classEx documentation

Release 1.0

classEx

Oct 21, 2025

CLASSEX DOCUMENTATION

Welcome to the documentation of classEx. In this documentation, you can find all the information on how to use classEx. General information on classEx can be found at https://classEx.de.

Note

You can also download a pdf version of the documentation. Please note that we change the documentation from time to time. So you might check from time if your PDF is still up-to-date. For this, just compare the release version. Current release is 1.0.

classEx is an online tool to run interactive experiments and surveys. It is directed both towards usage in the classroom and lectures as well as for running experiments with a group of people located in one place (cinema, public viewing event,...). Participants log in with their mobile device (smartphone or notebook) and can participate interactively in the lecture. They only need a standard browser and an internet connection. Results are shown instantly on the presentation screen of the lecturer with intuitive graphs. Quizzes and spontaneous tests can be integrated into lectures or presentations of all academic disciplines. classEx can be used in political science to simulate polls or elections, psychologists can test the effect of roles and scenarios, and economists can assess the strategic interaction in markets and the whole of behavioral economics.

This documentation is written for lecturers and experimenters who want to use classEx for conducting games in the classroom or in the field. In the following, we will talk about lecturers as the ones running the game. All information holds true for experimenters as well. **Participants do not need to read this documentation.** Participants are informed by the lecturer on how to log in.

The chapter *Getting started with classEx* explains the first steps for new classEx users.

In the chapter *Run ready-made games*, you find a step-by-step guide on how to run ready-made games in classEx.

The chapter *Organize games* tells you how to structure your games in your account and how to find games from other users.

The chapter *Tutorial: How to develop a simple game* shows a step-by-step introduction for how to develop an ultimatum

game.

The chapter *Develop your own games* is a manual for users who want to develop their own games.

The chapter *Elements* provides an overview of all elements which can be used to develop own games.

The chapter *Programming* provides a list of variables and functions for advanced programmers.

classEx can be used free of charge for academic and non-academic purposes. Read more in the terms of use. It is provided free of any warranty. classEx is constantly being improved. Users are being added and are delivering results to existing experiments. Further experiments are being designed and made available. Likewise, the functionality is continuously being revised and updated. If you have any questions, support issues or want to report bugs, please mail classEx@uni-passau.de or visit our forum.

CHAPTER

ONE

GETTING STARTED WITH CLASSEX

1.1 What you need

classEx is an online tool which runs centralized on the servers of the University of Passau. Therefore, no installation of software and no download of applications is necessary. Running games and developing games are done in a standard internet browser.

Lecturers and participants only need a mobile device with an internet connection and an up-to-date browser. We suggest using Firefox. Make sure that JavaScript is enabled and cookies are allowed (normally this is a default setting in many browsers). **Do not use Internet Explorer for using classEx because classEx does not run with it.**

For using classEx, the lecturer only needs to register in order to get login credentials.

1.2 Registration

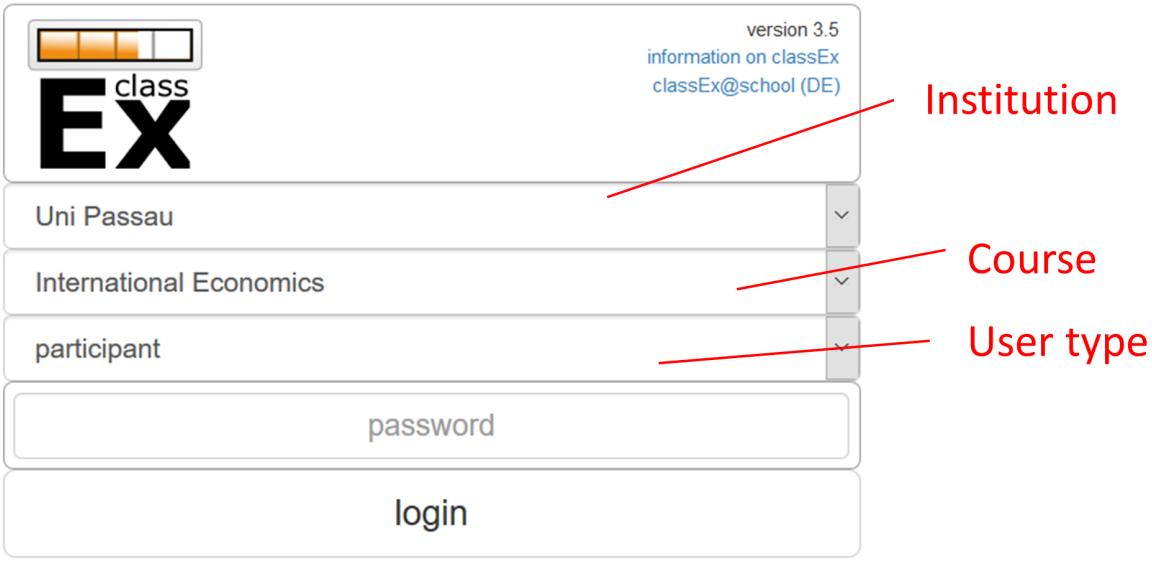
Participants do not need to register, but can access classEx with the course password provided to the lecturer with their login credentials.

To get login credentials, please fill in the registration form at https://classex.de/get-login-credentials.

For registration, you have to provide your name, an email address and information about your institution. In addition, you have to confirm that you accept the terms of use and the data privacy policy. **Please try to provide an institutional email address so that we can verify your affiliation.** Registered users receive an email with login credentials. Login credentials are normally sent within 1-2 days. If you do not receive credentials after 2 days, please contact us at classEx@uni-passau.de.

Each user has an account (called a course in classEx) which is linked to their institution. With the login credentials you obtain the right to use classEx for teaching, education, research and other purposes (for details please see terms of use).

1.3 Login



The website https://classex.uni-passau.de shows the login for lecturers and participants.

Institution
Select the institution you named during your registration, typically your university.

Course
Select your course or another course you have access to.

User types

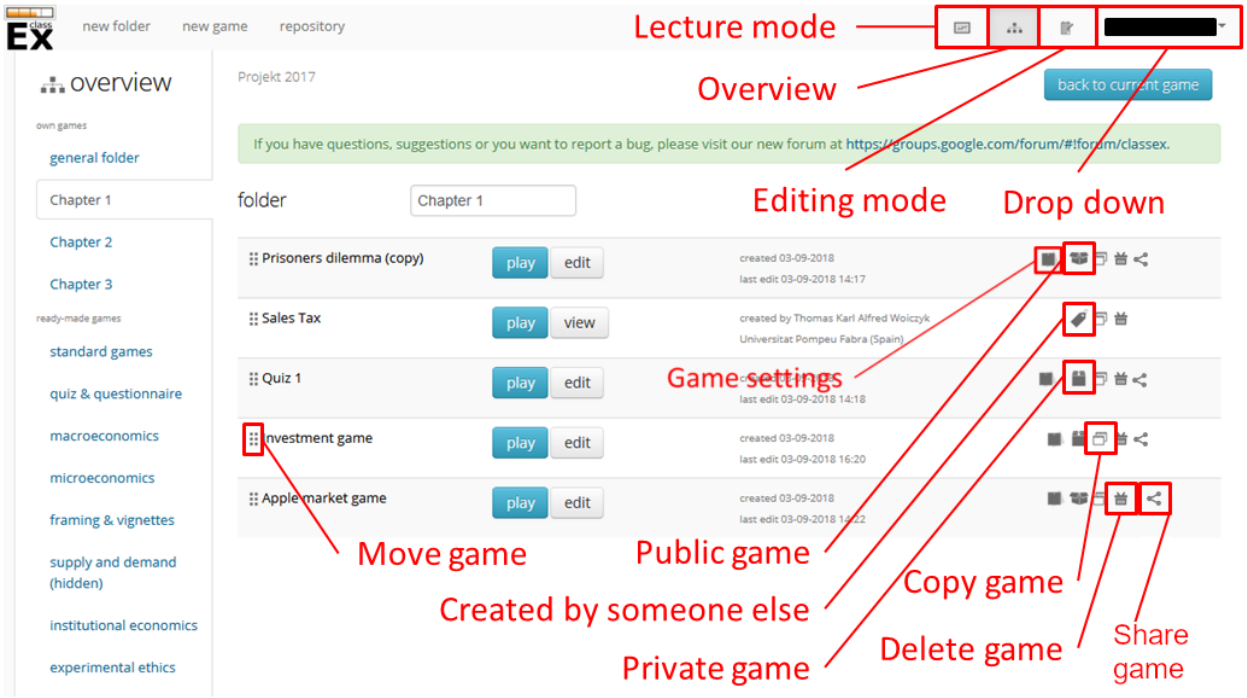
- Participants: They can participate interactively in a lecture or a presentation by logging in as a participant. They do not need to register and remain anonymous. They select the university and the course from the lecturer's registration. The lecturer provides the course password. Participation after logging in is self-explanatory and does not require any previous knowledge.
- Lecturer: Registered lecturers who have received a password can log in as experimenter/lecturer. As a lecturer, you can organize your account, start ready-made games and quizzes, adapt them to your liking or create new ones. This documentation mainly explains the possibilities and options classEx offers for lecturers.
- Administration: Individuals entrusted with the payoff can enter the administration mode to disburse the monetary payoff. This task can be delegated to a trustworthy person by the lecturer. You can find further instructions in *Disbursal of payoffs*.

Password
A registered lecturer has received a password upon registration. This password can also be used for the administration user type. Participants are provided with their password by the lecturer (under *Participants password* you see how to change it and what to do if the participants password is lost).

1.4 A short overview

You can find a short video on how to use classEx on https://www.youtube.com/embed/Zm0DpUzhOGg.

As the video shows, entering classEx as a lecturer offers three modes: overview mode, lecture mode and editing mode. After login, the screen shows the overview mode. Here you can organize your games and access all important features of classEx. In the lecture mode, you can run games, and in the editing mode, you can develop your own games.



In the top right-hand corner, you find the main navigation bar, which is always displayed in every mode. This allows you to switch from one mode to another and access your personal data.

The currently active mode is marked by a darker shade around its symbol, here the **overview mode**. The left symbol takes you to the **lecture mode**. The right symbol takes you to the **editing mode**. The **drop down menu** (which shows the name of the lecturer and is shaded in black in the figure above) provides access to your personal data and your course data, as well as the terms of use, the documentation, some general info on classEx and the log out button. More information on how to change your personal data can be found at *Personal data* and how to change your course data at *Course data*.

The different functionalities of copying, deleting,... are explained in the section *Game*.

The top bar is located in the top left corner. It is different in each mode. There, you can find the main function for each mode. The details are explained in the respective section (lecture mode = Run ready-made games *Settings in the top bar*, overview = Organize your games *Settings in the top bar*, editing mode = Develop your own games *Settings in the top bar*).

1.5 Terminology

This chapter clarifies the usage of some terms in this documentary. It can be used to look up terminology and can be skipped on first reading.

Lecturer
: The person conducting a game is the lecturer. The lecturer starts games, starts new rounds, ends games and

shows results. The lecturer controls the lecture screen that is visible to all participants (typically via a projector in the lecture hall).

Participant
Participants participate in games. All a participant needs for participation is a mobile device with internet access. No download is required. Sometimes participants are also called players.

Session
A session is a sequence of games in a lecture, meeting or presentation. Participants should not shut their browsers during a session.

> **Note**
>
> After the end of a session, you can use statistical tests to analyze whether there is a relationship between the different games of a session. For example, you can examine whether participants with higher mathematical abilities are more risk-averse. For this purpose, participants' ID-numbers are stored in an Excel sheet.

Game
Games consist of a sequence of stages. A game is typically characterized by a joint evaluation of the decisions and results at the end.

> **Note**
>
> If you want to conduct a quiz consisting of several questions with unrelated results, it is advisable to create a separate game for each question.

Stage
Games consist of several stages. There are at least 2 stages: one for the decision input and one for the result output. Stages are ordered sequentially and are meant to be synchronization points in the game. Synchronization means that for the next stage to begin, all elements of the previous stage must have been finalized. Stages can be configured with several options. You can find more information here.

Element
Elements are the modules of each stage. A stage has two areas in which you can add modules: participant and lecturer. You can choose from text elements, input elements (numerical input, likert scales, ...), program code elements and output elements (histograms, bar charts, ...). These can be combined and arranged as you like.

Treatment
Treatments allow you to treat participants differently throughout a game. You can assign participants to treatments and customize stages and elements for treatments.

Role
Many games require different roles of participants, e.g. producers and consumers. Stages and elements of a game can be customized according to the role of a participant.

Group
Active participants of a game can be sorted into groups, e.g. according to their role, internal ID, randomly or a combination of these.

Assignment and Matching
Assignment and matching refers to the procedure of how participants are assigned to treatments, roles and groups at the start of a game. Further, you can choose how you want to rematch participants at the beginning of each round if you play more than one round.

Round and Loop
The number of rounds a game should be played can be defined. The loop refers to the stages of a game that should

be repeated in every round. The loop is defined by selecting a starting stage, an ending stage and the number of rounds.

Internal ID
ClassEx creates a unique internal ID for each subject that logs in. This ID is generated randomly and does not allow any inference about the identity of the subject. Therefore, subjects are completely anonymous in classEx by default. The internal ID serves as a mean to be able to analyze the data and compare behavior of subjects across different games if you play several in one session.

External ID
On login, participants can be asked to provide an external ID (e.g. their matriculation number). The external ID can also be provided with the link for automatic login. Please make sure that you elicit external IDs in accordance with data privacy regulations, as the lecturer is responsible for this during data collection (see terms of use).

Subject ID
Subject IDs are used only within a game. Each participant gets an ID from 1 to the total number of participants. The first participant gets the Subject ID 1, the second participant the Subject ID 2, and so forth.

Global and subjects variables
Global variables are variables on the game level. They have the same value for all participants (e.g. an exchange rate). Subjects variables are variables on the subject level. The value of a subject variable is calculated separately for every participant (e.g. individual payoff).

Parameters
Parameters are global variables that are adjustable before running a game (e.g. the endowment). Parameters can be changed directly in the lecture mode. They have the same value for all participants.

Global and subject program code elements
Many games require calculations or algorithms. These are created in program code elements. The programming language used in these elements is PHP. Global program code is utilized for calculations on the game level. Subject program code is utilized for calculations on the subject level (for every participant).

CHAPTER

TWO

RUN READY-MADE GAMES

In the overview mode in classEx, you can find a set of ready-made games which can be played directly in the lecture mode. The interaction between the lecturer and the participants takes place in the lecture mode. The lecturer's browser is usually projected to a wall. Games are started and terminated in the lecture mode, and the results are also displayed in this mode. Just start a game by pressing "play" and open some test participants by clicking on the test participant symbol in the top navigation bar. Test participants open a new tab in the browser.

You can find a short video on how to play a ready-made game on https://youtu.be/zmAn0ZzTQPk.

If you are already familiar with running a ready-made game and want to prepare for your first classEx experiment in your lecture, you can also download the classEx cheat sheet. It gives a step-by-step guide on what to prepare for and do in class.

2.1 Easy mode

In order to facilitate your start, you can use the easy mode. It reduces the options available in classEx. You are only allowed to play games there and cannot edit them or create new ones. The easy mode is switched on by default in accounts for classEx@school. In a standard account, you can switch it on in the course settings in your profile. The easy mode gets active after relogging.

2.1.1 What you need

2.2 Participants' devices

Every participant who wants to take part in a session needs a mobile device (e.g. mobile phone or tablet) with a browser and internet connection. Browsers should be up-to-date, JavaScript and cookies enabled. Participants can connect via mobile Internet or available wifi.

2.3 Lecturer device

The lecturer conducting a session needs a device with a big screen (e.g. laptop, tablet) with a browser and internet connection. The screen of the lecturer's device should be visible for all participants (e.g. via a classroom projector).

> **Note**
>
> Make sure you have a stable internet connection as a lecturer. The best option is with a cable. It is best not to use the same connection (e.g. wifi) as the participants in case the network slows down.

> **Note**
>
> The ideal browser to use is Mozilla Firefox (in a current version). JavaScript has to be turned on and cookies allowed (normally the default setting). **Do not use Internet Explorer**.

2.3.1 Run a game

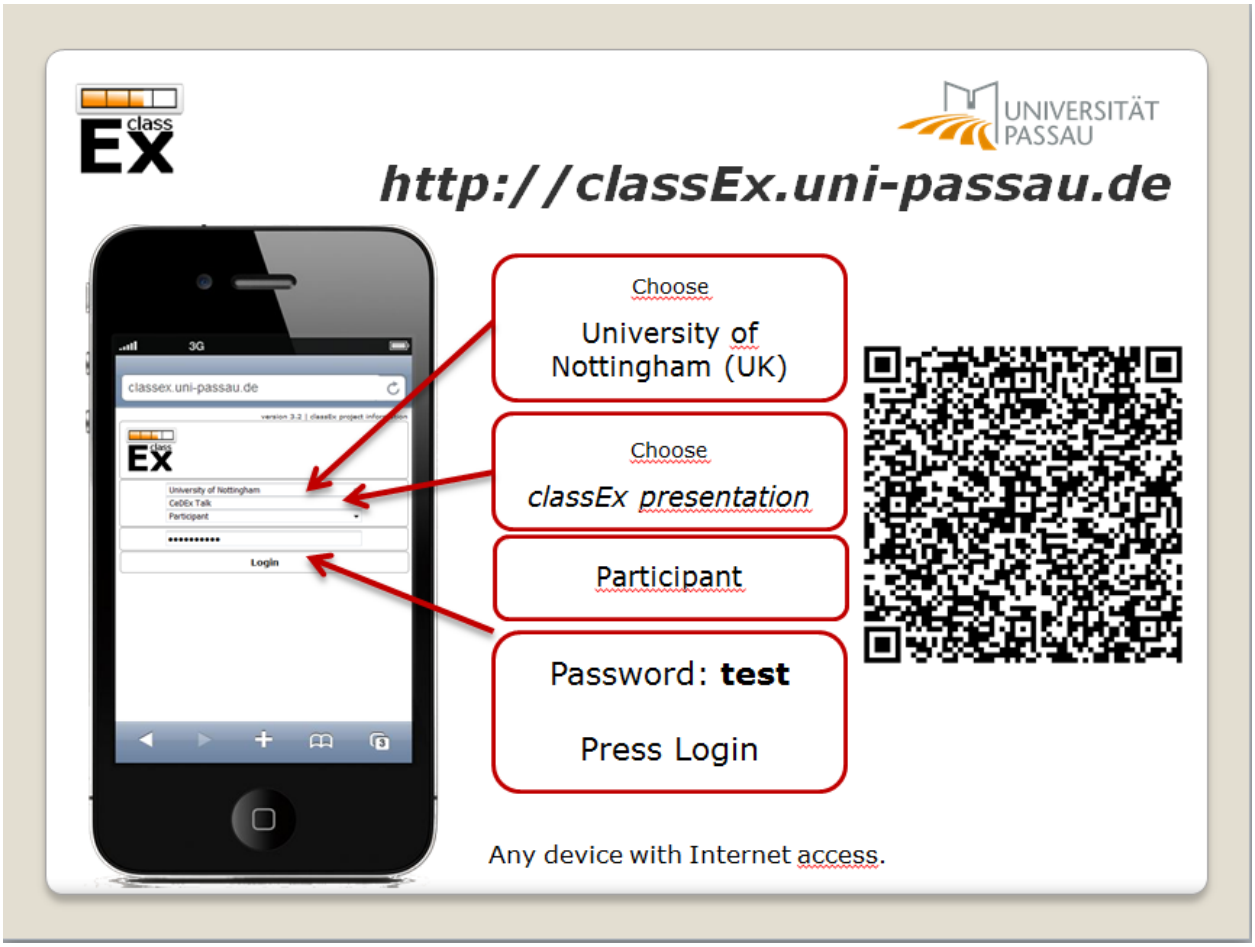
2.4 Information for participants before the course

You may inform participants upfront, e.g. via email, that you plan to run an experiment in class.

- participants have to bring a mobile device that has internet access, e.g. a smartphone, a tablet or a laptop (see *Participants' devices*).
- The browser they use should be up-to-date, and JavaScript and cookies enabled. We suggest using Firefox.
- The device has to have enough battery left (suggest bringing a charger).

Additionally, if relevant, you can inform participants that classEx only requires a small amount of data volume (for participants). When using classEx as a participant for the first time, loading all settings requires about 120 KB (this is cached and does not need to be reloaded when reentering). Each game that is played requires a data volume of roughly 20-50 KB. If you decide to use games with additional graphical libraries (e.g. for plots on the participants' screens), this amount may be higher. This information may be especially relevant if participants have limited amounts of data volume.

2.5 Information for participants in the course



The only important information participants need is how to log in. The easiest way is to provide them with a PowerPoint slide with the login data. You can add a QR code, but notice that QR codes may not work in big lecture halls. For all login possibilities (link, QR,...) see **`Login of participants and test participants`_**.

Before you start a game, you may also inform participants about the following:

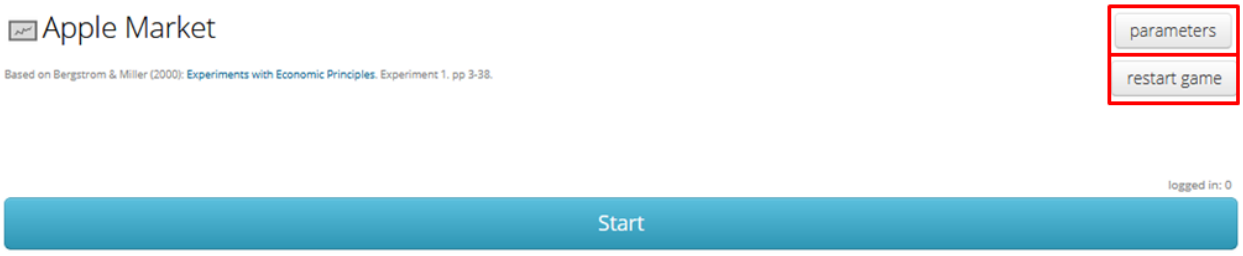
- Participants need to stay logged in until the experiment is over.
- Participants should close all apps. Having other apps open might reduce the internet speed and, therefore, the functionality of classEx.
- If the experiment has already started (and some decisions have already been entered), participants might not be able to join the experiment any more. This depends on the game, but usually joining is no longer possible after the first stage of a game has been concluded.
- If participants are chosen randomly to receive their payoff, they should make screenshots of their winner's notification to be sure to receive their payoff. (see *Disbursal of payoffs*).

2.6 Before the start

Ask all participants to log in. The counter over the start button shows how many participants are currently logged in. There is no minimum number of participants required to start a game.

> **Note**
>
> The best option is to let all participants log in first, then start a game. Participants can also log in when the game is running. Participation is always possible when the first stage of the game is running. Depending on the game, it may not be possible to participate if the game has already proceeded to later stages.

The lecturer can select a new game by choosing it from the drop-down list in the top left corner or by selecting it in the overview mode. The drop-down list shows all available games.



Parameters offer a possibility to adapt the setting of the game. If a game has changeable parameters, the *parameters* button appears in the lecture mode before you start the game. You can change the parameters of a game by clicking on *parameters*. For example, in a public goods game, you can change the endowment and restart the game with the new settings. The button *restart game* allows you to restart the current game.

2.7 Start a game

A selected game can be started by pressing the blue start button.

By pressing start, the lecturer initiates the first stage of the game. If a game consists of several stages, the start button for the next stage appears after pressing the start button for the first stage.

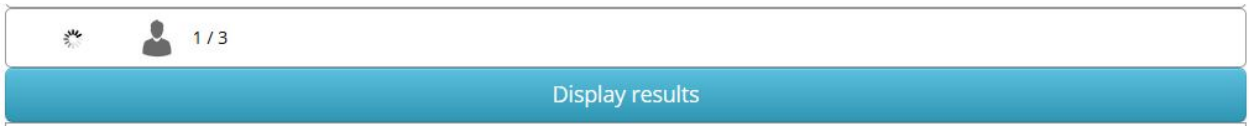
> **Warning**
>
> By starting a new game, the currently running game is stopped. There can only be one running game at a time.

If a game consists of several treatments and/or roles, the participants will be assigned into treatments/roles in the first stage. For more information on assignment of roles and treatments, see *Develop your own games*.

> **Note**
>
> If you have different roles and treatments, the first stage should be opened long enough that all participants are assigned to a role and treatment. The assignment is done when the stage has been loaded on the participants' devices. So make sure that you do not continue to the next stage too fast.

2.8 During the game

During the course of a stage, a display shows how many participants are logged in and how many of them have already made their decision in the current stage.



Here, 3 participants are logged in, and 1 has already made their decision.

Note

If you play a game with large groups, it can happen that participants take some time until they make their decision. You should wait for a while, but then terminate the stage and carry on if the added value of more input is fairly small.

Note

Note that there is no way back if you continue to feedback or next round/session.

Note

If you accidentally close the browser, no problem. Just open it again. Normally, you should still be logged in. Otherwise, just log in again.

2.9 End the game

When the participants have made their decisions, the lecturer can end the game by clicking on the blue button. In many cases, it shows „display results“.

If games are played for real money (and participants receive a payoff code), the lecturer does not only have the normal „display results“ button but also the enhanced button “Display results and payoff”. If you do not want to pay out any money, for example, in a practice round, you have the possibility of clicking on “display results only” below the actual button.

Note

A game does not have an end button. A game is only stopped when a new game is started.

2.10 After the game

Once a game is over, you can download all recorded data by selecting "download as Excel file" from the "data" menu in the top bar. If you want to go back to compare the current results with old ones or ones in different courses, click on "previous results".

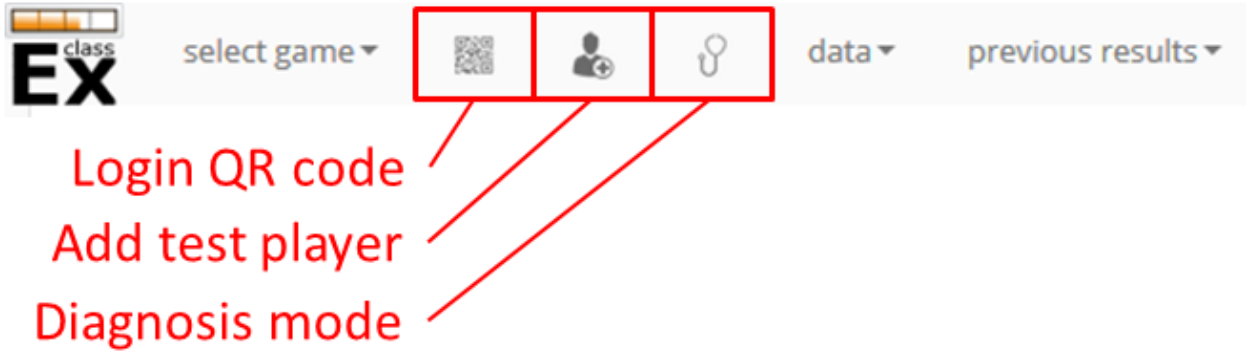
> **Note**
>
> Downloading data sometimes causes problems if datasets are too large. In this case, Excel will show an error when opening the dataset. Please contact the classEx-team in this case, and you will receive your dataset via email.

If you want to log out all participants after the game, you can do so by clicking on the "log out all participants" button in the *Course data*.

> **Note**
>
> Logging out all participants does not work correctly when trying it with your self-made test participants. If you want to test it, you can use different mobile devices and therefore simulate real participants.

2.10.1 Settings in the top bar

The top bar of the **lecture mode** provides the following functionality.



select games
: Select games from your own and ready-made games. The selected game opens up as soon as you click on it.

login QR code
: Click on the QR code to display the QR code for login. The QR code page also provides some details about other ways to log in (see **`Login of participants and test participants`_**)

test participant
: This button will open a test participant in a new tab. This can be very useful to test classEx games. If you want to open multiple test players at a time, just hold the Ctrl button and click several times on the test player icon.

diagnosis mode
: In the diagnosis mode, you can see all variables for the lecturer and the participants, which makes detecting programming errors much easier. It is only useful if you use advanced programming.

data
: The drop-down menu *data* offers two options. By clicking on **show data**, you can access a preview of participants' current decisions in real time. You can also download the results via **download as Excel file**. More information on the Excel file can be found under *Excelfile*

export to x-econ

classEx offers the possibility to directly export data to the data-repository xecon for long-term storage and publication. More information on the Excel file can be found under *xecon*.

previous results

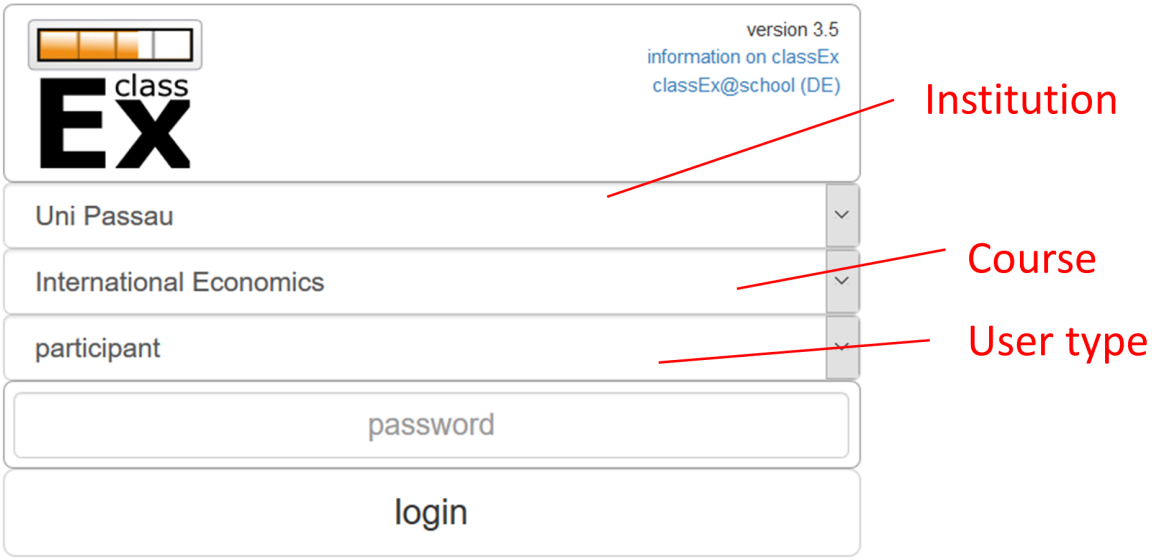
You can access previous results via the previous results drop-down menu. Simply choose which results from previous sessions you want to display. This way you can directly compare current outcomes with previous ones, you just played or, if available, old results of the same game. When you select a previous result, you can also download all data for this result by clicking on **show data** in the data drop-down menu. The previous result section also contains results from other lecturers if the game and the recorded data are public. If you use such a public game, your data will be available to other lecturers as well.

Note

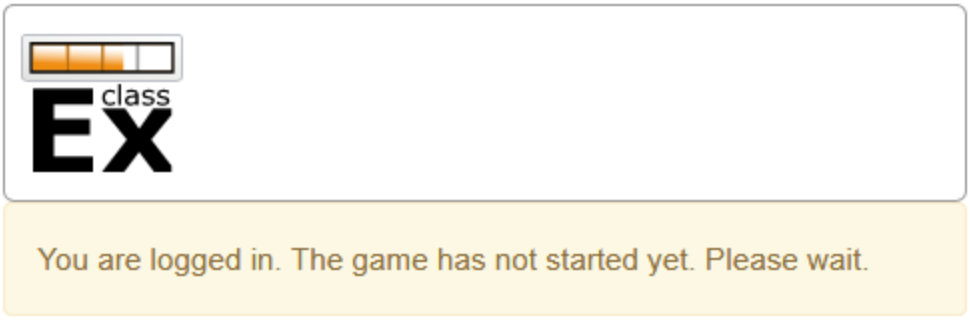
If you do not want others to see your recorded data when you use a public game, you should make a copy of the game and set it to private. Then play the private version of the game.

2.10.2 Log in of participants and test participants

2.11 Login via website (default)



In order to log in, participants go to the website http://classex.uni-passau.de and choose their university and then their course. They enter the password provided by the lecturer and click on “Login”.



If participants are logged in before the lecturer has started the game, participants see a waiting screen with the message displayed above. The lecturer can edit the text on the waiting screen in the *Course data*.

Note

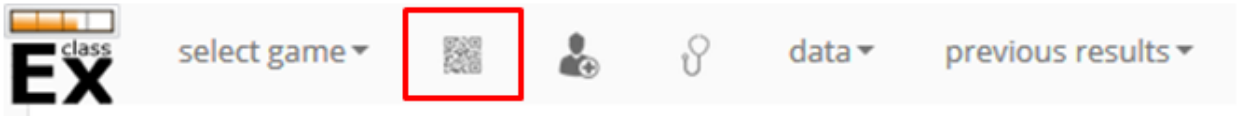
If a game has already been started, participants directly see the game and can play it. It may therefore be useful to ask all participants to log in and start the game after that has happened. If a game has ended (but no new one selected), new participants get the message on login that the current game is still running, and they cannot participate.

Note

If you want to identify participants, you can also add some additional fields to the login field. This can be done in Coursedata:Additional setting.

2.12 Log in with QR Code

All experiments can be accessed by participants via a QR code. This QR-Code is provided automatically in the lecture Mode in the top bar. Display the QR code by clicking on the symbol . When clicking on the QR code symbol, instructions on how to log in without using the QR code also appear on the screen.



Lecturers can either copy the QR code and print it on leaflets, for example, or display it on the screen.

Note

Be aware that in big lecture halls, the QR code cannot be scanned directly from the projection screen.

2.13 Automatic link for login

When you click on , you also get some information on how to log in participants directly with a link. They just have to copy the link in their browser and are logged into your course (without the need to select the course and enter a password). The link looks like this: https://classex.uni-passau.de/bin/?automatic=L3ZY2rNO2gz14YkeGUxxF-g (this link is just for demonstration and cannot be used.)

If participants have played classEx in another course (or you play games in two different courses in a row), they may still be logged in to the old course, and the automatic login would not work. To make them switch the courses, just add "&logout" to the link so that they are logged out from the current course they are in and are automatically logged in to your course.

2.14 Personalized link for login

You can augment the automatic link with a personalized ticket. This ticket will be stored as an external ID in classEx and allows you to identify participants. This way you can ensure that participants only take part on one device and also track the actions of specific participants. You simply need to add &tic= to the URL. The ticket is saved to the participant data and can be retrieved by the variable $tic; in the game. It is also saved to the Excel file. If the ticket is e.g. 12345, the link looks like this: https://classex.uni-passau.de/bin/?automatic=L3ZY2rNO2gz14YkeGUTsdsdsFs&tic=12345 (this link is just for demonstration and cannot be used.)

2.15 Add test participant

As a lecturer, you can run a game with fictional test participants in one browser. To add a test participant, click on the button in the top bar of the lecture mode:

For every test participant, a new tab in your browser will open. The tab for a test participant replicates the fully functional interface for a real participant. This enables you to make test sessions, which is especially useful when you develop your own games. If you want to open multiple test participants, just hold the Ctrl Button and click on the icon multiple times.

2.16 Logout

There is no (visible) logout button for participants. This is done in order to keep participants in the game, so that they cannot log out accidentally.

If you want to log out a participant (e.g. in case of a problem), click on the classEx logo on the participant's device. This displays a logout button.

As a lecturer, you can log out all participants that are currently logged in to your course by going into your *Course data* and by clicking on the button below. This will log out all participants immediately. This may be necessary if you run to lectures in a row and do not want to have the participants from the first lecture in the second one.

2.17 No refresh page needed

Participants' screens are updated automatically when their partner has made a decision or when the lecturer has started a new stage. Therefore, it is not necessary to press a refresh button to proceed. This way, participants can simply wait until the next stage appears on their mobile devices and do not have to keep refreshing their screens.

2.17.1 Participants' screens

The participants' interface should be self-explanatory. The top bar contains the classEx logo and shows if participants are assigned to a specific role by displaying a (colored) icon . The top bar may additionally show the internal participant ID of the participant.

The most common actions participants are asked to carry out are discrete decisions and numeric decisions, as shown in the following sample screens. Another common element is the contract element, where participants can trade items with other participants. There are also other input types such as text elements, radio buttons or sliders which are explained in the section *Elements*.

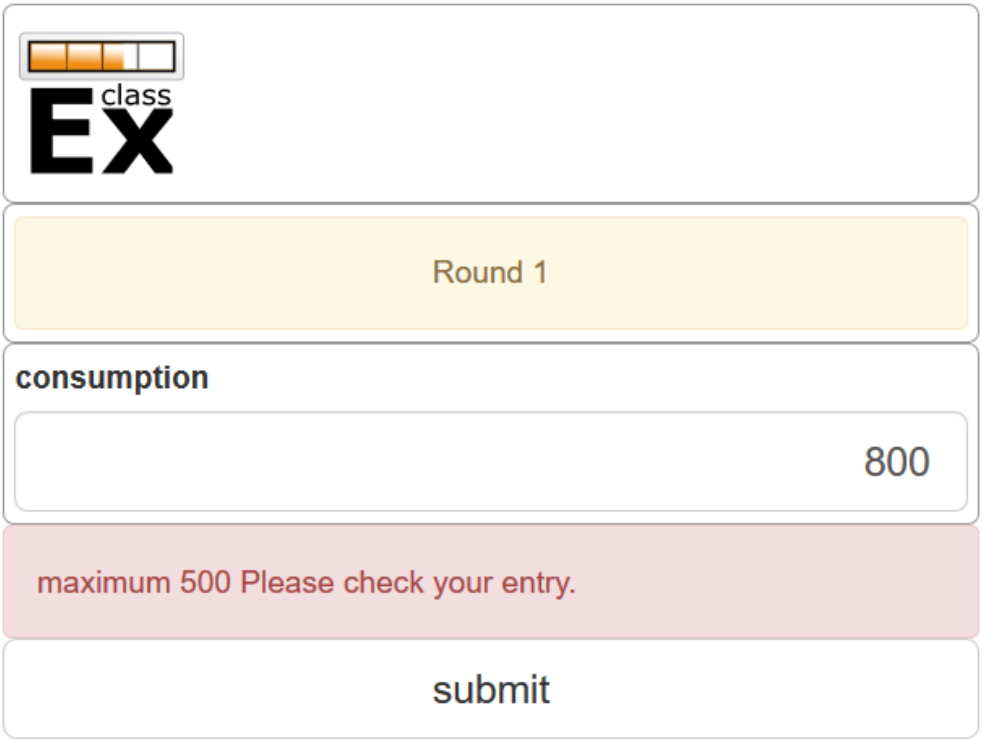
2.18 Discrete Decisions

Discrete options can be shown to the participant. By clicking on one of the options, the decision is submitted and saved. The participant is informed that his or her message has been stored.

Note

Participants cannot undo their decisions. You may add that participants have to confirm that they are sure to send their input.

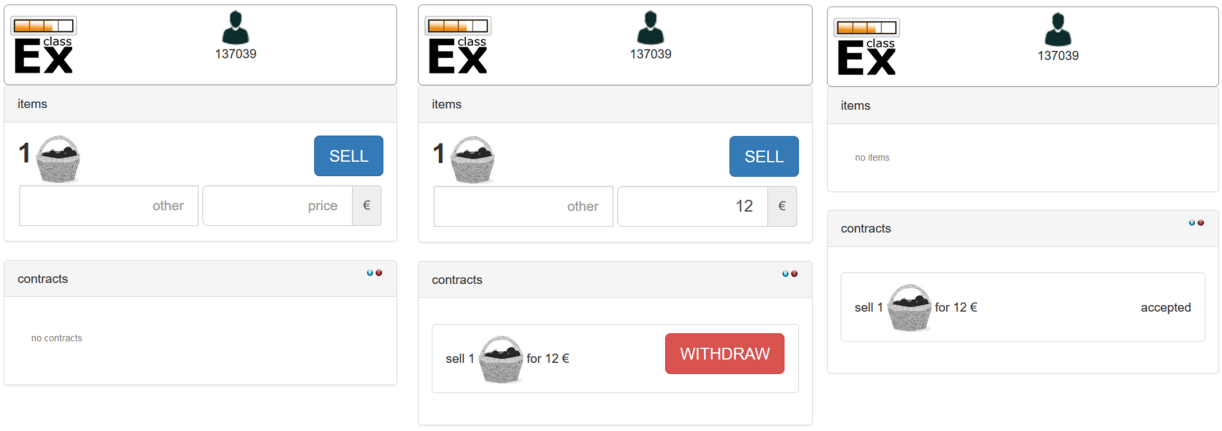
2.19 Numeric Decisions

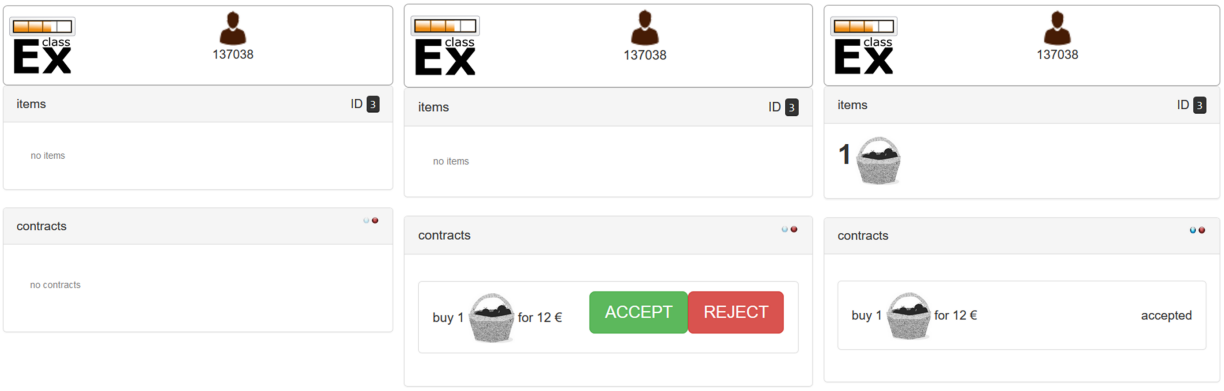


Numeric decisions can also be made by entering a number and pressing the submit button. If the input exceeds a predefined maximum or minimum, the participant has to redo his or her input. Besides minima and maxima, you can also specify the number of digits and whether entering an input is mandatory. For further information, see *Elements*.

2.20 Contracts

Participants can trade items in classEx. Therefore, they walk around in class to find a trading partner. If they found a trading partner, they have to conclude a contract in the following way.





The upper row shows the different steps for the seller, the lower row shows the different steps for the buyer. In this case, only sellers can send offers to buyers. This can also be changed in the *Elements* settings.

Each screen consists of two fields. The item field shows all items which a participant possesses. The contract field shows all contracts and offers.

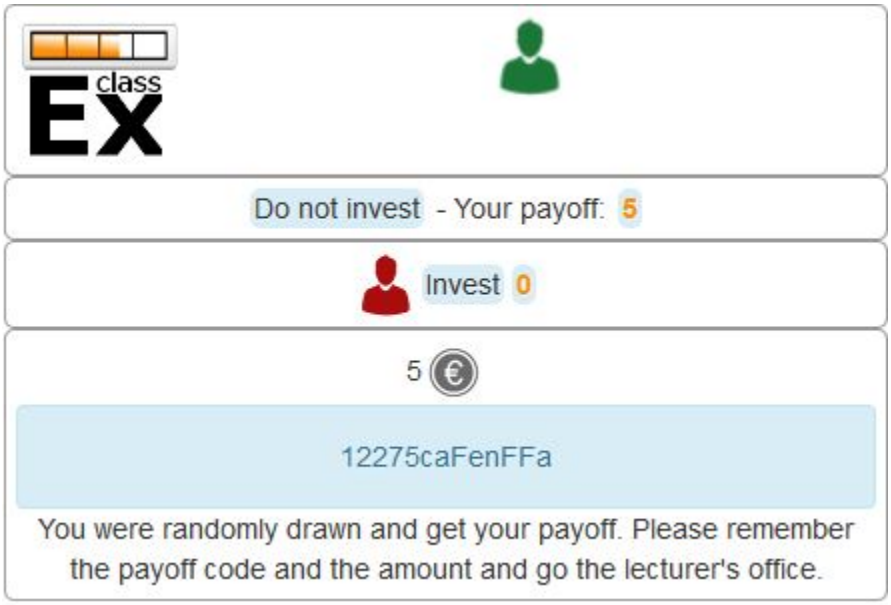
First, the seller asks for the ID of the buyer, which is displayed on the buyer's screen. In this example, the buyer has the ID 3. The seller enters the ID of the buyer and the price into the input field on his screen and presses *SELL*. Then an offer shows up for the buyer in the contract field. The buyer can accept or reject the offer. The seller can withdraw the offer. If the buyer accepts the offer, the item is transferred to the buyer and shows up in his or her item field. The contract is marked as accepted.

2.20.1 Disbursal of payoffs

In some games, participants will receive a real monetary payoff. The payoff is paid out by providing the participant with a payoff code.

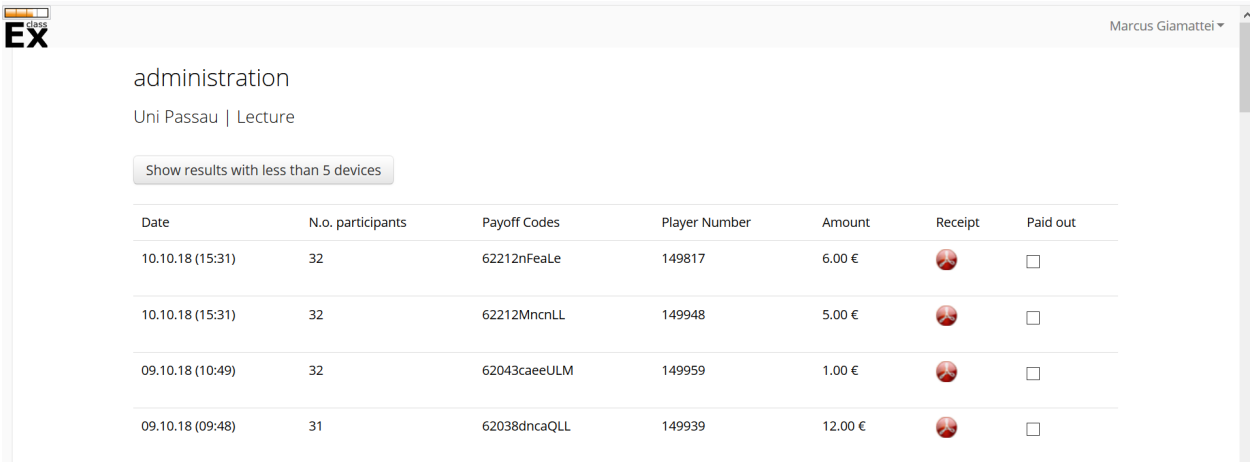
> **Note**
>
> Participants should not show their payoff code to others, as others could then claim the payoff. Therefore, it is advisable for participants not to let any other participant see the screen of their mobile device during the experiment.

> **Note**
>
> If a participant closes the browser, normally the payoff code is lost. Therefore, participants should take note of the payoff code or make a screenshot of it.



The participant can present the payoff code to the administrative staff after the end of the lecture in order to claim his or her payoff. The person entrusted with disbursing the payoff can log in to classEx with the user type "administration" (see *Login*). The payoff can also be disbursed directly by the lecturer. The password for the lecturer is always valid for the administration of payoffs as well.

If you log in as a user type "administration", you can see a list indicating the date, the payoff code, and the amount of money to be paid out to the participant. Further, clicking on the red icons opens a PDF with a receipt that can be printed out and signed by the participant. Also, the administrator can tick the box on the right indicating that the participant has picked up his or her payoff.



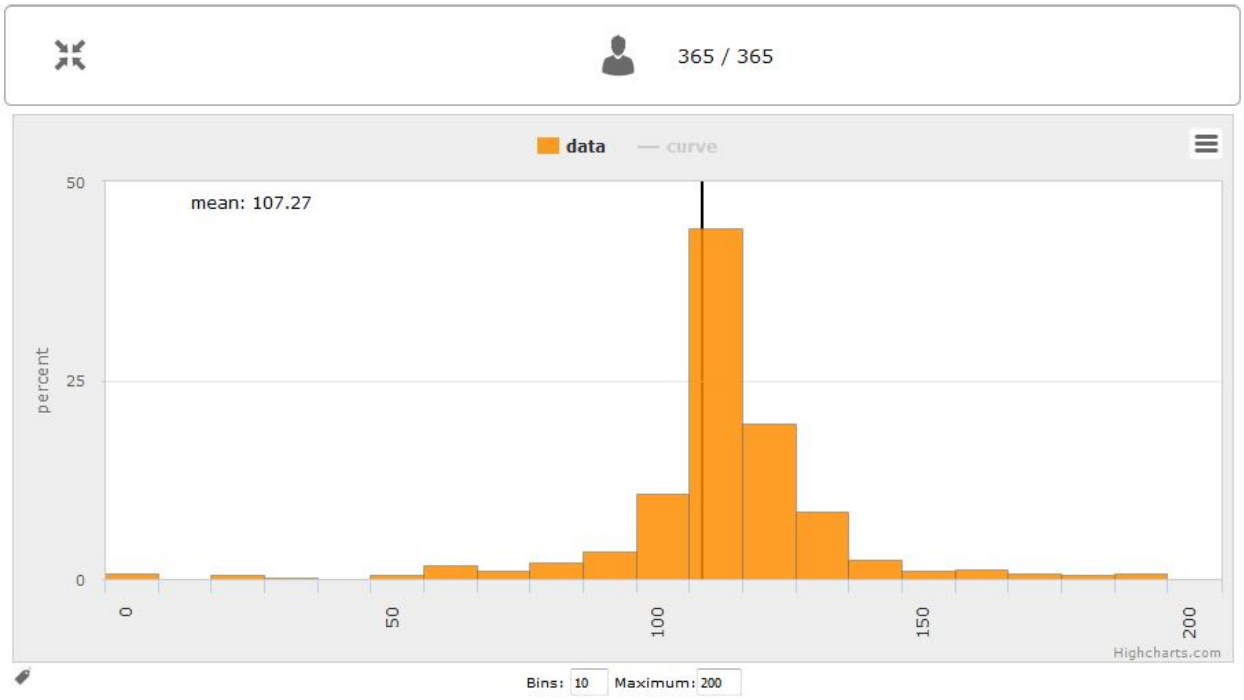
Note

The lecturer reserves the right to withhold the payoff in the event of error. classEx displays an automatic message which states: "In this game, payoffs could be made. The lecturer is responsible for the payoff, subject to a technical check for the correctness of the winning codes. Legal recourse is excluded."

> **Note**
>
> If a participant lost his or her payoff code, the list of payoffs also shows the internal player number of the player. If the participant logs in with the same device as he or she played the game, the player number can be retrieved by clicking on the classEx logo in the top right corner. This can be used then to verify that the person is entitled to receive the payoff.

2.20.2 Graphical results

At the end of a game, classEx displays summary results directly on the lecturer's screen. There are different result elements available like histograms, bar charts, line charts and pie charts. For a full list, see *Elements*. The figure shows an example of a histogram.



Some of the graphs have interactive features and can be adapted. All figures that are labelled with "Highcharts.com" (see the bottom right corner of the figure above) have a zoom function. You can zoom in by simply clicking and pulling the mouse over the section you want to zoom in on. The button "Reset zoom" resets the display back to the original size.

For histograms, you can also change the settings for the bins and the maximum by clicking on the little symbol under the bottom left corner of the chart. You simply change the values in the fields and then click beside the bins display. This can be useful if the default bin size was too small. The bins are then changed for all graphs.

Via the button *previous results* in the lecture menu, you can also access and display results (and their corresponding graphs) of previous sessions.

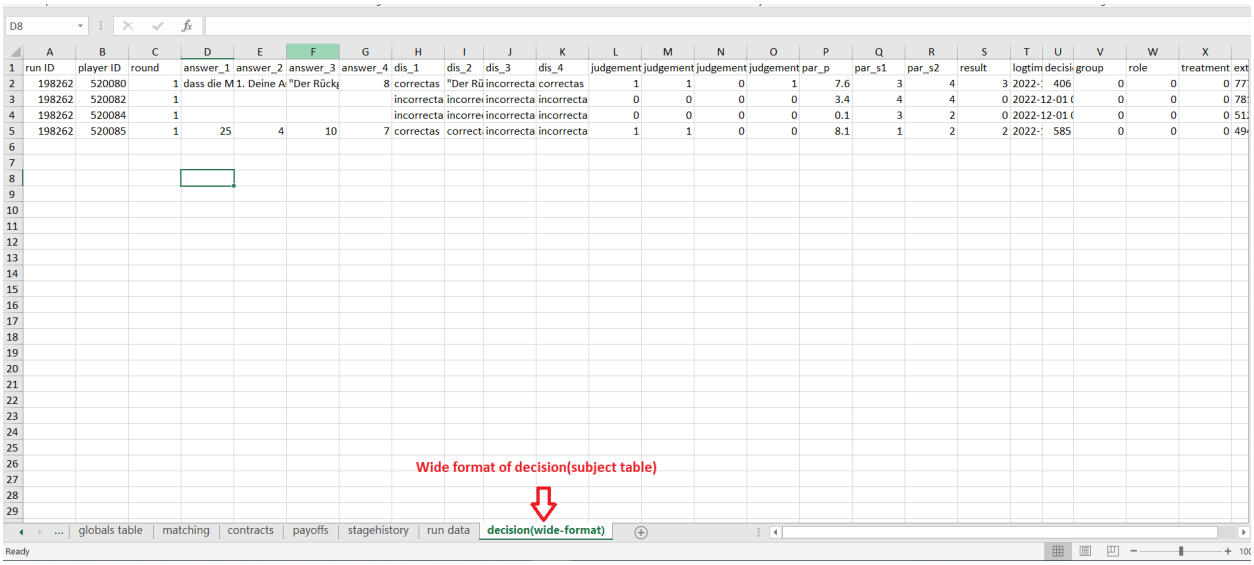
2.20.3 Data

2.21 Excelfile

The Excel files contain all variables recorded in the game. A file contains an overview of participants, decisions, contracts (if made), payoffs (if made) and assignment to roles, treatments & groups. The Excel file can be downloaded

at any time during the game and always shows all currently recorded data. It can also be accessed after a new game has started. In order to get previous data, just select the respective session from the previous results menu. When the game is open, you can download the results again via *download as excel*.

"Decision (Wide Format)", a new tab sheet, has been added to the run data Excel file. This format provides a broader view compared to the "Decision (Subject-Table)" tab, enabling easier analysis and comparisons by displaying data horizontally for a more streamlined overview.



2.22 xecon

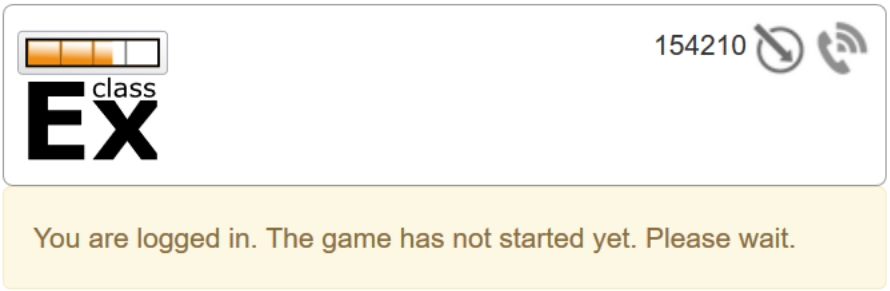
The data repository xecon provides long-term secure data storage. It is provided by gesis - Leibniz Institute for Social Sciences in Cologne. It allows for storing and publishing experimental data sets. Data sets can be directly exported to x-econ (including metadata) and can obtain a DOI to make them citable. The export is automated so that you only have to select which session to export. classEx creates a ZIP file, which is automatically transferred to xecon. You only need an account with xecon. All available metadata, like the number of players, groups, rounds,… is extracted automatically and put to xecon where they can be adjusted.

2.22.1 Dealing with problems

classEx runs smoothly with all standard browsers when JavaScript and cookies are enabled. Still, in rare cases, it may come to problems due to different mobile devices. If you encounter a problem, here are some strategies on how to deal with them. Typical problems involve subjects being unable to load the page or being stuck in the game.

In order to get more information if a participant cannot proceed, classEx offers some little helpers which can be displayed on the participant's device. Just click on the classEx logo, and some new buttons appear.

> **Note**
>
> This functionality is only for assistance in case of problems and should not be told to participants beforehand. Participants may use the functionality to log themselves out.



The new buttons are a logout button and a check for the internet connection. The internal ID of the participant is displayed as well.

2.23 Internet connection is slow

The most common problem which can arise is that the internet connection of the participant is too slow. If this is the case, it can happen that pages are not reloaded and the participant seems to be stuck in the game. Try another network or mobile internet.

You can check if the internet connection is working with the telephone symbol. If the connection works, the symbol should blink every 2-3 seconds. If there is a problem, instead the following symbol is shown .

2.24 Re-login of the participant

If a participant has problems which cannot be solved by a better internet connection or a reload of the web page, you can try a re-login. Therefore, you have to log out the participant by clicking on the logout button. Then, the participant should log in again.

> **Note**
>
> If the game is running and you have proceeded beyond the first stage, re-login may not be possible. If a participant tries to re-login, then he or she may get the message that the game is running and participation is no longer possible.

2.25 Check participant ID

Sometimes participants may claim that they clicked some button or made some input, but received different feedback. In many cases, this can also be a wrong perception by participants. But it could also be due to a programming error. To check this, you can download the data after the experiment and look for the respective (internal) ID of the participant. The ID of the participant is displayed when you click on the classEx logo on the participant's device. The ID can also be used if the participant loses his or her payoff code (see *Disbursal of payoffs*).

2.26 Other problems

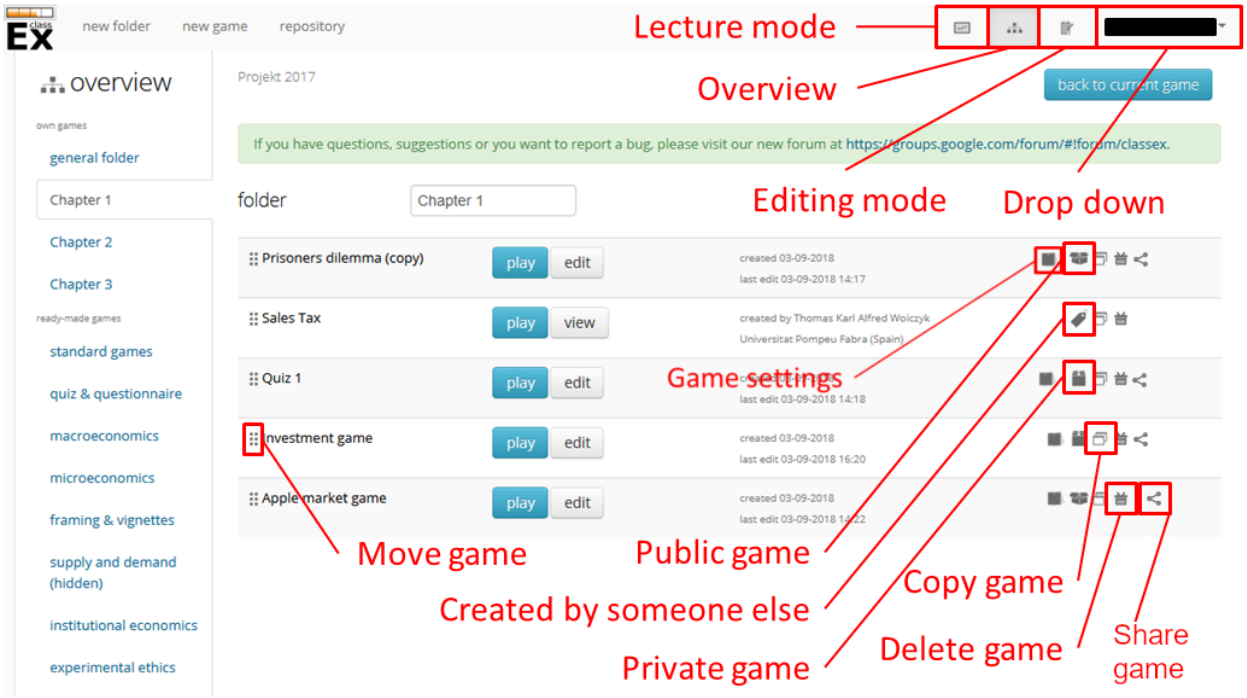
If the above strategies do not help, participants should try out a different browser. We always suggest using Mozilla Firefox. If you have persistent problems with a special browser, please let us know via classEx@uni-passau.de. For error retrieval, please tell us the ID of the participant and the date of your course.

CHAPTER

THREE

ORGANIZE GAMES

You can organize your games in the overview mode. Here, you can find all your own games as well as the ready-made games that are shown in your overview by default.



On the left-hand side, you find the folders for your own games (empty at first login) and the folders for ready-made games below them (standard games, quizzes and questionnaires, etc.).

In the top right-hand corner, you find the main navigation bar, which is always displayed in every mode. This allows you to switch between modes and access your personal data.

The currently active mode is marked by a darker shade around its symbol , here the **overview mode**. The left symbol takes you to the **lecture mode**. The right symbol takes you to the **editing mode**. The **drop down menu** (which shows the name of the lecturer and is shaded in black in the figure above) provides access to your personal data and your course data, as well as the terms of use, the documentation, some general info on classEx and the logout button.

3.1 Settings in the top bar

You can see the top bar for the **overview mode** in the picture above in the left-hand corner. It offers the following functionality:

new folder
: Create new folders to organize your games.

new game
: Create new games which you can design in the editing mode.

repository
: The repository, which provides access to games provided by other lecturers.

3.2 Folders

Games are sorted into folders, which can be named for example, after different chapters of lectures. They can be found on the left. You can create a new folder by clicking on *new folder* in the top bar. Folders can be renamed by clicking on the title of a folder in the main screen (above the list of games). This is not possible for the general folder that is provided by default. Folders are ordered alphabetically. It is not possible to change the order of folders.

> **Note**
>
> If you need a specific order, you may name the folders, e.g. 01 - intro, 02 - motivation, ...

Folders can only be deleted when they are empty. Then the delete symbol will appear. The deletion cannot be reversed.

Each account has a standard folder, which is called *general folder*. This cannot be deleted.

3.3 Order games

You can move games by clicking on the *move game* icon and dragging the game into another folder (in the left bar). You can also drag the game within your folder in order to change its position.

3.4 Play and edit games

If you click on the *play game* button, the game starts directly in the lecture mode.

> **Note**
>
> If you click on *play game*, the current game which is running will be stopped, and the game you clicked on will be started. If you want to switch back to your currently running game instead, use the buttons in the top navigation bar to go the lecture mode or click on the button *back to current game* below the navigation bar.

You can directly enter the editing mode by clicking on the *edit game* icon. The title of this button changes to *view* if you cannot edit the game. This may happen if you are not the owner of the game, but you imported the game (see below).

> **Note**

You can also switch to the lecture and editing mode by clicking on the symbols in the top navigation bar.

3.5 Public, private and imported games

All games are marked as *public*, *private* or as *created by someone else*. The *public game* icon means that you created the game, but other users can see and copy this game. The *private game* icon means your game is private and cannot be seen by other users. You can change the game status between private and public by clicking on the symbol. The *created by someone else* marks the ready-made games provided by classEx and all games you imported from the repository. You cannot edit them, but only view and play them. If you want to make your own version of such a game, you have to copy it.

3.6 Copy and delete games

If you want to make a copy of a game (e.g. to create your own version of a ready-made game), click on the copy button . This will create a copy of the game in your *general folder*. You can edit this copy and thus make your own version of any game in classEx. Alternatively, you can go to the editing mode and click on the drop-down menu *game* and then select *copy game*.

The *delete game* icon deletes a game. This can't be reversed.

3.7 Share games

By clicking on the sharing symbol, a sharing link will be created and copied to the clipboard. You can share this link with other users. The other user has to be logged into their classEx account first and then paste the link. Then a link to the shared game is added to the general folder, and the other user can test the game.

Note

Only a link to the game is copied to the other user, not the game itself. The game remains in the ownership of the original user. The new user can make a copy of the references game if they want to edit it.

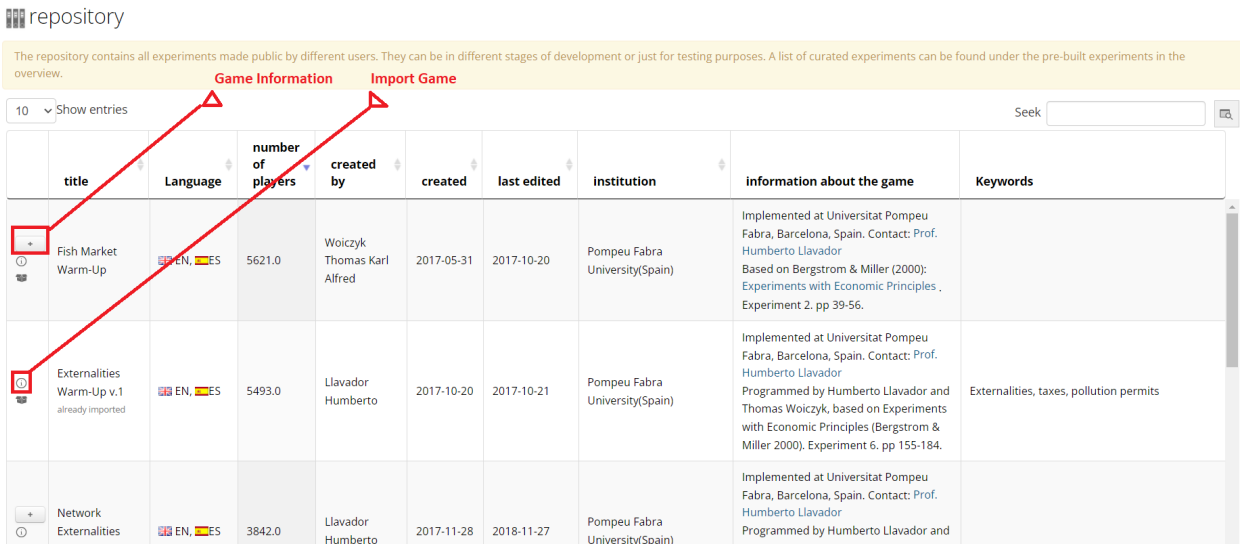
Note

You can only share your own games.

3.8 Repository

The repository lists all public games created by you and other lecturers and allows you to import these games to your account. You can enter the repository by clicking on the *repository* button in the top bar of the overview.

If you do not want your game to be listed in the repository, you have to change its status from public to private.



3.8.1 Search games

The repository lists all games which were set to public by their owners. It provides easy access to all available games, enabling you to search for and, when needed, import public games created by other users into your own panel. The list has been optimized for improved search speed and usability, ensuring faster performance. Additionally, the search box offers comprehensive searches across all fields, enhancing efficiency and providing a streamlined user experience.

3.8.2 Import games

All games from the repository can be imported to one's own account by clicking on the *Import game* button. You can then find the game in the general folder.

> **Note**
>
> If you import games, they are not copied to your account. classEx only sets a reference to the game of the other lecturer. Therefore, you cannot modify imported games. In order to modify an imported game, you have to copy it.

CHAPTER

FOUR

PERSONAL AND COURSE DATA

4.1 Personal data

If you click on the drop-down menu in the top right corner (which states your name), you can access your personal data.

4.1.1 Change personal data

There you can change your title, your name and your email address. Please make sure that you always have a valid email address. Otherwise, we cannot contact you. You can also change your password here.

Your affiliation is shown in the personal data as well. It cannot be changed. If you want to change your affiliation, please email us at classEx@uni-passau.de

> **Note**
>
> Please keep your personal data up to date. For data protection, have a look at our terms of use.

4.1.2 Password forgotten

If you have forgotten your lecturer's password and you cannot log in, classEx provides a function to get a new password. At the login screen, select your course and select "experimenter/lecturer" and click on the small text "get a new password" button underneath the login button. You only need to enter your email address with which you are registered. Then you will get a new password via email.

4.2 Course data

If you click on the drop-down menu in the top right corner (which states your name), you can access your course data.

4.2.1 Change course data

4.2.1.1 Course title

In the course data, you can change the title of your course, which also appears at the login screen. Course titles can be the name of the lecturer or the title of the course. It should be easily recognizable to participants in order to facilitate their login.

> **Note**
>
> Please make sure that you do not use the same title as another course at your institution.

4.2.1.2 Participants password

If you want to change the password, the participants need to log into your game - for example, to pick a password that is easy to remember - you can also do this in the course data. This also works in case the lecturer has forgotten the password he or she picked or received with the login credentials.

> **Note**
>
> The password is only changed once you log out and log in again. Please do not use an empty password, as you can not use test participants with an empty password.

4.2.1.3 Course language

You can also switch the language of the course in the course data. ClassEx provides support for German, English and Spanish. The latter is only available for the participants' interfaces and the lecture mode. The overview and the editing mode will still be in English in this case.

> **Note**
>
> The course language may differ from the language of the game. If an adequate translation is available, classEx uses the translation. Otherwise, it will look for an English version of the game. If this is not available, the game will be displayed in its original language.

> **Note**
>
> If you change the language, changes only apply if you log out and log in again. The same holds true for participants and test participants. Participants are not affected by these changes while a session is running.

4.2.1.4 Currency

The currency can be switched in the course data interface, too. Available currencies are Euro, Dollar, Pound, Yen or no currency.

4.2.2 Log out all participants

You can log out all participants by pressing the following button.

> **Note**
>
> This may be necessary if you run to lectures in a row and do not want to have the participants from the first lecture in the second one.

4.2.3 Additional settings

By clicking on "additional settings", you can change two default settings for the participants' login. In the first field, you can customize the message participants see when they are logged in, but the game has not started yet. If you leave the field empty, the following will be displayed to participants: "You are logged in. The game has not started yet.

Please wait.” In the second field, you can enter an ID. This will give you the possibility to track participants as they log in. If you enter a word in the ID field like “Name”, any participant logging into your game will be displayed the words “please enter Name” followed by a field to enter their answer. It is also possible to create two or more such fields by entering more words in the field in coursedate and separating them with a “,” like: Name, Studentnumber

Note

You can use this, for example, to make students enter their student number to rate their games and give credits or bonus credits to them for how they performed in the games during the term.

You can display the results in the data section while or after you played a game. You find the data section in your top bar while you are in the game in the lecture mode. If you download the data as an Excel file, you will find the column “external ID” with the answers the participants gave when logging in.

CHAPTER

FIVE

TUTORIAL: HOW TO DEVELOP A SIMPLE GAME

To help you get started with creating your own games, this section gives step-by-step instructions for how to implement a so-called Ultimatum Game.

5.1 The Ultimatum Game

In an Ultimatum Game, the participants play in groups of two. One of them takes the role of the proposer. He gets an initial endowment and decides upon how to divide it between himself and the other participant – the responder. The responder then decides whether he accepts the proposal. If he does so, the final payoffs of the participants are according to the proposal. If the responder does not accept the proposal, both participants receive nothing as final payoff.

5.2 Building the game

To create a new game, click on "new game" in the top bar in the overview mode. Type in a game name of your choice and change the availability to "private". Thus, other users cannot see "your" Ultimatum Game. Click "save". ClassEx will automatically switch into editing mode, and you can start building the game.

5.2.1 Assignment & Matching

Go to the tab "assignment & matching". Select "role & group" in the drop-down menu "assignment" and select "2" in the drop-down menu "number of roles". This means the participants in this game are selected into groups of two, with every group consisting of one participant of each role. Leaving "partner" as the matching mode means the groups stay together over several rounds of the game.

5.2.2 Stage 1: Proposer's decision

Go on to the tab "stage 1". In this first stage of the game, the instructions for all participants should be on the lecturer's screen (all participants can see it), and the proposers should decide how much they want to transfer to the responder. The stage should be started by the lecturer pressing the start button. In the first stage, we don't have to think about the responder whose decisions will be designed in stage 2.

First type in a name for the stage, e.g. "UltimatumProposer". "Late arrival" should be "possible". That means participants can still log in after the stage has been started by the lecturer.

> **Note**
>
> After writing the name of the stage into the empty field and clicking into another field, you might have noticed that the page refreshed itself. Doing this automatically saves every change you make. If you pause designing your game, you can just click into another field and make sure everything is saved. Later, you can get back to where you left via the overview mode. In the default screen of the overview mode, the general folder will be opened. There you find your created game and can open it by clicking on "edit".

The start button is already implemented by default in the first stage in the lecturer field on the right side. To add the instructions, click on “add new element” in the lecturer field and select “text box”. Click on the little symbol “paste element” on the right side above the start button field to insert the instructions above the start button.

> **Note**
>
> If you want to move an element, you can do this by clicking on the scissors symbol in the element field you want to move. Doing this, you cut out the element. You can insert it again in the spot you like by clicking on “paste element”.

Change the selection in the dropdown menu in the text box from “display always” to “before start”. Now insert the game instructions that should be visible for all participants. An exemple for those instructions could be: “You play with another participant here in the lecture hall. One player in each group gets 10 € at the start of the game. This person is called “the proposer” and decides how to divide those 10 € between you. Then the other person, called “the responder”, can decide to accept or reject the offer. If the responder accepts the offer, both will get this payoff. If the responder rejects, both player will get nothing.”

> **Note**
>
> If you want to have further options for text-editing, you can press on the small notepad icon on the left side of each text field. You can leave again from there by pressing “x” in the top left-hand corner of the box.

Now we start editing the participant’s field on the left side. An input element with one input field and some places to insert text is implemented by default. Change the “Type of input field” to “Numeric input field”. Every participant in the role of a proposer will be able to decide upon the amount they want to keep for themselves by entering it into this field. The field “variable names” next to the “Type of input field” defines the variable name. This is the internal name of the variable and will not be visible to the participant. Type in e.g. “keep”. Only the proposers should have an input field in this stage, while the responder has to wait for the offer. Therefore, change the field “for all roles” to “only role 1”. In the text field, you can insert a description of the input value that is visible for the participant. Type in e.g. “For me:”. Below the text field, you can edit some more settings of the input field. The “Minimum” should be “0” because the responder cannot keep a negative amount. The “Maximum” should be equal to the initial endowment because the proposer cannot keep more than this for himself.

> **Note**
>
> At this point, it could be helpful to add fields calculating the amount of money a proposer sends to the responder. These fields could dynamically display this information to the proposer. When the proposer would enter a “5” in his “send” field, classEx could display “The amount you send to the responder is ‘5’”. Although this calculation is easy for any participant, it ensures they are always aware of the game mechanism and do not make a calculation mistake. You will learn, e.g. how to insert such calculation fields in the chapters on creating your own games following this tutorial.

The initial endowment could be a parameter you maybe will want to change in the future. You should only need to change one value to do so. Therefore, you should define the initial endowment as a general parameter of the game. To do so, you need to add a “program code (subjects)” element (participants field -> add new element -> program code (subjects)). Click on the little paste symbol above the input field to insert the program field. In the program field, you can now define the initial endowment as a variable. Type in “$endow = 10;” for an initial endowment of 10.

Now you can use this variable to define the upper threshold of the amount the proposer can keep (if you deviate from “10”, remember to also adopt your instructions in the lecturer text box). Type “$endow;” into the “Maximum” field of your Numeric Input Field. Type in “0” into “decimal place” to not allow for decimal numbers and define the “unit”, e.g. as “€”.

For clarification, you should add a more general explanation of the stage for the proposers that is displayed above the input element. Click on "add new element" in the participants field and select "text box". Click on paste between the "program code (subject)" and the input element. Again, change the field "for all roles" to "only role 1". Then insert the instructions, e.g. "You decide how to divide $endow; € between you and participant 2. Participant 2 decides if he accepts or rejects. If he rejects, both of you get nothing. If participant 2 accepts, payoffs will be according to your proposal."

Note

What have we done by now? We are done with the assignment & matching and the first stage. So after logging in, participants are assigned to groups and roles. The instructions get displayed to both the proposer and the responder. We have a start button and everything prepared for the proposer to participate in the game. In the next two steps, we will model the decision of the responder, displaying the results and ending the game.

5.2.3 Stage 2: Responder's decision

In the second stage, the responders are informed about the proposals, and they decide whether to accept or reject.

Also, the second stage is already provided by default. Type in a name for stage 2 (e.g. "UltimatumResponder"). "Late arrival" should be "not possible" in this stage, because partners are already matched and newcomers cannot be integrated once the first stage has been played. The first thing we do is to inform the responder about the proposal. To do so, you need a "program code (subjects)" field (-> add new element -> program code (subjects)). Change "for all roles" to "only role 2". Type in the following code:

```
$keep = $findVariablePartner("keep",$round);
$receive=$endow-$keep;
```

The first line defines a variable "keep" and assigns to it the value of the participant's matching partner's "keep" variable. The second line calculates how much the receiver gets and assigns the value to a variable "receive". Now you can use both new variables to inform the responder about the proposal made to him. Therefore, we need to create a new text box in the participants field below the program code field (-> add new element -> text box -> paste element). Change "for all roles" to "only role 2" in the text box and type in the following instructions:

```
Participant 1 has decided to split $endow; as follows: $keep; for participant 1 and
↪$receive; for you. You can accept the proposal or reject it. If you reject it, both␣
↪get nothing.
```

Now you need an input element via which the responder can accept or reject the proposal. Insert an input element beneath the text box and insert a "new input field" within the input element. As the responder can only decide between "Accept" and "Reject", we change the type of input field to "Buttons (Single Choice)". Set the variable name to e.g. "accepted" and define the input field as visible for "only role 2". Write a text into the text box that should appear above the "accept" and "reject" buttons (e.g. "Your decision"). To insert these buttons, type "2" into the text field next to "add new possible answer" and click on the little plus left of it. Insert "Accept" and "Reject" into the new text fields. The values assigned to the decision buttons are very important. Choose the value "1" for the accept button and the value "0" for the reject button.

The second stage should start for a responder automatically as soon as "his" proposer has sent a proposal. Therefore, delete the "results" field in the lecturer field by clicking on the rubbish bin icon in the top right corner of the field. Then insert an "automatic start" via "add new element". Change the mode to "wait for others". To display how many proposers and responders have already made their decisions on the lecturer's screen, set the counter to "display" and the count to "by role".

5.2.4 Stage 3: Results

When the responders have accepted or rejected the proposals, you can display the results in a third stage. Add a new stage and name it, e.g., "Results". "Late arrival" again is "Not possible". The two fields next to the "late arrival" field define how often stages get repeated and where to jump after finishing this stage. Using this, you can define the number of rounds you want to play. Choose "back to stage 1" and e.g. "2x" (for repeating the stages two times).

For both participants, the payoff depends on whether the responder accepted the proposal or not. You have to distinguish these two cases. To do so, you use two program code (subjects) fields in the participant field. Insert them above the default text box. You need one for "only role 1" and one for "only role 2".

The program for role 1 is:

```
$accepted=$findVariablePartner("accepted");
$payoff=$keep*$accepted;
if($accepted==0) {
$text="Participant 2 has rejected your proposal.";
} else {
$text="Participant 2 has accepted your proposal.";
}
```

The program for role 2 is:

```
$payoff=$receive*$accepted;
if($accepted==0) {
$text="You have rejected the proposal.";
} else {
$text="You have accepted the proposal.";
}
```

Afterwards, insert two text boxes in the participants' field. Again, one for role 1 and one for role 2. In these text boxes, you inform the participants about their final payoff. For role 1, the text could be:

> You proposed to keep $keep; € from the initial endowment $endow; €. $text; Your payoff is $payoff; €.

For role 2, the text could be:

> Participant 1 has proposed to split $endow; as follows: $keep; € for him, and $receive; € for you. $text; Your payoff is $payoff; €.

In the lecturer field, you can show the results. Delete the start button that is implemented in a new stage by default. Then add a results matrix element. Change "decision role 1" from "stage 2 # 1" to "stage 1 # 1". Change "count" to "by role" and "display results" to "by round".

5.3 Testing the game

Congratulations! You just finished designing your first own game!

To test the game, change into lecture mode. If you have already started another game, your self-designed game won't open automatically. In this case, you can just "close" the running game by clicking on "select game" in the top bar and picking your "own" game. You can test the game on your own PC without other devices by clicking on "new test participant" in the top bar of the lecture mode. This opens a participant screen in a new tab. You will see the game just as your participants will see it when actually playing the game. You can open as many screens as you want, where each screen represents a participant. After opening enough test participant screens, click "Start" in the lecturer screen. Then you can go through the game with all the test participants.

CHAPTER

SIX

DEVELOP YOUR OWN GAMES

In the following three chapters of this documentation, you will learn how to design and implement your own games. The chapter “Develop your own games” will begin with the basic functions of the editing mode and explain some patterns that every game in classEx follows, such as the way participants are assigned and matched when the game starts. The chapter “Elements” describes all the different elements you can use to build your own game. To avoid the need for experimenters to learn a lot of programming, classEx games are designed to be modular. Thus, every element is like a brick you need to construct your game. The last chapter, “Programming”, shows how to create more powerful, dynamic games by combining already existing elements with some PHP programming.

To develop your own games in classEx, change to the editing . In the editing mode, you can create games according to your own needs. Games can be clicked together with an easy-to-use, modular backend system. You divide your game into stages, and you can add different elements (input, output, calculation,...) to your games.

Note

In the editing mode, changes are stored automatically. If you change an element of your game and click next to the element, the element is stored. Most of the elements blink green when they are stored.

Note

If a game has already been played by at least 10 participants, it cannot be adapted any more. The same applies if the game was created by another person. You can, however, copy the game and then adapt it.

6.1 Settings in the top bar

The top bar in the editing mode looks like this:

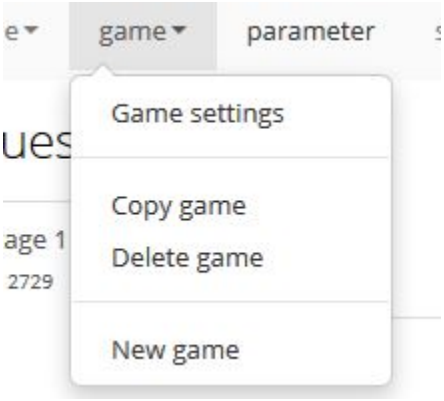
It provides you with several options, which are described below.

6.1.1 Select game

Click on the button in the left corner to open the drop-down list of your existing games. Clicking on a game will open the selected game in editing mode. If you open the editing mode, the currently running game is preselected.

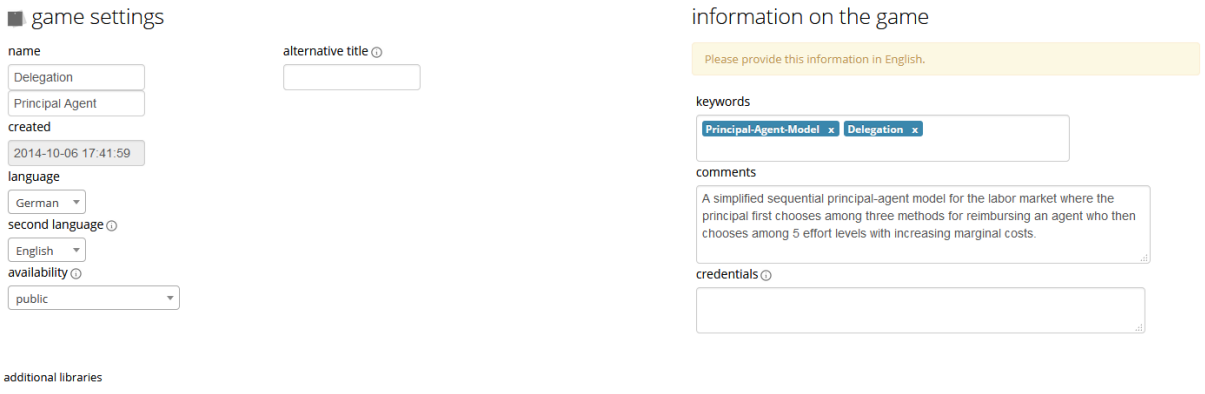
6.1.2 Game

By clicking on *game*, a dropdown menu will open, which shows up to four possible options. The number of options is reduced if you do not own the game. Then, e.g. you cannot delete the game.



6.1.2.1 Game settings

Clicking on game settings, a new screen appears. On this screen, you can change different settings of the game. You can also access this screen from the overview by clicking on the ■ symbol next to the name of a game. Changes to any field are again saved automatically.



Name of the game

First of all, you can specify the name of the game. If your game is available in different languages, you have to provide a translation of the title as well.

Alternative title
You can also define an alternative title, which is displayed instead of the name wherever the game is listed in your own account. Other users will see the original name of the game and not the alternative title. .. note:: This feature is useful if you imported a game which has the same name as one of your games. Then you can use the alternative title to distinguish from your game, because you cannot change the original name of the game, which belongs to another user.

Creation date
The created field shows the creation date of the game, which is not editable.

Language
For each game, you can specify a primary language and optionally a second language. If you add a second language, all text fields will be shown twice for both languages.

> **Note**
>
> If a text field misses the multi-language feature, you can input the text in both languages separated with $$. One example would be: Please decide now$$Bitte entscheide jetzt.

> **Note**
>
> classEx always chooses automatically which language to use based on the account language. If the game is available in the account language, it uses this language. Then it tries to find an English version. Otherwise, it displays the available language.

Availability
You can specify whether you would like a game and its results to be public or not. By default, all games (and their results) are public. You can set a game (and its results) to private, or you can set only the results to private and the game to public. In the latter case, other experimenters can still copy your game, but they do not see the previous results by you. If you decide to set games or results to public, they become subject to a creative commons license as described in the terms of use. You can change the setting of public and private also in the overview by clicking on  for private or  for public next to the name of the game. The option to set only the game public, but results private, is not available there.

If you set your game public, it can be found in the *Repository* and others can play and copy the game. If you set your game private, it is not listed in the repository anymore, but previously made copies of your game will remain with their owners and are not revoked.

Additional libraries
If you click on *additonal libraries*, new settings appear. You can select to load different libraries for the participants. Libraries are packages for special usages. They come with an increased necessity to load data from the server (for each participant). Therefore, you should only turn them on if you need them. For lecturers, they are loaded automatically. The three available libraries are plotly or highcharts for drawing graphs on the participant screen. The library phaser can be used for creating game-like interactions. For more information, visit the respective website of the library by clicking on the library's name.

6.1.2.2 Information on the game

Here you can classify your game and provide meta-information on the content. This information can be accessed by other users and provides them with more details on your game. Please provide this information in English.

> **Note**
>
> classEx promotes the idea of sharing games. Therefore, it is important to provide meta-information on games so that they can be found easily. Another advantage is that you can transfer your meta-information directly to the data repository:ref:*Run:Data*.

Keywords
: You can provide a set of keywords to better describe your game. Many standard keywords for typical games are offered automatically when typing in some characters. Keywords are shown in the repository.

Comments
: In the comments section, you can provide a brief description of your game. Comments are shown in the repository, and if others import your games to their account, it is shown in their overview.

Credentials
: This field can be used to state a reference or source of your game. This will be shown in the repository and displayed in the lecture mode below the title of the game.

6.1.2.3 Copy game

If you click on *copy game*, the currently selected game is copied and can then be edited and adapted.

> **Note**
>
> The difference between copying and importing is that with the latter classEx only sets a reference to the original game. Therefore, it cannot be modified, but only used. A copied game, instead, is a complete copy of the original game and can be changed.

6.1.2.4 Delete game

By clicking on *delete game*, the currently selected game is deleted. For your safety, you will be asked if you really want to delete the game. It is not possible to delete the game if it has already been started in the lecture mode. You then need to start a different game in the lecture mode before being able to delete the selected one. You cannot undo the deletion of a game.

> **Note**
>
> If you accidentally delete a game, please email classEx@uni-passau.de as soon as possible. Internally, we completely remove deleted games only each month, so that recovery is possible.

6.1.2.5 New game

Clicking on *new game* creates a new game. A standard new game is always a single-choice question with four possible answers. Before you can edit the game, classEx takes you to the *Game settings* of the created game, where you have to provide a title. You have to select a language and choose whether the game should be public or private. Once you are done, click on *save* to create the game. classEx automatically takes you to editing mode, where you can proceed with designing the game.

6.1.3 Parameter

If you click on parameters, you can edit the parameters of a game. Parameters are global variables that can be changed right before starting a game. They allow other lecturers to run your game without changing the implementation of the game. More information can be found under *Parameters*.

6.2 Test a game

Before actually using a game in your lecture or while you develop, you can always test a game. To do so, switch to the lecture mode and select your game if it is not selected yet.

Next, open as many test participants as you need for testing your game by clicking on the *add test participant* icon  . This opens a participant screen in a new tab. You will see the game just as your subjects will see it when actually playing the game. You can open as many test participants as you want, which enables you to also test interactions between participants.

> **Note**
>
> If you use Chrome as a browser, you can open multiple test participants by holding the Ctrl-Key and clicking multiple times on the test participant icon  .

Then start your game. You can perform the interaction required in the browser tabs for each participant, and you can see how your game is running.

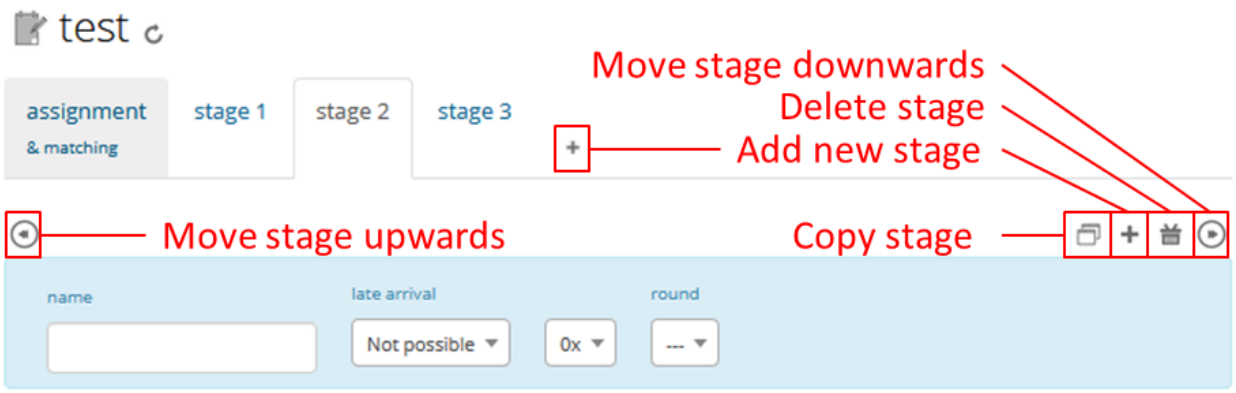
> **Note**
>
> Test participants are not reload-safe. This means that if you reload the page, in some cases, the content of the page may change. Real participants cannot do this.

If something is not working, go back to the editing mode and check your settings there. If you used variables and programs, you can use the *Diagnosis tool* for error spotting. The diagnosis mode shows all available variables and helps to debug them.

> **Note**
>
> All major browsers also provide their own development tools, which can be very helpful for error spotting. They provide a console which gives feedback on potential errors. In this console, you can also observe the background task performed by classEx and whether they are running correctly. Finally, it allows you to see JavaScript errors. In Firefox, the development tools are started by hitting F12.

6.3 Define Stages

Stages are points of synchronization in a game. Synchronization means that for the next stage to begin, all elements of the previous stage must have been finalized. Generally, the input phase is one stage, and the results phase is a different stage, as the output can only be displayed after all participants have entered their input. Stages are ordered in tabs in a horizontal way in classEx. The first tab is not a stage - it shows the options for *Assignment and Matching* of roles, treatments and groups.



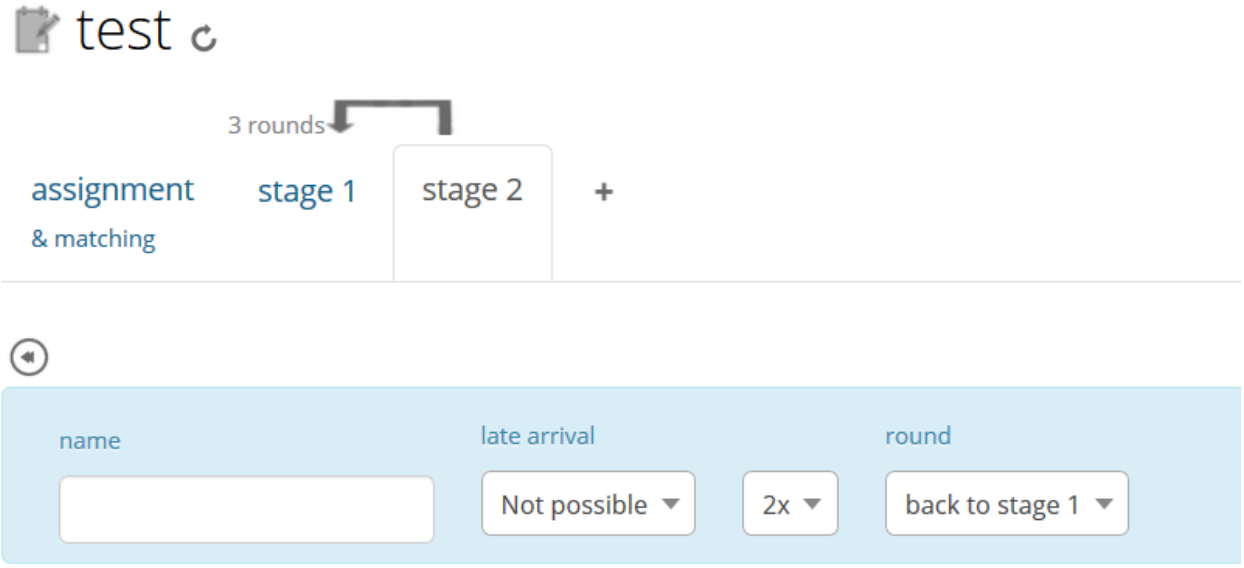
6.3.1 Name of a stage

You can choose to give the stages names instead of numbers in order to identify them more easily. To give them a name, simply enter it in the box. The name is then displayed below the stage number in the tab.

Note

classEx stores stages internally with a unique ID (which has between 4-6 digits) and neither with the name of the stage nor the number of the stage within the game (stage 1, stage 2,...). If you want to get the unique ID, just hover over the stage tab, and the unique ID will be displayed. This may be useful if you want to compare, e.g. results from the Excel sheet (see *Data*).

6.3.2 Rounds



If you want to run one or more stages more than once, you can define loops with a certain number of rounds. You can determine how often you would like to return to a certain stage. E.g. if you want to repeat stages 1 and 2 three times, you have to specify in stage 2 that you want to return two times (2x) to stage 1. With this, stages 1 and 2 are repeated three times, as shown by the arrow above the tabs, because you go through both stages a first time and then two times back to stage 1.

If you set the number to zero times (0x) or if the stage has been run for the predetermined number of times, classEx will continue to the next regular stage.

> **Note**
>
> You should only define one loop per game. Decisions and other variables are stored with the same variable name but with an increasing round number.

6.3.3 Late arrival

You can specify whether participants can arrive late, i.e. if they log in after the game has already started. You can choose for this to be possible, not possible, or only possible in the first round of a game.

> **Note**
>
> If you allow for late arrival in later stages, participants miss the first stages. Make sure that in this case, participants miss nothing which is necessary for later stages (e.g. declaration of variables,...). Assignment and matching are done in the first stage, so you should only allow later arrival if assignment and matching are not necessary.

> **Note**
>
> Matching is done on-the-fly. This means if a subject arrives late (only in the first stage), it will be matched according to your settings.

6.3.4 Move stages

When you create a new stage, this stage will automatically be defined as the next stage. You can move stages by pressing *Move stage backward* (left) or *Move stage forward* (right). The order in which the stages are run is always from left to right.

6.3.5 Add stage

You can add a new stage by clicking on *Add new stage* beside the tabs displaying the different stages or on the top right of the current stage.

6.3.6 Copy stage

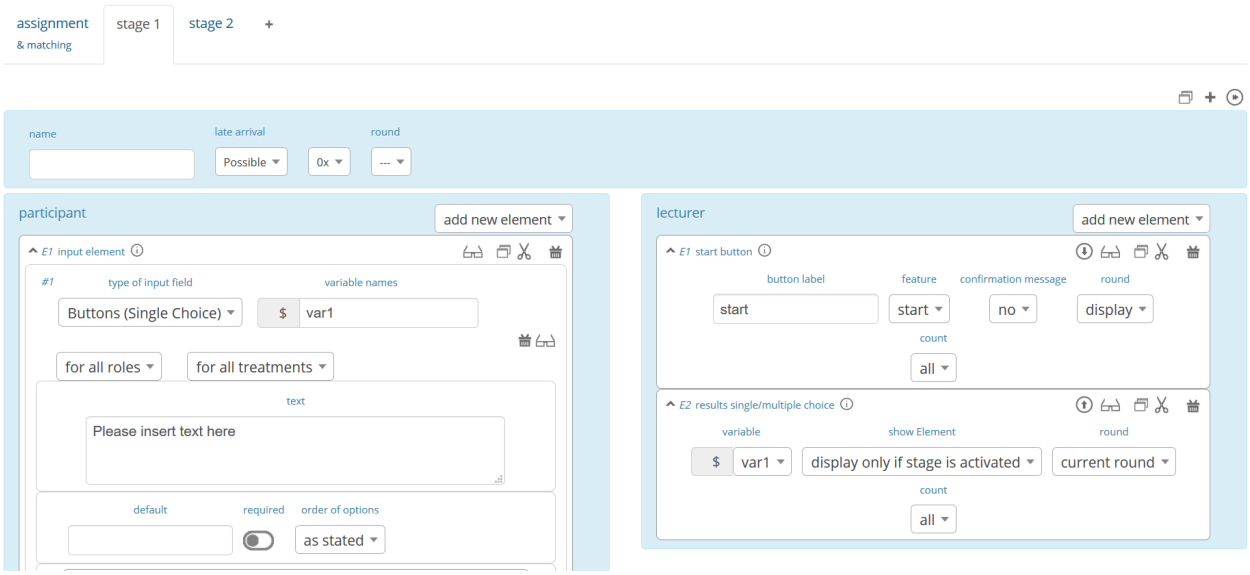
You can copy a stage, including all elements and settings of this stage. If you click on copy, the paste symbol will appear. The left icon can undo the copying. The right icon pastes the stage after the current stage. You can also copy stages across different games. Just copy the stage, open the new game and paste it there.

6.3.7 Delete stage

You can delete a stage by pressing *Delete stage*. Deletion can not be reversed.

6.4 Define Elements

Each stage consists of one or more elements. Elements are the modules of a stage. A stage has two areas in which you can add elements: participants and lecturer.



The left side shows the elements for the participant. Elements added here are displayed on the participants' devices. The program code (so-called subjects programs) added here is run for every single participant.

The right side shows the elements for the lecturer. Elements added here are displayed on the lecturer's screen in the lecture mode. Program code (so-called globals programs) added here is run once for all participants.

Some elements are the same for participants and lecturers (e.g text boxes), but most elements are different. Typical elements for participants are input elements, program codes, text boxes and winning notifications. Typical elements for lecturers are start buttons, program codes, text boxes and many different result elements.

Note

A full list of all elements can be found in the section *Elements*.

Note

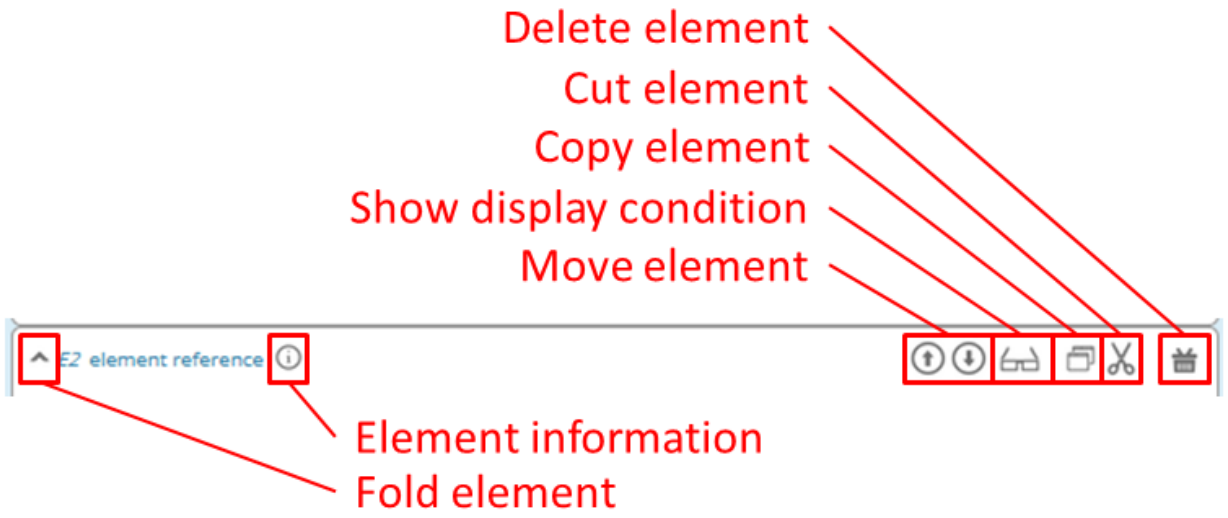
Elements are always displayed and executed from top to bottom.

6.4.1 Adding elements

You can add an element by clicking on *add element* and selecting the type of element you want to add. Note that there are different elements for lecturers and participants. After that, you have to choose where you want to place the element. The paste symbol will appear for every possible location of the element. Choose a location for your element by clicking on the corresponding *paste element* icon or cancel placing the icon by clicking on any *do not paste* icon . If there are no elements defined yet (for the participant or the lecturer), the element is automatically added to the first place.

6.4.2 Handling elements

All elements share some common characteristics which are described here. The details for every element are described in the section:ref:*Elements:Elements*.



Fold element
Each element can be folded to save space on the screen.

Element number
The elements are numbered (E1, E2, ...). This also defines the order of display in a stage. Elements can be moved within a stage with the *move element* arrows or by cut-and-paste.

Element type and help
Besides the number of the element, you can see the element type. Clicking on the info button next to the element type leads you to the respective description in this documentation.

Display condition
If showing the element should be conditional (e.g. not for every role or dependent on other variables), you can specify the display condition for an element in the code line that appears when you click on *show display condition*. The code will be evaluated as being true or false. Display conditions are defined in PHP. It should only contain comparison and should *not* end with a semicolon.

```
$round>2 || ($role==1 & $treatment==2)
```

You can combine different combinations with the and-operator & and the or-operator ||. The example only displays the element if the round is higher than 2 (so from round 3 on) or if the role is 1 and the treatment is 2 (independent of the round).

Copy element
You can copy an element by pressing *copy element*. Then the paste symbol appears on every possible location . You can copy and paste elements across all stages of the game.

Cut element
You can cut an element by pressing *cut element*. Then the paste symbol appears on every possible location . You can copy and paste elements across all stages of the game. This can also be used to order elements instead of moving them up or down with the move element arrows.

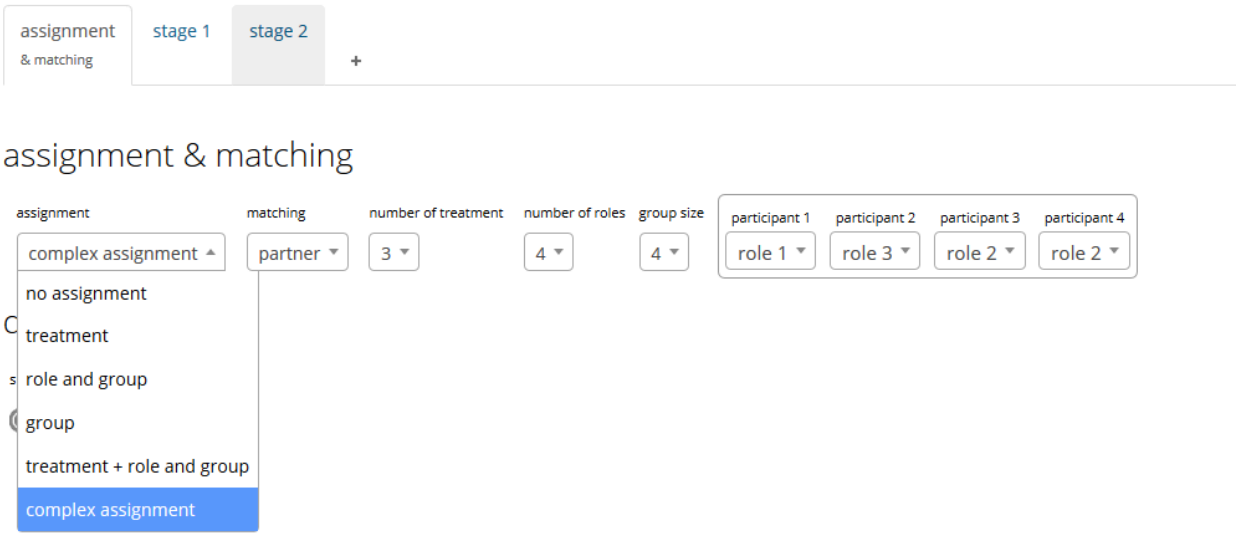
Delete element
You can delete the element by pressing *delete element*. You have to confirm a deletion. After that, a deletion cannot be reversed.

Groups, treatments and roles boxes
If you have defined groups, treatments or roles (more information about this in the next chapter), a box appears where you can also choose whether the element shall be displayed for all groups, treatments or roles or for special groups, treatments or roles only.

> **Note**
>
> Restricting the display and execution of elements for certain groups, treatments or roles can be done both with the boxes and with the display condition. With the boxes, you can limit the display to one group, role or treatment. With the display condition, you can combine different conditions or allow for multiple groups, roles or treatments.

6.5 Assignment and Matching

Left to the tab *stage 1*, you find the tab *assignment and matching*. Here, you can specify whether you want to assign participants to treatments, groups, roles or a combination of all (complex assignment).



Normally, the total number of participants is not known before the start of the experiment. This requires matching-on-the-fly. After they are logged in, subjects wait in a virtual lobby. Once the experimenter starts the experiment, the assignment of roles and treatments and the matching into groups takes place.

> **Note**
>
> The number of participants may not match the composition of groups of players that a lecturer seeks to assemble. For example, a mismatch would occur with an uneven number of participants in a game in which players must be matched into pairs of two. classEx offers different methods to deal with this. Either the decisions of other participants can be duplicated and matched to excess participants as a clone, or random decisions can be used. The experimenter can specify in the *Functions to retrieve variables* which method shall be employed. This makes sure that participants always get feedback, which can be important in order to avoid disappointing participants. As a further option, participants with no partner can be excluded from the game. Certainly, cloned or random observations may have to be deleted prior to using data for research.

6.5.1 Available roles

Up to 13 role symbols (and therefore roles) are available (and an additional grey role 0 for no role assignment). Role 1 is shown with a red figure. Role 2 is shown with a green figure and distinguishable by a different form to allow distinction for people who have red–green colour blindness.

These roles are standardized items and are shown in the header of the participant's page. If you want to display the role figure in a text box, just add role1.png to the text (for role 1). This will be replaced with the respective figure. **Make sure that role1.png is followed by a space; otherwise, the replacement will not work.**

6.5.2 Assignment at the beginning of a game

classEx allows you to flexibly adapt to an unknown number of participants, meaning that you choose the number of different roles, the number of treatments and the size of groups. classEx then assigns participants automatically. classEx allows lecturers to either set the size of a group (for example, two for a trust game or four for a public goods game), in which case the number of groups is automatically determined at the start of the experiment. Alternatively, an experimenter specifies the number of groups (for example, if each plays a different treatment), in which case their size is determined automatically. Specifically, you have the following available options:

no assignment
: Participants are all assigned to role 0, treatment 0 and group 0.

treatments
: This allows you to assign participants to treatments. A division into treatments will distribute participants evenly over treatments. You can select any number of treatments between 1 and 10. Treatments will be distributed according to arrival in the experiment (e.g. with two treatments, the first player will be treatment 1, the second treatment 2, the third again treatment 1,...). If you have defined groups as well, members of a group will always be assigned to the same treatment.

role and group
: This allows you to assign participants to a number of different roles in the game. Participants will be allocated to role 1, role 2, role 3... alternately. Participants will also be assigned to a group which contains one participant with each role. E.g. if you have defined 3 roles, a group will consist of role 1, role 2 and role 3. If you want to have groups with an asymmetric combination of roles, please use complex assignment.

group
: Allows you to assign participants to groups (all participants will have the same role 0). Groups are filled one after another. You are free to select any group size. The number of groups is determined automatically by classEx.

treatment + role and group
: Allows you to assign both role+group and treatments. It combines the two above options. Members of a group are always assigned to the same treatment.

complex assignment
: Allows you to assign participants to a different number of roles, treatments and groups. Again, members of a group are always assigned to the same treatment.

> **Note**
>
> The so-called between-subject design examines how a controlled variation of the game influences the behavior of different participants. This can be implemented using treatments. The groups in one treatment only interact with participants in their own treatment and never with participants of the other treatment. The game can be adapted for every treatment, for example, by providing different information, altered probabilities of random events or diverse strategic interactions.

6.5.3 Matching

At the beginning of a game, the assigned participants are always matched randomly to roles, treatments and groups. If your game consists of several rounds, you can specify how you want them to be rematched. You can choose from the following options:

partner
: Participants stay in the same groups and keep their roles throughout the entire game.

random
: Participants are randomly assigned to a new role, group and treatment (if specified).

> **Note**
>
> Absolute stranger matching, ensuring that participants never interact with participants they have interacted with before, is not available.

6.5.4 Random matching with constant roles

Random matching with constant roles means randomly matching the subjects into new groups at the beginning of each round, but at the same time keeping the subjects' roles constant. This is not provided as an option in classEx by default, but can be implemented manually as follows.

1. The assignment selected must be "role and group". It is also possible to include treatments. In this case, select complex assignment. The matching method selected should be "partner".

> **Note**
>
> The provided code only works for groups where each role is only used once. E.g. groups of 3 people need to have roles 1, 2 & 3. It is not possible to use the code for groups with, e.g. twice role 1 and one role 2.

2. You need to add a globals program in the lecturer field in the first repeated stage of your experiment and a subjects program in the participants field.
3. Insert the following code in the globals program:

```
unset($group_nr_a); //clears the array where group numbers are stored in the assignment␣
↪process

// This gives arrays with the internal participant ID and the role or treatment
$rolesarray = $getRoles();
$treatarray = $getTreatments();
if ($treatarray == null) { //in case only one treatment is selected, everyone is␣
↪assigned to one array
    foreach($rolesarray as $IDs => $roles) {
```

(continues on next page)

(continued from previous page)

```
        $treatarray[$IDs] = 1;
    }
}


foreach($treatarray as $IDs => $treat) { //in case there are different treatments, an␣
↪array with IDs and respective roles is created for each treatment
    ${"tr_specific_array_$treat"}[$IDs] = $rolesarray[$IDs];
}

$numberofroles = max($rolesarray);
$numberoftreatments = max($treatarray);


//this for-loop shuffles all the different treatment-specific arrays that have been␣
↪created and keeps the ID -> role connections
for($z = 1; $z <= $numberoftreatments; $z++) {

    $IDs = array_keys(${"tr_specific_array_$z"});
    shuffle($IDs);
    foreach($IDs as $internalplayerid) {
        $new[$internalplayerid] = ${"tr_specific_array_$z"}[$internalplayerid];
    }
    unset(${"tr_specific_array_$z"});
    ${"tr_specific_array_$z"} = $new;
    unset($new);
    unset($internalplayerid);
    unset($IDs);
}


for ($i = 1; $i <= $numberofroles; $i++) {
    $count[$i] = 1; //Initializing group count per role array
}

//new group numbers are assigned
for($z = 1; $z <= $numberoftreatments; $z++) {

    foreach (${"tr_specific_array_$z"} as $IDs => $roles){

        for ($i = 1; $i <= $numberofroles; $i++) {
            if ($roles == $i) { //If role fits
                $group_nr_a[$IDs] = $count[$i]; //Group assignment to group count
                $count[$i] = $count[$i]+1; //Increase group count for the role
            }
        }
    }
    unset(${"tr_specific_array_$z"});
    unset($IDs);
    unset($roles);
}
```

4. Insert the following code in the subjects program:

```
// reset the ID of the group with the new value generated by the globals program
$resetGroupNr($group_nr_a[$id]);
```

> **Note**
>
> You can find an example for the implemented code in the repository. The game is called "Assignment".

6.5.4.1 Further settings

On the page *assignment and matching*, you can further choose if the role should be displayed in the header of the participants' page and if the internal ID of the participant should be displayed there as well.

6.6 Additional CSS for participants

If you want to add additional CSS for participants, you can do so by clicking on additional CSS for participants in the top right corner. By clicking on the program box, a CSS editor opens, and you can add custom CSS. The CSS is loaded for every page of the participant.

6.7 Parameters

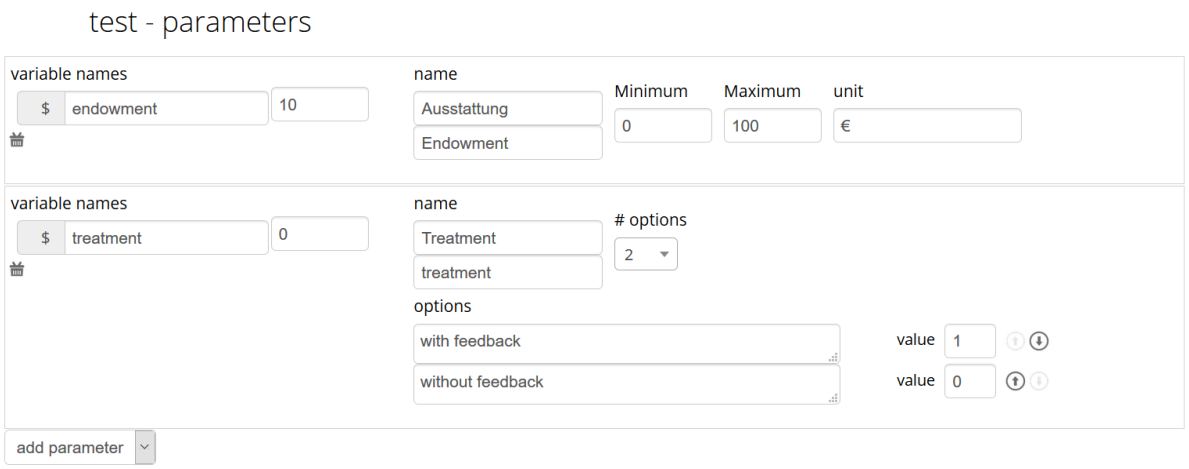
Parameters are global variables that can be adjusted in the lecture mode directly before starting an experiment. You can define parameters to enable adaptation of the game for lecturers without any knowledge of how to edit games. You can then play the same game several times with different parameters. This feature is very useful if you want to introduce some flexibility in the game.

> **Note**
>
> Parameters are initialized at the start of the game. They are read-only. They are the same for all participants.

You can define parameters by clicking on the *parameter* button in the top bar of the editing mode. Here you can see all defined parameters for the active game, edit them and add new ones. After adding a parameter, you can use it as a global variable in the whole game.

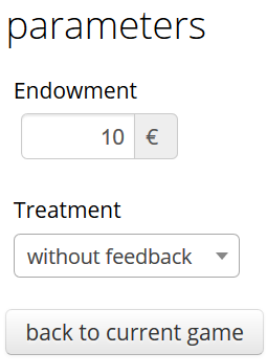
In the editing mode, parameters and their values are shown in the top right corner (if defined). If you display old results in the lecture mode, parameters are shown there as well.

Here is an example with a numeric parameter and a select list.



For *numeric parameters*, you have to define a variable name (here $endowment) and a default value. The default value is used if the lecturer does not change the parameter. On the right-hand side, you can define how the parameters are displayed to the lecturer (when they set the parameters in the lecture mode). They should not be bothered with variable names; therefore, providing an easy name is useful. If the game is implemented in two languages, you can provide two different names. Additionally, you have to provide a minimum and a maximum. You can also provide a unit (e.g. €, meters,...).

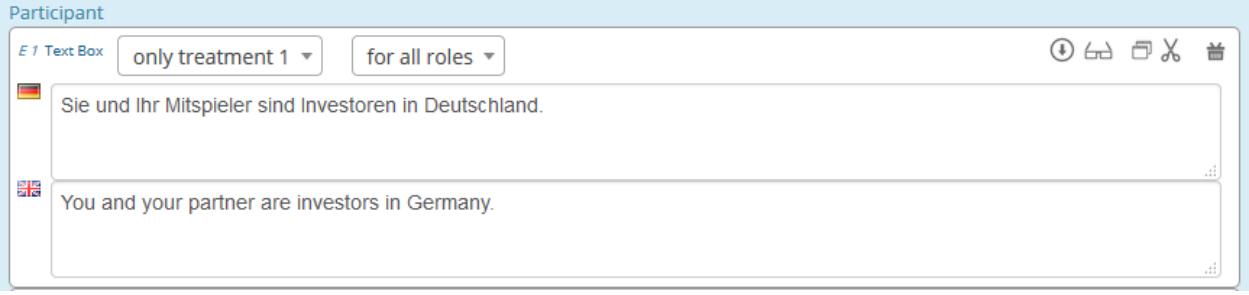
For *select lists*, you have to define a variable name and a default value as well. Also, a name is required. For *options*, you can specify a label and a respective value.



In the lecture mode, the lecturer can then set the endowment and choose from different treatment options as shown in the figure above.

6.8 Languages

If you defined a second language in the game settings, all text fields show up twice so that you can enter the text in the two different languages, here German and English.



To display only one language in the editing mode, you can click on the flag symbols next to the game name. Then all fields with the respective language are hidden. Note that this does not mean that the language is deactivated when the game is played, but only that you can hide some fields while implementing the game.

For other elements, this function has not been implemented yet. In this case, you need to enter both languages in one text box, separated by $$. For more information, see *Text box*.

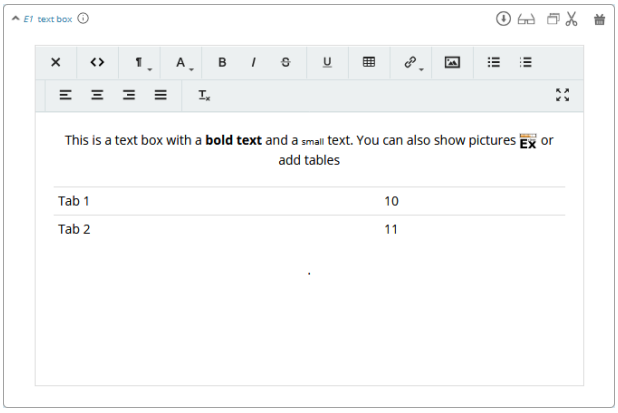
CHAPTER

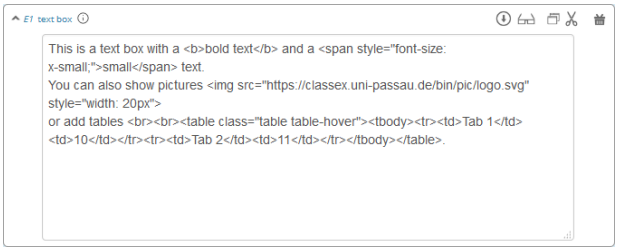
SEVEN

ELEMENTS

7.1 Elements for lecturers and participants

7.1.1 Text box





The text box is the simplest element. The entered text will be displayed to the participants or a the lecturer's screen, depending on where you place the element. The text box is equipped with a WYSIWYG editor, which allows you to insert tables, symbols, etc. If you double-click into the text element, the WYSIWYG editor opens (see left figure). You can switch back to the normal text box by clicking on the **<>** symbol (see right figure).

If you are not in the WYSIWG editor, you can use standard HTML to design your texts. You can do, e.g. the following:

```
This is a text box with a <b>bold text</b> and a <span style="font-size: x-small;">small
↪</span> text.

You can also show pictures <img src="https://classex.uni-passau.de/bin/pic/logo.svg"␣
↪style="width: 20px">
or add tables <br><br>

<table class="table table-hover"><tbody>
```

(continues on next page)

(continued from previous page)

```
        <tr><td>Tab 1</td><td>10</td></tr>
        <tr><td>Tab 2</td><td>11</td></tr>
</tbody></table>.
```

Special Characters

Special	Name	Function Example
role1.png	Symbol Role 1	Red participant symbol _ is displayed. static/pic/role1.PNG
role2.png	Symbol Role 2	Green participant symbol _ is displayed. static/pic/role2.PNG
$variable;	Variables	Besides normal text, you can also insert variables into the text box.

If you have defined variables (see *Programming*), you can have these displayed by inserting the character "$", the variable name followed by ";". Make sure not to forget the ";" at the end! Variables and normal text can be combined.

Make sure that role1.png, role2.png,... are followed by a space. Otherwise, the figure will not be replaced.

7.1.1.1 Conditional text

So far, we have only tackled how to read the PHP variables and display them in the text field (e.g. $variable;), but sometimes we would like to display conditional text. For example, we might have a variable that tells whether a participant is a buyer or seller. We can achieve this task by a program element where you define:

```
if ($isBuyer) $buyerText="You are buyer"; else $buyerText="You are seller";
```

Then you can enter and output $buyerText; in the text box.

7.1.1.2 Display formulas

Although classEx does not directly support LaTeX, it is possible to display formulas to participants in a more elegant way in a text box with MathML. It is best to use the NON WYSIWYG editor in the text box. Switching to the WYSIWYG version already translates your HTML code into the formula.

An example - the following HTML code leads to the following formula:

<math xmlns="http://www.w3.org/1998/Math/MathML">
<mi>a</mi> <msup>
<mi>x</mi> <mn>2</mn>
</msup> <mo>+</mo> <mi>b</mi> <mi>x</mi> <mo>+</mo> <mi>c</mi>
</math>

$$ax^2 + bx + c$$

It is also possible to display symbols and indices like in this example:

<math xmlns="http://www.w3.org/1998/Math/MathML">
<msub><mi>π</mi><mn>i</mn></msub>
</math>

This appears like this in classEx:

Your payoff π_i is determined by

You can find an overview with different tutorials here: https://www.w3.org/Math/#learn

In case your browser does not directly support MathML, here is a tutorial on how to work with it nevertheless: https://developer.mozilla.org/en-US/docs/Web/MathML/Authoring

7.1.2 Element Reference

In order to avoid redundancies, you can reference elements and add them in a different place in the game (instead of copying them directly). This place has to be after or at the same stage as the original element. For this, you can use the reference element. If the original element is altered, the reference is adapted automatically. The reference is created by selecting the stage number and the element number you are referring to.

Note

For example, if you require the same text in two stages, an element reference is a far more elegant version than a simple copy because any changes to the original element are adopted automatically. Also, a reference object in a participant field can just link to another participant object (the same principle holds for the lecturer fields).

Note

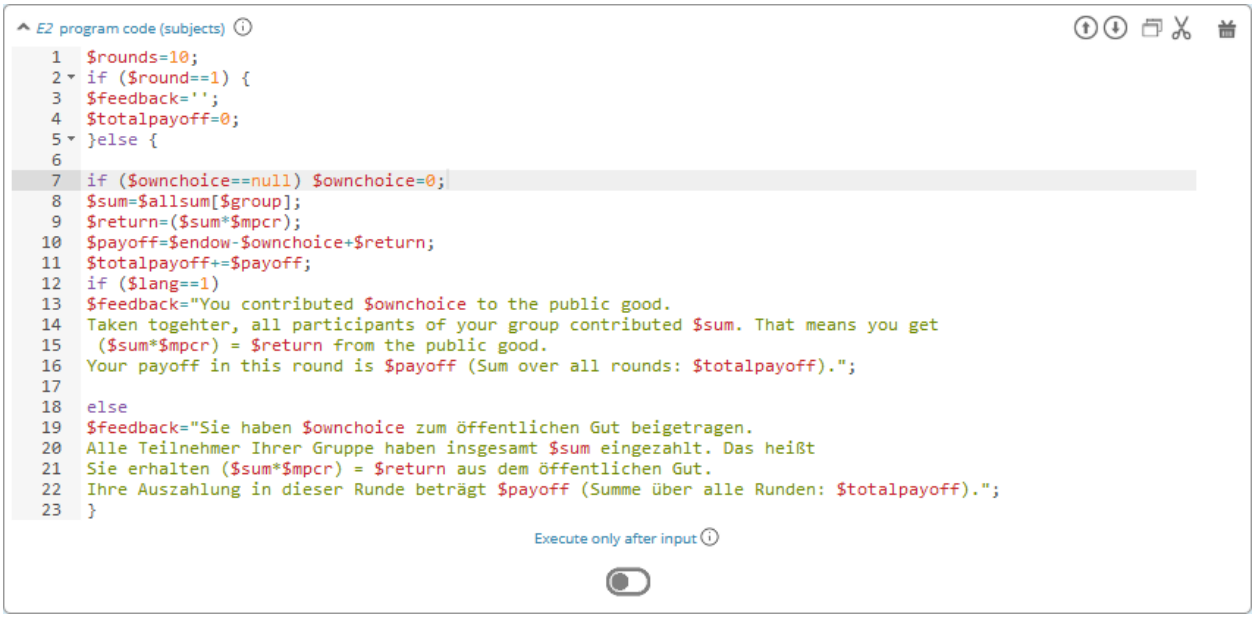
Please notice that the display condition is not taken from the referenced element but from the reference itself.

Warning

If you change the order of referenced elements, the reference does not automatically adapt, but has to be changed manually.

7.1.3 Program code

Program snippets can be implemented to calculate results for each subject. For further information, see *Programming*.



Note

Program elements for the participant are always executed before all other elements.

7.2 Elements for participants

7.2.1 Input element

In this element, you can insert several input fields. These are numbered #1, #2,... You can add input fields by clicking on *add new input field*. The input fields are displayed one after another. Input elements always provide a submit button automatically. In the following, the different types of input fields are described in more detail.

The following settings are available for every input field:

type of input field
: The type of input specifies how the input should be taken by the input field, like numeric input, discrete choice, sliders, ...

variable name
: The variable name is the identifier of the decision input. The variable is automatically stored in the subjects table. The variable name can then be used in programs. For example, if your variable is called $e, you can access it by writing "$e;" in a text box or use `$e` in a program element.

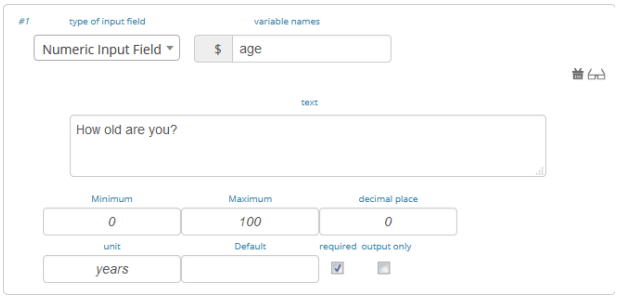
Furthermore, you can delete an input field by clicking on or provide a display condition by clicking on . Display conditions are defined in the same way as for elements (see *Handling elements*).

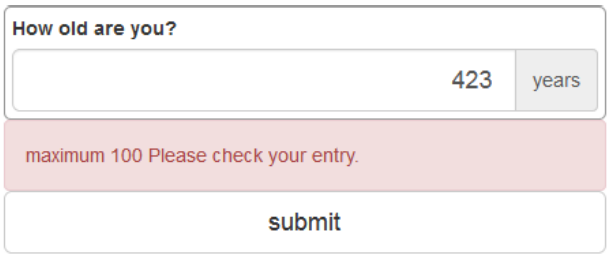
Warning

Please notice that only one input element is allowed per stage. For several inputs, add additional input fields to the first input element.

> **Note**
>
> In all input fields, you can also use variables instead of numbers or text. This can e.g. be useful if you want to set a maximum depending on a variable `$x`. Just enter $x; in the maximum field. This holds true for other fields as well.

7.2.1.1 Numeric input field





The numeric input stores the numeric input by participants and provides a basic check (minimum, maximum). It automatically rounds the input to a given decimal place and allows for input of digits with as `2.34` or `2,34`. classEx also automatically changes the input to numeric on mobile devices and shows the correct keyboard.

text
: The text is displayed on top of the input field.

minimum
: The minimum specifies the minimal value. If the participant enters a value below the minimum, a warning is displayed, and he or she cannot proceed.

maximum
: The maximum specifies the maximum value. If the participant enters a value above the maximum, a warning is displayed, and he or she cannot proceed (see figure).

decimal place
: The number of decimal places. classEx automatically rounds accordingly.

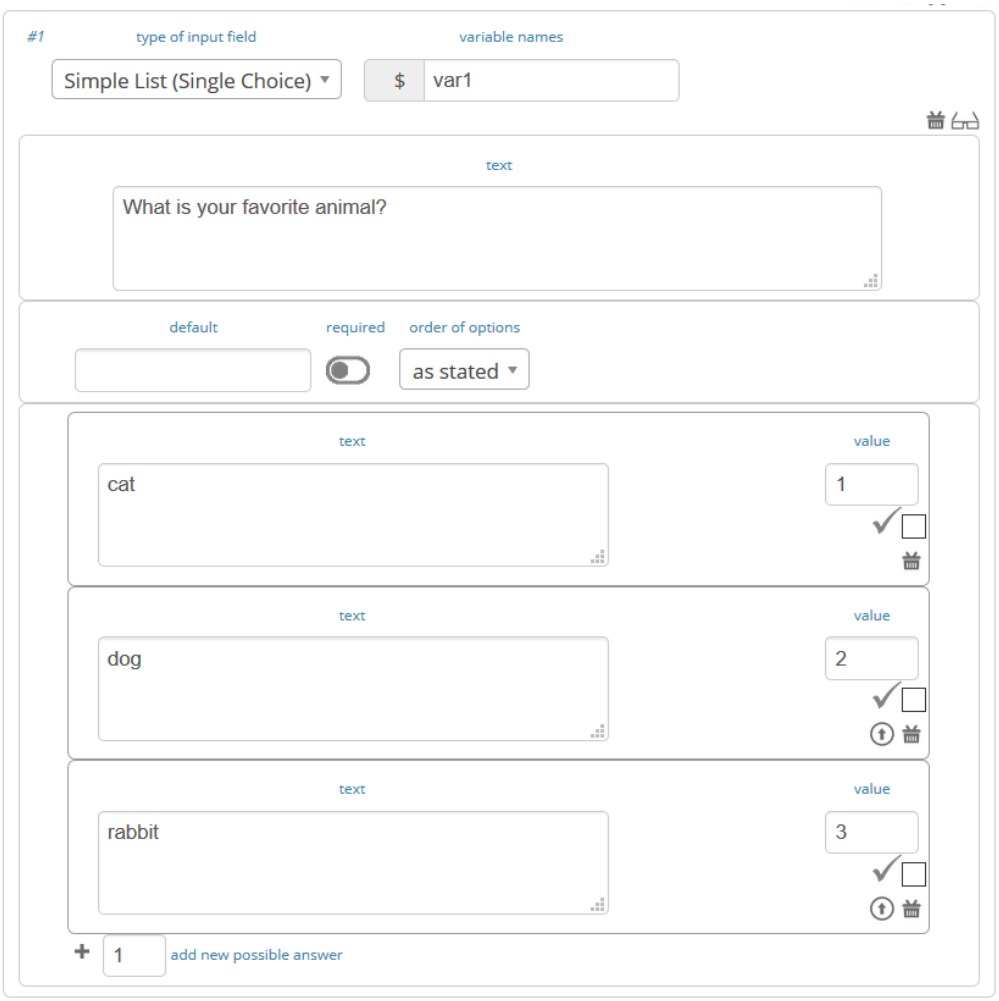
unit
: A unit (e.g. %, €, mm, ...) can be specified that will be displayed on the right of the input field (here "years").

default
: A default value that is displayed to participants at the start.

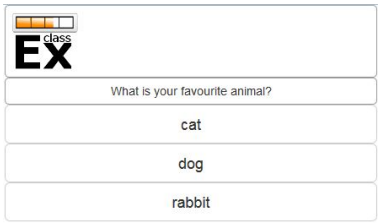
required
: This determines whether input is mandatory. In this case, participants cannot proceed without entering a value.

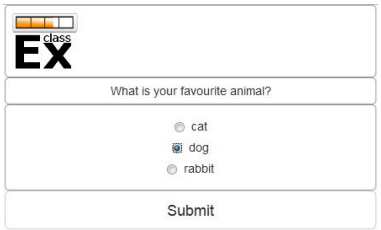
output only
: This means that the field is read-only. Still, the value of the field is stored as a normal variable. It can, e.g. be used to provide *Live feedback on input* with JavaScript.

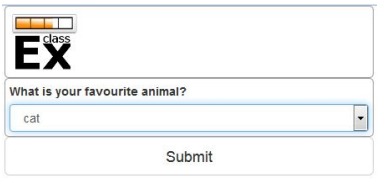
7.2.1.2 Buttons, simple list and drop list (single choice)



Those three types of input fields are used for discrete decisions. You can implement single-choice questions using buttons (first figure below), simple lists (second figure) or drop lists (third figure). This is what they look like in the participants' display. The settings are the same for these input fields.







text
The text is displayed on top of the input field.

default
The default value is pre-marked at the start. For buttons, it is highlighted with a colour. For drop lists, it is preselected.

required
This determines whether input is mandatory. In this case, participants cannot proceed before making a choice. For button input, you can set that only the correct answer allows proceeding.

order of options
The order of options can be set to *as stated* or to *random*. In the latter case, the random order is determined separately for each participant.

options
Options can be defined in the lower area of the element. Participants make a decision by choosing one of the options. The order of options can be altered by clicking on the arrow. You can also delete or add options.

Options come with a displayed text and a value. The value is stored in the data and as a variable.

You can mark the correct answer by clicking on the symbol static/elements/check.png PNG. In this case, if you use the single/multiple choice result element for the lecturer, the correct answer is marked there.

7.2.1.3 Checkboxes (multiple choice)

Choosing multiple options is possible by using checkboxes. Checkboxes work in exactly the same way as single-choice options shown above. Only the form of display is slightly different, as these are displayed as a list from which participants can pick several options. This way, multiple inputs can occur in one stage. In addition to the single choice, you can specify the following settings:

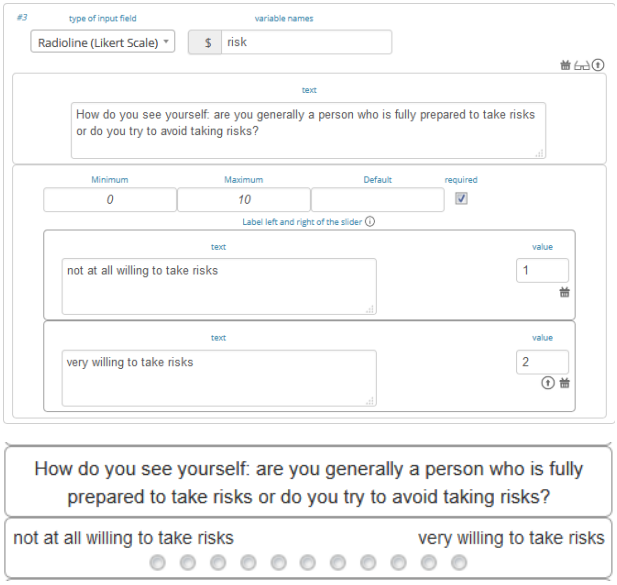
minimum and maximum number of choices and
: This restricts how many options can be picked by the participant.

number of answers per row
: This states how many options are displayed in each row (here, 2).

Important

Multiple choice inputs are stored separated with _, e.g. **2**_3 if options with values 2 and 3 were selected. If you specify a default, please specify it accordingly.

7.2.1.4 Radioline



Radiolines, like Likert scales, offer stepwise input. The Settings are

minimum and maximum
: The first bullet of the slider is the minimum value, and the last is the maximum value. For each bullet, the value increases by one. In the example, a minimum of 0 and a maximum of 10 give 11 input possibilities.

default
: The default value is pre-marked at the start.

required
This determines whether input is mandatory. In this case, participants cannot proceed before making a choice.

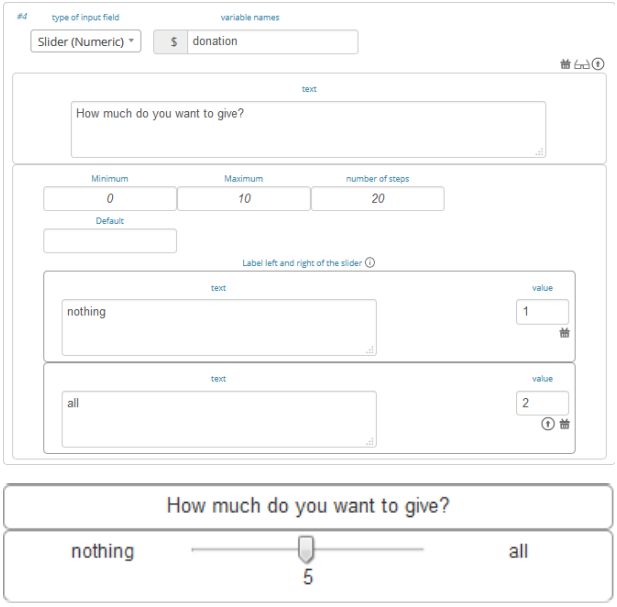
label options
You can show all labels, which are then shown with rotated text below the radioline. Or you can show only the right and left labels. If you choose this option, intermediate labels are ignored.

Label of the radioline values
You can add labels to each value of the radioline. Make sure that you specify the correct value.

Important

The labels are only shown if you specify the correct value. E.g. if the radioline goes from 0 to 10 and you want a label at the very left, it has to have the value 0. If you choose the option to show only min/max labels, intermediate labels are ignored.

7.2.1.5 Slider



Sliders are a similar concept to radiolines. In this form of input, the participant moves a slider along a bar of predetermined positions.

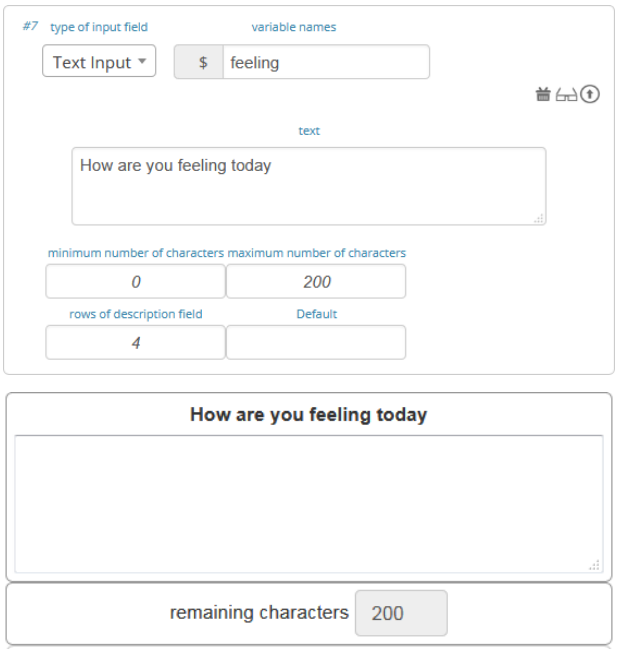
minimum and maximum
The minimum and maximum determine the range of the slider.

number of steps
The number of steps determines how many positions the slider can be moved. In the example, 20 steps between 0 and 10 mean that each step increases by 0.5.

default
The default value is pre-marked at the start. If no default is set, the slider is positioned in the middle of the range.

label left and right of the slider
You can add a label on the left and the right of the slider.

7.2.1.6 Text input



Text input fields allow participants to enter text. If a maximum is specified, it shows the number of remaining characters.

minimum and maximum number of characters
: This limits the number of characters which can be written by participants.

default
: The default value is written in the text input field.

number of rows
: This determines the height of the text input field.

7.2.1.7 Hidden field

The hidden fields allow for saving values. The only setting is to provide a default.

7.2.1.8 Next button

The next button allows you to add an additional button. Note that classEx normally provides buttons automatically. This may be useful if you have no input but only a button. The only setting is the label of the button.

7.2.1.9 Other input fields

There are other input fields available (urn, infotext, mean of all input fields). Those are outdated and should only be used with care.

The contract input field is also outdated. Please use the *Contract* element instead.

7.2.1.10 Additional settings

At the bottom of each input element, you can find additional settings which open by clicking on *additional settings*.

button label
Here you can specify the label of the button. The default is "Submit". This button label is only used if there is an own button for submitting your decision. E.g. in *Buttons (single choice)*, buttons are labelled with the possible answers the editor inserted.

confirmation message
If you enter a text here, participants are asked for confirmation before submitting. E.g. you can enter "Are you sure?" which is then displayed together with an OK button to participants.

hide decision after submitting
After submitting, participants see a confirmation message and their input. If you switch this setting on, the input is not displayed anymore after submitting.

directly to next stage
Normally, participants are moved to the next stage by the experimenter (see *Start button and automatic start*). If participants play individually and this setting is switched on, they move autonomously to the next stage. In this case, you have to select *no forwarding* if you use an automatic start.

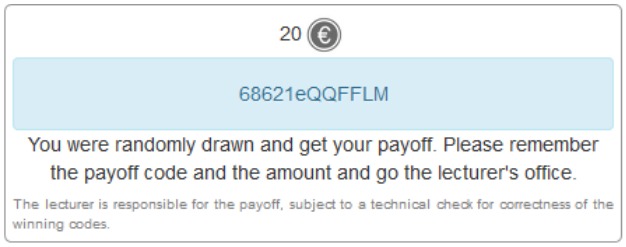
show button later
The appearance of the buttons can be delayed, e.g. if you want participants not to click too fast.

allow resending the input
The input is not blocked after sending, but inputs can be resubmitted as long as the stage is active.

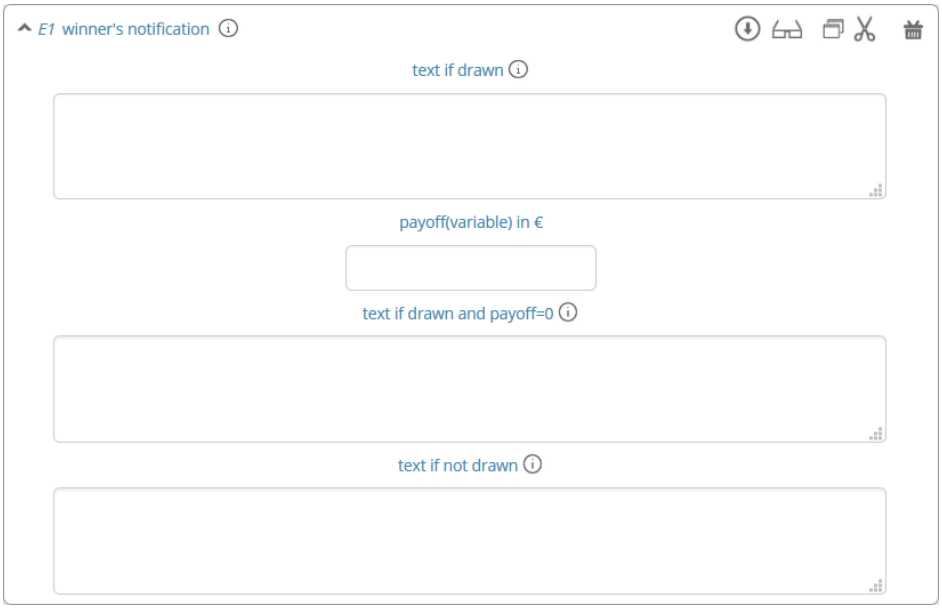
space between input fields (px)
This allows you to add a certain space in pixels between input fields.

highlight button
The send button can be highlighted in blue.

7.2.2 Winner's Notification



If a game is played with real payoffs, this element displays the payoff code to participants and the respective payoff. The payoff can be a number or also text (e.g. two tickets for the theater). The element automatically provides a legal disclaimer. The payoff notice can be changed. Also, the currency of the payoff can be changed under **`course data`_**.



A winning notification is necessary for games with monetary payoff. The participants who are randomly drawn receive a winning notification as well as a code to cash in their earnings.

payoff(variable) in €
: The amount of earnings can be determined by this setting. You can enter a fixed amount or a variable that is calculated beforehand. If, for example, the variable `$payoff` is calculated in a program during the game, you can enter `$payoff;` in the earnings field.

text if drawn
: The default text is: You were randomly drawn and get your payoff. Please remember the payoff code and the amount, and go to the lecturer's office.

text if not drawn
: The default text is: You were not randomly drawn to be paid out.

text if drawn and payoff = 0
: The default text is: You were randomly drawn, but your payoff is X. It is therefore not paid out.

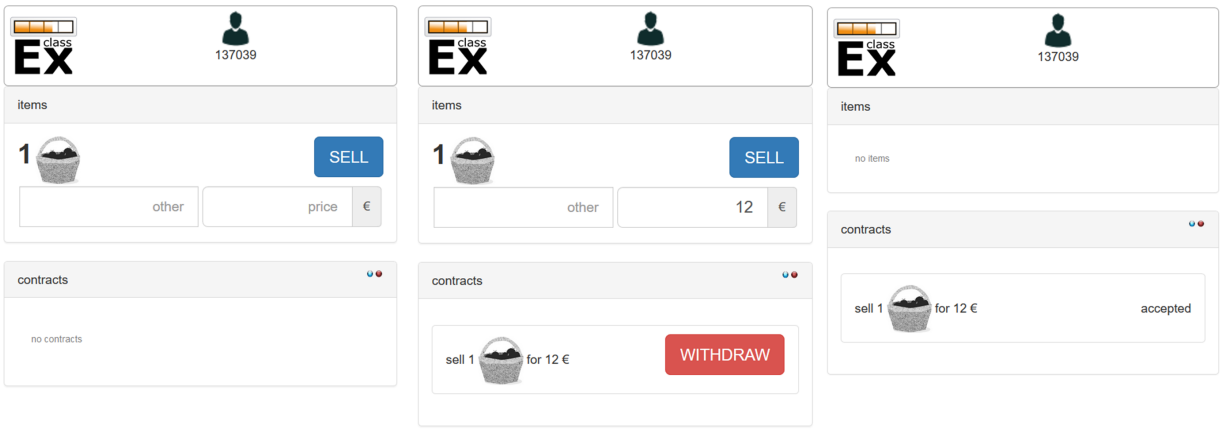
Important

The winning notification can only be displayed if you also define a *Winner's draw* on the lecturer side. Otherwise, no winner can be determined. Winners are always drawn with a lecturer element.

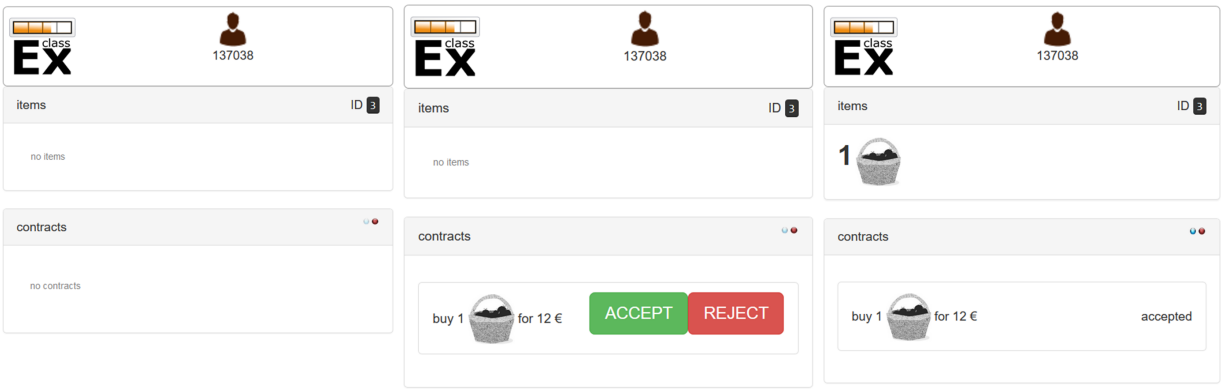
7.2.3 Contract

With this element, you can enable participants to conclude contracts. By adjusting the settings, you can customize the contract to your needs. Contracts can be used to trade a commodity between subjects in real time. Subjects move around in the classroom and talk to each other. When they agreed on a price, they enter it into the input mask together with the signature of the counterpart (see seller screen). The counterpart has to accept the trade (or reject it, see buyer screen).

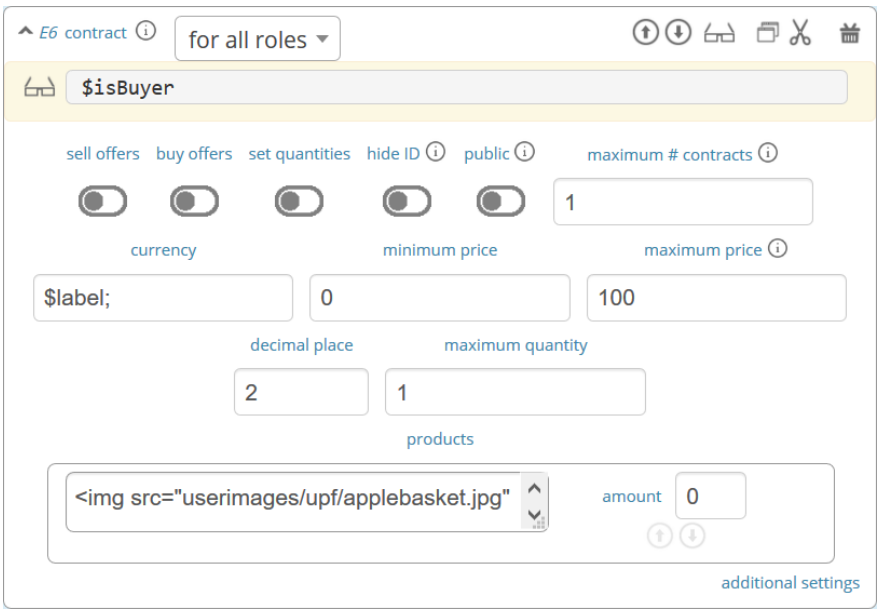
The screen for the seller



The screen for the buyer



The contracts made can be shown at the lecturer's screen with the *Contract table*. In addition, there are special functions in globals and subjects programs to retrieve contracts (see *Programming*). All contracts are also stored in the standard Excel file, which can be retrieved in the data menu (see *Data*).



Contracts can be set up by adding contract elements **both for buyers and sellers**.

sell offers/buy offers
If you turn this on, you allow for sell or buy offers made by the respective subject.

set quantities
Allows setting quantities (otherwise quantity is always 1). With quantities, prices are set as price/unit.

hide ID
allows to disable the ID. The ID is needed for sell and buy offers to be sent to a specific person. E.g. if the buyer can make buy offers, she needs the ID of the seller to send the offer to.

> **Note**
>
> Contract elements use the subject ID as ID for trading (because it is very short). But contracts are stored with the internal participant ID.

public
If you activate this, offers are made to all other subjects who can accept them. This feature is still in a beta version. So test it with care.

max # contracts
The maximum number of (accepted) contracts limits how many contracts can be made by a subject.

currency/min price/max price/decimal place
Currency of the prices and minimum, maximum and decimal places can be specified here.

maximum quantity
The maximum quantity a subject is allowed to possess. Limits how many items can be bought.

products
You can specify a name (or a small image) and the initial amount of the good (e.g. the seller has 1 unit, the buyer 0 units).

By clicking on additional settings, further advanced settings can be adjusted.

retractable
This determines if an offer can be withdrawn or not. The default is that offers can be withdrawn.

concurrent offers
In case of public offers, this determines if more offers can be made at the same time. Note that this is only checked for the respective contract element. If you provide more contract elements to the same persons, the check is only made for each contract element.

message
You can provide a static message (e.g. about the buyer type) with the contract.

proprietary trading
This allows you to sell to or buy from yourself.

label price
The label price can be changed, e.g. to wage. The label is shown in the price field if the field is empty.

label quantity
The label quantity can be changed, e.g. to hours. The label is shown in the quantity field if the field is empty.

label /unit
The label /unit can be changed, e.g. to /hour. The label is shown at the price input field.

permit
If you enter a text or a number here, contracts can only be concluded if you enter a number.

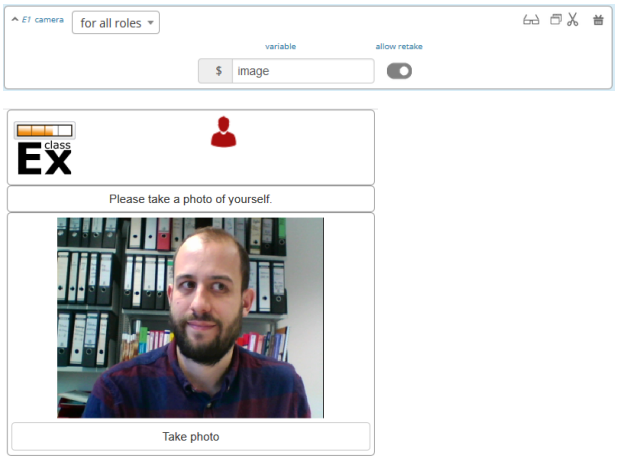
7.2.4 Payoff matrix game

This element helps display the payoff from a two-role game. This is a simplification compared to the use of subject programmes to determine the value of the other participant and calculate the payouts. The same can be achieved with text boxes and subjects programs.

decision role 1 and role 2
You need to specify which input field contains the decision of the respective participant for the row participant and for the column participant. The labels of the payoff matrix are determined by the specified input fields.

results matrix
In the table, you enter the payoff for the row participant first, followed by the payoff for the column participant. The payoff is stored as variable $payoff; which can then be used for the winning notification or further calculations.

7.2.5 Camera



With this element, you can enable participants to take a picture of themselves.

variable
The variable name, under which the picture is stored, has to be defined.

allow retake
: You can also specify whether the participants are allowed to take a picture again. Only the last picture taken will then be saved.

> **Note**
>
> Informed Consent: Participants are asked by the browser if the browser can access the webcam or not. Please make participants aware that they do not have to take a picture and ask them for their consent.

7.2.5.1 Retrieving Pictures

Pictures can be retrieved in the following ways:

At the participant's screen
: You can use the normal variable notation ($image;) to display pictures in text boxes.

At the lecturer's screen
: You can use `$getValues(...)` to retrieve the pictures of all participants and display them (see *Functions*).

From the stored data
: In the downloaded data, you find stored images in the subjects table. They are base64 decoded and can be encoded with free online tools. Just take away "" from the string, so that it starts e.g. with "/9j/....".

7.2.6 Javascript program

You can also add small JavaScript programs to the participant screen. More information can be found at *Javascript*.

7.2.7 Filled in form

This element allows you to display the filled-in input element of the previous stage. This element is outdated and should only be used with care.

7.2.8 Similarity check(beta)

In this feature, the main part consists of an input field and a text field to define a variable and get the student's answer. In the next phase, you can use the predefined variable name and enter the correct solution to compare the correct solution with the student's answer. You can enter a prompt, questions and a sample solution in the textbox at this stage. Then use the variable name (which you defined in the previous phase) to send the student's answer and your prompt to the OpenAI server. In response, the AI will send you feedback on the student's response depending on the prompt you provided. At the end, the feedback is displayed to the participants.

In the Similarity check, the main part consists of one input field and a textbox to define a variable and get the student's answer. In the next stage, you can use the predefined variable name and enter the correct solution to compare the correct solution with the student's answer. .. image:: _static/elements/firststep.png :width: 15px .. image:: _static/elements/variablesimilarity.png :width: 15px There are more options to set by clicking on the additional settings: .. image:: _static/elements/additionamset.png :width: 15px

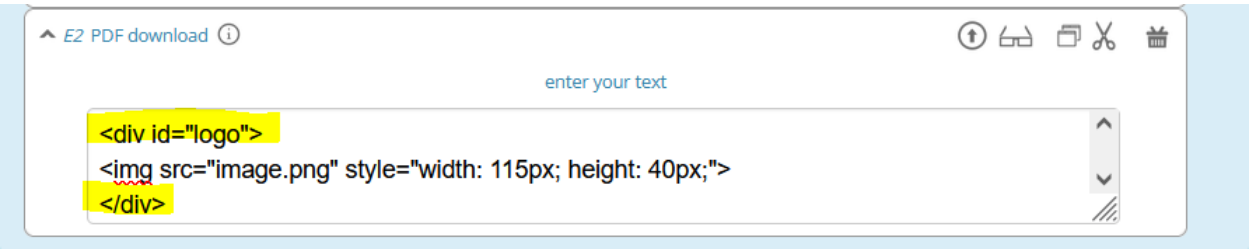
It is possible to define different messages that are displayed on the participant's screen after the star rating. There are also 4 boxes that can be used to define a separate star range for each game. Depending on the number of sentences in the correct solution, the range of numbers should go higher.

> **Warning**

Please note that the score ranges have a default value each; if you don't change it depending on the game, it will use the defined value, and there might be a chance of having a wrong result.

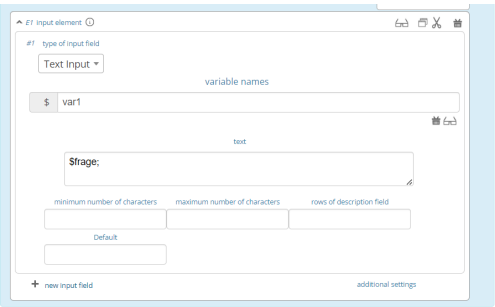
In the end, the similarity check calculation method is: Some of the points obtained from the comparison of each pair of sentences (one sentence from the answer with one sentence from the correct solution)

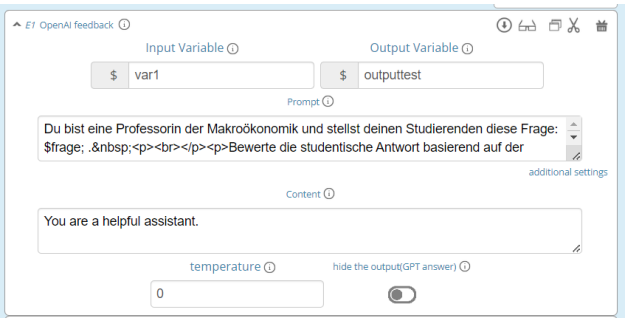
7.2.9 PDF download



The PDF Download element integrates a user-friendly button within the game interface, allowing students to download specific parts of the game material in PDF format. This feature includes a text field where you can add custom text, such as questions and answers, and other relevant content. Additionally, you can add image tags, which will be rendered as images in the final PDF document. Once the content is added, students can easily download it as a PDF document, facilitating access to the material for offline review and study.

7.2.10 OpenAI feedback





The OpenAI feedback element allows you to send a request that includes a student's answer and a specific prompt to the OpenAI server to receive tailored AI-generated feedback. This feedback is crafted based on the provided prompt and is intended to be displayed to students.

To use the feature, you need to configure the input element, variable name and a textbox for capturing the student's answer in the first stage. In the next stage of the game, you need to use the same input variable that you set previously and define the output variable name(to store the feedback), and the prompt. For the prompt, you can define a scenario, questions, a sample solution, etc, in the textbox.

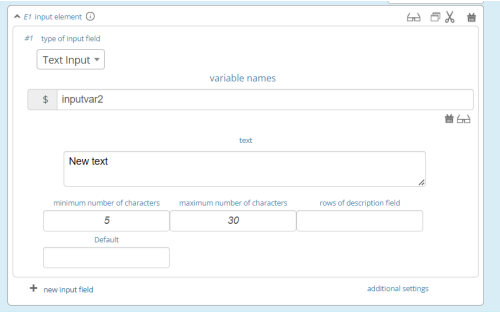
variable
Ensure all variable names are clearly defined and used consistently.

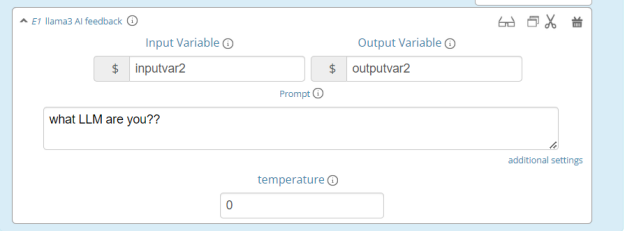
The content field is associated with the system-level message and provides context or high-level instructions for the conversation. It helps set the tone or guide the behavior of the AI model during the conversation. Please only change this field if you want to have another role!

Temperature is a parameter that influences the randomness of the model's responses. A higher temperature value leads to more varied and creative answers, while a lower value makes the answers more concentrated and deterministic. It is a measure of how much randomness is introduced into the generated text.

Output Storage: store the response of AI(output), without showing it to the participants.

7.2.11 llama3 AI feedback





The llama3 AI feedback element allows you to send a request that includes a student's answer and a specific prompt to receive tailored AI-generated feedback. This feedback is crafted based on the provided prompt and is intended to be displayed to students.

LLaMA 3 is an open-source AI model developed by Meta AI, designed to significantly improve natural language understanding and generation, showcasing advancements in performance, efficiency, and scalability.

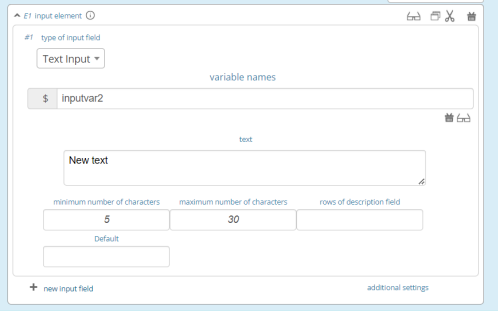
To use the feature, you need to configure the input element, variable name and a textbox for capturing the student's answer in the first stage. In the next stage of the game, you need to use the same input variable that you set previously and define the output variable name(to store the feedback), and the prompt. For the prompt, you can define a scenario, questions, a sample solution, etc, in the textbox.

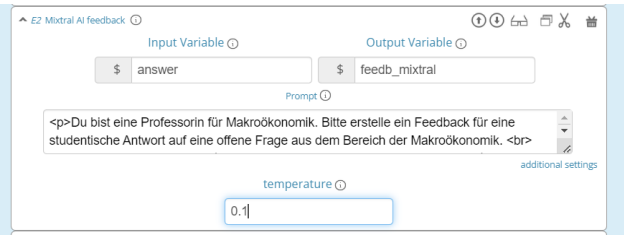
variable
Ensure all variable names are clearly defined and used consistently.

The temperature is a parameter that influences the randomness of the model responses. A higher temperature value results in more varied and creative responses, while a lower value makes the responses more focused and deterministic. It is a measure of how much randomness is introduced into the generated text.

Output Storage: store the response of AI(output), without showing it to the participants.

7.2.12 Mixtral AI feedback





The Mixtral AI feedback element allows you to send a request that includes a student's answer and a specific prompt to receive tailored AI-generated feedback. This feedback is crafted based on the provided prompt and is intended for display to students.

Mixtral AI, developed by Mistral AI, harnesses a sparse mixture of experts (SMoE) to deliver unmatched efficiency and superior performance. Licensed under Apache 2.0, this open-source model excels in cost-effective computation, outperforming standard benchmarks and offering a robust solution for advanced AI applications.

To use the feature, you need to configure the input element, variable name and a textbox for capturing the student's answer in the first stage. In the next stage of the game, you need to use the same input variable that you set previously and define the output variable name (to store the feedback) and the prompt. For the prompt, you can define a scenario, questions, a sample solution, etc, in the textbox.

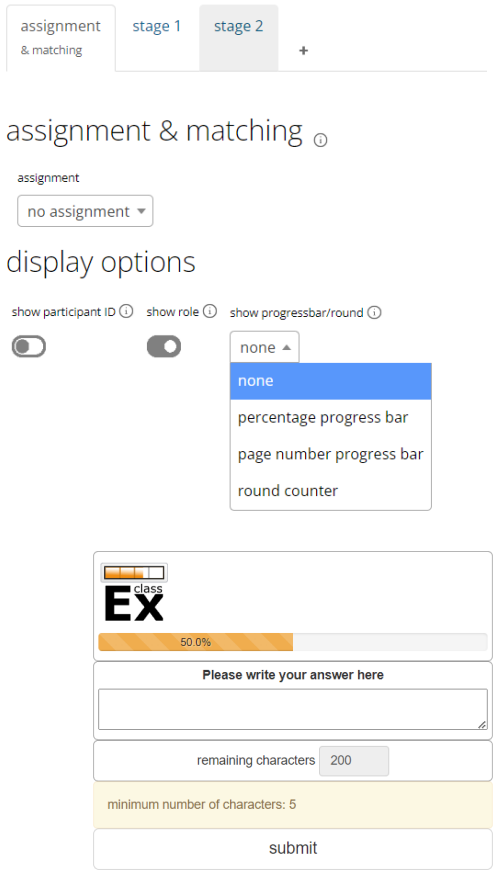
variable

Ensure all variable names are clearly defined and used consistently.

The temperature is a parameter that influences the randomness of the model responses. A higher temperature value results in more varied and creative responses, while a lower value makes the responses more focused and deterministic.

Output Storage: store the response of AI(output), without showing it to the participants.

7.2.13 Progressbar/round



It is possible to set the Progressbar/Round in the "assignment & matching" tab of each game. This feature allows you to add either a progress bar or a round counter to each game. By default, this feature is disabled (set to 'none'). When enabled, the following will be shown to the participant:

Percentage Progress Bar: Displays how much of the game has been completed.

Page Number Progress Bar: Shows the number of completed stages.

Round Counter: Shows the number of completed rounds.

Only one option can be active at one time to ensure clarity for participants.

7.3 Elements for lecturers

7.3.1 Start button and automatic start

The start button is used to initiate a stage.

Important

Each stage requires a start button or an automatic start. If stages have result elements, this is not true, as result elements automatically provide a button to start the stage (if no other button is defined).

There are two alternatives. A start button which has to be clicked by the experimenter or an automatic start.

7.3.1.1 Start button

The start buttons allow you to start stages of a game. It is marked in blue and clearly visible to ease guidance for the lecturer. Before the start of the first stage, it additionally states the number of logged-in participants (here: 1).

After pressing the start button, the stage is started, the button disappears, and a counter is shown, which counts the number of participants in that stage. In the example, it means that there are no red participants (0/0) and one green participant who finished the stage (1/1). Counting can be done over all, by role, group or treatment.

The start button can be configured according to the own needs.

button label
: You can name the button (e.g. Start Trade).

feature
: Instead of starting the current stage, you can also use the start button to jump to different stages. In case you jump to the stage where payoffs are distributed, this will suppress the distribution. This can be prevented by following the explanation for the setWinner-function in `Functions`_.

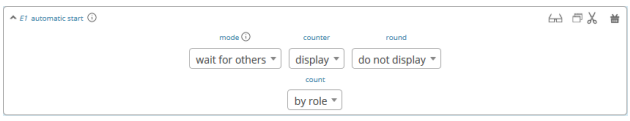
confirmation message
: You can set whether a pop-up should appear after clicking to confirm the action.

round
: You can set whether the current round number should be displayed or not.

stage name
: You can set whether the stage name should be displayed or not.

count
: You can set the counter which appears after the start button is clicked. It can count decisions (also by role, treatment or group if set).

7.3.1.2 Automatic start



The automatic start button allows you to start stages when subjects have finished the previous stage. With this feature, they can move through the game autonomously.

mode
The mode can be set to:

- start if possible: If a subject finishes the previous stage, it is forwarded to the next stage.
- wait for all: Subjects are only forwarded if everyone in the group is done with the previous stage.
- no forwarding: Subjects are not forwarded (This feature is only used if subjects forward themselves by clicking on a button. This can be set in additional settings of the input element).

counter
Setting this additionally allows you to deactivate the counter completely.

round
You can set whether the current round number should be displayed or not.

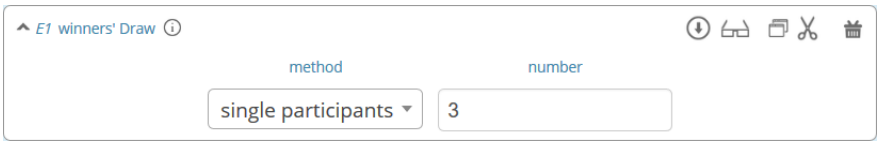
count
You can set the counter which appears after the start button of the previous stage is clicked. It can count decisions (also by role, treatment or group if set).

stage name
You can set whether the stage name should be displayed or not.

..note:: classEx executes the start element always at the beginning of a new stage. If you, e.g. want to forward all members of a group from stage 2 to stage 3 as soon as all of them have made their decision, you have to implement an automatic start button set to "wait for all" at the beginning of stage 3.

..note:: If you use an automatic start in a stage, you can not use globals programs in this stage, as participants move autonomously through the game and the globals program is only called once.

7.3.2 Winner's draw



This element should be implemented in the last stage and draw a winner among all participants. The earnings have to be calculated individually on the participant side (see *Winner's notification* for participants). You should draw winners only once in a game, as the payoff codes do not distinguish between rounds.

method
You can determine whether single participants or coupled participants shall be drawn. Drawing coupled participants only makes sense if you have defined roles. For games with two roles, it is advisable to draw coupled participants as winners because the possibility that only one of the two participants could be drawn might overshadow considerations of fairness or reciprocity.

number
You can also decide how many participants or groups you want to draw.

Important

Payoffs per game are restricted to 100€ per default. If you need higher payoffs, you have to overwrite the variable $maxWin in a global program (e.g. `$maxWin=1000;`).

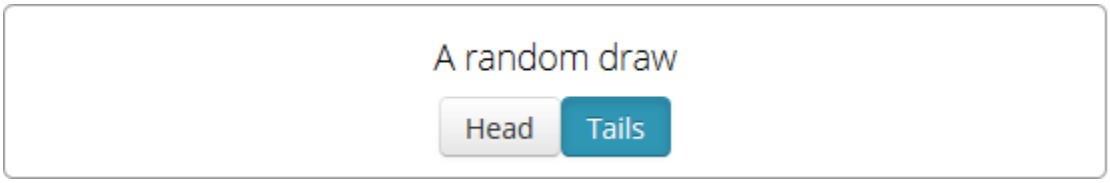
Note

Experience has shown that earnings of less than 5€ are usually not cashed in. Therefore, games should be calibrated in a way that ensures that earnings are at least 10€.

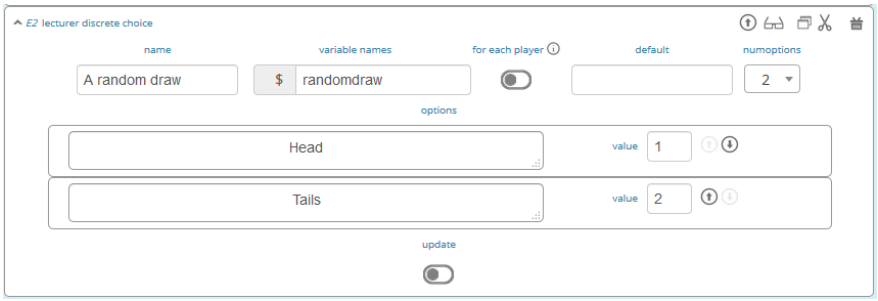
Important

Winners are only drawn from participants who made a decision to avoid inactive participants from being drawn. Therefore, it does not make any sense to put the winners' draw in the first stage.

7.3.3 Lecturer discrete choice



With this element, the lecturer/experimenter can make decisions for all participants during the game, e.g. tossing a coin in front of the class and entering the value in classEx so that payoffs can be calculated based on the coin toss.



name
: This name will be displayed on the screen to identify the input button.

variable name
: The value will be saved under this name as a global variable and can be retrieved by that name.

for each participant
: If you switch this on, you can set the value for each participant separately. The value will be stored as a global variable in an array with the participant ID as the index.

default
: You can set a default.

options
: You can specify options with different values.

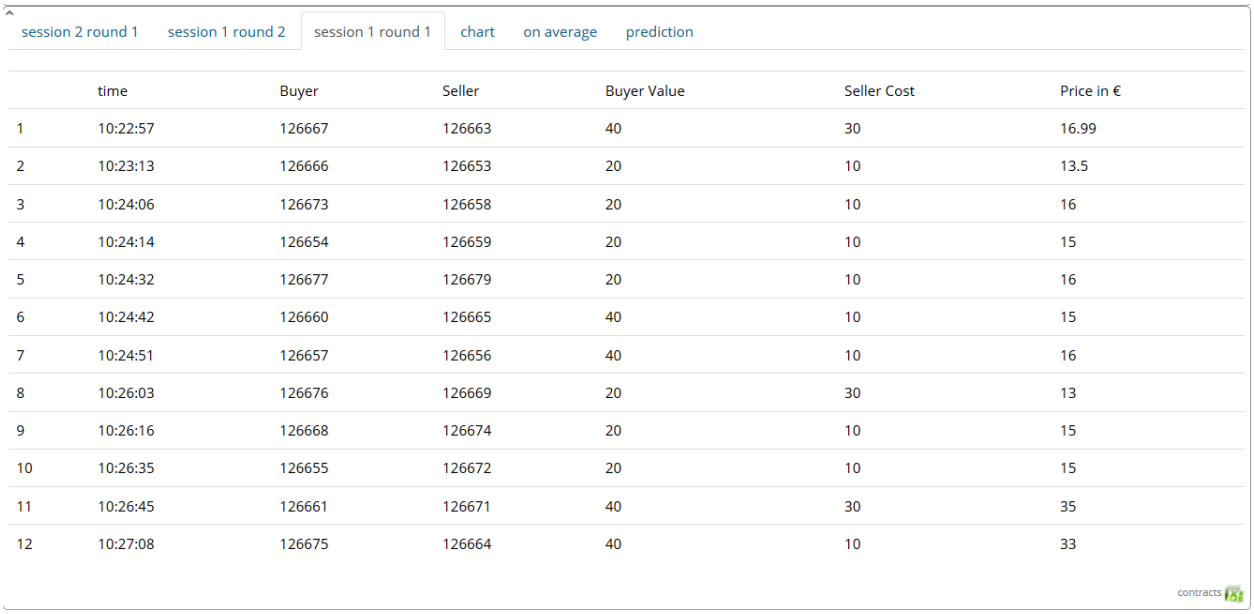
update
: If you switch on the update, the element will check every two seconds if new participants have arrived (only necessary when you switched on "for each participant").

7.3.4 Contract table

With this element, all contracts that were concluded by participants, as well as a chart and the average, are displayed on the lecturer's screen. In the contract table, you have several tabs that you can switch between. You can see them in the figures below.

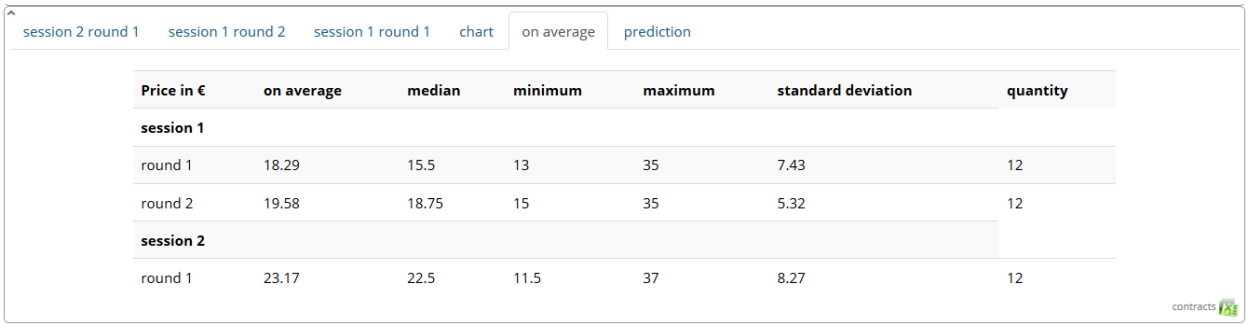
The first tab lists the contracts for each round separately. If set, you can also display seller costs and buyer values along with the contract.

> **Note**
>
> The combination of session and round is a special feature in some games (apple market, fish market, ...). Normally, classEx only knows rounds.



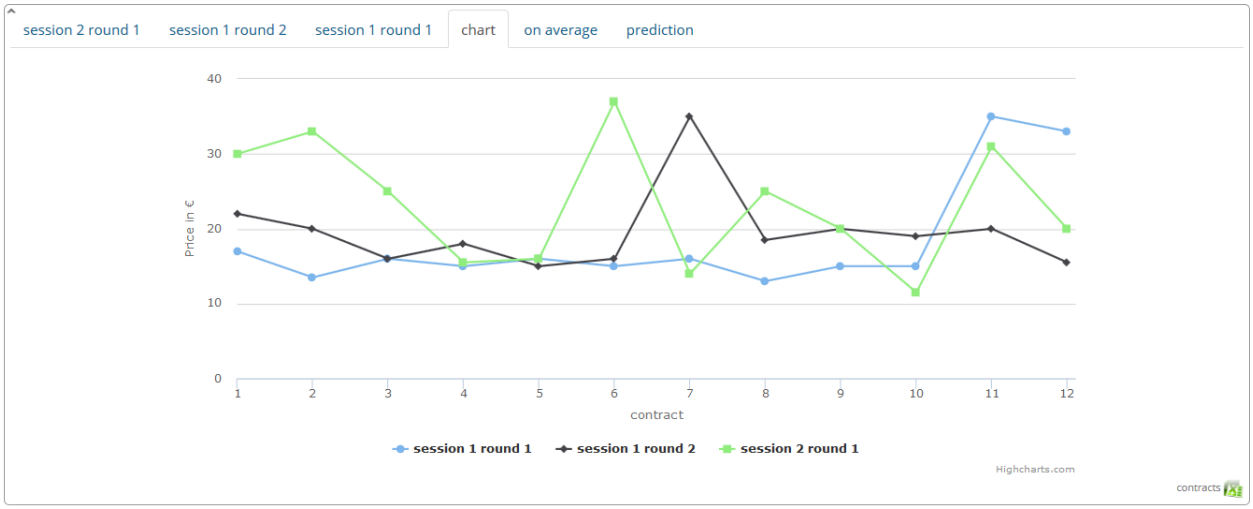
	time	Buyer	Seller	Buyer Value	Seller Cost	Price in €
1	10:22:57	126667	126663	40	30	16.99
2	10:23:13	126666	126653	20	10	13.5
3	10:24:06	126673	126658	20	10	16
4	10:24:14	126654	126659	20	10	15
5	10:24:32	126677	126679	20	10	16
6	10:24:42	126660	126665	40	10	15
7	10:24:51	126657	126656	40	10	16
8	10:26:03	126676	126669	20	30	13
9	10:26:16	126668	126674	20	10	15
10	10:26:35	126655	126672	20	10	15
11	10:26:45	126661	126671	40	30	35
12	10:27:08	126675	126664	40	10	33

The tab *on average* provides summary statistics for each round (mean, median, min, max, std dev).

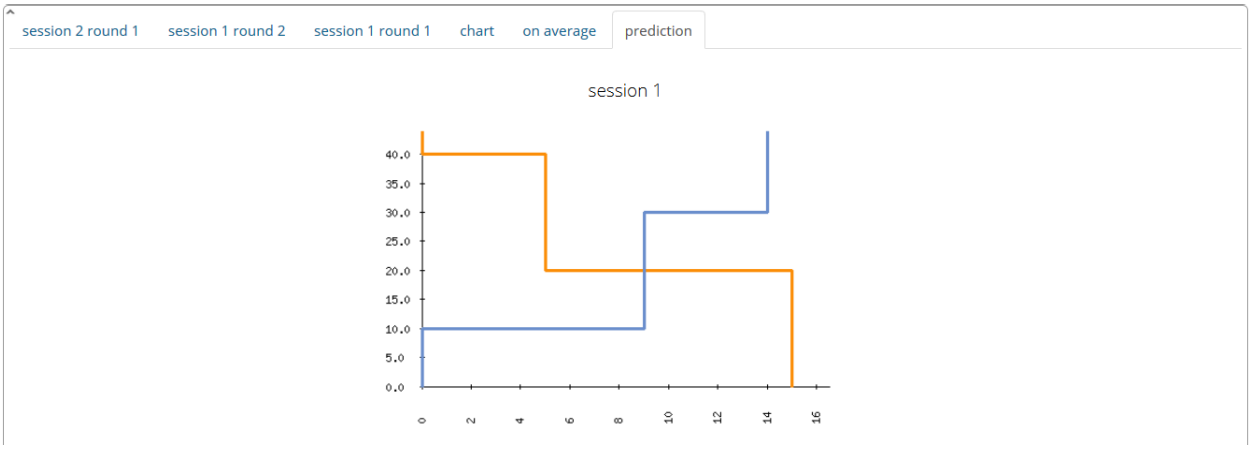


Price in €	on average	median	minimum	maximum	standard deviation	quantity
session 1						
round 1	18.29	15.5	13	35	7.43	12
round 2	19.58	18.75	15	35	5.32	12
session 2						
round 1	23.17	22.5	11.5	37	8.27	12

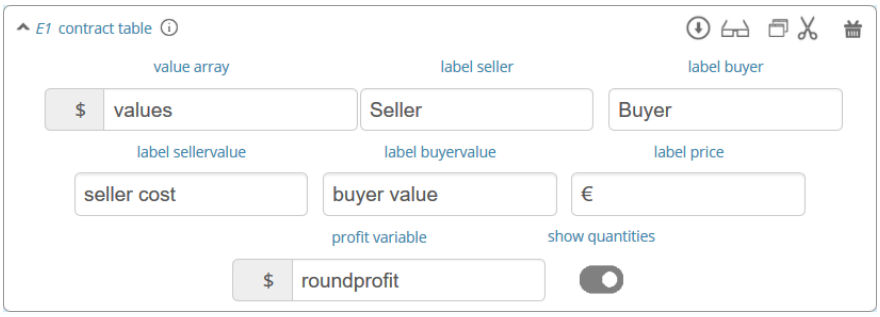
The tab *chart* shows contracts made over time. In case of different quantities, it also shows a bubble chart for the combination of quantities and prices.



The tab *prediction* shows a prediction (if set). To create a prediction, the variables `$demand` and `$supply` have to be filled in a global program. `$supply` and `$demand` should be arrays which contain prices as indices and the resulting quantity as a value.



The following settings can be adapted:



value array
: Gives the name of a (pre-filled) array which contains the role of the participant as index and the respective buyer or seller value as value. This is shown in the table as buyer/seller value.

label seller/buyer/price
: All labels in the table can be changed according to needs.

label seller value/buyer value

If these are left empty, the columns seller value or buyer value are not displayed. If a text is provided, columns are labelled with the text, and classEx reads the value array to provide seller/buyer values.

profit variables
Can be left empty.

show quantities
Additionally shows quantities in the contract table and a bubble chart with quantities and prices.

7.3.5 Result element

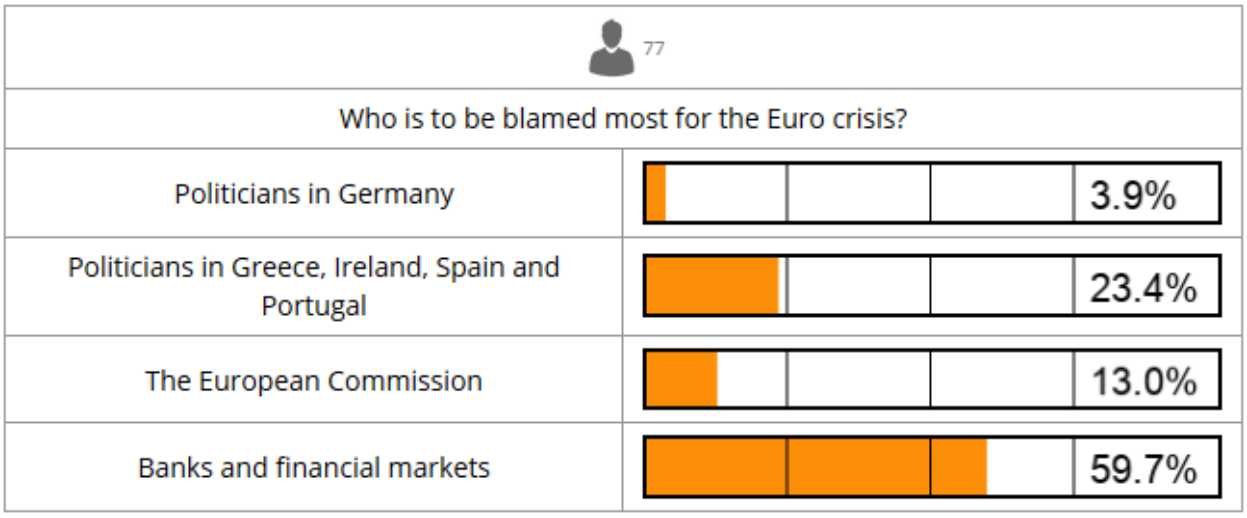
For displaying the results of a game, various types of charts are available.

> **Important**
>
> Note that you can only display saved subject variables. Decisions for input fields are saved automatically so that they can be displayed as well.

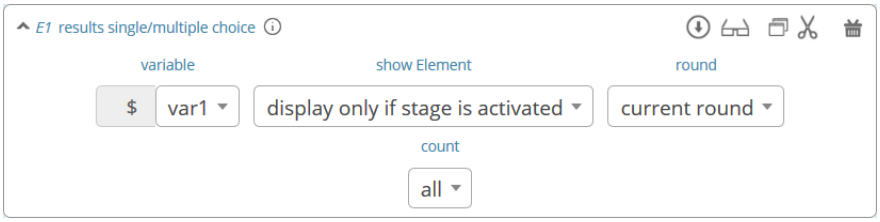
Whenever you can select variables in a field, you only need to insert the variable name (e.g. "payoff"). If you want to use variables in other settings (e.g. the maximum), you have to use the standard notation (e.g. "$payoff;"), which gives the value of the variable.

For most result elements, you can change the setting for *count*. By this, you can determine whether results shall be displayed separately for groups, treatments or roles (if defined).

7.3.5.1 Results single/multiple choice questions



The results are displayed with percentage bars. The element automatically detects if the input is multiple choice or single choice. Hovering over the bars gives the absolute frequency of participants who opted for that option. The element should only be used with input fields with predefined options (otherwise you should use the *Results counter*).



The following settings can be changed:

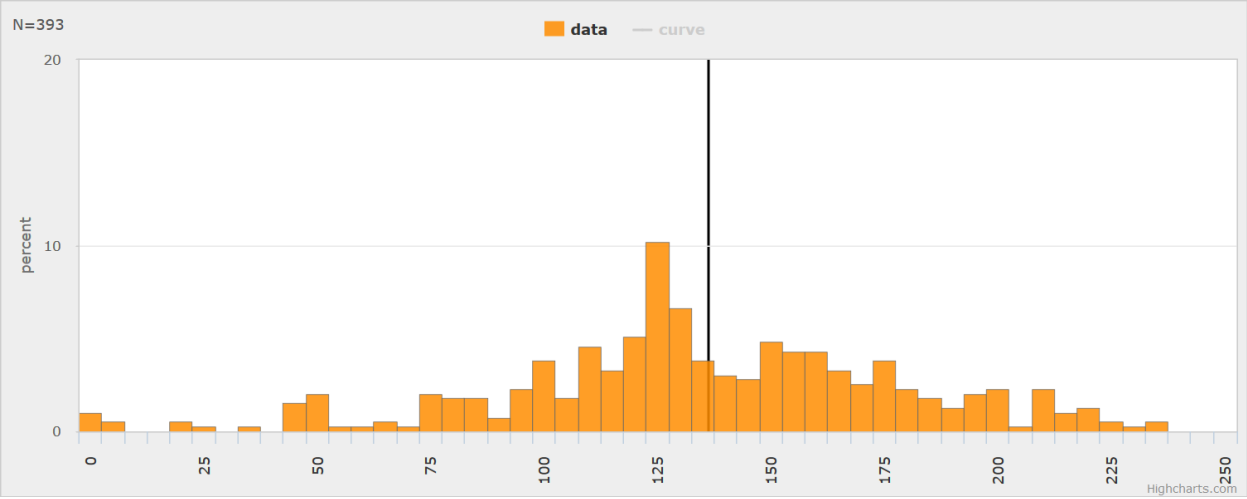
variable
Provide the name of the subjects variable.

show element
Always display the element, or only if the stage is activated.

round
Select if only the current round or all rounds should be displayed.

count
Participants are counted altogether (or per treatment/role/group).

7.3.5.2 Results histogram



The histogram draws the distribution of a variable. The black line marks the mean. Decisions are clustered into so-called bins (here bins of 5). The graph allows for zooming and for adjusting the bin size and the maximum.

_static/elements/histogram_screen_new.PNG

The following settings can be changed:

variable
Choose which variable you want to display.

show element
Element is always displayed, only if the stage is activated or if it is activated and after.

min
The minimum of the histogram (Default 0).

max
The maximum of the histogram.

size of bins
How the values shall be pooled into *bins*. For example, if you define the bin 10, the data will be pooled in groups of ten.

label x-axis
The label you choose for the x-axis is displayed below the x-axis.

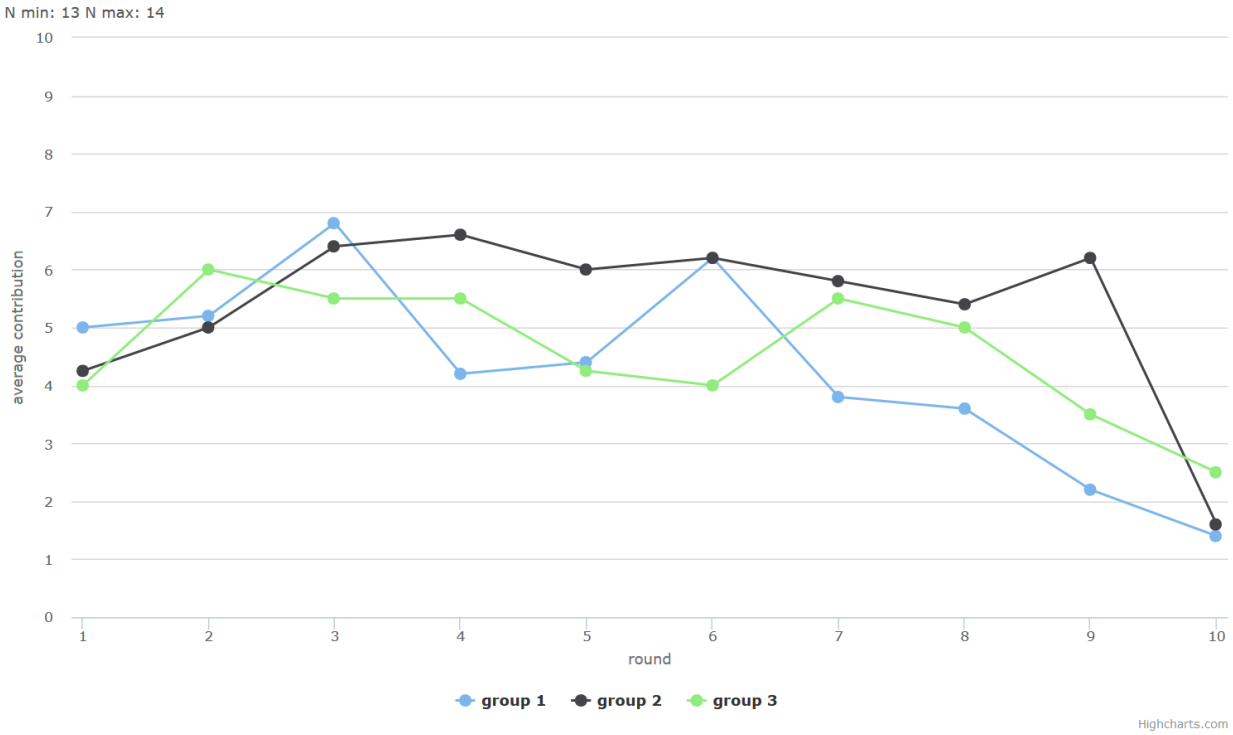
vertical line
An additional red vertical line is drawn at the selected x-value (e.g. to specify a correct or true value).

round
Select if only the current round or all rounds should be displayed.

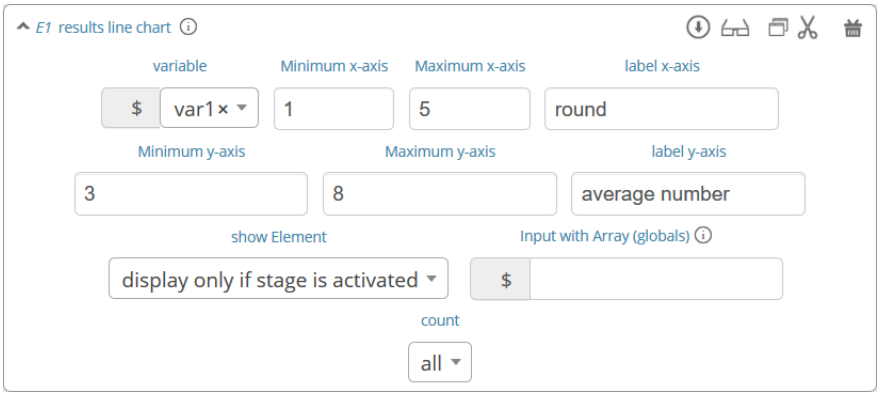
count
Participants are counted altogether or per treatment/role / group.

> **Note**
>
> All values that are larger than the displayed maximum value are automatically pooled into the last bin.

7.3.5.3 Results Line Chart



A line chart enables the display of the results of several rounds. The line chart automatically calculates the average of the input variable over all subjects. If the input variable is a discrete choice variable, the result is displayed as a percentage of choices. In the example, you can see a public goods game with three groups. The graph allows for zooming.



The following settings are available:

variable
: The variable which should be displayed.

minimum/maximum/label x-axis
: Minimum, maximum and label of the x-axis

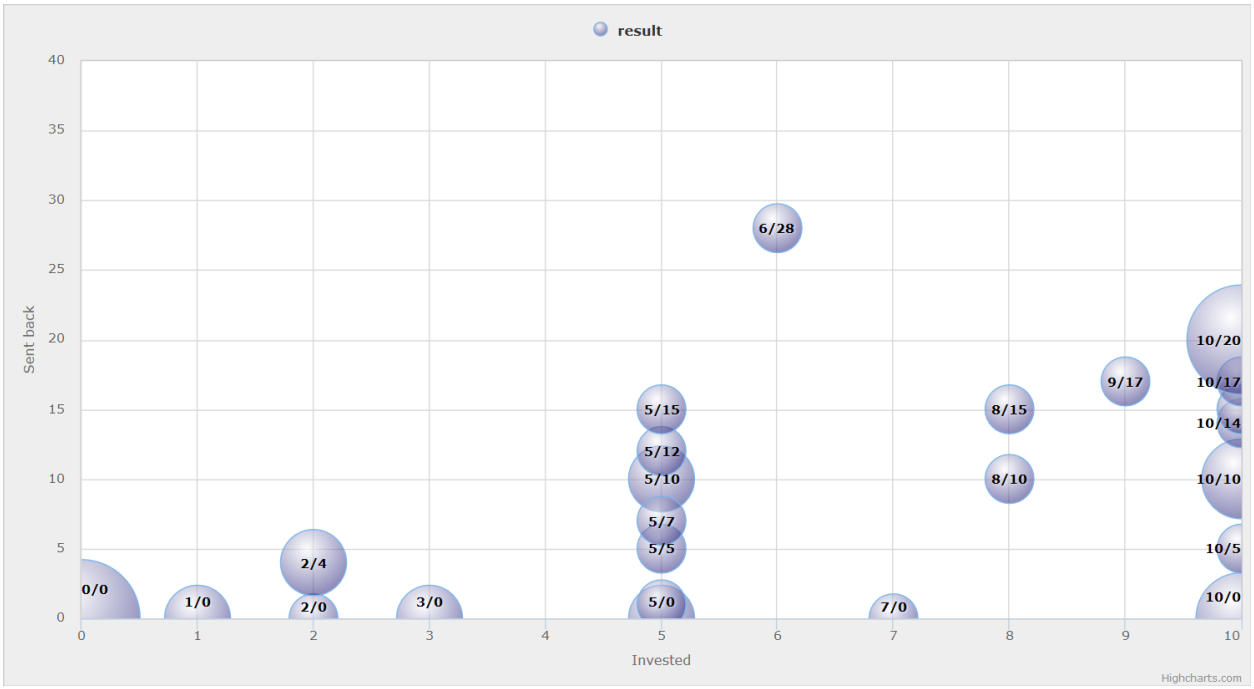
minimum/maximum/label y-axis
: Minimum, maximum and label of the y-axis

show element
: Always display the element, or only if the stage is activated.

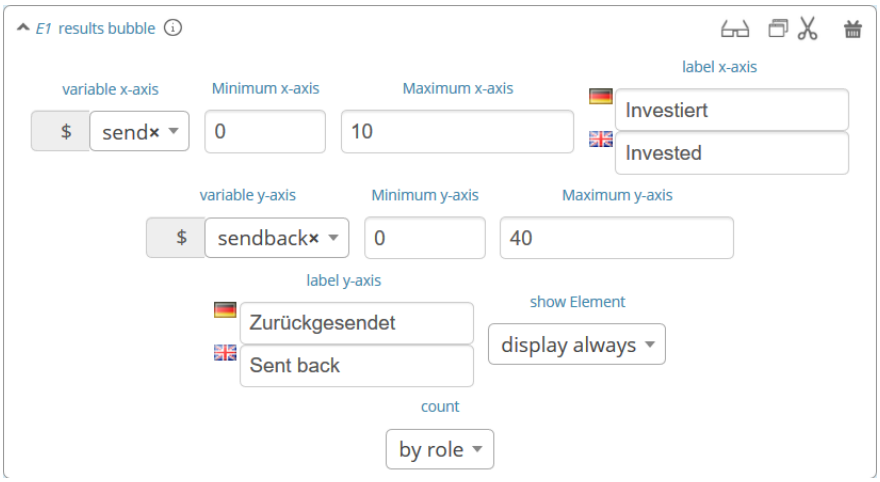
input with array (globals)
: If you provide the name of a globals variable here, the variable setting is overwritten and the data is taken directly from the globals array. The array should have the x-value as the index and the respective y-value.

count
: Participants are counted altogether (or per treatment/role/group).

7.3.5.4 Results bubble



The bubble chart displays the linkage between two continuous decisions, e.g. amount sent and amount returned. The size of the bubble shows how often the number was chosen. The number in the bubble shows x-value / y-value. If you display by role, group or treatment, bubbles in different colors are displayed. Hovering over the bubble shows the total number of observations. The graph allows for zooming.



variable x-axis and y-axis
: The variable which should be displayed on the x-axis and y-axis.

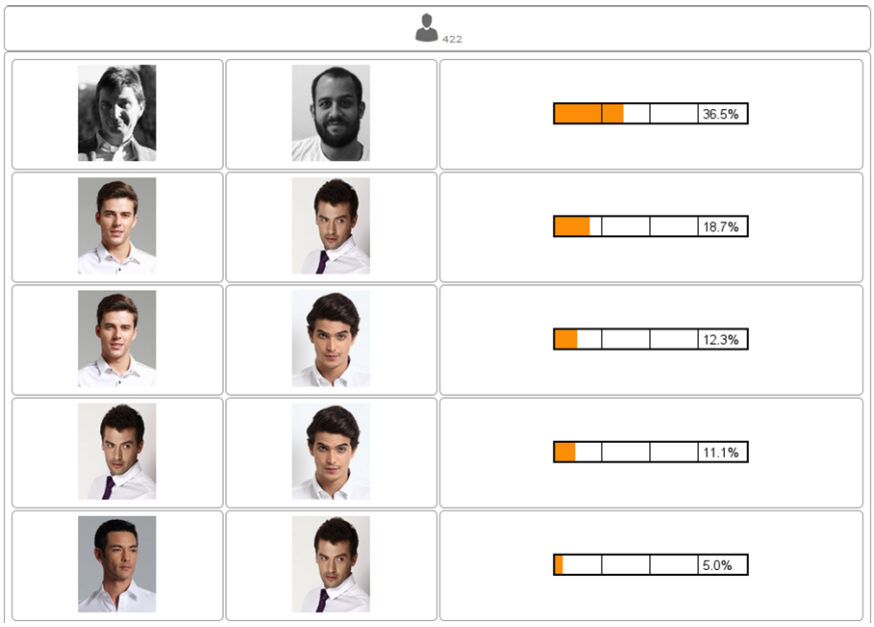
minimum/maximum/label x-axis
: Minimum, maximum and label of the x-axis

minimum/maximum/label y-axis
: Minimum, maximum and label of the y-axis

show element
: Always display the element, only if the stage is activated or only if it's activated and after.

count
: Participants are counted altogether (or per treatment/role/group).

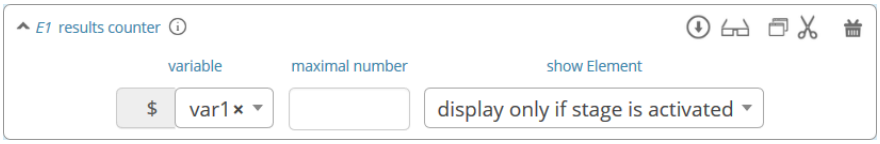
7.3.5.5 Results counter



The counter enables you to display the relative frequency with which a specific answer was chosen. Hovering over the bars gives the absolute frequency of participants who opted for that option. The counter can be useful if the set of answers is open (e.g. text input). It lists all variable inputs according to their frequency.

Note

Using a multiple choice input field will result in the listing of combined answers. E.g. You can select A, B, C (multiple choice). Then the counter element will display how many per cent chose A, A&B, A&C,... If you want to have the items analyzed separately (only A, B, C), you should use *Results single/multiple choice questions*.

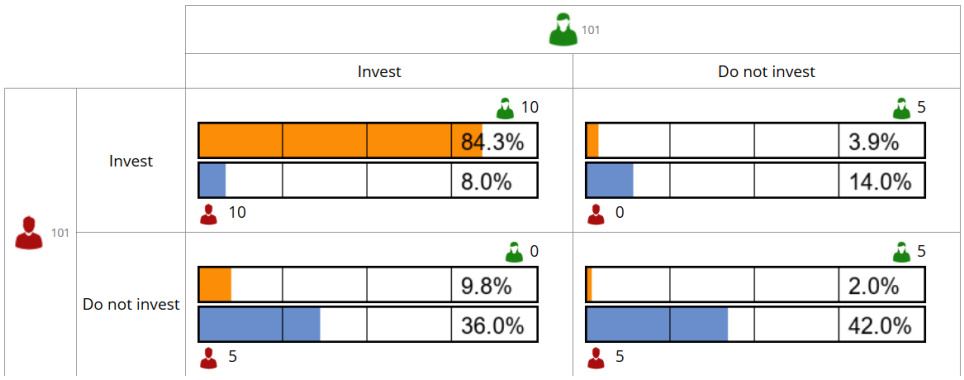


variable
: The variable which should be displayed.

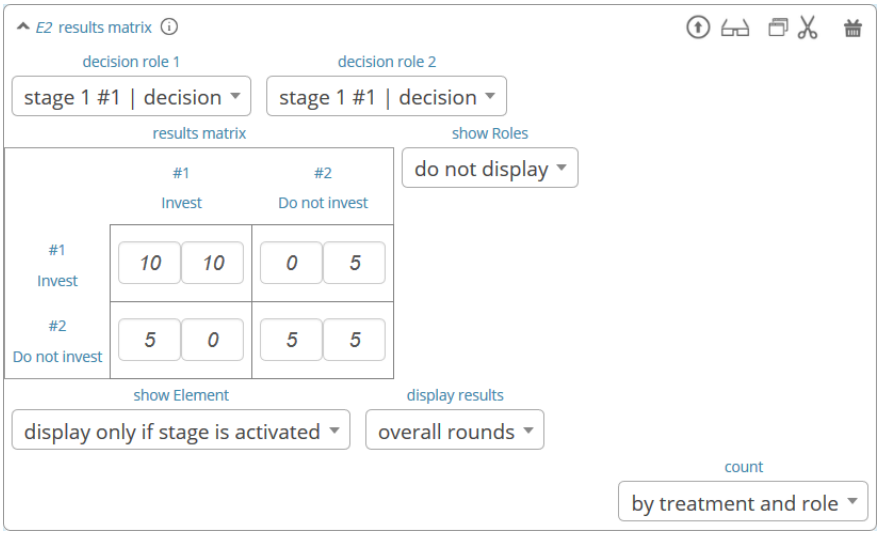
maximal number
: This limits the maximum number of answers to be displayed. E.g. if this is set to 10, only the 10 most frequent answers are shown.

show element
: Always display the element, only if the stage is activated or only if it's activated and after.

7.3.5.6 Results game matrix



The matrix displays the linkage between two discrete decisions. If a game, e.g. is played with two different participants, the results can be displayed as a matrix. Or if a participant makes two different decisions, the linkage between the two can be shown.



decision role 1
: The decision for the row participant.

decision role 2
: The decision for the column participant.

results matrix
: Here you can specify the payoffs for each combination. The first value is the row participants, the second value is the column participants.

show roles
: This setting determines if the role figure is displayed next to the payoffs in the matrix table.

show element
: Always display the element, or only if the stage is activated.

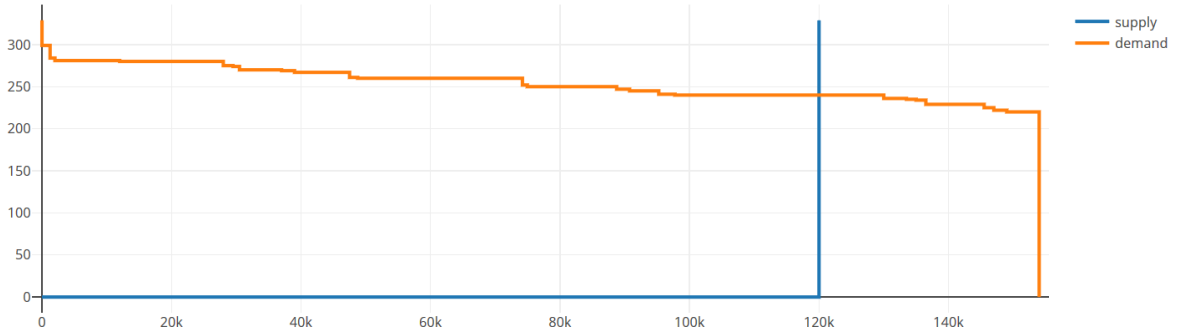
display results
: Here you can choose if results should be pooled over all rounds or displayed separately for each round.

count
: Participants are counted altogether (or per treatment/role/group).

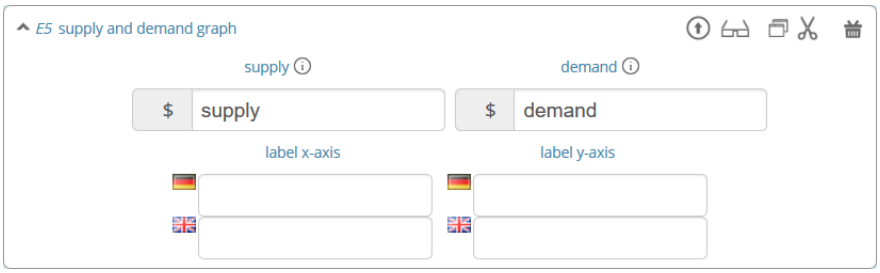
Note

The displayed matrix only determines the image on the lecturer's screen and not the payoff for participants. The payoff is calculated individually for the participants (either through the element *Payoff matrix game* or through a subjects program).

7.3.5.7 Results supply and demand



This element displays a supply and demand graph. It uses arrays, which are calculated in a globals program to display supply and demand. The graph allows for zooming.



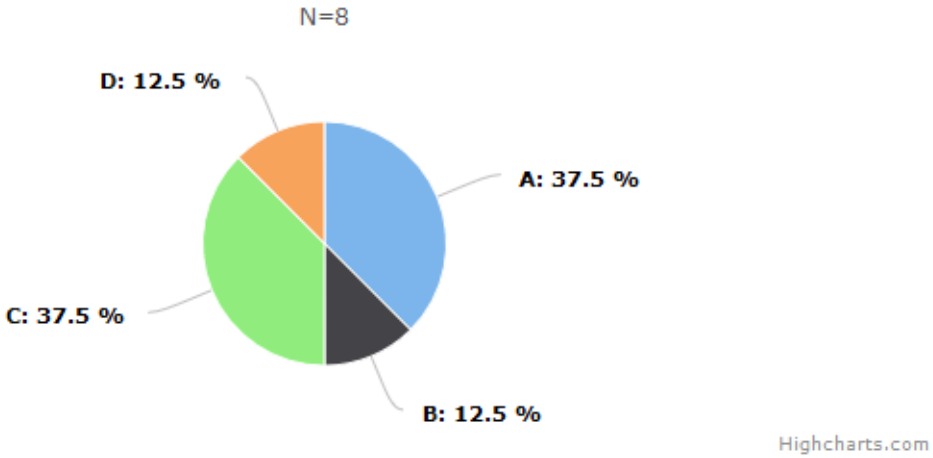
supply
: This is the name of the globals variable which contains the supply data. The index of the array is the price, and the value of the array is the number of suppliers which are willing to supply at this price. E.g. if you have 2 sellers with seller cost 20 and 4 sellers with seller cost 40, the supply array should be `$supply = array(20=>2, 40=>4);`. Note that the number of sellers is not cumulative across prices. This is automatically done by classEx. If you provide a two-dimensional array, you can plot more than one supply line, e.g. `$supply = array(1=>array(20=>2, 40=>4), 2=>array(10=>3, 20=>1));` draws two supply lines.

demand
: This is the name of the globals variable which contains the demand data. The index of the array is the price, and the value of the array is the number of demanders who are willing to buy at this price. The logic follows the description for supply.

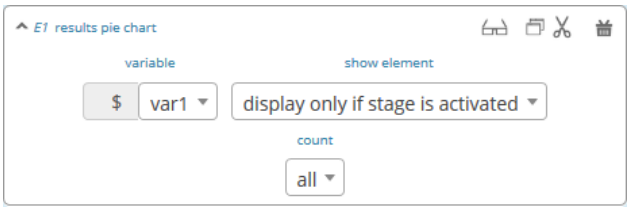
label x-axis/y-axis
: Labels can be provided for both axes.

count
: Participants are counted altogether (or per treatment/role/group).

7.3.5.8 Results pie chart



The pie chart shows the distribution of discrete choices as a pie chart. Slices of the pie can be highlighted by clicking on them. The graph allows for zooming.



variable
The name of the variable

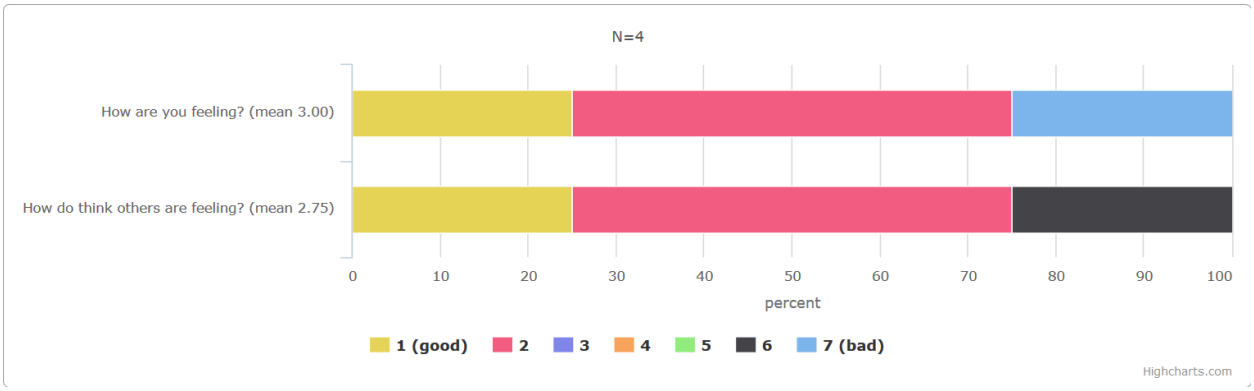
show element
Always display the element, or only if the stage is activated.

count
Participants are counted altogether (or per treatment/role/group).

Note

If you provide a numeric input as a variable, classEx plots all numeric inputs from the stage in the pie chart. This only makes sense if you ask for percentages (which add up to 100%). For example, see the game Ricardian equivalence in the macroeconomics folder.

7.3.5.9 Results Likert scale



This graph allows you to show data from Likert scales if the input was provided with a *Radioline* or *Slider*. It shows the distribution and provides the mean. The graph allows for zooming. The graph allows for showing multiple variables in one graph.

first variable / last variable
: Here you can select the variable(s) to be displayed. If you select, e.g. variable #1 to #3 from a stage, the graph shows all three variables #1, #2, and #3. If you want to show only one variable, just select the same variable for the first and last variable.

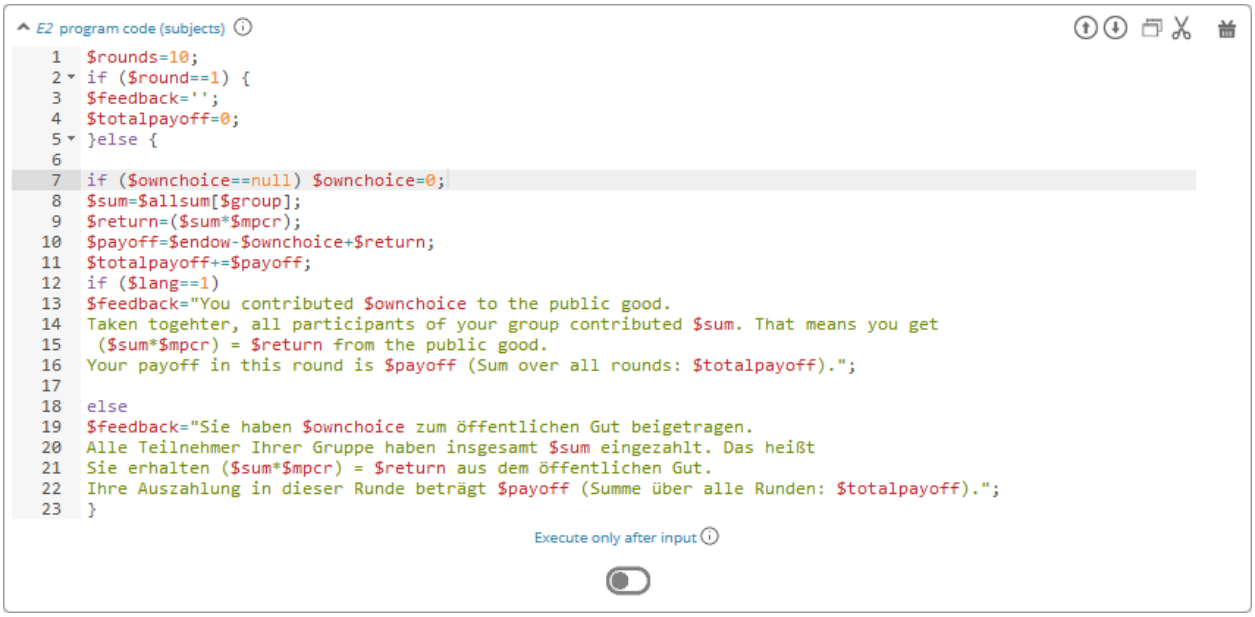
count
: Participants are counted altogether (or per treatment/role/group).

CHAPTER

EIGHT

PROGRAMMING

Programs are very useful for designing dynamic games. Programs are elements of stages and therefore created like any other element (see *Define Elements*).



There are two types of programs: subjects and globals. Subjects programs are executed for each participant, global programs are executed once for all participants.

> **Note**
>
> It is always good to comment the code if you try to understand it later. Add comments with `/* comment */` to your program code.

8.1 Programming language

Variables and programs are specified via PHP. Currently, we use PHP 7.0. This is a well-documented standard which enables easy programming. Details can be found on the internet, for example, here. You can utilize the normal standard PHP functions (e.g. `round()`, `rand(...)`, `number_format()`,...).

Programs are entered in an editor that comprises syntax-highlighting as well as a simple error check of the entered code.

Furthermore, the editor contains a completion system which will show you all available variables. If you start entering the beginning of a variable ($...) and then press Ctrl+space, the automatic completion system will show you all corresponding variables and functions.

Warning

You can not declare your own PHP functions in classEx.

8.2 Declaration of variables

Variables are defined by starting with $. It does not matter whether the variable is a number or text. Variable names are case sensitive.

```
/* numeric variable */
$endowment = 10;

/* string variable with a variable */
$info = "Your endowment is ".$endowment." Euros.";

/* array */
$values = array(1=>"one", 2=>"two", 3=>"three");
```

Warning

Do not use single quotes within double quotes as this may produce errors (e.g. $text="don't"), instead of ' you should use the HTML code ' (e.g. $text="don't") in texts.

8.3 Type and scope of variables

There are two different scopes: globals and subjects variables.

Globals variables are

- available for all participants (can be accessed by subjects program),
- are calculated at the lecturer's side,
- are the same for every participant,
- are calculated first (i.e. before subjects variables).

Subjects variables are

- only available for a specific participant
- saved by default if they are decision variables (set via input elements).
- not saved by default if you create or calculate them in subject programs; to do so use the *Function to save variables*

Note

Please notice that globals and subjects variables share the same namespace. Using the same variable name may overwrite variables. E.g. if you define `$payoff=1;` as a globals variable, and `$payoff=2;` as a subjects variable,

the first will be overwritten by the latter as globals are executed first.

Note

Subjects variables are only available to a specific participant. This means if you want to use the decision of one participant in another's screen, you have to use the functions below to retrieve the decision (e.g. from the partner or group member). You can also retrieve decisions as a globals variable (which then are available to all participants) and retrieve the globals variable for a specific participant.

In addition to globals and subjects variables, there is a third table where data is stored in classEx - contracts. Contracts are always concluded between a buyer and a seller, and a contract contains a price and a quantity. Some of the functions below help to retrieve contract data.

8.4 Execution, Synchronization and Lifetime

8.4.1 Execution

Variables and program code are always executed in a sequential order. This means that elements which come first are executed first.

The overall order is the following:

- First, parameters are set as globals variables (see also *Parameters*).
- Second, globals programs are executed (in the order stated for the lecturer screen).
- Third, subjects programs are executed (in the order stated at the participant screen). Subject programs are executed before other elements are displayed.

Subjects programs offer the option to delay them after the decision has been made. This is provided by the setting *execute only after input*. This means that the subjects program is not executed on the loading of the respective screen, but only after a participant submits his or her decision.

Note

This feature can be useful if you want to do some calculations with the current input before the next stage has started. You can, e.g. add to inputs `$a` and `$b` and save them as a new variable with `$save('sum',$a+$b)`. Logically, this can only be done once the inputs are provided. In some cases, it might be useful to have the sum already available in the next stage (e.g. to display the sum in a graph).

8.4.2 Synchronization

It is always important to keep in mind how synchronization works in classEx in order to retrieve variables at the correct moment in time. Within one stage, participants make their decisions (and therefore create their input variables) at any point in time. So you do not know if the value has been set or not. For this reason, you should only retrieve values in the following stage.

E.g. if you want to display the input `$a` of participant A to another participant B, you can only use `$other = $findVariablePartner("a");` in the next stage and not in the current one. It may be the case that A has not made his or her decision while B is trying to retrieve it. The same holds true if you want to display the input `$a` in a result graph or do some calculations with it in a globals program.

One exception is the usage of *execute only after input* in the section *Execution*. This allows for some calculations after the input of a participant. Still, keep in mind that you do not know when this program will be executed, as the participant may submit his or her input at any point in time.

Another exception is that you can repeat globals programs every 2 seconds. This can be set by selecting *update every 2 s* next to the program element. Then the program is executed every 2 seconds, and calculations are updated. This allows for getting input decisions in real time.

8.4.3 Lifetime

Variables and their values can be used after their declaration throughout the whole game. They can be read (and also overwritten) at any point after their declaration. After the last stage of the game, all subjects variables (which are not decision inputs or were stored) are deleted. Globals variables are automatically stored.

Subjects values can only be changed by subjects programs, and globals values only by globals programs.

If you e.g. set `$a=1;` in stage 1 (as a globals variable), you can use this value in all stages after stage 1 as well. Keep in mind that for subjects variables, this only holds true for the participant's own variables.

8.5 Description of functions in the documentation

Functions are described in the following way. It follows the standard way of documenting functions. Let's take the following example:

```
$findVariablePartner('varname',  $roundselected = $round, $partnerRole = null, $no_
↪decision = null);
```

Function name
The name of the function is *$findVariablePartner*.

> **Note**
>
> Notice that in contrast to standard PHP functions, internal functions in classEx always start with a $ sign.

Arguments
A function has arguments, which are the values the function is called with. In this case, the function has four arguments. The first argument is mandatory; the other three arguments are optional. Arguments can be strings, numbers or variables.

Arguments without default value (mandatory)
Arguments which are **not** marked with a = sign and a default value are *mandatory*. This means you have to specify them in order to make the function work. In the example, you have to specify the variable name `'varname'`. The quotes indicate that you have to specify it as a string.

> **Note**
>
> Note that variable names in functions are specified without the $ sign.

Arguments with default value (optional)
All the other arguments in the function have default values, which means that they are optional. You can specify the function with only one parameter as well. The values after the = sign are taken as default. In the example, the variable `$currentRound` (which is available as a pre-defined globals variable) is taken as the default for the round. If you want to use a different round, you have to overwrite the default value. The same holds true for the other arguments. `$partnerRole` and `$no_decision` are set to `null` as a default, where `null` means no value.

Here are some examples:

```
/* This gives the value of the third round. */
$findVariablePartner('varname', 3);

/* This gives the value of the previous round. */
$findVariablePartner('varname', $round - 1);

/* This gives the value of the current round for partner role 2. */
$findVariablePartner('varname', $round, 2);

/* This gives the value of the current round and returns 0 in case of no decision. (
↪$partnerRole is set to its default.) */
$findVariablePartner('varname', $round, null, 0);
```

Note

If you want to change some of the default values in arguments at the end of the function, you also have to specify the arguments before the argument you want to change. You can see this in the last code example, where we want to leave `$partnerRole` on its default value and only change `$no_decision`.

8.6 Internally used variable names

Some variable names are internally used in classEx and **should not be used as own variable names**. These are the following variables:

```
$newMatchedPartner, $newMatchedGroup, $currentStage, $lastStage,
$sendprog, $history, $simpleID, $signRound, $idCourse, $player
```

8.7 Variables for participants (subjects)

8.7.1 Pre-Defined Variables

The variables `$group`, `$role` and `$treatment` can be overwritten in a subjects program.

Note

Pre-defined variables are not saved automatically in the subjects table. Therefore, they can only be retrieved with e.g. `$findVariablePartner(...)`, `$getValues(...)` or other functions if they are saved before. This is explained in Function to save variables.

8.7.2 Functions to retrieve variables

The following functions can be used to retrieve subjects variables.

```
$findVariablePartner('varname', $roundselected = $round, $partnerRole = null,
$no_decision = null);
```

Function retrieves the variable from another participant in the same group. The function makes sure that participants always get feedback, which can be important in order to avoid disappointing participants. Certainly, cloned or random observations may have to be deleted prior to using data for research.

Returns variable of the other participants. In case the other has not made a decision, it tries to clone a decision from a different participant who has the respective role but is in a different group. If `$no_decision` is specified, the function returns the value of `$no_decision` if no value is available. In this case, the function does not look for a cloned decision.

> **Note**
>
> The function may retrieve a value if the other participant submitted an empty form. In this case, the value is an empty string `""` or `null`.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.
- `$partnerRole` the role of the partner can be specified. E.g. in a group of 3 participants (role 1, role 2 and role 3), it is necessary to specify from which partner to take the variable from.
- `$no_decision` can be used to provide random values in case no decision was made.

Examples are provided in the section *Description of functions in the documentation*.

```
$findGroupAverage('varname', $roundselected = $round, $includingOwn = false)
```

Function retrieves the average of a variable for the own group.

Returns the average value, or 0 otherwise.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.
- `$includingOwn` specifies if the own value should be included or not. `$includingOwn = true` means the own value will be included.

```
$findGroupSum('varname', $roundselected = $round, $includingOwn = false)
```

Function retrieves the sum of a variable for the own group.

Returns the sum, or 0 otherwise.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.
- `$includingOwn` specifies if the own value should be included or not.

```
$findGroupFreq('varname', $roundselected = $round, $includingOwn = false)
```

Function retrieves the frequency of each value of a variable for its own group.

Returns an array with the value as index and the frequency as value. E.g. `$returnedValue = array(1=>12, 2=>13)` would indicate that the value 1 was chosen 12 times, and the value 2 was chosen 13 times. If no decisions were made, an empty array is returned.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.
- `$includingOwn` specifies if the own value should be included or not.

`$findSold($roundselected = $round)`

Function retrieves the number of items sold (in a contract element).

Returns an array with the unit number as index and the corresponding price as value, or an empty array otherwise. E.g. `$returnedValue = array(1=>12, 2=>13)` would indicate that the first unit was sold for 12, and the second unit was sold for 13. If no decisions were made, an empty array is returned.

Arguments are:

- `$roundselected` the round from which the variable should be retrieved.

`$findBought($roundselected = $round)`

Function retrieves the number of items bought (in a contract element). The logic is exactly the same as in `$findSold(...);`

`$findSoldTo($roundselected = $round)`

Function finds the IDs of the trading partners who bought something in a round.

Returns an array with the unit numbers of the bought goods as index and the IDs of the trading partners that bought the items.

- `$roundselected` the round from which the variable should be retrieved.

`$findBoughtFrom($roundselected = $round)`

Function finds the IDs of the trading partners who sold something in a round. The logic is exactly the same as in `$findSoldTo(...);`

`$findOldVariable('varname', $idRound = null)`

Function Find old variables from previous rounds, Variable or 0 if not found.

Here you can find some coding examples:

```
/* get group average of variable test, from previous round, including own */
$groupAvg = $findGroupAverage("test", $round - 1, true);

/* get sum of variable test2, from current round, not including own */
$groupSum = $findGroupSum("test2");

/* get frequency of value 1 in variable choice */
$groupFreq = $findGroupFreq("choice");
echo $groupFreq[1];

/* get all items sold in the previous round and calculate total revenues and amount sold␣
↪*/
$sells = $findSold($round-1);
$revenues = 0;
$amountSold = 0;
foreach ($sell as $unit => $price) {
        $revenues += $price;
        $amountSold++;
}
```

8.7.3 Function to save variables

If you want to retrieve variables, those variables have to be stored beforehand. For decision inputs, this happens automatically. If you want to retrieve variables defined or used in the subjects program, you have to save them beforehand by using the following function:

```
$save('varname', $value, $roundselected = $round);
```

Function stores a value in the subjects table.

Returns true if the storage was successful, false in case of an error.

Arguments are:

- `'varname'` the variable name (mandatory).
- `$value` the value to be stored. The value can also be a variable itself.
- `$roundselected` the values are saved for a certain round. If you, e.g. calculate a result in round 6 according to inputs made in round 5, you can enter $round = $currentRound-1 in order to save the result in the round it belongs to.

Here you can find some coding examples:

```
/* store the value 1 as variable "shown" */
$save("shown", 1);

/* store the value 7 as variable "test" in the previous round */
$a = 7;
$save("test", $a, $roundselected = $round-1);
```

> **Note**
>
> Keep in mind that you should only retrieve variables at least one stage after saving them. See *Synchronization*.

8.7.4 Function to change role or group ID

It is possible to change the role or group ID of a subject. This enables you to create your own matching algorithms and assign new groups to your subjects. Treatments however, cannot be changed this way.

`$resetGroupNr($value);`

> **Function** overwrites the $group variable in the subjects table.
>
> **Returns** true if overwriting was successful, false in case of an error.
>
> **Arguments** are:
>
> - `$value` the new group ID for the subjects. The value can also be a variable itself or an array. Usually, the most convenient way is to create a matching algorithm in a globals program code returning an array (e.g. called $any_array_name) with participants' IDs as keys and the new group IDs as values. The code for the reset-function could then be `$resetGroupNr($any_array_name[$id]);`

`$resetRole($value);`

> **Function** overwrites the $role variable in the subjects table.
>
> **Returns** true if overwriting was successful, false in case of an error.
>
> **Arguments** are:
>
> - `$value` the new role for the subject.

..note:: Both functions change the $role or $group variable of the respective player and also change the table containing all the matches.

8.8 Variables for lecturers (globals)

8.8.1 Pre-defined variables

All parameters are also available as pre-defined globals variables. All globals variables (including the ones calculated in globals programs) are stored automatically.

The globals variable `$maxWin` is used for limiting the maximum which is paid out via *Winner's draw*.

8.8.2 Functions

The following functions can be used to retrieve globals variables.

`$getValues('varname', $roundselected = $currentRound)`

> **Function** retrieves the values of one variable for all participants.
>
> **Returns** an array with the internal participant ID as index and the respective values or `null` if no values available.
>
> **Arguments** are:
>
> - `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
> - `$roundselected` the round from which the variable should be retrieved.

> **Note**

The function can only retrieve subjects variables which were saved before or which were decision inputs. Predefined variables (like role, group and treatment) can not be retrieved with this function. For these functions use `getRoles()`, `getTreatments()`,... (see below).

`$getTimes('varname', $roundselected = $currentRound)`

Function retrieves the time participants need for a decision concerning a certain variable in seconds.

Returns an array with the internal participant ID as index and the respective decision time in seconds or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

`$getFreq('varname', $roundselected = $currentRound, $multiple = false)`

Function retrieves the frequencies of values over all participants for one variable.

Returns an array with the value as index and the frequency. If not available, it returns an empty array.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.
- `$multiple` If multiple is set to true, answers from multiple-choice questions are decomposed into single answers.

`$getAverage('varname', $roundselected = $currentRound)`

Function retrieves the average value of a variable over all participants.

Returns the average value or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

`$getMedian(array)`

Function retrieves the median value of an array.

Returns the median value or `null` if no values available.

Arguments are:

- `array` an array with data (mandatory). The function needs an array with data (this can be, e.g. retrieved with the getValues function).

`$getAveragePerRole('varname', $roundselected = $currentRound)`

Function retrieves the average of a variable over all participants grouped by their role.

Returns an array with role ID as index and the average value for each role. If not available, it returns `null`.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

`$getAveragePerTreatment('varname', $roundselected = $currentRound)`

Function retrieves the average of a variable over all participants grouped by treatment. The logic is the same as for `$getAveragePerRole(...)`.

Returns an array with the treatment ID as index and the average value for each treatment.

`$getAveragePerGroup('varname', $roundselected = $currentRound)`

Function retrieves the average of a variable over all participants grouped by group. The logic is the same as for `$getAveragePerRole(...)`.

Returns an array with the group ID as index and the average value for each treatment.

`$getValuesPerGroup('varname', $roundselected = $currentRound)`

Function retrieves the values of a variable for all participants arranged by groups.

Returns a multidimensional array with group ID as index and another array with the participants ID as index and the retrieved value as key. If not available, it returns `null`.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

`$getVarSum('varname', $roundselected = $currentRound)`

Function retrieves the sum of values of a variable over all participants.

Returns the sum or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

Also available as `$getVarSumPerGroup(...)`, `$getVarSumPerTreatment(...)` or `$getVarSumPerRole(...)` following the same logic as described above for `$getAveragePerRole(...)`.

`$getMin('varname', $roundselected = $currentRound)`

Function retrieves the minimum of a variable over all participants.

Returns the minimum or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

Also available as `$getMinPerGroup(...)`, `$getMinPerTreatment(...)` or `$getMinPerRole(...)` following the same logic as described above for `$getAveragePerRole(...)`.

`$getMax('varname', $roundselected = $currentRound)`

Function retrieves the maximum of a variable over all participants.

Returns the maximum or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

Also available as `$getMaxPerGroup(...)`, `$getMaxPerTreatment(...)` or `$getMaxPerRole(...)` following the same logic as described above for `$getAveragePerRole(...)`.

`$getMostFrequent('varname', $roundselected = $currentRound)`

Function retrieves the most frequent outcome over all participants.

Returns the most frequent or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

`$getRoles()`

Function retrieves the specific role for each participant.

Returns an array with the internal participant ID as index and the respective role. If no roles are available, an empty array is returned.

`$getGroups()`

Function retrieves the specific group for each participant.

Returns an array with the internal participant ID as index and the respective group. If no groups are available, an empty array is returned.

`$getTreatments()`

Function retrieves the treatment for each participant.

Returns an array with the internal participant ID as index and the respective treatment. If no treatments are available, an empty array is returned.

`$getNumRoles()`

Function retrieves the total number of roles.

Returns an array with the role ID as index and the number of participants with this role. If no roles are available, an empty array is returned.

`$getNumPlayer()`

Function retrieves the total number of participants.

Returns the number of participants.

`$getSubjectIDs()`

Function retrieves corresponding subject IDs to participant IDs.

Returns an array with the internal participant ID as index and the respective subject ID. With no participants, it returns an empty array.

`$getNumDecisions('varname', $roundselected = $currentRound)`

Function retrieves the number of decisions made.

Returns the number of decisions made or `null` if no values available.

Arguments are:

- `varname` the variable name (mandatory). The function can retrieve subjects variables which were saved before or which were decision inputs.
- `$roundselected` the round from which the variable should be retrieved.

Also available as `$getNumDecisionsPerGroup(...)` following the same logic as described above for `$getAveragePerRole(...)`. Not available per treatment or per role.

`$setWinner($winnerID, $winnerAlwaysAssigned = false)`

Function adds participants to the payoff table.

Returns true if successful, false if not.

Argument is:

- `$winnerID` the internal participant ID of the winner (mandatory). You can also assign multiple winners with one function call. Then, `$winnerID` should be an array with the internal participant IDs as values. E.g. `$setWinner([$winnerpot[0], $winnerpot[1], $winnerpot[2]]);`.

- `$winnerAlwaysAssigned` can be set to "true" in order to force classEx to make a payoff in this stage. It will make a payoff also if the lecturer enters the stage by clicking on "do not pay out payoffs" or by jumping there with the help of a start button that skipped several rounds.

The amount is set directly by the participant in the *Winner's Notification*.

Note

If the winner is drawn by an own program code, this code has to be placed before the start button of the stage.

```
$getAvgPriceContract($roundselected = $currentRound, $status = 1)
```

Function retrieves the average price of all contracts made.

Returns the maximum or `null` if no values available.

Argument is:

- `$roundselected` the round from which the variable should be retrieved.
- `$status` Only contracts with this status are selected (1 = confirmed, 2 = open, 3 = rejected, 4 = withdrawn)

```
$getNumContracts($roundselected = $currentRound, $perSeller = true, $status = 1)
```

Function retrieves the number of contracts per seller or buyer.

Returns an array with the internal participant ID as index and the number of contracts. With no data, an empty array is returned.

Arguments are:

- `$roundselected` the round from which the variable should be retrieved.
- `$perSeller` If this is true, contracts are counted for sellers. If it is false, contracts are counted for buyers.
- `$status` Only contracts with this status are selected (1 = confirmed, 2 = open, 3 = rejected, 4 = withdrawn)

```
$getContractPrices($roundselected = $currentRound, $seller = true, $status = 1)
```

Function retrieves prices per seller of buyer.

Returns an array with the internal participant ID as index and the prices of each seller or buyer. With no data, an empty array is returned. If a seller or buyer concluded multiple contracts, the sum of prices is returned.

Arguments are:

- `$roundselected` the round from which the variable should be retrieved.
- `$seller` If this is true, contracts are counted for sellers. If it is false, contracts are counted for buyers.
- `$status` Only contracts with this status are selected (1 = confirmed, 2 = open, 3 = rejected, 4 = withdrawn)

```
$getContractQuantities($roundselected = $currentRound, $seller = true, $status = 1)
```

Function retrieves quantities per seller of buyer.

Returns an array with the internal participant ID as index and the quantities of each seller or buyer. With no data, an empty array is returned. If a seller or buyer concluded multiple contracts, the sum of quantities is returned.

Arguments are:

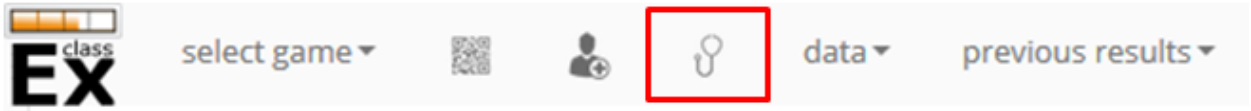
- `$roundselected` the round from which the variable should be retrieved.
- `$seller` If this is true, contracts are counted for sellers. If it is false, contracts are counted for buyers.
- `$status` Only contracts with this status are selected (1 = confirmed, 2 = open, 3 = rejected, 4 = withdrawn)

8.9 Diagnosis tool

Small errors can cause the programs not to run. Therefore, error detection is an important issue. First, have a look at the editor, which provides basic syntax checking. Errors are marked by red crosses. If the program is running but not performing in the expected way, you may use the diagnosis tool to find the error.

With severe errors, the lecture mode can not be started. In order to find the error, go through the code and look for errors. Check if the function names are correctly spelt.

The diagnosis tool is very useful for troubleshooting and testing your game. You can access the diagnosis tool by clicking on the stethoscope icon in the top bar of the lecture mode.



Clicking on the symbol opens up a window beside the usual display on the lecturer's screen, which shows you all variables and their current values.

The different tabs allow you to access the globals or the variables for each participant. This makes programming and error finding much easier than having to jump back and forth between the lecture mode and the editing mode.

8.10 Javascript

Programming in classEx is normally done in PHP. For some applications, it may be useful to also use some JavaScript. Javascript runs on the client side (i.e. on the participant's browser) and also for interactive change of content (with no need to reload the page).

For this reason, you can also add some JavaScript code by adding a JavaScript element. An additional option is to put JavaScript code directly in the text box encapsulated by `<script></script>`. You can also use jQuery, which is loaded automatically for each page.

8.10.1 Manipulation of HTML elements

All elements in classEx have their own identifiers so that they can be used for JavaScript interaction.

Input fields have the identifier "fieldX", where X is the number of the input field (#1, #2,...). An example would be `<input type="hidden" id="field1" name="bid1" value="">`. If you want to do some JavaScript interaction, you can manipulate the value of the input field, e.g. with `$("#field1").val(1);`. Then the value is set to 1.

Buttons have the identifier "buttonX" where X is the number of the stage. The number of the stage can be found by hovering over the tabs in the editing mode. This allows you to add click events to the standard button, e.g. `$("#button1234").click(...)`.

You can also define your own parts in the text boxes, which can be manipulated with JavaScript. E.g. define `<span id="text1">This is my text</span>` in the text box. Then you can change this text in a JavaScript program with `$("#text1").html("This is my new text.");`.

8.10.2 Live feedback on input

JavaScript allows, e.g. for live feedback when subjects enter numbers. Let's take an ultimatum game where a participant can split 10 Euros between him or herself and another participant. The participant enters the amount to send to the other in the input field #1. With JavaScript, you can display the amount he or she keeps in real time. Each time the participant changes his input, he or she gets live feedback on the amount he or she keeps.

Just add a text box with the following code.

```
The amount you keep is <span id="keep">--</span>.
```

Then add a little JavaScript program which is executed when you change the input field.

```
$("#field1").keyup(function() {
        /* JavaScript takes the input as a string, which has to be parsed toan  Integer␣
↪or Float to
                make calculations with */
        let send = parseInt($(this).val());
        let keep = 10 - send;
        $("#keep").html(keep);
});
```

Alternatively, you can also add another input field, which is marked with *output only*. E.g. you added such an input field as input field #2. Then you do not have to add the text box, but only the JavaScript program with:

```
$("#field1").keyup(function() {
        /* JavaScript takes the input as a string, which has to be parsed to an Integer␣
↪or Float to
                make calculations with */
        let send = parseInt($(this).val());
        let keep = 10 - send;
        $("#field2").val(keep);
});
```

Note that for input fields you have to use `$("#field2").val(keep)` instead of `$("#keep").html(keep)` for changing HTML.

If you want to add live feedback for sliders, radiolines or buttons, you have to use `$("#field1").change(...)` instead of `$("#field1").keyup(...)` for fields where you type the input.

8.10.3 Reading PHP variables

As all variables are only stored in PHP, you need to get the values for the JavaScript programs by hand.

To read PHP variables, one currently needs a two-step approach:

- write a PHP variable in a (hidden) text field
- parse text field content in JavaScript

Assume we have a PHP variable `$foo` that contains a value we want to use in JavaScript.

Then you should add a text box with:

```
<span id="php_var_foo" style="display: none;">$foo;</span>
```

The ID does not need to have this format, but it must be unique.

In JavaScript, you can retrieve the variable with:

```
let foo = JSON.parse($('#php_var_foo').html());
```

This command finds the HTML element with the id of the span containing the variable content. Its inner HTML (the content) is taken. Now the variable `foo` in JavaScript contains the content of the PHP variable `$foo`.

8.10.4 Writing PHP variables

This can be achieved via hidden input fields that are filled with JavaScript. Just add a *Hidden field* in your input element.

Input fields have the identifier "fieldX", where X is the number of the input field (#1, #2,...). An example would be `<input type="hidden" id="field1" name="bid1" value="">`. If you want to save a JavaScript value, you write the value into the input field, e.g. with `$("#field1").val(1);`. Then the value is set to 1 and it is stored automatically when the participants submit the input element.

Note

At the moment, PHP and JavaScript are not integrated, so that variables have to be transferred manually from PHP to JavaScript and vice versa. In one of the next versions of classEx, this should be automatized.

Note

In the lecture mode, you can use the JavaScript function `setGlobalVar(varName, value)` to store variables in the global variables.

CHAPTER

NINE

CHANGELOG

9.1 Release 3.7

9.1.1 Major features

- Developed a fully automated registration process for classEx users.
- Optimized the repository list for better search performance.
- Added functionality to view, manage, and delete outdated run data in each game.
- Enabled progress bar feature to display stages of the game during gameplay.
- Enabled a PDF download feature for exporting game materials.
- Added file upload functionality for lecturers within each game.
- Provided a wide-format export of game and player information as a tab in the Excel data.
- Enabled a quick switching button between lecturer mode and new games for better transitions.

9.2 Release 3.6

9.2.1 Major features

- Forwarding of players has been rewritten. The forwarding of players no longer relies on the Lecturer page being active and open on the Experimenter side

9.3 Launch of new Documentation

9.4 Release 3.5

9.5 Release 3.4.5

9.5.1 Major Features:

- camera input
- discrete choice element for lecturers
- new contract element

9.6 Release 3.4.3

9.7 Release 3.4

9.8 Release 3.3.2

- QR code is provided automatically for all experiments.
- The login page was restructured. Prior to selecting the course, the respective institution has to be selected.

9.9 Release 3.3.1

- All elements in the editing mode are directly linked to the documentation, which explains the different features.

9.10 Release 3.3