



臺大醫學院研發分處 第一共同研究室顯微影像核心

IMAGEJ顯微影像分析 與程式設計

零基礎的學生也能掌握基本顯微影像分析能力



海報網址



報名網址

2025 3.3-4.28 周一 13:30-14:30 共7堂
影像前處理、AI應用、自動化分析

課程資訊及授課教師

2025/3/3(一) 【生物影像分析概論】
溫榮崑 中央研究院 生化所 生物影像核心設施
研究助技師

2025/3/10(一) 【生物影像流程與小組討論編組】
許紹君 臺灣大學分子影像重點技術平台
助研究專家

2025/3/17(一) 【影像分析自動化】
張仁乾 日本理化學研究所
專門技術員

2025/3/24(一) 【互動式影像分析流程建立】
朱章臣 中央研究院 細生所 公共儀器室影像組
專案研發學者

2025/3/31(一) 【物件追蹤分析】
黃紀穎 中央研究院 植微所 細胞核心實驗室光學顯微鏡組
專案研究人員

2025/4/7(一) 【AI: 機器學習與深度學習工具介紹】
羅安琦 臺灣大學分子影像重點技術平台
副技師

2025/4/28(一) 小組發表
許紹君 臺灣大學分子影像重點技術平台 助研究專家
朱章臣 中央研究院 細生所共備影像組 專案研發學者

主辦單位: 臺大醫學院研發分處 第一共同研究室顯微影像核心
協辦單位: 中央研究院 生物化學研究所
地點: 基醫大樓講堂區 5 樓 未來教室 (原508教室)

課程簡介

本課程將介紹生物影像的基本元素、如何利用FIJI進行影像前處理、影像切割、特徵萃取、程式設計與編程、互動式影像分析流程與GPU加速、AI(機器學習與深度學習工具)、物件追蹤、常用的資料庫以及如何分享自己的作品。將視報名人數進行小組發表與討論, 利用工作中學習的方式提升課程效果。

課程目標

希望零基礎的學生參與課程後, 都能具備基本分析顯微影像的能力。

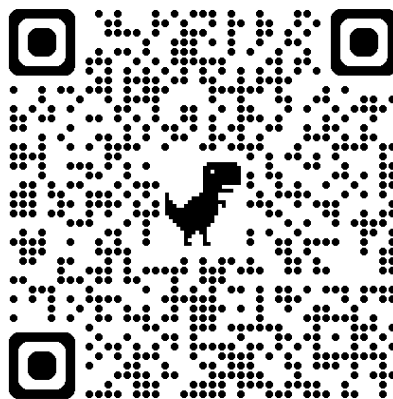
上課須知

- 即日起開放報名, 報名方式如下:
 - 提供姓名, EMAIL, 任職/就學單位, 實驗室主持人姓名。
 - 以一張A4篇幅文字說明實驗目的與欲解決的問題, 並以一張投影片頁面作為輔助材料。
- 優先錄取自備影像分析問題並想透過課程學會如何自己解決問題者。
- 課程會同步紀錄影音並於課後上傳至教學影音平台, 每堂課皆會點名, 上課出勤不得缺課超過一堂。
- 需自備筆電。

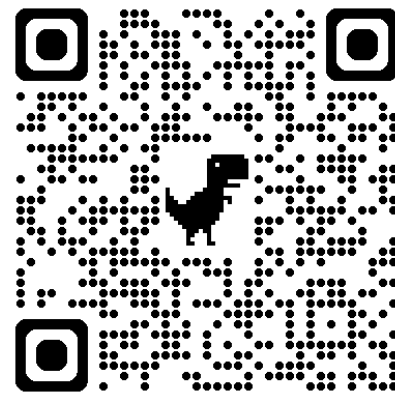
招生人數: 實體招收24人, 線上30人。
報名截止日: 額滿為止, 恕不開放現場候補。
聯絡人: 第一共研顯微影像核心 林思廷 szuting@ntu.edu.tw

上課注意事項:

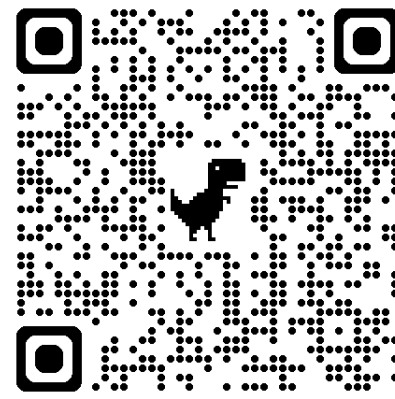
1. 教室內禁止攜帶食物飲料入內, 僅允許“白開水”, 請大家將食物飲料放置於教室外的桌上。
2. 請實體與線上學員掃描以下QR code進行線上簽到。
3. 請線上學員於課程開始前關閉自己的麥克風。
4. 線上學員若有問題, 請先按下“舉手”, 或於聊天室寫下問題, 將於課程結束後在場地時間允許下, 安排QA時間。
5. 現場學員發問時請使用麥克風才可進行收音。



線上簽到



課程材料與資訊連結



課後意見調查

ImageJ 顯微影像分析與程式設計

影像分析自動化 – ImageJ Macro



Jen-Chien Chang (張仁乾)
Research DX Foundation Team, RIKEN
2025.03.17



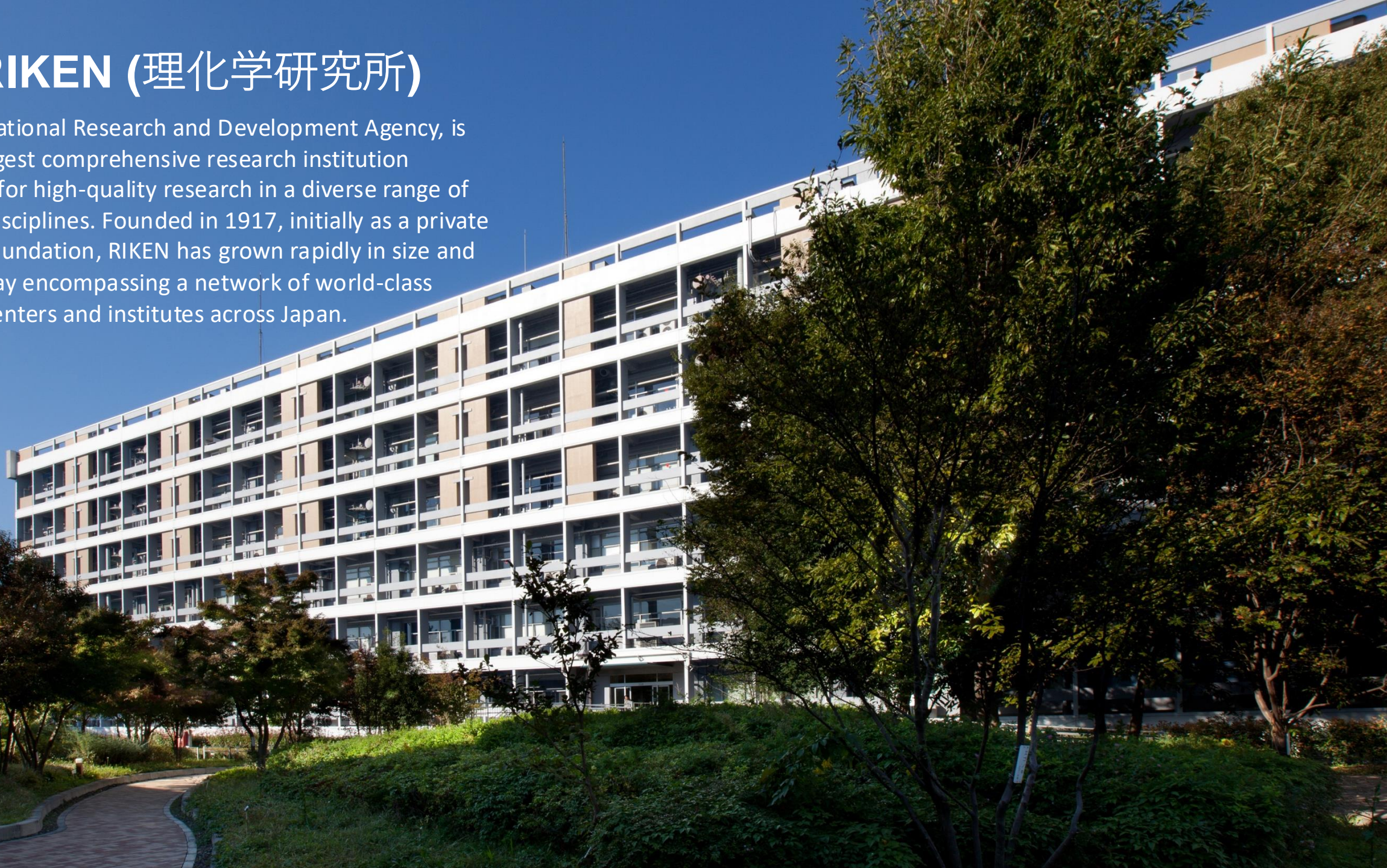
This document is licensed by [Jen-Chien Chang](#) under [CC-BY 4.0 license](#) unless mentioned otherwise





RIKEN (理化学研究所)

RIKEN, a National Research and Development Agency, is Japan's largest comprehensive research institution renowned for high-quality research in a diverse range of scientific disciplines. Founded in 1917, initially as a private research foundation, RIKEN has grown rapidly in size and scope, today encompassing a network of world-class research centers and institutes across Japan.



Jen-Chien Chang



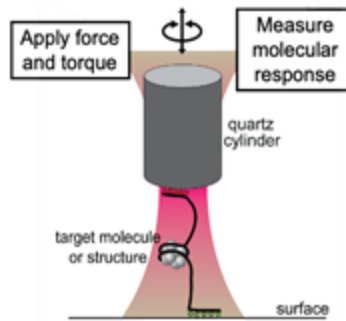
Taiwan 2008 USA 2014 Riken Yokohama 2023 Riken Yokohama

Physics B92202...

- Quantum optics
- Protein folding simulation

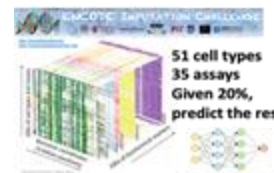
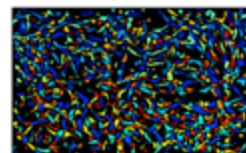
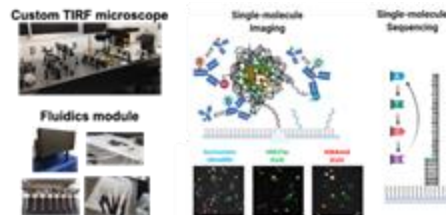
Biophysics

- Single-molecule biophysics
- Optical tweezers



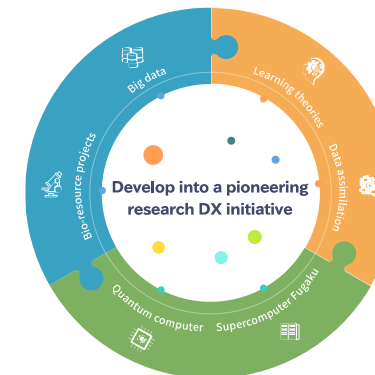
Biology & Medical science

- Epigenomics
- Sequencing technology
- Bioinformatics – genomics
- Bioinformatics – imaging



Research Digital Transformation (DX)

- Technical support for data management & analysis
 - Imaging & Sequencing data
 - Core facilities & individual labs



RIKEN TRIP

Main RIKEN campus & core facilities



Harima 播磨
(SPring-8 synchrotron radiation facility)
- X-ray CT image



Wako 和光
- Proteomics core lab
- Brain images



Tsukuba 筑波
(BioResource Center)
- Cell bank image
- Mouse CT image

Kobe 神戸
- Proteomics core lab
- Bioimaging facility



Yokohama 横浜
- NMR facility
- Genome platform



SSBD:repository

Public repository for sharing bioimaging data and quantitative biosystems dynamic data used in papers

SSBD:repository

[SSBD:database](#)
[SSBD:repository](#)
[Resources](#)
[About](#)

Organism
ex. C. elegans
Search
Clear

Introduction

System Science of Biological Dynamics repository (SSBD:repository) is an open data archive that stores and publishes bioimaging and biological quantitative datasets that are associated with published or to be published studies. It allow other researchers to access and download those datasets for reference or for further investigations.

Find the dataset from search box above, or see the dataset list in [Resources](#).

SSBD:repository has started operation in 2016, under the life science database integration promotion project of the Japan Science and Technology (JST), National Bioscience Database Center (NBDC). Currently it is funded from RIKEN, JST and Grant-in-Aid for Scientific Research from the Ministry of Education, Culture, Sports, Science and Technology of Japan (MEXT).

For an overview of the SSBD:database/repository, please refer to the paper, Tohsato, Y., Ho, K. H. L., Kyoda, K., and Onami S. (2016) "SSBD: a dataset of quantitative data of spatiotemporal dynamics of biological phenomena." *Bioinformatics*, 32(22): 3471-3479 <https://doi.org/10.1093/bioinformatics/btw417>

Share your data in SSBD:repository

SSBD:repository covers a wide range of biological phenomena, from single molecules to cells / individuals. DOI (Digital Object Identifier) can be assigned to each dataset. Datasets in SSBD:repository are licensed under [CC BY](#), [CC BY-NC-SA](#), etc, upon request of the data contributors.

If you would like to share your image data or quantitative data of biological phenomena, or any

News

2021-02-09,
System maintenance on Feb 9
Due to system maintenance, the 4D visualizer and APIs will not be available on Feb 9, 9:00-16:00.
Sorry for the inconvenience.

2020-12-24,
SSBD 2020 Update
SSBD 2020 update, the metadata database fully renewed, and 77 image datasets (7GB) and 4 quantitative datasets are added. We are planning next database update in a half year for COVID-19 situation.

2020-08-12,
A new paper on BD5 is published in PLOS ONE
A new paper "BD5: an open HDF5-based data format to represent quantitative biological dynamics data" is now published in PLOS ONE.
<https://doi.org/10.1371/journal.pone.0237468>

[Older news...](#)

Funding



Team Leader:
ONAMI Shuichi
大浪修一

SSBD:database

Added-value database that shares highly reusable bioimaging data and quantitative biosystems dynamics data **with rich metadata**.

SSBD:database

[SSBD:database](#)
[SSBD:repository](#)
[SSBD:OMERO](#)
[Resources](#)
[Tools](#)
[About](#)

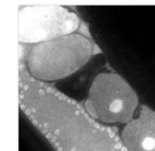
Organism
ex. C. elegans
Search
Clear

Introduction

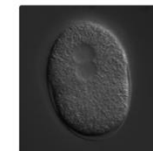
System Science of Biological Dynamics database (SSBD:database) provides a rich set of open resources for analyzing quantitative data and microscopy images of biological objects, such as single-molecule, cell, gene expression nuclei, etc. Quantitative biological data and microscopy images are collected from a variety of species, sources and methods. These include data obtained from both experiment and computational simulation.

Samples

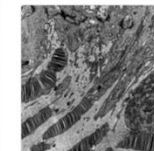
Microscopy Images



Calcium response and shape changes in oocyte of C. elegans



DIC image of nuclear division dynamics in C. elegans embryo



TEM image of retinal tissue from human embryonic stem cells

News

2020-12-24,
SSBD 2020 Update
SSBD 2020 update, the metadata database fully renewed, and 77 image datasets (7GB) and 4 quantitative datasets are added. We are planning next database update in a half year for COVID-19 situation.

2020-08-12,
A new paper on BD5 is published in PLOS ONE
A new paper "BD5: an open HDF5-based data format to represent quantitative biological dynamics data" is now published in PLOS ONE.
<https://doi.org/10.1371/journal.pone.0237468>

[Older news...](#)

Funding



Automate the image analysis workflow with ImageJ Macro

- Goal: Automate your analysis workflow using **ImageJ Macro**
 - For those with programming experience: Become familiar with ImageJ Macro syntax
 - If you have no programming experience: Learn to read and adapt templates for your workflow

- Outline

- Part1: Introduction to reproducible analysis with ImageJ Macro

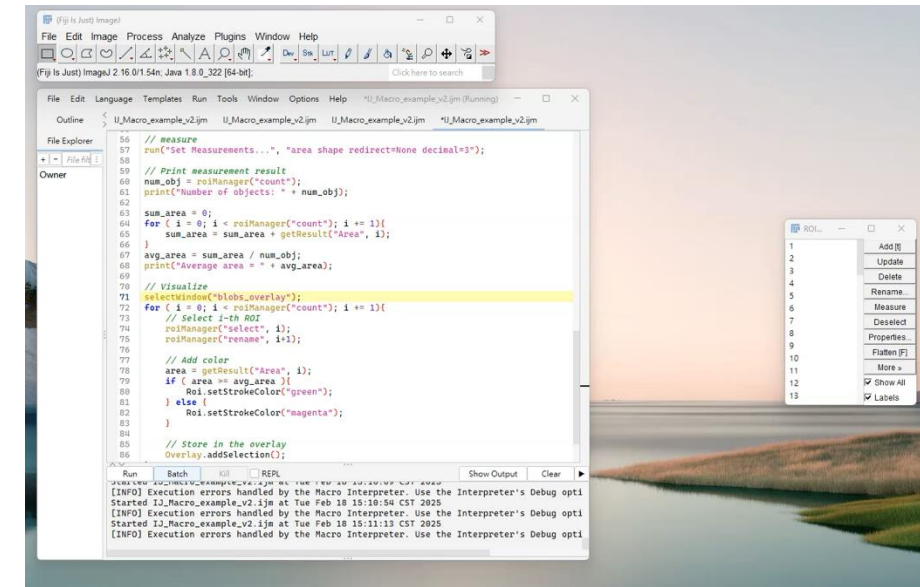
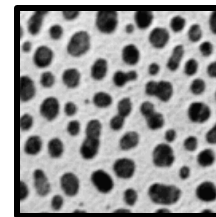
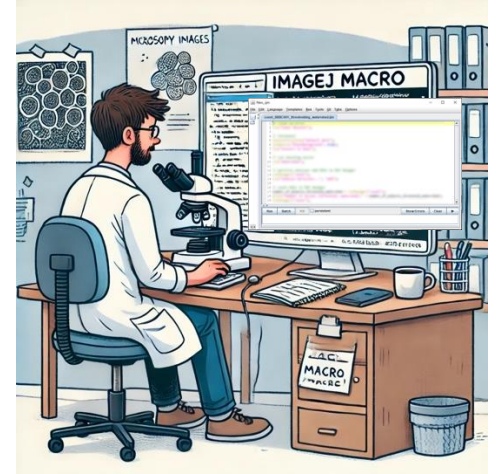
Common
concept

- Variables, array, and operations
- Conditions and loops
- Functions
- Writing good code
- Recording Macros and working with images
- ROIs and Overlays
- Plots and Tables
- Batch processing, File I/O

IJ-specific

- Part2: Hands-on session

- Ex1: Automate the flow for object counting
- Ex2: File I/O & batch processing



Reference

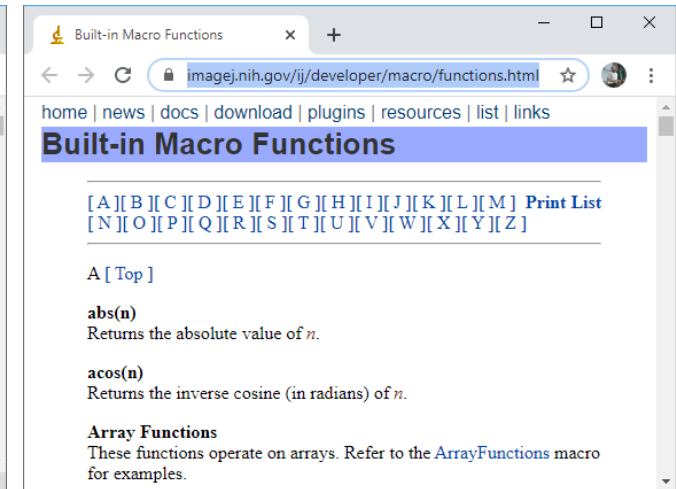
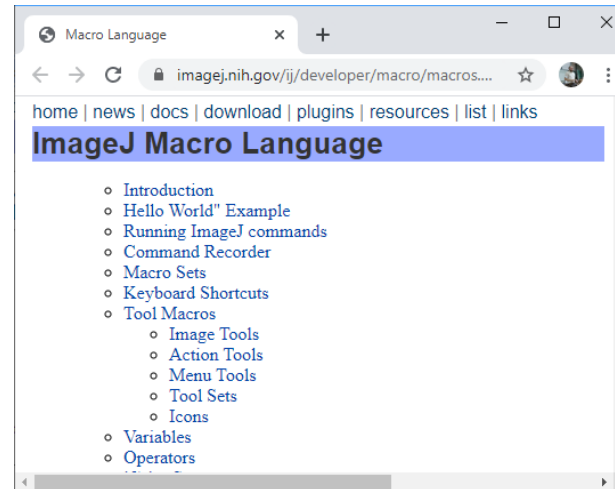
- This document is adapted mainly from the materials maintained by **Robert Haase** under CC-BY 4.0 license
 - Slide for the virtual course at CCI Gothenburg, October 2021
<https://f1000research.com/slides/10-1000>
 - Example codes: https://github.com/BiAPoL/CCI_Gothenburg_ImageJ_Macro_Course_2021
 - Youtube: <https://www.youtube.com/@haesleinhuepf/videos>



Resources

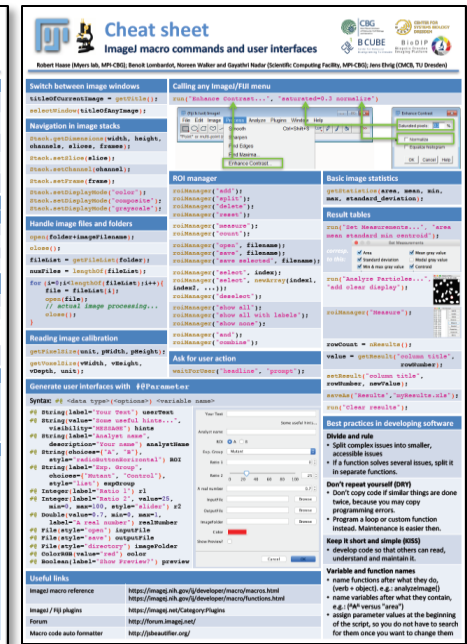
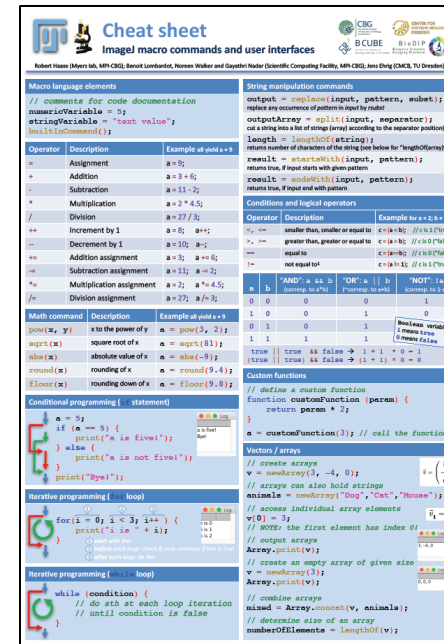
- ImageJ Macro official websites

- <https://imagej.net/ij/developer/macro/macros.html>
- <https://imagej.net/ij/developer/macro/functions.html>



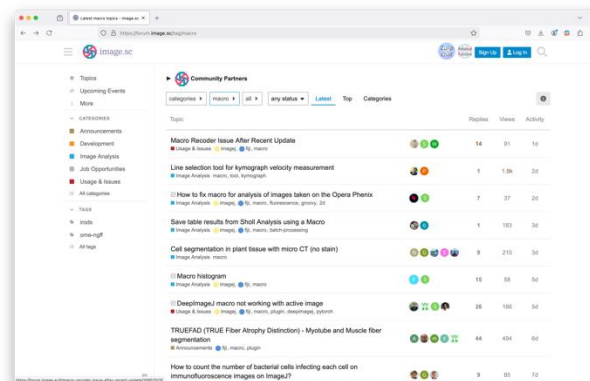
- Cheat sheet

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



- image.sc

- <https://forum.image.sc/>



Part1: Introduction to reproducible analysis with **ImageJ Macro**

Variables

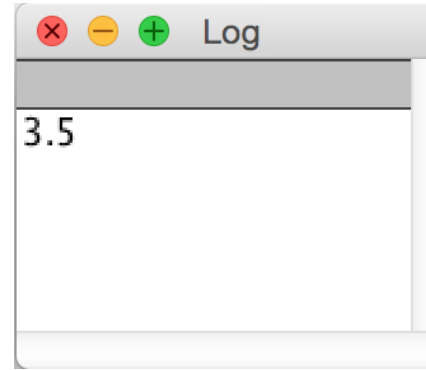
- Variables hold **values** or **text** and can be connected using **operators**

```
// initialise program
a = 1;
b = 2.5;

// run complicated algorithm
final_result = a + b;

// print the result
print( final_result );
```

variables.ijm



- Texts are called “**strings**” in the world of programming

- They represent a chain of characters
- Strings can be concatenated with “+”

```
// initialise program
firstname = "Shohei";
lastname = "Ohtani";

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

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

string_variables.ijm

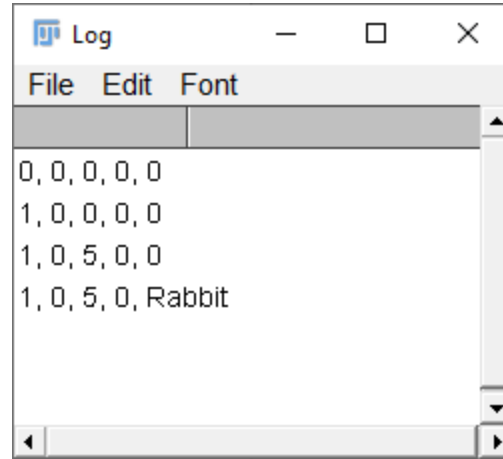


Array

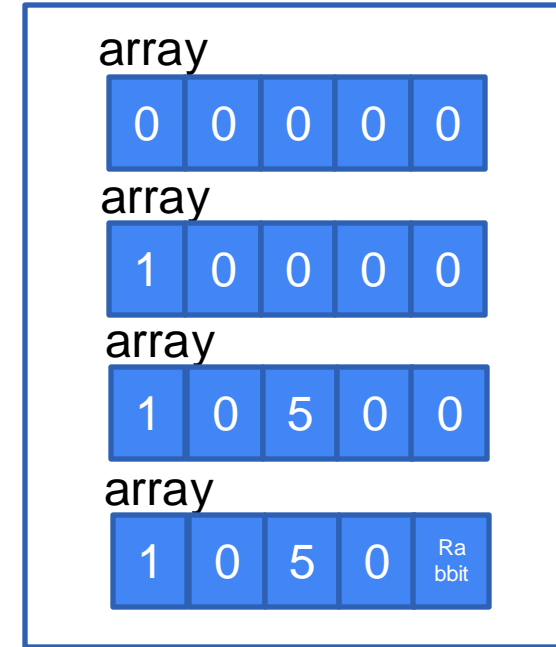
- Arrays are variables, where you can store multiple stuff

```
1  
2 array = newArray(5);  
3 Array.print(array);  
4  
5 array[0] = 1;  
6 Array.print(array);  
7  
8 array[2] = 5;  
9 Array.print(array);  
10  
11 array[4] = "Rabbit";  
12 Array.print(array);  
13
```

arrays_access.ijm



Computer memory



- Arrays can be initialized with values (default: 0)

```
v = newArray(3, -4, 0);
```

```
animals = newArray("dog", "cat", "mouse");
```

- Arrays can be concatenated:

```
mixed = Array.concat(v, animals);
```


// Comments

- **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 );
```

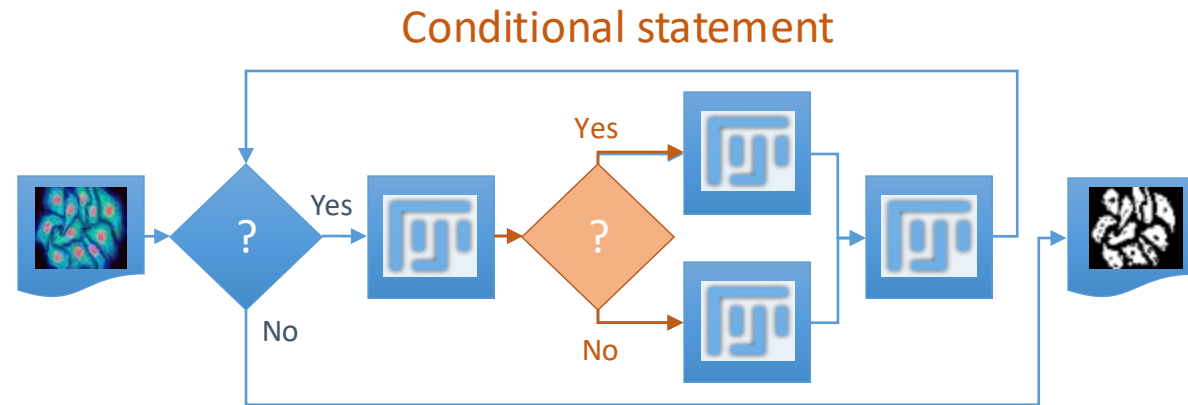
good_comments.ijm

Mathematical operations

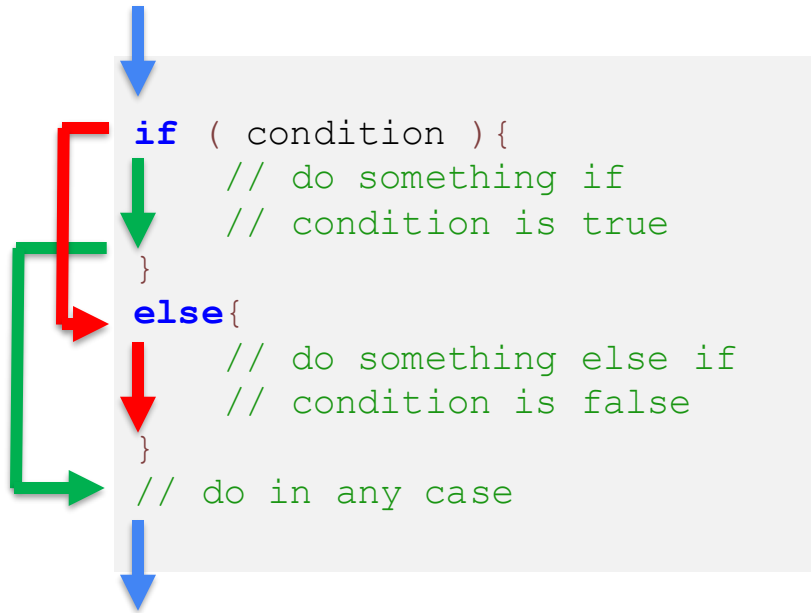
- Math commands supplement operators to be able to implement any form of calculations

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>

Conditions (if / else)



- The if / else statement controls analysis workflow, allowing 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.



```
// initialise program
quality = 99.5;

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

Comparison operators & logic operators

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

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

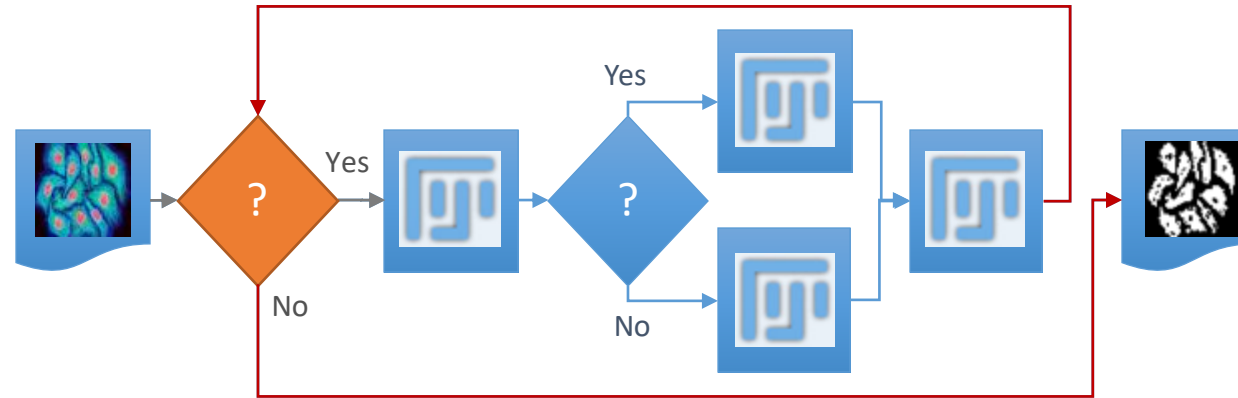
- Logic operators always take conditions as operands and result in a condition

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

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

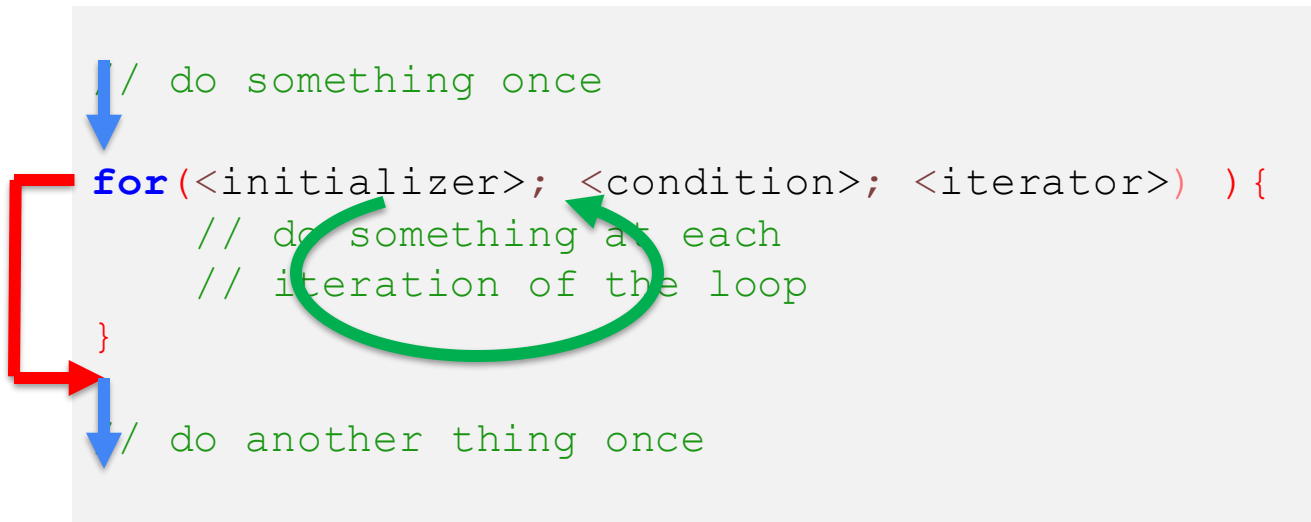
Operator	Description	Example
&&	and	a < b && b < c
	or	a < b b > c
!	negation ("not")	!(a == b)

Loops (for)



Loop statement

- The `for` statement allows us 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.



```
for( i = 0; i < 10; i += 1 ){
    print(i);
}
```

Log	
0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Functions

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

The screenshot shows a code editor window titled `*New_.ijm` with a menu bar (File, Edit, Language, Templates, Run, Tools, Tabs, Options). The code is as follows:

```
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
```

Annotations with callouts point to specific parts of the code:

- function call**: Points to the `sum(3, 4)` call on line 3.
- name**: Points to the `sum` keyword in the function definition on line 11.
- parameters**: Points to the `a, b` parameters in the function definition on line 11.
- function definition**: Points to the `function sum(a, b) {` line on line 11.
- return value**: Points to the `return result;` statement on line 16.
- user documentation**: Points to the comment `// doing some math with two numbers` on line 12.

The function definition on lines 11-17 is highlighted in yellow. Below the code editor, a list of functions is visible, including `sum(a, b)`. To the right, a documentation pane shows the details for `sum(a, b)`, including the comment `// doing some math with two numbers` and the note `User defined function as specified in line 11.`

Writing good (readable) codes

- 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;

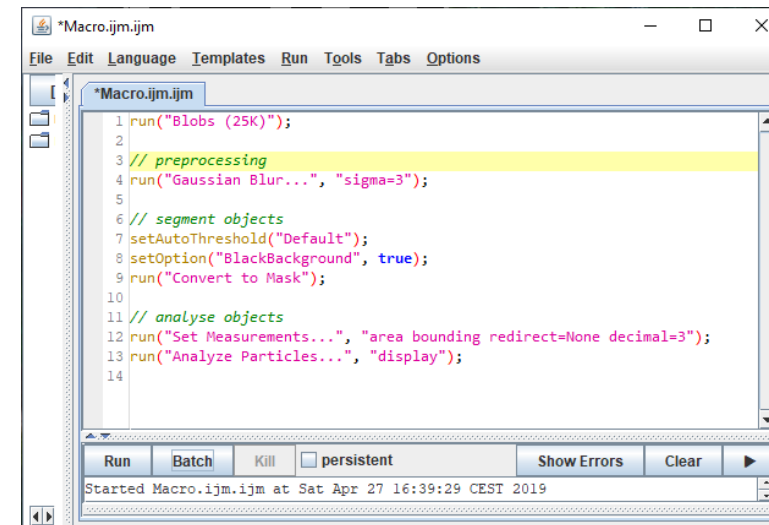
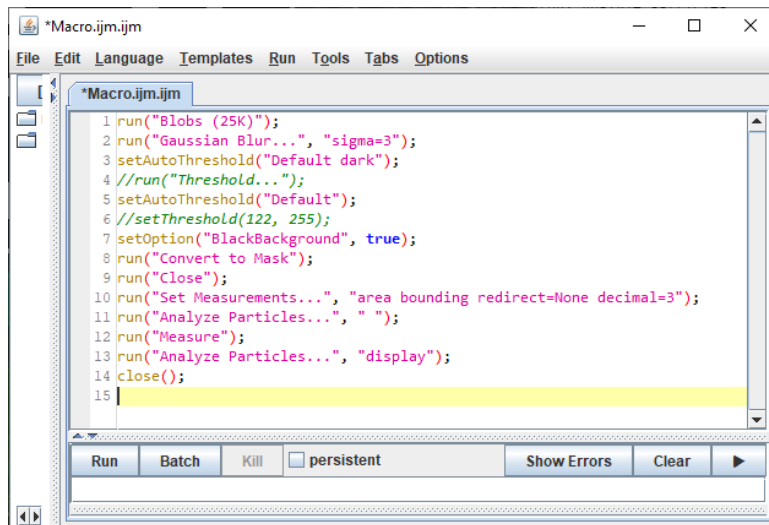
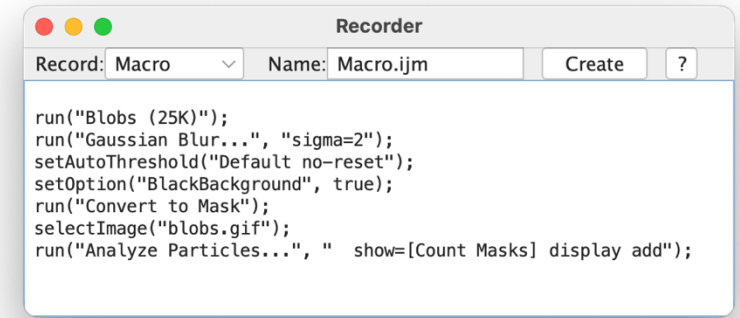
// execute algorithm
d = (a + b) / c;

// evaluate result

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

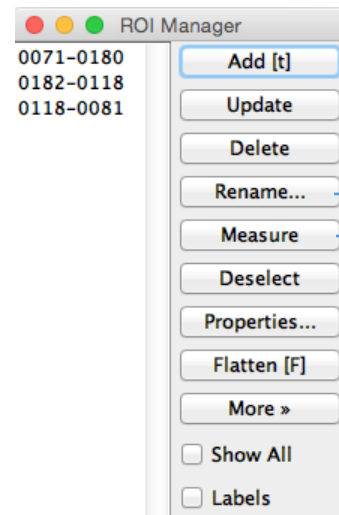
Macro recording

- Almost every action you perform in ImageJ by clicking is recorded!
- The script in recorder can be edited and used for creating .ijm file
- 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.



ROI Manager

- ROI (Regions of interest)
 - ROIs can be used for visualization of (intermediate) results
 - Every button in the ROI Manager can be called from a macro.



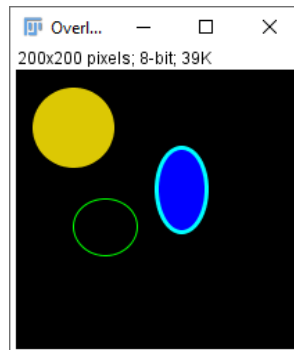
Open...
Save...
Fill
Draw
AND
OR (Combine)
XOR
Split
Add Particles
Multi Measure
Multi Plot
Sort
Specify...
Remove Positions...
Labels...
List
Interpolate ROIs
Translate...

Built-in command

roiManager("add");
roiManager("delete");
roiManager("rename", name);
roiManager("measure");
roiManager("deselect");
roiManager("open", filename);
roiManager("save", filename);
roiManager("save selected", filename);
roiManager("select", index);
roiManager("select", newArray(index1, index2, ...));
roiManager("show all");
roiManager("show all with labels");
roiManager("show none");
roiManager("and");
roiManager("combine");
roiManager("reset");
roiManager("split");
roiManager("count");

- Overlay: collection of ROIs

```
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 overlays.ijm
```



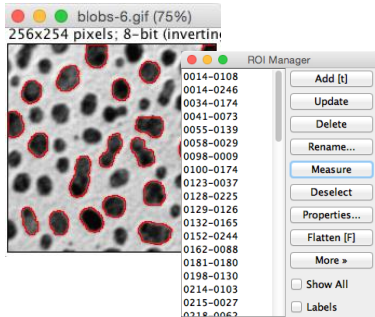
Results tables

- ImageJ/FIJIs tool for collecting data/measurements
- Two ways for creating result tables

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

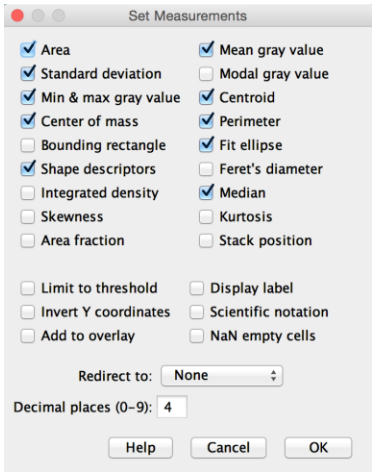


```
roiManager("Measure");
```



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

- Set Measurements



Results tables

- 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();  
print(r); // number of rows
```

Log
10

Built-in command

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

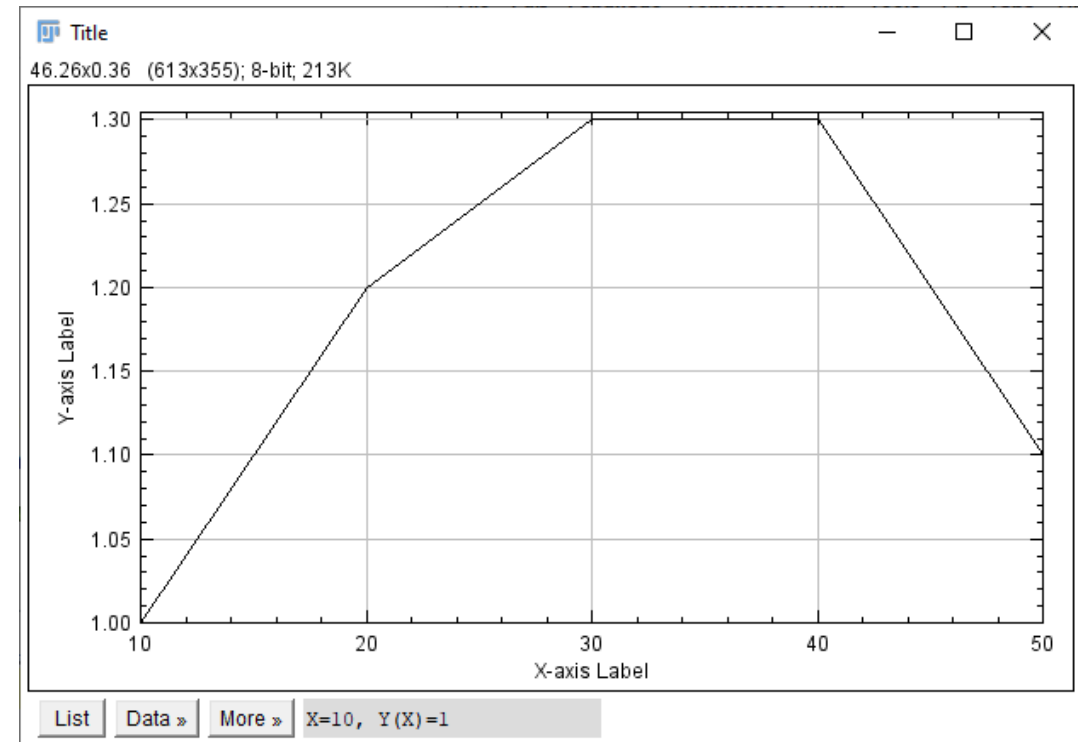
```
rowCount = nResults();
```

Plots in ImageJ

- Simple plots can be drawn using arrays

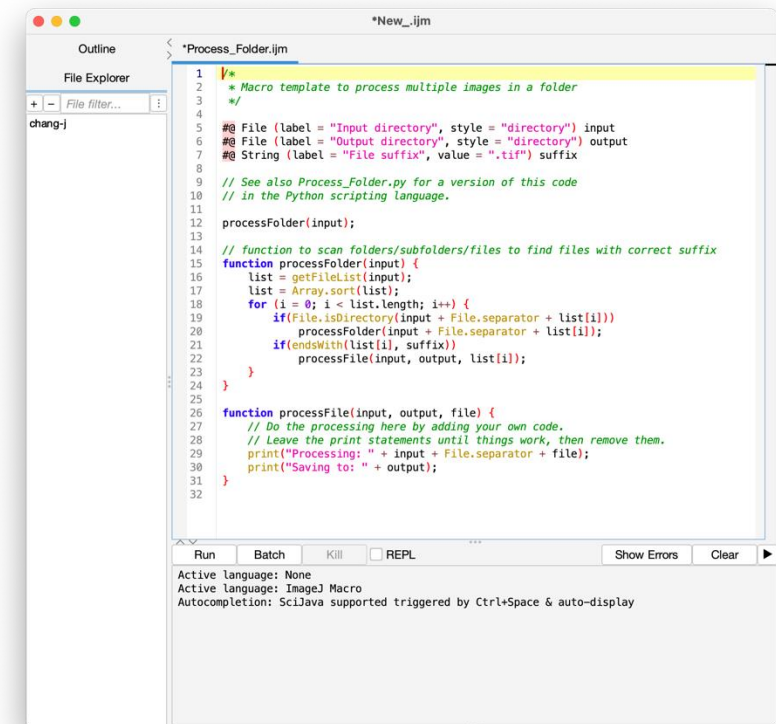
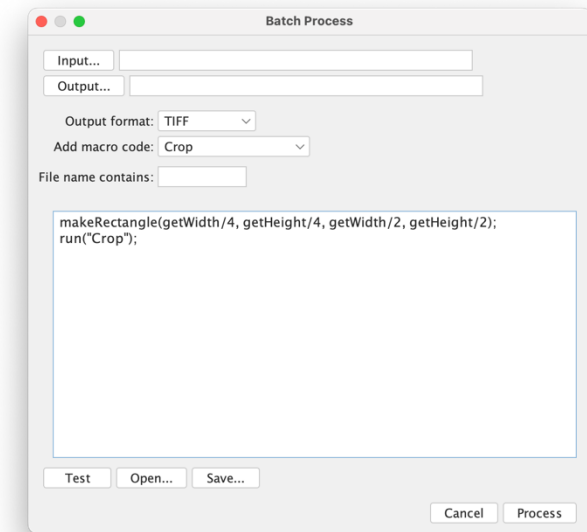
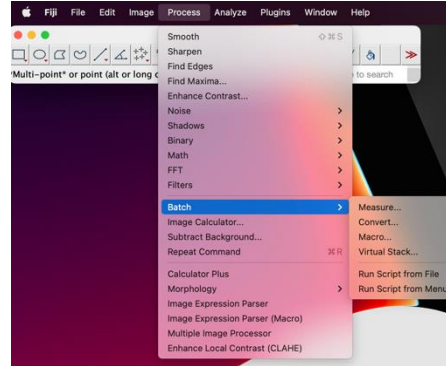
```
array_example.ijm (Running)
File Edit Language Templates Run Tools Git Tabs Options
array_example.ijm
1 // Define two arrays with 5 elements
2 x_values = newArray(5);
3 y_values = newArray(5);
4
5 // put some numbers in these 2x5 array elements
6 x_values[0] = 10;
7 y_values[0] = 1;
8
9 x_values[1] = 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 ijmmcd_example.ijm at Wed Apr 15 18:57:02 CEST 2020
```



Batch processing

- Use ImageJ submenu
 - Measure
 - Convert
 - Macro
 - Virtual Stack
 - Note: silently override existing files with the same name
- Create your own
 - Start with a template in script editor
 - *Template › ImageJ 1.x › Batch › Process Folder (ImageJ Macro)*



Working with image files

- 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	

Working with image files in a folder

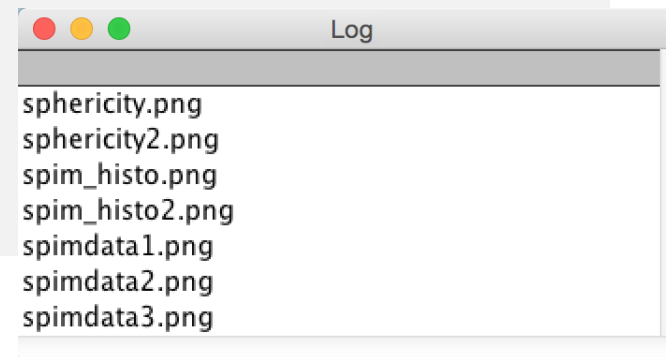
- The files in a folder are represented as an array `[]`

```
// 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];

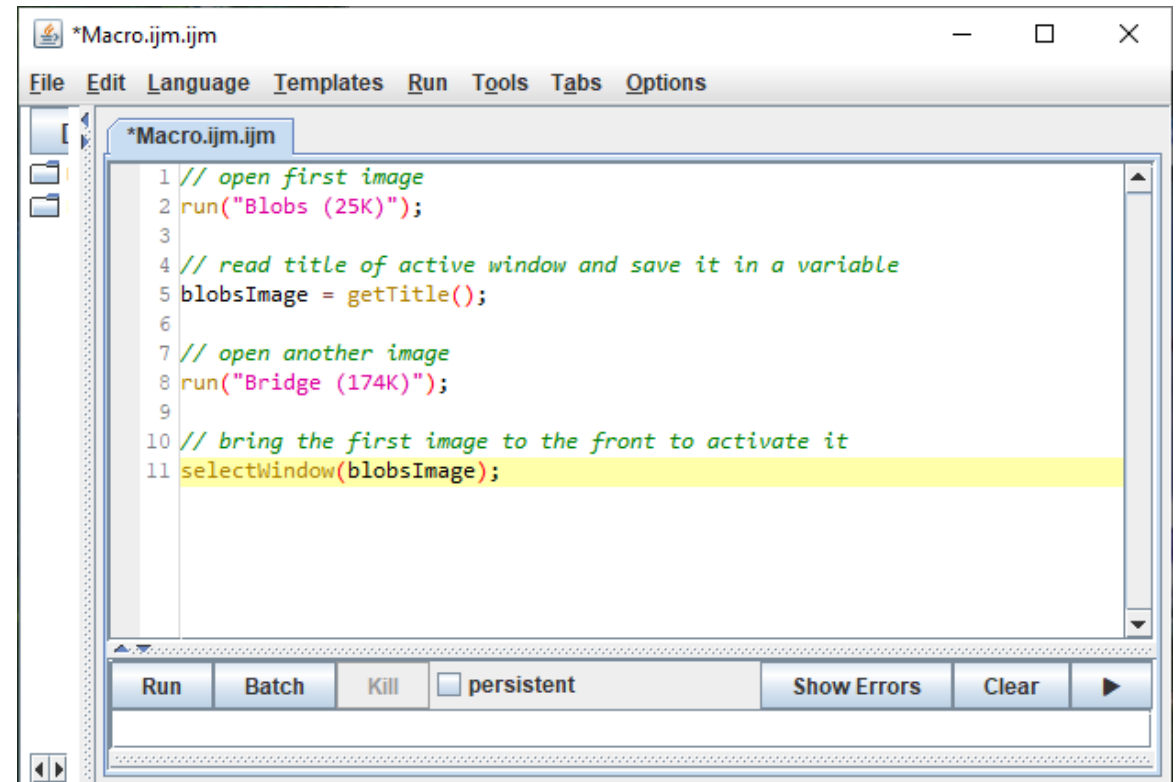
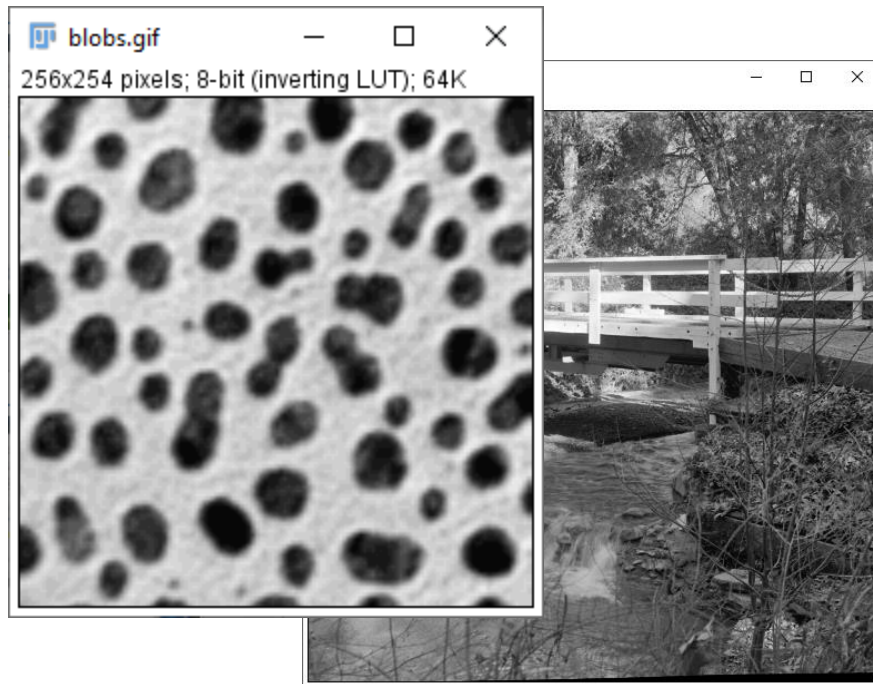
    print( filename );
}
```



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

Hint: switching between windows

- 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



Hint: logging

- 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 }
```

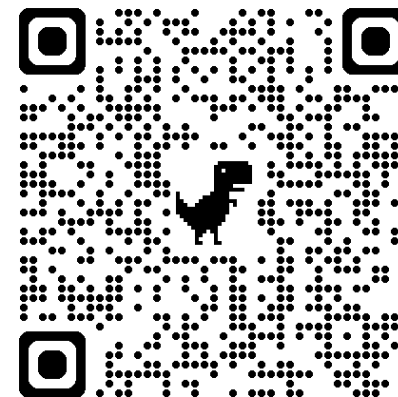
save_csv_file.ijm

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

Part2: Hands-on session

Summary & Acknowledgement

- ImageJ Macro intro
 - Common programming concept
 - ImageJ Macro syntax and functions
- Exercise
 - Ex1: automate a basic workflow
 - Ex2: batch processing
- Next week: Interactive analysis with CLIJ



課後意見調查

