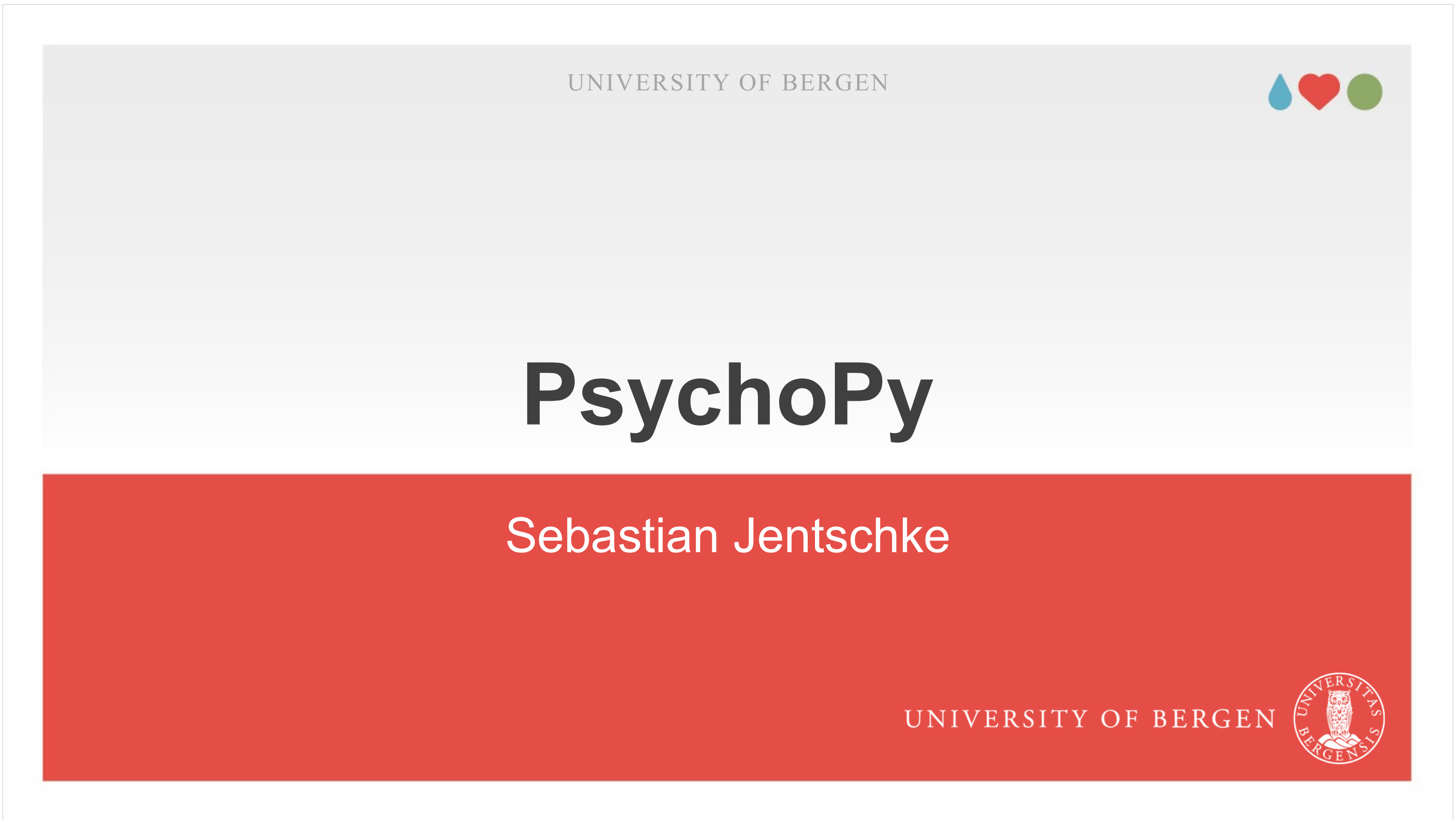


UNIVERSITY OF BERGEN

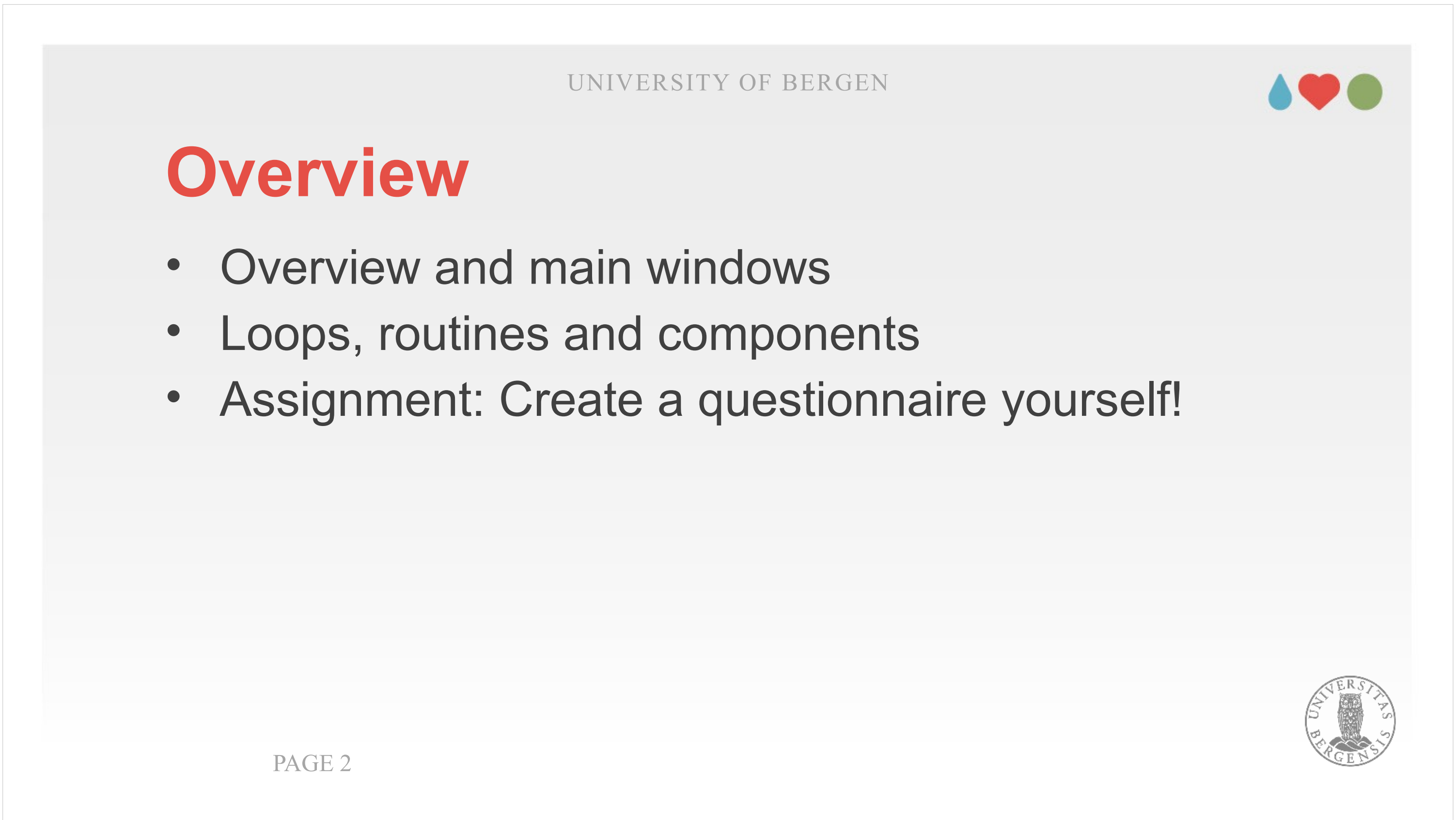
PsychoPy

Sebastian Jentschke

UNIVERSITY OF BERGEN

Welcome to our session about creating computerized experiments using PsychoPy.

Today, we will in a first step get us an overview about the main functions in PsychoPy. Afterwards, your task is to assemble your first computer experiment (which actually is a questionnaire).

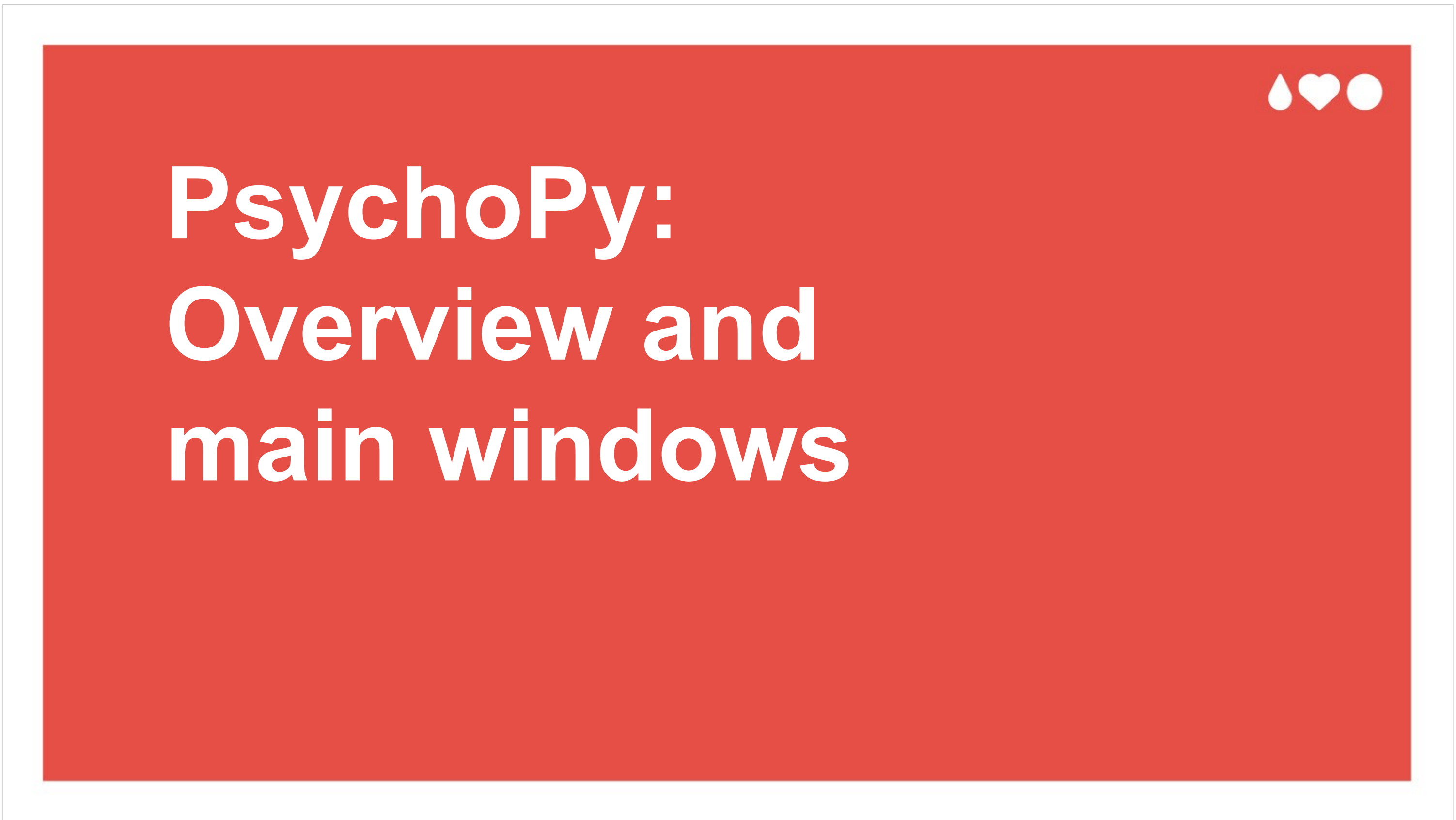


Overview

- Overview and main windows
- Loops, routines and components
- Assignment: Create a questionnaire yourself!

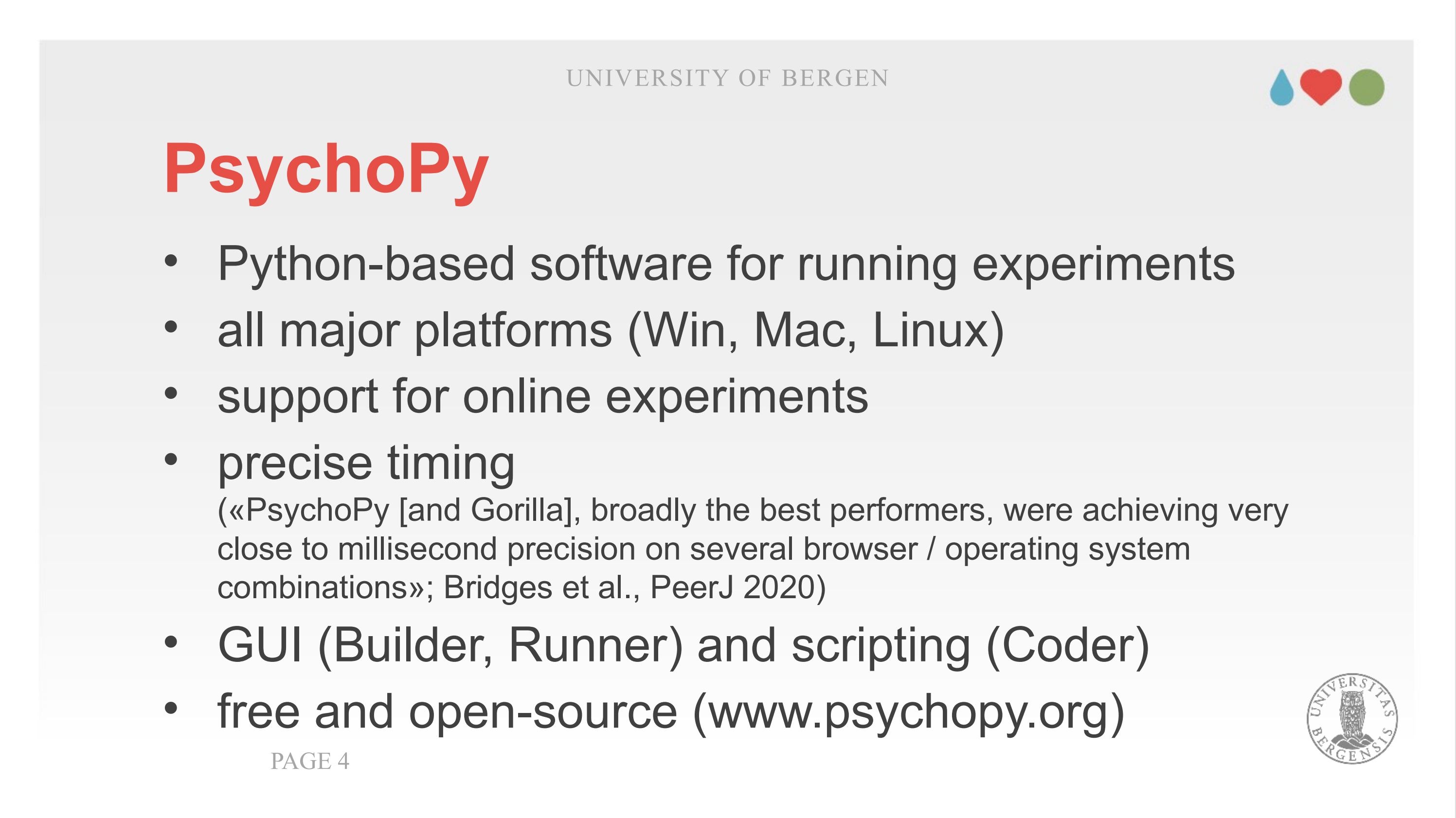
Today, we will in a first step get us an overview about what PsychoPy is and how it looks like when you open it. Afterwards, we will speak about the different „ingredients“ that you put into your experiments: Loops, routines and the components you put into the routines.

Finally, you get the task to assemble your first computer experiment (which actually is a questionnaire).



PsychoPy: Overview and main windows

The next couple of slides introduces PsychoPy and how it will look like when you first open it.



UNIVERSITY OF BERGEN

PsychoPy

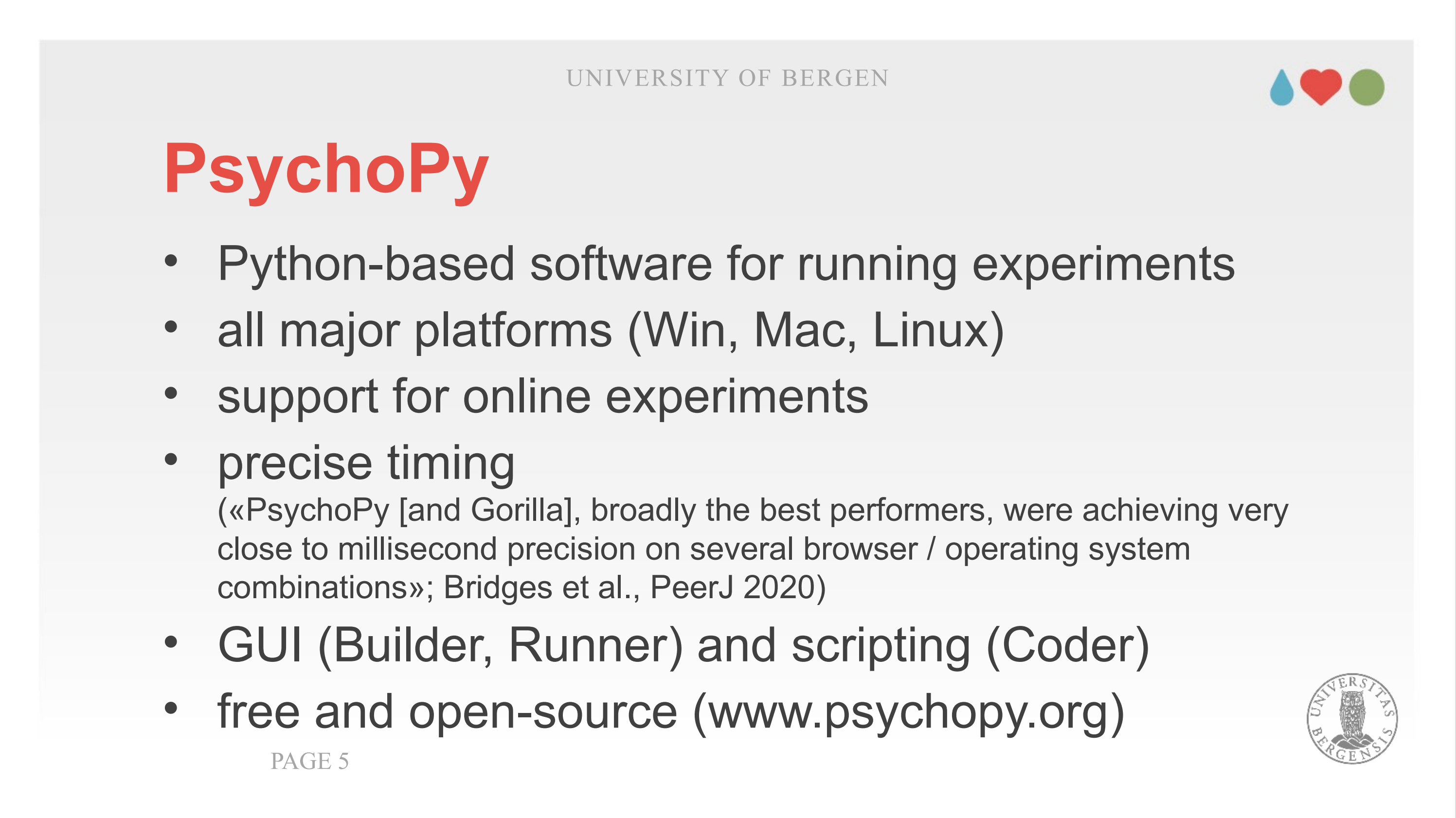
- Python-based software for running experiments
- all major platforms (Win, Mac, Linux)
- support for online experiments
- precise timing
  («PsychoPy [and Gorilla], broadly the best performers, were achieving very close to millisecond precision on several browser / operating system combinations»; Bridges et al., PeerJ 2020)
- GUI (Builder, Runner) and scripting (Coder)
- free and open-source (www.psychopy.org)

PAGE 4

PsychoPy is a software package for running experiments on your computer or on a web platform. It is based upon Python which is a popular programming language.

It is available for all major operating systems (i.e., Windows, Mac OS, and Linux) and since version 3 it is suited to prepare web experiments which you then can run on a web page called pavlovia.org

It originally was created for conducting visual research where very precise timing is key – often you present stimuli only very briefly (i.e., a time range of several milliseconds). By now, it has a much more comprehensive set of tools also permitting e.g. sounds, etc.



PsychoPy

- Python-based software for running experiments
- all major platforms (Win, Mac, Linux)
- support for online experiments
- precise timing
  («PsychoPy [and Gorilla], broadly the best performers, were achieving very close to millisecond precision on several browser / operating system combinations»; Bridges et al., PeerJ 2020)
- GUI (Builder, Runner) and scripting (Coder)
- free and open-source (www.psychopy.org)

It combines a relatively low threshold for learning – it is easy to set up basic experiments by just clicking components together – with a mighty scripting language – Python – which allows you to fulfil (more or less) any requirement that you might have.

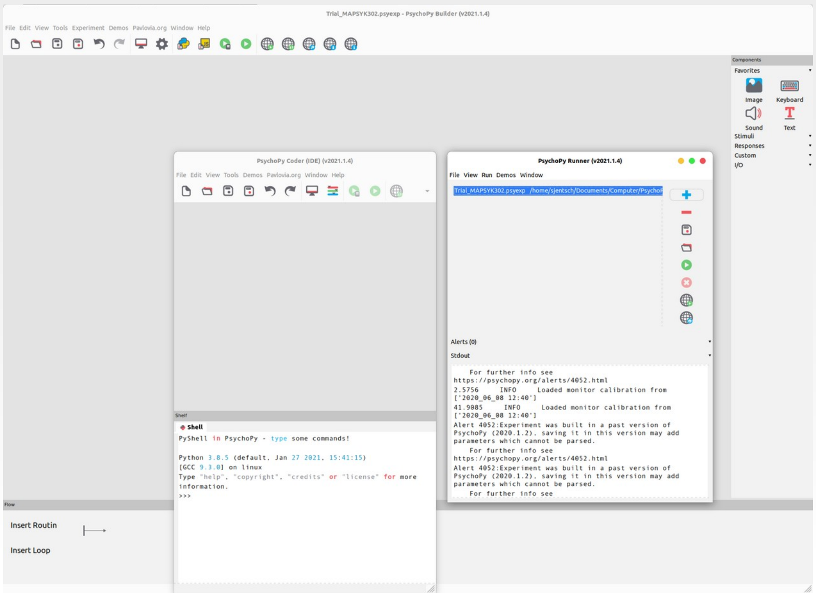
Finally, it is free and open source: This means a couple of things. It is free of charge. You can use just download and use it. Open source software typically also comes with a community which provides documentation, example experiments, help in forums, etc.

PsychoPy also has a strong involvement with the Open Science Foundation, it is easy to upload experiments and data to share them.

One comment on the slides: PsychoPy is evolving at a quite fast pace. I tried updating them, but some of the pictures on the slides might already look a little different from what you see in PsychoPy now. But typically, the main elements stay in place.

UNIVERSITY OF BERGEN

PsychoPy


Trial_MAPSYK302.psyexp - PsychoPy Builder (v2021.1.4)
File Edit View Tools Experiment Demos Pavlovia.org Window Help
Components
Favorites
Image
Keyboard
Sound
Text
Stimuli
Responses
Custom
I/O
Flow
Insert Routin
Insert Loop
PsychoPy Coder (IDE) (v2021.1.4)
File Edit View Tools Demos Pavlovia.org Window Help
Shelf
Shell
PsychoPy Runner (v2021.1.4)
File View Run Demos Window
Alerts (0)
Stdout

Trial_MAPSYK302.psyexp /home/sjentsch/Documents/Computer/PsychoP

```
PyShell in PsychoPy - type some commands!

Python 3.8.5 (default, Jan 27 2021, 15:41:15)
[GCC 9.3.0] on linux
Type "help", "copyright", "credits" or "license" for more
information.
>>>
```

```
    For further info see
https://psychopy.org/alerts/4052.html
2.5756     INFO     Loaded monitor calibration from
['2020_06_08 12:40']
41.9085     INFO     Loaded monitor calibration from
['2020_06_08 12:40']
Alert 4052:Experiment was built in a past version of
PsychoPy (2020.1.2), saving it in this version may add
parameters which cannot be parsed.
    For further info see
https://psychopy.org/alerts/4052.html
Alert 4052:Experiment was built in a past version of
PsychoPy (2020.1.2), saving it in this version may add
parameters which cannot be parsed.
    For further info see
```

When you open PsychoPy you will encounter three windows: The Runner, the Builder, and the Coder.

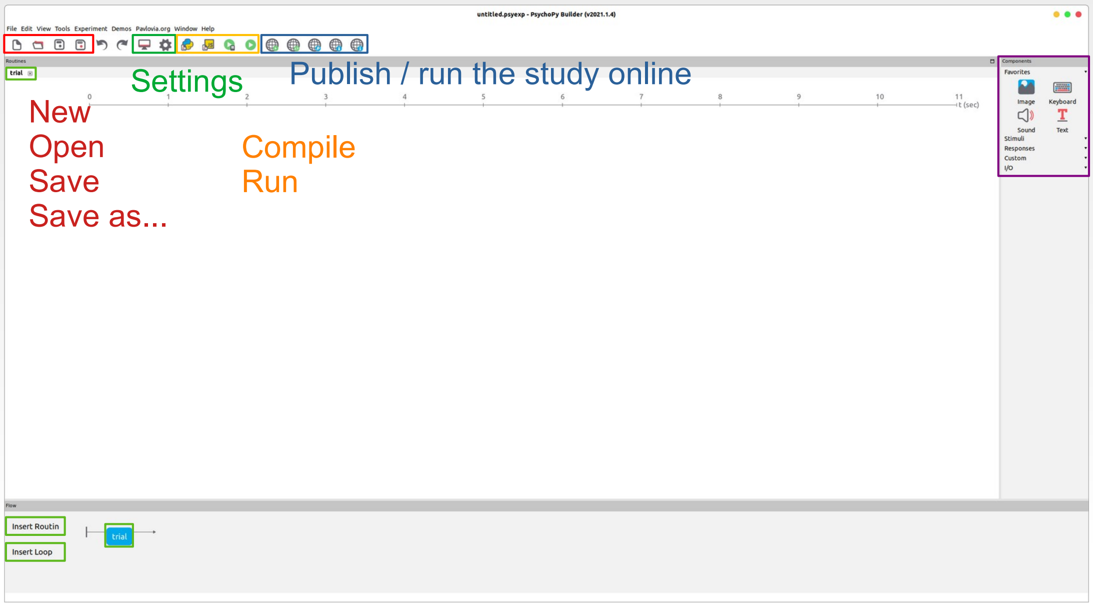
The Runner is something like a central control panel from where you start experiments (if you run them on your local computer) and possibly convert them into web experiment in order to upload and run them on Pavlovia.

The Builder is the part we will spent most of our time with today. This is where you assemble your experiments in a graphical user interface (GUI). All the experiments you create will be automatically converted into Python-code in order to run them.

The Coder permits you to make adjustments to this Python-code. If you are willing to dive into that, you can adapt your experiments to nearly every requirement you might have (though, the learning curve is much steeper than for using the GUI).

UNIVERSITY OF BERGEN

PsychoPy



Settings

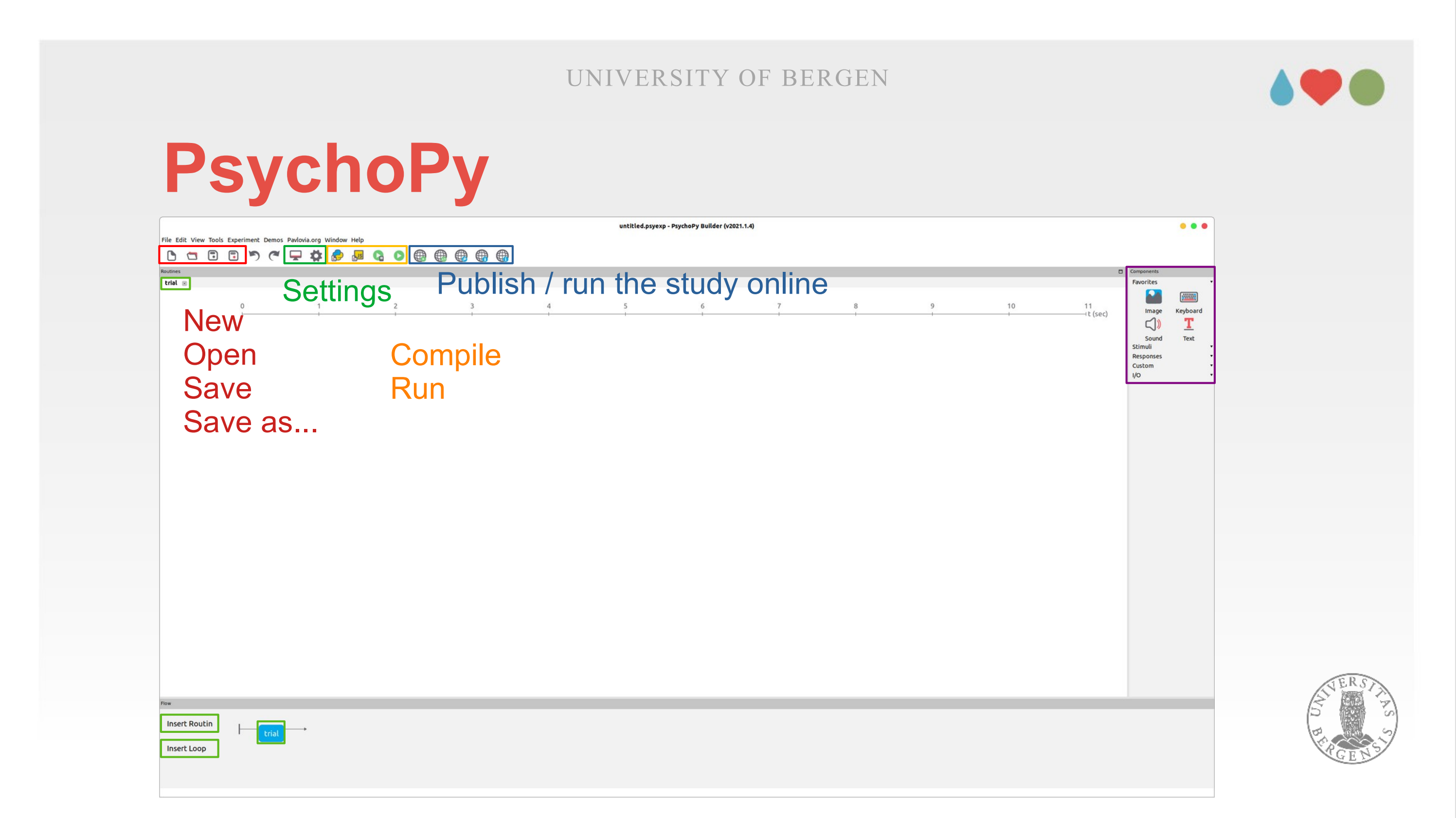
Publish / run the study online

New
Open
Save
Save as...

Compile
Run

As I said, we will spend most of the time of todays lecture in the Builder. If it’s not open press the menu “View” in Coder or Runner. The Builder consists of five parts:

1) The menu bar gives you access to all functions, including file operations, creating new routines, access to example experiments, etc.
2) The icon bar gives quick access to the main functions. The are colour-coded on this slide. The first four icons marked in red are the typical file operations (New, Open, Save, Save as). The second set marked in green gives you access to settings, with the first you adjust for your monitor, with the second you can define specific settings for your experiment. The third group, marked in orange, converts your experiments to Python (so that you can adapt them in the Coder. The fourth group marked in blue permits sending your experiment to Pavlovia to run it in the web.



UNIVERSITY OF BERGEN

PsychoPy

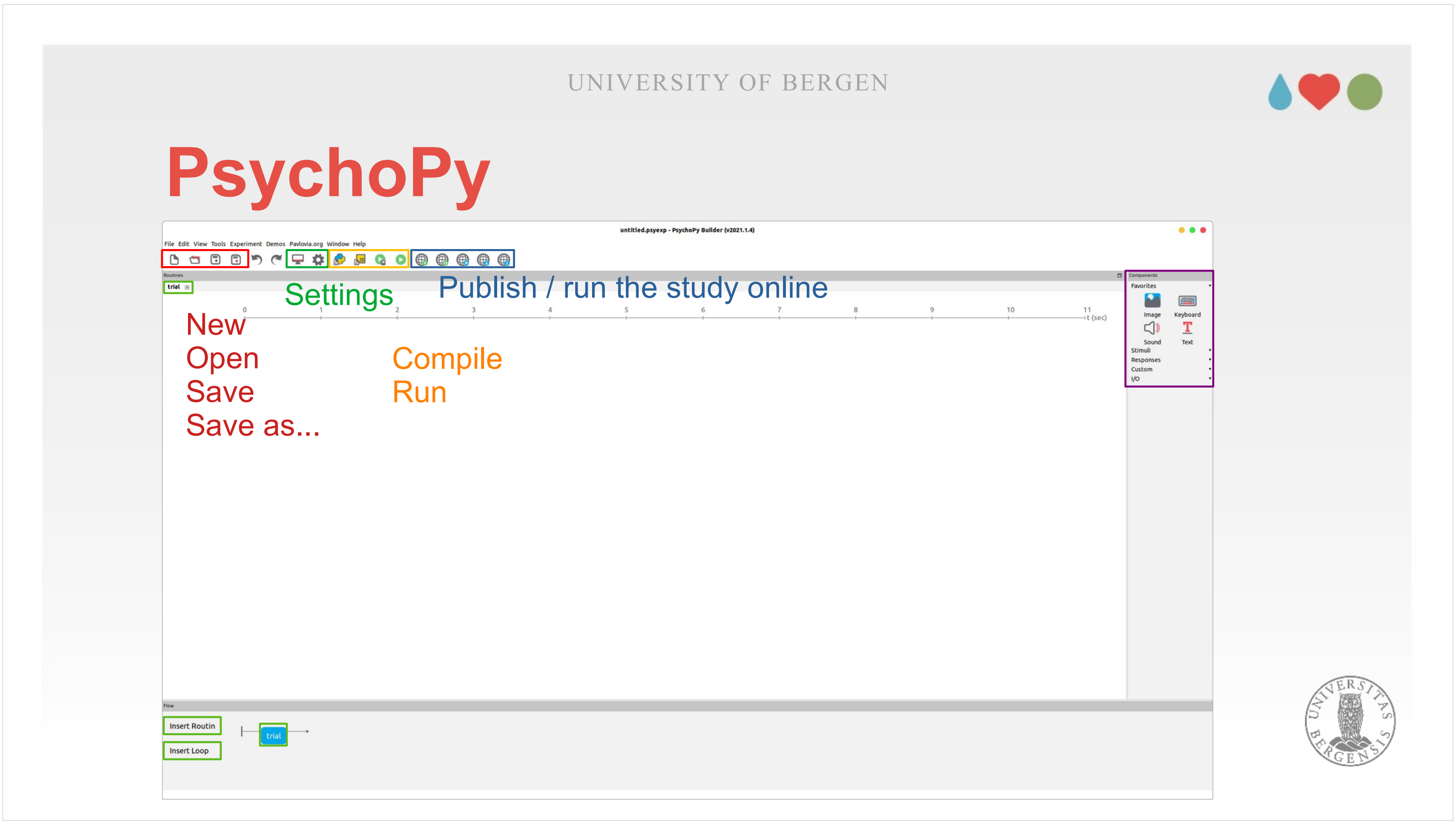
Settings

Publish / run the study online

New
Open
Save
Save as...

Compile
Run

3) Underneath the icon bar – the part which takes most of the screen is where you assemble your routines. A routine is something like a step within your experiment (e.g., a screen showing instructions). You see a timeline showing you when certain components appears and for how long. A component might be a text shown on the screen, waiting for a user response via keyboard or mouse, etc. For example, you might define that a question from your questionnaire is shown for 0.5 seconds before the user can react with a button press. You will see a tab where „trial" is written. Each routine has such a name which can also be found in the other time line at the bottom of the screen.



UNIVERSITY OF BERGEN

PsychoPy

Settings

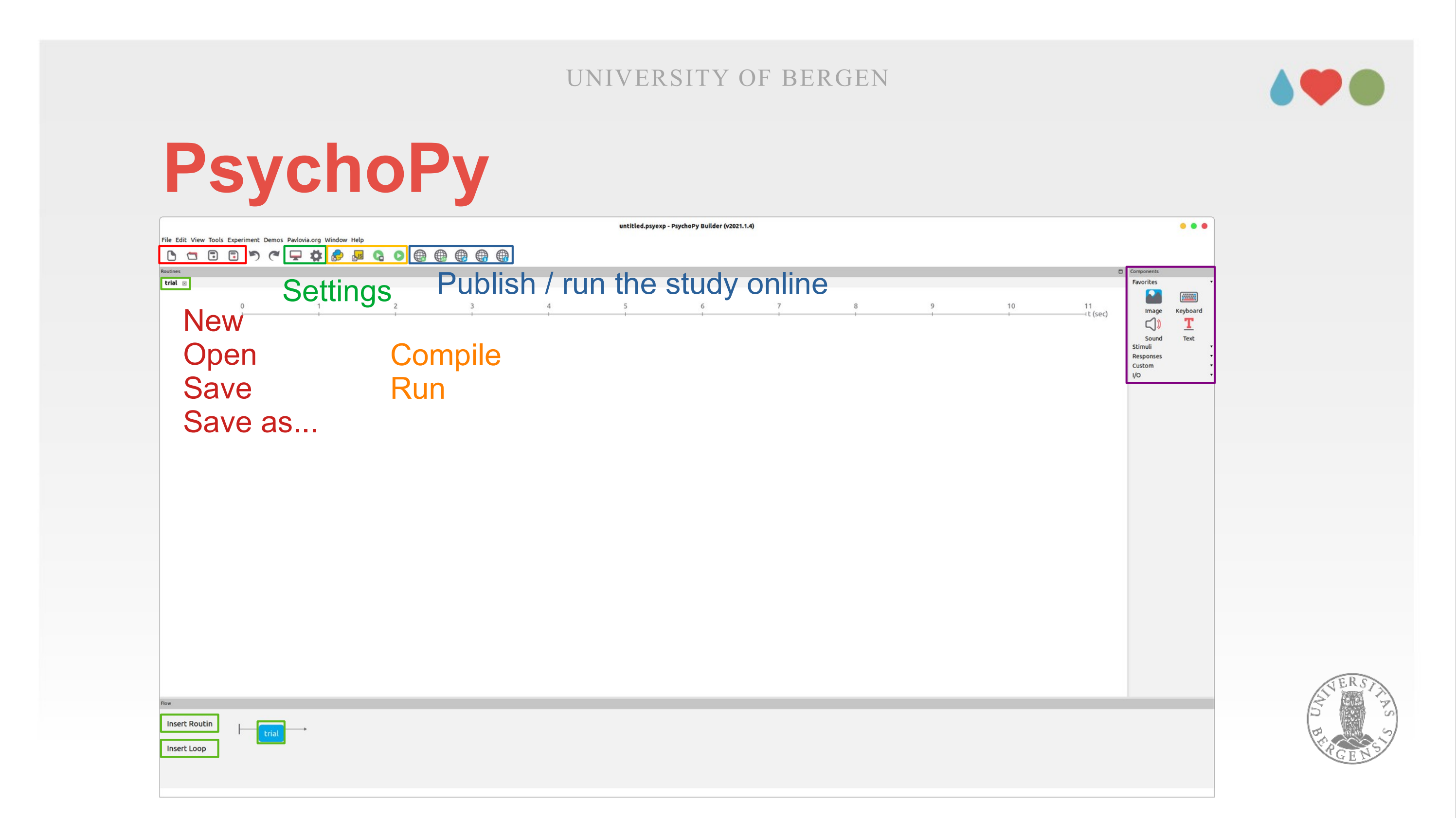
Publish / run the study online

New
Open
Save
Save as...

Compile
Run

4) In the part at the bottom you can create, arrange, and edit your routines. In an experiment, you typically would start with one (or more) routines for welcome, showing instructions, etc. This is followed by some routines (typically in a loop) where you actually run your experiment (present stimuli, get responses, evaluate them and give feedback, etc.). Finally, you would have some Thank you-routine at the very end (acknowledging the participation).

I mentioned loops which are very handy for repeating things. Let‘s assume that you are running a questionnaire. The basic setup is always the same: showing a question, getting a response. However, it is easier if you don’t have to create new routines for each question but if the the questions could by dynamic. That is, you prepare an Excel-file with all questions and PsychoPy presents them one after another.



UNIVERSITY OF BERGEN

PsychoPy

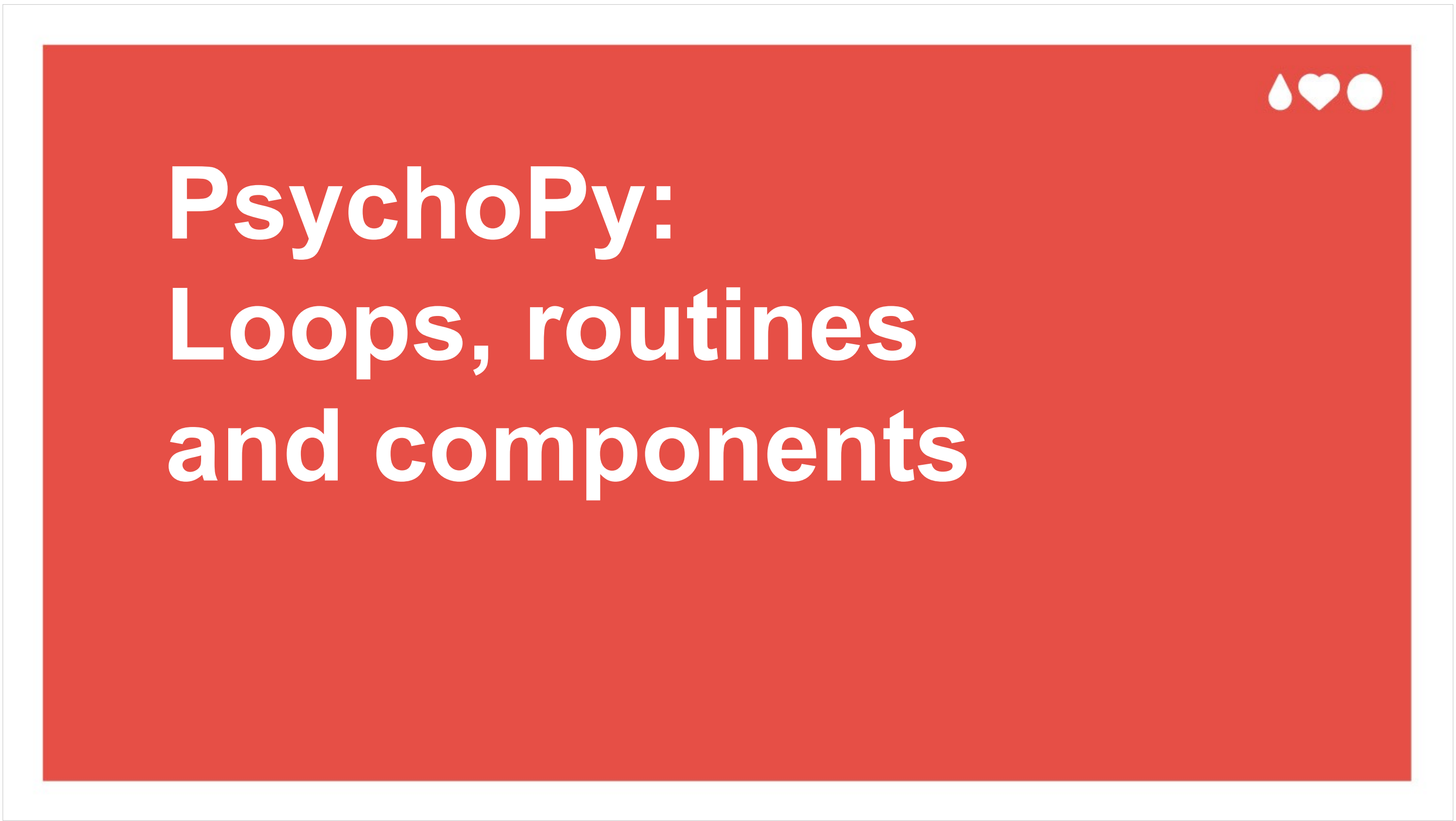
New
Open
Save
Save as...

Settings

Compile
Run

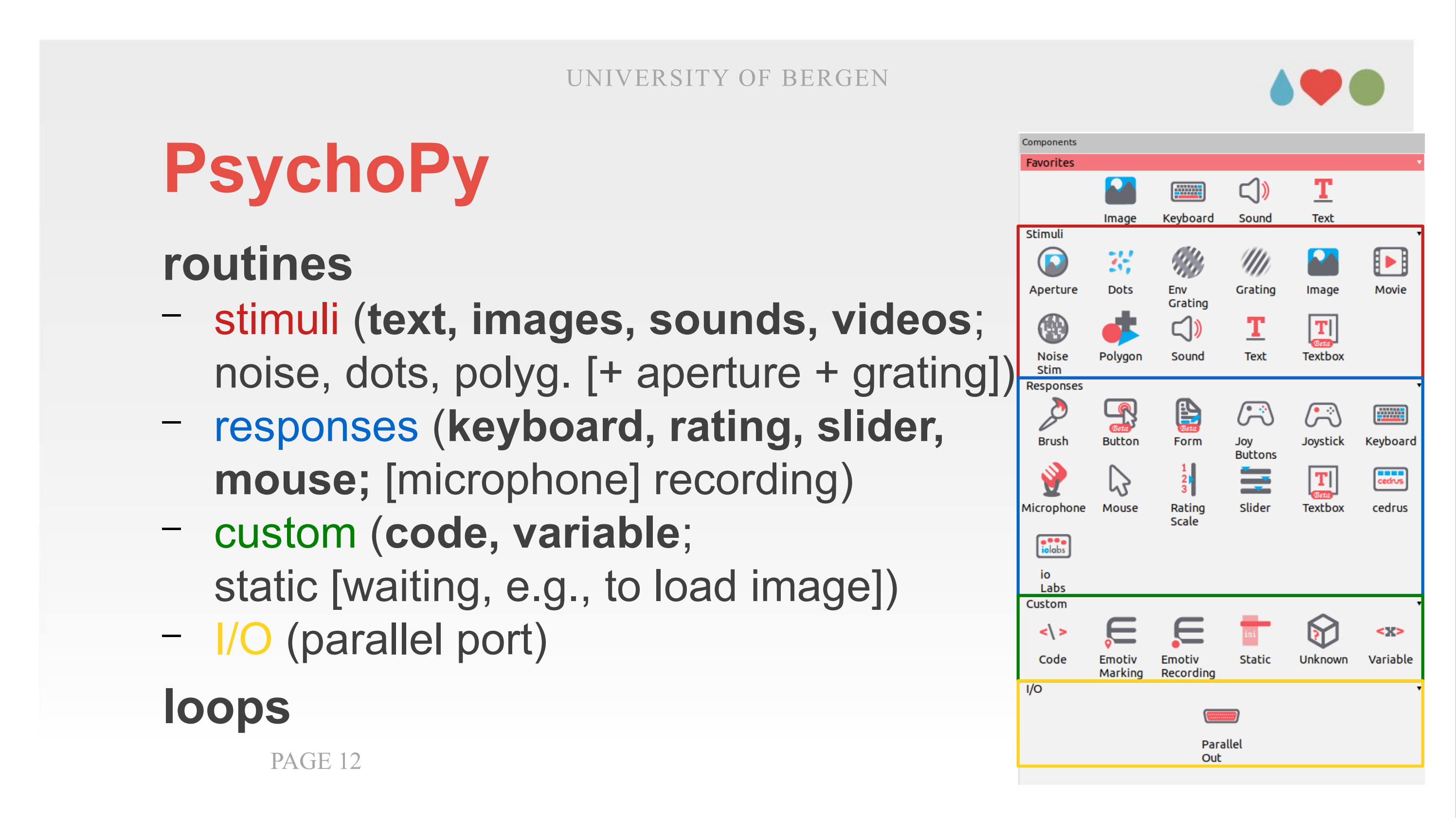
Publish / run the study online

5) Finally, we have a toolbox on the right-hand side with the components which we can use in our routines (marked in violet). You can see the four most frequently used components in the favourites (asking for a response via keyboard, presenting text, presenting images, and presenting sound files). The other components are “hidden” in the dropdown-menus for “Stimuli”, “Responses”, “Custom”, “EEG”, “Eyetracking” and “I/O”. We will have a more detailed overview of these components on the next slide.



PsychoPy: Loops, routines and components

The next section covers the different elements that you need to create experiments. Using loops you can repeat things (e.g., individual items in a questionnaire or experimental conditions). Routines are the individual steps within an experiment. Those routines can either be run individually (e.g., for presenting instructions) or within loops (e.g., to present questionnaire items). Within those routines, you have different components that permit you either to present the participant with text, images, sounds or videos or to collect responses from them.



UNIVERSITY OF BERGEN

PsychoPy

routines

- stimuli (**text, images, sounds, videos**; noise, dots, polyg. [+ aperture + grating])
- responses (**keyboard, rating, slider, mouse;** [microphone] recording)
- custom (**code, variable**; static [waiting, e.g., to load image])
- I/O (parallel port)

loops

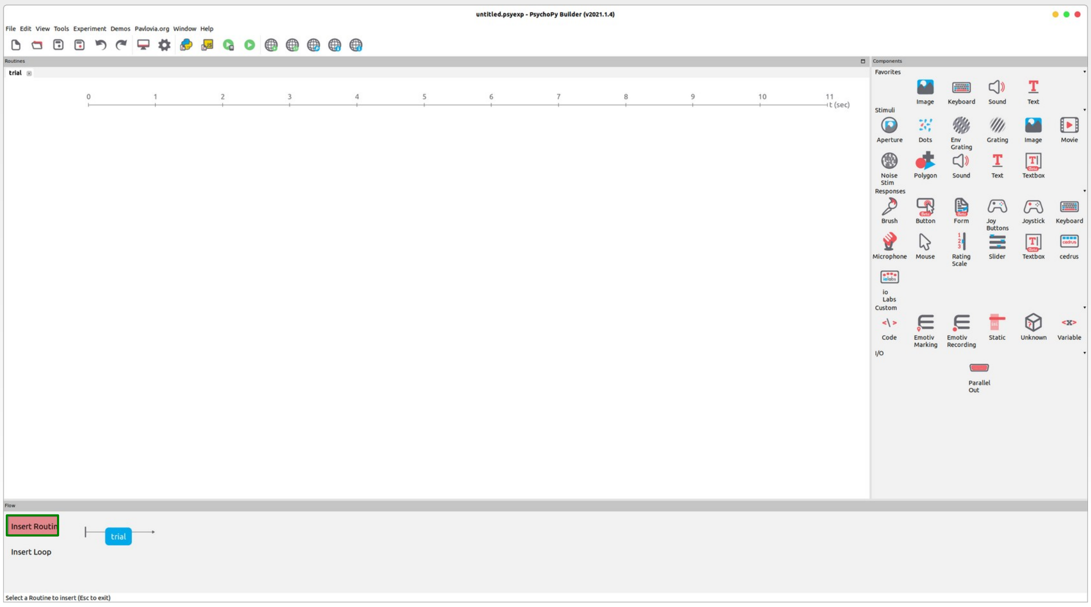
I briefly mentioned that PsychoPy experiments are organized in routines and loops. Routines are the main building block since these are the “interfaces” with your participant, i.e., where you present something to the participants and then ask for and register their feedback.

You use loops to repeat these routines (note that a loop can contain several routines; e.g., a fixation cross, followed by presenting an image, and a routine for getting feedback on that image).

The components are organized into six categories: Stimuli, Responses, Custom, EEG, Eyetracking and I/O. The most frequently used of them are marked with bold. Most of them will be introduced, one after another, over the next couple of slides.

UNIVERSITY OF BERGEN

PsychoPy

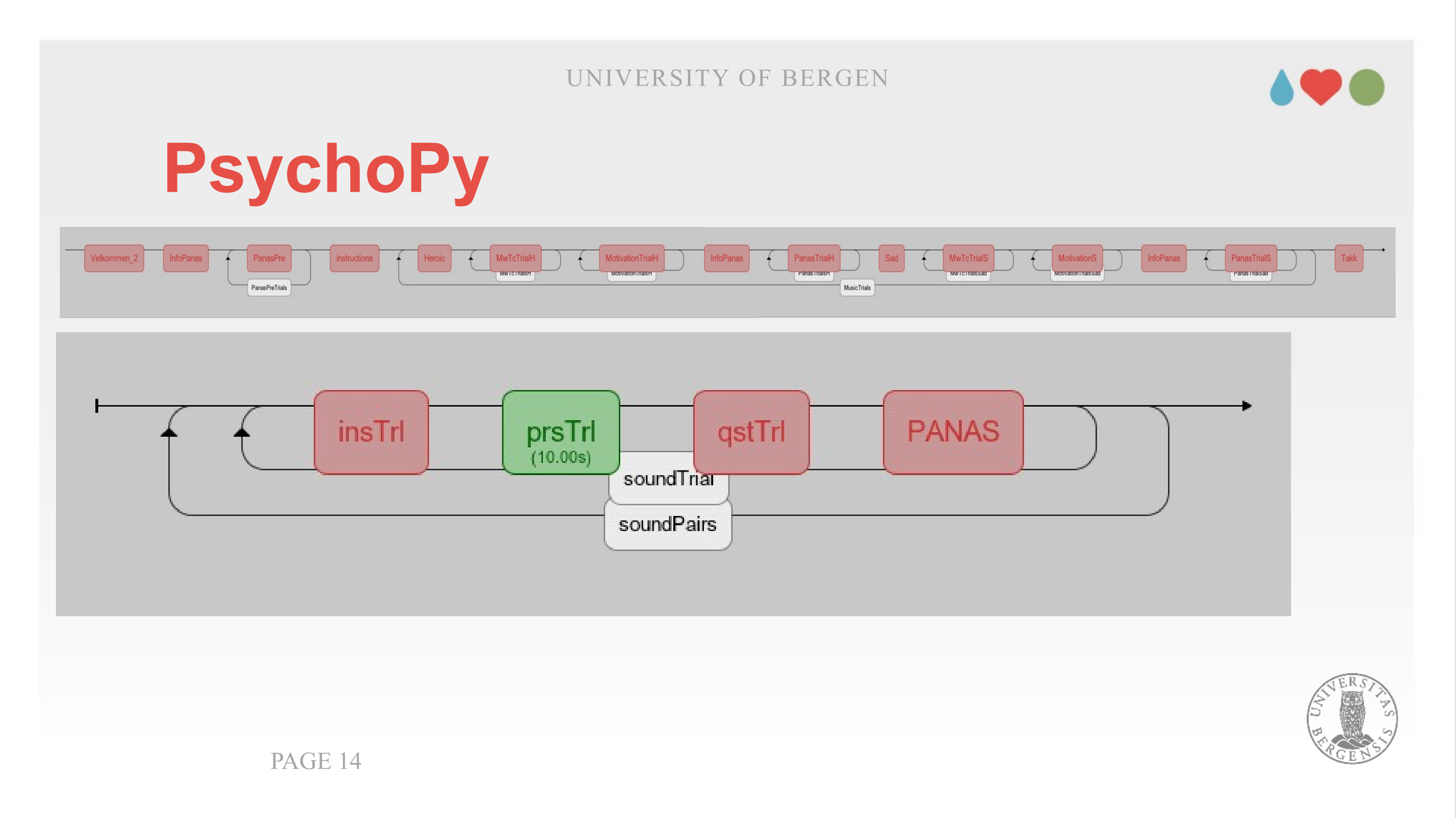


Select a Routine to insert (Esc to exit)

But before we do that, we have to do one step back and have a look at how to create routines and loop.

At the bottom of the PsychoPy Builder window will you find a part that is denoted “Flow”. This part also has a time line, this time showing how your routines and loops are arranged in your experiment.

Now, we have only one routine “trial” – this is the default when we open the Builder. Using the two button on the left hand side: Insert Routine and Insert Loop you can add Routines or Loops to your experiment.



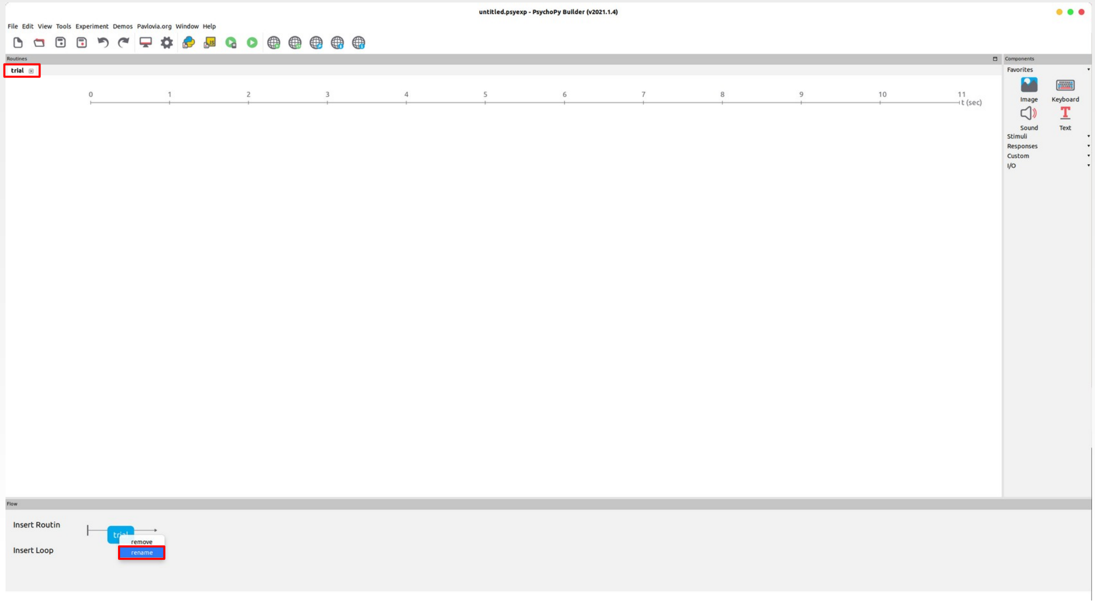
PsychoPy

This slide shows two examples for Flows from experiments. I mentioned before that loops can help when certain tasks are repeated and only slightly changed. You can take this further by nesting loops like in the example in the bottom.

The two Flows are actually the same experiments. In the first case (top), the experiment is rather organized as a long sequence of Routines. In the second example (bottom) is is organized as loops. This is certainly more efficient but, especially if the loops are nested, also more challenging (as it might require you to write some Python-code and it is more difficult to find the source of errors if you encounter them). For beginners, the sequential solution is maybe easier.

UNIVERSITY OF BERGEN

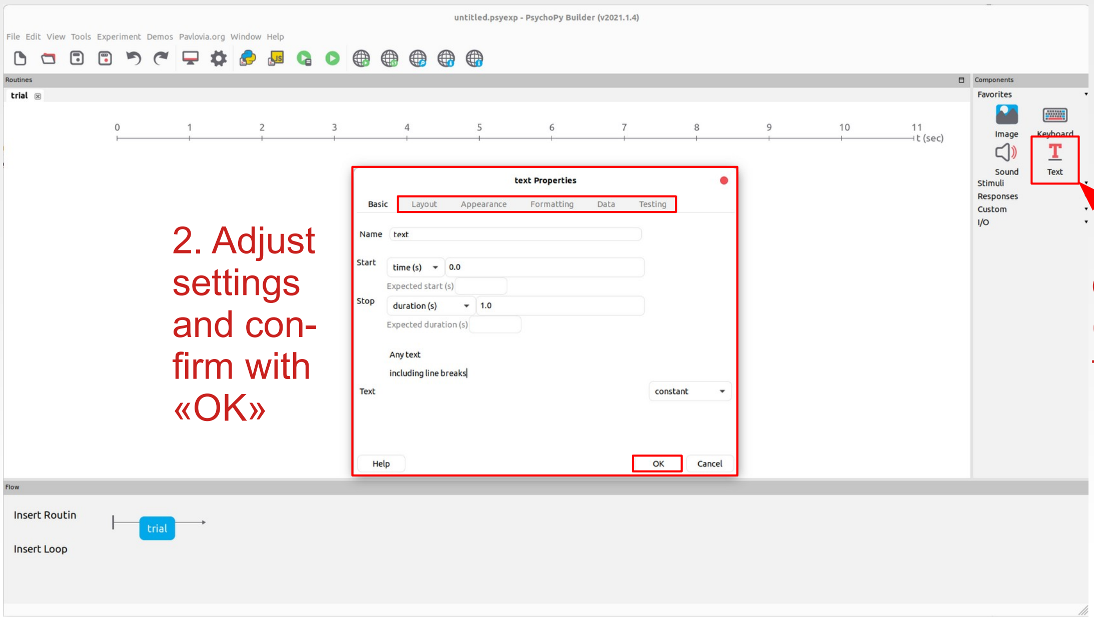
PsychoPy



Another opportunity which you can do in the “Flow” is to rename, delete, and move around your routines. One thing is important to mention. Routines you generated can be “reused”, e.g., you can use the same routine presenting a fixation cross several times. But please note that you have to create a new routine if you want to have even tiny changes.

UNIVERSITY OF BERGEN

PsychoPy: Components



text Properties

Basic | Layout | Appearance | Formatting | Data | Testing

Name: text

Start: time (s) 0.0

Expected start (s)

Stop: duration (s) 1.0

Expected duration (s)

Text: Any text
including line breaks
constant

2. Adjust settings and con-firm with «OK»

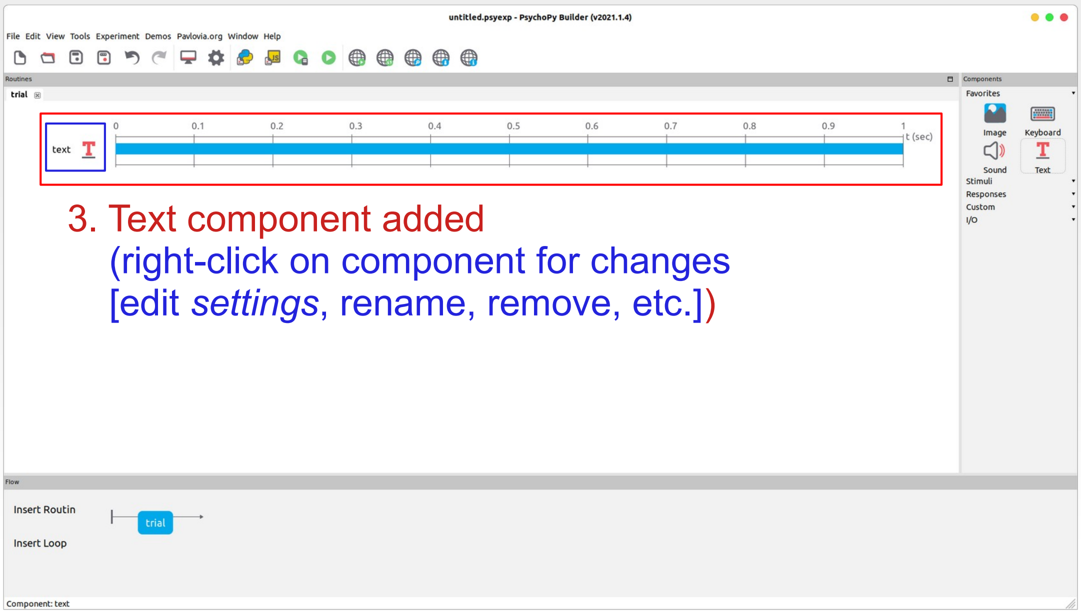
1. Click on component (e.g., Text) to add it

Within routines, you can create new components by clicking on the respective icon under “Components” on the right-hand side. In the example on the slide a “Text”-component is created.

This opens a new window where you can see and adjust the properties of that component. This allows you, e.g., to set when the component appears (Start), how long it is shown (Stop / duration), and then several properties that are specific to a component (e.g., which font to use for the Text-component, which text size and text colour).

There are several tabs where you can set further properties. The one that you see first “Basic”, contains the properties that are elementary. In addition, there are typically “Data” and “Testing”, and for all components that are placed on the screen (text, images) “Layout” and “Appearance”.

PsychoPy: Components

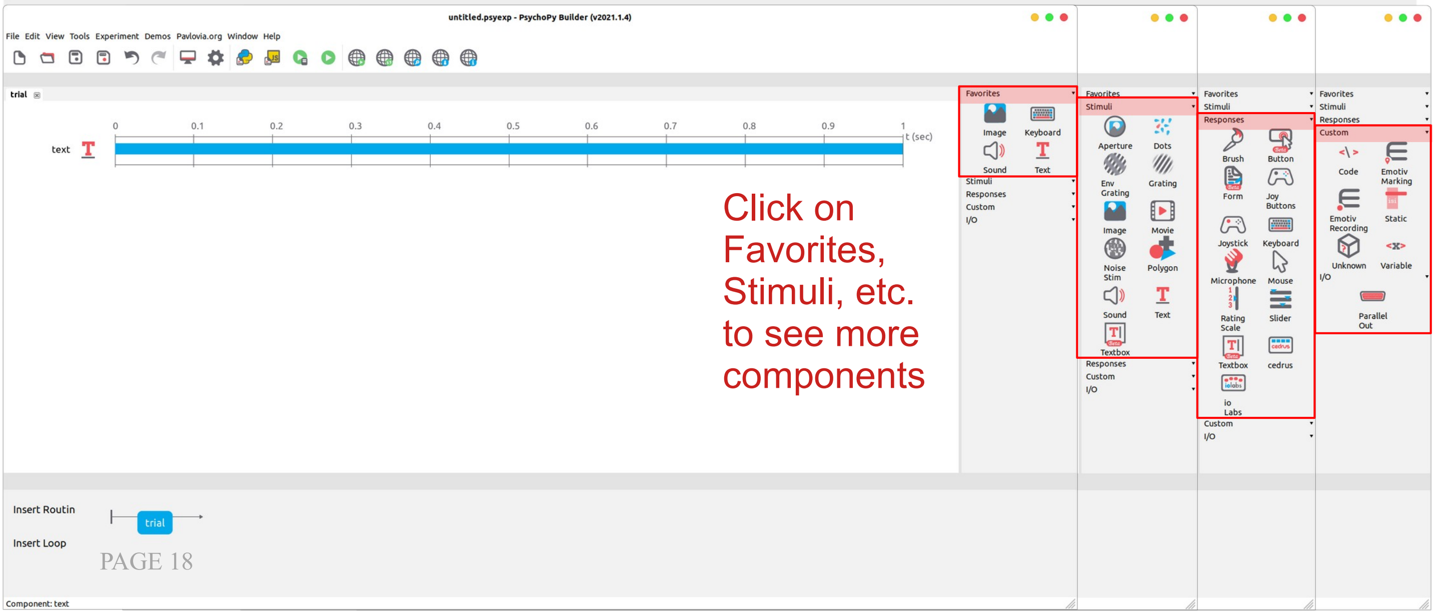


3. Text component added
(right-click on component for changes
[edit *settings*, rename, remove, etc.])

Once you added a component can you copy, edit (this opens the property-window again) or remove it.

You also can move it up and down in the list. The latter (moving) doesn’t really have much consequences apart from that it helps you to to visually organize the components within your routines.

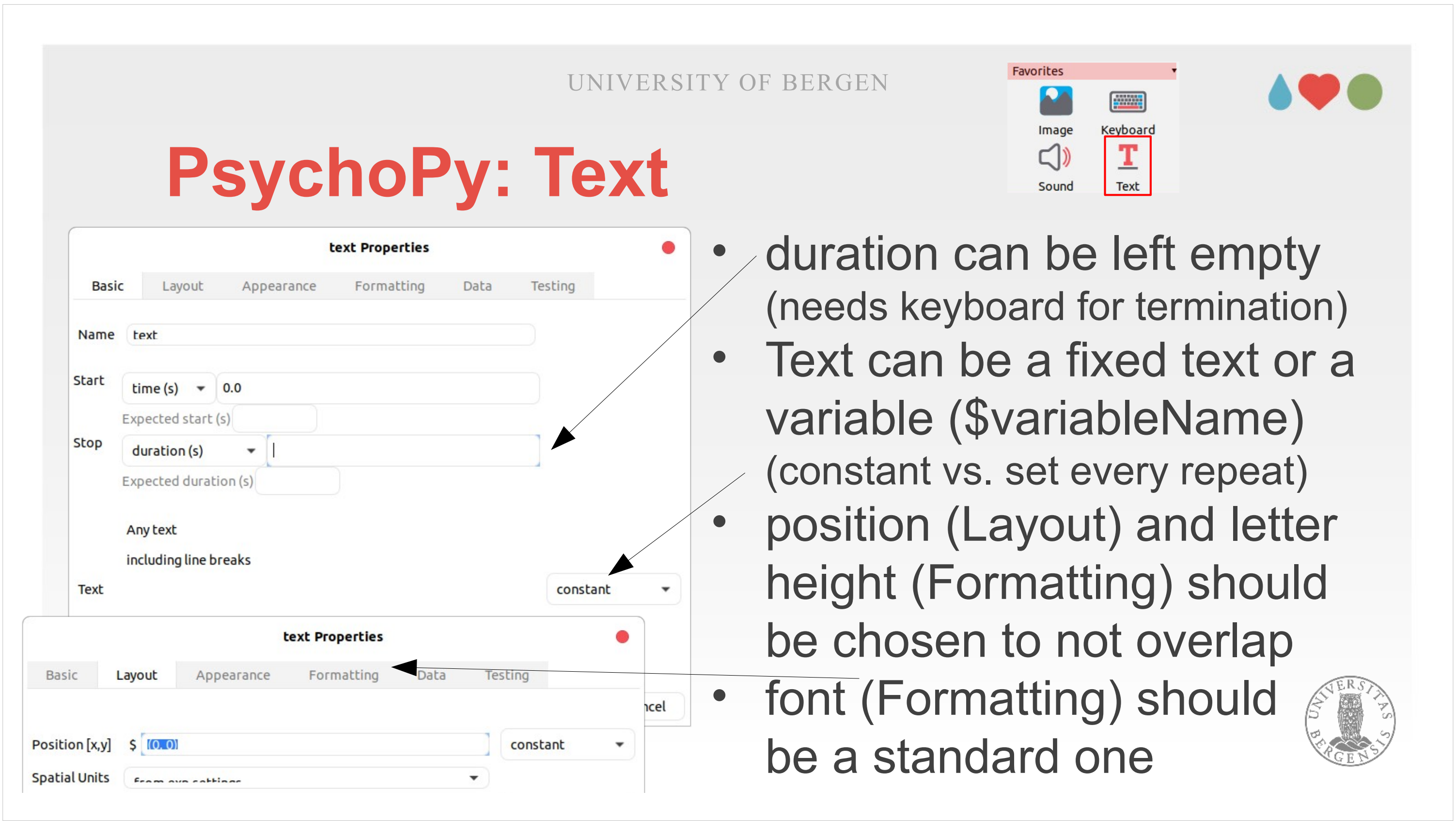
PsychoPy: Components



Click on Favorites, Stimuli, etc. to see more components

Within the comonents on the right-hand side, you have dropdown-menus that you can open to select your components. You could see all of them unfolded on slide 9. As I said there, the most commonly used can be found all the time under “Favourites”, and of course you can add those you would like to have in “Favourites” by right-clicking on them and choosing “Add to favourites”. “Stimuli” and “Responses” contain further, less often used component.

The most common components are described in more detail on the next slides.



UNIVERSITY OF BERGEN

PsychoPy: Text

Name: text

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

Text: Any text

including line breaks

constant

text Properties (Layout)

Position [x,y] $ (0, 0) constant

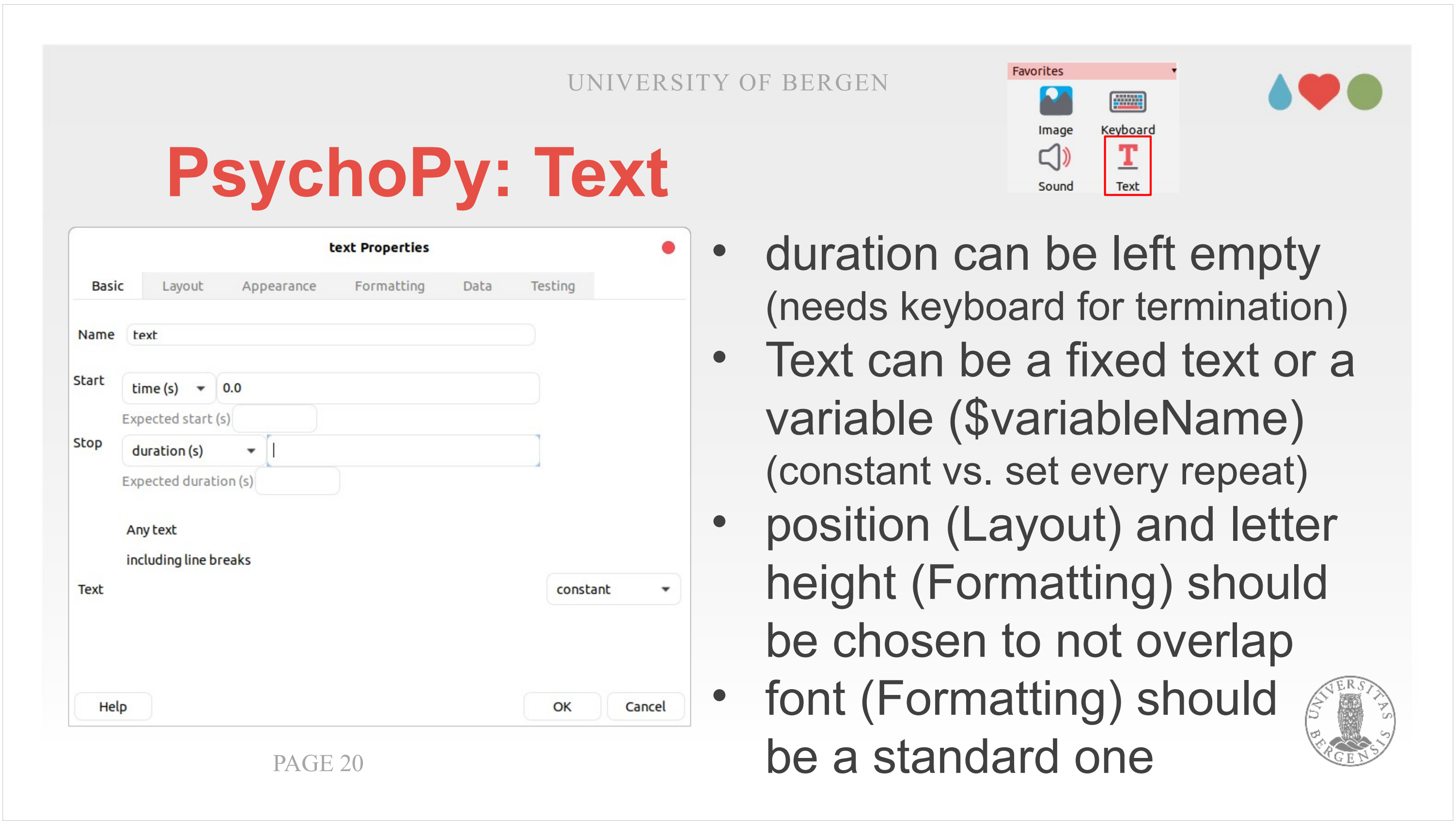
Spatial Units: from exp settings

- duration can be left empty (needs keyboard for termination)
- Text can be a fixed text or a variable ($variableName) (constant vs. set every repeat)
- position (Layout) and letter height (Formatting) should be chosen to not overlap
- font (Formatting) should be a standard one

The one that you might use most often is the text component. It can be used to present instructions, fixation crosses (“+”), text art (if you are into that), etc.

One general remark about how the screen is handled in PsychoPy: All is centered. That means a couple of things: The position (0, 0) is in the middle of the screen, changing the first coordinate (**0**, 0) moves your text (or any other visual stimulus) on the left-right-axis, changing the second coordinate (0, **0**) moves your stimulus on the up-down-axis.

The range is -0.5 (left, down) to 0.5 (right, up). Size is also relative to the screen (0.1 means that the component occupies about 0.1 = 10% of the screen). This makes the size independent of the screen resolution and helps when taking an experiment from one computer to another which might have a different resolution.



UNIVERSITY OF BERGEN

PsychoPy: Text

text Properties

Basic | Layout | Appearance | Formatting | Data | Testing

Name: text

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

Text: Any text
including line breaks

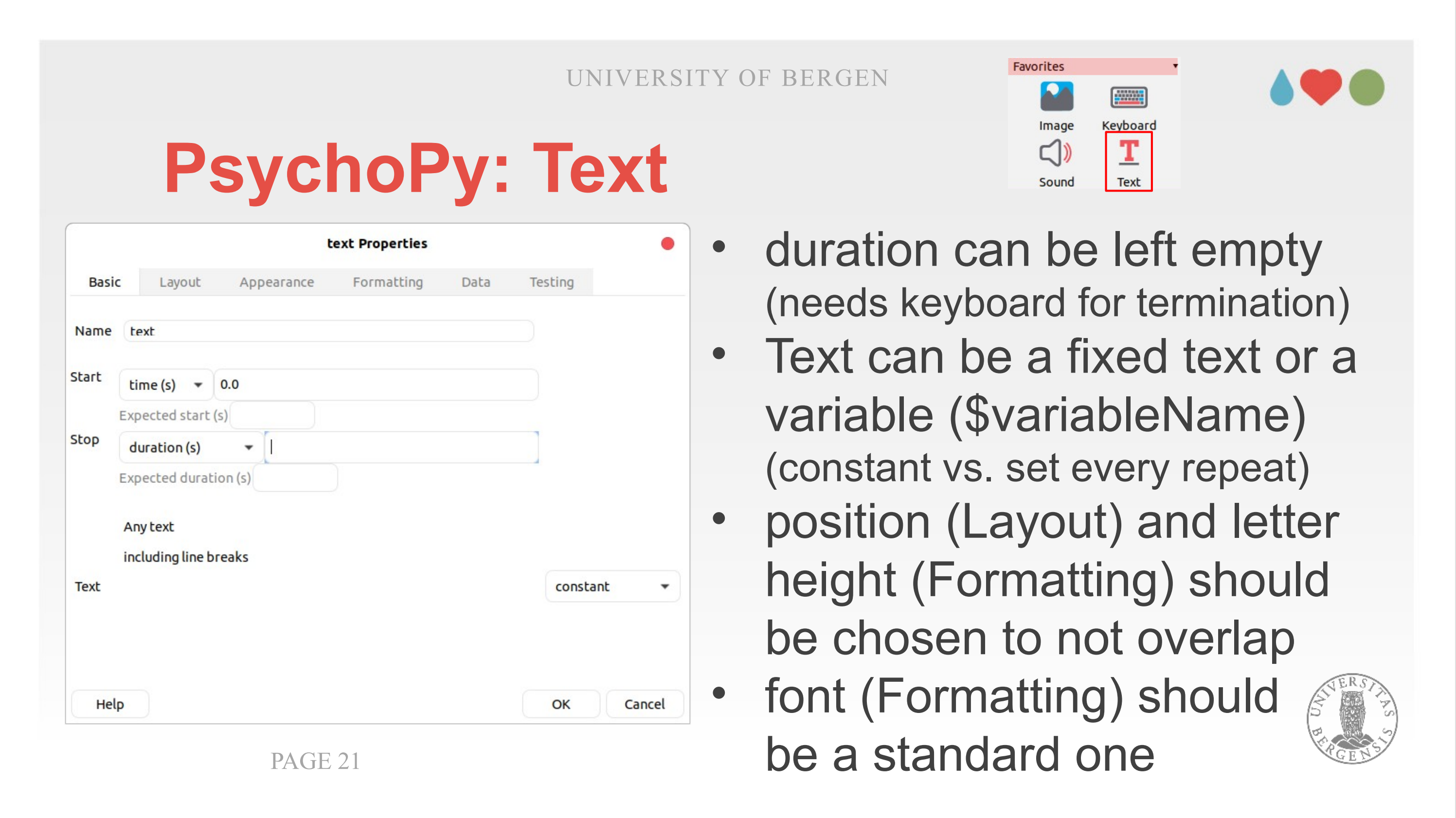
constant

- duration can be left empty (needs keyboard for termination)
- Text can be a fixed text or a variable ($variableName) (constant vs. set every repeat)
- position (Layout) and letter height (Formatting) should be chosen to not overlap
- font (Formatting) should be a standard one

The most important bit is what you enter under Text. You can enter several lines of text there. Keep in mind that each line represents one “piece” of screen size, i.e., if you set size to 0.1 then you can put maximally 10 lines on the screen.

“Color” and “Font” should be self-explanatory. For longer texts, a font with serifes (e.g., Times New Roman) might be easier to read.

Start and Stop / duration where already mentioned. You could show the text for, e.g., 10 seconds if you put 10 in “Duration”. If you leave duration blank, then your text is shown until an event. Typically, what you do it to finish the routine when a response is given.



UNIVERSITY OF BERGEN

PsychoPy: Text

text Properties

Name text

Start time (s) 0.0

Expected start (s)

Stop duration (s)

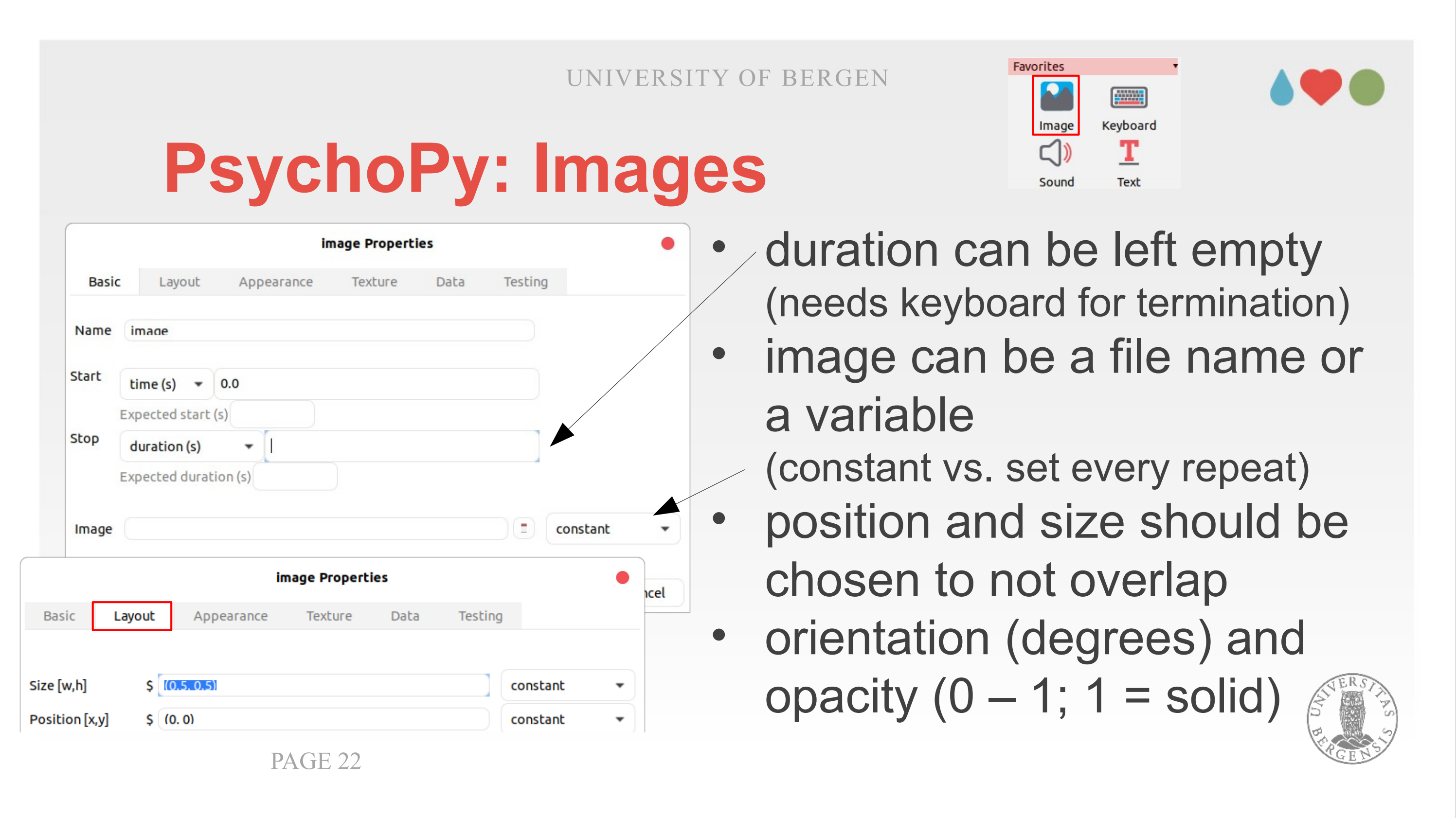
Expected duration (s)

Text
Any text
including line breaks
constant

- duration can be left empty (needs keyboard for termination)
- Text can be a fixed text or a variable ($variableName) (constant vs. set every repeat)
- position (Layout) and letter height (Formatting) should be chosen to not overlap
- font (Formatting) should be a standard one

In addition, you could have two pieces of instructions (note that you need two Text components for that). One, presenting a question or instructions, is shown immediately, another one, showing response options or how to proceed, after a bit of delay (0.5 sec). After the delay, you would also permit inputs from the keyboard. In order to set such delays, you have to enter 0.5 under Start (for both the Text how to proceed as well as for the Keyboard component).

This is “nudging” the participants to read the instructions (since they can’t go faster than the set time).



PsychoPy: Images

Name: image

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

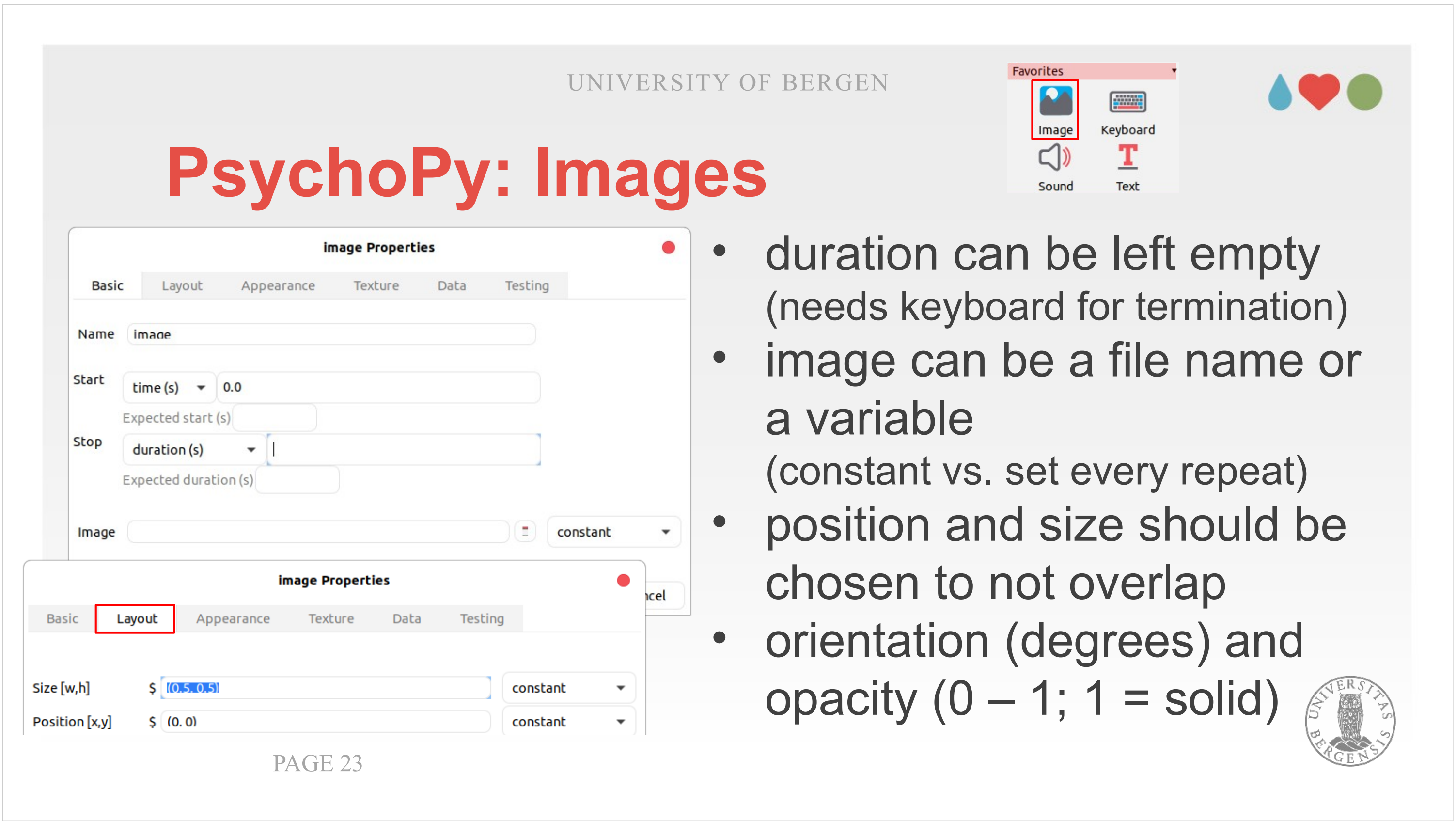
Image: constant

Size [w,h]: $ [0.5, 0.5] constant

Position [x,y]: $ (0, 0) constant

- duration can be left empty (needs keyboard for termination)
- image can be a file name or a variable (constant vs. set every repeat)
- position and size should be chosen to not overlap
- orientation (degrees) and opacity (0 – 1; 1 = solid)

For positioning images on the screen, the same that I said on the previous slides applies. In the example shown on the slide, the picture is horizontally positioned in the centre and vertically a little (20%) up. The size indicates that it is an image in portrait format (i.e., the extension left-right is smaller [0.4] than the up-down extension [0.6]). Position and size should be chosen such that the elements on the screen don‘t overlap and hinder each other. You wouldn‘t want that you can‘t read the instructions anymore because an image is hiding a part of them.



PsychoPy: Images

Name: image

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

Image: constant

Size [w,h] $ [0.5, 0.5] constant

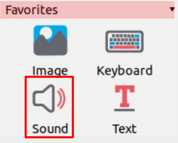
Position [x,y] $ (0, 0) constant

- duration can be left empty (needs keyboard for termination)
- image can be a file name or a variable (constant vs. set every repeat)
- position and size should be chosen to not overlap
- orientation (degrees) and opacity (0 – 1; 1 = solid)

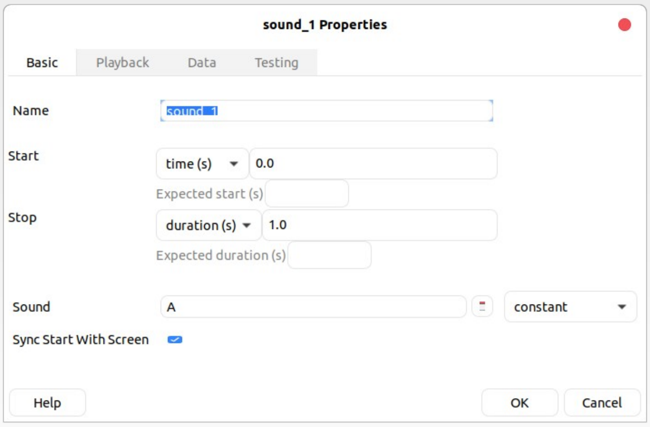
You will see $image in the Image-field. This means that the image is set dynamically. I will later say more on loops and where we use them. In principle, you present your stimuli dynamically by putting a list with file names in a CSV or Excel-file. PsychoPy reads from that. If you use variables ($image), the value in the drop-down box next to the variable has to be changed from „constant“ to „set every repeat“.

Finally, there are two parameter that determine how your image looks like. Orientation presents an image tilted at a certain angle (e.g., 45°).

The other parameter is opacity. You could use this to make one picture „appear“ by increasing opacity or overlay two pictures by increasing the opacity in one and decreasing it in another picture.



PsychoPy: Sounds



sound_1 Properties

Name sound_1

Start time (s) 0.0

Expected start (s)

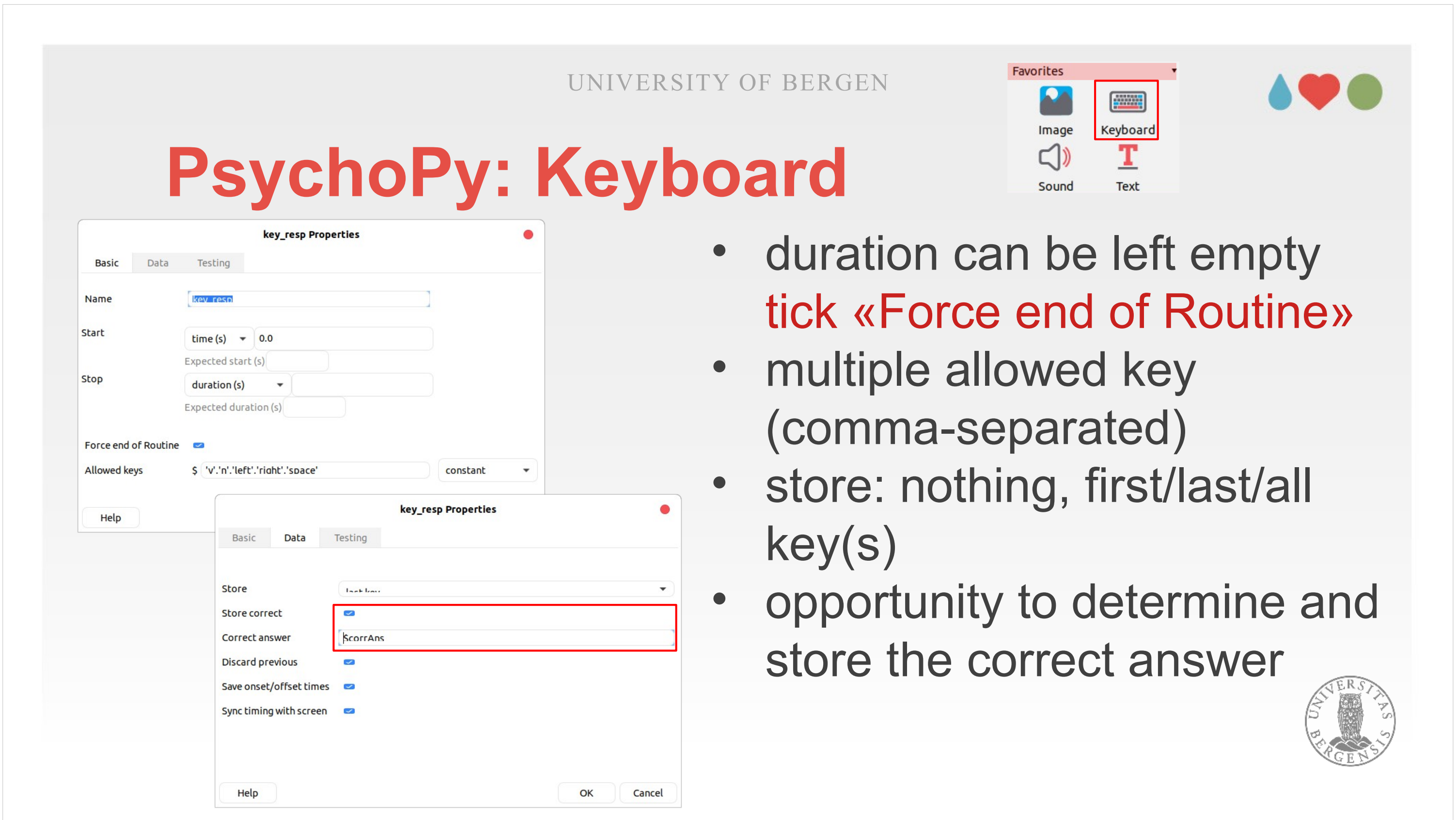
Stop duration (s) 1.0

Expected duration (s)

Sound A constant

Sync Start With Screen ☑

- if duration is left empty, the sound is played indefinitely or as long as the sound file is
- sound can be a tone name (C, D, …), a frequency (Hz), a file name or a variable (containing either of these three options)
(constant vs. set every repeat)



UNIVERSITY OF BERGEN

PsychoPy: Keyboard

key_resp Properties

Basic | Data | Testing

Name: key_resp

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

Force end of Routine ☑

Allowed keys: $ 'v','n','left','right','space' constant

Help

key_resp Properties

Basic | Data | Testing

Store: last key

Store correct ☑

Correct answer: $corrAns

Discard previous ☑

Save onset/offset times ☑

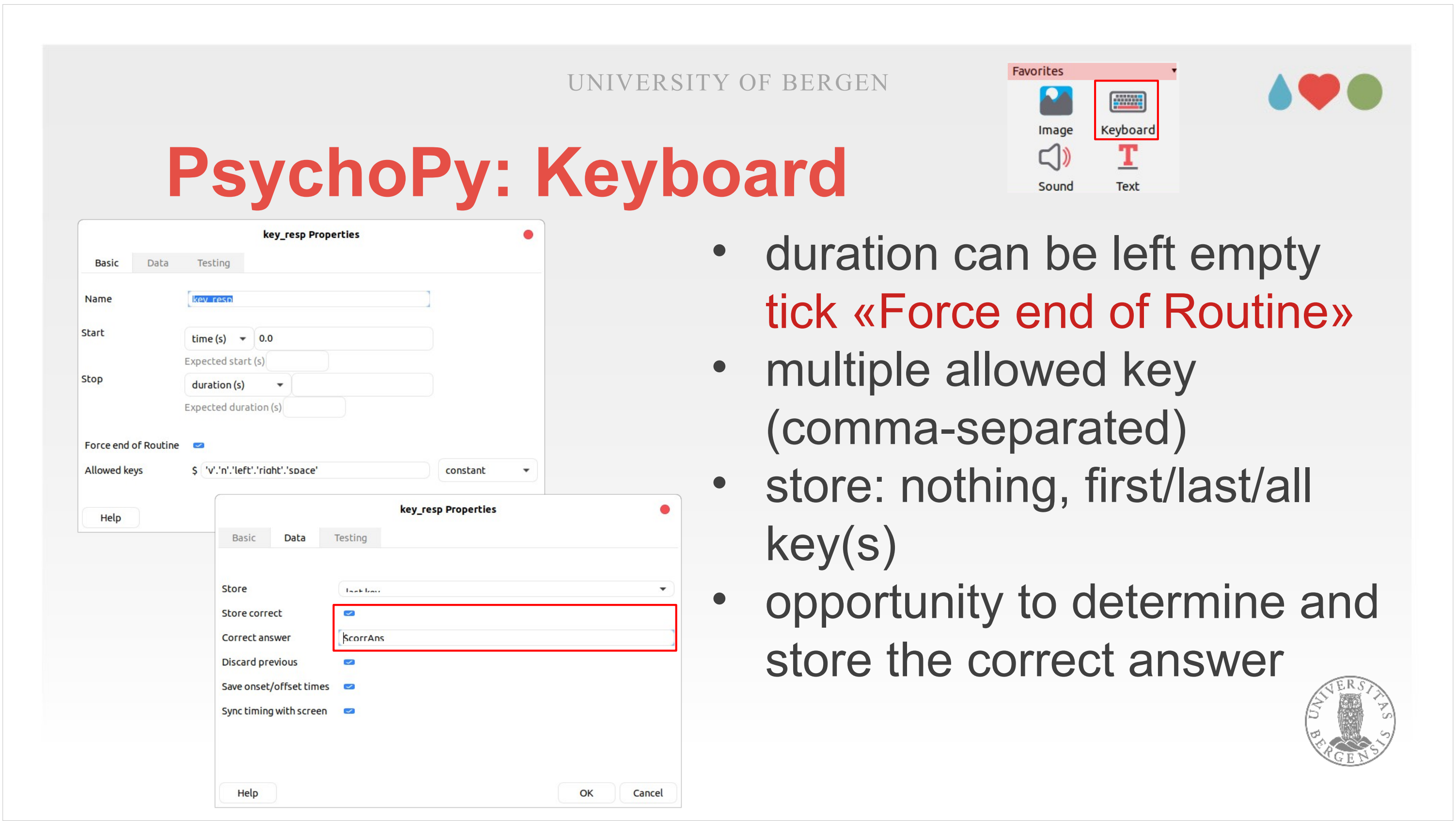
Sync timing with screen ☑

Help OK Cancel

- duration can be left empty
  tick «Force end of Routine»
- multiple allowed key (comma-separated)
- store: nothing, first/last/all key(s)
- opportunity to determine and store the correct answer

Another commonly used component is the keyboard. As I said, I often “nudge” participants by delaying the opportunity to respond a little bit. This would be done with setting “Start” to 0.5 (sec). Typically, you wouldn’t want to set a duration, unless it is a timed task where participants have to react quick and to reply within a certain period of time.

For questionnaires etc., it is more appropriate to leave it blank and to tick “Force end of routine”. This finishes the current routine / trial and goes on to the next one when a button is pressed. A quite common use case is: Participants are shown one (or more) stimuli, their response is collected and then the trial is finished.



UNIVERSITY OF BERGEN

PsychoPy: Keyboard

key_resp Properties

Basic | Data | Testing

Name: key_resp

Start: time (s) 0.0

Expected start (s)

Stop: duration (s)

Expected duration (s)

Force end of Routine ☑

Allowed keys $ 'v','n','left','right','space' constant

Help

key_resp Properties

Basic | Data | Testing

Store: last key

Store correct ☑

Correct answer: $corrAns

Discard previous ☑

Save onset/offset times ☑

Sync timing with screen ☑

Help OK Cancel

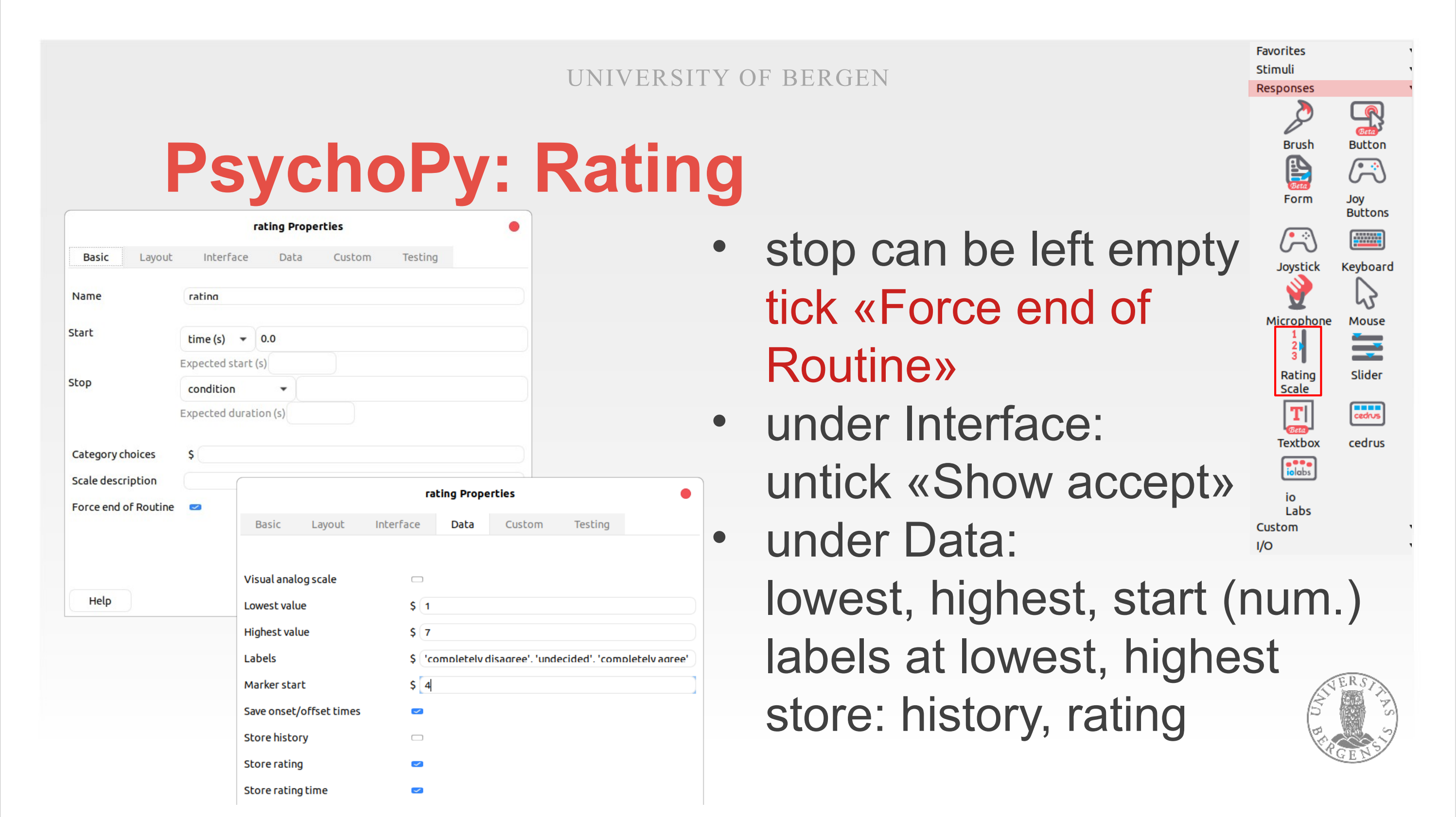
- duration can be left empty
  tick «Force end of Routine»
- multiple allowed key (comma-separated)
- store: nothing, first/last/all key(s)
- opportunity to determine and store the correct answer

In the field “Allowed keys” you define which keys are permitted for responding. “Space” is often a good choice for proceeding. For questionnaires, responding with the numbers is maybe most natural.

„Store“ depends on (of course) whether you are interested in the response (or possibly reaction times associated with it) or not.Typically, you wouldn‘t be interested in the keys that indicate somebody read the instructions and wants to proceed. Here you would choose to store „nothing“.

The other three options depend a little bit on which duration you set. If the routine was set to end with the button press („Force end of routine“), it doesn‘t matter which of those options is chosen.

Otherwise, if you have a certain duration during which you would like to register respones, you have to choose whether you would like to keep only the first or the last response or all.



UNIVERSITY OF BERGEN

PsychoPy: Rating

rating Properties

Basic | Layout | Interface | Data | Custom | Testing

Name: rating

Start: time (s) 0.0

Expected start (s)

Stop: condition

Expected duration (s)

Category choices: $

Scale description

Force end of Routine ☑

Help

rating Properties

Basic | Layout | Interface | Data | Custom | Testing

Visual analog scale □

Lowest value: $ 1

Highest value: $ 7

Labels: $ 'completely disagree', 'undecided', 'completely agree'

Marker start: $ 4

Save onset/offset times ☑

Store history □

Store rating ☑

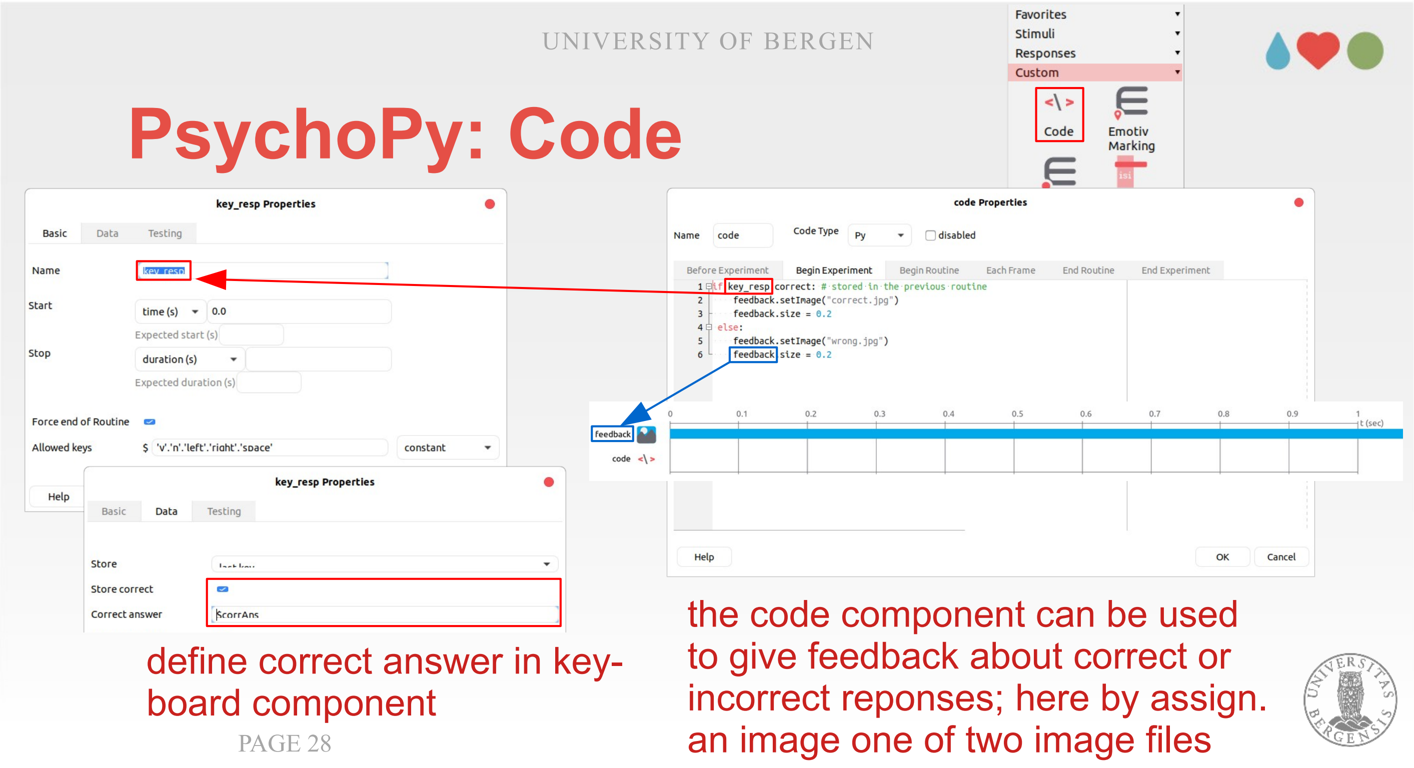
Store rating time ☑

- stop can be left empty
  tick «Force end of Routine»
- under Interface:
  untick «Show accept»
- under Data:
  lowest, highest, start (num.)
  labels at lowest, highest
  store: history, rating

The Rating component is more demanding when it comes to set it up. For most applications “Stop” can be left empty, but it is also possible to set durations or even a condition: if a certain key / response was chosen: a possible use case might be that you set the marker to the middle category (“neither / nor”), but don’t allow that category as a response option because you would like the participant to rather agree or disagree. If you leave it blank, set “Force end of Routine” which is “hidden” under the tab “Advanced”.

Lowest, highest and start are defined as numerical values. If you have a 7-point-rating scale, then those would be 1 (low), 7 (high) and 4 (start).

The tab “Advanced” also allows you to set what you would like to store: the rating itself, the reaction time or possibly the history (how much the person moved the pointer around, i.e., how uncertain the person was). Ticking “Show accept” asks for a confirmation of the choice when using the mouse.



UNIVERSITY OF BERGEN

PsychoPy: Code

key_resp Properties

Basic | Data | Testing

- Name: key_resp
- Start: time (s) 0.0
- Expected start (s)
- Stop: duration (s)
- Expected duration (s)
- Force end of Routine: ☑
- Allowed keys: $ 'v','n','left','right','space' constant

Help

key_resp Properties

Basic | Data | Testing

- Store: last key
- Store correct: ☑
- Correct answer: $corrAns

code Properties

Name: code   Code Type: Py   □ disabled

Before Experiment | Begin Experiment | Begin Routine | Each Frame | End Routine | End Experiment

```
if key_resp correct: # stored in the previous routine
    feedback.setImage("correct.jpg")
    feedback.size = 0.2
else:
    feedback.setImage("wrong.jpg")
    feedback size = 0.2
```

Help   OK   Cancel

define correct answer in keyboard component

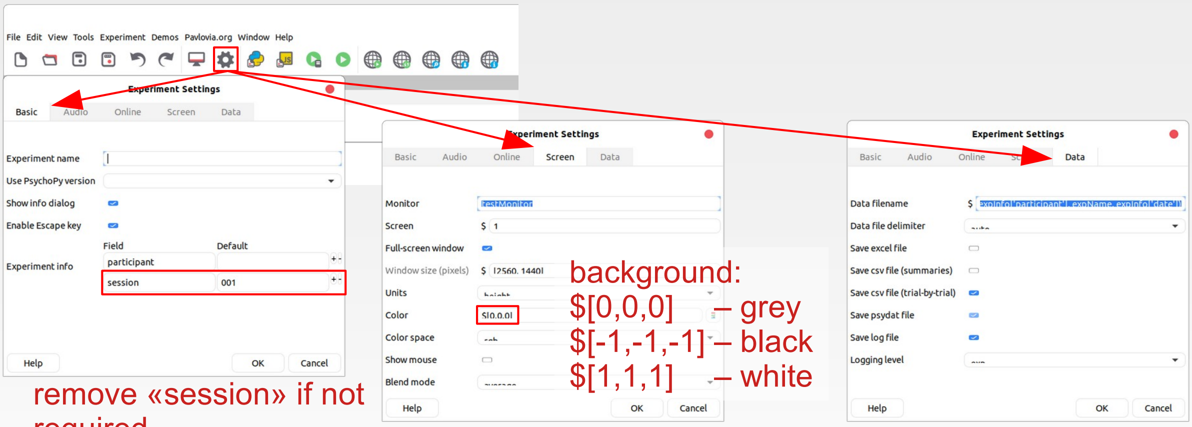
the code component can be used to give feedback about correct or incorrect reponses; here by assign. an image one of two image files

Sometimes, you would like to give feedback (particularly in practice trials). In order to do this, you need to define what the correct answer is. The screen shot above is taken from a Stroop-test. Let’s assume that blue is the correct response (and “b” the correct key). You would then put “b” in “Correct answer”. Of course, when having many trials it is better to store the reponse for each trial in a variable like in the example shown on the slide.

Later, you can tailor your feedback to whether the answer was correct or not. This is done using a Code component where you define, what is shown as feedback (e.g., which picture). You need to define an Image component and to assign a certain file to that image with code like the one shown in the example. This may sound cryptic: I will upload an example (Stroop-test) on MittUiB where you can have a look. This is maybe easier to understand.

UNIVERSITY OF BERGEN

PsychoPy: Experiment Settings



File Edit View Tools Experiment Demos Pavlovia.org Window Help
Basic Audio Online Screen Data
Help
OK
Cancel

Experiment Settings – Basic

Experiment name |
Use PsychoPy version
Show info dialog ☑
Enable Escape key ☑

Experiment info

Field	Default
participant	
session	001

Experiment Settings – Screen

Monitor testMonitor
Screen $ 1
Full-screen window ☑
Window size (pixels) $ [2560, 1440]
Units height
Color $[0,0,0]
Color space rgb
Show mouse □
Blend mode average

Experiment Settings – Data

Data filename $ expInfo['participant'], expName, expInfo['date'])
Data file delimiter auto
Save excel file □
Save csv file (summaries) □
Save csv file (trial-by-trial) ☑
Save psydat file ☑
Save log file ☑
Logging level exp

remove «session» if not required
add characteristics you want to store
(age?, gender?)

background:
$[0,0,0] – grey
$[-1,-1,-1] – black
$[1,1,1] – white

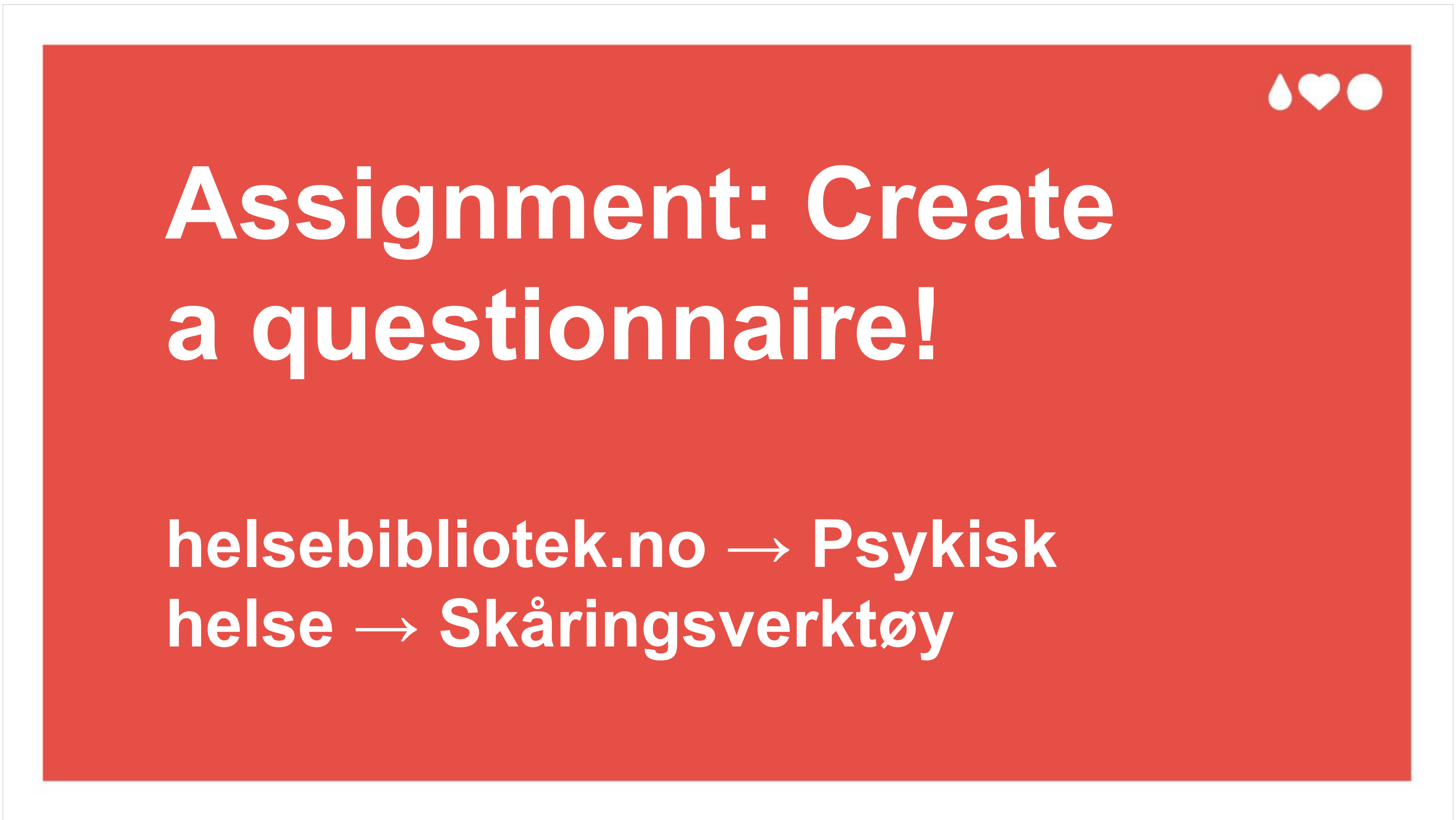
tick excel + csv (trial-by-trial) and untick log file

The final bit when preparing an experiment (or sometimes the first) is that you change the settings for that experiments.

For example, you often might not need the session question that is shown in the input window shown at the start of the experiment. In that window you or your participant enters a certain ID or a code. Instead of “session”, you possibly rather would like to ask for demographics like age, gender, etc.

In the “Data”-tab, “Save Excel file” and “Save csv file (trial-by-trial)” should be chosen, “Save csv file (summaries)” can be unticked.

Screen permits setting the background color. By default, it is set to grey. If you want a black background, you need to set it to “$[-1,-1,-1]”, for white to “$[1,1,1]”.



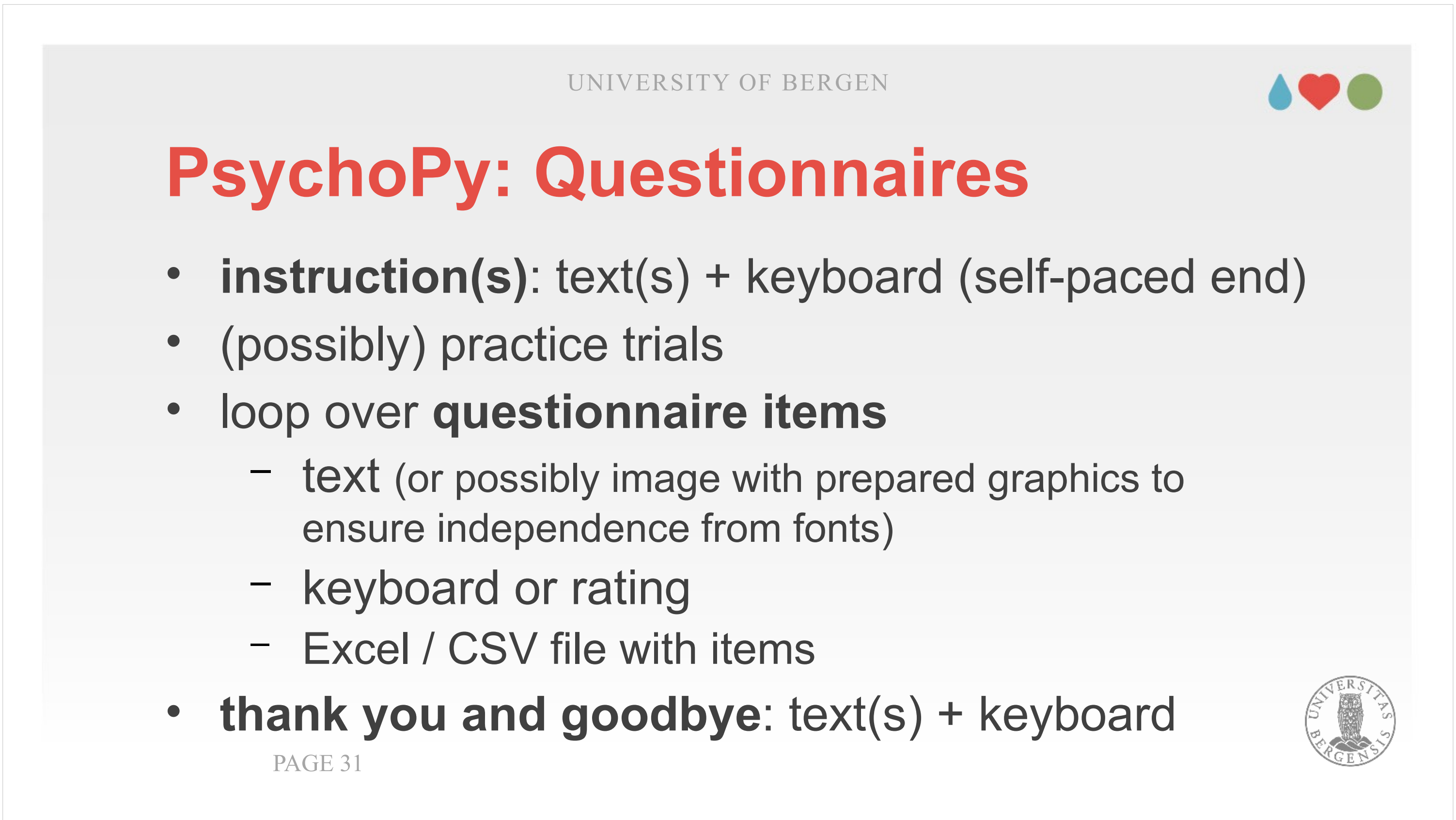
Assignment: Create a questionnaire!

helsebibliotek.no → Psykisk helse → Skåringsverktøy

After a lot of theory, it is time for a practical task. The task for you in this week is to create a computerized questionnaire. It is maybe easiest if you take the questions from an existing one. A lot can be found on https://www.helsebiblioteket.no/psykisk-helse/skaringsverktoy but you may as well choose one from another source.

To make your job not unnecessarily difficult, choose one where the responses always have the same format (e.g., 1 = ikke i det hele tatt … 5 = helt enig), i.e., where the answer doesn‘t change from question to question.

For the task, please arrange among you that you use different questionnaires.



UNIVERSITY OF BERGEN

PsychoPy: Questionnaires

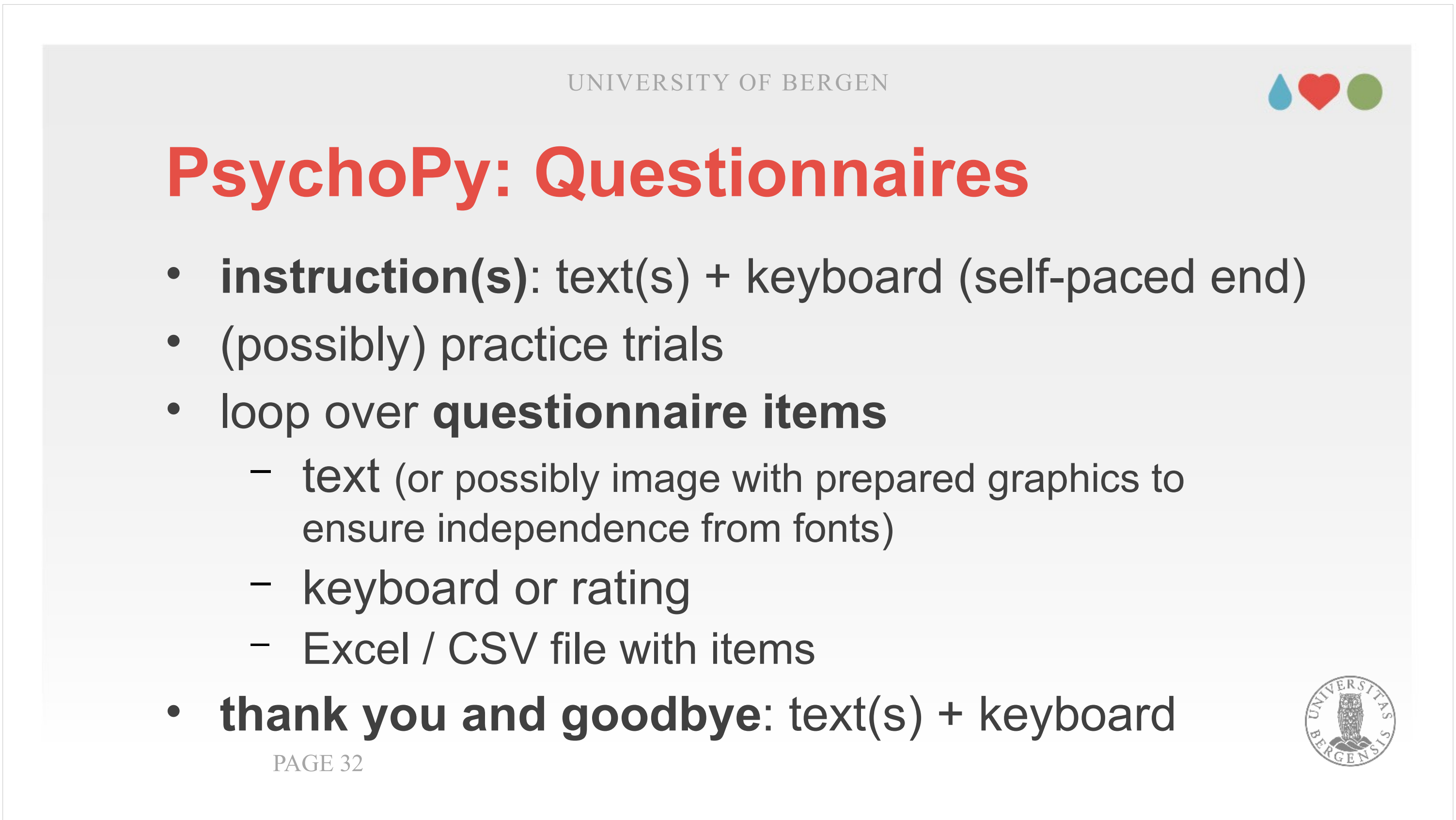
- **instruction(s)**: text(s) + keyboard (self-paced end)
- (possibly) practice trials
- loop over **questionnaire items**
  - text (or possibly image with prepared graphics to ensure independence from fonts)
  - keyboard or rating
  - Excel / CSV file with items
- **thank you and goodbye**: text(s) + keyboard

PAGE 31

Generally, when setting up such a questionnaire as computer experiment it is supposed to have up to four parts.

You always would start with instructions. Often they are already included in the original questionnaire (on paper). You might to have to adopt them slightly so that they fit the procedure when administering the questionnaire via a computer.

The second part – practice – might not be required. However, if your questionnaire is targetting groups like elderly people (who might be not very computer-savvy) or people with cognitive impairments, it might be wise to include practice trials.



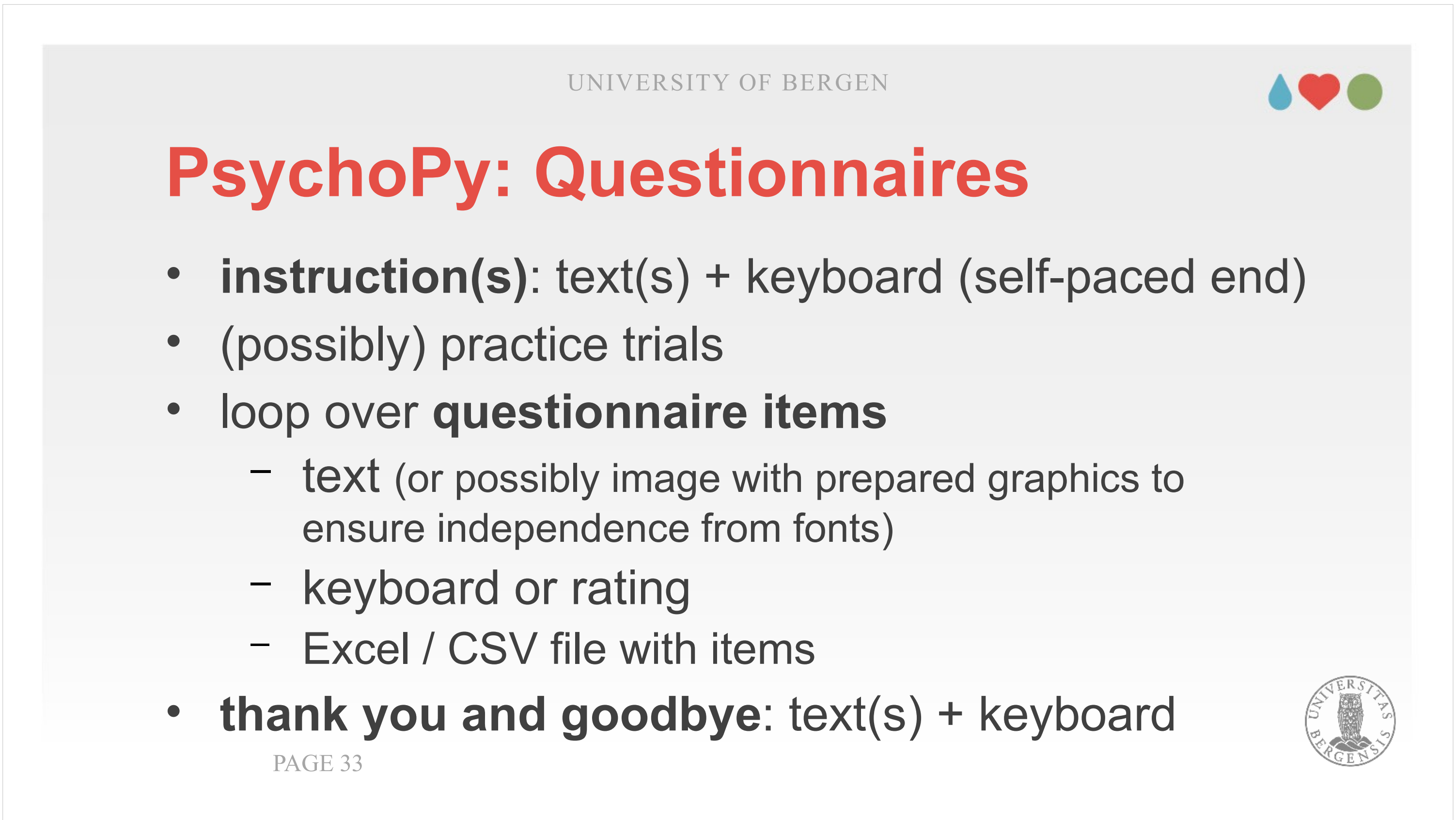
UNIVERSITY OF BERGEN

PsychoPy: Questionnaires

- **instruction(s)**: text(s) + keyboard (self-paced end)
- (possibly) practice trials
- loop over **questionnaire items**
  - text (or possibly image with prepared graphics to ensure independence from fonts)
  - keyboard or rating
  - Excel / CSV file with items
- **thank you and goodbye**: text(s) + keyboard

Then you have the main part, where you actually present your questions. Typically, you would put them in an spreadsheet (like Excel) and then have PsychoPy go through them step-by-step. I will put out an video where I demonstrate that as well as some example experiments / questionnaires on MittUiB. Please note that these are meant for you to learn not for copy-and-paste. That means, create your own experiment and use them primarily to have a look when you get stuck. I won’t be happy if I recognize to much of my routines in your submissions. Sorry, that I have to mention this, and it might not apply to you but there have been occurrences in the course with students showing a preference for using such “short cuts”.

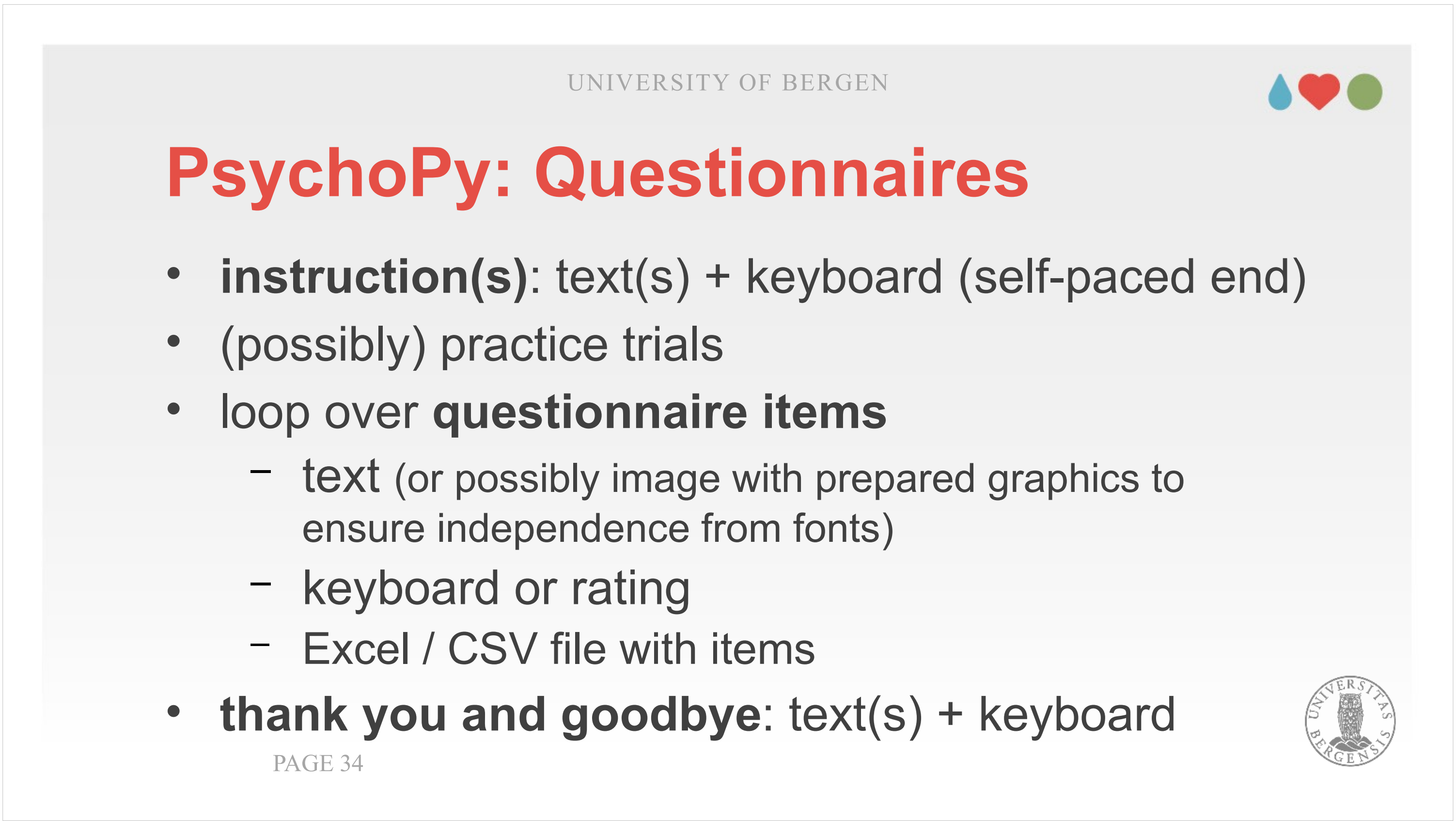
The final bit is a slide where you thank you participants.



PsychoPy: Questionnaires

- **instruction(s)**: text(s) + keyboard (self-paced end)
- (possibly) practice trials
- loop over **questionnaire items**
  - text (or possibly image with prepared graphics to ensure independence from fonts)
  - keyboard or rating
  - Excel / CSV file with items
- **thank you and goodbye**: text(s) + keyboard

Let‘s have a look what components you would have to use for such a questionnaire: instructions and thank you and goodbye are pretty simple. Both would contain some text plus the keyboard component together with an easy accessible key (like „space“) to proceed. Remember to set the tick mark for not storing those data (otherwise, you have to filter them later out of your log files).

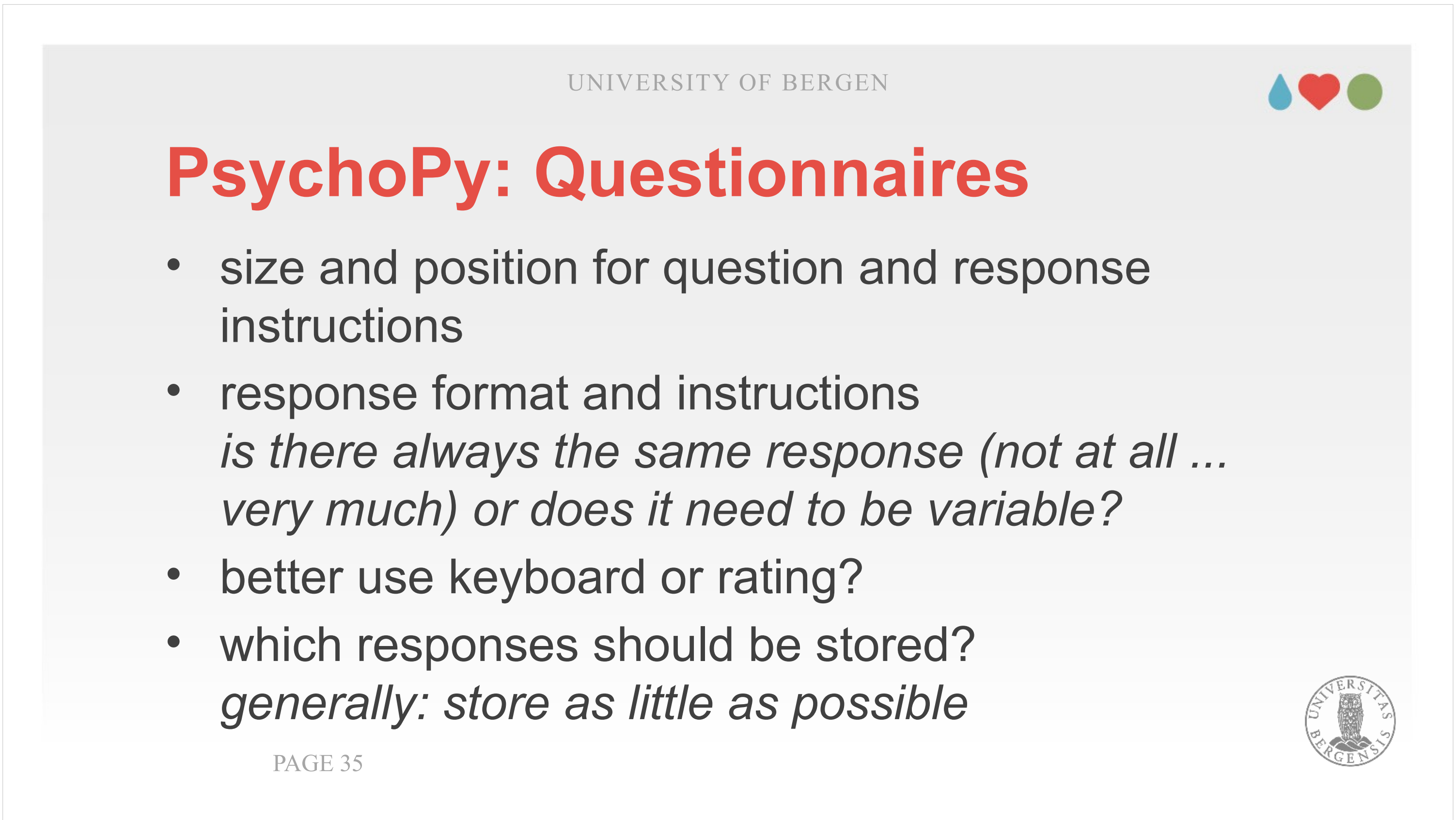


PsychoPy: Questionnaires

- **instruction(s)**: text(s) + keyboard (self-paced end)
- (possibly) practice trials
- loop over **questionnaire items**
  - text (or possibly image with prepared graphics to ensure independence from fonts)
  - keyboard or rating
  - Excel / CSV file with items
- **thank you and goodbye**: text(s) + keyboard

For you actual questionnaire items you would use a text component where you assing the content variably. To do that you would put a variable in the „Text“ field. This variable has to have the same name as the header (first line) in your spreadsheet. Don‘t forget an $ in front of that field name and remember that the field name has to be one word (e.g. itemText in Excel → $itemText in PsychoPy). Remember to change the dropdown-menu next to the Text-field from „constant“ to „set every repeat“.

For the response you need to consider which format is best. The Rating component can be used with either mouse or keyboard, but the Keyboard component might be faster if no mouse is used. As a practical advice: If you collect data from many participants, it might be worth the effort, to build yourself a mask (some cardbord or plastic) to hide the unused keys on you computer keyboard when running the experiment.



UNIVERSITY OF BERGEN

PsychoPy: Questionnaires

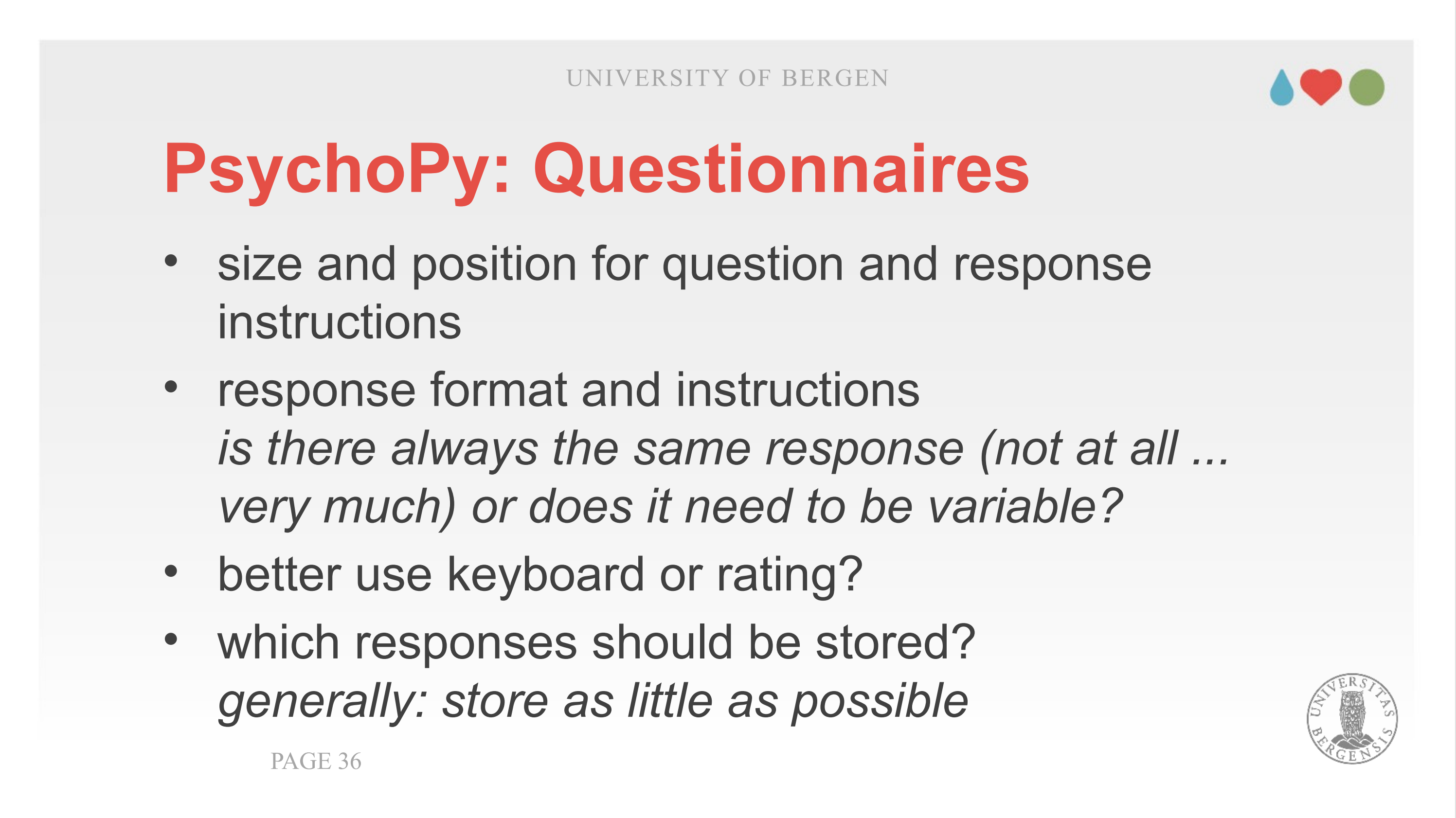
- size and position for question and response instructions
- response format and instructions
  is there always the same response (not at all ... very much) or does it need to be variable?
- better use keyboard or rating?
- which responses should be stored?
  generally: store as little as possible

PAGE 35

Finally, there are a couple of further questions to consider when creating your questionnaire. Most of those are most easily solved by trying them out.

The first question is the size of questions and responses instructions. In my experience 0.08 for questions and 0.04 for response instructions are a good size. In any case, take care that what is shown on the screen doesn’t get to small and unreadable.

The second question regards the response format. Some questionnaires always have the same format. That also has the advantage that participants get used to them over time (and quicker and more reliable when responding). Sometimes, the questionnaire might be constructed to have different answer options for every question. Then you should think about how to make them into variables (additional columns in your Excel-file).

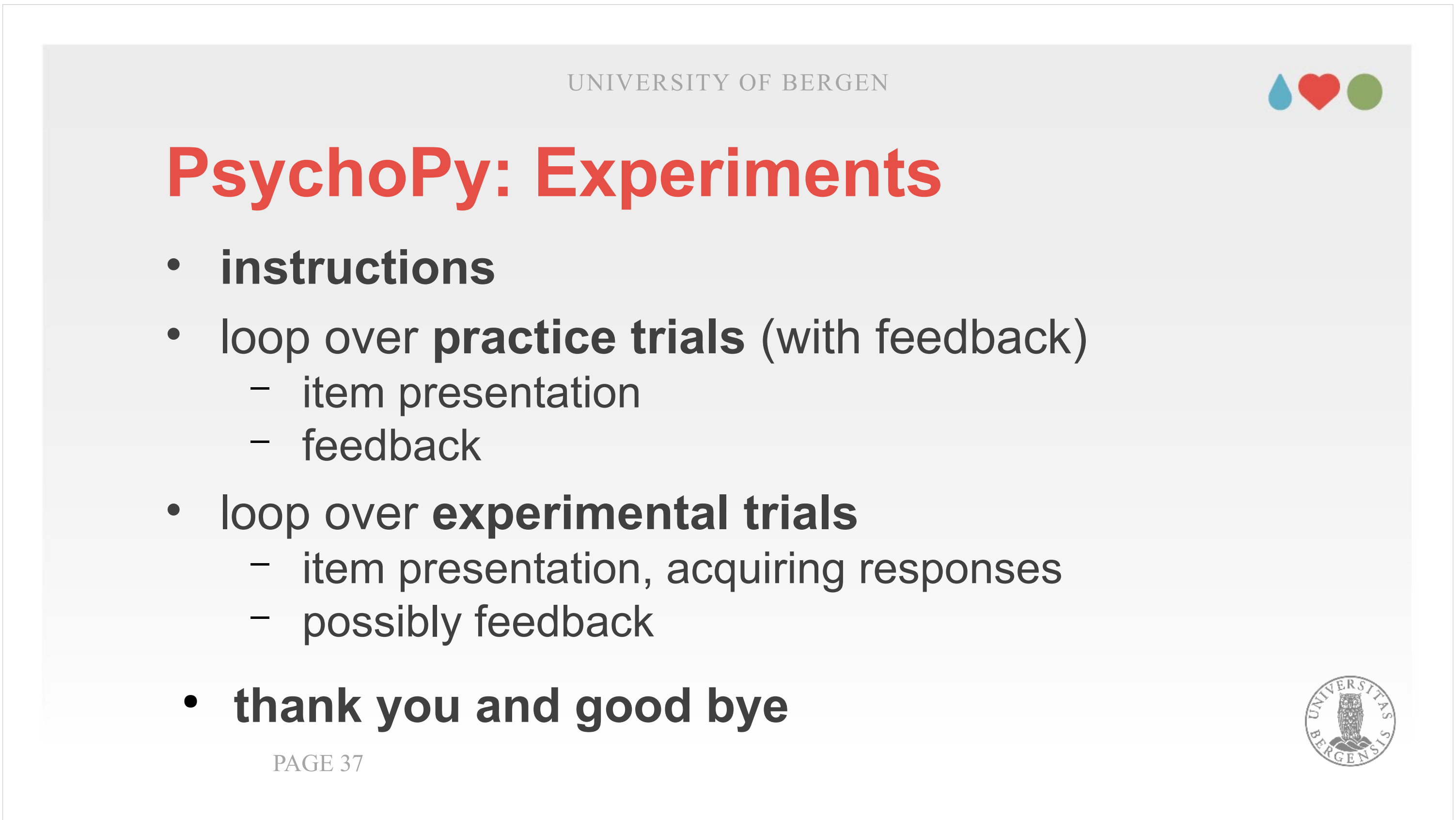


PsychoPy: Questionnaires

- size and position for question and response instructions
- response format and instructions
  is there always the same response (not at all ... very much) or does it need to be variable?
- better use keyboard or rating?
- which responses should be stored?
  generally: store as little as possible

The question regarding whether it is better to use the Keyboard or the Rating component was already raised. Rating is visually a little more intuitive so that might be an advantage. It requires to a larger amount of settings though, so maybe the Keyboard is easier for beginners.

Finally, think about which responses you want to store. In principle you are interested in the replies to the questionnaire items and wouldn’t like to have the keypress-responses from instructions, breaks, thank you, etc. in your data file.



UNIVERSITY OF BERGEN

PsychoPy: Experiments

- **instructions**
- loop over **practice trials** (with feedback)
  - item presentation
  - feedback
- loop over **experimental trials**
  - item presentation, acquiring responses
  - possibly feedback
- **thank you and good bye**

PAGE 37

Creating an experiment is something that will be covered in the next session. I would like you to already begin thinking about that. My ideal is that at least some of you can make first steps into preparing an experiment that can later be used for their theses.

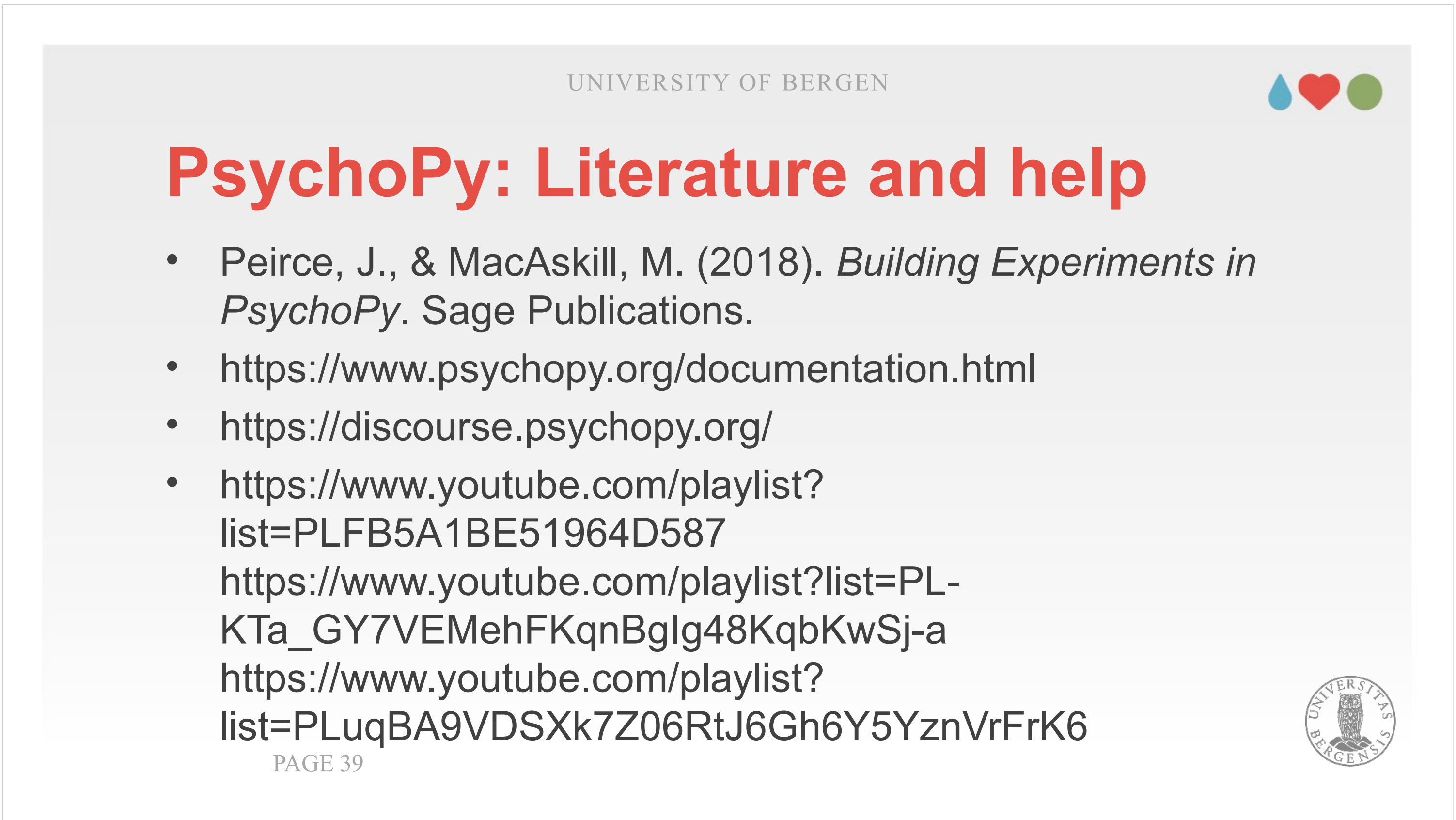
You will see from that overview that the order of the parts and much of the content of what you use in a questionnaire is the same or very similar to what you present in an experiment.

The main difference likely is the kind of stimuli you present: Whereas for the questionnaire, the stimuli are typically presented in text form (questions), in an experiment you might present images, sounds or videos. What also differs is that you might want to provide feedback (e.g., whether the reponse was correct, reaction times, etc.). You could use the Code component that was introduced earlier for it.

Literature

Further resources for help

Finally, I would like to point you to the literature and where to find further resources for help.



UNIVERSITY OF BERGEN

PsychoPy: Literature and help

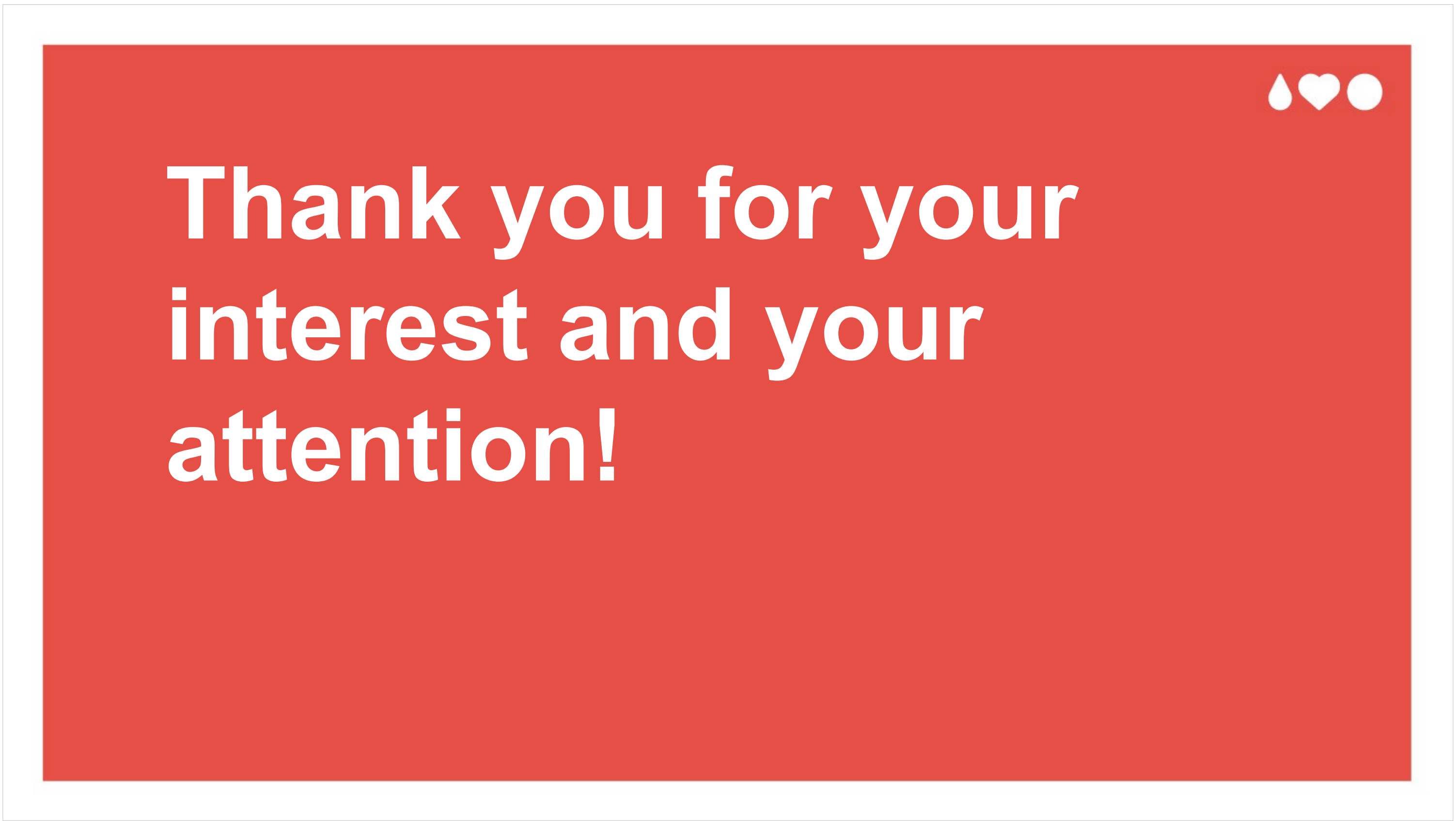
- Peirce, J., & MacAskill, M. (2018). *Building Experiments in PsychoPy*. Sage Publications.
- https://www.psychopy.org/documentation.html
- https://discourse.psychopy.org/
- https://www.youtube.com/playlist?list=PLFB5A1BE51964D587
  https://www.youtube.com/playlist?list=PL-KTa_GY7VEMehFKqnBgIg48KqbKwSj-a
  https://www.youtube.com/playlist?list=PLuqBA9VDSXk7Z06RtJ6Gh6Y5YznVrFrK6

PAGE 39

You can borrow the Peirce and MacAskill-book from the library or from me. Unforunately, the library informed me that buying an electronic copy is too expensive. But, including mine, there are currently three copies (one is pensum and must stay in the library).

Therefore, you preferrably may use the electronic resources and video resources listed above. Typically, the discourse is most helpful if you run into an error or get stuck otherwise (just copy the error message in the search field of your search engine [Google]).

There is a couple of YouTube-videos showing how to create experiments (first two entries), and how to use programming for more complex experiments (last entry).



Thank you for your interest and your attention!

I hope the lecture was helpful. Thanks for your attention!