INTRODUCTION TO Z-TREE

Antonio Alonso Arechar
Yale University

SCHEDULE

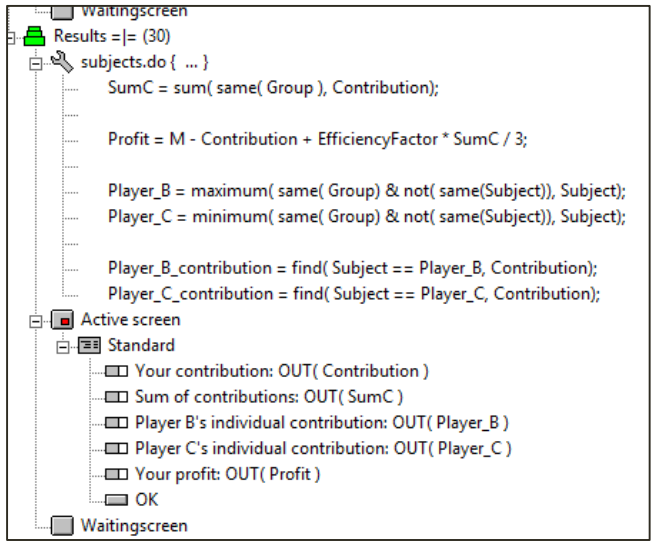
- Interactive games
- Sequential games
- **Advanced Topics**
- Frontiers

ADVANCED TOPICS

1. Contracts table
2. Chat boxes +
3. External programs

CONTRACTS TABLE

- In our PGG we calculated and displayed the sum of contributions
- What if we wish to show **individual contributions**?
- In the **Subjects table** we could sketch some rules to find group members

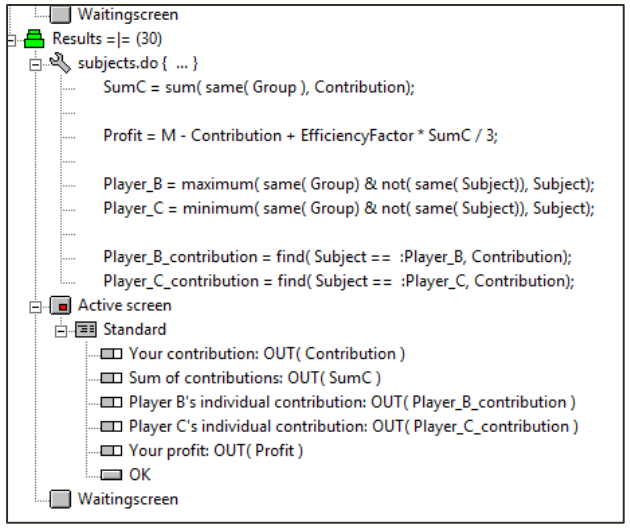


Subject	Group	Profit	TotalProfit	Participate	Contributi	TimeOKDe	SumC	Player_B	Player_C	Player_B_	Player_C_	T
1	1	13	13	1	1	9	6	3	2	0	0	
2	1	12	12	1	2	13	6	3	1	0	0	-
3	1	11	11	1	3	16	6	2	1	0	0	-
4	2	16	16	1	4	20	15	6	5	0	0	-
5	2	15	15	1	5	23	15	6	4	0	0	-
6	2	14	14	1	6	26	15	5	4	0	0	-

CONTRACTS TABLE

- How to find subjects' contributions using *current* records?
- **The scope operator (:)!**

A	B	C = sum(A * B);	D = sum(:A * B);	E = sum(:A * :B);
2	5	10 + 48 + 56 = 114	10 + 24 + 14= 48	10 + 10 + 10 = 30
4	12	10 + 48 + 56 = 114	20 + 48 + 28 = 96	48 + 48 + 48 = 144
8	7	10 + 48 + 56 = 114	40 + 96 + 56 = 192	56 + 56 + 56 = 168



Subject	Group	Profit	TotalProfit	Participate	Contributi	TimeOKDe	SumC	Player_B	Player_C	Player_B_	Player_C_
1	1	13	13	1	1	22	6	3	2	3	2
2	1	12	12	1	2	18	6	3	1	3	1
3	1	11	11	1	3	12	6	2	1	2	1
4	2	16	16	1	4	7	15	6	5	6	5
5	2	15	15	1	5	2	15	6	4	6	4
6	2	14	14	1	6	-3	15	5	4	5	4

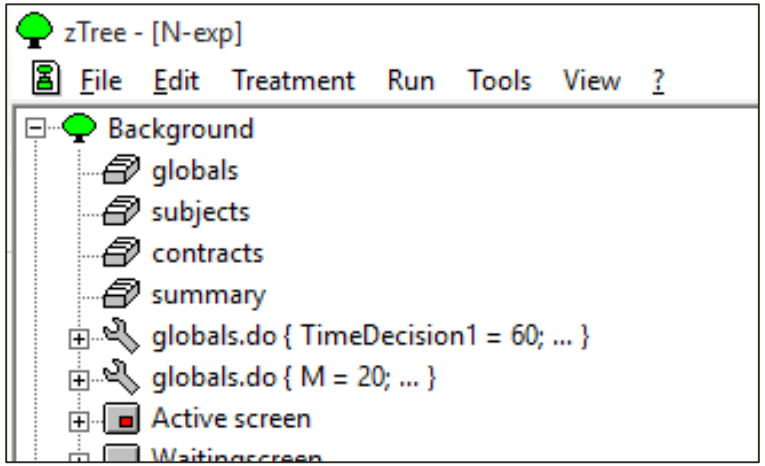
CONTRACTS TABLE

- The scope operator is so useful z-Tree set a default we've already seen!
- `SumC = sum( same( Group), Contribution);`
- `SumC = sum(Group == : Group), Contribution);`

"(My) `SumC` *is the sum over the contributions of those whose* `Group` *is equal to mine"*

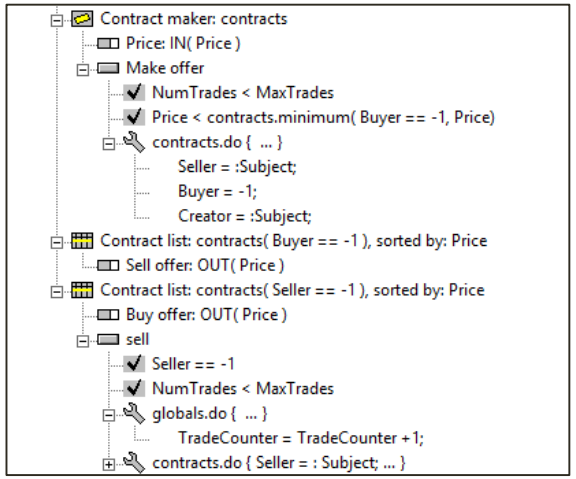
CONTRACTS TABLE

- It is hard to handle others' records in the **Subjects table** as group size increases
- **Contracts table** can contain an indefinite number of rows
- Holds default variable `Period` only



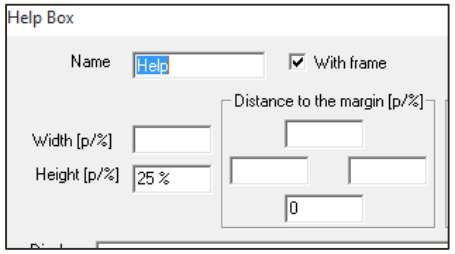
CONTRACTS TABLE

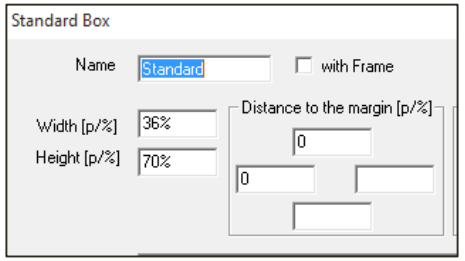
- Records in the **Contracts table** can be created, selected, viewed and edited in boxes:
  - Contract creation box
  - Contract list box
  - Contract grid box
- By using several boxes, it is possible to do different things in one screen – in one stage.

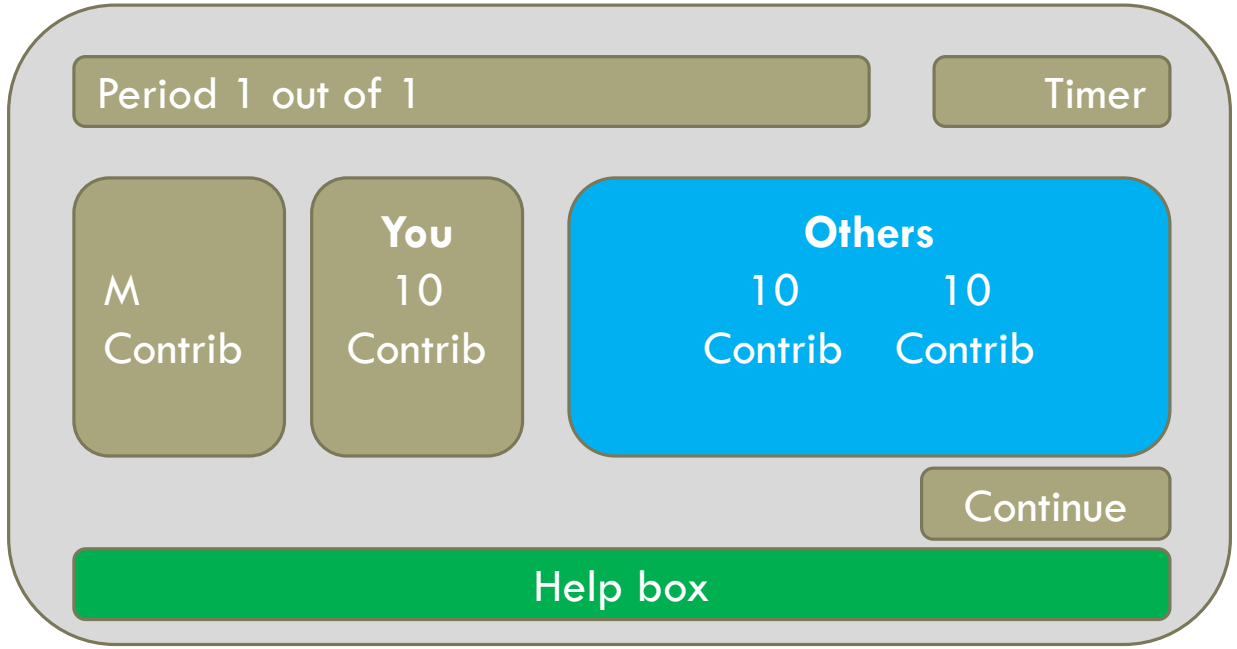


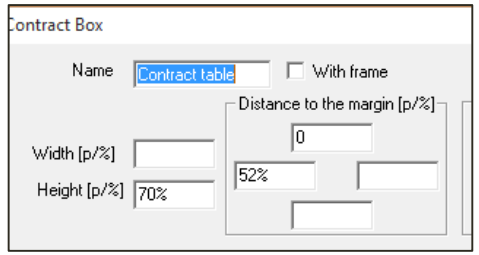
CONTRACTS TABLE

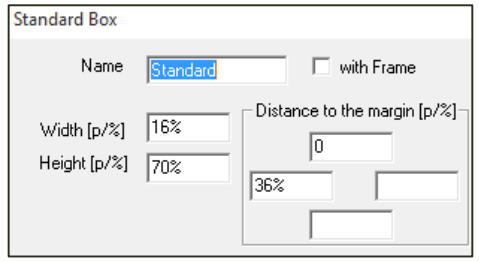
- Here we will display individual records in a new stage called *Individual Contributions*
- We need to plan the active screen's layout
- **Treatment → New Box → Contact Grid Box** (sic)

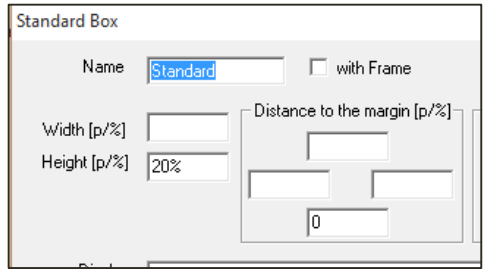






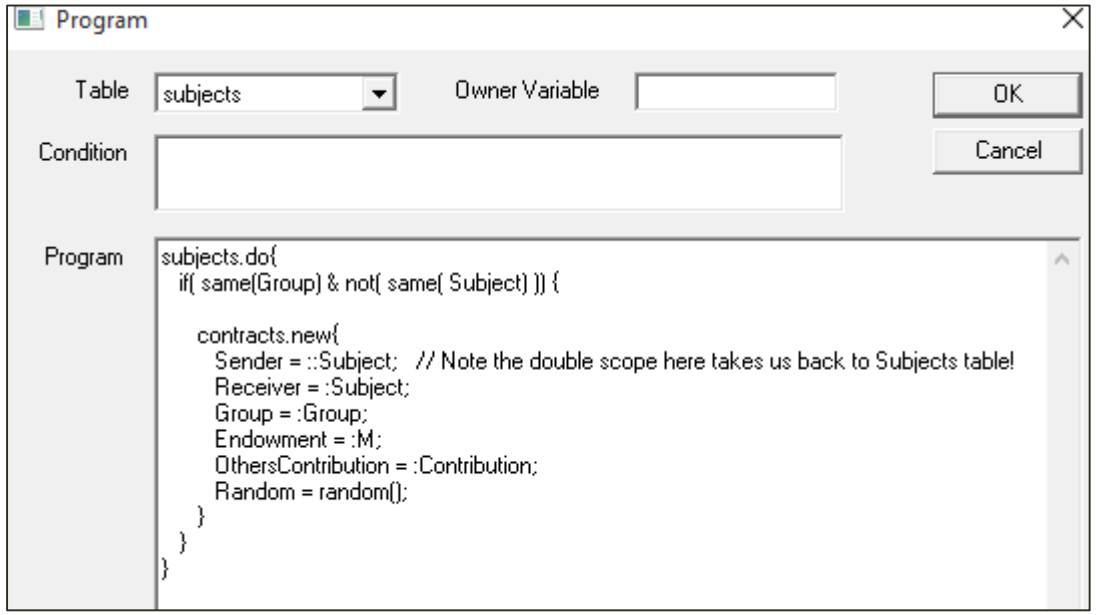






CONTRACTS TABLE

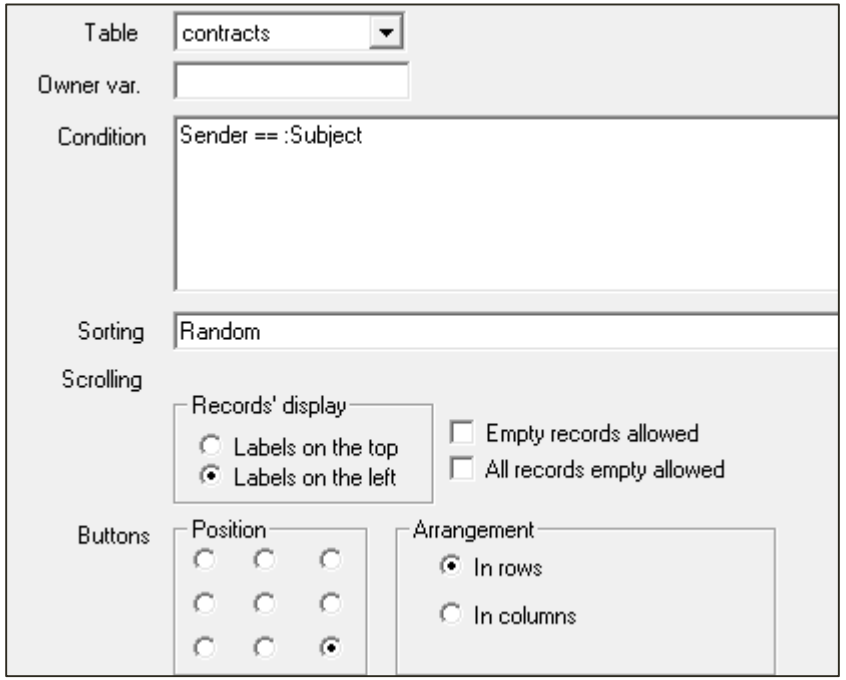
- Now that we have the skeleton of our active screen we need to populate it
- Input of the Standard boxes is easy and already defined for the subject (M, C)
- For the contracts box we need a program:



contracts	Period	Sender	Receiver	Group	Endowme	OthersCor	Random
contracts	1	1	2	1	10	3	0.415471
contracts	1	1	3	1	10	2	0.482208
contracts	1	2	1	1	10	1	0.609844
contracts	1	2	3	1	10	2	0.977137
contracts	1	3	1	1	10	1	0.3239
contracts	1	3	2	1	10	3	0.987702

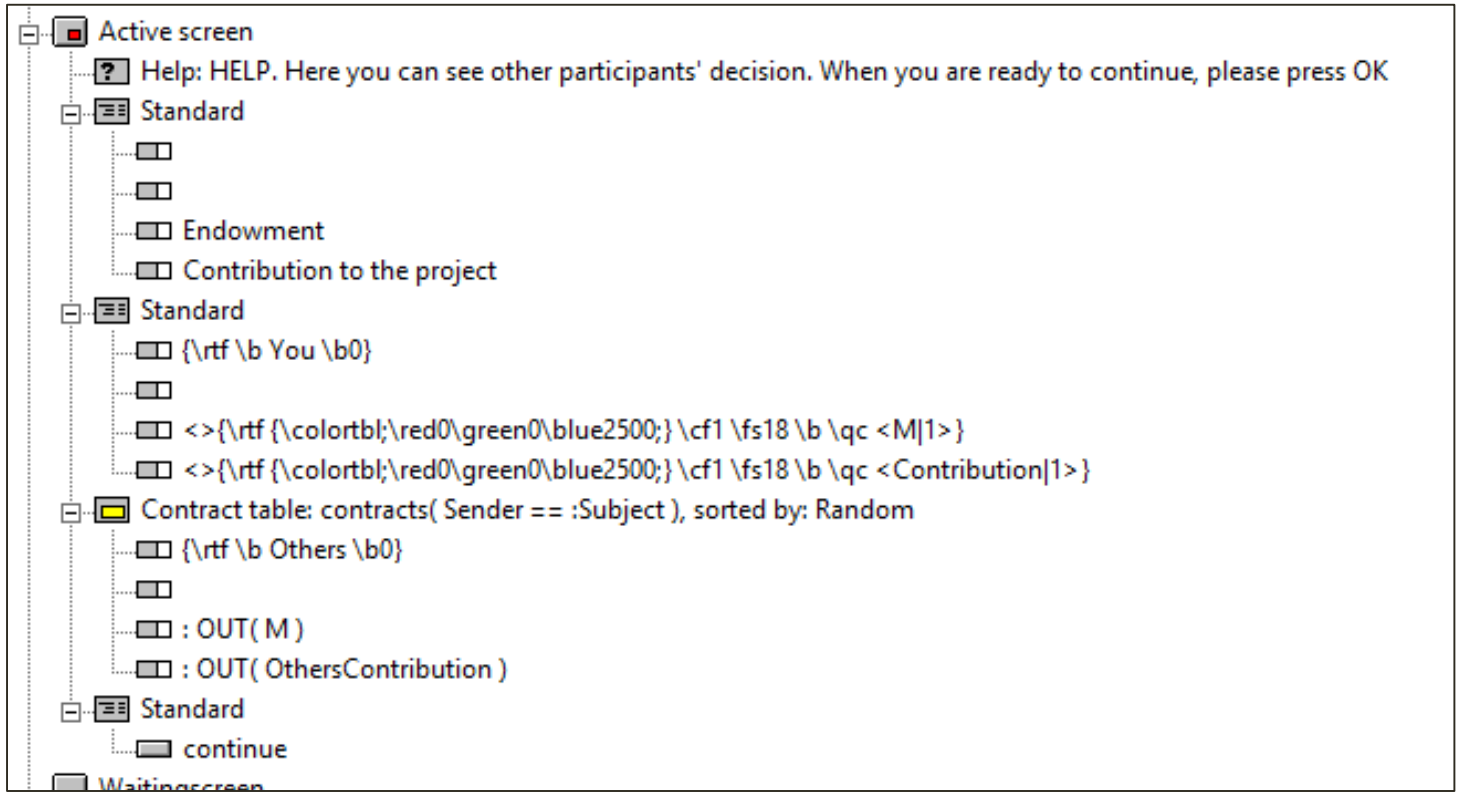
CONTRACTS TABLE

- Note that the Contract Box has specific features:
  - Condition: Defines what records to show (if left blank will produce all records)
  - Sorting: Sorts the variables introduced in the requested order



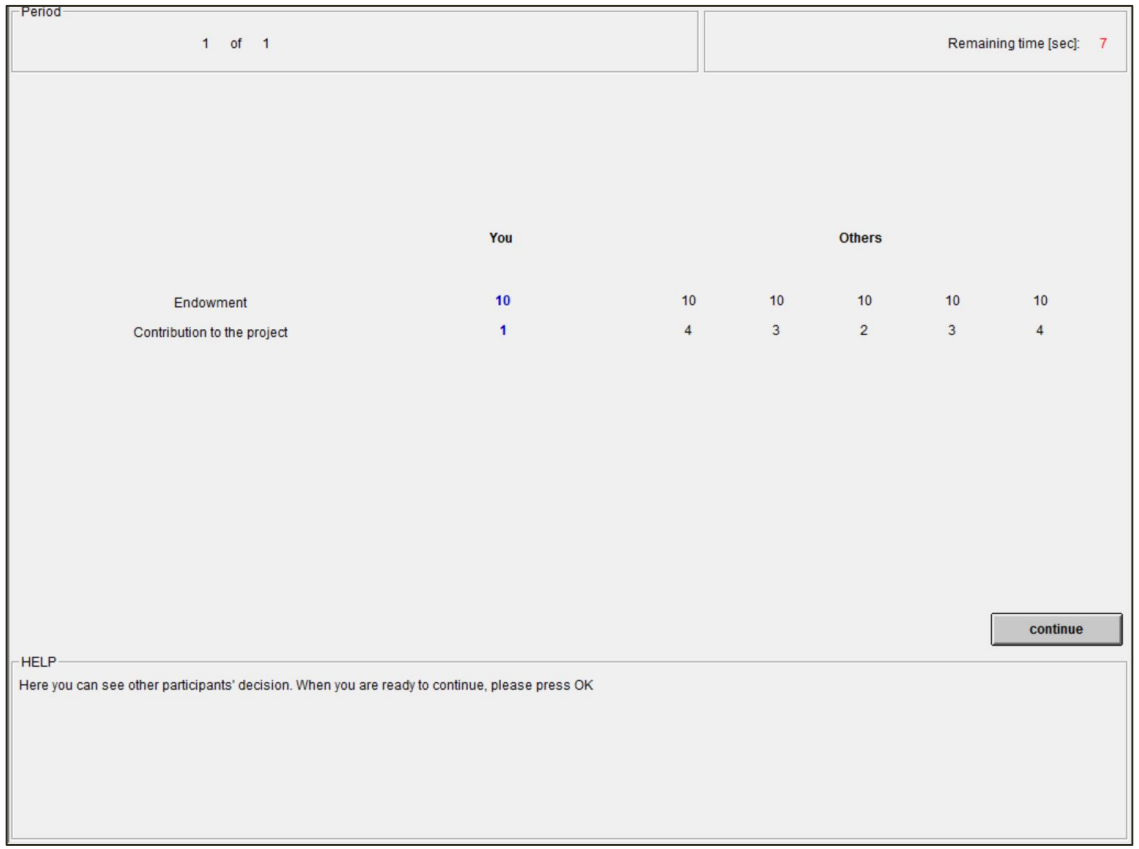
CONTRACTS TABLE

- That's it! We can now start filling in the boxes:



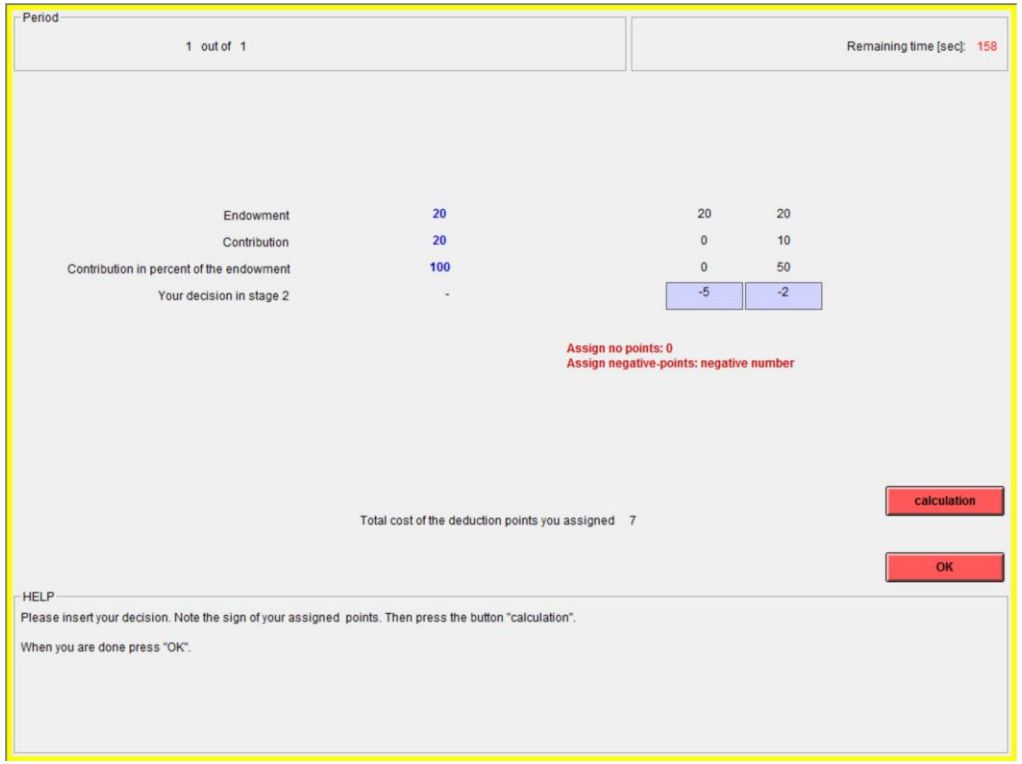
CONTRACTS TABLE

- Note that the contracts table also allow us to scale up our group sizes:



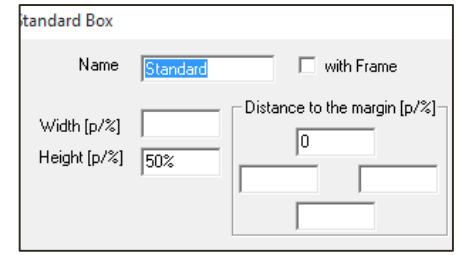
CONTRACTS TABLE

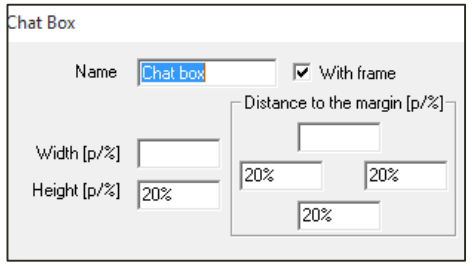
- Or include a stage with individual punishment / rewards:

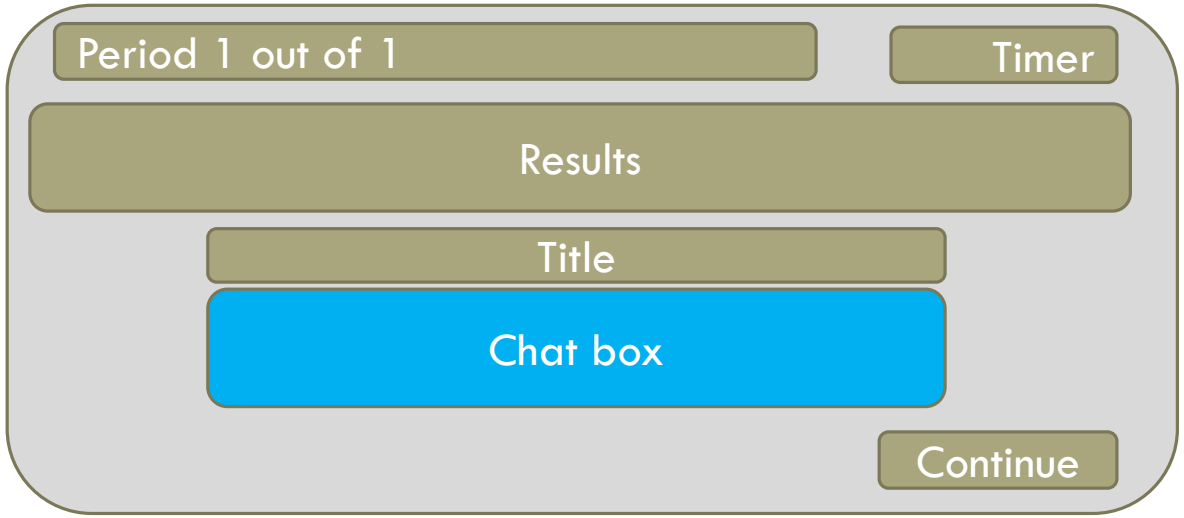


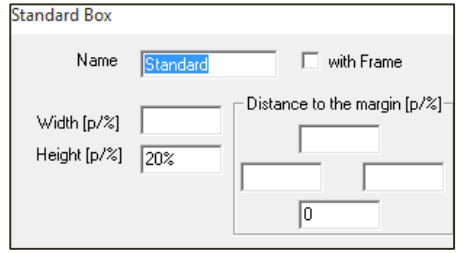
CHAT BOXES

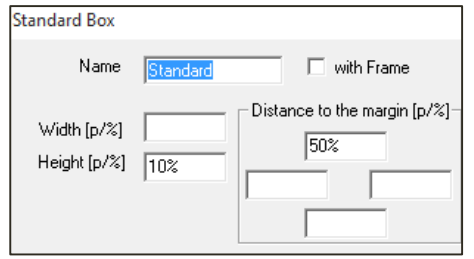
- The **chat box** allows controlled communication between subjects
- Subjects see a text entry field, and when they press ⏎, the text is stored in contracts
- Suppose we wish to see if *post-decision* communication affects cooperation in the PD
- Open **PD.ztt** and add a new Chat Box on the active screen of Results





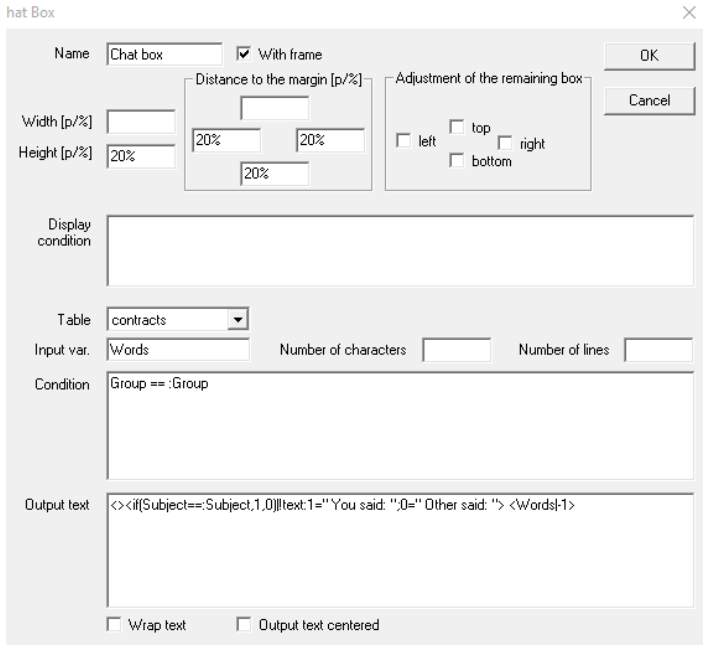






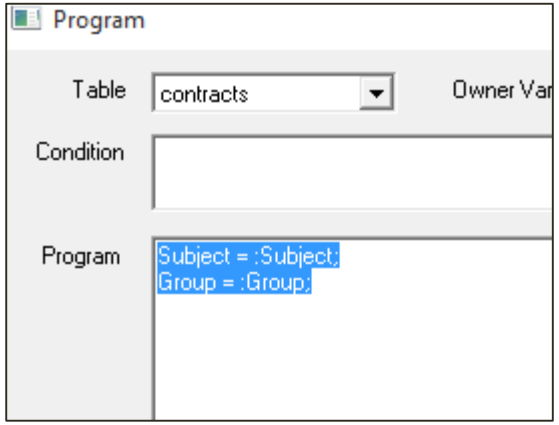
CHAT BOXES

- Chat boxes have not only display conditions but also **Condition**
- In Condition we specify what output from Contracts we want to be displayed
- Chat boxes work as contracts boxes so we can show *current* input as output
- Remember to fill in the other boxes!



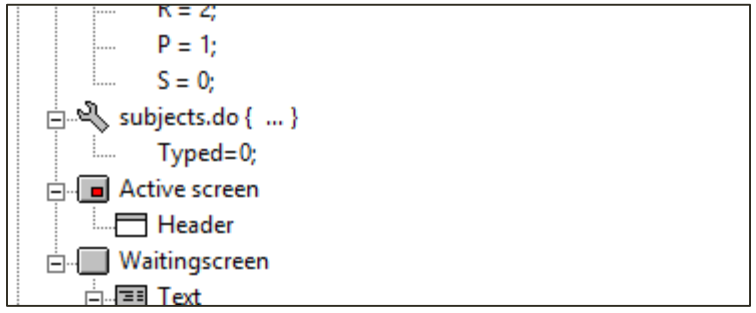
CHAT BOXES

- Note that our chat box is storing records of *Words* but not of the people sending them
- Once participants are ready open the xls file we are editing
- Create a program for the Chat Box that stores those variables!



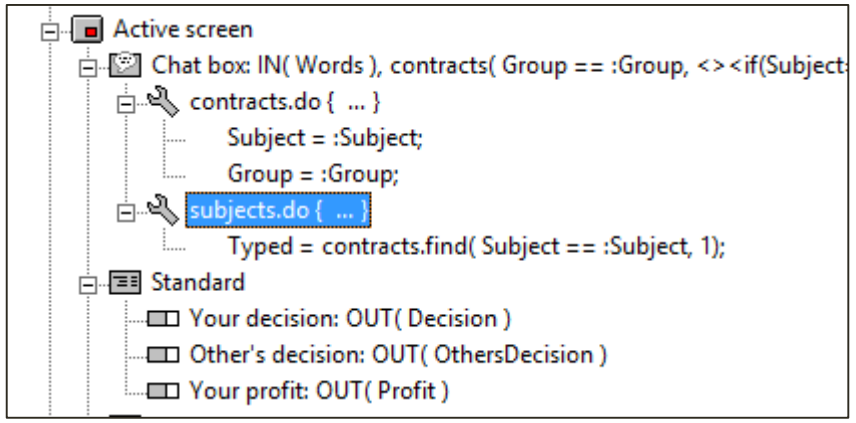
CHAT BOXES

- What if we want to **force** participants to enter some text before leaving the stage?
- We could do so by having some way to check that a participant typed something
- **Checkers** allow you to do so!
- First let's *initialize* a subjects variable called Typed in the Background
- Run a period of this treatment and open **Run → Subjects Table**



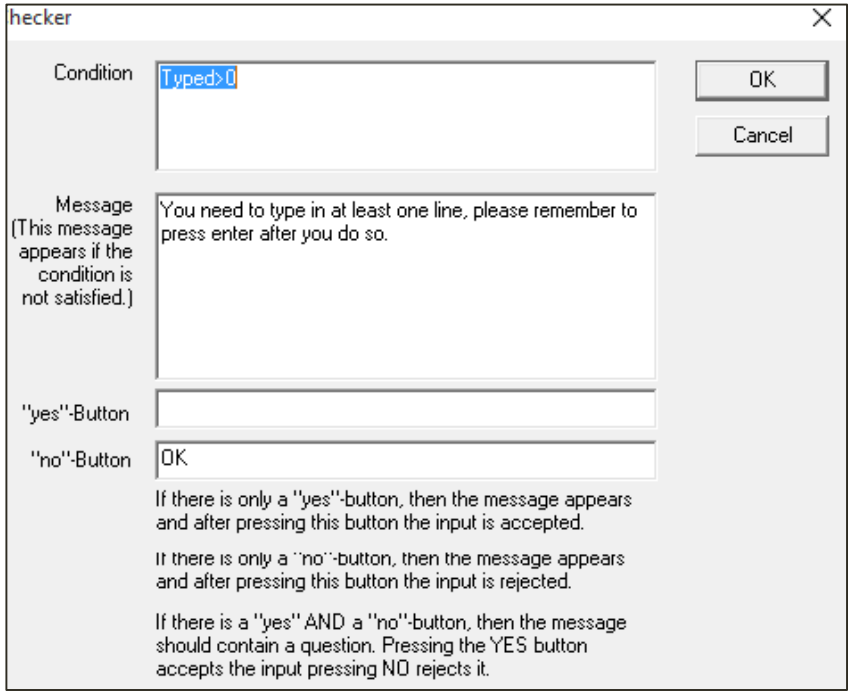
CHAT BOXES

- Now we need to *activate* Typed, how?
- Recall that records for the chat are only created when something is typed
- Let's create a trigger program once the event happens
- Run a period of this treatment and open **Run → Subjects Table & Contracts Table**



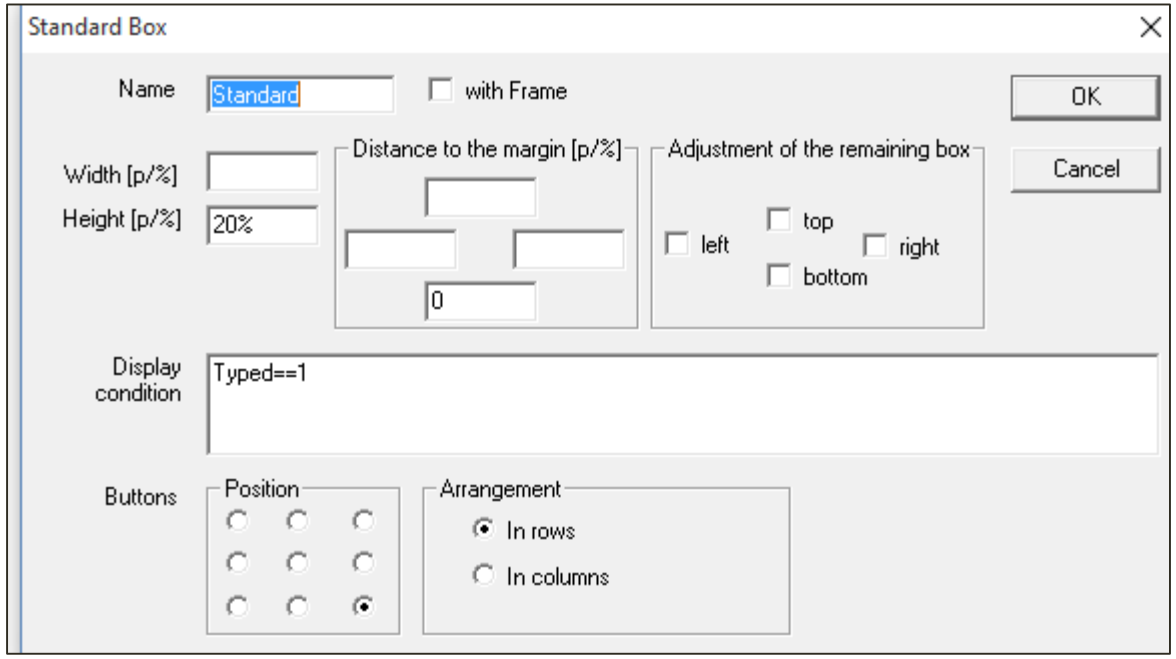
CHAT BOXES

- A natural place to put a checker is a button
- Select the OK button and press **Treatment → New Checker**



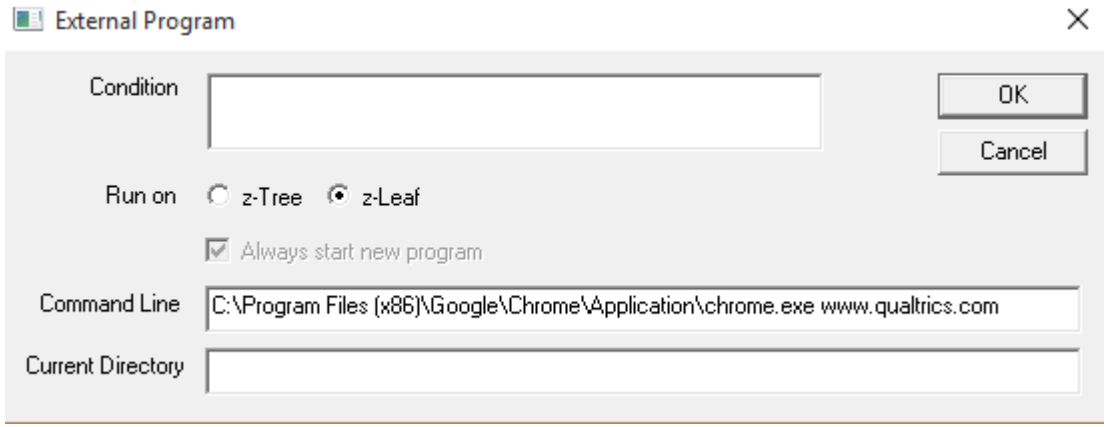
CHAT BOXES

- A *nicer* alternative would be to show the OK button only if `Typed>0`
- Note that a similar approach allows you to conditionally display anything!



EXTERNAL PROGRAMS

- Sometimes you might want participants to go to Qualtrics instead of a questionnaire
- z-Tree allows you to open external programs
- Create a final screen (Participate?) and select **Treatment → New External Program**
- As participants work on the survey you can run an empty Address form!



EXTERNAL PROGRAMS

- How to link participants' choices and their demographics though?
- One solution is to ask participants to enter their computer id, but errors happen!
- Remember that Qualtrics records IP addresses
- You can get each of the computers' ip addresses in the lab by googling "my ip"