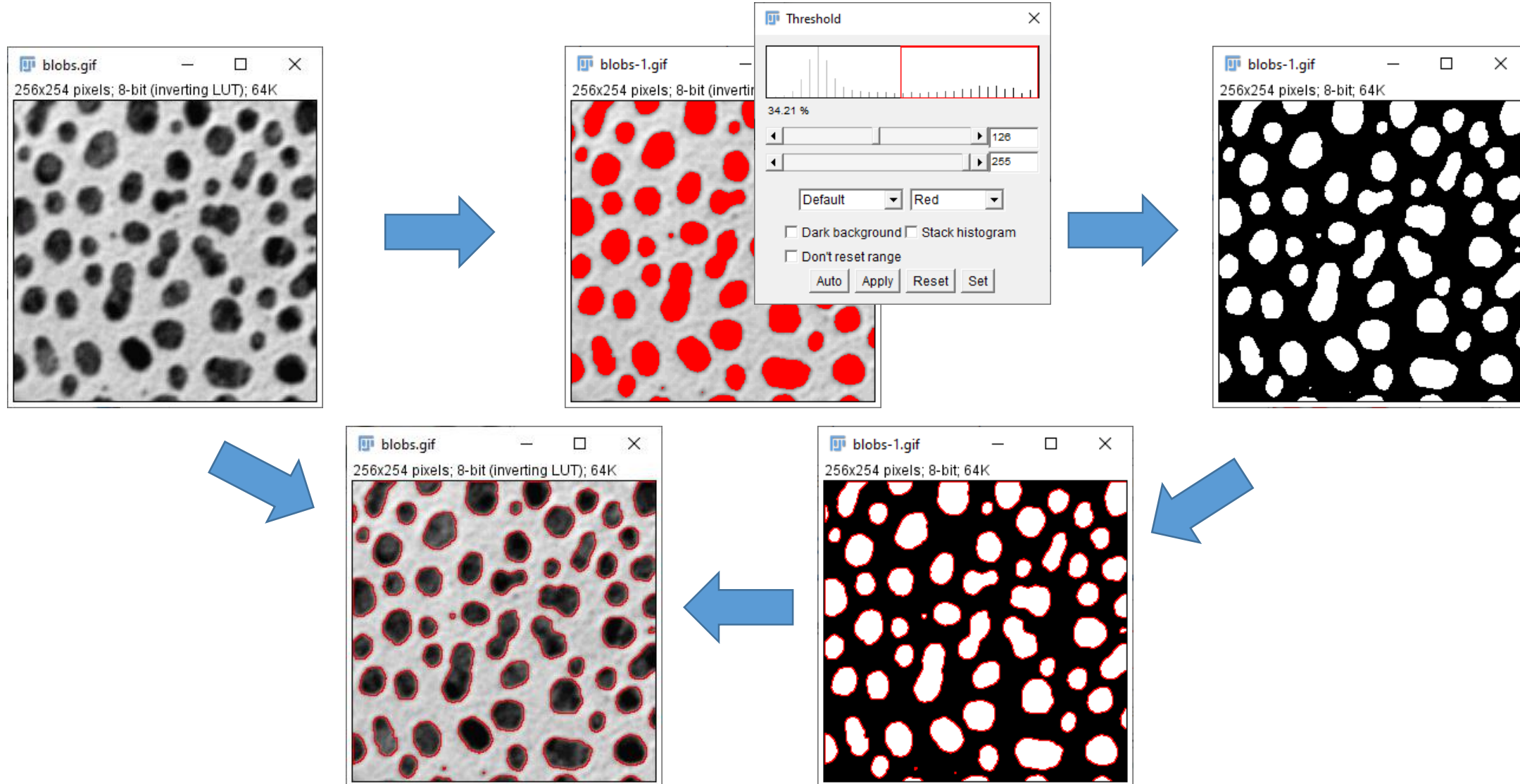
Built-in command	Description	Parameters
<code>getStatistics(area, mean, min, max, std)</code>	get statistics of an image or a region in the image	area in physical units, mean average grey value, minimum grey value, maximum grey value, standard deviation of grey value

- If you work with several images, you may want to switch between them. Use
 - getTitle() to get the headline of the current window
 - selectWindow(title) to select a window to bring it to the front
 - rename(new_title) to change the name of a window



Switching between windows

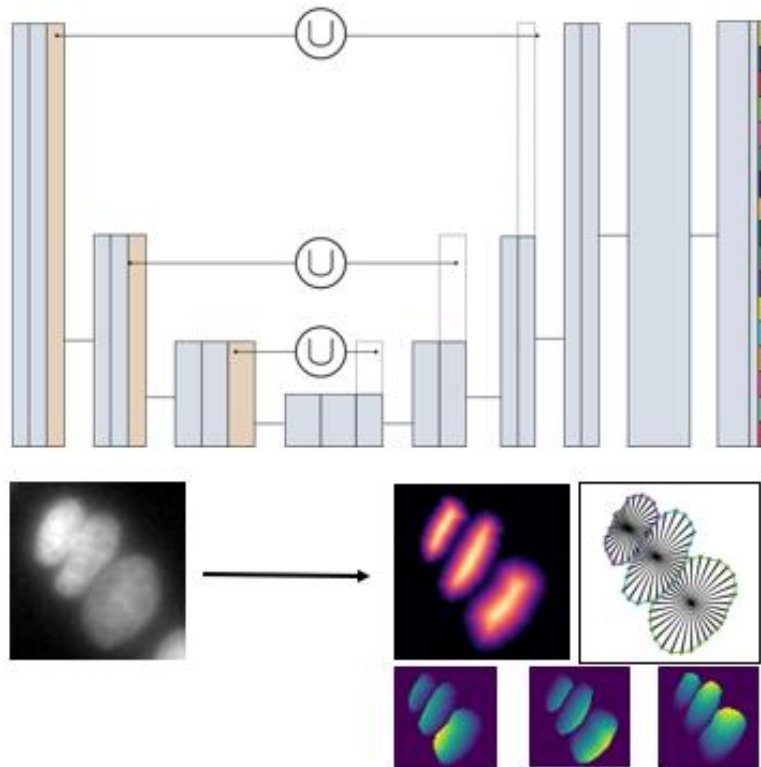
- Before “destroying” an image, make a duplicate!



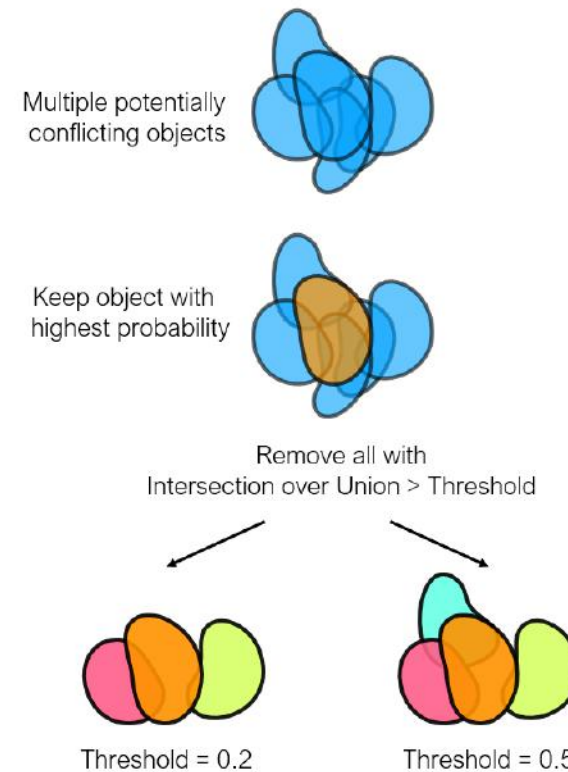
- Better segmentation using modern computational methods (deep learning)

and additional constraints

Dense Candidate Prediction

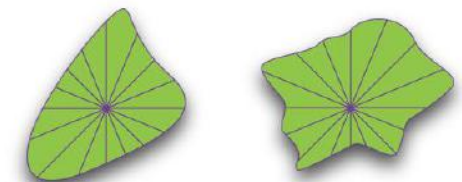


Non-Maximum-Suppression (NMS)

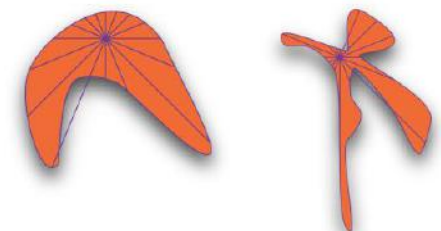


Important:
Assumes objects are *star-convex*

Star-Convex



Not Star-Convex

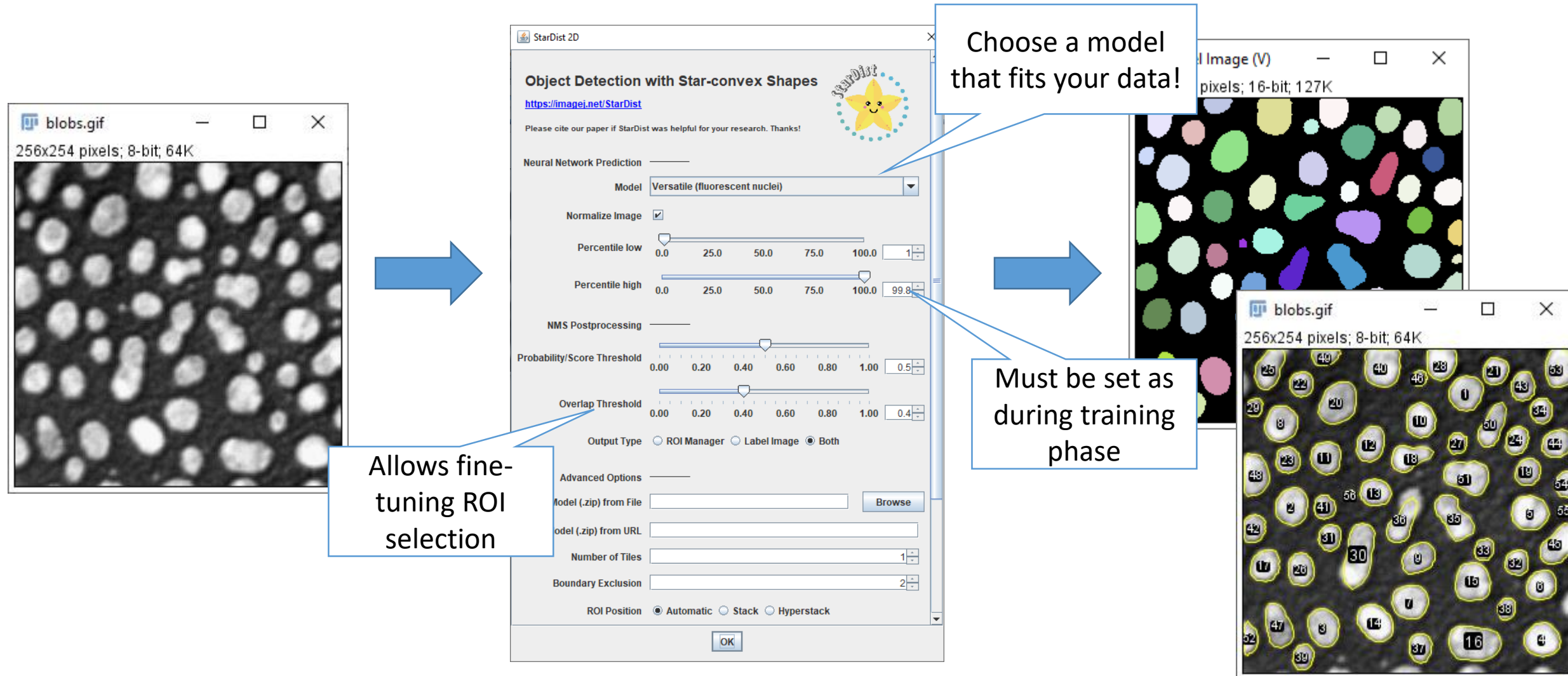


Slide adapted / courtesy of Martin Weigert, EPFL  @martweig

EPFL

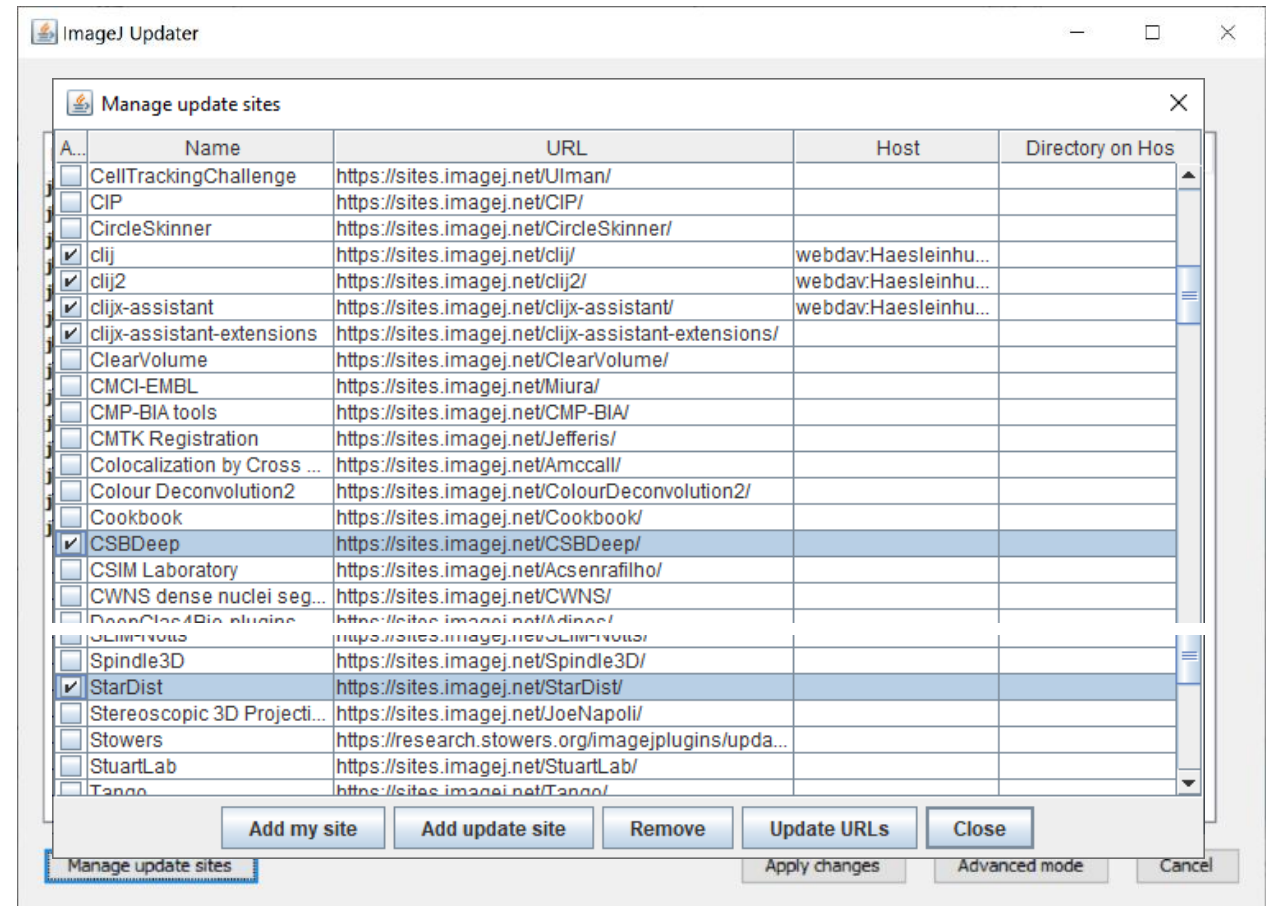
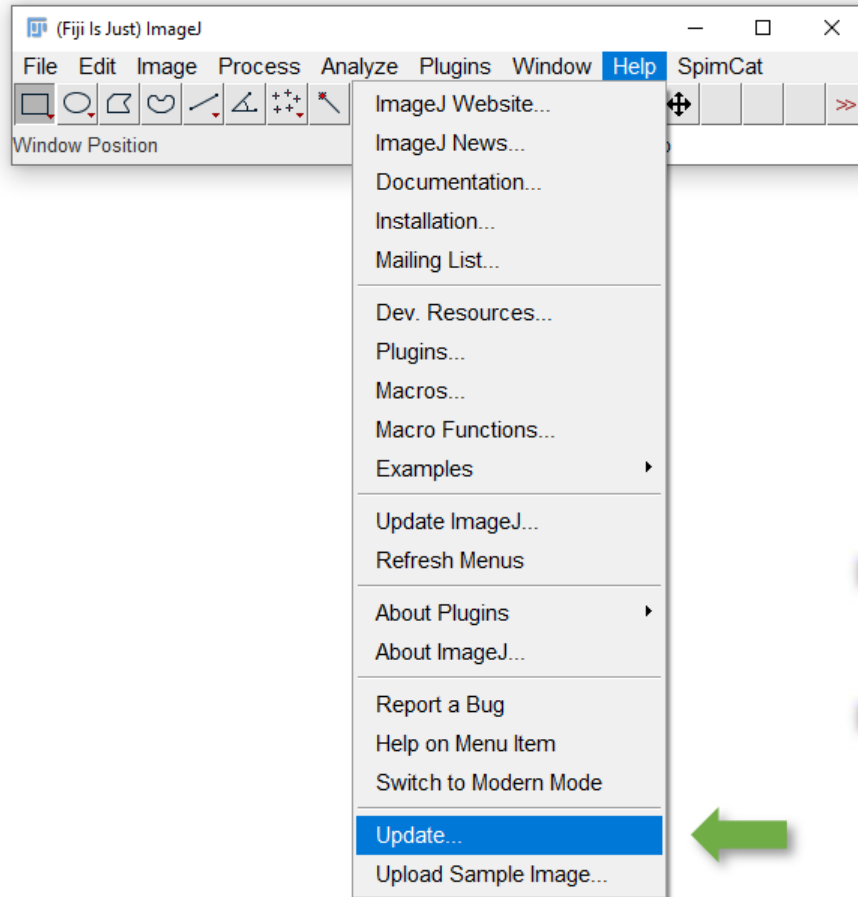
Deep-learning based segmentation: StarDist

- Deep-learning based image segmentation



Exercise: StarDist Installation

- Activate the “CSBDeep” and “StarDist” update sites.





Break: 15 min

Image data flow graphs

Robert Haase

With materials from
Elisabeth Kugler, The University of Sheffield,
Daniela Vorkel, MPI CBG / CSBD

September 2021

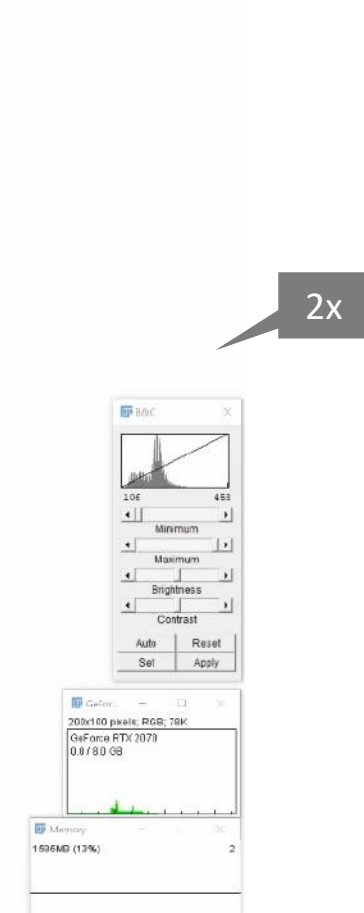
- State-of-the-art software for more than 20 years: ImageJ / Fiji



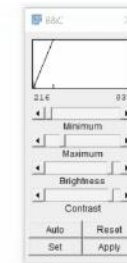
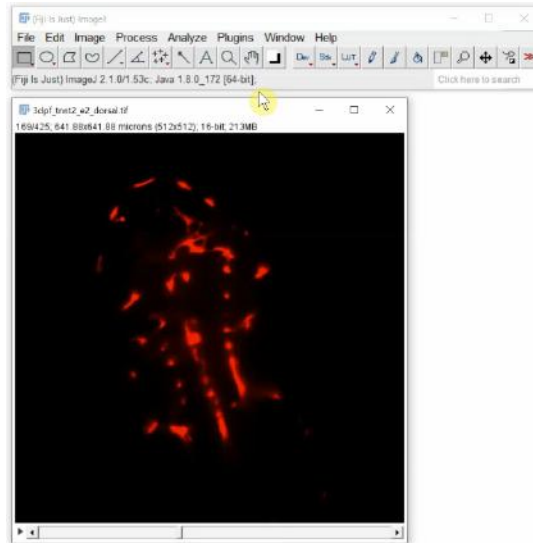
2x

How image processing is supposed to be

- (... in my honest opinion)



- After setting up the workflow, generate code!



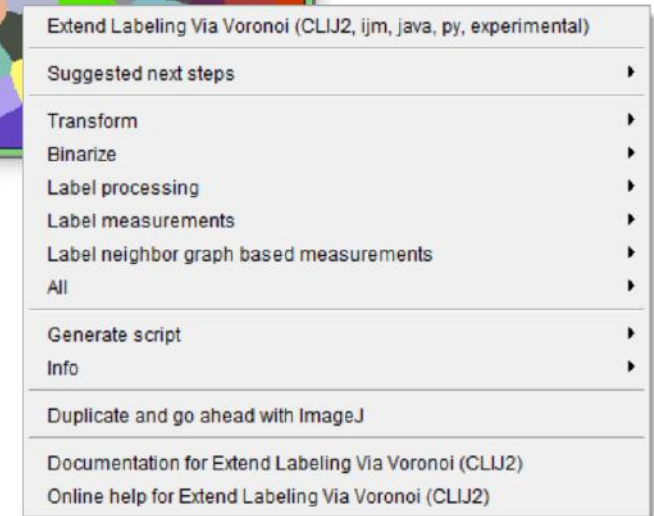
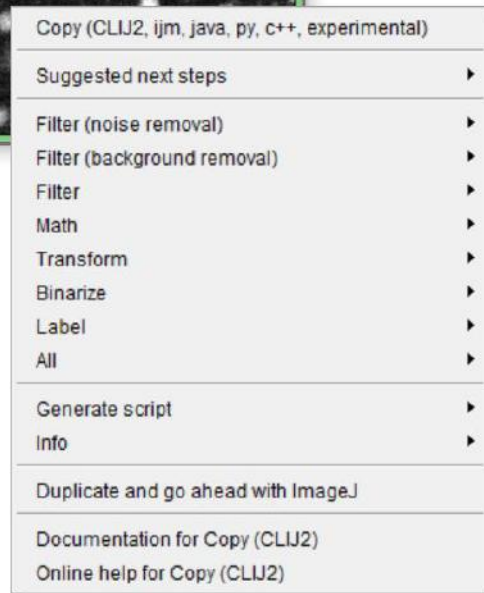
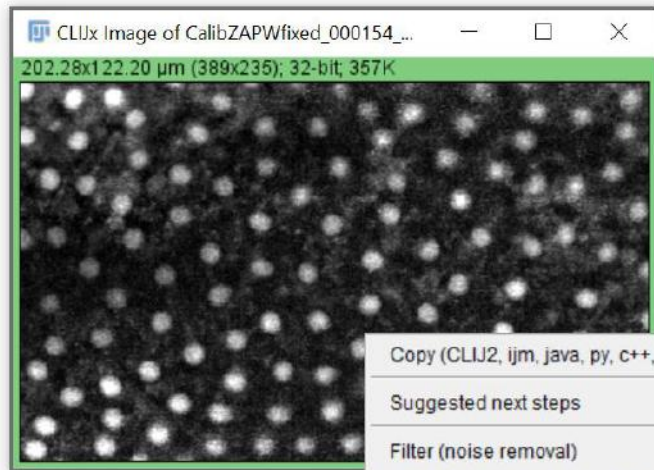
Special
thanks to
Elisabeth
Kugler!



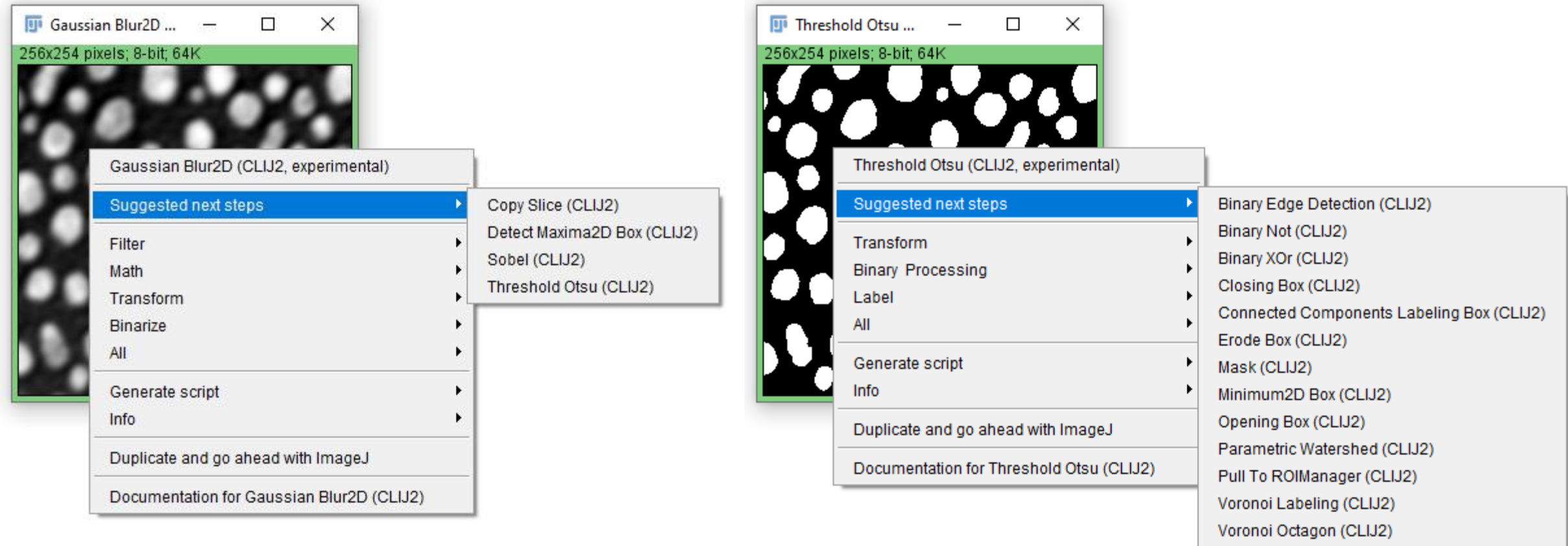
Elisabeth Kugler
@KuglerElisabeth

Image data source: Elisabeth Kugler; labs of Tim Chico and Paul Armitage, The University of Sheffield (UK)" <https://zenodo.org/record/4204839#.X8DCRGj7Q2w>

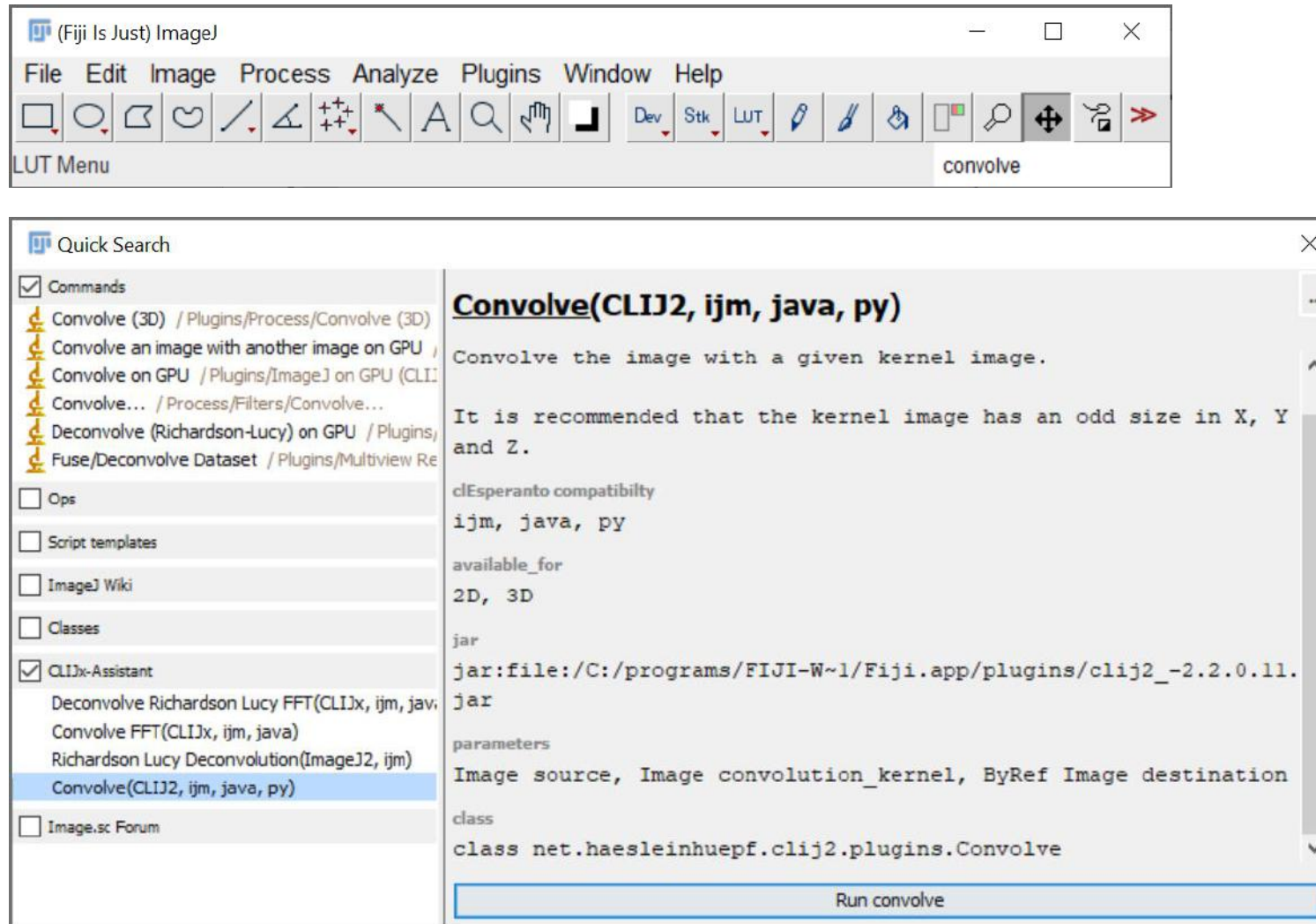
- The menu order is intentional: From preprocessing to analysis



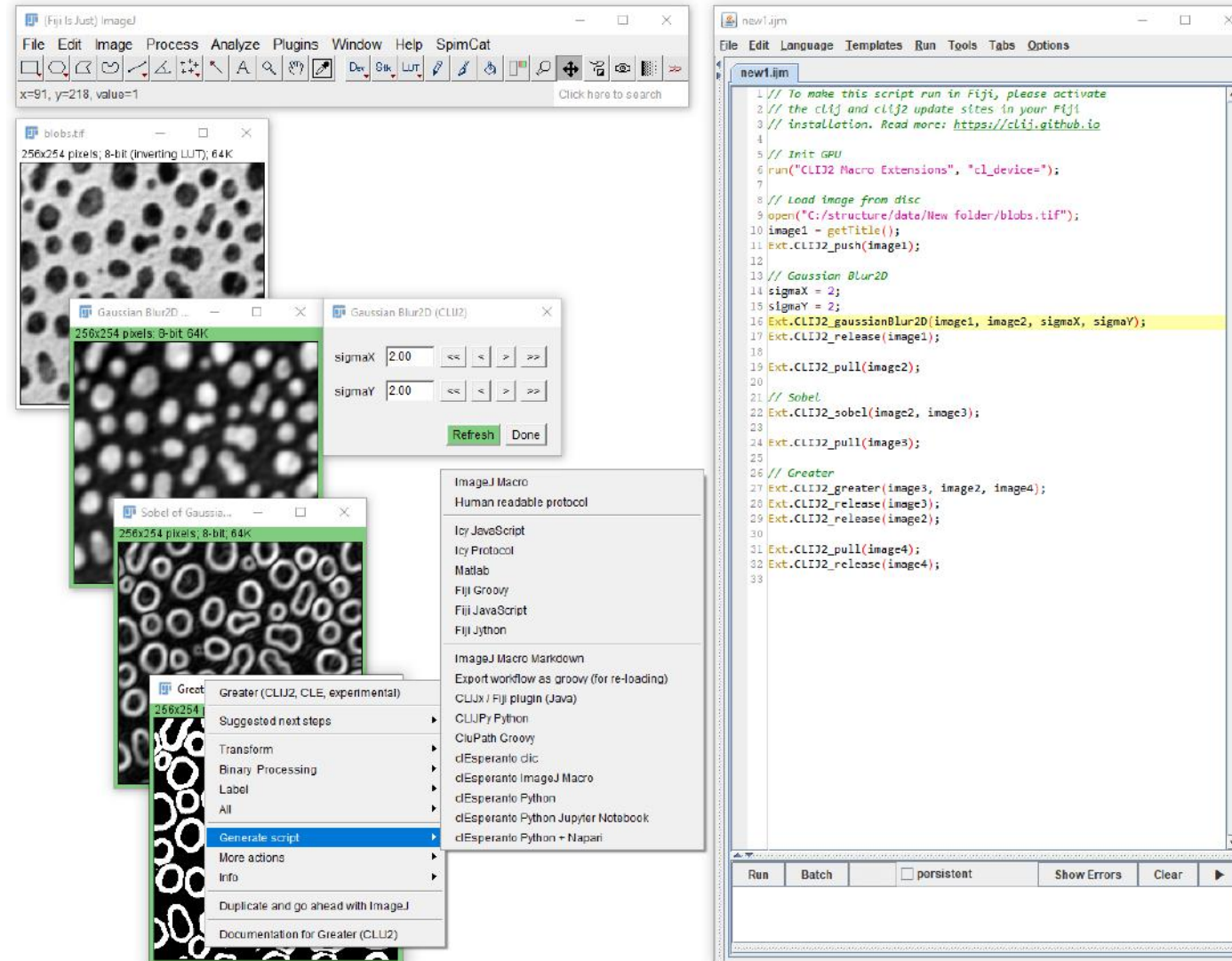
- Explore suggestions!



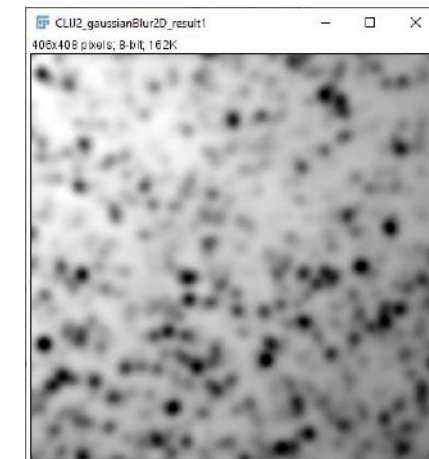
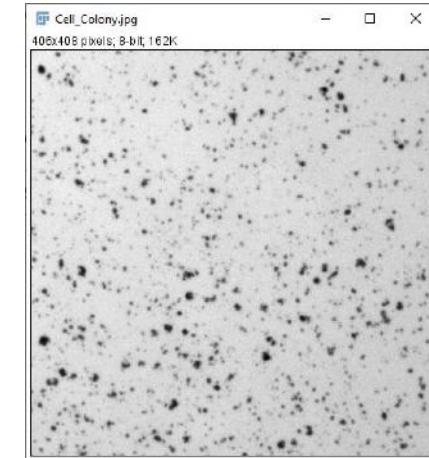
- In Fijis search bar result, there is a new category: CLIJx-assistant which offers IDFG operations



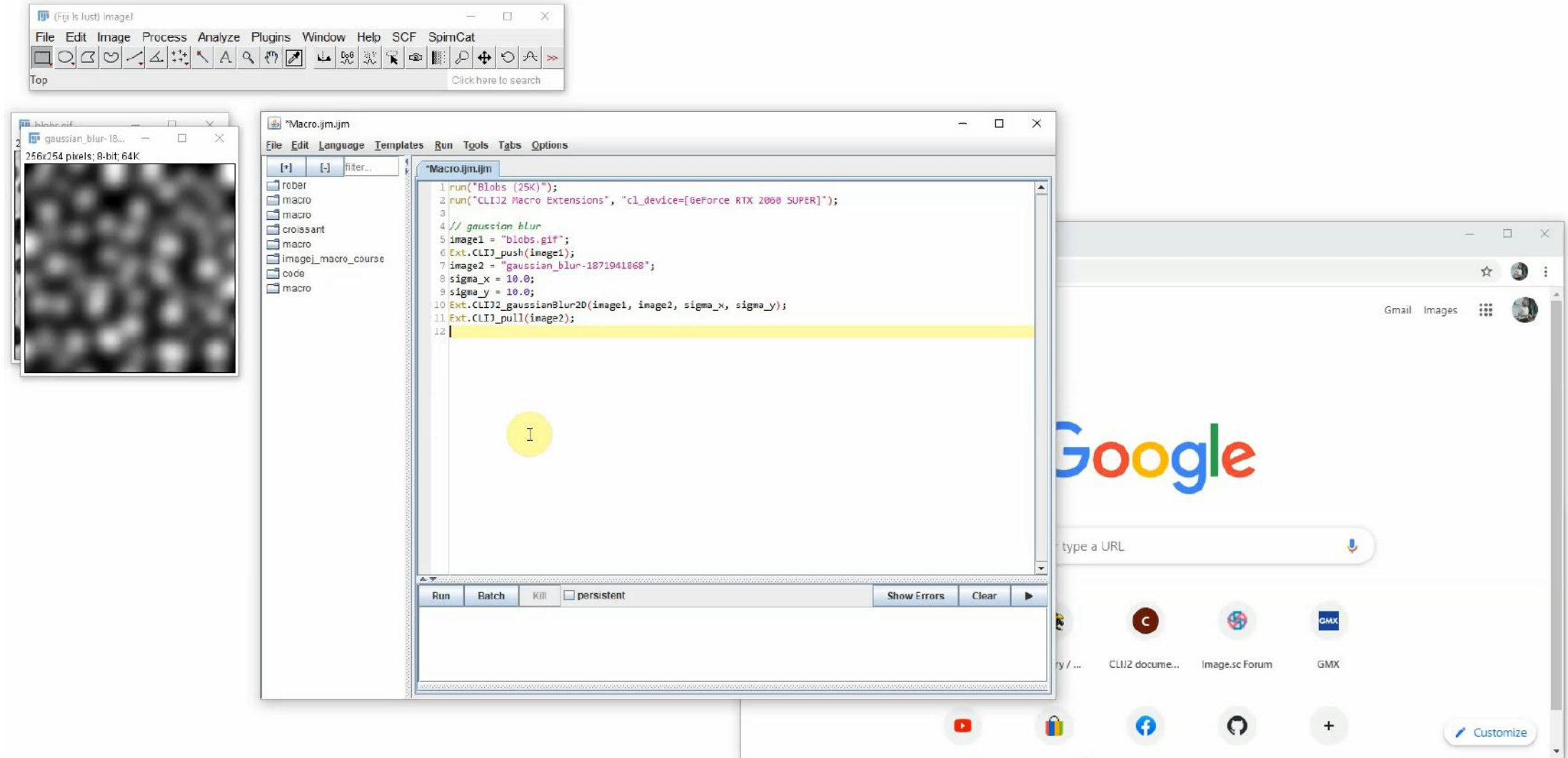
- Generate scripts in multiple languages from a given Image Data Flow Graph



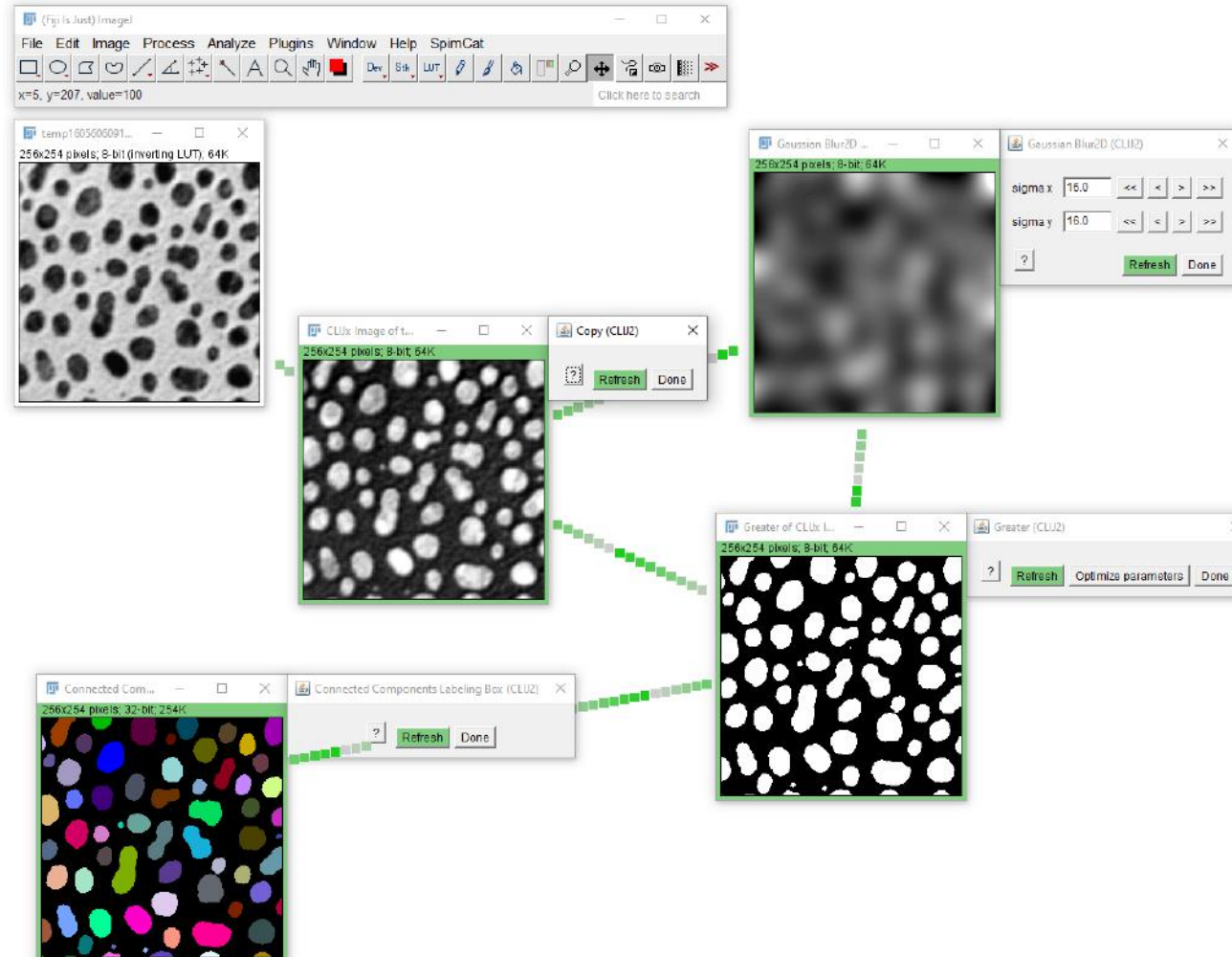
CLIJ2: What every ImageJ Macro script must have

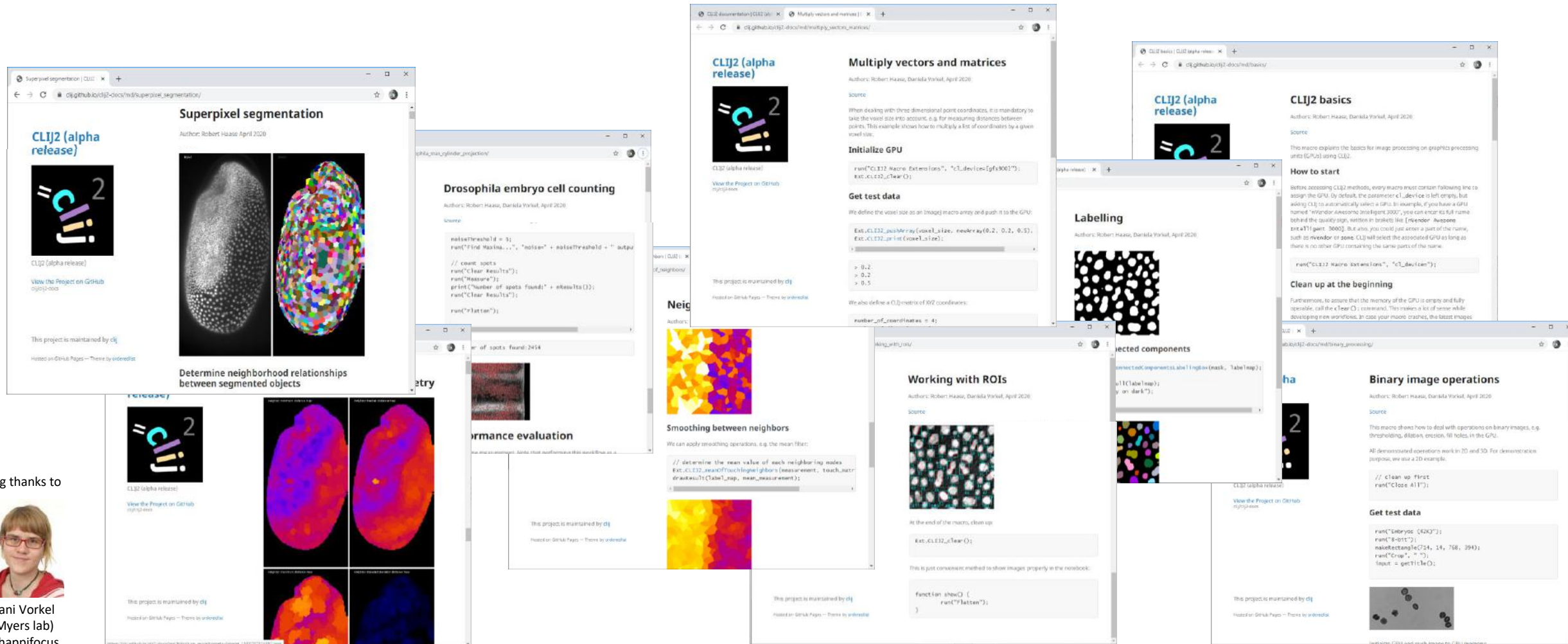


- Use the integrated online help!



- Design a workflow for segmenting blobs.gif (File > Open Samples...)
- Export the workflow as ImageJ Macro script for and Fiji Jython.





Big thanks to



Dani Vorkel
(Myers lab)
@happifocus

- Cheat sheets show the most important methods with input and output parameters visually.

CLIJ2 cheat sheet: ImageJ macro I

GPU-accelerated image processing in Fiji

Operation	Parameters	Result	Dim
Initialize CLIJ	[], HD, GFX or CPU		
Push			2D 3D
Pull			2D 3D
Create	1024, 1024, 8		2D 3D
Convert			2D 3D
Copy			2D 3D
Copy slice	50		2D 3D
Crop	20, 20		2D 3D
Paste	9, 9		2D 3D
Release			2D 3D
Clear			2D 3D

CLIJ2 cheat sheet: ImageJ macro II

GPU-accelerated image processing in Fiji

Operation	Parameters	Result	Dim	Examples
Filters	Gaussian blur	, 10, 10	2D 3D	<code>Ext.CLIJ2_gaussianBlur2D(input, result, sigmaX, sigmaY);</code>
	Difference of Gaussian	, 2, 2, 20, 20	2D 3D	<code>Ext.CLIJ2_differenceOfGaussian2D(input, result, sigma1x, sigma1y, sigma2x, sigma2y);</code>
	Invert		2D 3D	<code>Ext.CLIJ2_invert(input, result);</code>
	Laplace		2D 3D	<code>Ext.CLIJ2_laplaceBox(input, result);</code>
	Mean	, 5, 5	2D 3D	<code>Ext.CLIJ2_mean2DBox(input, result, radiusX, radiusY);</code>

CLIJ2 cheat sheet: ImageJ macro III

GPU-accelerated image processing in Fiji

CLIJ2 cheat sheet: ImageJ macro IV

GPU-accelerated image processing in Fiji

https://clij.github.io/clij2-docs/CLIJ2-cheatsheet_V3.pdf

- If you work with CLIJ and friends, please cite the paper(s). It'll be hard to apply for grants otherwise.

nature > nature methods > correspondence > article

nature methods



Cornell University

the Sir

Correspondence | Published: 18 November

CLIJ: GPU-accelerated image processing for everyone

Robert Haase , Loic A. Royer , Peter Steiner , Dilyan Dibrov, Uwe Schmidt, Martin Weigert, Nicolai H. Voigt, Eugene W. Myers

Nature Methods **17**, 5–6(2020) | Cite this article

arXiv.org > cs > arXiv:2008.11799

Computer Science > Mathematical Software

[Submitted on 26 Aug 2020]

GPU-accelerating ImageJ Macro image processing

Daniela Vorkel, Robert Haase

This chapter introduces GPU-accelerated image processing in ImageJ. Core concepts such as variables, for-loops, and functions are explained. We present in a step-by-step tutorial how to translate ImageJ macros into GPU-accelerated code.

Subjects: **Mathematical Software (cs.MS)**; Distributed, Parallel, and Cluster Computing

Cite as: arXiv:2008.11799 [cs.MS]

(or arXiv:2008.11799v1 [cs.MS] for this version)

Submission history

From: Robert Haase [\[view email\]](#)

[v1] Wed, 26 Aug 2020 20:38:31 UTC (2,079 KB)



Cold Spring Harbor Laboratory

bioRxiv

THE PREPRINT SERVER FOR BIOLOGY












HOME | ABOUT

Search

New Results

 Comments (1)

Interactive design of GPU-accelerated Image Data Flow Graphs and cross-platform deployment using multi-lingual code generation

 Robert Haase,  Akanksha Jain,  Stéphane Rigaud,  Daniela Vorkel,  Pradeep Rajasekhar,  Theresa Suckert,  Talley J. Lambert,  Juan Nunez-Iglesias,  Daniel P. Poole,  Pavel Tomancak,  Eugene W. Myers

doi: <https://doi.org/10.1101/2020.11.19.386565>

This article is a preprint and has not been certified by peer review [what does this mean?].

Abstract

Full Text

Info/History

Metrics

 Preview PDF

Exercises

With material from

Anne Esslinger, Albert Lab Biotec, MPI CBG

Benoit Lombardot, Scientific Computing Facility, MPI CBG

October 2021

Exercise: basic math with variables

- Calculate the distance of two points!

```
// initialise program
x1 = 3;
y1 = 5;

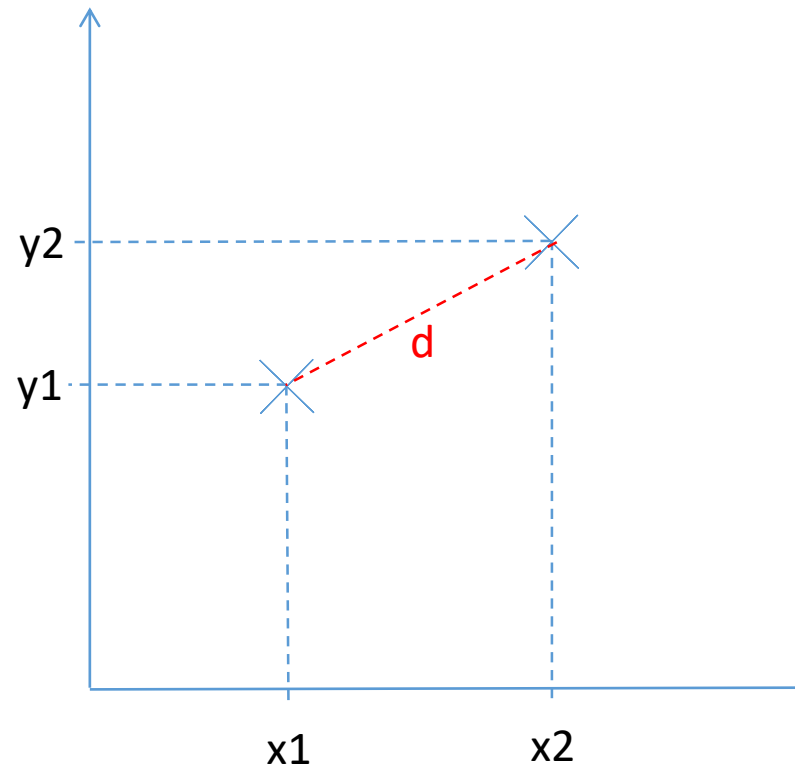
x2 = 7;
y2 = 8;

// run complicated algorithm

d = sqrt(pow(x1 - x2, 2) +
          pow(y1 - y2, 2));

print("The distance is " + d);
```

exercise_math.ijm



Optional Exercise: Write a function doing this! Hint:

```
1 |
2 | function distance(x1, y1, x2, y2) {
3 |
4 | }
5 |
```

- Play with the macro language. Try things like the following and observe what's happening.

```
// initialise program
a = "1";
b = 2;

// do some calculations
print(a + b);
print(b + a);
```

```
// initialise program
a = "1";
b = 2;

// do some calculations
print(a / b);
print(b / a);
```

```
// initialise program
a = "1";
b = 2;

// do some calculations
print(0 + a + b);
print("" + b + a);
```

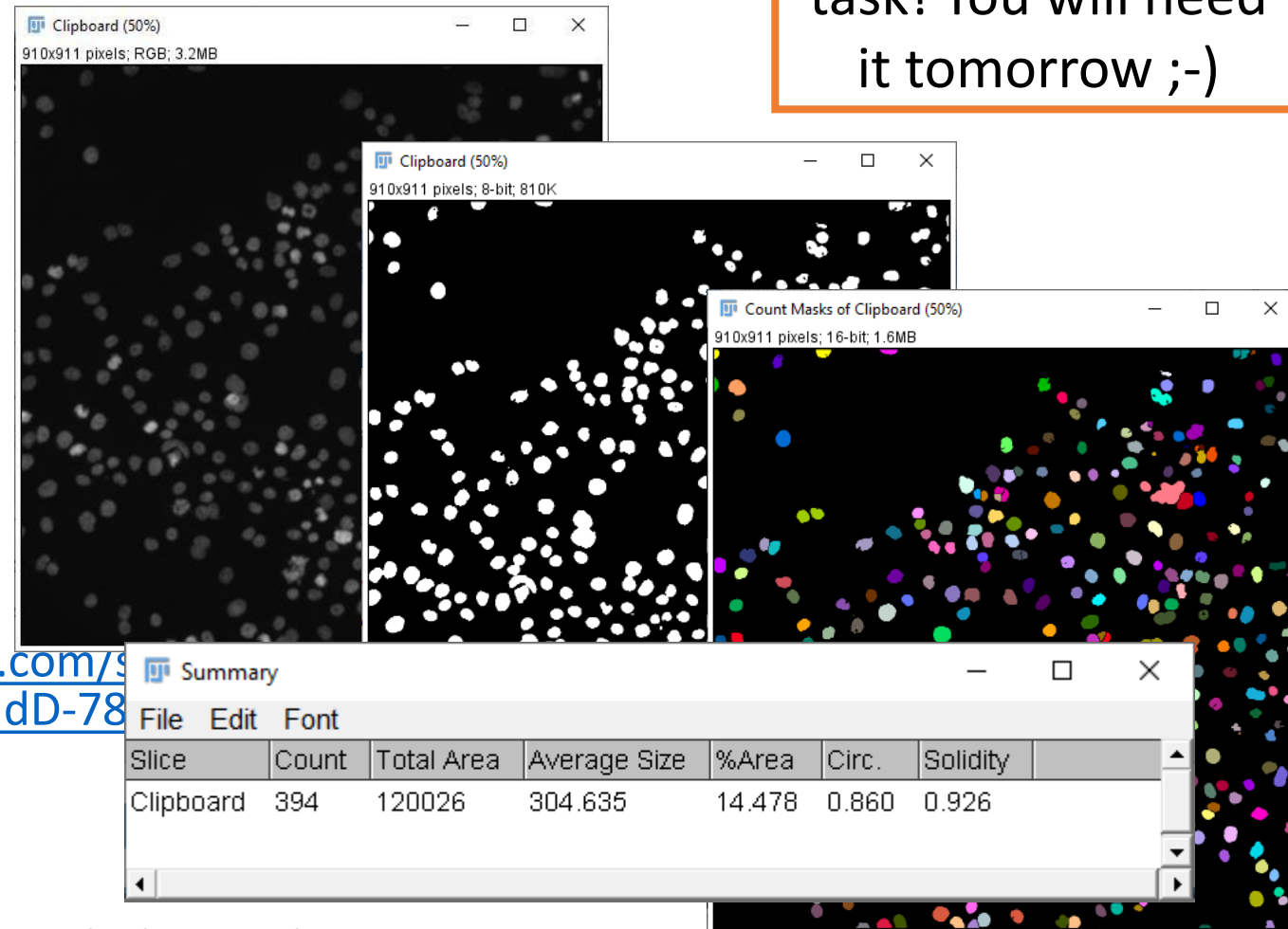
```
// initialise program
a = "1";
b = 2;

// do some calculations
print(a / b);
print(b / a);
```

Exercise: Segmentation

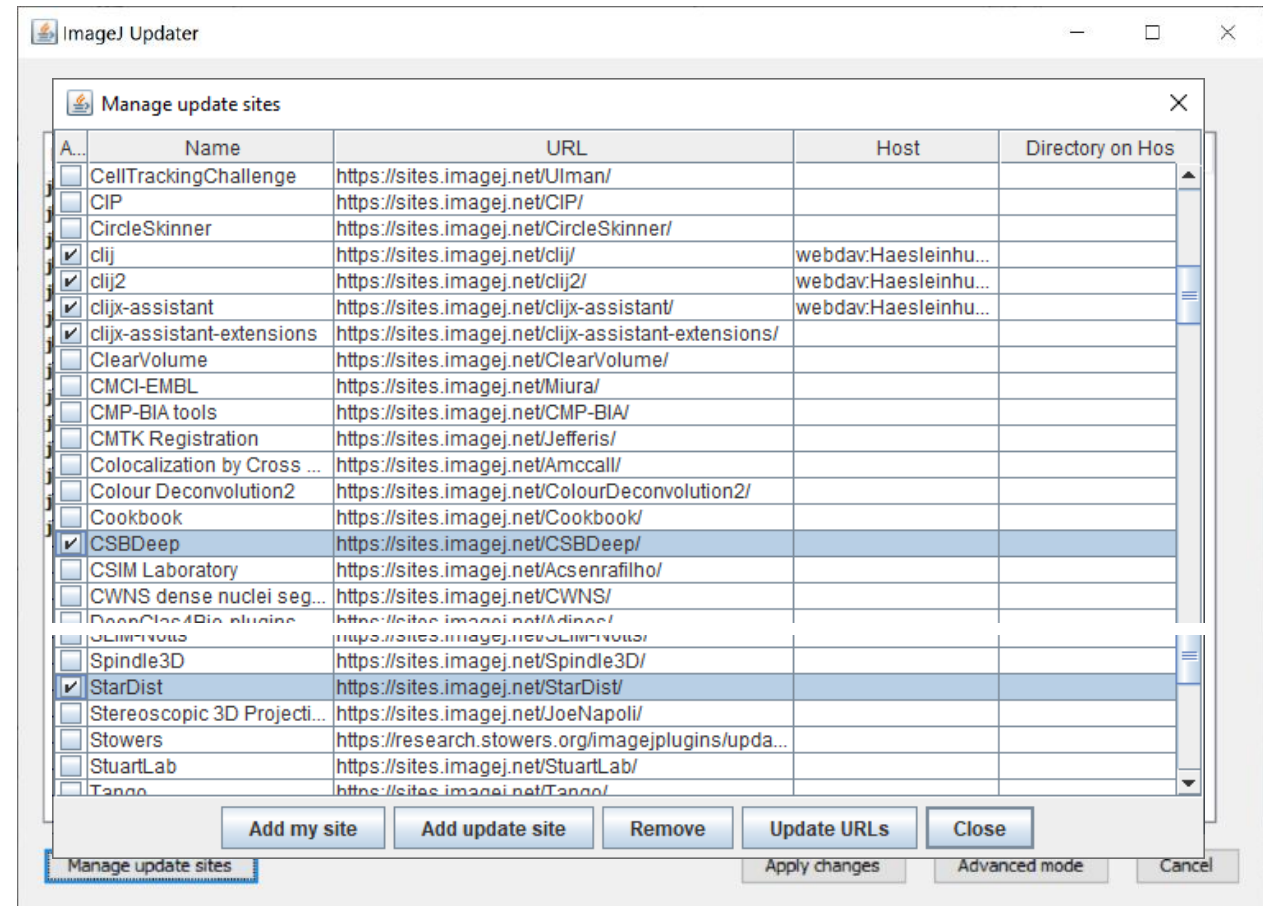
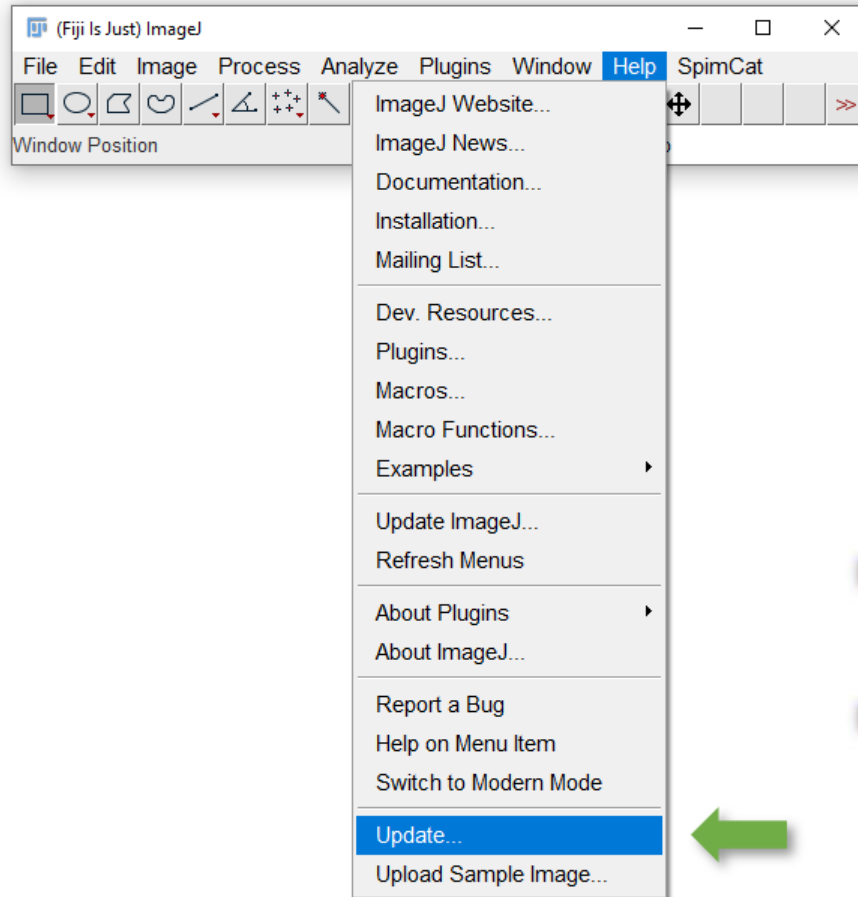
- Download and install Fiji
 - <http://fiji.sc/Downloads>
- Download the BBBC001 dataset (TIF), unzip it and count nuclei in the first image file
 - <https://bbbc.broadinstitute.org/BBBC001>
 - AS_09125_050118150001_A03f00d0.tif
- Hints:
 - Use automatic thresholding and the Watershed algorithm
 - Summarize your measurement in the Particle Analyzer
 - Select “Show: Count Masks”
 - Use the “Glasbey on dark” lookup table / color map
- Enter your count here: <https://docs.google.com/sheets/d/23Soro5XZ3y1kJHpvaTaa1f4n2C7G3WX0qddD-78/edit>
- Be a good scientist: Don't cheat! ;-)

Record and curate a macro for this task! You will need it tomorrow ;-)



Exercise: StarDist Installation

- Activate the “CSBDeep” and “StarDist” update sites.



Exercise

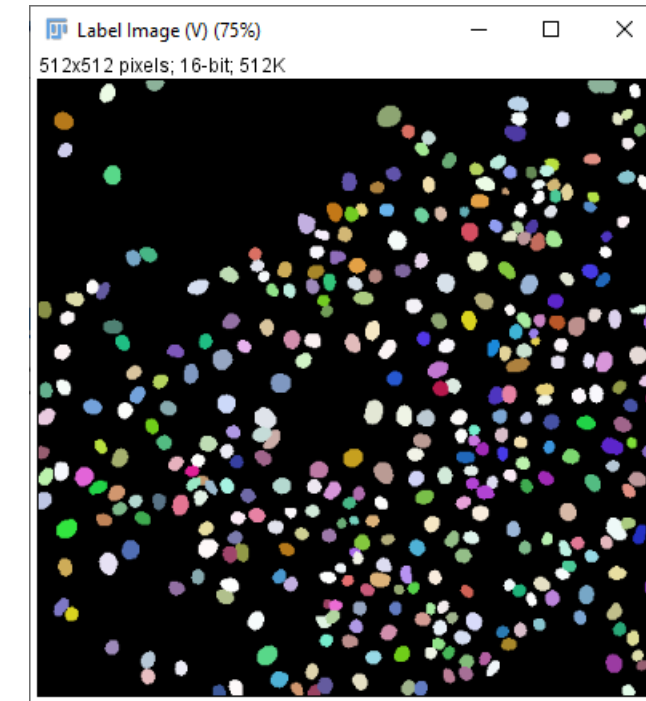
- Download and install Fiji
 - <http://fiji.sc/Downloads>
- Download the BBBC001 dataset (TIF), unzip it and count nuclei in the first image file
 - <https://bbbc.broadinstitute.org/BBBC001>
 - AS_09125_050118150001_A03f00d0.tif

Hints:

- Use Plugins > StarDist > StarDist2D to segment the nuclei
- Use Analyze > Set Measurements and Analyze > Measure to determine the maximum intensity in the label image (= number of labels)



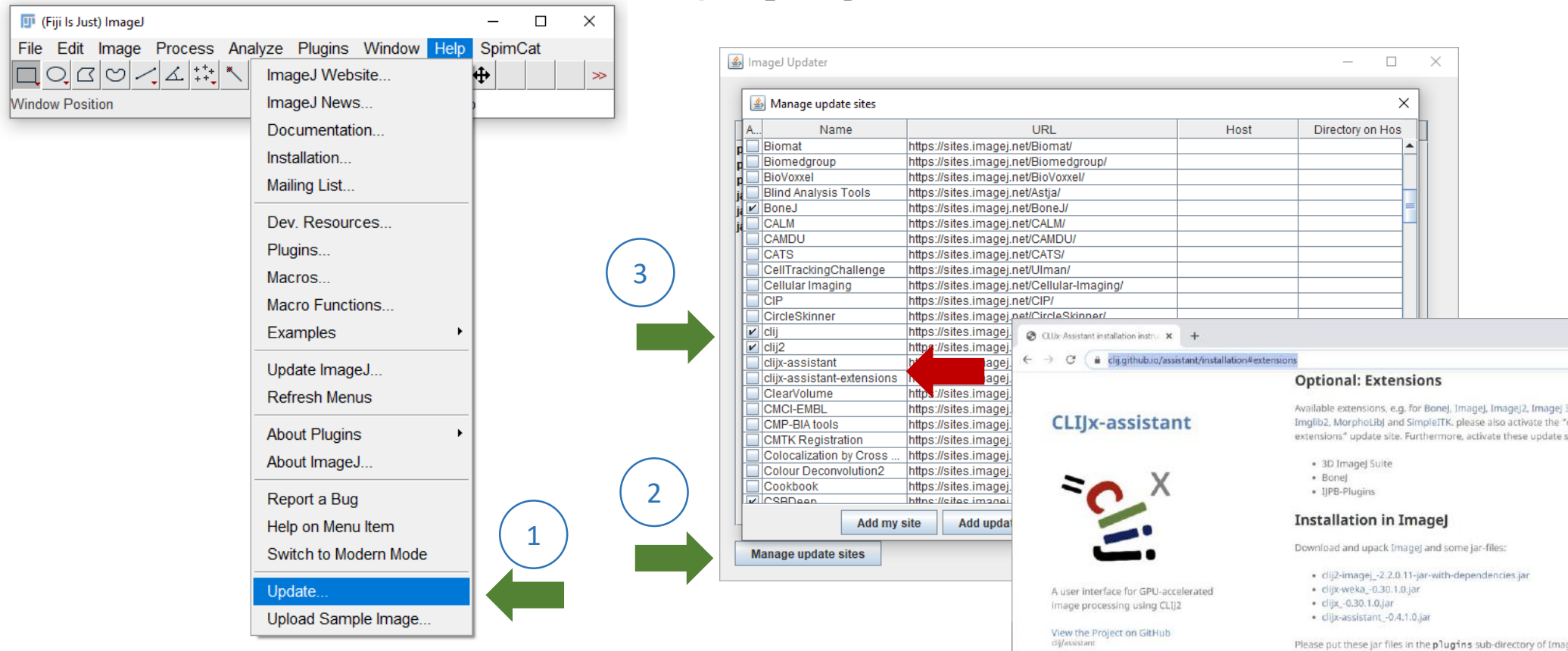
Record and curate
a macro for this
task! You will need
it tomorrow ;-)



- Enter your count here: <https://docs.google.com/spreadsheets/d/1Ek-23Soro5XZ3y1kJHpvaTaa1f4n2C7G3WX0qddD-78/edit?usp=sharing>
- Be a good scientist: Don't cheat! ;-)

Exercise CLIJ Installation

- Just activate the CLIJ and CLIJ2 update sites, in menu Help > Update...
 - Do *not* install clijx! That's experimental stuff you're free to try out later.
- Linux users: you may need to install OpenCL, e.g. `apt-get install ocl-icd-devel`



The screenshot illustrates the installation process for CLIJ in ImageJ. It shows three main steps:

- Step 1:** In the ImageJ application, the 'Help' menu is open, and the 'Update...' option is selected.
- Step 2:** The 'ImageJ Updater' window is open, showing a list of update sites. The 'CLIJ' and 'CLIJ2' sites are checked.
- Step 3:** The 'Manage update sites' dialog is open, showing a table of update sites. The 'CLIJ' and 'CLIJ2' sites are checked.

The 'Manage update sites' dialog contains the following table:

Name	URL	Host	Directory on Host
Biomat	https://sites.imagej.net/Biomat/		
Biomedgroup	https://sites.imagej.net/Biomedgroup/		
BioVoxxel	https://sites.imagej.net/BioVoxxel/		
Blind Analysis Tools	https://sites.imagej.net/Astja/		
BoneJ	https://sites.imagej.net/BoneJ/		
CALM	https://sites.imagej.net/CALM/		
CAMDU	https://sites.imagej.net/CAMDU/		
CATS	https://sites.imagej.net/CATS/		
CellTrackingChallenge	https://sites.imagej.net/Ulman/		
Cellular Imaging	https://sites.imagej.net/Cellular-Imaging/		
CIP	https://sites.imagej.net/CIP/		
CircleSkinner	https://sites.imagej.net/CircleSkinner/		
CLIJ	https://sites.imagej.net/CLIJ/		
CLIJ2	https://sites.imagej.net/CLIJ2/		
CLIJx-assistant	https://sites.imagej.net/CLIJx-assistant/		
CLIJx-assistant-extensions	https://sites.imagej.net/CLIJx-assistant-extensions/		
ClearVolume	https://sites.imagej.net/ClearVolume/		
CMCI-EMBL	https://sites.imagej.net/CMCI-EMBL/		
CMP-BIA tools	https://sites.imagej.net/CMP-BIA tools/		
CMTK Registration	https://sites.imagej.net/CMTK Registration/		
Colocalization by Cross ...	https://sites.imagej.net/Colocalization by Cross .../		
Colour Deconvolution2	https://sites.imagej.net/Colour Deconvolution2/		
Cookbook	https://sites.imagej.net/Cookbook/		
CSRD	https://sites.imagej.net/CSRD/		

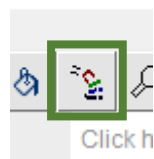
The 'CLIJx-assistant' window is also shown, providing information about the assistant and its extensions.

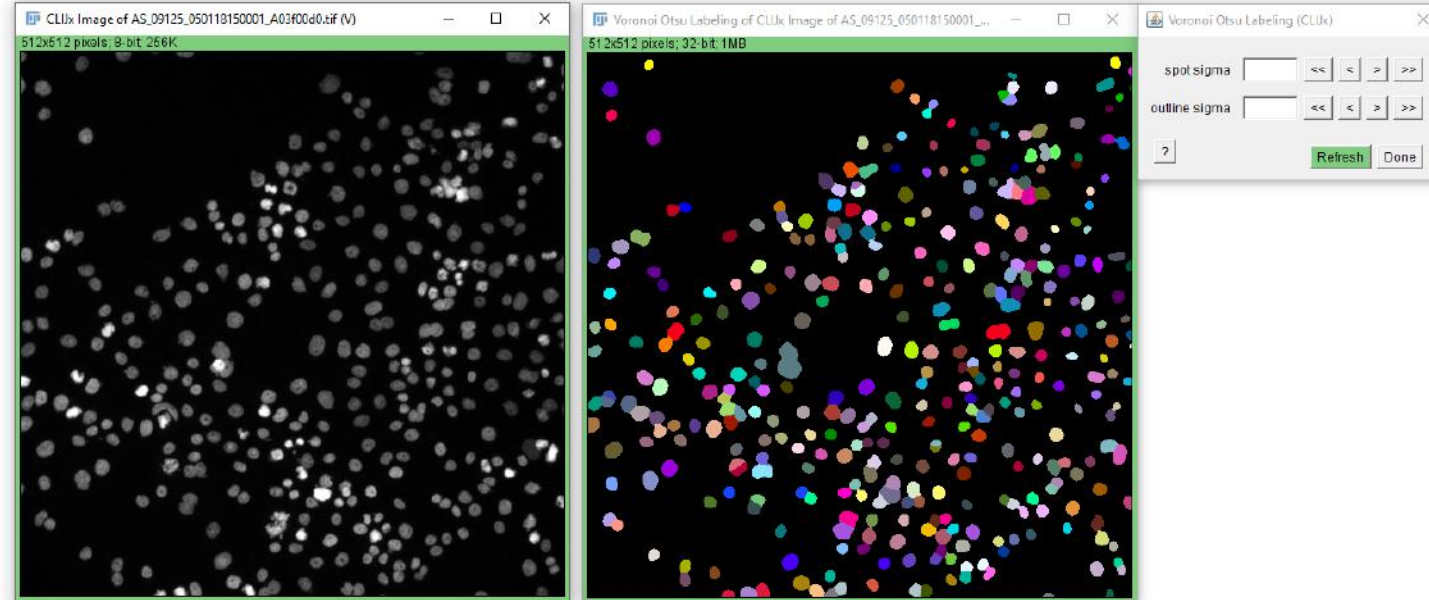
Exercise

- Download and install Fiji
 - <http://fiji.sc/Downloads>
- Download the BBBC001 dataset (TIF), unzip it and count nuclei in the first image file
 - <https://bbbc.broadinstitute.org/BBBC001>
 - AS_09125_050118150001_A03f00d0.tif

Generate a macro
for this task! You
will need it
tomorrow ;-)

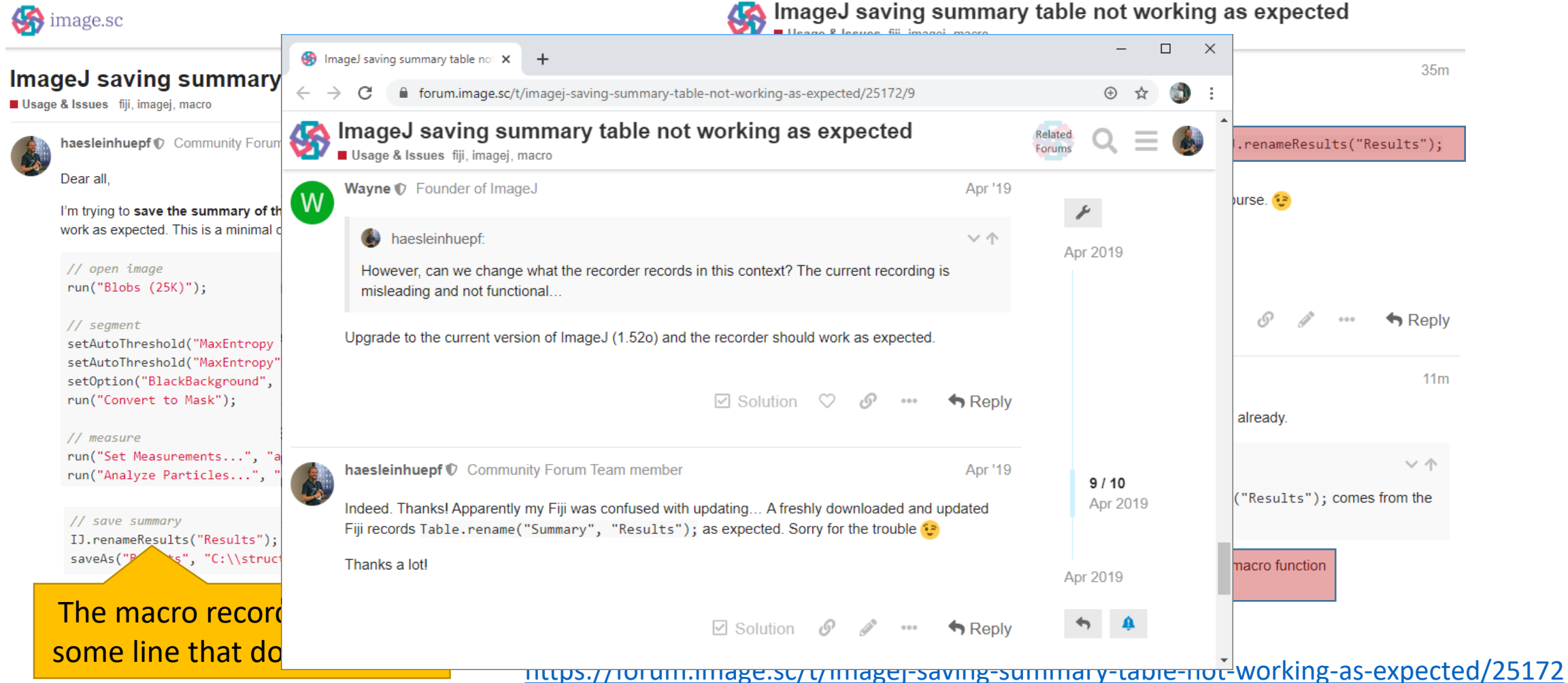
Hints:

- Use CLIJ Assistant [Right-click menu] Label > Voronoi-Otsu-Labeling to segment the nuclei 
- Use Analyze > Set Measurements and Analyze > Measure to determine the maximum intensity in the label image (= number of labels)



- Enter your count here: <https://docs.google.com/spreadsheets/d/1Ek-23Soro5XZ3y1kJHpvaTaa1f4n2C7G3WX0qddD-78/edit?usp=sharing>
- Be a good scientist: Don't cheat! ;-)

- In case you run into trouble: Visit image.sc and ask. They may ask within 5 hours, even Saturdays.



The screenshot shows a forum thread on image.sc. The thread title is "ImageJ saving summary table not working as expected". The original poster, haesleinhuepf, describes a problem with saving the summary table. Wayne, the founder of ImageJ, responds with a solution: "Upgrade to the current version of ImageJ (1.52o) and the recorder should work as expected." haesleinhuepf replies, thanking Wayne and stating that the problem was solved after updating Fiji. A yellow callout box points to a line of code in the macro: `IJ.renameResults("Results");`. The URL at the bottom of the screenshot is <https://forum.image.sc/t/imagej-saving-summary-table-not-working-as-expected/25172>.

image.sc

ImageJ saving summary

Usage & Issues fiji, imagej, macro

Dear all,

I'm trying to **save the summary of the work** as expected. This is a minimal code snippet:

```
// open image
run("Blobs (25K)");

// segment
setAutoThreshold("MaxEntropy");
setAutoThreshold("MaxEntropy");
setOption("BlackBackground", true);
run("Convert to Mask");

// measure
run("Set Measurements...", "area perimeter centroid");
run("Analyze Particles...", "size");

// save summary
IJ.renameResults("Results");
saveAs("Blobs", "C:\\structure\\Blobs.txt");
```

The macro recorder recorded some line that does not work.

ImageJ saving summary table not working as expected

Usage & Issues fiji, imagej, macro

Wayne Founder of ImageJ Apr '19

haesleinhuepf:

However, can we change what the recorder records in this context? The current recording is misleading and not functional...

Upgrade to the current version of ImageJ (1.52o) and the recorder should work as expected.

haesleinhuepf Community Forum Team member Apr '19

Indeed. Thanks! Apparently my Fiji was confused with updating... A freshly downloaded and updated Fiji records `Table.rename("Summary", "Results");` as expected. Sorry for the trouble 😊

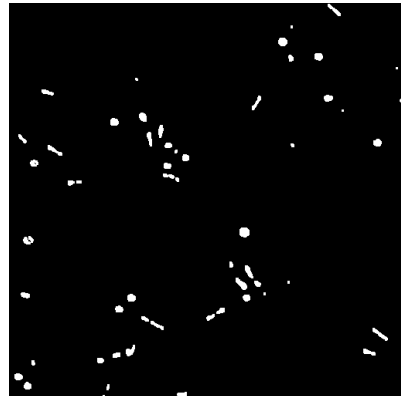
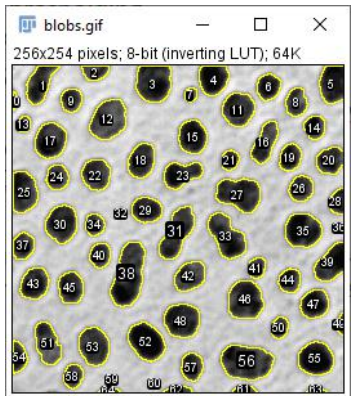
Thanks a lot!

9 / 10 Apr 2019

<https://forum.image.sc/t/imagej-saving-summary-table-not-working-as-expected/25172>

Today, you learned

- ImageJ macro
 - Variables
 - Working with files and images
 - Macro recording
 - Using the ROI manager
 - Custom functions
- Exercises:
 - Macro recording and
 - image segmentation



Coming up next

- ImageJ macro
 - Loops and conditions
 - Interacting with tables
 - Custom dialogs
- How to write readable code.



Break: 21h

ImageJ macro programming

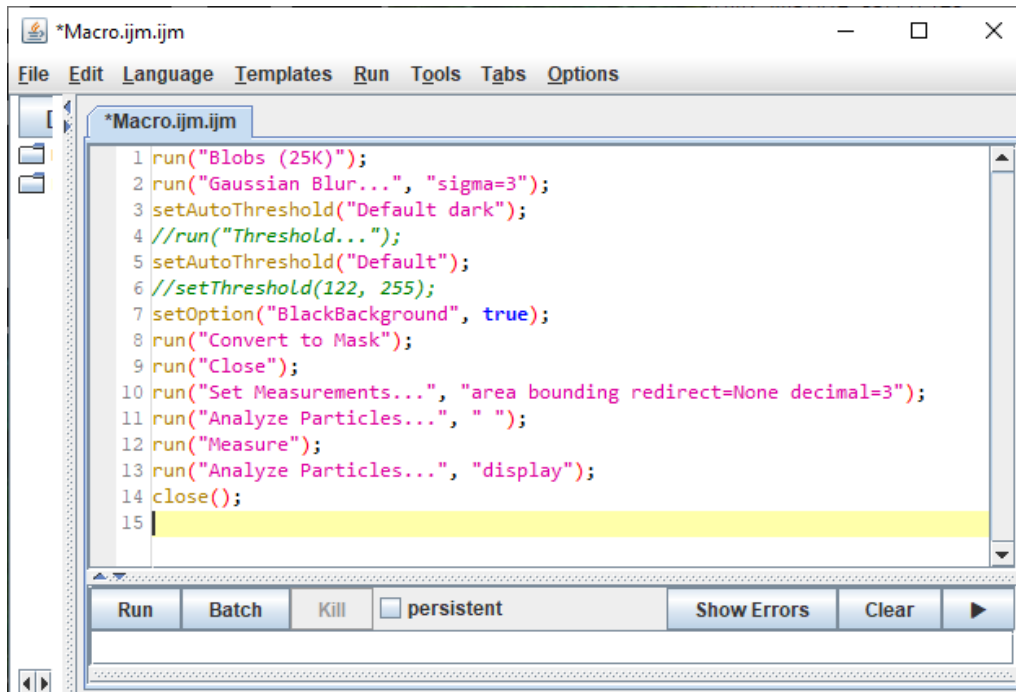
Robert Haase

With material from

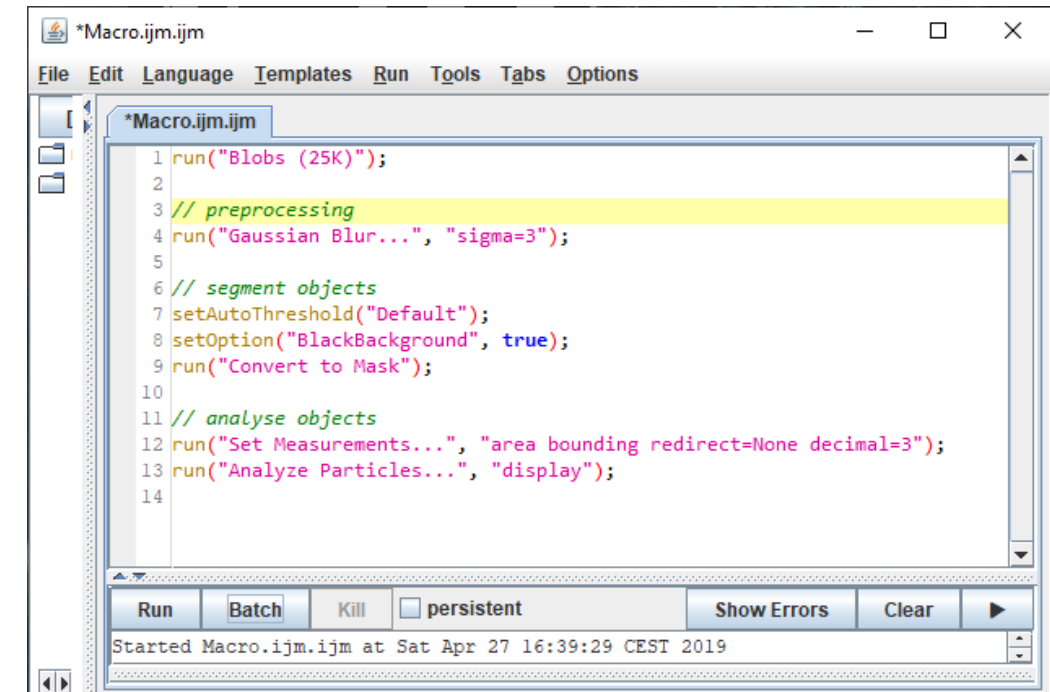
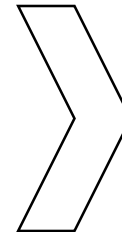
Benoit Lombardot, Scientific Computing Facility, MPI CBG

Virtually at CCI Gothenburg, October 2021

- Editing recorded macros needs to be trained. It's 80% reading and 20% writing
- Hints:
 - Put comments first. Try to understand what was recorded and why.
 - Do it in tiny steps. As soon as you have a working workflow consisting of 4-5 steps, create a macro.
 - Collect macros. When you do something new, do cherry picking from the old macros.



```
*Macro.ijm.ijm
File Edit Language Templates Run Tools Tabs Options
1 run("Blobs (25K)");
2 run("Gaussian Blur...", "sigma=3");
3 setAutoThreshold("Default dark");
4 //run("Threshold...");
5 setAutoThreshold("Default");
6 //setThreshold(122, 255);
7 setOption("BlackBackground", true);
8 run("Convert to Mask");
9 run("Close");
10 run("Set Measurements...", "area bounding redirect=None decimal=3");
11 run("Analyze Particles...", " ");
12 run("Measure");
13 run("Analyze Particles...", "display");
14 close();
15
```



```
*Macro.ijm.ijm
File Edit Language Templates Run Tools Tabs Options
1 run("Blobs (25K)");
2
3 // preprocessing
4 run("Gaussian Blur...", "sigma=3");
5
6 // segment objects
7 setAutoThreshold("Default");
8 setOption("BlackBackground", true);
9 run("Convert to Mask");
10
11 // analyse objects
12 run("Set Measurements...", "area bounding redirect=None decimal=3");
13 run("Analyze Particles...", "display");
14
```

Started Macro.ijm.ijm at Sat Apr 27 16:39:29 CEST 2019

ImageJ macro programming: Conditions and loops

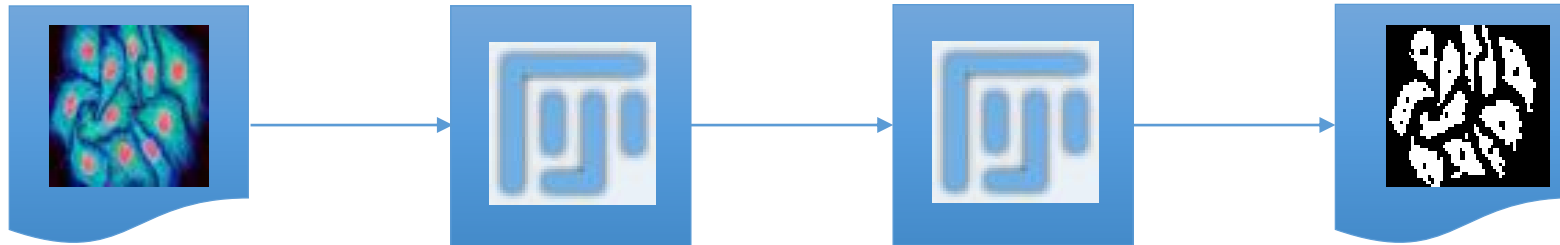
Robert Haase

With material from

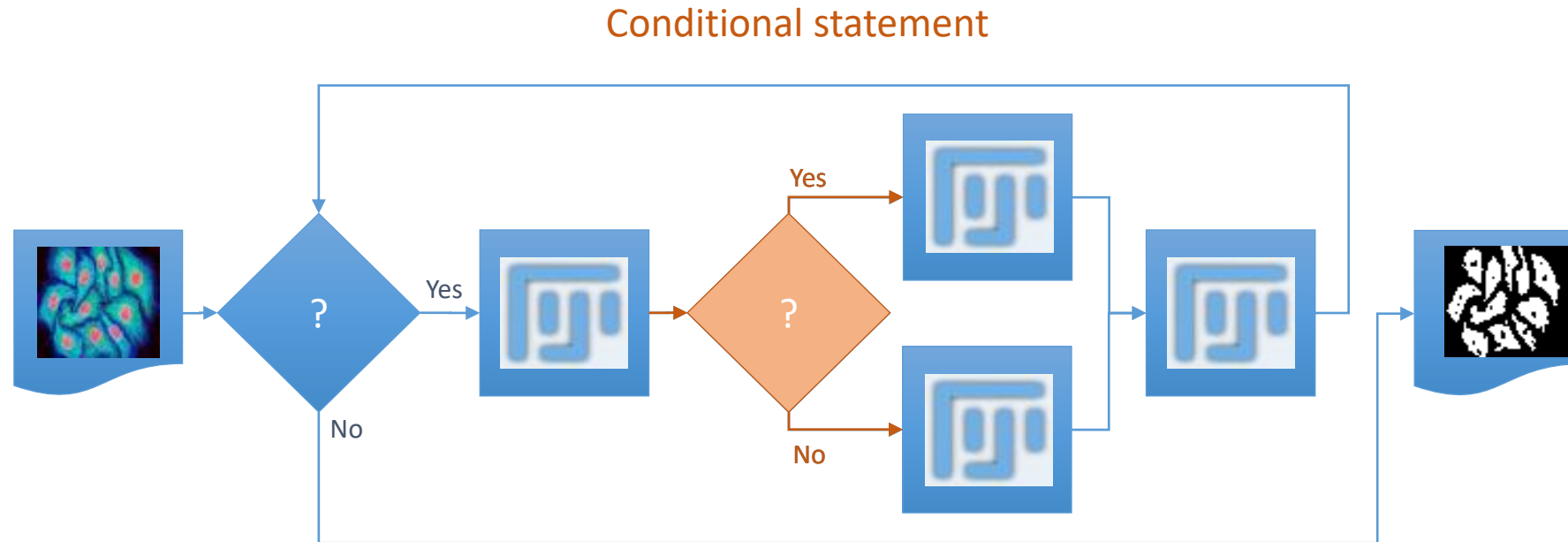
Benoit Lombardot, Scientific Computing Facility, MPI CBG

Virtually at CCI Gothenburg, October 2021

- Image processing workflows *rarely* look like this



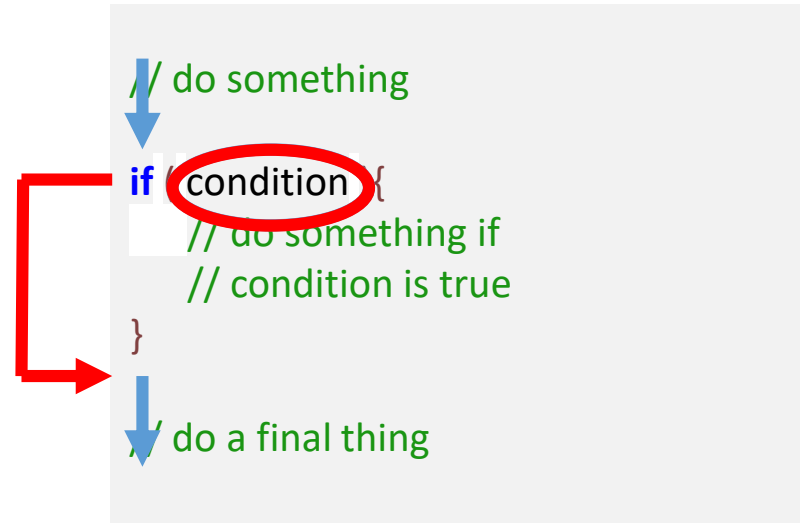
- Image processing workflows *rather* look like this



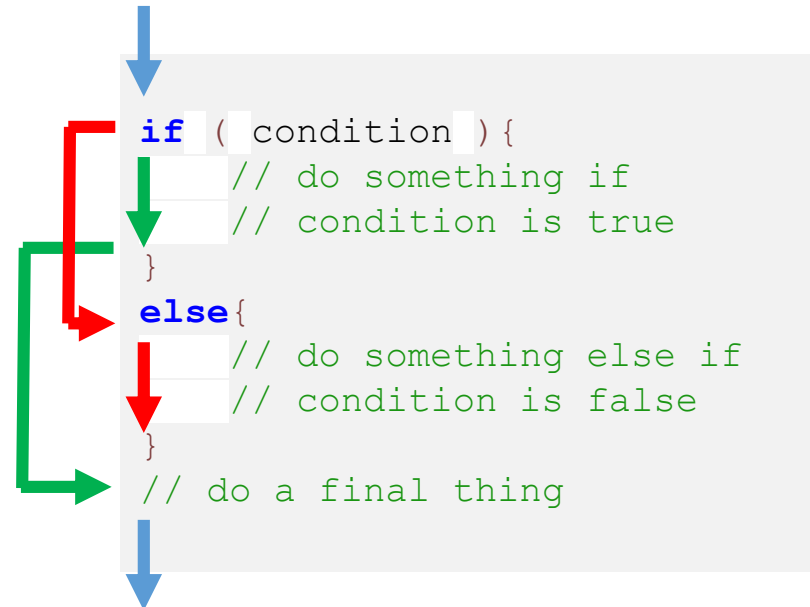
- Depending on a condition, some lines of code are executed or not.

```
// do something  
↓  
if condition {  
  ↓ // do something if  
  ↓ // condition is true  
}  
↓  
do a final thing
```

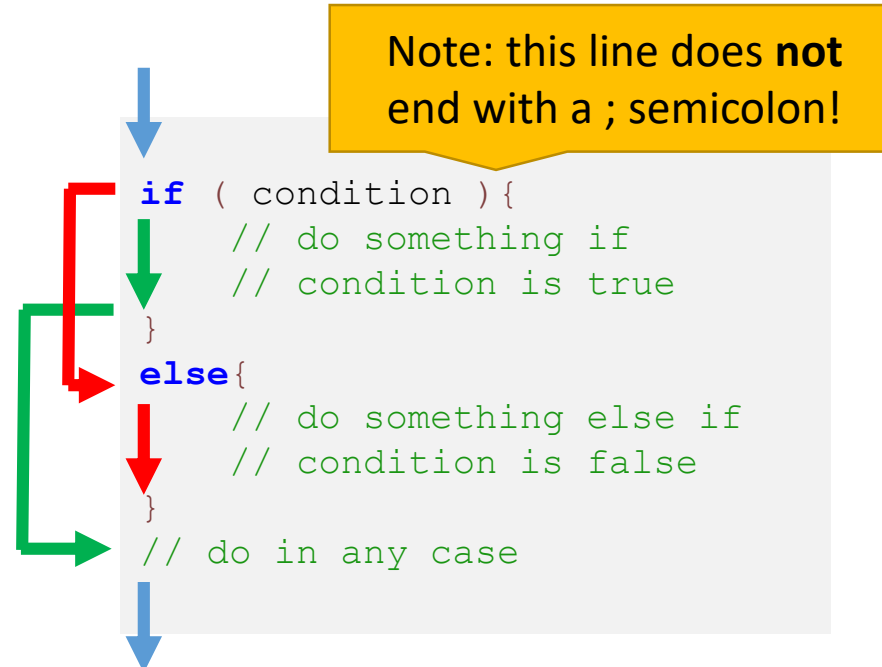
- Depending on a condition, some lines of code are executed or not.



- The else statement allows to program alternatives.



- The if / else statement allows to program alternatives.
- Depending on a condition, either the one or the other block is executed.
- Curly brackets {} are used to mark where a block starts and ends.
- Indentation helps reading blocks.



- Comparison operators always have *true* or *false* as results.

```
// initialise program
quality = 99.5;

// evaluate result
if (quality > 99.9) {
    print("Everything is fine.");
} else {
    print("We need to improve!");
}
```

Operator	Description	Example
<, <=	smaller than, smaller or equal to ¹	a < b
>, >=	greater than, greater or equal to ¹	a > b
==	equal to ¹	a == b
!=	not equal to ¹	a != 1

¹ these operators work also with string values

- Logic operators always take conditions as operands and result in a condition.
- Conditions can be either true (1) or false (0).

```
// initialise program
quality = 99.9;
age = 3;

if ((quality >= 99.9) && !(age > 5)) {
    print("The item is ok.");
}
```

Operator	Description	Example
&&	and	$a < b \ \&\& \ b < c$
	or	$a < b \ \ b > c$
!	negation ("not")	$!(a == b)$

Why is indentation important?

- Does this program say “Yin” or “Yang”?

```
// initialise program
a=5;b=3;c=8;
// execute algorithm
d=(a+b)/c;
// evaluate result
if (b>0) {
if (a==5)
{
print("Yin");
}
else
{
if (c<5)
{
}
}
if (d!=0)
{
print("Yang");
}
}
```

yin_yang.ijm

```
// initialise program
a = 5;
b = 3;
c = 8;

// execute algorithm
d = (a + b) / c;

// evaluate result
if (b > 0) {
    if (a == 5) {
        print("Yin");
    } else {
        if (c < 5) {}
    }
    if (d != 0) {
        print("Yang");
    }
}
```

Yes, it's the same
program as on the
left 😊

Yin

Yang

Rules for readable code

- Every command belongs on its own line
- Insert empty lines to separate important processing steps
- Put spaces between operators and operands, because:

This is easier to read than that, or isn't it?

- Indent every conditional block (if/else) using the TAB key
- Hint: put the “{” behind the if; it makes your program shorter.
- Make use of tools: <http://jsbeautifier.org>

```
// initialise program
a = 5;
b = 3;
c = 8;

// execute algorithm
d = (a + b) / c;

// evaluate result

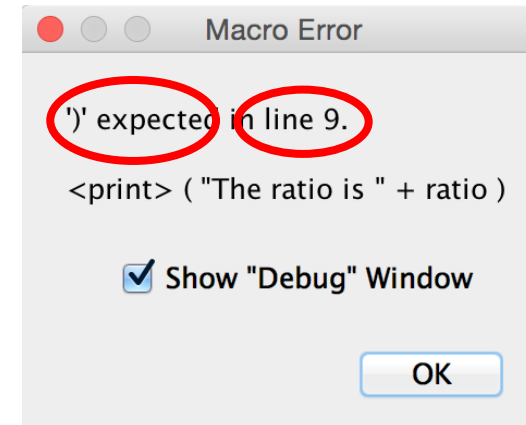
if (a == 5) {
    print("Yin");
} else {
    print("Yang");
}
```

If your program throws error messages:

- Don't panic.
- *"There are two ways to write error-free programs; only the third one works."*

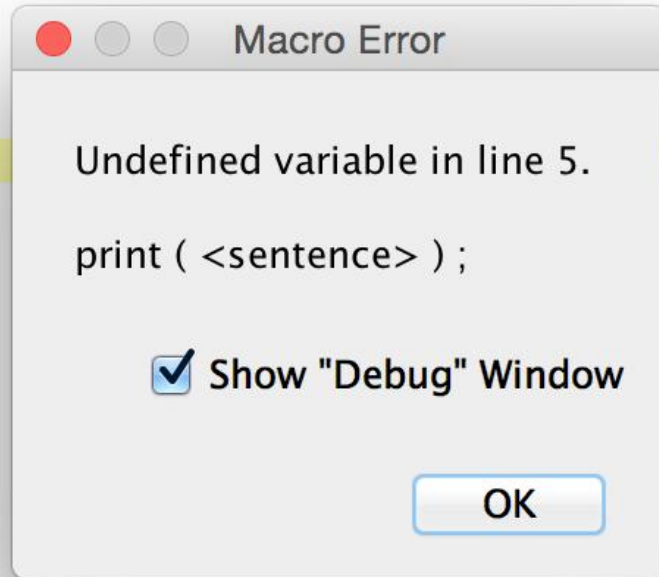
Alan J. Perlis, Yale University

- Read where the error happened. You may see your fault immediately, when looking at the right point.
- Read what appears to be wrong. If you know, what's missing, you may see it, even if it's missing in a slightly different place.
- Don't take error messages too strict. It's just a program reporting about an issue with another program. It may also have issues.

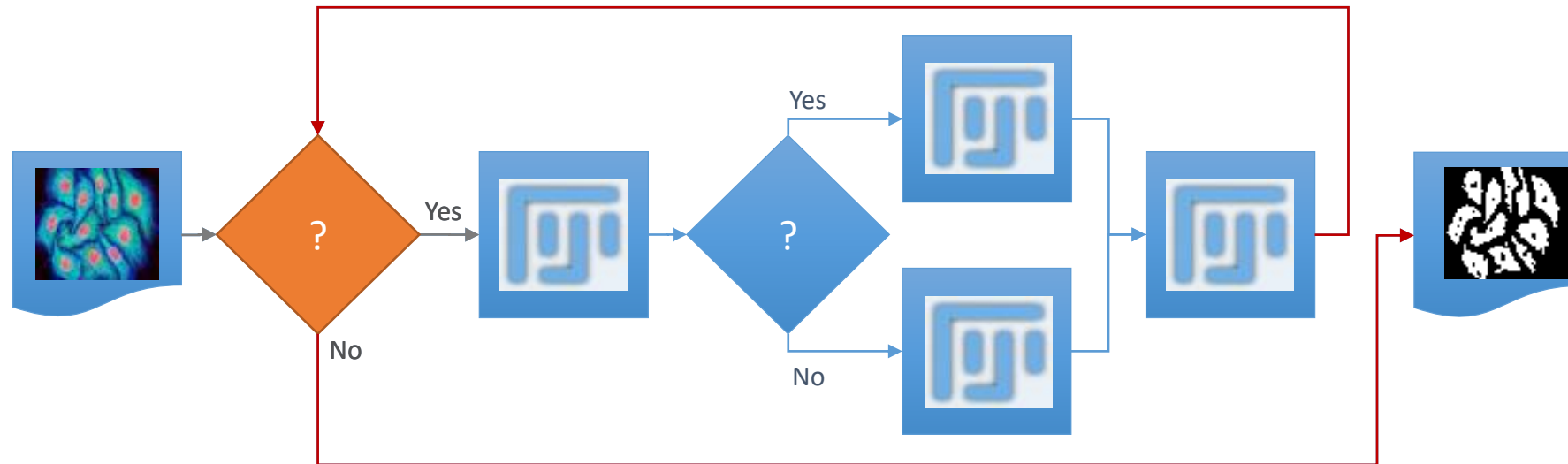


```
1 Sentence = "Can you can a can as a canner can can a can?"
2
3 if (startsWith(Sentence, "Can") {
4     print(sentence);
5 }
6
7
8
9
```

```
1 Sentence = "Can you can a can as a canner can can a can?"  
2  
3 if (startsWith(Sentence, "Can")) {  
4     print(sentence);  
5 }  
6  
7  
8  
9
```



- To repeat actions, you run code in loops



Loop statement

- The `for` statement allows us to execute some lines of code *for* several times

```
// do something
```

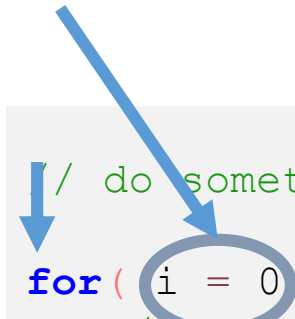
```
for( <initializer>; <condition>; <iterator>) {  
    // do something repeatedly  
}
```

```
// do something
```

Note: this line does **not**
end with a ; semicolon!

- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.

1. start: initialize counter `i`

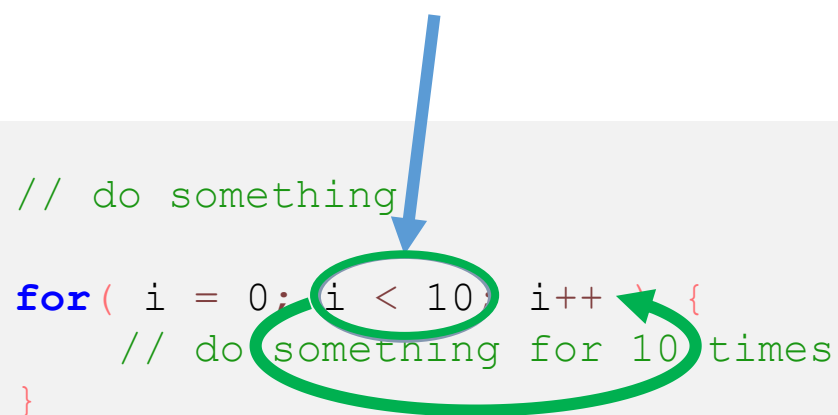


```
// do something
for ( i = 0; i < 10; i++ ) {
    // do something for 10 times
}
```

- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - The condition is checked every time before the whole block is executed.

2. check condition
If true code is executed

```
// do something  
for ( i = 0; i < 10; i++ ) {  
    // do something for 10 times  
}
```



- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - The condition is checked every time before the whole block is executed.
 - After block execution, the counter is increased.

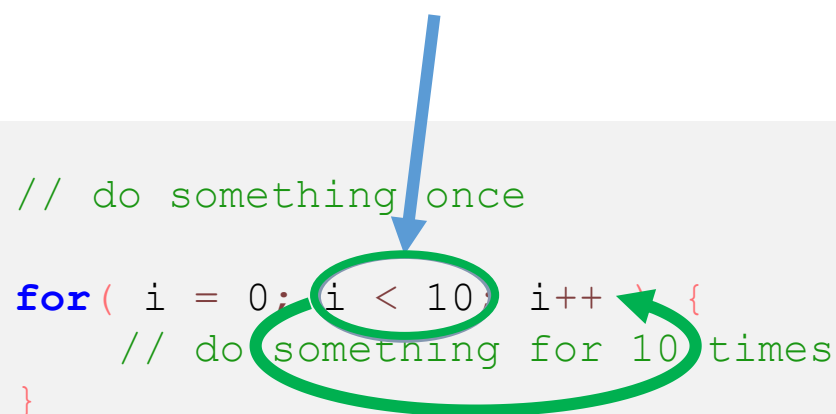
3. increase counter

```
// do something  
for( i = 0; i < 10; i++) {  
    // do something for 10 times  
}
```

- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - **The condition is checked every time before the whole block is executed.**
 - After block execution, the counter is increased.

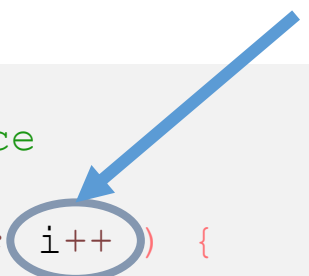
2. check condition
If true: code is executed

```
// do something once  
for ( i = 0; i < 10; i++ ) {  
    // do something for 10 times  
}
```



- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - The condition is checked every time before the whole block is executed.
 - **After block execution, the counter is increased.**

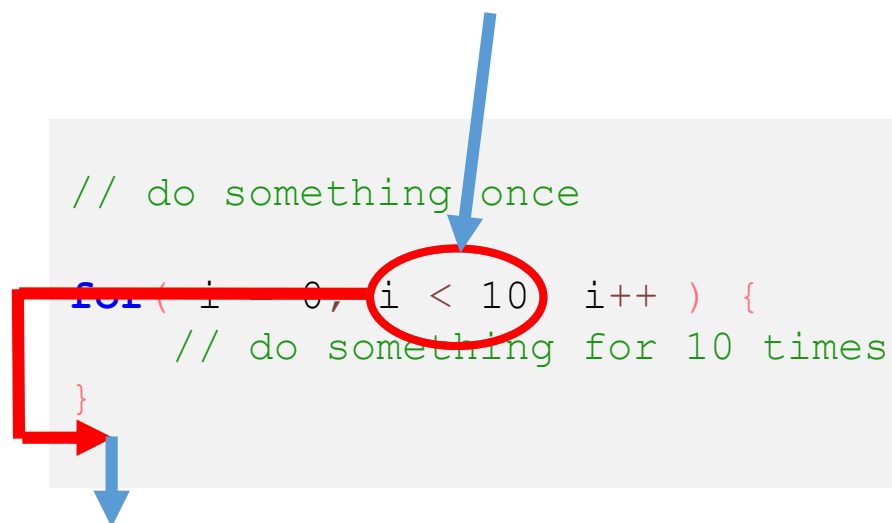
```
// do something once  
for( i = 0; i < 10; i++) {  
    // do something for 10 times  
}
```



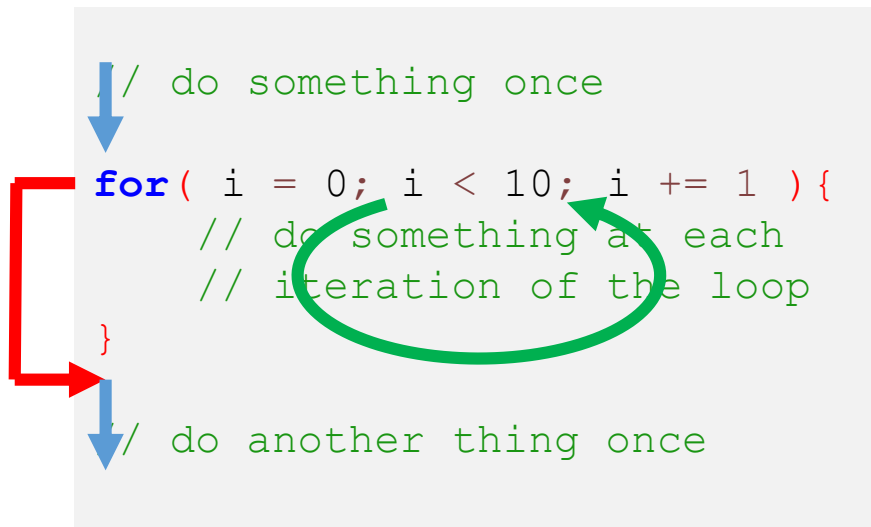
- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - **The condition is checked every time before the whole block is jumped over.**
 - After block execution, the counter is increased.

2. check condition
If false: exit the loop

```
// do something once
for ( i = 0, i < 10 i++ ) {
    // do something for 10 times
}
```



- The `for` statement allows to execute some lines of code *for* several times
 - The initializer is only executed once; at the beginning.
 - The condition is checked every time
 - After block execution, the counter is increased.



```
// do something once
for( i = 0; i < 10; i += 1 ){
    // do something at each
    // iteration of the loop
}
// do another thing once
```

The diagram illustrates the execution flow of a `for` loop. A blue arrow points down to the `for` statement. A red arrow loops from the closing brace of the `for` loop back to the opening brace, indicating the repetition of the loop body. A green arrow points from the `i += 1` increment part of the `for` statement back to the `i < 10` condition part, showing the check and update cycle.

1. The counter is initialized
2. check condition:
 - if true: execution code in the loop
 - if false: exit the loop
3. increase counter

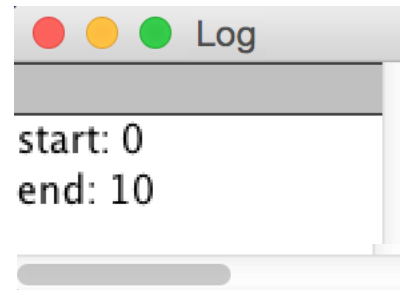
Step 2 and 3 are repeated until the condition is false and the loop exits

- *Tracing* means printing something out after every line of code to ensure the path of execution is gone right.



```
print("start: " + start);  
print("end: " + end);  
  
// print numbers  
for ( i = start; i > end; i += 1 ) {  
    print("is this ever printed?");  
    if (i > 5) {  
        print(i);  
    }  
}
```

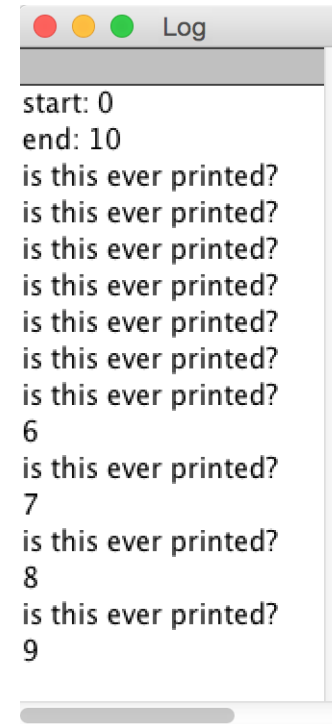
troubleshooting_tracing.ijm



- Tracing means printing something out after every line of code to ensure the path of execution is gone right.

```
print("start: " + start);  
print("end: " + end);  
  
// print numbers  
for ( i = start; i < end; i += 1 ) {  
    print("is this ever printed?");  
    if (i > 5) {  
        print(i);  
    }  
}
```

troubleshooting_tracing.ijm



Log

start: 0
end: 10
is this ever printed?
is this ever printed?
is this ever printed?
is this ever printed?
is this ever printed?
is this ever printed?
is this ever printed?
6
is this ever printed?
7
is this ever printed?
8
is this ever printed?
9

- Don't forget to remove the "traces" after you found the bug!

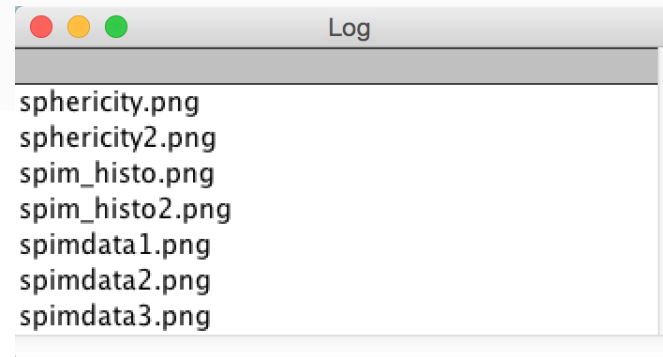
- The files in a folder are represented as an `array[]`

```
// initialise program
foldername = "/Users/rhaase/temp/";

// get all files in the folder as array
list = getFileList( foldername );

// print out the array; item by item
for (i = 0; i < lengthOf(list); i += 1 ) {
    filename = list[i];

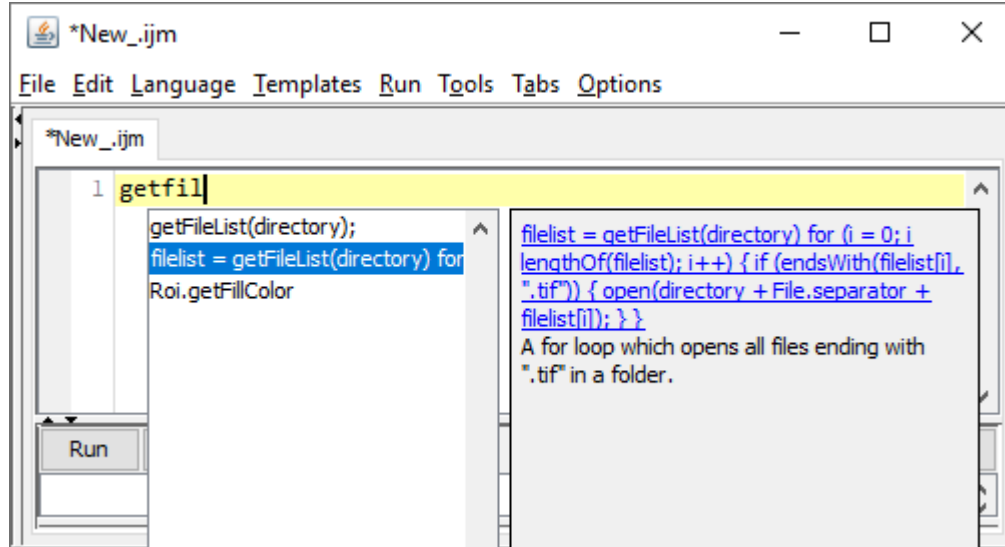
    print( filename );
}
```



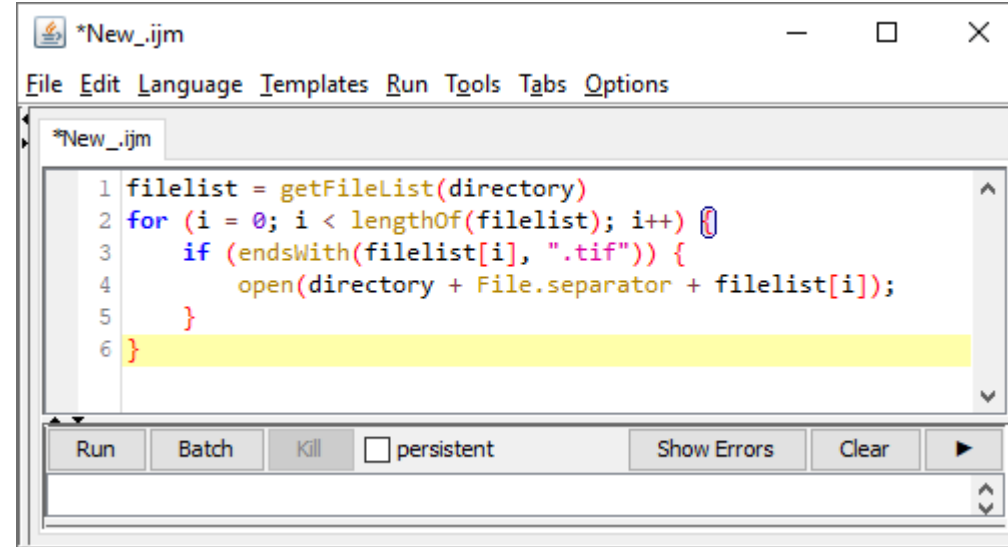
process_folder.ijm

Built-in command	Description	Parameters
filenameList = <code>getFileList</code> (foldername)	return a list of file names	location name of the folder

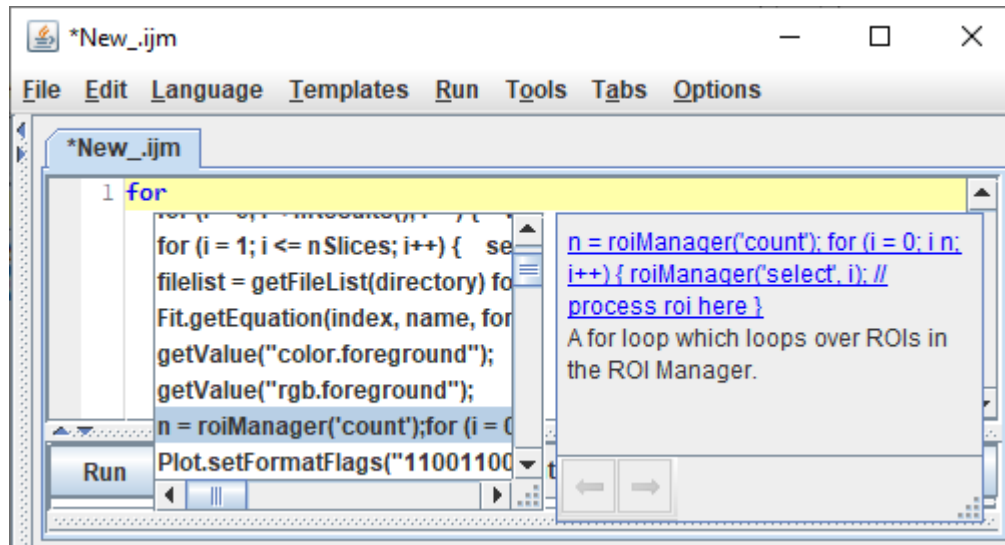
- Use auto-completion!



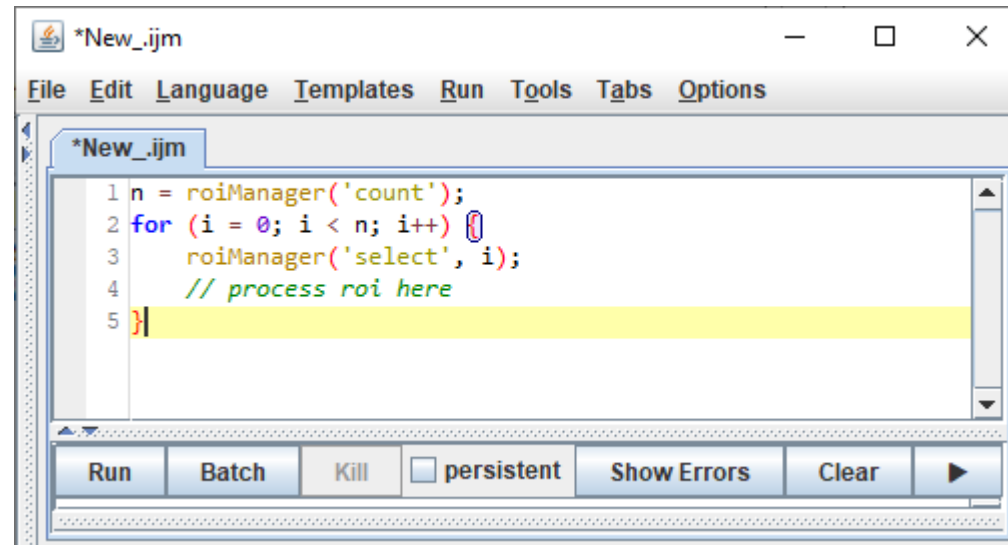
The screenshot shows the IJM editor with the text 'getfil' entered. A dropdown menu displays suggestions: 'getFileList(directory);', 'filelist = getFileList(directory) for', and 'Roi.getFillColor'. The second suggestion is highlighted. A tooltip on the right explains the selected suggestion: 'filelist = getFileList(directory) for (i = 0; i < lengthOf(filelist); i++) { if (endsWith(filelist[i], ".tif")) { open(directory + File.separator + filelist[i]); } }'. It also notes: 'A for loop which opens all files ending with ".tif" in a folder.'



```
1 filelist = getFileList(directory)
2 for (i = 0; i < lengthOf(filelist); i++) {
3     if (endsWith(filelist[i], ".tif")) {
4         open(directory + File.separator + filelist[i]);
5     }
6 }
```



The screenshot shows the IJM editor with the text 'for' entered. A dropdown menu displays suggestions: 'for (i = 1; i <= nSlices; i++) {', 'filelist = getFileList(directory) for', 'Fit.getEquation(index, name, for', 'getValue("color.foreground");', 'getValue("rgb.foreground");', and 'n = roiManager("count"); for (i = 0; i < n; i++) {'. The last suggestion is highlighted. A tooltip on the right explains the selected suggestion: 'n = roiManager("count"); for (i = 0; i < n; i++) { roiManager("select", i); // process roi here }'. It also notes: 'A for loop which loops over ROIs in the ROI Manager.'



```
1 n = roiManager("count");
2 for (i = 0; i < n; i++) {
3     roiManager("select", i);
4     // process roi here
5 }
```


ImageJ macro programming: Arrays, Plots and Tables

Robert Haase

October 2021

- Variables are memory blocks where you can store stuff

```
2 | measurement = 5;
```

Computer memory

measurement

5

name

"Drosophila"

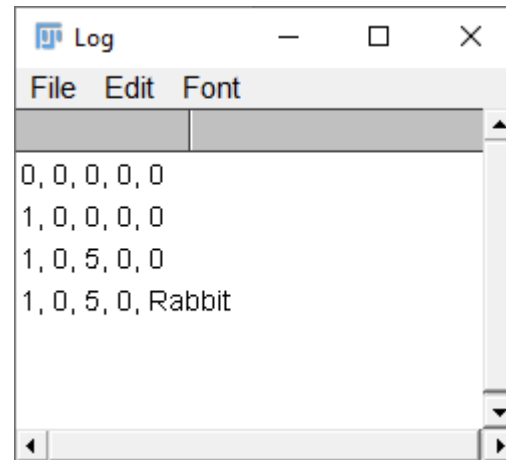
combination

"Drosophila5"

- Arrays are variables, where you can store multiple stuff

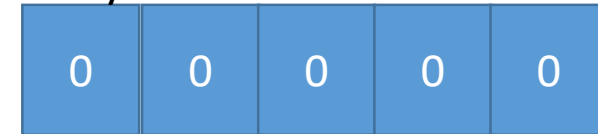
```
1  
2 array = newArray(5);  
3 Array.print(array);
```

arrays_access.ijm

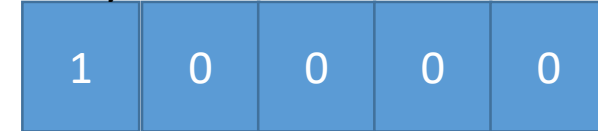


Computer memory

array



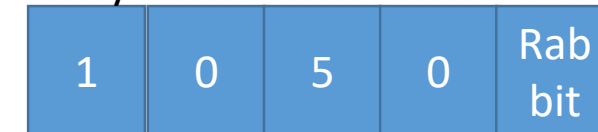
array



array

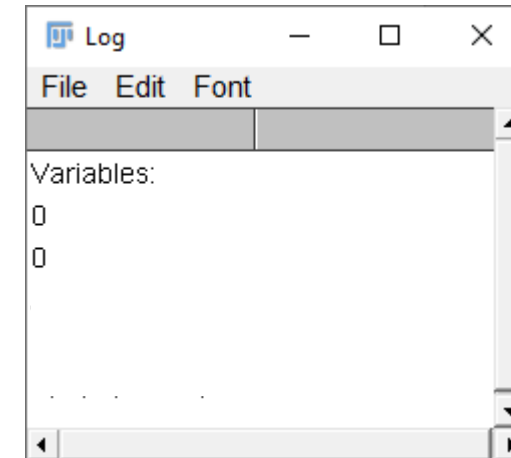


array



- Variable assignment: by value
- Array assignment: by reference

```
1 // working with variables
2 print("Variables:");
3 a = 0;
4 print(a);
5
6 b = a;
7 b = 5;
8 print(a);
```



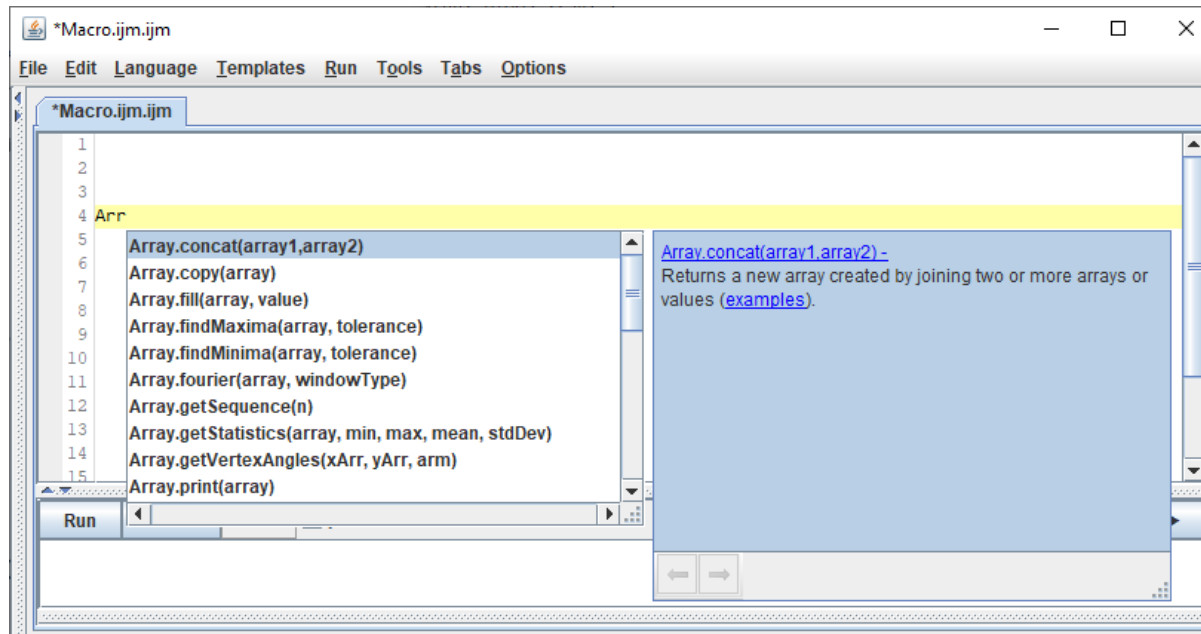
arrays_variable_comparion.ijm

- Arrays can be initialized with values (more than 1)

```
v = newArray(3, -4, 0);  
  
animals = newArray("dog", "cat", "mouse");
```

- Arrays can be concatenated:

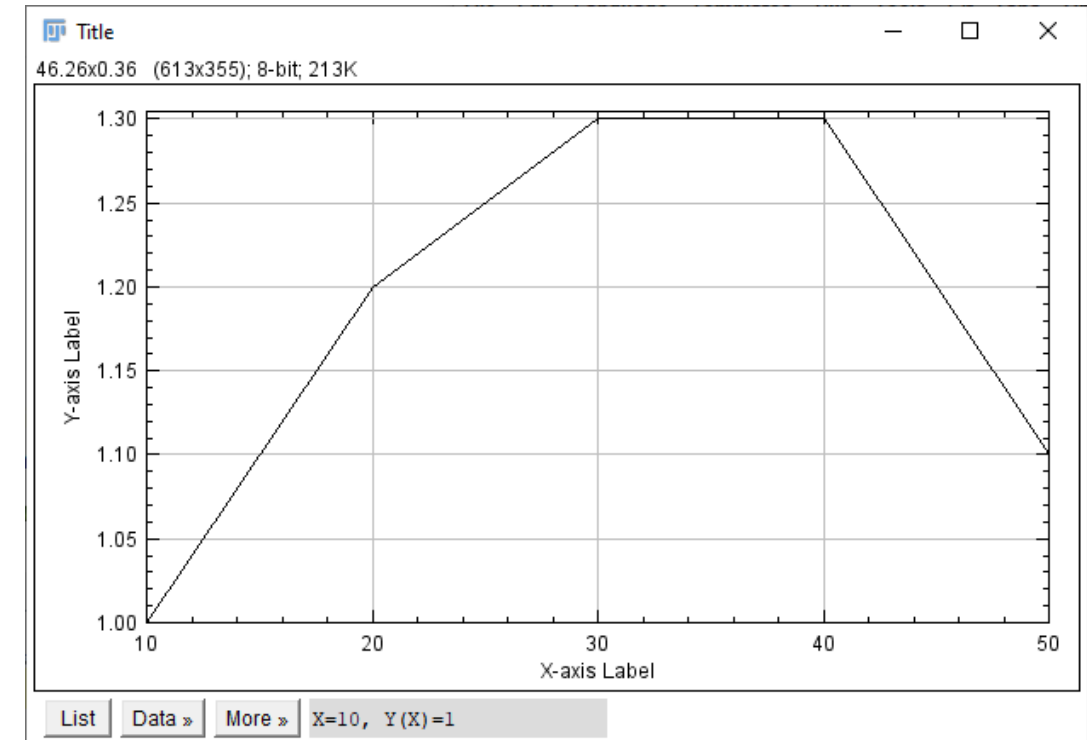
```
mixed = Array.concat(v, animals);
```
- Find out what else you can do with Arrays by starting typing:



- Simple plots can be drawn using arrays

```
array_example.ijm (Running)
File Edit Language Templates Run Tools Git Tabs Options
array_example.ijm
1 // Define two arrays with 5 elements
2 x_values = newArray(5);
3 y_values = newArray(5);
4
5 // put some numbers in these 2x5 array elements
6 x_values[0] = 10;
7 y_values[0] = 1;
8
9 x_values[1] = 20;
10 y_values[1] = 1.2;
11
12 x_values[2] = 30;
13 y_values[2] = 1.3;
14
15 x_values[3] = 40;
16 y_values[3] = 1.3;
17
18 x_values[4] = 50;
19 y_values[4] = 1.1;
20
21 Plot.create("Title", "X-axis Label", "Y-axis Label", x_values, y_values);
22 Plot.show();
23

Run Batch ☐ persistent Show Errors Clear ▶
Started New_.ijm at Wed Apr 15 18:50:02 CEST 2020
Started ijmmnd_example.ijm at Wed Apr 15 18:57:02 CEST 2020
```





Break: 15 min

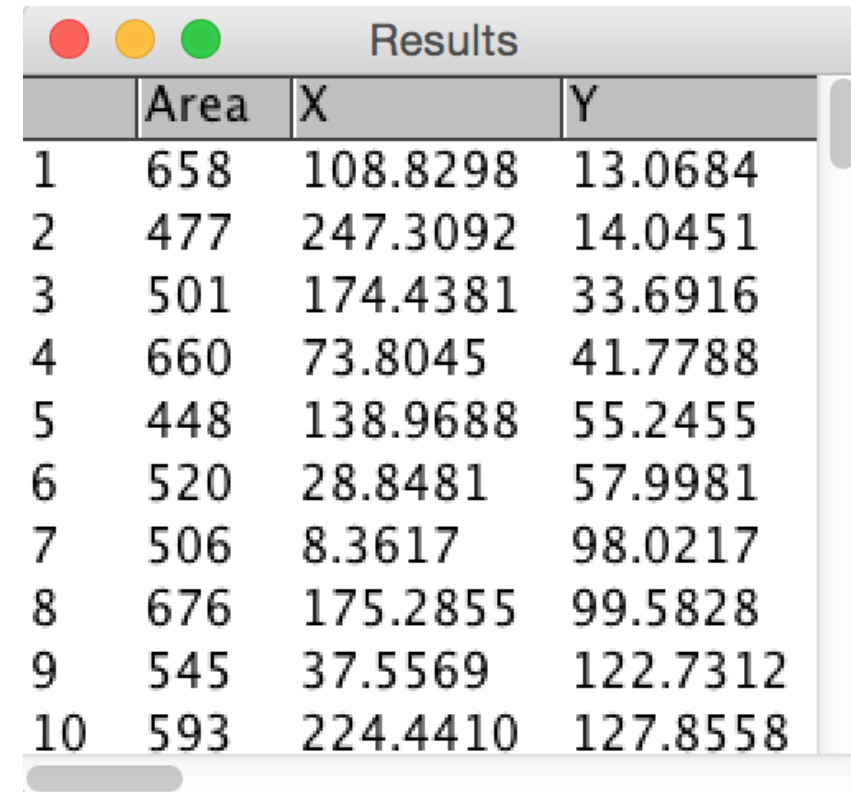
Working with tables

Robert Haase

With materials from
Benoit Lombardot, Scientific Computing Facility, MPI CBG

October 2021

- ImageJ/FIJI tool for collecting data/measurements
- programmable as well
 - Create tables with measurements
 - Read cells
 - Write cells
 - Add columns
 - Save as CSV or XLS file

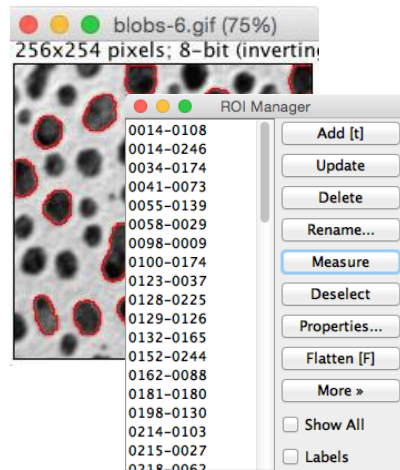


	Area	X	Y
1	658	108.8298	13.0684
2	477	247.3092	14.0451
3	501	174.4381	33.6916
4	660	73.8045	41.7788
5	448	138.9688	55.2455
6	520	28.8481	57.9981
7	506	8.3617	98.0217
8	676	175.2855	99.5828
9	545	37.5569	122.7312
10	593	224.4410	127.8558

- Two ways for creating result tables



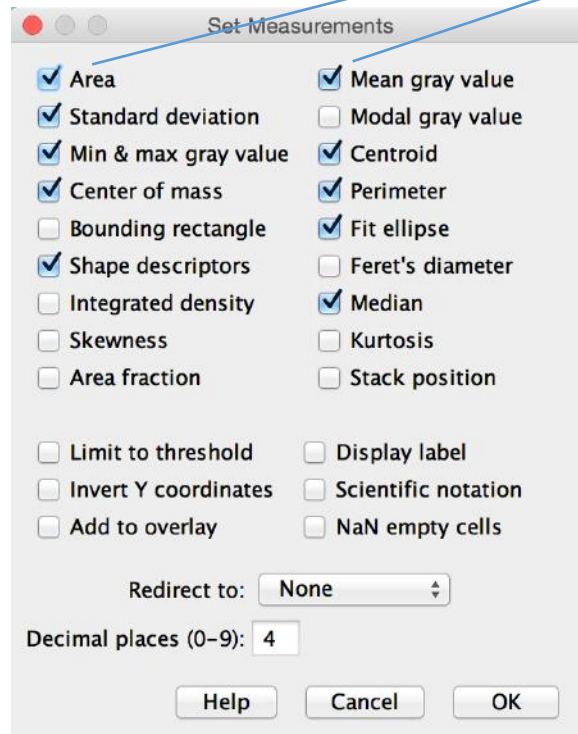
```
run("Analyze Particles...", "display");
```



```
roiManager("Measure");
```

Results			
	Area	X	Y
1	658	108.8298	13.0684
2	477	247.3092	14.0451
3	501	174.4381	33.6916
4	660	73.8045	41.7788
5	448	138.9688	55.2455
6	520	28.8481	57.9981
7	506	8.3617	98.0217
8	676	175.2855	99.5828
9	545	37.5569	122.7312
10	593	224.4410	127.8558

- Configure what to measure...
- Menu *Analyze > Set measurements...*



```
run("Set Measurements...",  
"area mean standard min centroid center  
perimeter fit shape median redirect=None  
decimal=4");
```

... by entering the first word
behind the checkbox.

- Reading elements in tables

	Area	X	Y
1	658	108.8298	13.0684
2	477	247.3092	14.0451
3	501	174.4381	33.6916
4	660	73.8045	41.7788
5	448	138.9688	55.2455
6	520	28.8481	57.9981
7	506	8.3617	98.0217
8	676	175.2855	99.5828
9	545	37.5569	122.7312
10	593	224.4410	127.8558

```
area = getResult("Area", 0);  
print(area);
```

Log
658

```
r = nResults() - 1; // last row  
area = getResult("Area", r);  
print(area);
```

Log
593

```
r = nResults();  
print(r); // number of rows
```

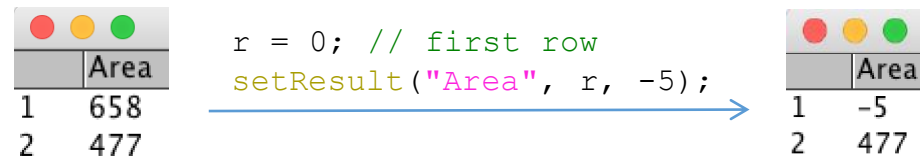
Log
10

Built-in command

```
value = getResult("column title", rowNum);
```

```
rowCount = nResults();
```

- Manipulating (and adding) elements in a table



Handle with care!

Built-in command

```
setResult ("column title", rowNum, newValue);
```

- There is an alternative to write simple tables and text files to disc

```
1 path = "C:/structure/teaching/lecture_applied_bioimage_analysis/06_example_code/test.csv";
2
3 headline = "Number, number squared";
4
5 File.append(headline, path);
6
7 for (i = 0; i < 10; i++) {
8     contentline = "" + i + ", " + pow(i, 2);
9     File.append(contentline, path);
10 }
```

save_csv_file.ijm

test.csv ×	
1	Number, number squared
2	0, 0
3	1, 1
4	2, 4
5	3, 9
6	4, 16
7	5, 25
8	6, 36
9	7, 49
10	8, 64
11	9, 81

That might be useful in
the exercise tomorrow.

ImageJ macro programming: ROI and Overlays

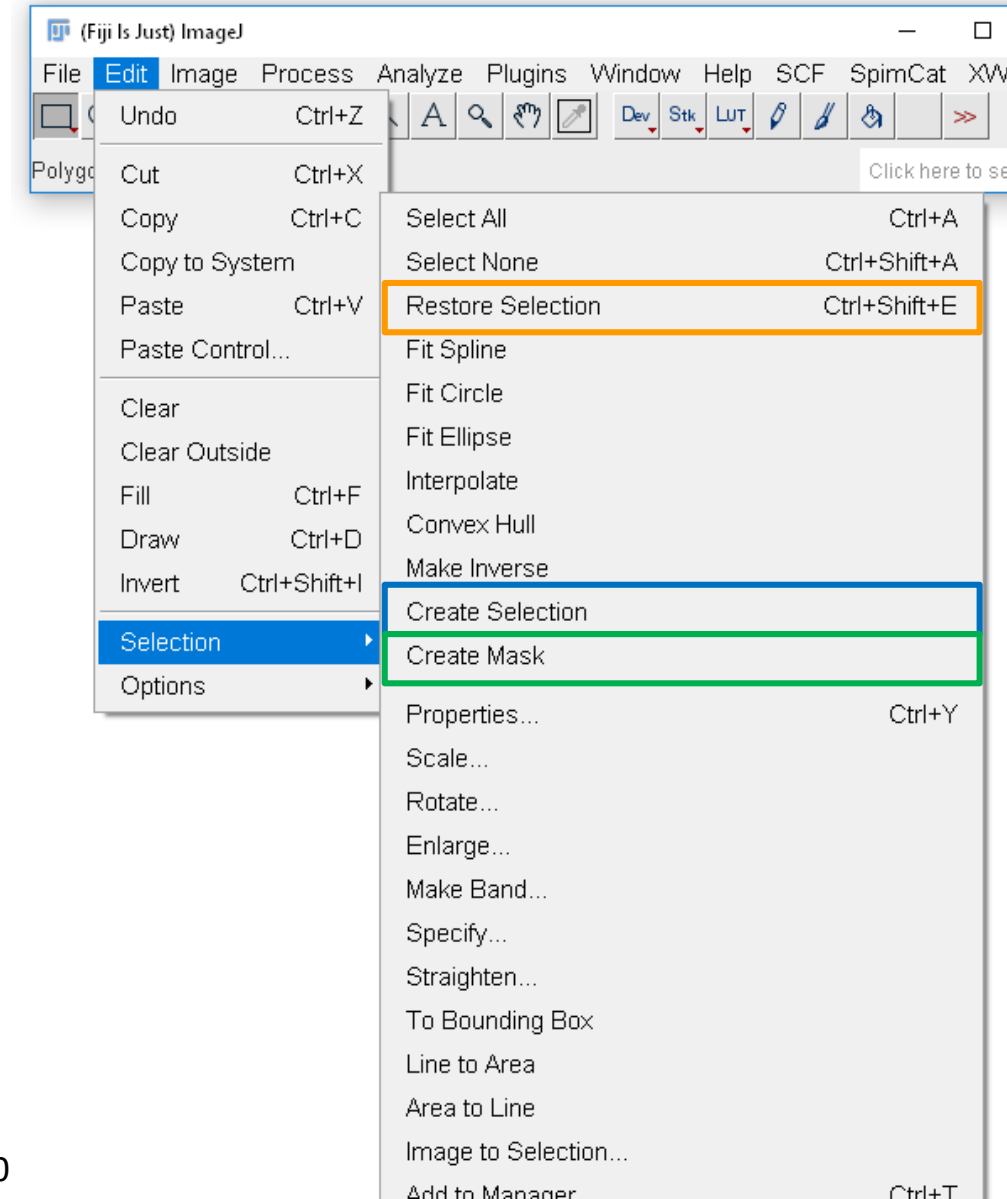
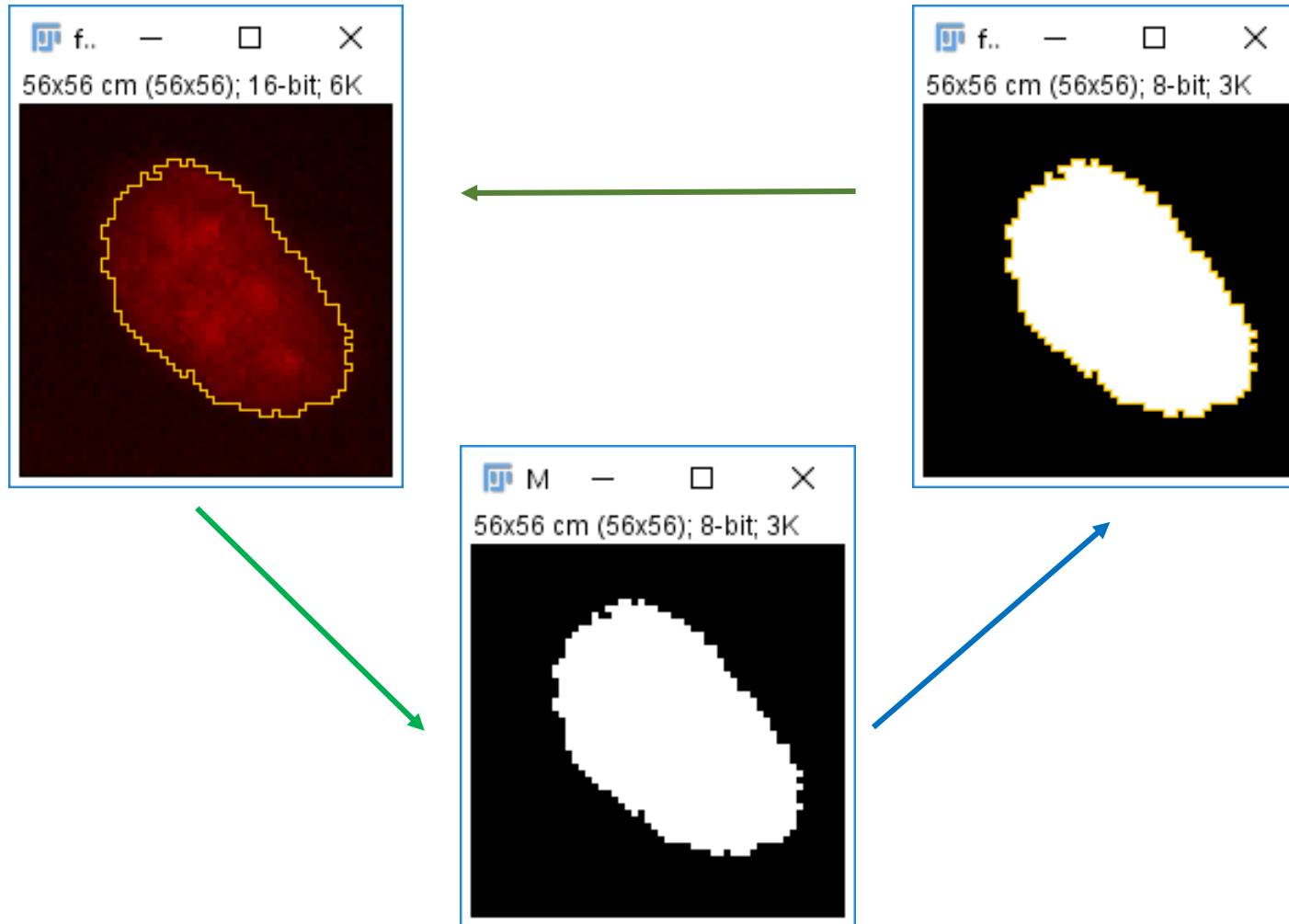
Robert Haase

With material from

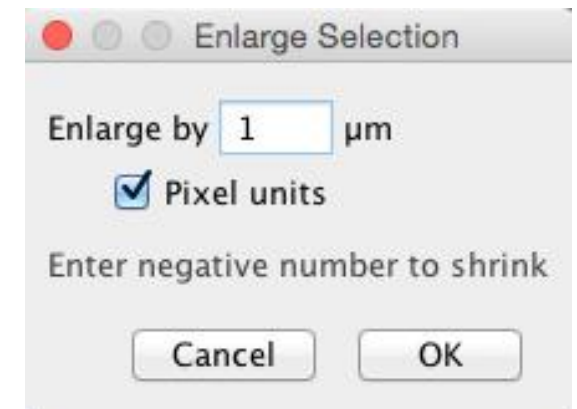
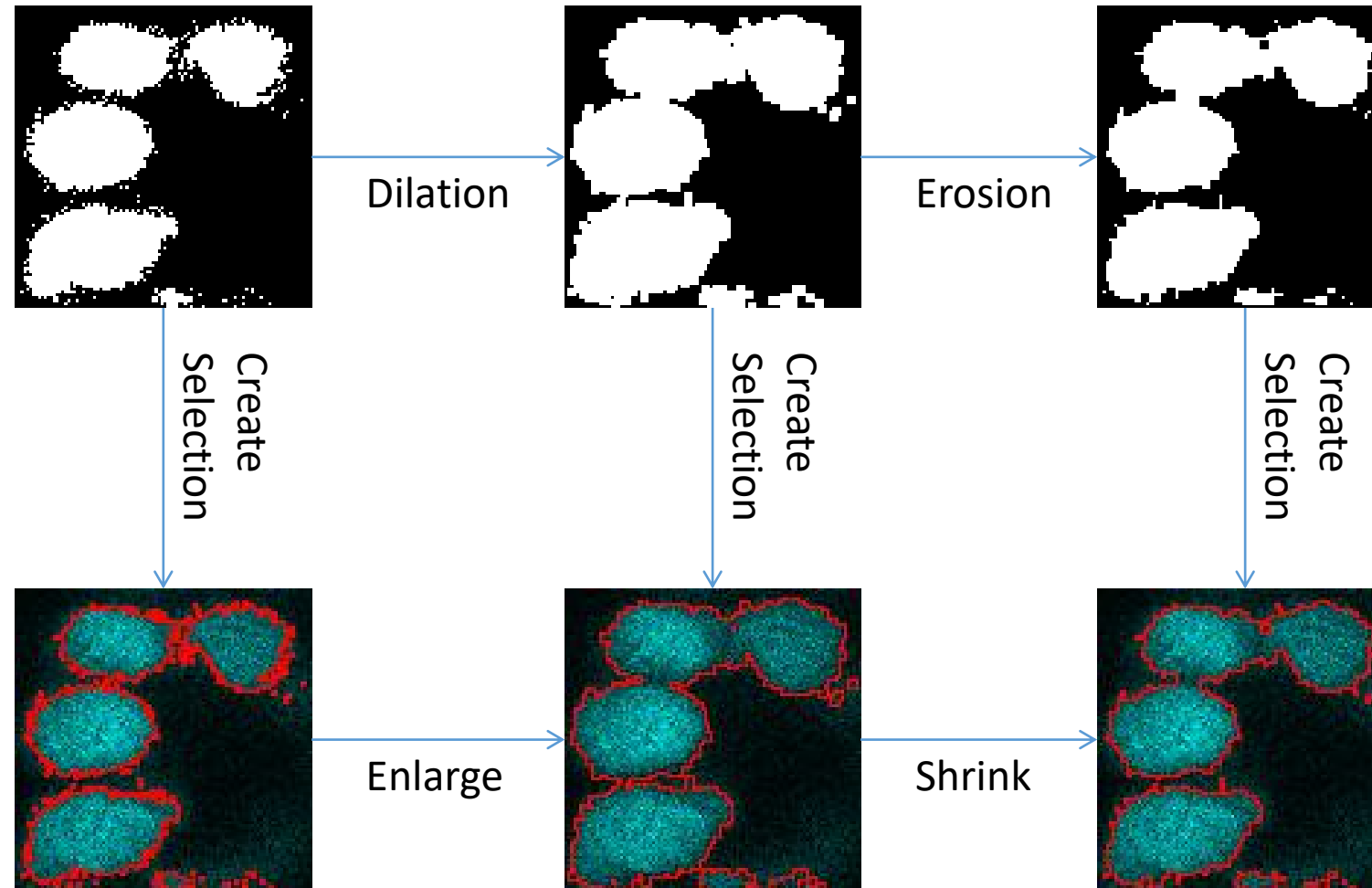
Benoit Lombardot, Scientific Computing Facility, MPI CBG

Virtually at CCI Gothenburg, October 2021

- Selections and masks are exchangeable.



- Some operations for selections you know already from masks.

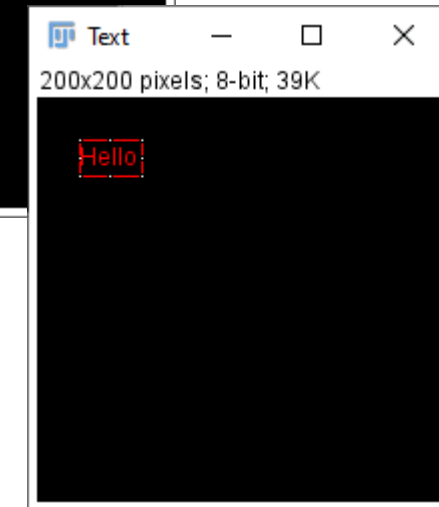
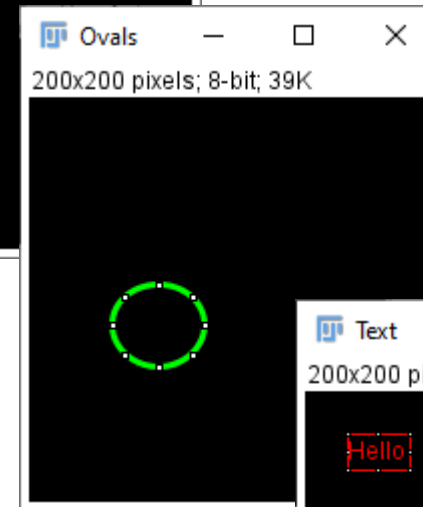
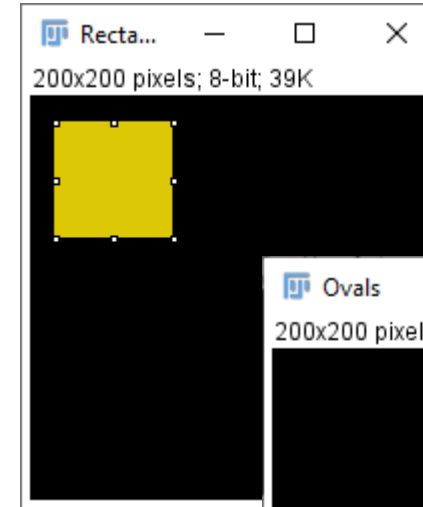


- Hint: Use the macro recorder!

[illegible]

- ROIs can be used for visualization of (intermediate) results

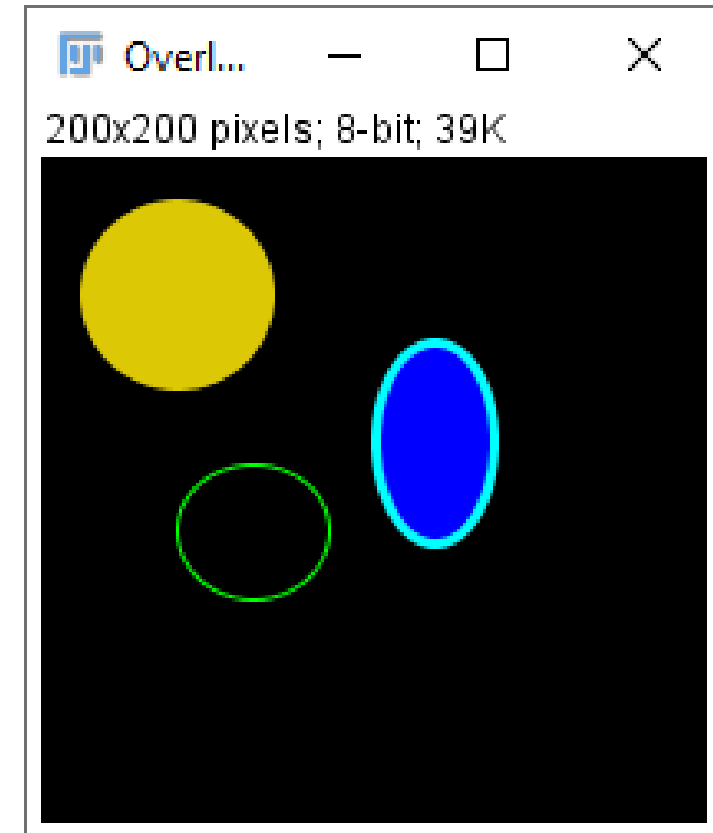
```
3 // Draw rectangles
4 newImage("Rectangles", "8-bit black", 200, 200, 1);
5 makeRectangle(11, 12, 59, 58);
6 Roi.setFillColor(220, 200, 4);
```



- You can collect ROIs in the “Overlay”

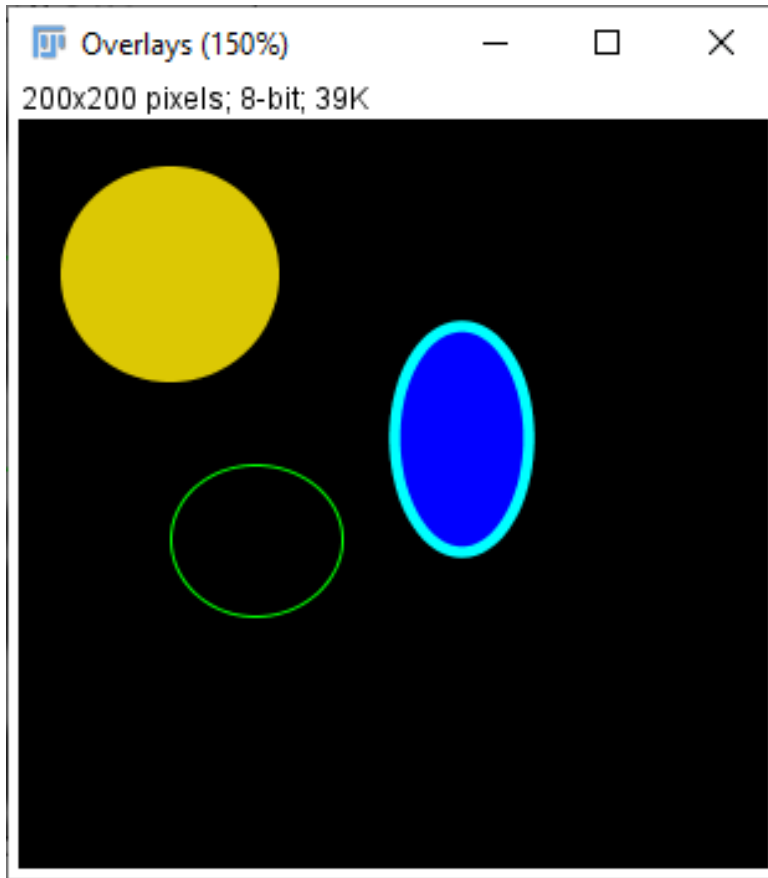
```
3 // Draw filled ellipses
4 newImage("Overlays", "8-bit black", 200, 200);
5 makeOval(11, 12, 59, 58);
6 Roi.setFillColor(220, 200, 4);
7 Overlay.addSelection();
8
9 // Draw outline ellipses
10 makeOval(40, 92, 46, 41);
11 Roi.setStrokeColor("Green");
12 Overlay.addSelection();
13
14 // Draw filled ellipses, with outline
15 makeOval(100, 55, 36, 61);
16 Overlay.addSelection("", 0, "blue");
17 makeOval(100, 55, 36, 61);
18 Overlay.addSelection("Cyan", 3);
19
```

overlays.ijm

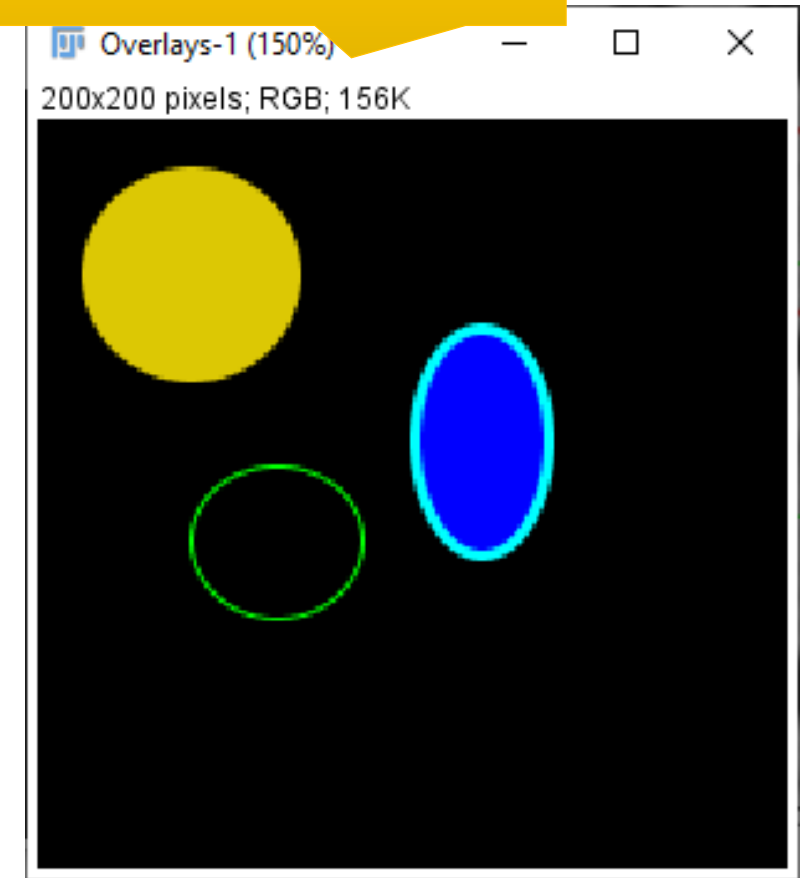


- Overlays can be flattened – for visualization purposes

This is no longer
an 8-bit image!



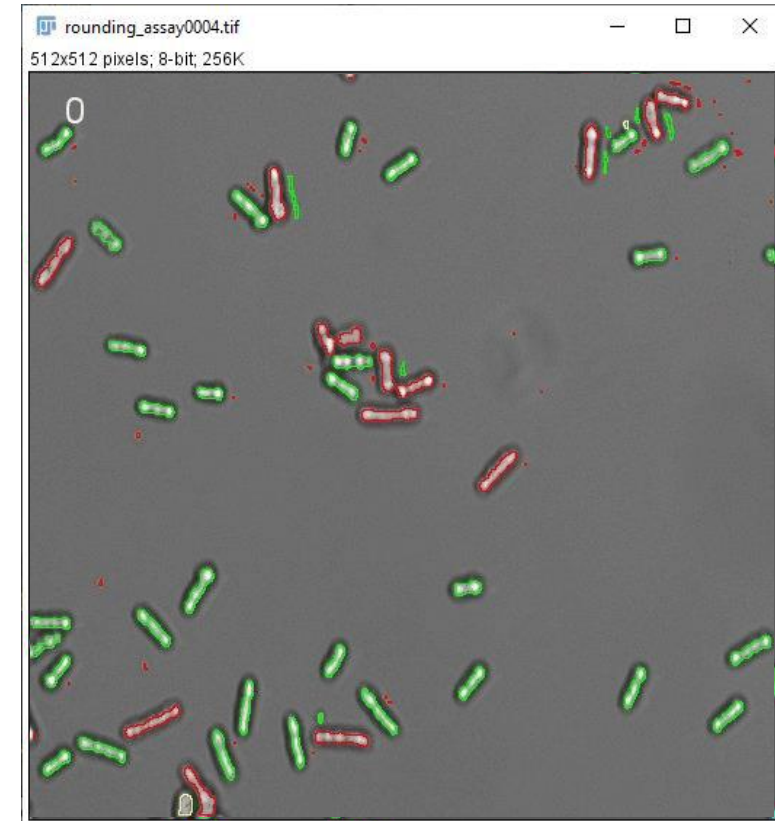
```
// burn in the overlay  
// in the image  
run("Flatten");
```



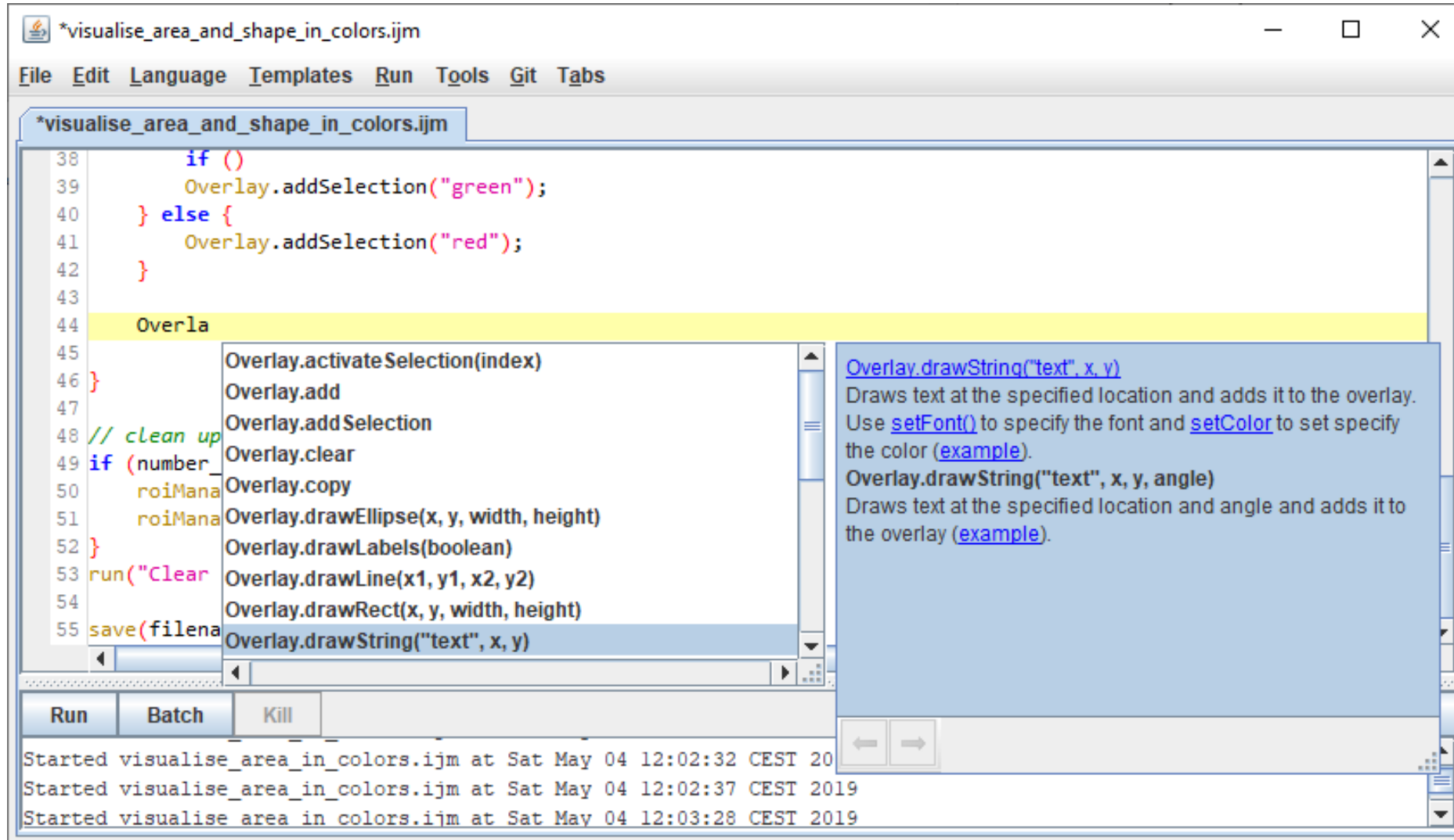
- Custom visualization with a for loop

```
17 // connected components analysis -> send results to ROI Manager
18 run("Analyze Particles...", " show=Nothing add");
19
20 // switch back to original image
21 selectWindow(original_image_title);
22 roiManager("Show None");
23
24 number_of_regions = roiManager("count");
25 for (i = 0; i < number_of_regions; i++) {
26
27     // measure area ( = pixel count)
28     run("Set Measurements...", "area redirect=None decimal=3");
29     roiManager("Select", i);
30     roiManager("Measure");
31     pixel_count = getResult("Area", nResults - 1);
32
33     // visualise if ROIs are too small or not
34     if (pixel_count > 10) {
35         Overlay.addSelection("green");
36     } else {
37         Overlay.addSelection("red");
38     }
39 }
```

visualise_area_in_colors.ijm



- Explore what you can do with overlays by starting to type...



The screenshot shows the Fiji software interface with the file `*visualise_area_and_shape_in_colors.ijm` open. The code editor displays the following code:

```
38     if ()
39         Overlay.addSelection("green");
40     } else {
41         Overlay.addSelection("red");
42     }
43
44     Overla
45
46 }
47
48 // clean up
49 if (number_
50     roiMana
51     roiMana
52 }
53 run("Clear
54
55 save(filena
```

A dropdown menu is visible below the word "Overla", listing the following methods:

- Overlay.activateSelection(index)
- Overlay.add
- Overlay.addSelection
- Overlay.clear
- Overlay.copy
- Overlay.drawEllipse(x, y, width, height)
- Overlay.drawLabels(boolean)
- Overlay.drawLine(x1, y1, x2, y2)
- Overlay.drawRect(x, y, width, height)
- Overlay.drawString("text", x, y)

On the right side of the interface, a tooltip provides detailed documentation for the `Overlay.drawString` methods:

- `Overlay.drawString("text", x, y)`
Draws text at the specified location and adds it to the overlay. Use `setFont()` to specify the font and `setColor` to set specify the color ([example](#)).
- `Overlay.drawString("text", x, y, angle)`
Draws text at the specified location and angle and adds it to the overlay ([example](#)).

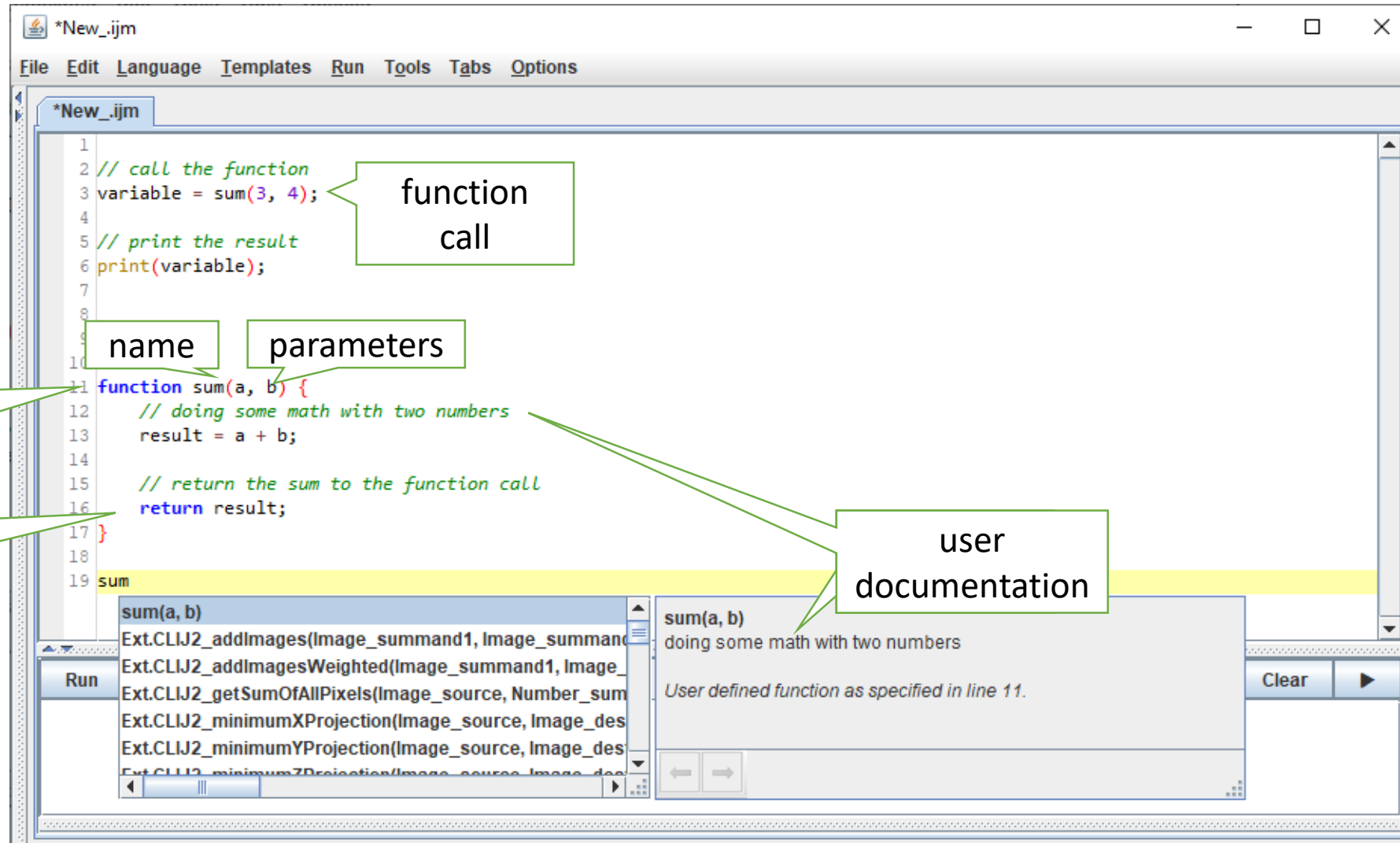
At the bottom of the window, the Run, Batch, and Kill buttons are visible, along with a log showing the execution history of the script.

Custom Macro Functions

Robert Haase

October 2021

- For reusing code and for organizing code: Use functions!



```
1
2 // call the function
3 variable = sum(3, 4);
4
5 // print the result
6 print(variable);
7
8
9
10
11 function sum(a, b) {
12     // doing some math with two numbers
13     result = a + b;
14
15     // return the sum to the function call
16     return result;
17 }
18
19 sum
```

function call

function definition

name

parameters

return value

user documentation

sum(a, b)
doing some math with two numbers
User defined function as specified in line 11.



Exercises

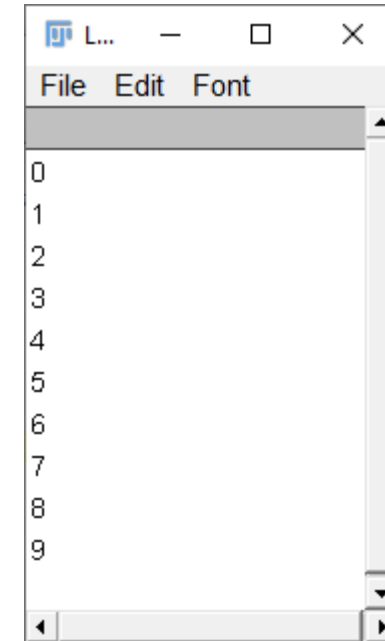
Virtually at CCI Gothenburg, September 2020

```
// initialisation
loop_count = 10;

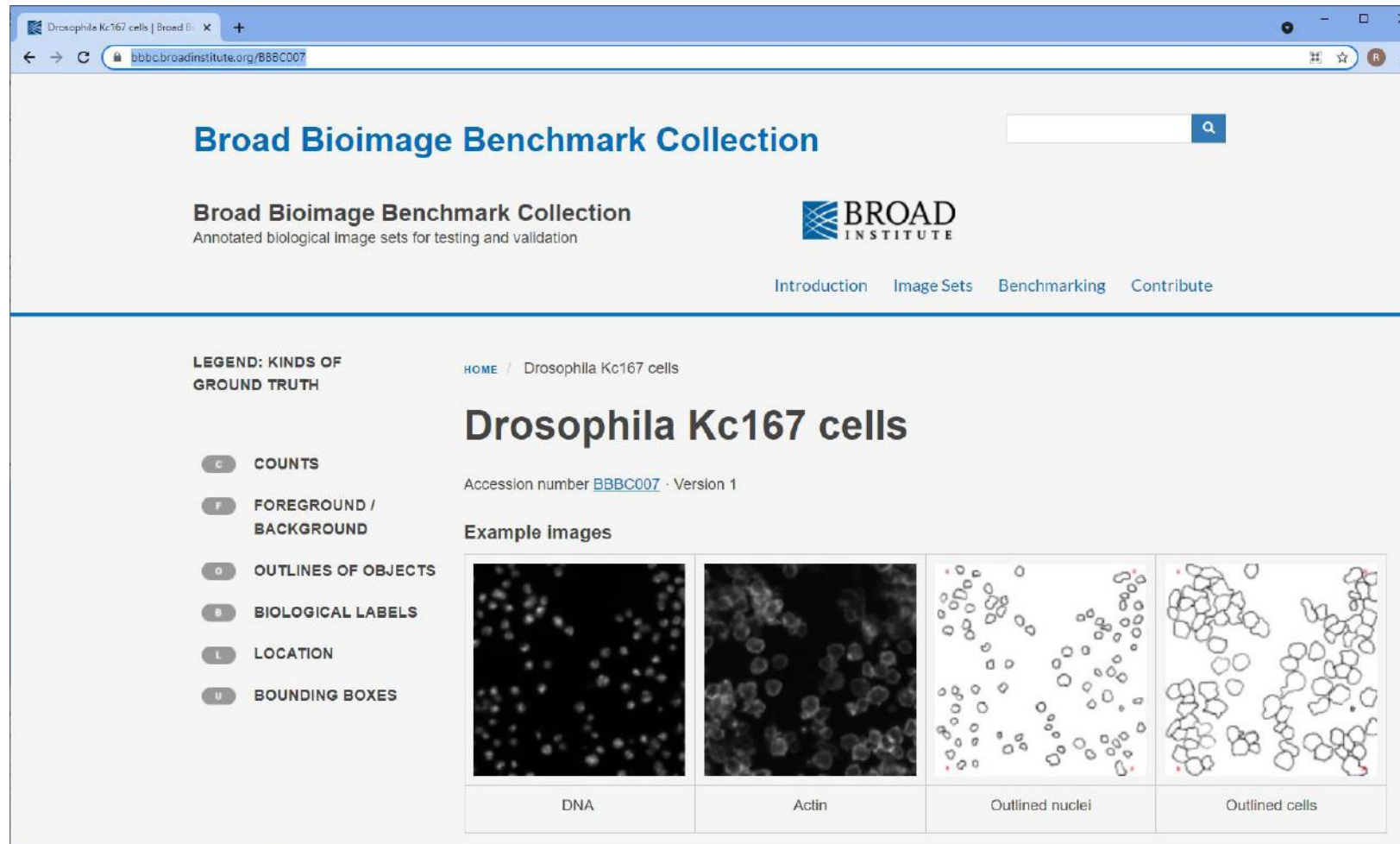
// print 10 numbers
for( i = 0; i < loop_count; i++ ){
    print(i);
}
```

exercise_for_loop.ijm

- Use this example code to program a `for` loop which counts backwards from a given number to 0.

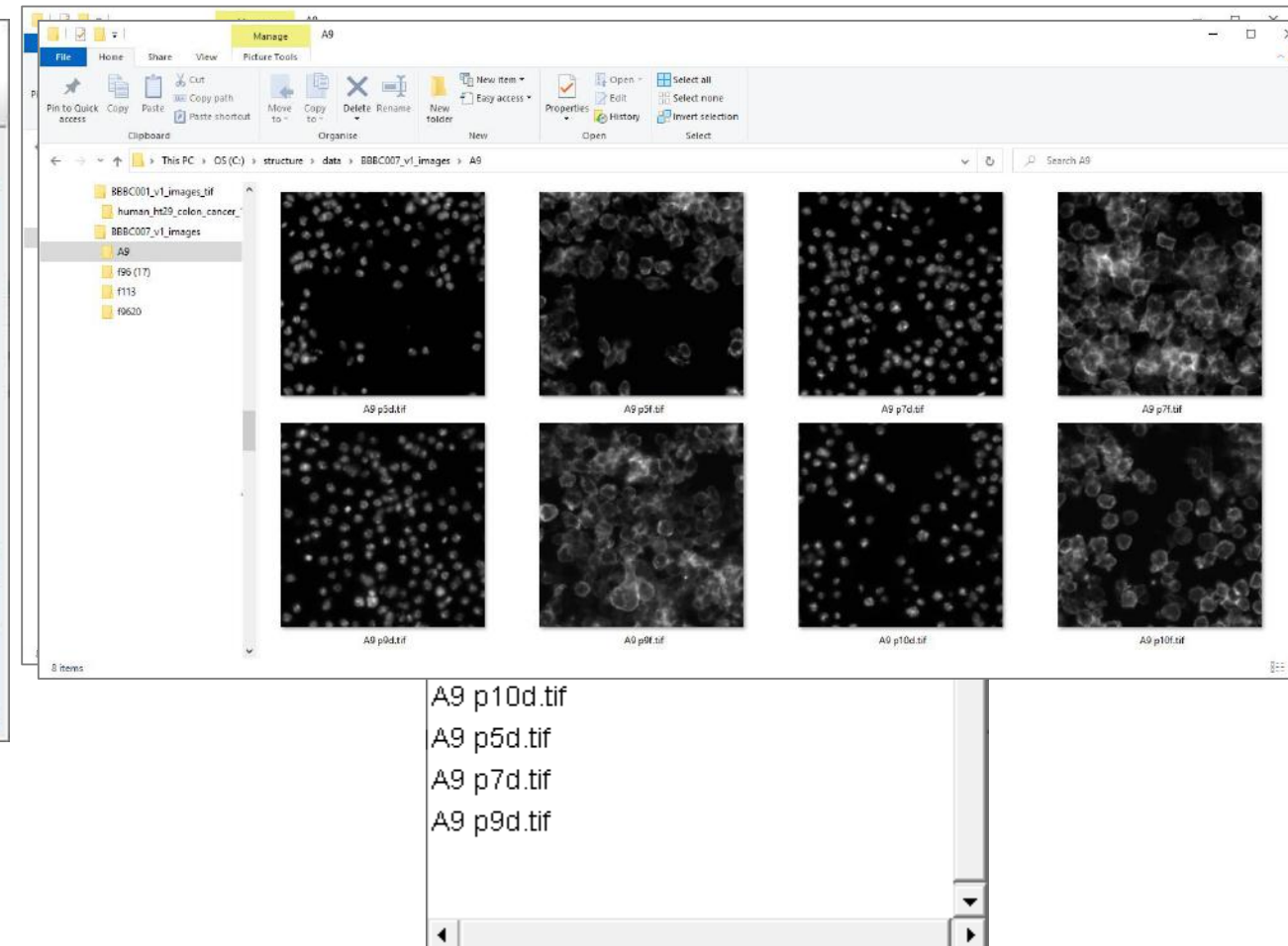
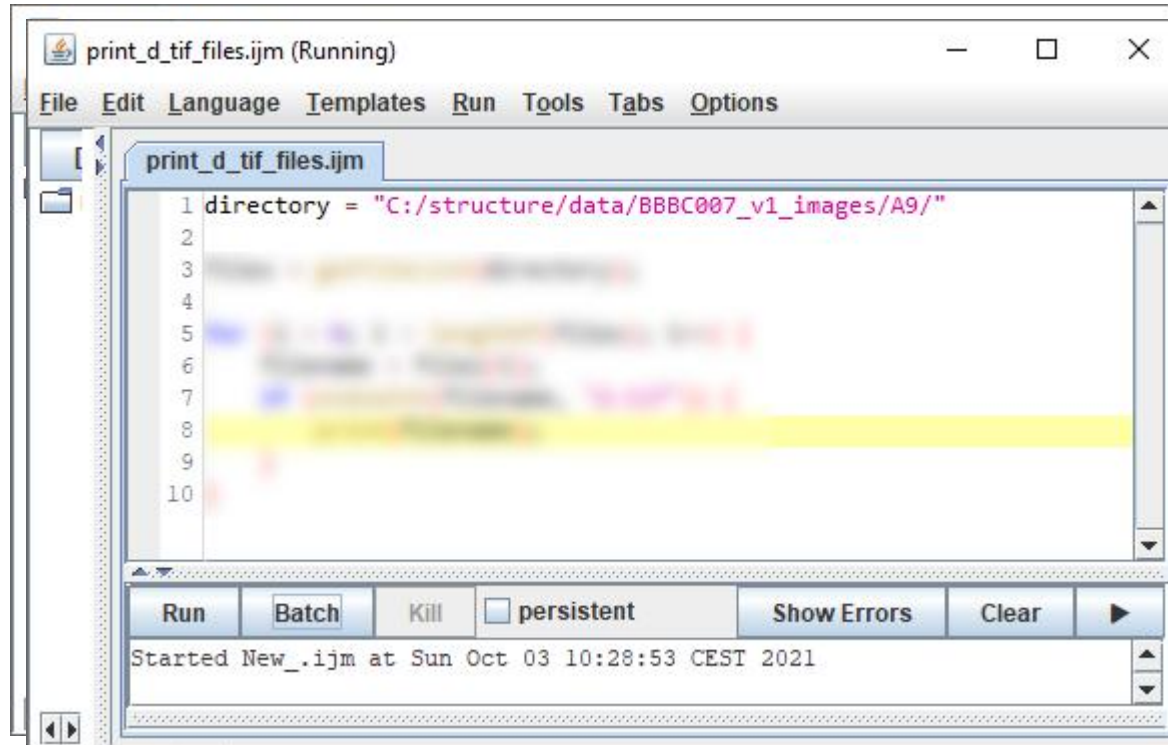


- Visit the Broad Bioimage Benchmark Collection Website and download BBBC007
<https://bbbc.broadinstitute.org/BBBC007>



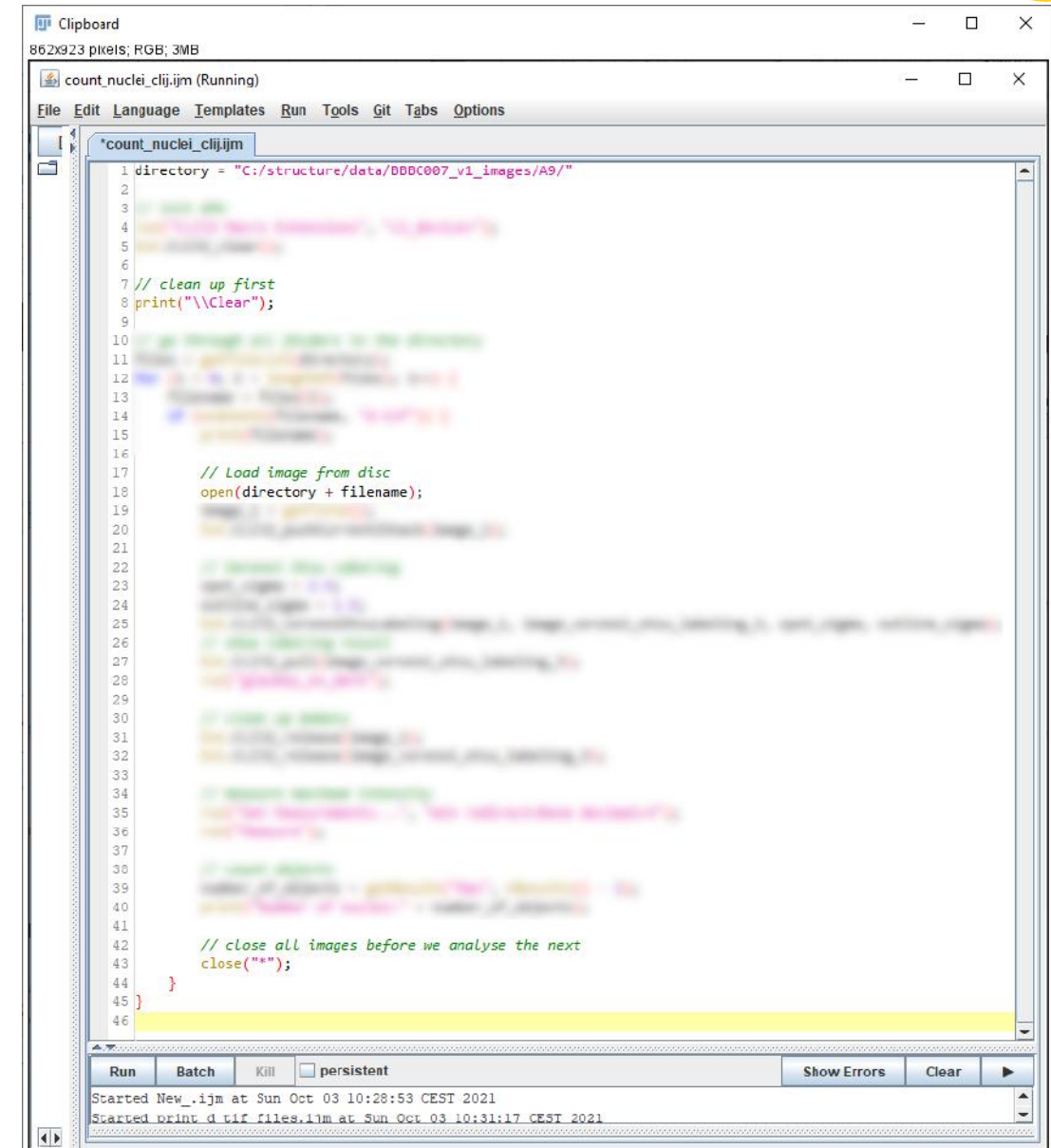
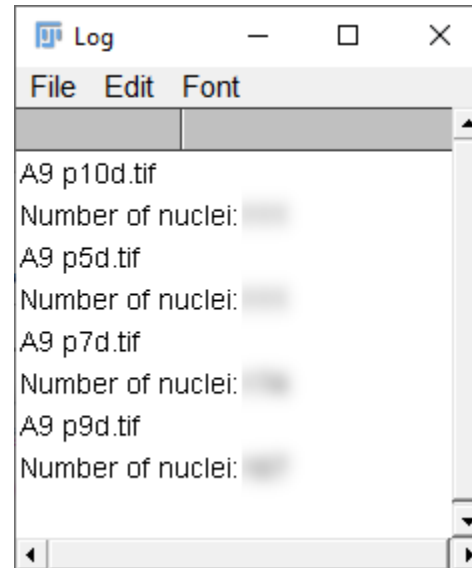
Exercise: Counting nuclei

- Write a for-loop that goes through the A9 folder of BBBC007 and prints out all filenames ending with "d.tif"

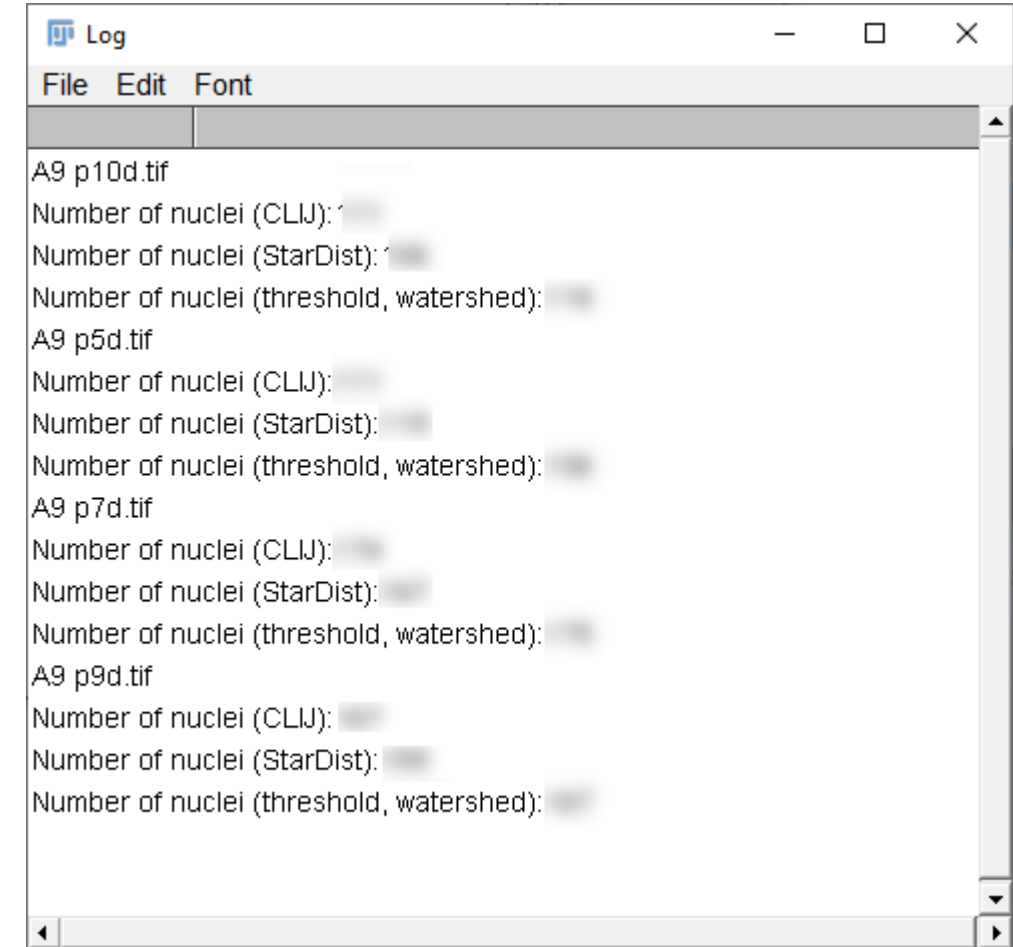
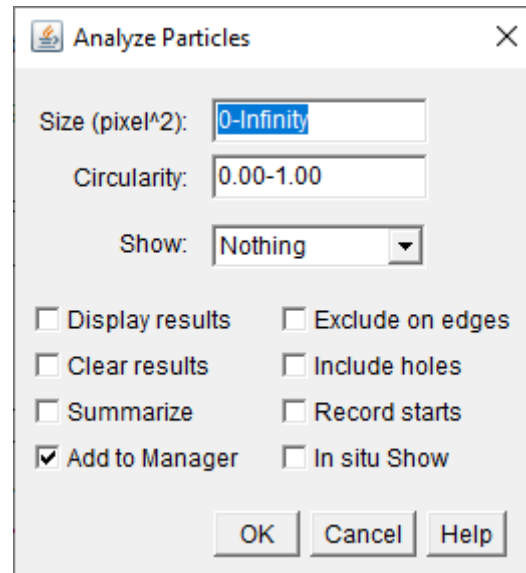


Exercise: Counting nuclei

- Extend your script so that it counts nuclei in these images using one method of your choice.
- Hint: The number of objects in a label-image equals the maximum intensity of that image.

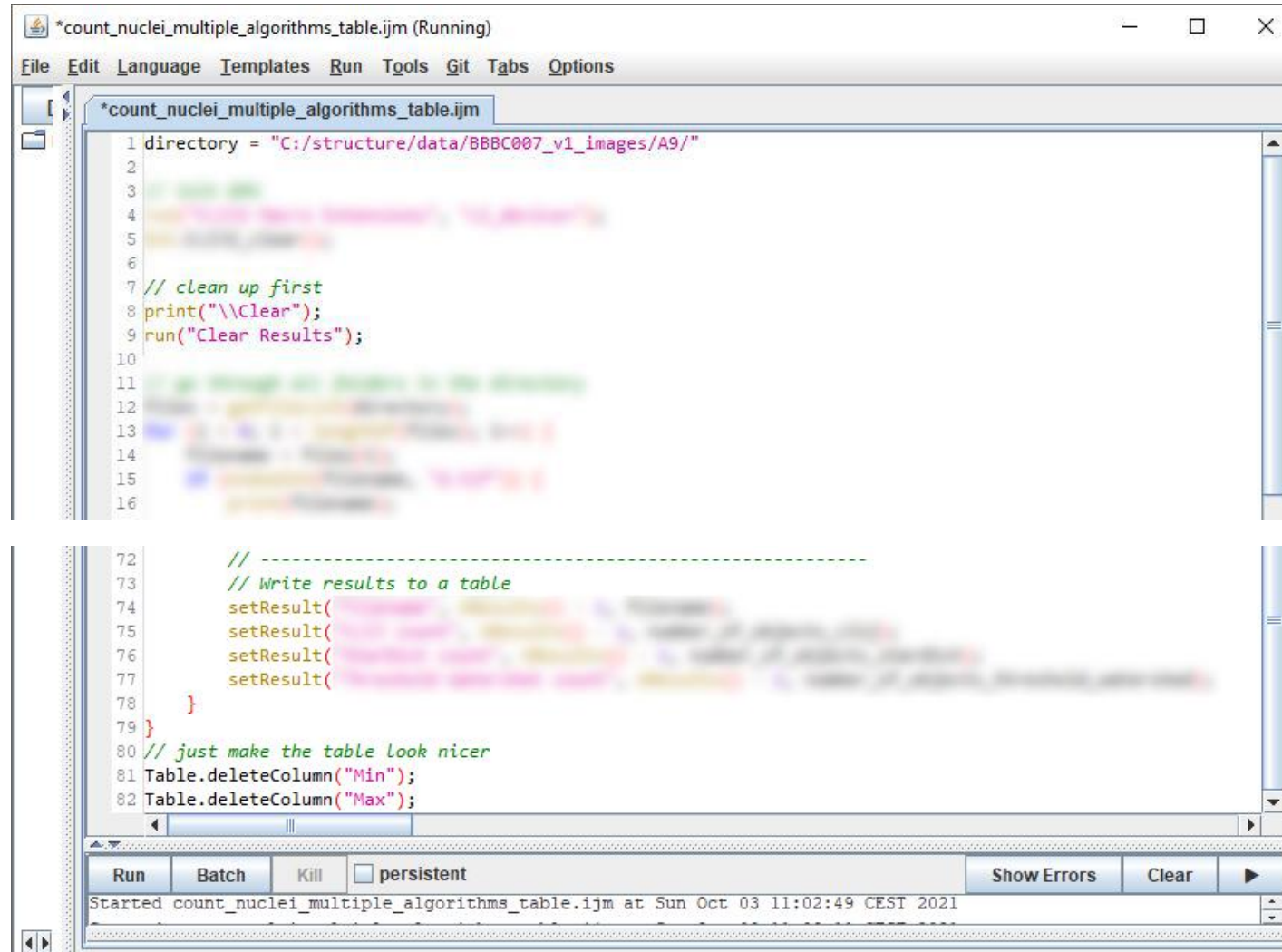


- Extend your script so that it applies the algorithms from yesterday to your image
 - Thresholding + Watershed
 - CLIJ
 - StarDist
- Hint: If you make the algorithms save ROIs to the ROI manager, you can count the elements in the ROI manager to know how many objects there are in the image.

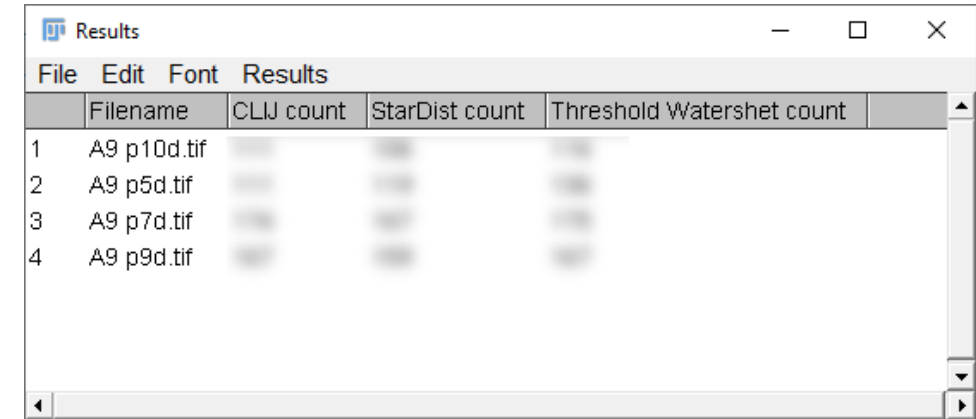


Exercise: Counting nuclei

- Extend your script so that it outputs the results to a table.

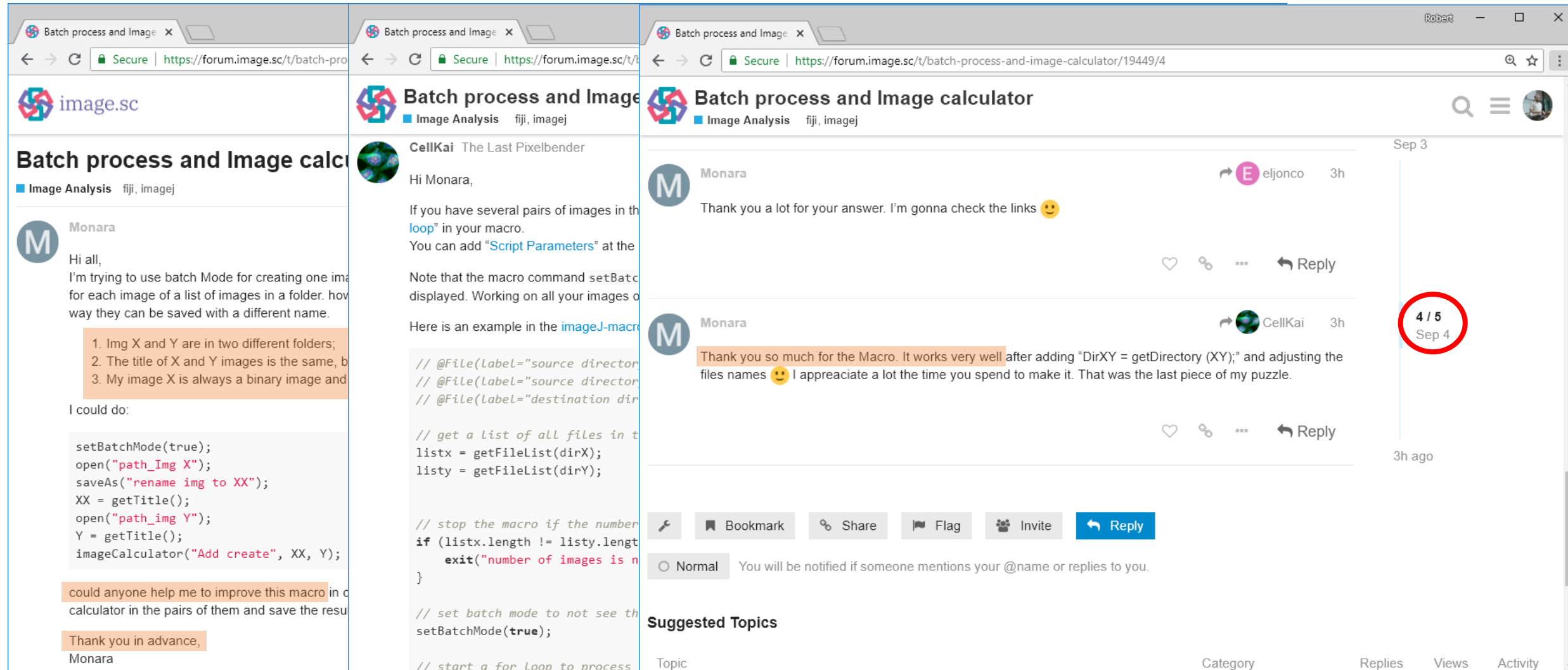


```
*count_nuclei_multiple_algorithms_table.ijm
1 directory = "C:/structure/data/BBBC007_v1_images/A9/"
2
3
4
5
6
7 // clean up first
8 print("\Clear");
9 run("Clear Results");
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
54
55
56
57
58
59
60
61
62
63
64
65
66
67
68
69
70
71
72 // -----
73 // Write results to a table
74 setResult(
75 setResult(
76 setResult(
77 setResult(
78 }
79 }
80 // just make the table look nicer
81 Table.deleteColumn("Min");
82 Table.deleteColumn("Max");
```



	Filename	CLIJ count	StarDist count	Threshold Watershet count
1	A9 p10d.tif			
2	A9 p5d.tif			
3	A9 p7d.tif			
4	A9 p9d.tif			

- Visit <http://forum.image.sc> !



The image displays three sequential screenshots of a forum thread on the ImageJ forum (forum.image.sc). The thread is titled "Batch process and Image calculator" and is part of the "Image Analysis" category. The first screenshot shows the initial post by user "Monara" asking for help with a macro for batch processing. The second screenshot shows a reply by user "CellKai" providing a macro example. The third screenshot shows a reply by user "Monara" thanking "CellKai" for the macro. A red circle highlights the page number "4 / 5" and the date "Sep 4".

Batch process and Image calculator
Image Analysis | fiji, imagej

Monara
Hi all,
I'm trying to use batch Mode for creating one image for each image of a list of images in a folder. how way they can be saved with a different name.

1. Img X and Y are in two different folders;
2. The title of X and Y images is the same, but
3. My image X is always a binary image and

I could do:

```
setBatchMode(true);
open("path_img X");
saveAs("rename img to XX");
XX = getTitle();
open("path_img Y");
Y = getTitle();
imageCalculator("Add create", XX, Y);
```

could anyone help me to improve this macro in the calculator in the pairs of them and save the result

Thank you in advance,
Monara

CellKai The Last Pixelbender
Hi Monara,
If you have several pairs of images in the "loop" in your macro.
You can add "Script Parameters" at the beginning of the macro.
Note that the macro command setBatchMode(true) is not displayed. Working on all your images of the folder.
Here is an example in the imageJ-macro language:

```
// @File(label="source directory", type="directory")
// @File(label="source directory", type="directory")
// @File(label="destination directory", type="directory")

// get a list of all files in the source directory
listx = getFileList(dirX);
listy = getFileList(dirY);

// stop the macro if the number of files is not the same
if (listx.length != listy.length) {
    exit("number of images is not the same");
}

// set batch mode to not see the progress bar
setBatchMode(true);

// start a for loop to process the images
```

Monara
Thank you a lot for your answer. I'm gonna check the links 😊

Monara
Thank you so much for the Macro. It works very well after adding "DirXY = getDirectory(XY);" and adjusting the files names 😊 I appreciate a lot the time you spend to make it. That was the last piece of my puzzle.

4 / 5
Sep 4

3h ago

Bookmark | Share | Flag | Invite | Reply

Normal | You will be notified if someone mentions your @name or replies to you.

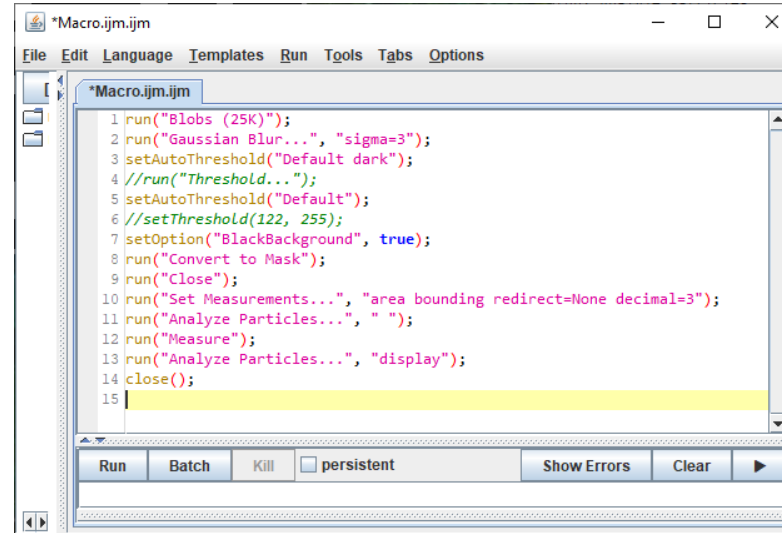
Suggested Topics

Topic	Category	Replies	Views	Activity
-------	----------	---------	-------	----------

Today, you learned

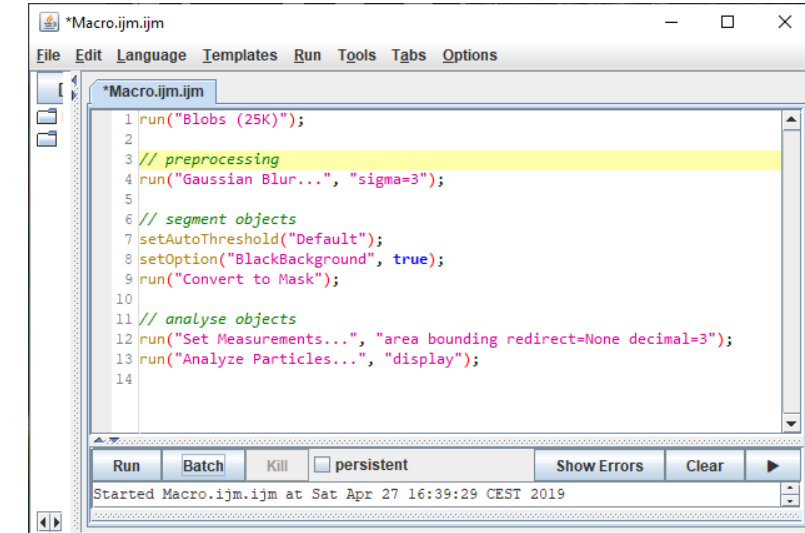
- Conditions (if)
- Loops (for)
- Results tables
- Arrays
- Plots
- Tables
- Processing folders

Reminder: Clean up recorded code! Write comments!



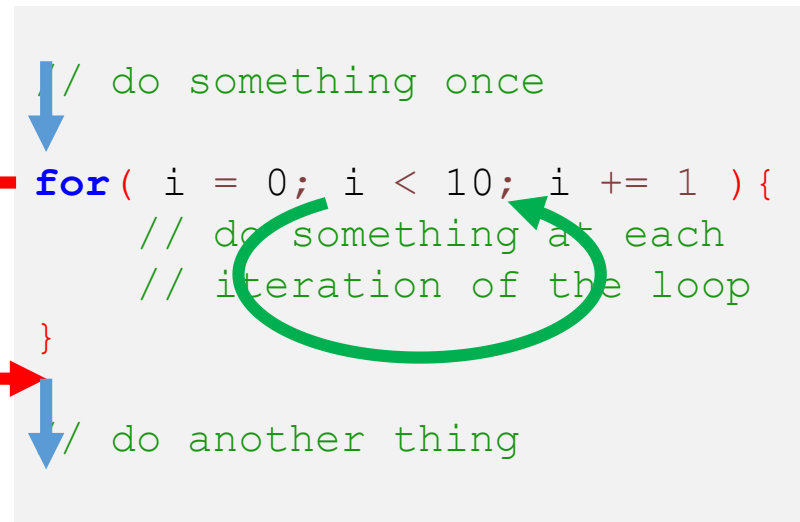
```

1 run("Blobs (25K)");
2 run("Gaussian Blur...", "sigma=3");
3 setAutoThreshold("Default dark");
4 //run("Threshold...");
5 setAutoThreshold("Default");
6 //setThreshold(122, 255);
7 setOption("BlackBackground", true);
8 run("Convert to Mask");
9 run("Close");
10 run("Set Measurements...", "area bounding redirect=None decimal=3");
11 run("Analyze Particles...", " ");
12 run("Measure");
13 run("Analyze Particles...", "display");
14 close();
15
    
```



```

1 run("Blobs (25K)");
2
3 // preprocessing
4 run("Gaussian Blur...", "sigma=3");
5
6 // segment objects
7 setAutoThreshold("Default");
8 setOption("BlackBackground", true);
9 run("Convert to Mask");
10
11 // analyse objects
12 run("Set Measurements...", "area bounding redirect=None decimal=3");
13 run("Analyze Particles...", "display");
14
    
```



```

// do something once
for( i = 0; i < 10; i += 1 ){
    // do something at each
    // iteration of the loop
}
// do another thing
    
```

