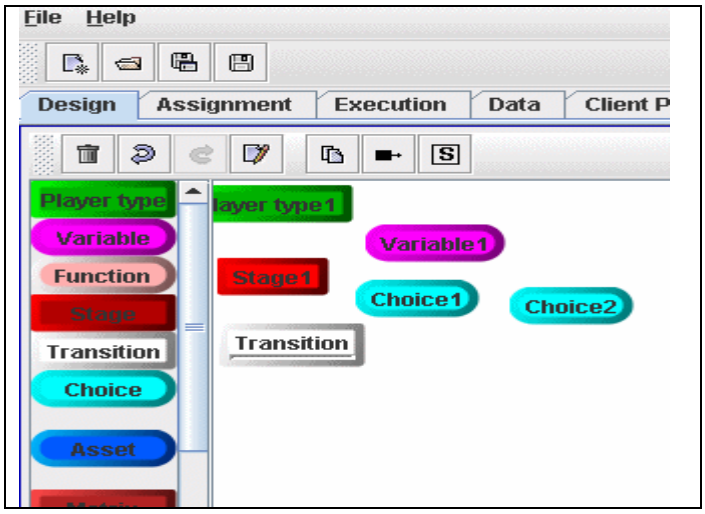
1. Basic manipulation on the design window

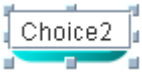
a. Creating elements for your experiment

Drag and drop the appropriate item from the left side of the window anywhere into a design window. The items do not need to be created upfront. They can be added as the design of the experiments progresses.



b. To rename any created item

- Click on an item that was created in the design window to select it : 

, double click on the interior part of the item: 

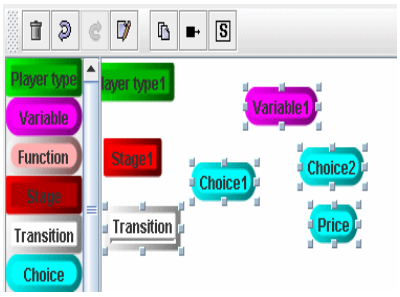
and rename it: 

. Click anywhere on the design window to update the change: 

. Note: in order to design and run an experiment, you do not need to rename items. The program automatically adds a consecutive number to each additional item created: 

.

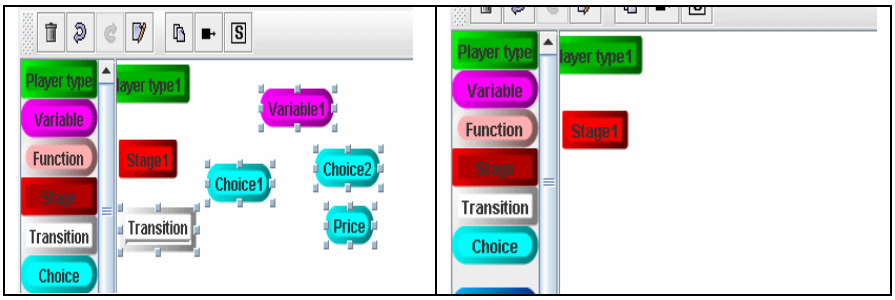
c. To select several items created in the design window, use “Ctrl” and drag the mouse on the design window to select the item or use “Ctrl” and click on each separate item until you selected the desired items (i.e. standard Windows feature for selecting).



d. To delete any created item or a group of items that are in design window:

- Click an item to select it:  and click on  to remove it.

- To delete a group of items, select the appropriate items and click on  to remove them.



e. To move an item or several items, select them, go with the mouse to one of the selected item, click on “Ctrl”, hold it and move the mouse to the appropriate place on the design window.

f. To open an item created in design window:

- Double click on the edge of the item (i.e. the darker color) and the editor for that item will appear.

Or,

- Click on the item to select it (), click on  to edit it.

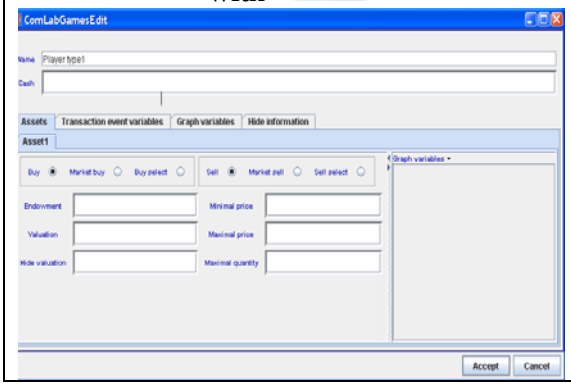
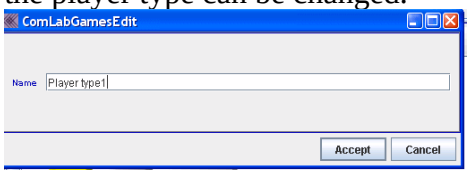
For example, for  editor the following form appears:

- The items that can be edited are:

- The items that can only be renamed:

 . Note that the  color can be changed by double clicking on the edge of the item. 

- The item that can be edited when  is created in a design window and only renamed in all other situations:  . Note the content of  changes when  is created.

 with  	 when  is not created. In this case only name of the player type can be changed. 

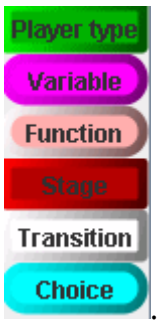
2. Designing games with discrete transitions (bargaining, auctions, contracts, individual choice, questionnaire)

a. Open the program

Select 

window which is the default opitoin.

b. Elements for designing games with 



On the left side of the design window these elements are: 

. is not essential for designing an experiment but it allows moderator to simplify the expressions and can be used in several steps.

To create a game drag and drop the necessary elements.

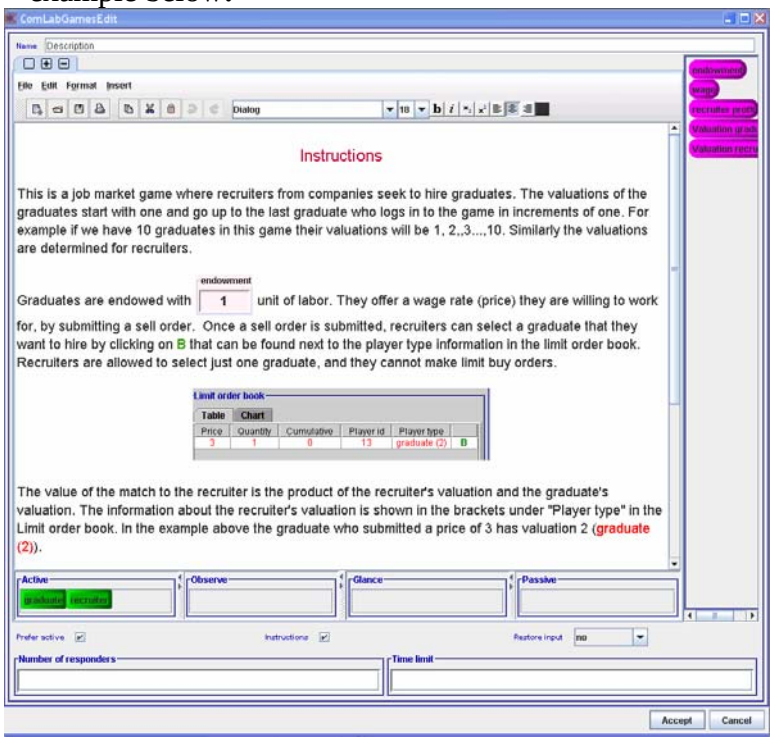
c. Use of 

to write instructions :

- Open 

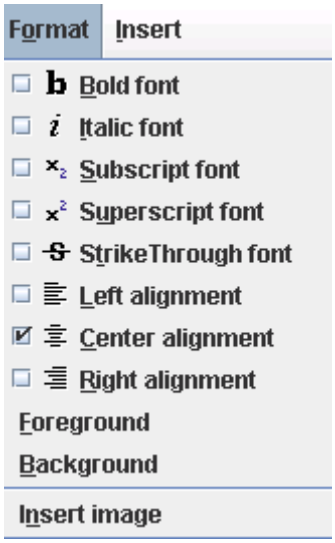
double clicking, select 

that is located in the in the bottom of the editor and write the content of the instructions like in the example below**:**



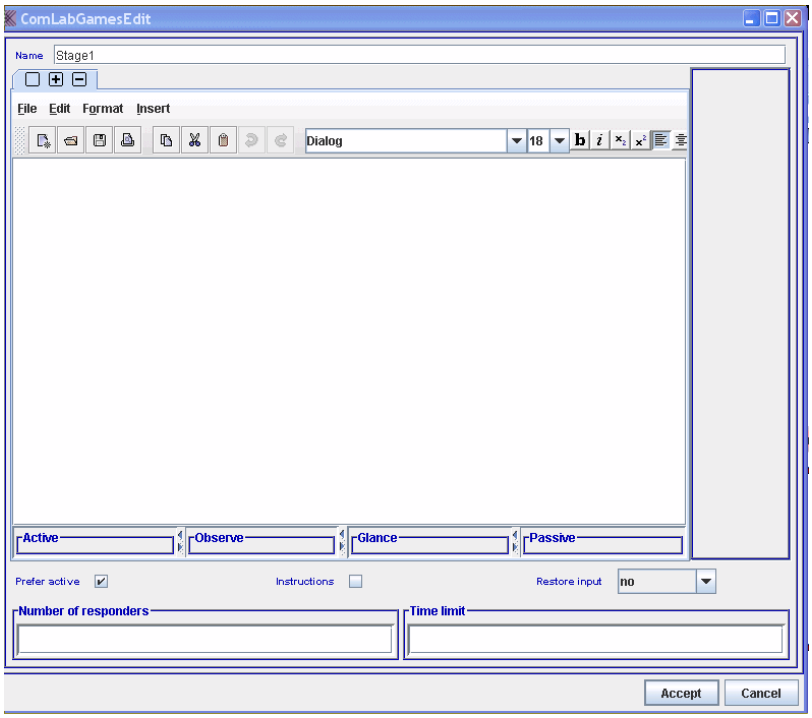
Instruction window is a stand alone Stage. It can be used for writing instructions that accompany Matrix1 games, Tree entry1 games, market games.

- The instructions can have only text, or Variable1, or pictures can be added.
- To insert picture, click on Format and select



-

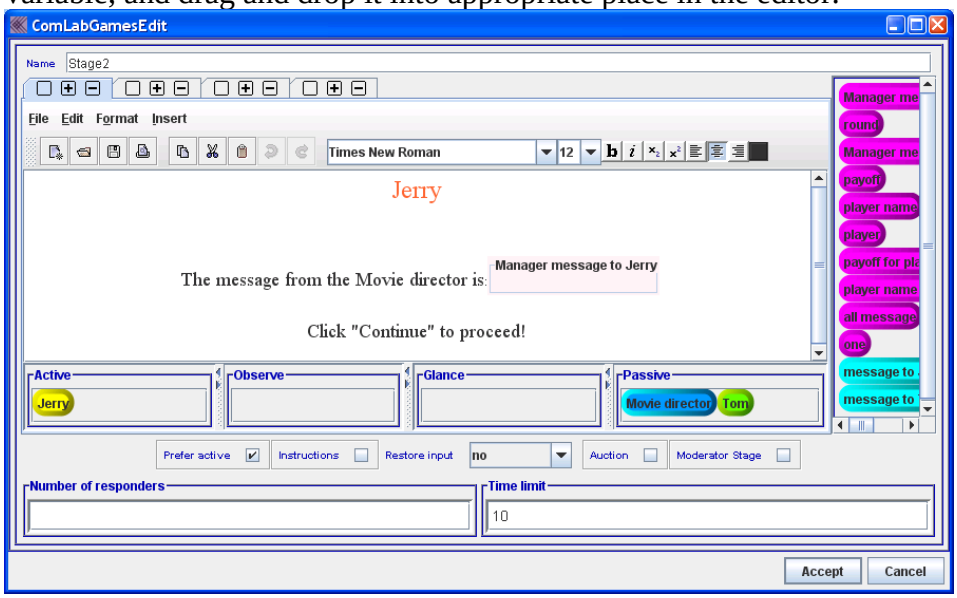
- The window(s) where the content of the experiment is displayed for the moderator and the subject are in Stage.
- To open it, double click on Stage. The window should look like in the picture below.



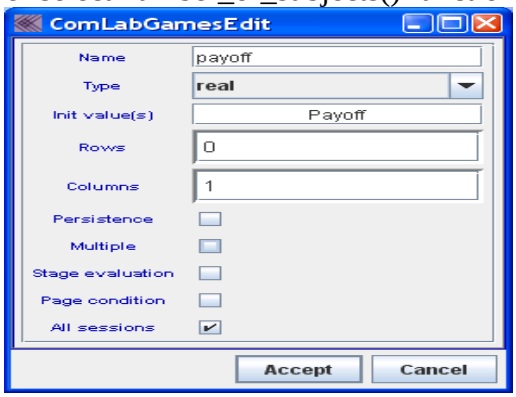
- In the editor write the appropriate information that you want subject to see and to make decisions.
- Add a 

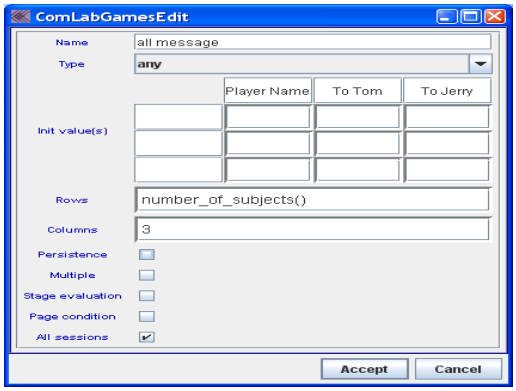
from the list of created choices or first create a new choice, and drag and drop it into appropriate place in the editor.
- Add a 

from the list of created variables or first create a new variable, and drag and drop it into appropriate place in the editor.



- Another stand alone Stage is a moderator stage. Select Moderator Stage to view the decisions subjects made. Note that the variables should have "All sessions' selected. Also number of rows is determined by number of subjects. If 0 is written for Rows that the program automatically assigns the number of rows or select number_of_subjects() function.





d. Transition for determining the path from one Stage to another.

a. Creating Transition between different Stage.

Transition allows moderator to update the variables and to determine the path from one state to the next stage.

- Go with the cursor on Bidding window . will appear below and drag the cursor from to Transition. You should get the connection that looks like this:

Alternative way of adding Transition: Select Stage1 and drag and drop Transition on Design window.

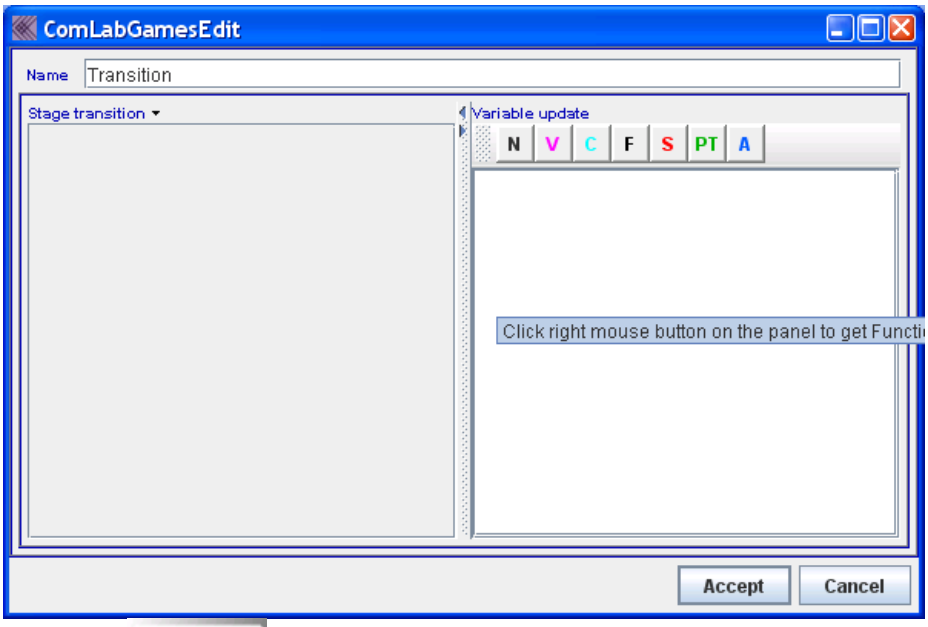
b. Editing Transition

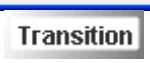
- Open 

by double clicking at the edge or selecting 

and clicking on  . The 

has the following form:



- 

is divided into two parts:
  - The Left side where 

is written allows a moderator to determine a rule from going from the current 

to any other 

. The rule is expressed as Boolean.
  - The right side where 

is written allows moderator to update the variables after the stage to which 

is connected.
  - You have to add 

to the 

in order to proceed to a next 

otherwise the experiment stops after the first stage.
  - The only time we did not need 

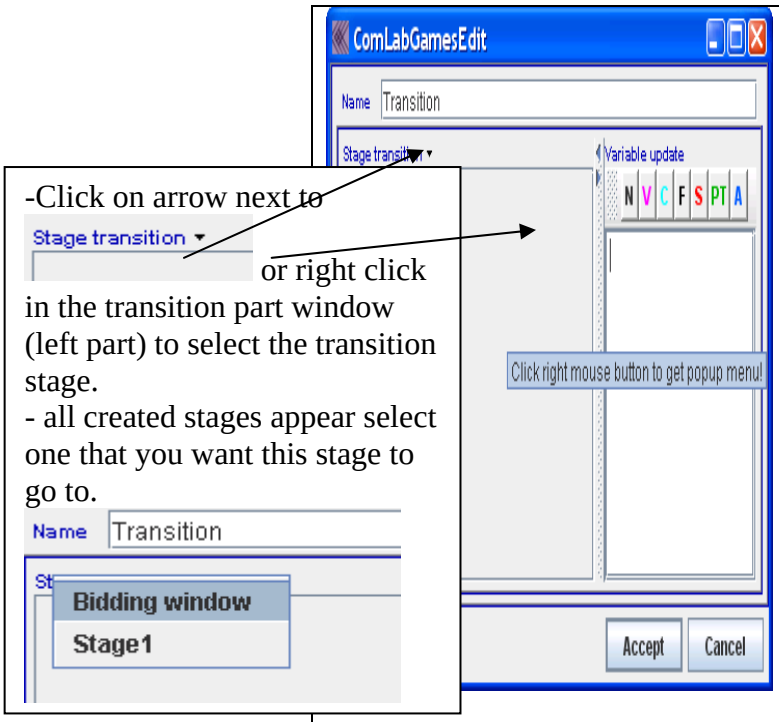
after the stage is when the follow up stage is the “Summary” stage and we do not need to update variables (To designate a stage for a summary stage, select 

and click on 

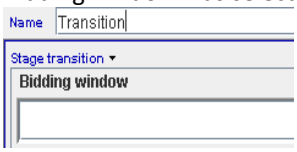
to designate the stage to be a summary

stage: 

. Note the color of letters in the stage is green as opposed to black.

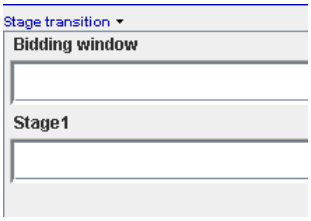


- Bidding window was selected :



- Write a condition to go back to bidding window. If editor under bidding window is left empty it means that with probability 1 the game goes back to 

- You can have several stages that you can go from the current stage. We do not need this for the auctions. Select them in a similar way:



If nothing is written for the condition it means that we will go with probability ½ to 

and with probability ½ to 

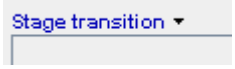
.

- To delete the transition right click on the name of the stage within transition 

and delete it.

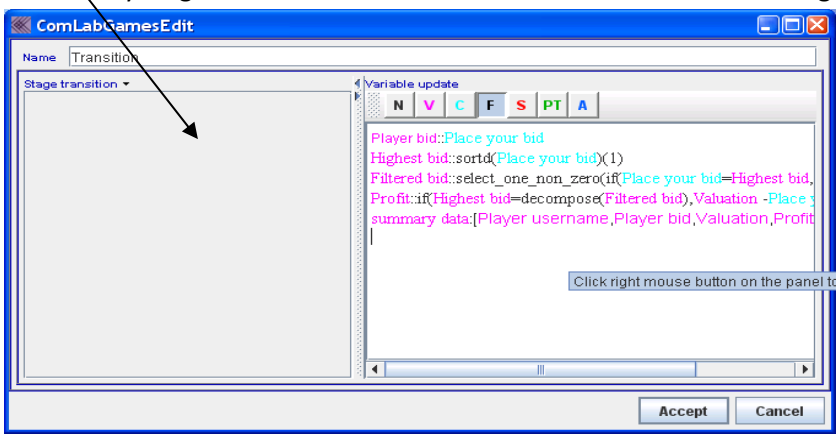
c. Differences in 

setup between different type of auctions

- 

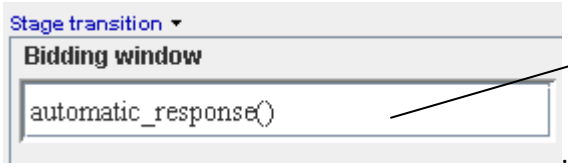
in private value and common value <u>sealed bid</u> auction.

Stage Transition is empty because we go directly to summary stage after Bidding window stage. In these auctions everybody has to make a decision before we go to summary stage. There is no need to return to Bidding window stage.

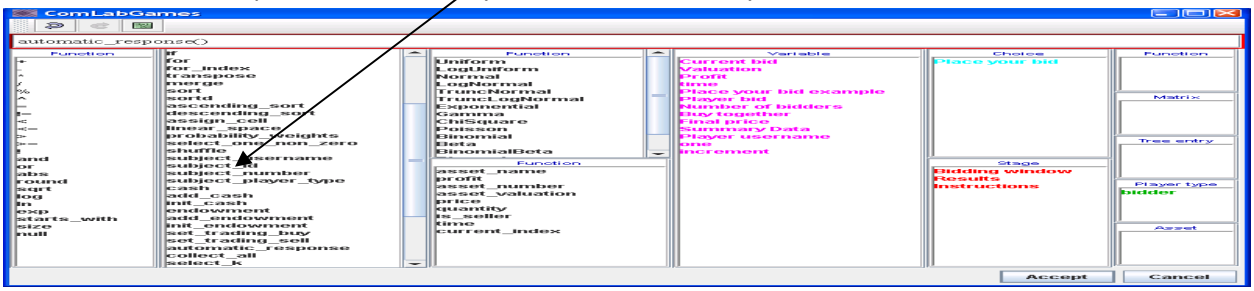


- Stage transition ▾ **in Dutch auction**

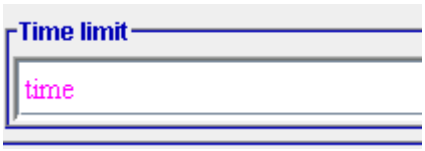
- Select Bidding window for the stage transition and write the condition automatic_response() to return to Bidding window:



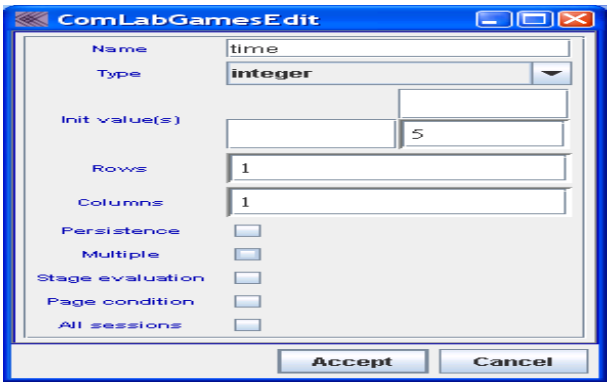
. To get the functions right click in the bidding window editor to get the function editor where all the functions are listed. See below how the function editor looks. Find the function, double click on it and it will appear in the editor line. Then click "Accept". You can directly write the function if you know it.

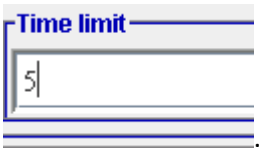


Automatic_response() means that the game will return to Bidding window if nobody responds within time limit set at the bottom of the Bidding window stage:



In our example we created a variable time where the time limit was determined . The initial value was set to 5 seconds which means that after 5 seconds the experiment will return to Bidding window if no subject makes a decision.



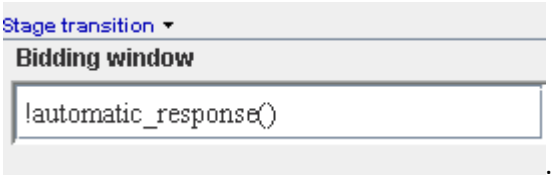


We could have directly wrote . An advantage to create it as a variable is that you can directly change it in the variable and it will update the value wherever it was inserted.

- 

in English auction

- Select Bidding window for the stage transition and write the condition **!automatic_response()** (not automatic response: ! means not) in the editor:



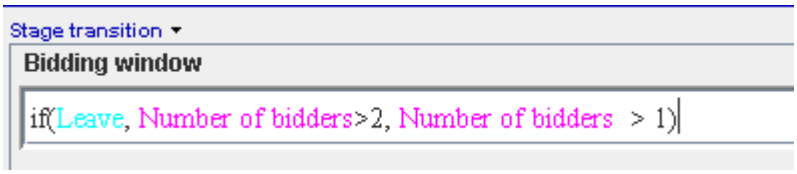
. This condition means that the experiment will return to Bidding window if at least one subject makes a decision, otherwise the game will end. (because Number of responders inside Bidding window was set to 1

)

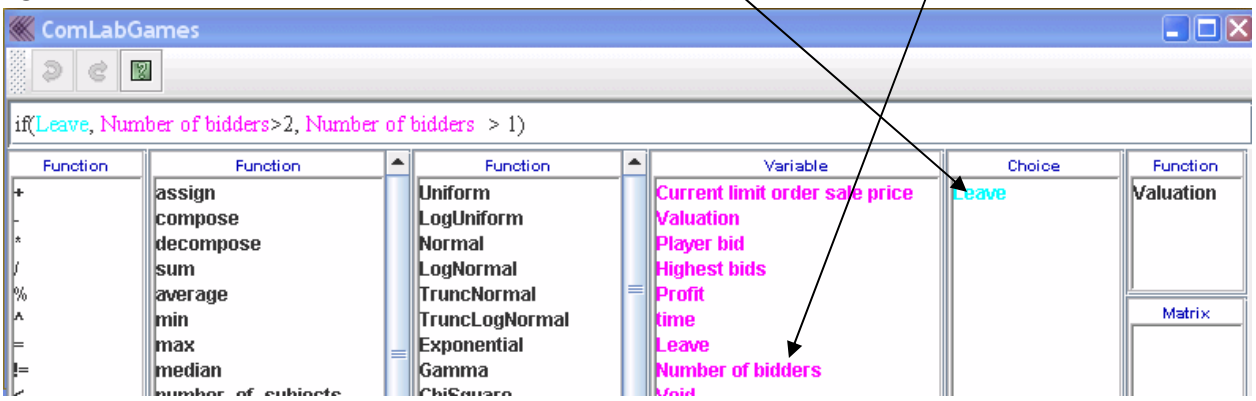
- 

in Japanese auction

- Select Bidding window for the stage transition and write the condition: if(Leave, Number of bidders>2, Number of bidders > 1) in the editor:

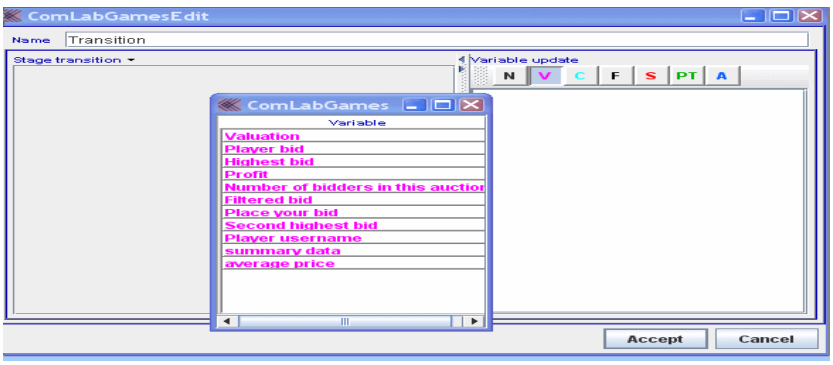


The condition means that the program will return to Bidding window if Leave is a choice created for the Japanese auction, and all choices are in turquoise color. Number of bidders is a variable and it is always in pink color. To write the condition above, right click to for function editor and double click on if(, ,) and on choices and variables.

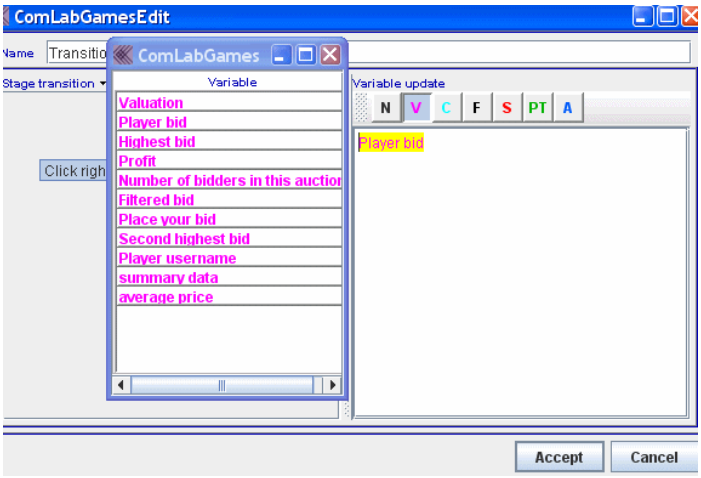


d. General Rule for updating variables in Transition for later use.

- Go to the right part with variable update:
- Click on V and all created variables appear in a separate window like in the example below:



- Double click on the variable that you want to update, and it appears in the transition window highlighted in yellow.



. A colon (:) has to follow the variable if the updated value is not used in the next steps within the same transition but in the follow up stages.

summary data:[Player username,Player bid,Valuation,Profit]

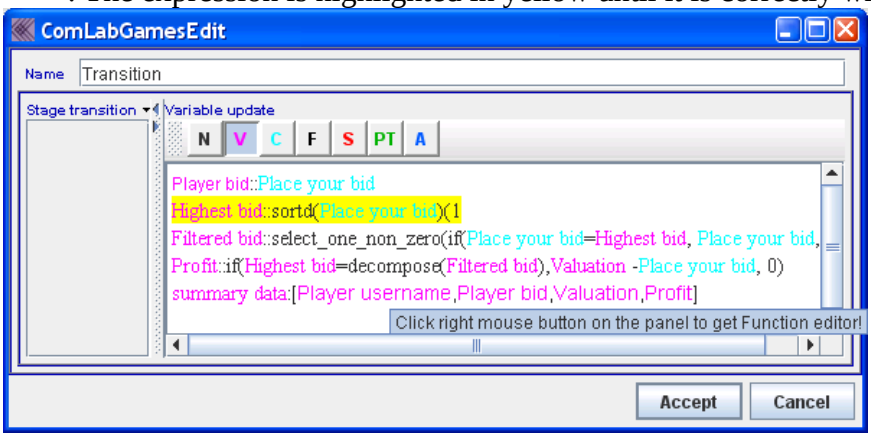
. If double colon (::) follows the is selected variable the updated value will be used within the same transition. To illustrate this example see below:

Highest bid::sortd(Place your bid)(1)

The value of the highest bid is used in the filtered bid, profit.

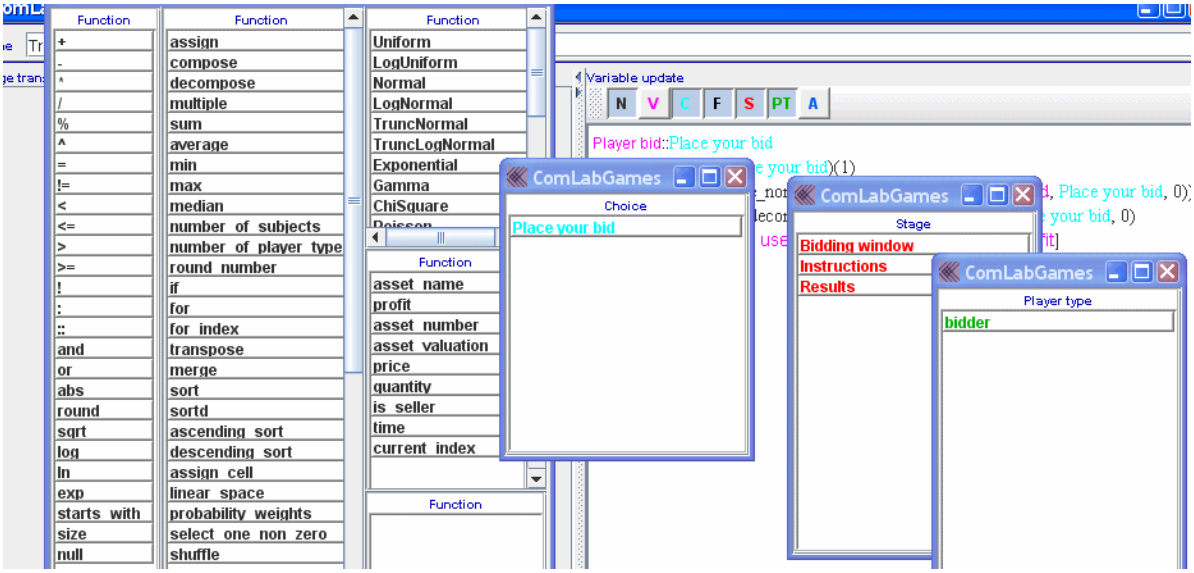
```
Player bid::Place your bid
Highest bid::sortd(Place your bid)(1)
Filtered bid::select_one_non_zero(if(Place your bid=Highest bid, Place your bid, 0))
Profit::if(Highest bid=decompose(Filtered bid),Valuation -Place your bid, 0)
summary data:[Player username,Player bid,Valuation,Profit]
```

. The expression is highlighted in yellow until it is correctly written:

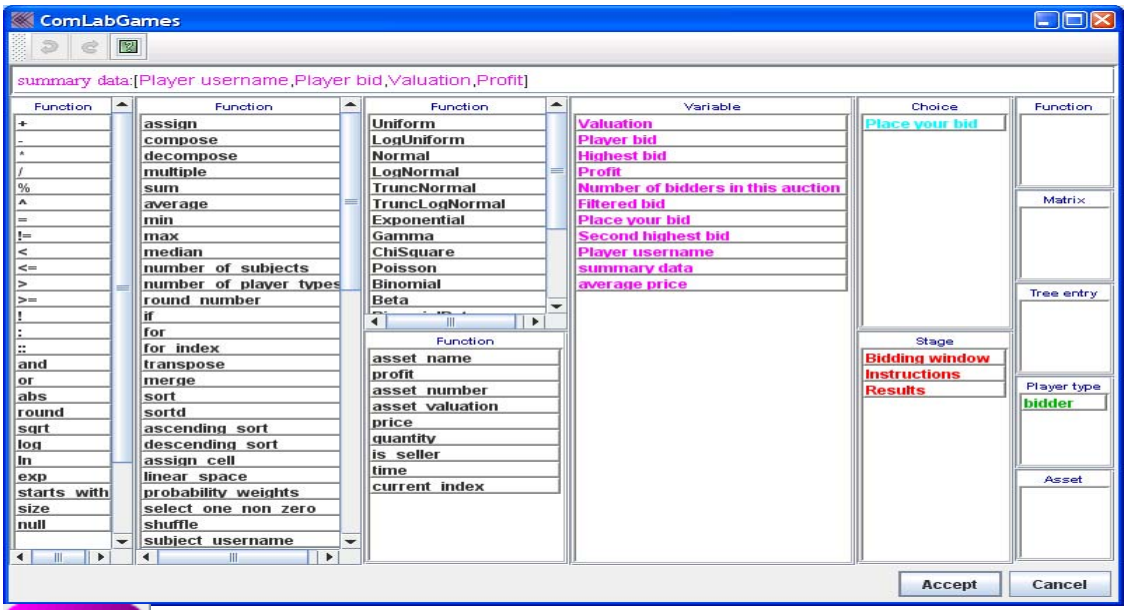


- **Do not click "Return"** until you finish the expression for the updated variable.

- **Click "Return"** to start updating the next variable.
- **To insert choices, functions, stages, player types, assets** into the expression, just click on C , **F**, **S**, **PT** and **A.** Everything is listed in the separate windows, just double click on the appropriate item, and it will be added where the cursor is in the editor.



- Alternative way of writing the expression is to use function editor. (The function editor has all the elements together). In this case **right click** in the transition editor and the function editor appears. Write the expression, click on "Accept" to close and save the expression. It will be saved in the transition editor.



e. Variable **creation (initialization) for <u>first price private value sealed bid auction</u>**

Variable have to be created (i.e. initialized) before you can update them in the Transition. Variable has to have the right dimensions and type in order to perform any mathematical operation. If you do not know dimensions, the program allows you to choose "0" as an option for number of rows and columns. It means that the program will determine the dimensions based on the data and mathematical operations. If you do not know the type of a Variable, you can use "any" (the program selects the correct type of a variable in this case).

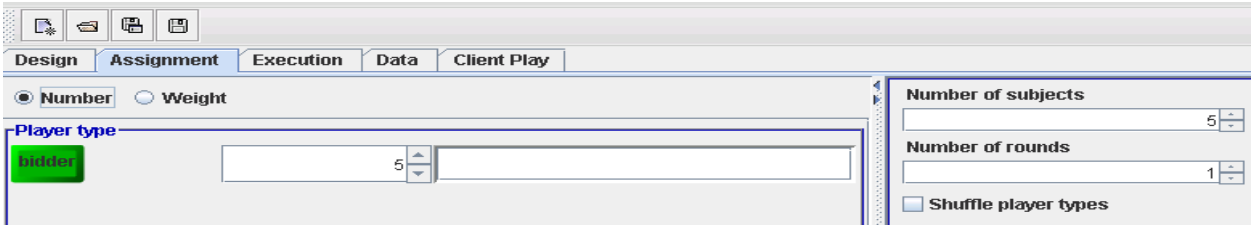
For the first price sealed bid auction we have to create Player bid, Filtered bid, Highest bid, Profit, Player username and summary data before we can update them in .

- Player bid collects all subject's bids and distributes them to each of them individually. It must have the following selection when you open this variable:

Multiple ☑ so that each subject gets to see her own bid. Type real

- Highest bid must be Type real
- Filtered bid should have Type real.

If Multiple ☑ is not used in the variable, then all the choices will be put in a column vector. The dimension of a column vector (i.e. number of rows in a variable) corresponds to the number of subjects in Assignment tab. If you change number of subjects in Assignment, the number of rows will adjust to the new number.

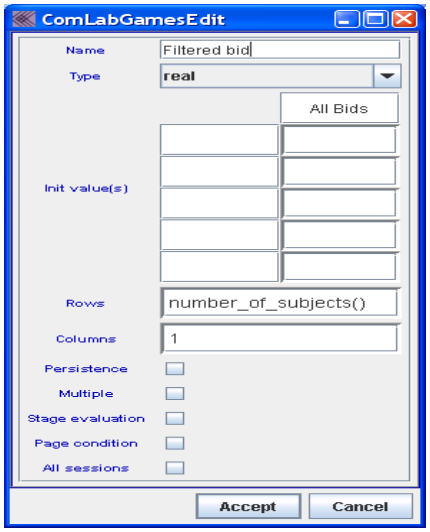


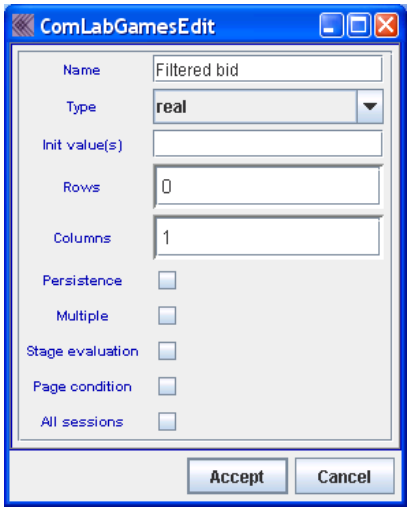
There are two ways to determine the dimension a variable, and there are equivalent:

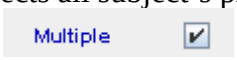
a. Number of rows in Filtered bid should correspond to the number of subjects. Use function number_of_subjects() for Rows number_of_subjects()	b. You can leave the program to determine the number of rows in Filtered bid. Use 0 for rows.

.

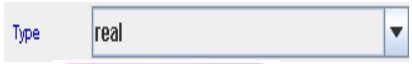
Number_of_subjects() is a function that changes the dimension of a variable if we change the number of subjects in the “Assignment” tab.

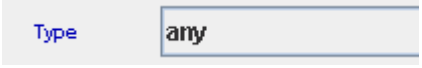




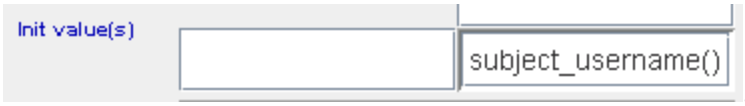
Profit collects all subject’s profits and distributes them to each of them individually. It should have 

so that each subject gets to see her own bid, and



- Player username collects all subject’s usernames individually. Player username should have 

because we are collecting names, and any can be any type. For the initial value write the function subject_username() which takes the usernames and assigns them to the right subject.



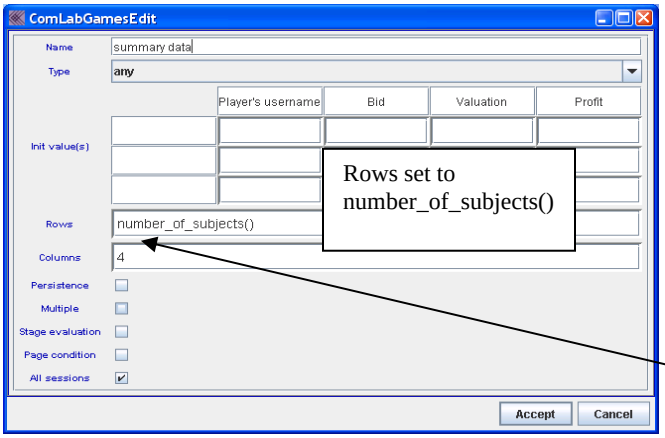
. In order to assign a correct name to each subject, 

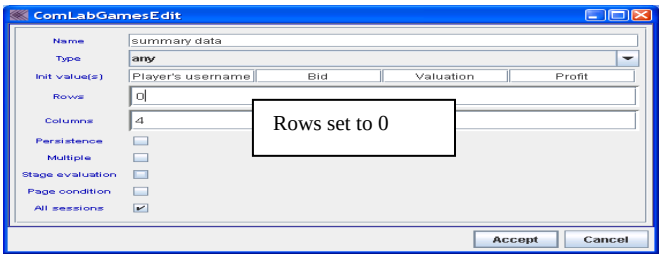
has to be selected.

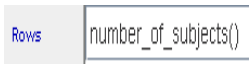
- summary data is a variable used in the Results stage it collects all the data for all the sessions if “immediate summary stage” is not selected in Assignment. It should look like: 

. If a summary stage is selected then each session will see only its own summary. It is up to the moderator to decide what the summary table should

include. Below is the example that which variables we selected to present in the summary presentation.





- Select  because we will be combine different type of data.
- The variables that we want to put in a summary tab consists of **Player username**, **Player bid**, **Valuation** and **Profit**.
- Four columns were created  and labeled to reflect the variables that we want to present 
- Number of rows can either be 0 or 
- Select All sessions to get the summary data for all sessions

f. Variable update in the Transition

```
Player bid::Place your bid
Highest bid::sortd(Place your bid)(1)
Filtered bid::select_one_non_zero(if(Place your bid=Highest bid, Place your bid, 0))
Profit::if(Highest bid=decompose(Filtered bid),Valuation -Place your bid, 0)
summary data:[Player username,Player bid,Valuation,Profit]
```

Use double colon (::) if you need to use the updated variable in the calculations below; otherwise use column(:) so that the previous value is used. However in this case it will be updated for the next stage.

- **Player bid**::**Place your bid**

  This expression updates the **Player bid** with the bids that subjects made. Because double colon is used (::) the updated values are used in the follow up expressions

- **Highest bid**::sortd(**Place your bid**)(1)
  sortd() function sorts all the bids in descending order; (1) means that it takes the first raw in a vector with dimensions nx1. Note: to select an element in fourth row and third column from a

	matrix, write (Place your bid)(4,3) - Filtered bid::select_one_non_zero(if(Place your bid=Highest bid, Place your bid, 0)) If there is tie for the highest bid "select_one_non_zero()' randomly picks one with the condition written inside. - Profit::if(Highest bid=decompose(Filtered bid),Valuation -Place your bid, 0) - Filtered bid is a vector with dimentions nx1 (n stands for number of subjects in a session). Decompose() creates a scalar in order to calculate the profit. - summary data:[Player username,Player bid,Valuation,Profit] In the example above we put four variables in four columns and we created a matrix nx4. Each variable had the same dimension: nx1. **Comma** separating variables creates columns, square brackets:[ ] means we are creating a matrix. (Note Semi-column (;) puts each variable in a new row)

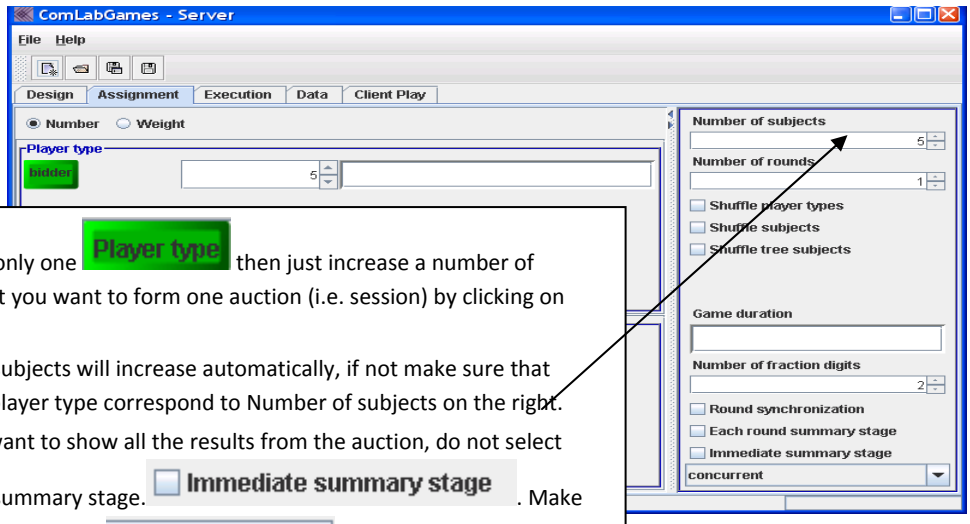
g. Variable update for second price private value sealed bid auction

h. Variable update for Dutch private value sealed bid auction

i. Variable update for English price private value sealed bid auction

j. Variable update for Japanese private value sealed bid auction

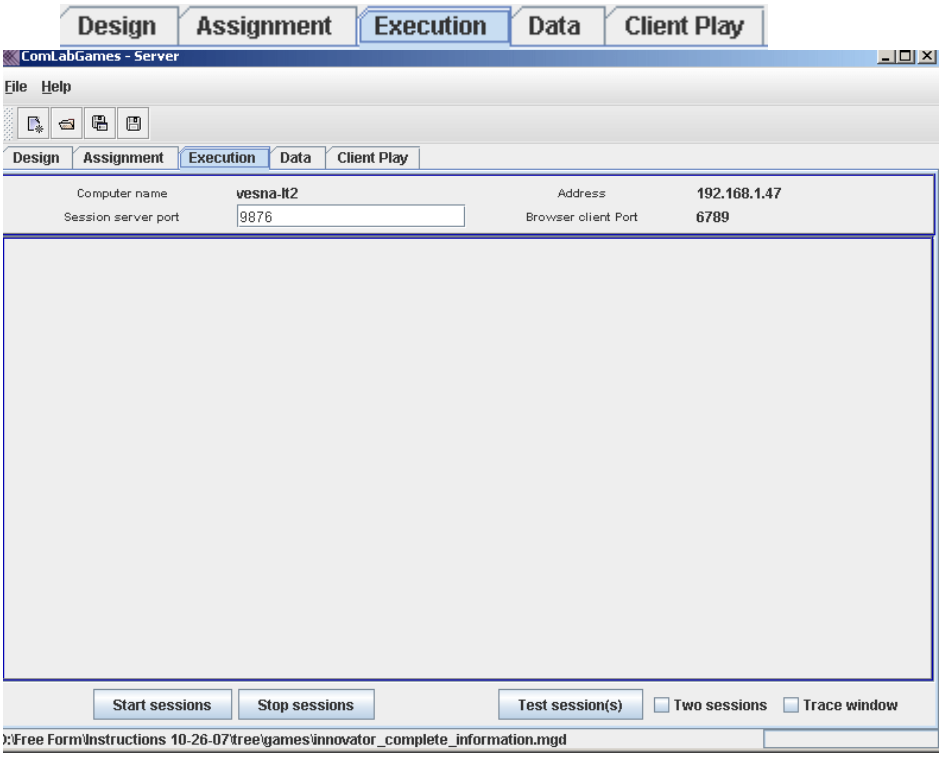
1. Assigning subjects a role in Assignment window



- If you have only one Player type then just increase a number of subjects that you want to form one auction (i.e. session) by clicking on arrow.
- Number of subjects will increase automatically, if not make sure that number of player type correspond to Number of subjects on the right.
- When you want to show all the results from the auction, do not select Immediate summary stage. Immediate summary stage. Make sure to stop the game (Stop sessions on Execution when you use this option in order to see the results for all sessions.
- In your experiment, you might want to select ☑ it so only subjects for their session see the data.

2. Conducting an experiment

a. Select Execution on

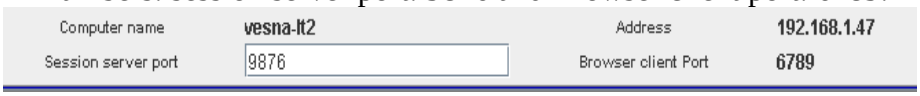


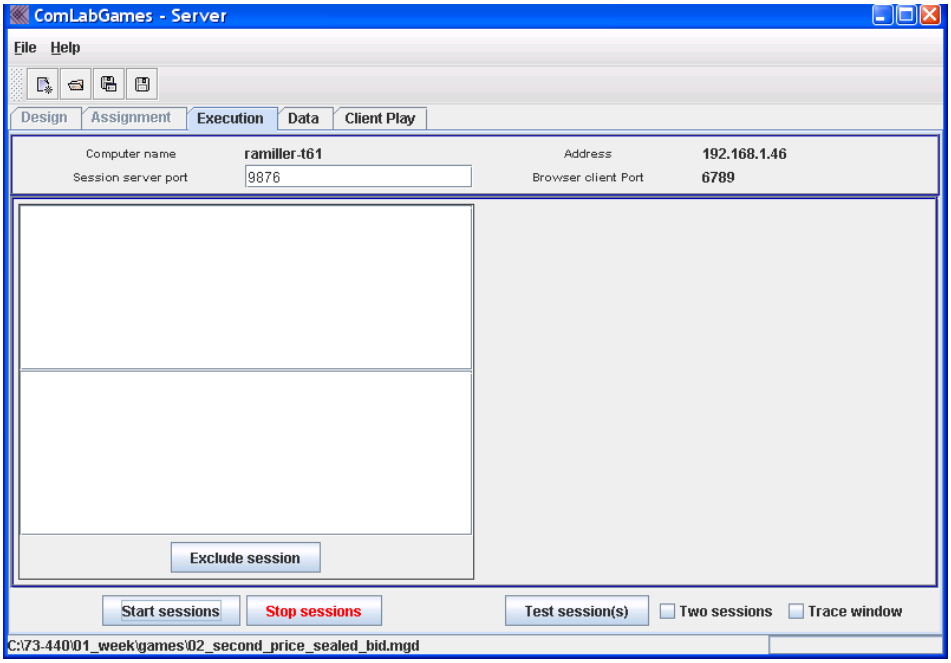
b. Make sure the game is loaded on the design window (the name of the game is shown at the bottom of the window or just click on Design to see if the game is loaded)

c. To test the game before conducting an experiment, click on Test session(s) that can be found at the bottom right end of the wind**ow.** This option allows the moderator to see how the client window looks like and to go thorough the experiment on the same screen as the program. Selecting ☑ Two sessions tests two sessions running at the same time. ☑ Trace window shows all the calculations during the test. Trace uses a lot of CPU time so it might be slow, use only one test session and limit the number of subjects in the Assignment window.

d. To run an experiment, click on 

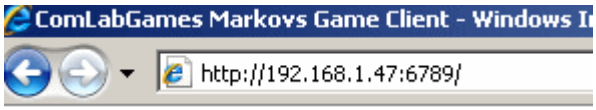
and give the address of the server to the subjects. The address consists of two parts: the address and the port number. The address is located in the top right corner of the window. In our example the address is: 192.168.1.47. There are two port numbers: session server port: 9876 and Browser client port: 6789.



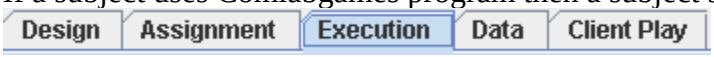


e. Provide the Url address to the subjects:

- If a subject uses a browser the address written in the browser should be as follow: http://192.168.1.47:6789 (do not omit http://)

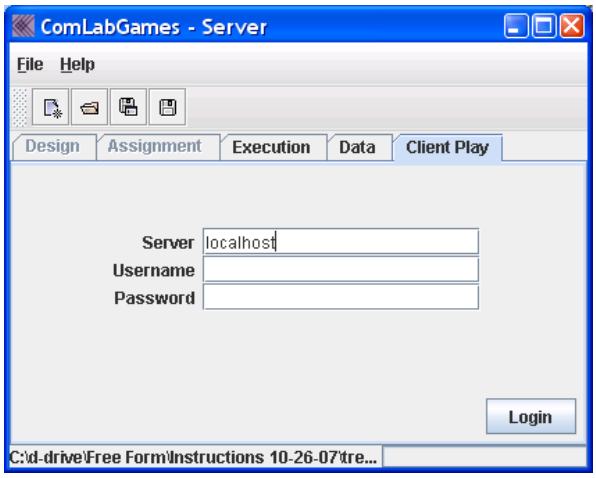


- If a subject uses Comlabgames program then a subject should select 

on 

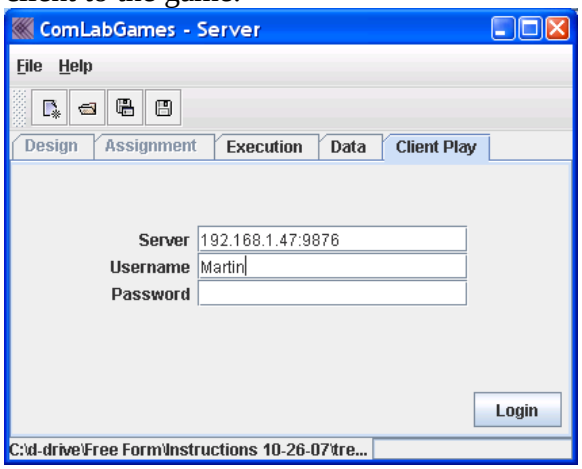
.

The following window will appear:



- Provide Url address, colon and session server port for users of Client Play:
  192.168.1.47:9876 (Note session server port is: 9876)

- A subject writes a Url address under server, login name that can be any name. Password is not necessary. Clicking on Login connects the client to the game.

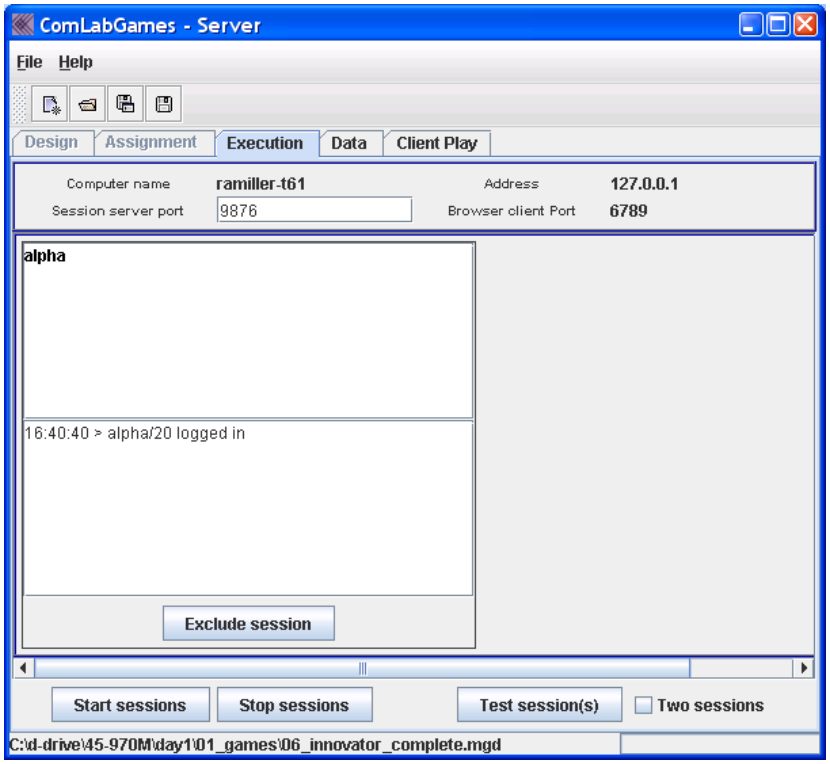


3. Moderator Viewing the Data

- On the execution window the moderator can follow the status of each subject:

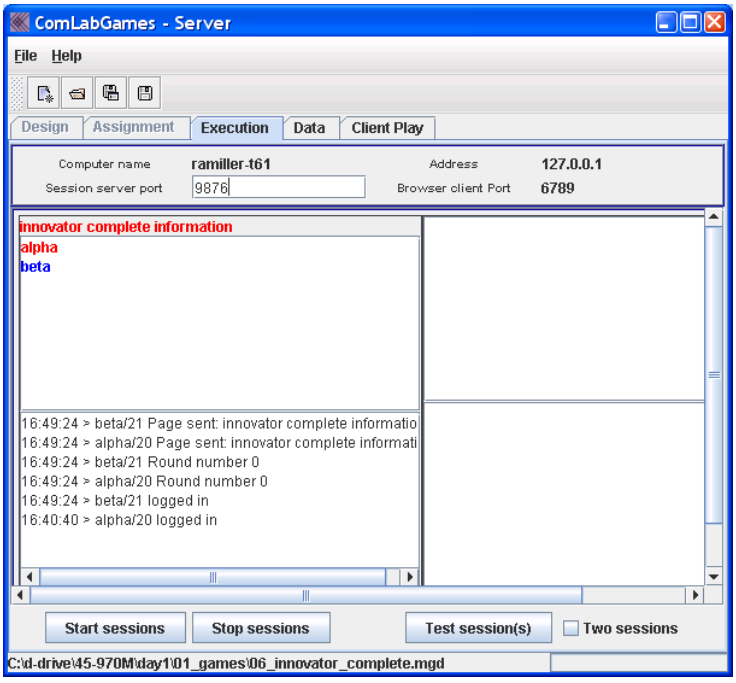
(1) A Subject connected to the game and not all the subjects for the session connected to the game (Login names are in black).

Subject “alpha” connected to the game.

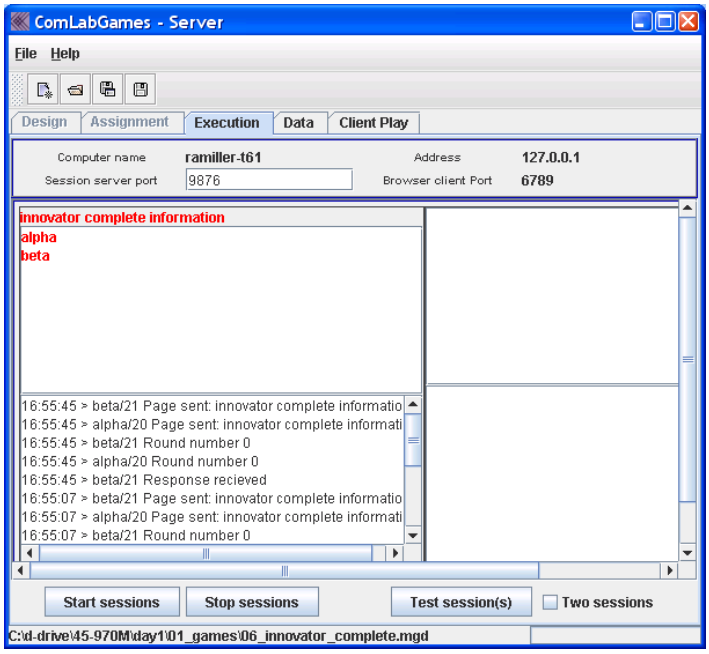


(2) All subjects are connected to the game but nobody made a decision yet

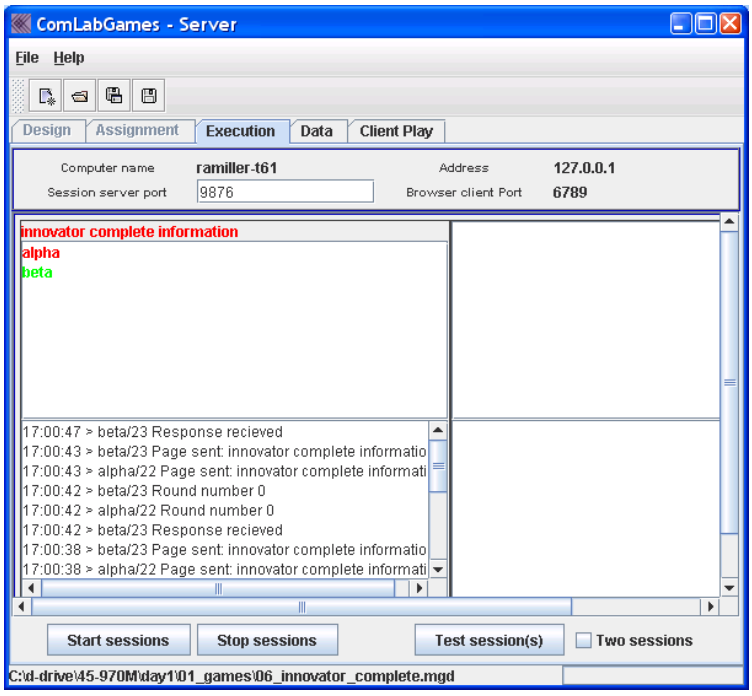
Subject in red is the one who has to make a decision and a subject in blue has to wait until the first subject selected a choice.



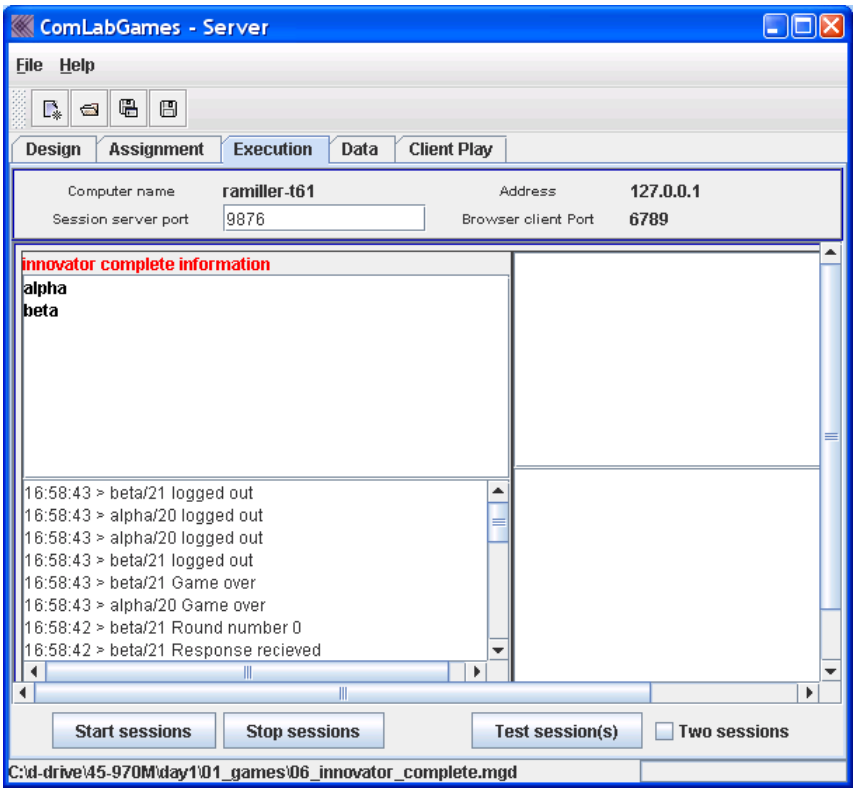
(3) All subjects have to make a decision (all login names for the session are in red)



(4) Subjects who finished the tasks are in green. Subjects who did not finish the task are in red.



(5) The game is over. All subjects finished all the tasks.

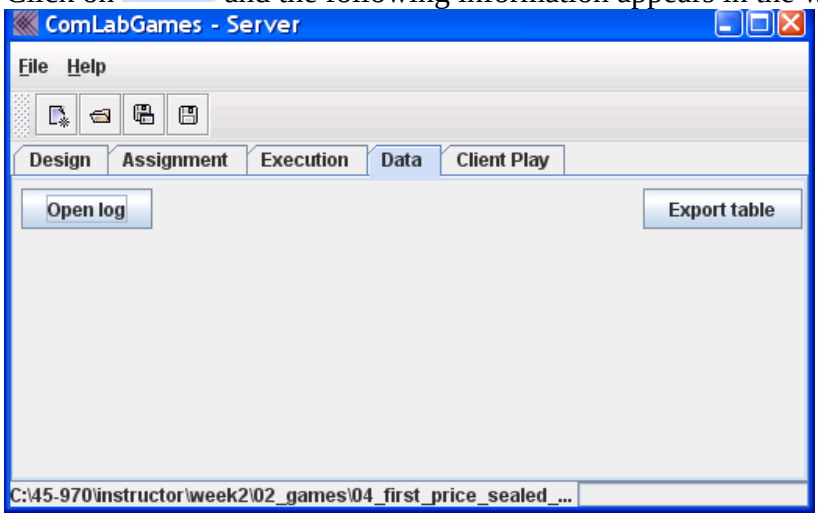


- If any of the sessions have technical problems, moderator can disconnect them by clicking on Exclude session below the session that has a problem.

g. Viewing the Data after the experiment (i.e. after clicking on Stop sessions)

a. Moderator can show the data immediately after the experiment

b. Click on Data and the following information appears in the window:



c. Click on 

and select the appropriate data file. Data file have start with “log-date-time-name-of-the-experiment”.