zTree

Zurich Toolbox for Readymade Economic Experiments

Filip Vesely
University of Wisconsin - Milwaukee

Print Slides: 1,3,4,6,16,29,36,39-42,44-47,49-60,62-66,68,70,72,74,76-83,85-87,89-94,101-103,107,109,113,115,117,119-124,126,130-134,136,138,140,142,144,146-148,150,152,154-156,158,160-163,165,167,169-171,174-178,180-181,183-187,190-193,199,200,202,203,206-221,223-311

What is zTree for?

Zurich Toolbox for Readymade Economic Experiments

Designed for simple experimenter that does not have:

- competent programmers that can write ad hock programs
- time to design ad hock programs
- ability to specify full experiment without pilots, modifications, …

Designed for simple games, that do not require:

- detailed timing (time is measured in seconds)
- major graphical input/output
- unexpected changes in design once the experiment is in progress

How to get zTree?

z-Tree by urs fischbacher

About z-Tree
How to get z-Tree ?
Compatibility
Docu and Support
z-Tree Wiki
Tips & Tricks
Treatments
Papers
News & Versions
Known Bugs
Links

Licence

The program can be licensed free of charge.
When you report results of experiments conducted with z-Tree, the licence requires that you mention it's use in the publication and cite the article forthcoming in Experimental Economics.

The correct citation is:

The experiment was programmed and conducted with the software z-Tree (Fischbacher 2007).
Reference:
Urs Fischbacher (2007): z-Tree: Zurich Toolbox for Ready-made Economic Experiments, Experimental Economics 10(2), 171-178.

How to order z-Tree?

To get a licence, download the English license contract or the German license contract, print two copies, sign them, fill in postal and email address , and send them to Sally Gschwend, Institute for Empirical Research in Economics, University of Zurich, Bluemlisalpstrasse 10, CH-8006 Zurich. (I need the postal address to send you back one copy of the contract and **I need an e-mail address, to send you login and password for downloading z-Tree.** .) You will then receive login and password for downloading z-Tree. If you do not receive the information by email within reasonable time, please send an email to ztreeadm [at] iew.uzh.ch.

Last modified: September 11, 2007 Urs Fischbacher (fiba@iew.unizh.ch)

Institute for Empirical Research in Economics:
http://www.iew.uzh.ch/ztree/howtoget.php

How does it work?

Manuals:

- 2.1 Tutorial Manual (2001): http://www.iew.uzh.ch/ztree/ztree21tutorial.pdf
- 2.1 References (2006): http://www.iew.uzh.ch/ztree/ztree21ref.pdf
- 3.x Wiki: https://www.uzh.ch/iew/ztree/ssl-dir/wiki/

Two programs:

zTree

- programming editor
- experiment server

zLeaf

- client program for subjects' computers
- client program for the experimenter's input during the game

zTree step-by-step:

Step 1: Install “toolbox” programs.
zTree and zLeaf (for testing)
for you. zLeafs for subjets.

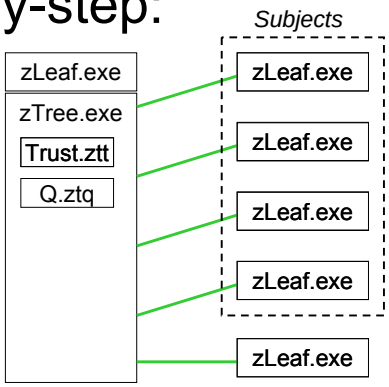
Step 2: Open zTree,

Step 3: Download, create, save your
- program (e.g.Trust.ztt)
- questionnaire (e.g.Q.ztq)

Step 4: Open zLeaf on each subjet’s
computer (or on your computer
if only testing.) + One for manual changes of parameters if needed.

Step 5: Seat the subjects, go through instructions (if not part of the program)

Step 6: Start your program (Trust.ztt)



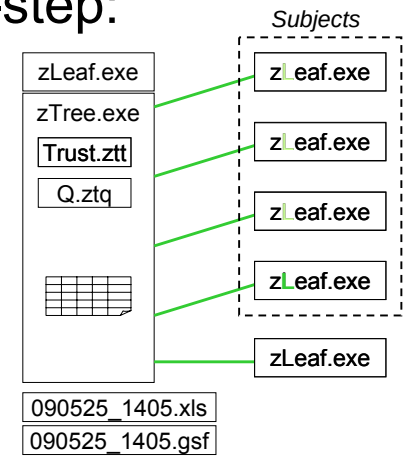
zTree step-by-step:

Step 7: Subjects make decisions

decisions travel to server
program records
decisions in tables
some information is
being passed to subjects

You can observe the tables
You can use your zLeaf to
change parameters

Computer records the tables in
*.xls file every period
*.gsf file continuously
name is given time (Trust.ztt)
program started.



Step 8: Experiment ends. (Most tables in zTree disappear.)
You can run another experiment or questionnaire.

zTree step-by-step:

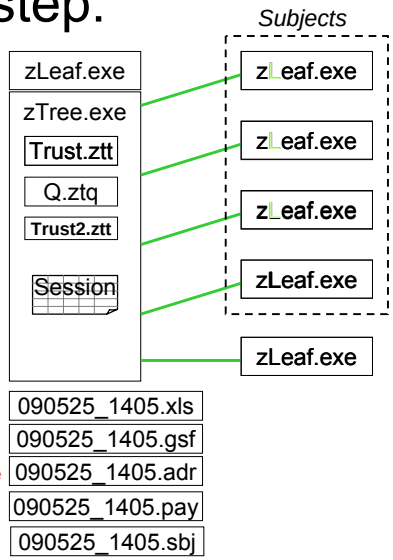
Step 9: If you run another
experiment, data
continue are being
collected into same files

(=>You cannot change
# of participants unless
all programs restart)

Step 10: If you run a questionnaire
a file with answers and
a file with names & payments
are created.

Step 11: use *.xls, *.pay or “session
table” to pay subjects **so they leave**

Step 12: Close zLeafs (ALT+F4)
Close zTree **(!!!Make sure all zLeafs**
Keep *xls, *adr, *pay files **completed their tasks so Trust.ztt is not running !!!)**

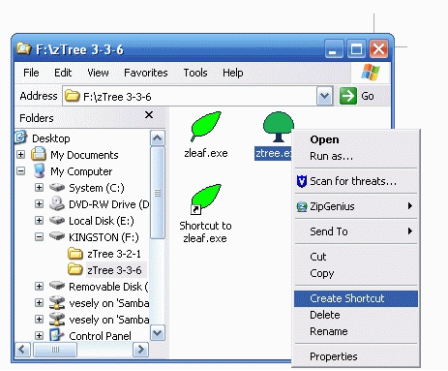


Step 1:

Install “toolbox” programs.
zTree and zLeaf (for testing)
for you. zLeafs for subjets.

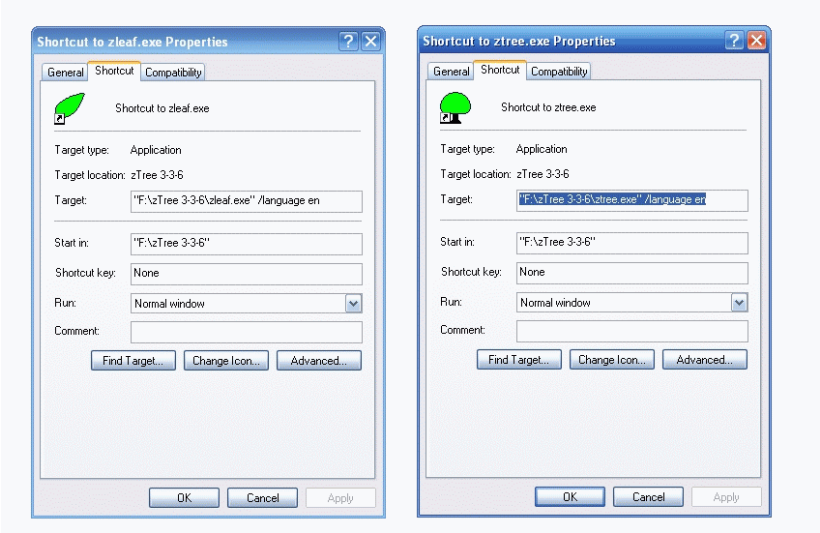
- http://www.go.to/econ/zTree
- Since there will be many files added each time you open zTree, create a new directory for it.
- Copy zTree and zLeaf there.
- Create a shortcut for both.
- Go to properties to change language to english =en (chinese = cn)

by adding “/language en”



Step 1

- Go to properties to change language to english =en (chinese = cn) by adding "/language en"
- if you use multiple zLeaf for testing, create multiple shortcuts with "/name A", "/name B" and so on



Step 1

- Go to properties to change language to english =en (chinese = cn) by adding "/language en"
- if you use multiple zLeaf for testing, create multiple shortcuts with "/name A", "/name B" and so on
- In a big network or if you ran multiple zTrees on same network simultaneously add a server's IP address to zLeaf as well:

/language en /name A /server 144.214.99.243

Alternatively, replace "server.eec" with the IP address in it.

- if you want to open zTree and multiple zLeafs with one click you can create a *.bat file (= a text file ending with ".bat") like this:

```
start ztree /language en
PING 1.1.1.1 -n 1 -w 2000 >NUL
start zleaf /size 640x480 /position 0,0 /language en/name A
PING 1.1.1.1 -n 1 -w 500 >NUL
start zleaf /size 640x480 /position 640,0 /language en/name B
PING 1.1.1.1 -n 1 -w 500 >NUL
start zleaf /size 640x480 /position 0,480 /language en/name C
PING 1.1.1.1 -n 1 -w 500 >NUL
start zleaf /size 640x480 /position 640,480 /language en/name D
```

Step 1

- If you run more than one zTree, you can use different chanels (/chanel CH) to indicate what zTree connects to which zLeafs (CH is a number)
- You can also direct different files into different directories:

  /datadir DIR for xls (data) file
  /adradir DIR for adr (address) file
  /gsfdir DIR for gsf (gamesafe) fille
  …

- if you have older programs (from version 2.x) do not mix them with the new ones as they will not work.

Step 3

Step 2: Open zTree (through the shortcut)

Step 3: download, create, save your program (e.g. PD.ztt)

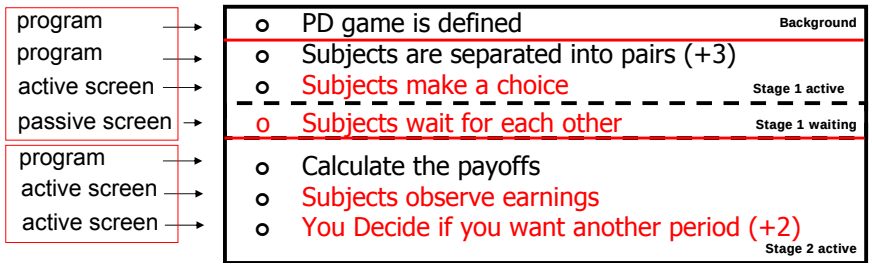
We will write a program for PD experiment subjects earn $10 + half of the these:

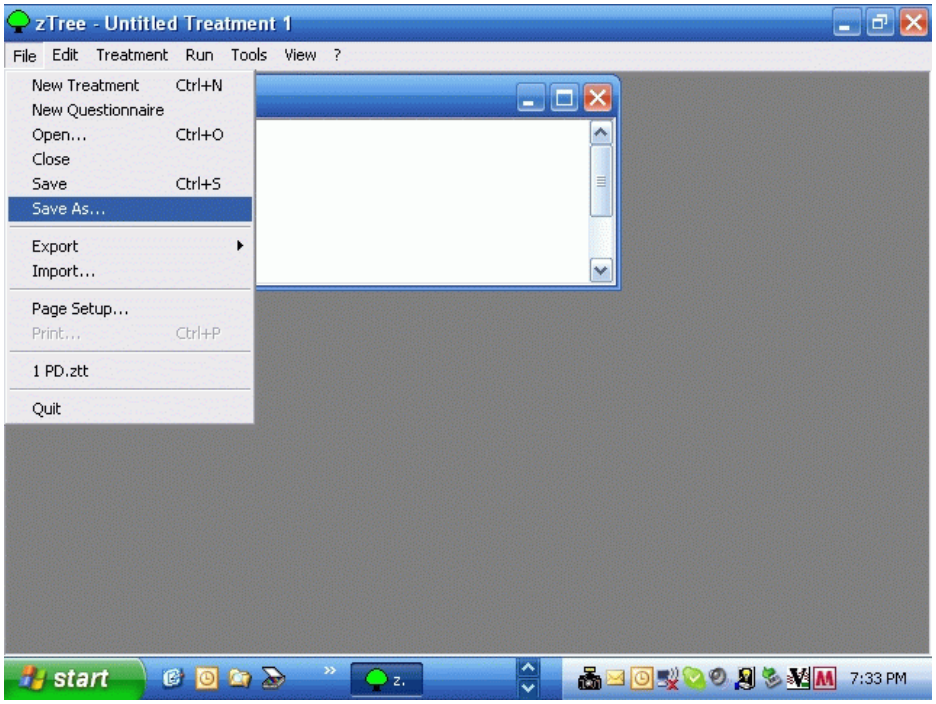
	C	D
C	5,5	0,8
D	8,0	2,2

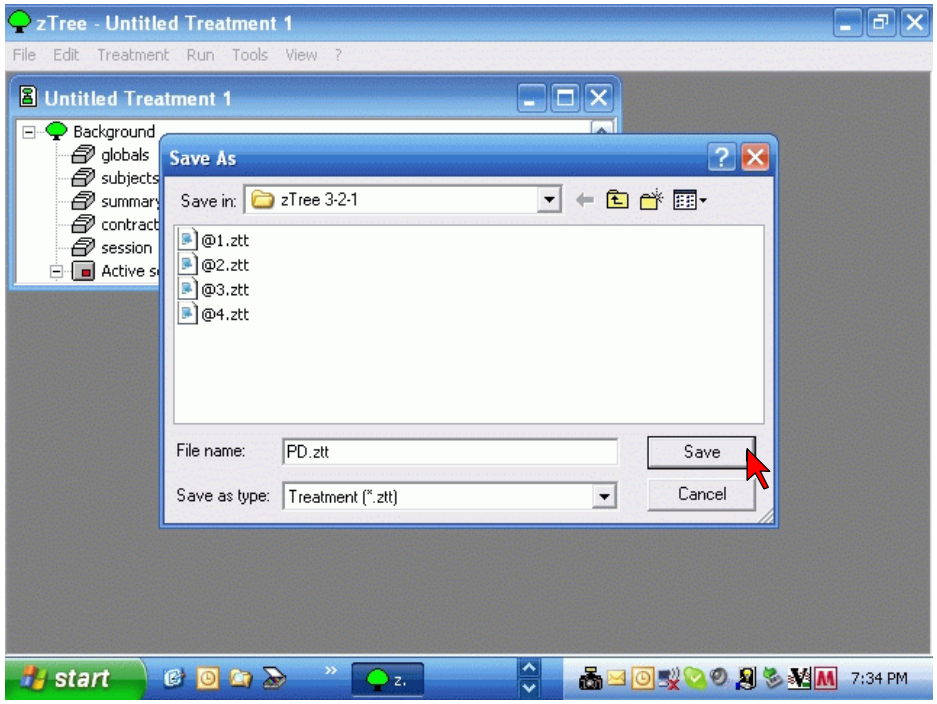
(1) two groups of two players playing once

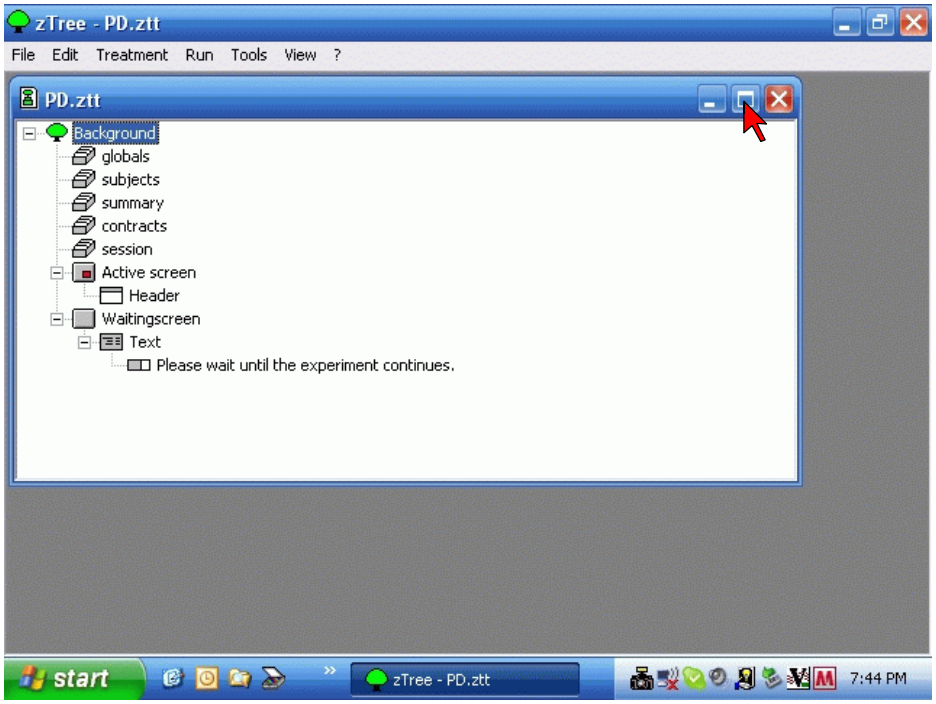
(2) two groups of two players playing once or more…

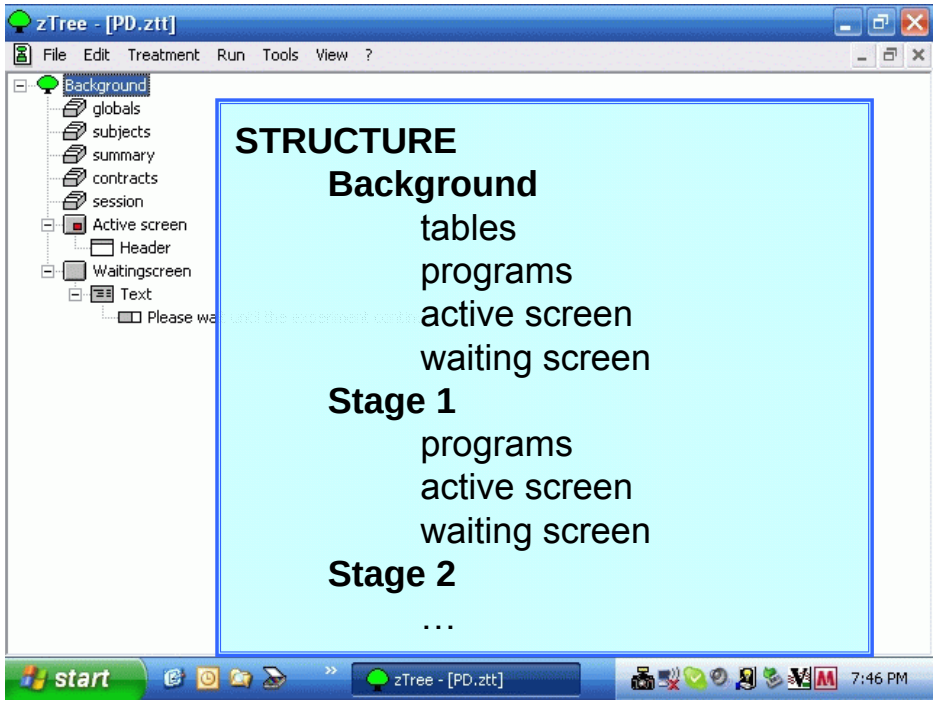
(3) rematch agents based on last period earnings (highest with 2nd highest and so on….)

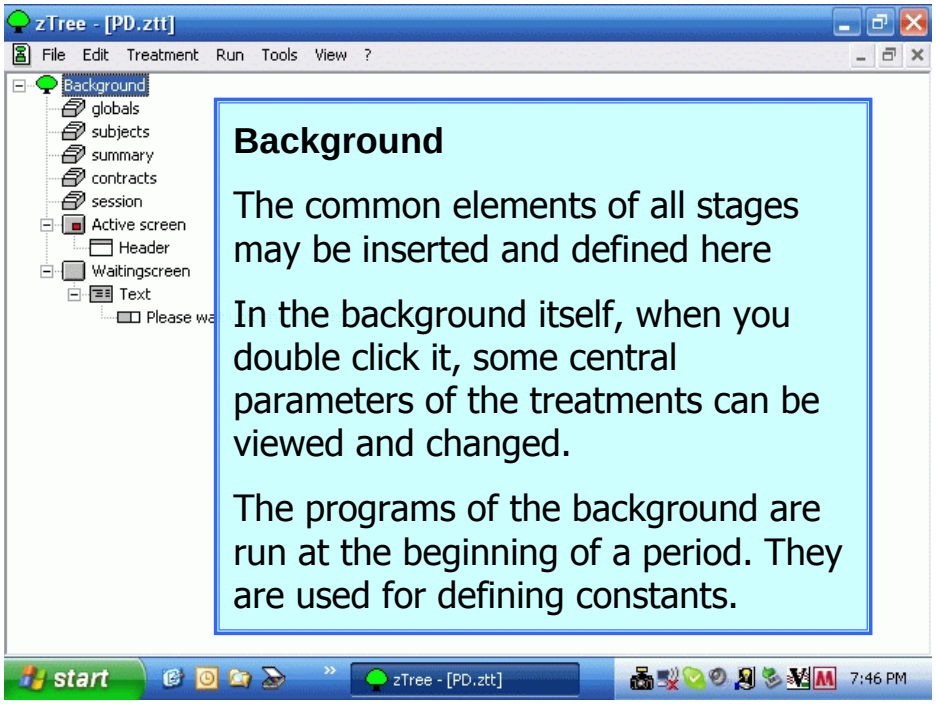



zTree - Untitled Treatment 1
File Edit Treatment Run Tools View ?
New Treatment Ctrl+N
New Questionnaire
Open... Ctrl+O
Close
Save Ctrl+S
Save As...
Export
Import...
Page Setup...
Print... Ctrl+P
1 PD.ztt
Quit
start
7:33 PM


zTree - Untitled Treatment 1
File Edit Treatment Run Tools View ?
Untitled Treatment 1
Background
globals
subjects
summary
contracts
session
Active s
Save As
Save in: zTree 3-2-1
@1.ztt
@2.ztt
@3.ztt
@4.ztt
File name: PD.ztt
Save as type: Treatment (*.ztt)
Save
Cancel
start
7:34 PM


zTree - PD.ztt
File Edit Treatment Run Tools View ?
PD.ztt
Background
globals
subjects
summary
contracts
session
Active screen
Header
Waitingscreen
Text
Please wait until the experiment continues.
start
zTree - PD.ztt
7:44 PM


zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
Active screen
Header
Waitingscreen
Text
STRUCTURE
Background
tables
programs
active screen
waiting screen
Stage 1
programs
active screen
waiting screen
Stage 2
…
start
zTree - [PD.ztt]
7:46 PM

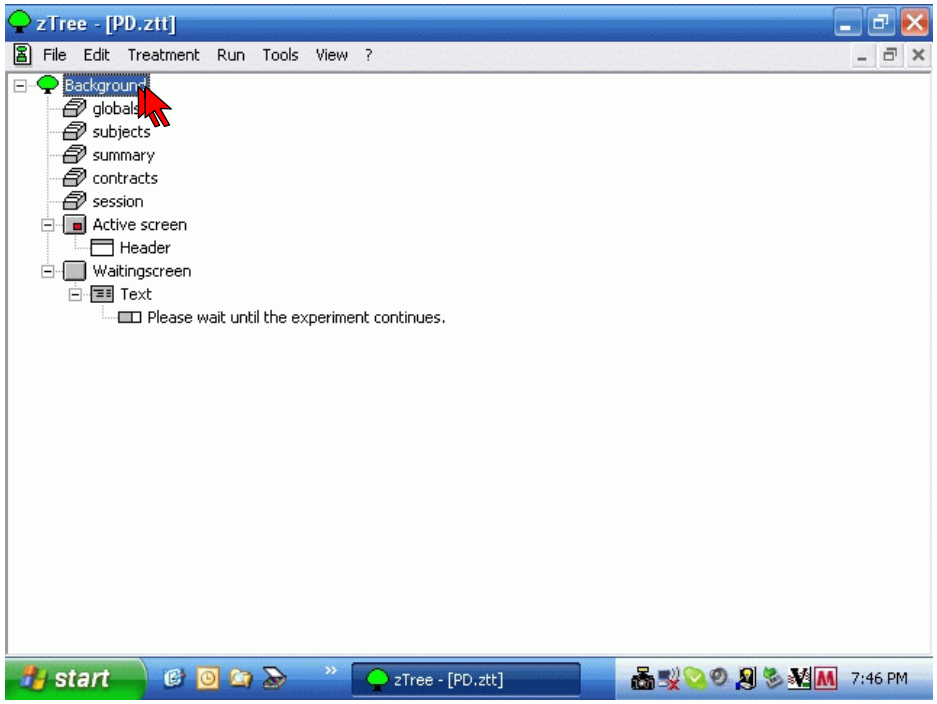


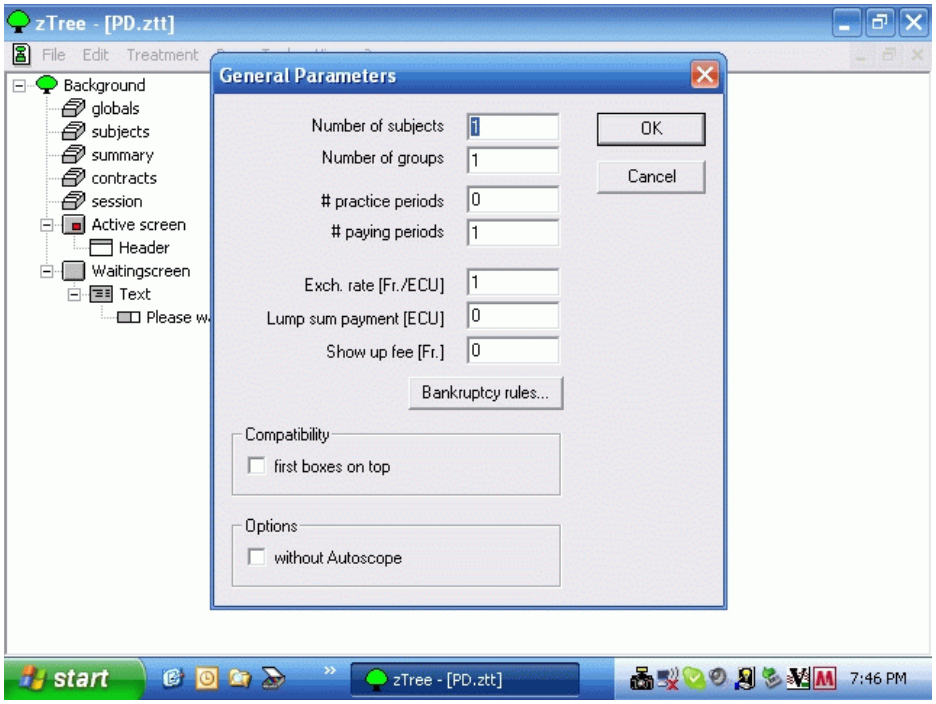
Background

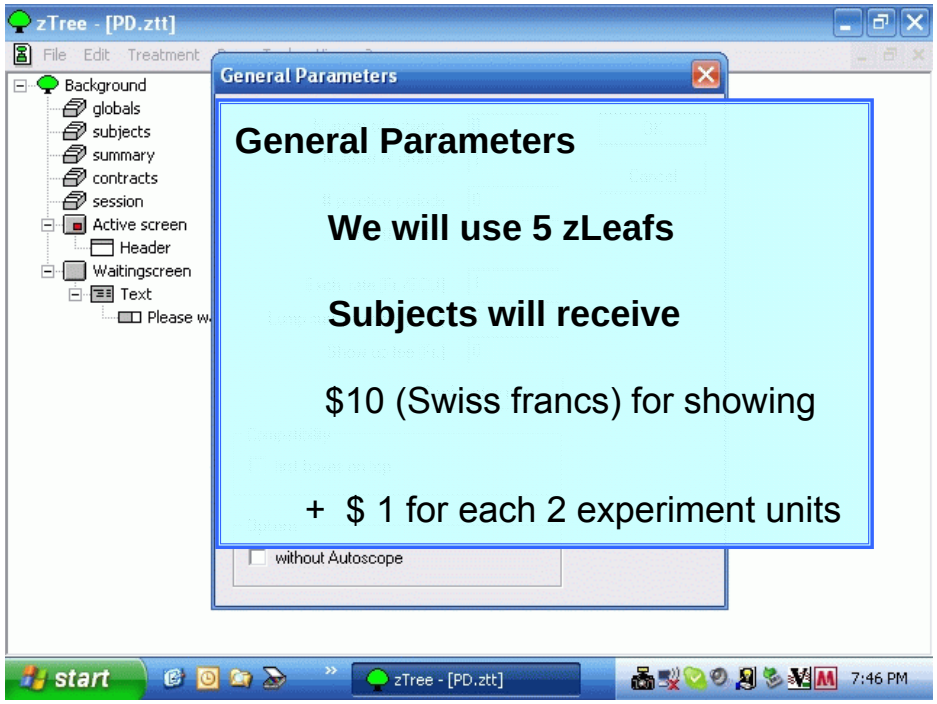
The common elements of all stages may be inserted and defined here

In the background itself, when you double click it, some central parameters of the treatments can be viewed and changed.

The programs of the background are run at the beginning of a period. They are used for defining constants.







General Parameters

We will use 5 zLeafs

Subjects will receive

$10 (Swiss francs) for showing

+ $ 1 for each 2 experiment units

zTree - [PD.ztt]

File Edit Treatment

Background
globals
subjects
summary
contracts
session
Active screen
Header
Waitingscreen
Text
Please w

General Parameters

Number of subjects: 5
Number of groups: 1
practice periods: 0
paying periods: 1
Exch. rate [Fr./ECU]: 0.5
Lump sum payment [ECU]: 0
Show up fee [Fr.]: 10

OK
Cancel

Bankruptcy rules...

Compatibility
first boxes on top

Options
without Autoscope

start | zTree - [PD.ztt] | 7:46 PM

Bankruptcy Rules

Text "Do you invest your show up fee ?"

You are making losses. Do you want to invest the showupfee in order to continue?

"yes" Yes — Show up fee is invested and experiment goes on

"no" No — Message "BancruptShowupNo" appers in client's table*

Text "Do you want to go on"

You are making losses. Do you want to continue?

"yes" Yes — Message "BancruptMoreYes" appers in client's table*

"no" No — Message "BancruptMoreNo" appers in client's table*

Text "Please wait until the experimenter unlocks your PC."

Please wait until the experimenter unlocks your PC.

* The experimenter can either
- give permission to go on (give a credit)
- replace the subject (nullify payoffs)

Cancel | OK

start | 7:21 PM

---

zTree - [PD.ztt]

File Edit Treatment

Background
globals
subjects
summary
contracts
session
Active screen
Header
Waitingscreen
Text
Please w

General Parameters

Number of subjects: 5
Number of groups: 1
practice periods: 0
paying periods: 1
Exch. rate [Fr./ECU]: 0.5
Lump sum payment [ECU]: 0
Show up fee [Fr.]: 10

OK
Cancel

Bankruptcy rules...

Compatibility
first boxes on top

Options
without Autoscope

start | zTree - [PD.ztt] | 7:46 PM

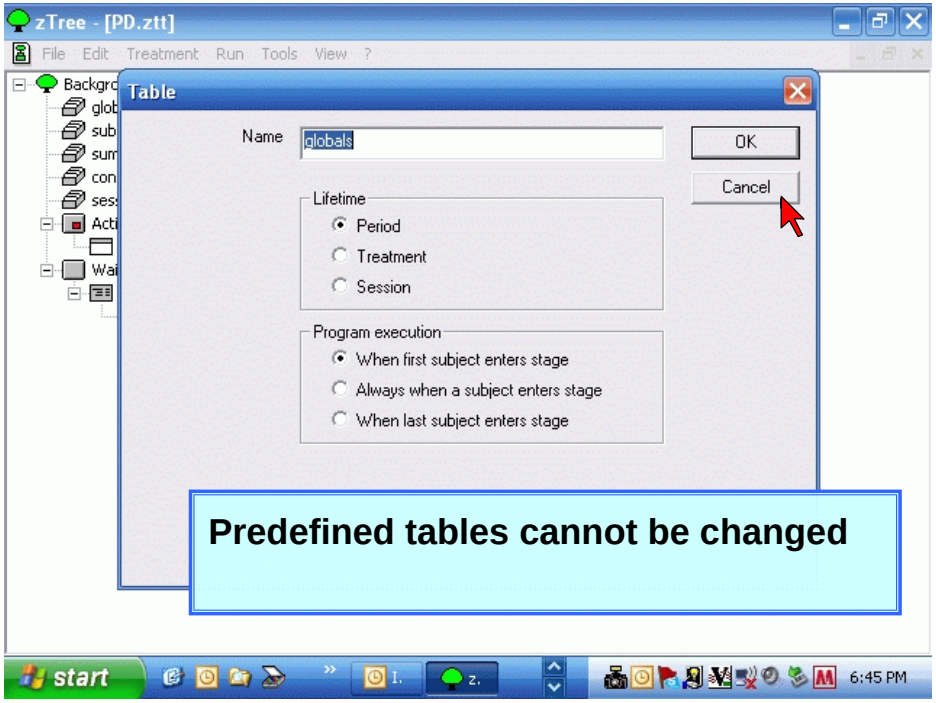
---

zTree - [PD.ztt]

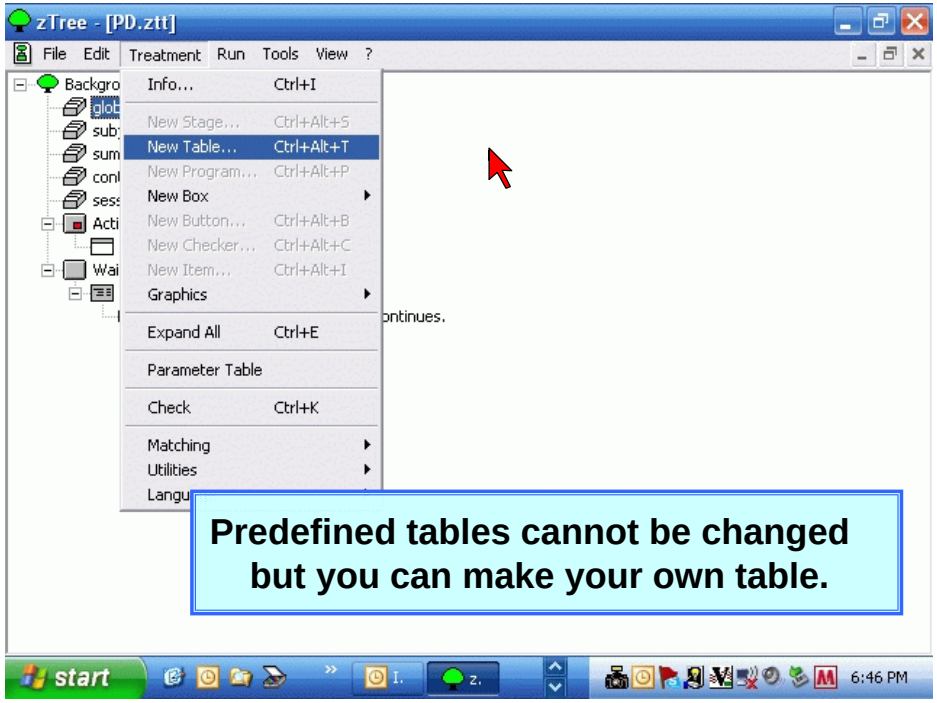
File Edit Treatment Run Tools View ?

Background
globals
subjects
summary
contracts
session
Active screen
Header
Waitingscreen
Text
Please wait until the experiment continues.

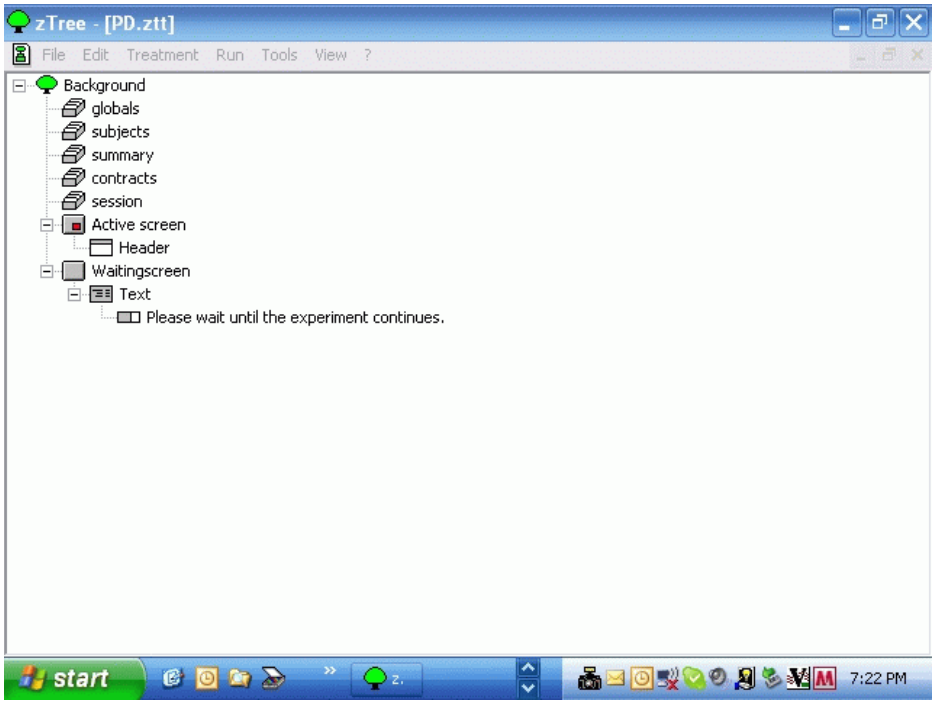
start | 6:45 PM

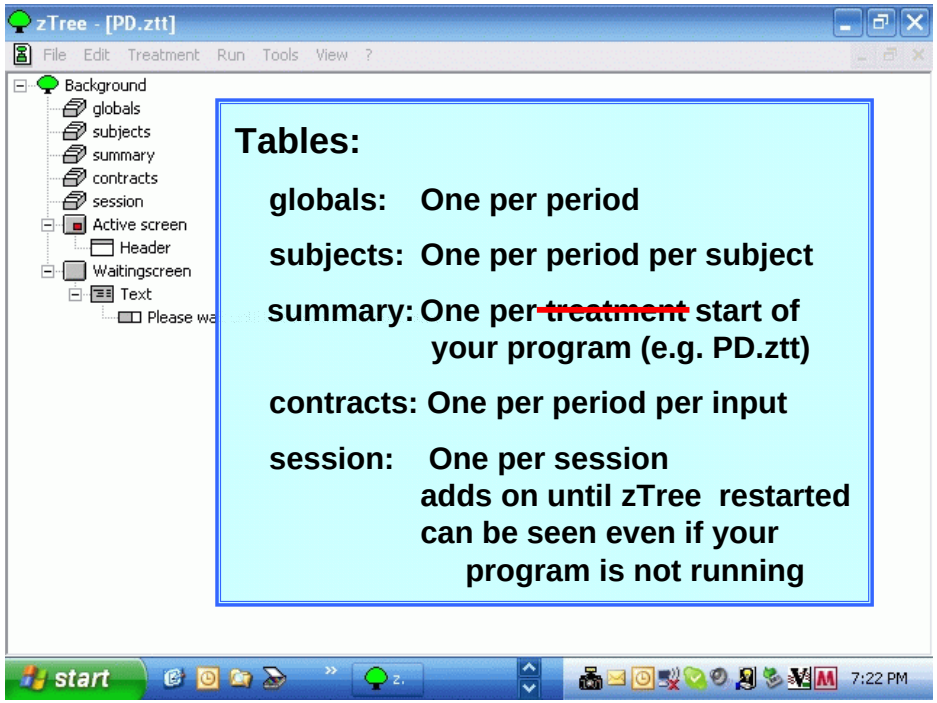


Predefined tables cannot be changed



Predefined tables cannot be changed but you can make your own table.





Tables:

globals: One per period

subjects: One per period per subject

summary: One per ~~treatment~~ start of your program (e.g. PD.ztt)

contracts: One per period per input

session: One per session adds on until zTree restarted can be seen even if your program is not running

zTree - [PD.ztt]

File Edit Treatment Run Tools View ?

- Background
  - globals
  - subjects
  - summary
  - contracts
  - session
  - Active screen
    - Header
  - Waitingscreen
    - Text
    - Please wa

globals:

Period (firs of three trials = -2)
NumPeriods
RepeatTreatment (=0)

subjects:

Period
Subject
Group
Profit
TotalProfit(adds from last period)

Participate (0=> skip all stage)
LeaveStage (0=> go to stage end)

start 7:22 PM

zTree - [PD.ztt]

File Edit Treatment Run Tools View ?

- Background
  - globals
  - subjects
  - summary
  - contracts
  - session
  - Active screen
    - Header
  - Waitingscreen
    - Text
    - Please wa

globals:

X, x, x1, ...Xx[xX]

OLDglobals.find(X);

subjects:

Y, y, y1, ...Y2[Y3], X

OLDsubjects.find (Subject>N , Y)

start 7:22 PM

zTree - [PD.ztt]

File Edit Treatment Run Tools View ?

- Background
  - globals
  - subjects
  - summary
  - contracts
  - session
  - Active screen
    - Header
  - Waitingscreen
    - Text
    - Please wa

summary:

Period
+ your aggregates to be observed during experiment

contracts:

Period + contracts & other multiple actions per period

sessions:

Subject
FinalProfit (<=Subject's TotalProfit)

start 7:22 PM

zTree - [PD.ztt]

File Edit Treatment Run Tools View ?

- Background
  - globals
  - subjects
  - summary
  - contracts
  - session
  - Active screen
    - Header
  - Waitingscreen
    - Text
    - Please wa

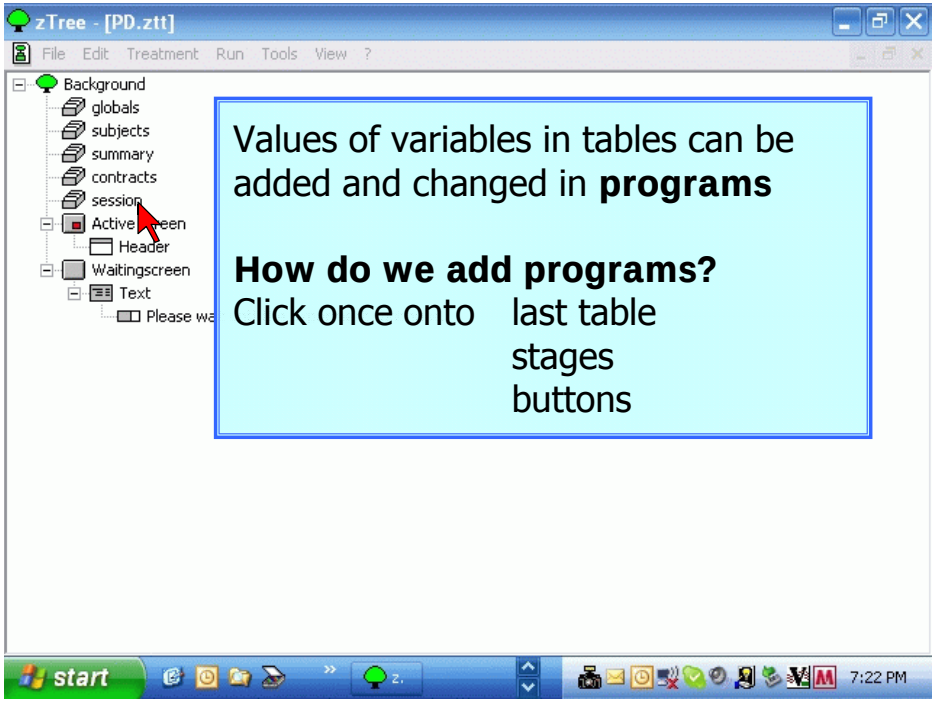
summary

contracts

sessions:

Subject
FinalProfit (<=TotalProfit in $)
ShowUpFee
ShowUpFeeInvested (1=yes)
MoneyAdded
MoneyToPay
(=FinaProfit+ShowUpFee+MoneyAdded)
MoneyEarned(=FinaProfit+ShowUpFee)

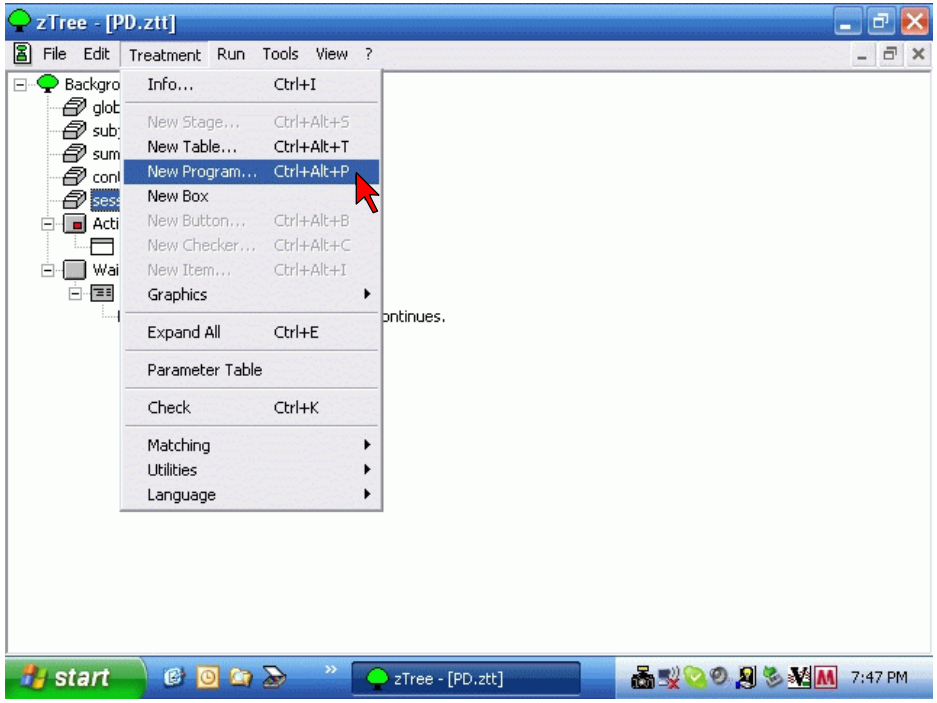
start 7:22 PM

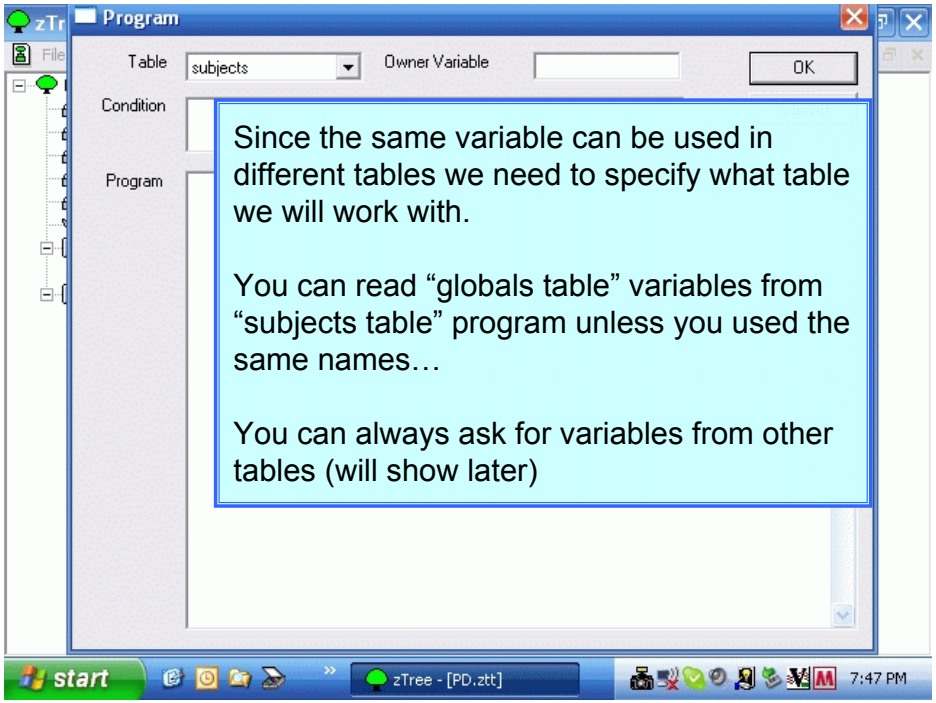


Values of variables in tables can be added and changed in **programs**

How do we add programs?

Click once onto last table
stages
buttons

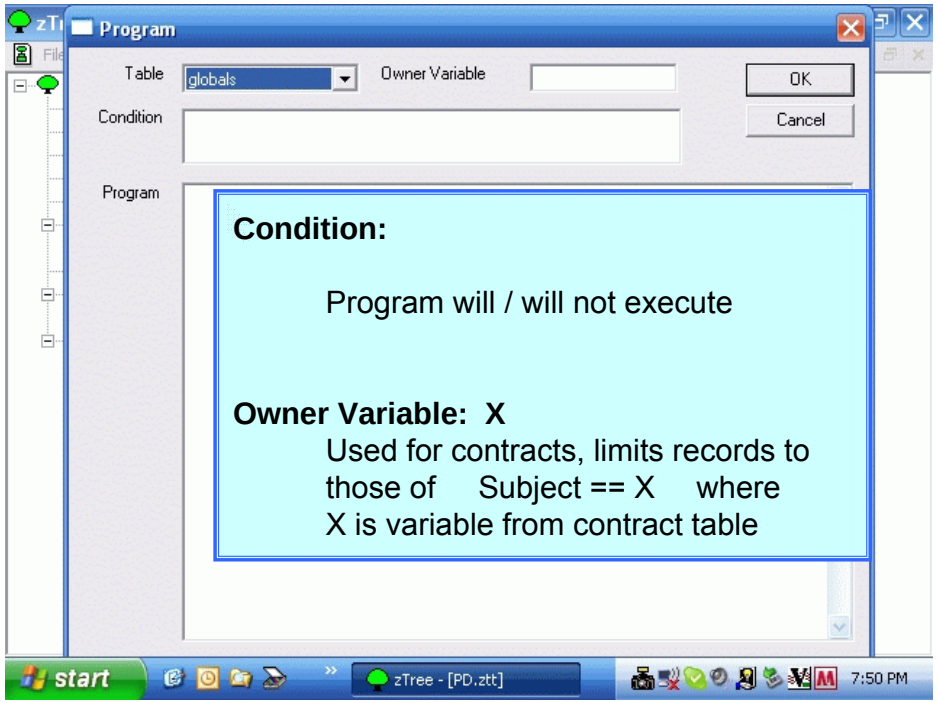




Since the same variable can be used in different tables we need to specify what table we will work with.

You can read “globals table” variables from “subjects table” program unless you used the same names…

You can always ask for variables from other tables (will show later)



Condition:

Program will / will not execute

Owner Variable: X

Used for contracts, limits records to those of Subject == X where X is variable from contract table

Program

Table: globals | Owner Variable | OK | Cancel

Condition

Program

In this program, we will

find number of subjects

decide number of groups

define constants

so we can change them easily for different treatments

start | zTree - [PD.ztt] | 7:50 PM

Program

Table: globals | Owner Variable | OK | Cancel

Condition

Program

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS / 2 , 1);

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;    CD = 0;
DC = 8;    DD = 2;
```

We could also define strategies by names

e.g. **cooperation = 1;**
defection = 0;

but we will just remember this.

start | zTree - [PD.ztt] | 8:01 PM

Program

Table: globals | Owner Variable | OK | Cancel

Condition

Program

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS / 2 , 1);

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;    CD = 0;
DC = 8;    DD = 2;
```

subjects.count();

subjects.count(Subject<5);

start | zTree - [PD.ztt] | 8:01 PM

Program

Table: globals | Owner Variable | OK | Cancel

Condition

Program

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS /

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;    CD = 0;
DC = 8;    DD = 2;
```

TABLE FUNCTIONS:

T.count(a)

T.average(a,x) first value: **T.find(a,x)**

T.maximum(a,x) **T.median(a,x)**

T.minimum(a,x) **T.sum(a,x)**

T.product(a,x) **T.stddev(a,x)**

T.regressionslope(a,x)

T = table
x = variable
a = condition **(i.e. Group==Subject)**

start | zTree - [PD.ztt] | 8:01 PM

Program

Table: globals | Owner Variable: | OK | Cancel

Condition:

Program:

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS / 2 , 1);

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;   CD = 0;
DC = 8;   DD = 2;
```

Program

Table: globals | Owner Variable: | OK | Cancel

Condition:

Program:

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS /

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;   CD = 0;
DC = 8;   DD = 2;
```

FUNCTIONS:

if(a, x, y) (ie. Equals to y if a is FALSE)

abs(x) sqrt(x) power(x,y)
sin(x) random() ∈ [0, 1]

min(x,y) max(x,y) mod(x,y)
round(x,y) = multiples of y
roundup(x,y) = nearest higher (-3.2 => -3)

and(a,b) or(a,b) not(a)

gettime() # of seconds since computer was turned on
...

CONSTANTS: TRUE FALSE

Program

Table: globals | Owner Variable: | OK | Cancel

Condition:

Program:

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS /

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;   CD = 0;
DC = 8;   DD = 2;
```

CONSTANTS: TRUE FALSE

OPERATORS:

+ - * /
< <= == != >= >

& (==logical and) | (==logical or)

SCOPE OPPERATORS:

\ globals table: e.g. \SUBS
: next higher scope

Program

Table: globals | Owner Variable: | OK | Cancel

Condition:

Program:

```
// Parameters

SUBS = subjects.count();
GRPS = rounddown( SUBS /

TimeStage1 = 30;
TimeStage2 = 30;

CC = 5;   CD = 0;
DC = 8;   DD = 2;
```

CONSTANTS: TRUE FALSE

OPERATORS:

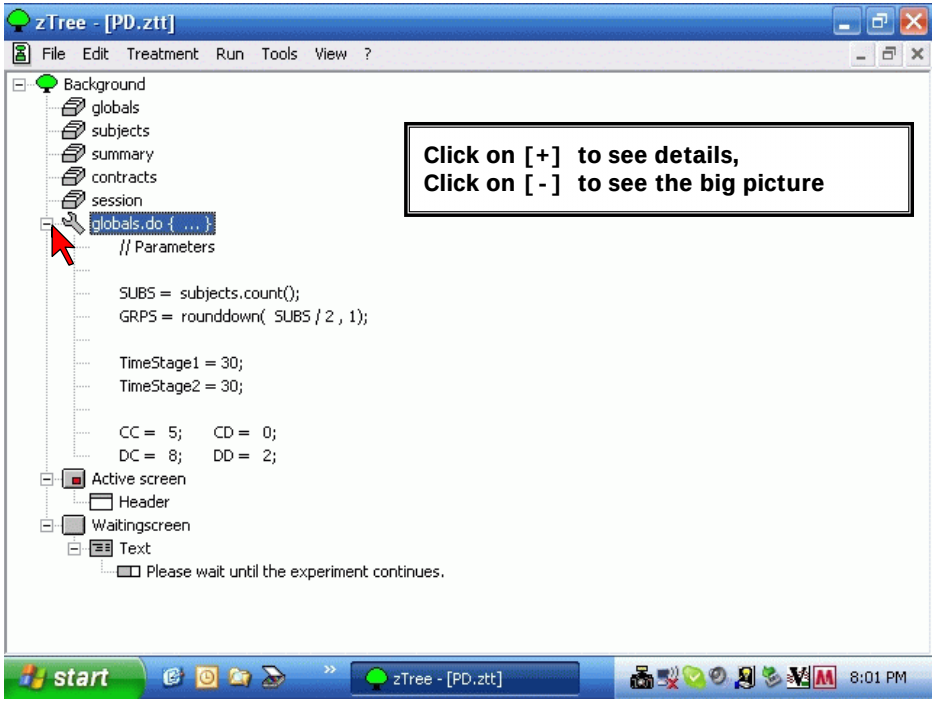
+ - * /
< <= == != >= >

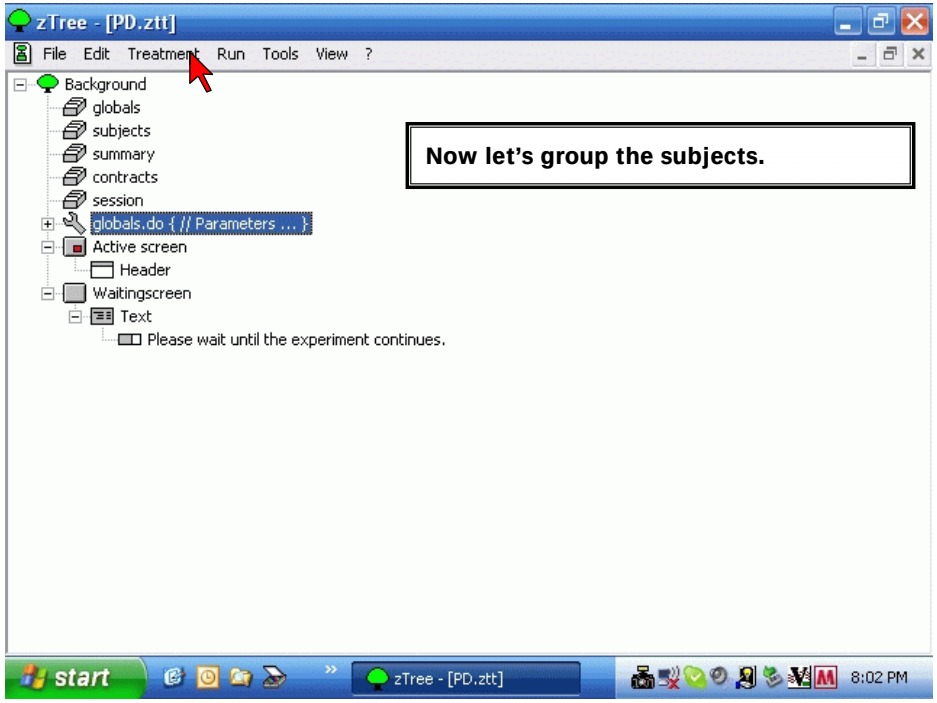
& (==logical and) | (==logical or)

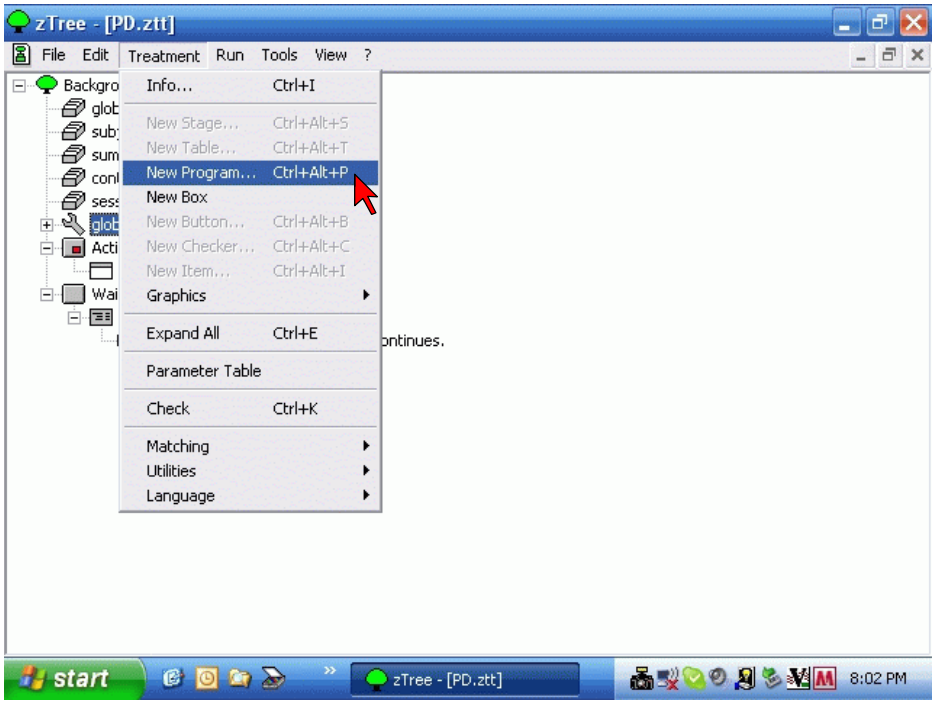
SCOPE OPPERATORS:

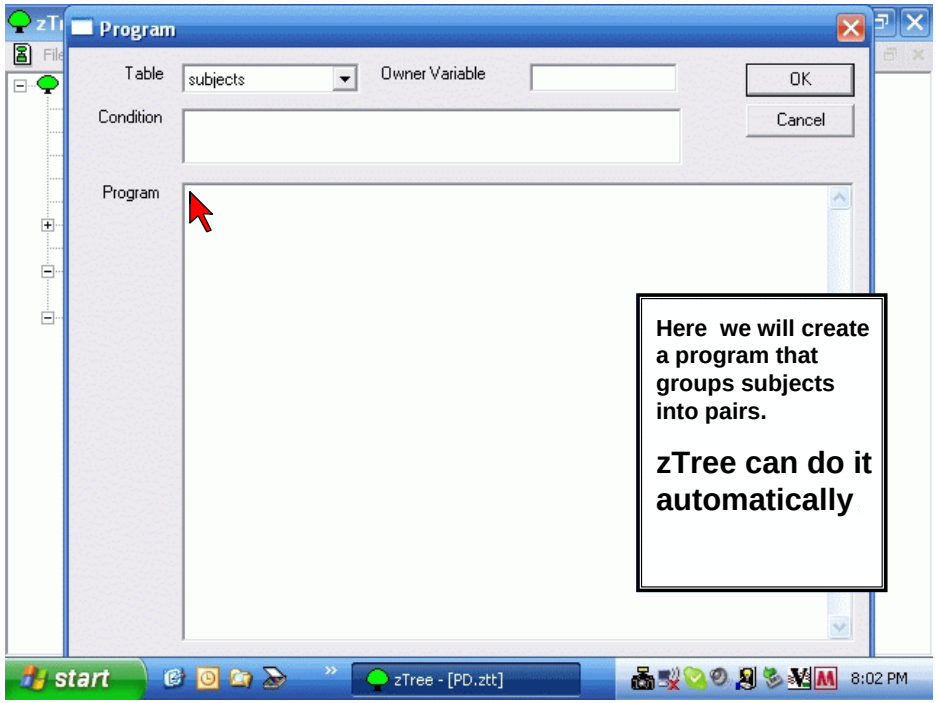
\ globals table: e.g. \SUBS

: next higher scope
e.g. ":Subject" is ID of subject that asked to see a table
(you will see example later)

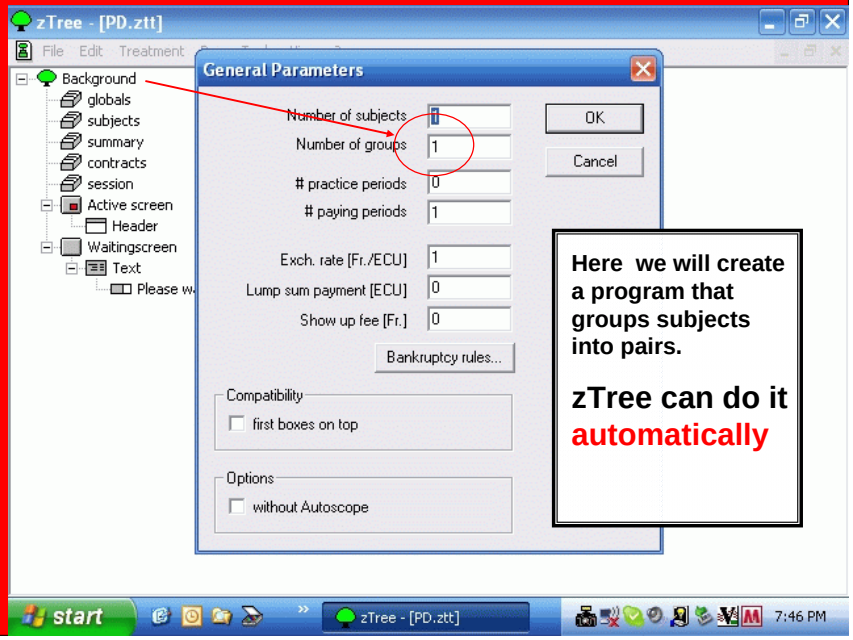

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { ... }
// Parameters
SUBS = subjects.count();
GRPS = rounddown(SUBS / 2 , 1);
TimeStage1 = 30;
TimeStage2 = 30;
CC = 5; CD = 0;
DC = 8; DD = 2;
Active screen
Header
Waitingscreen
Text
Please wait until the experiment continues.
Click on [+] to see details,
Click on [-] to see the big picture
start
zTree - [PD.ztt]
8:01 PM


zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { // Parameters ... }
Active screen
Header
Waitingscreen
Text
Please wait until the experiment continues.
Now let's group the subjects.
start
zTree - [PD.ztt]
8:02 PM


zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
start
zTree - [PD.ztt]
8:02 PM


Program
Table
subjects
Owner Variable
OK
Cancel
Condition
Program
Here we will create a program that groups subjects into pairs.
zTree can do it automatically
start
zTree - [PD.ztt]
8:02 PM

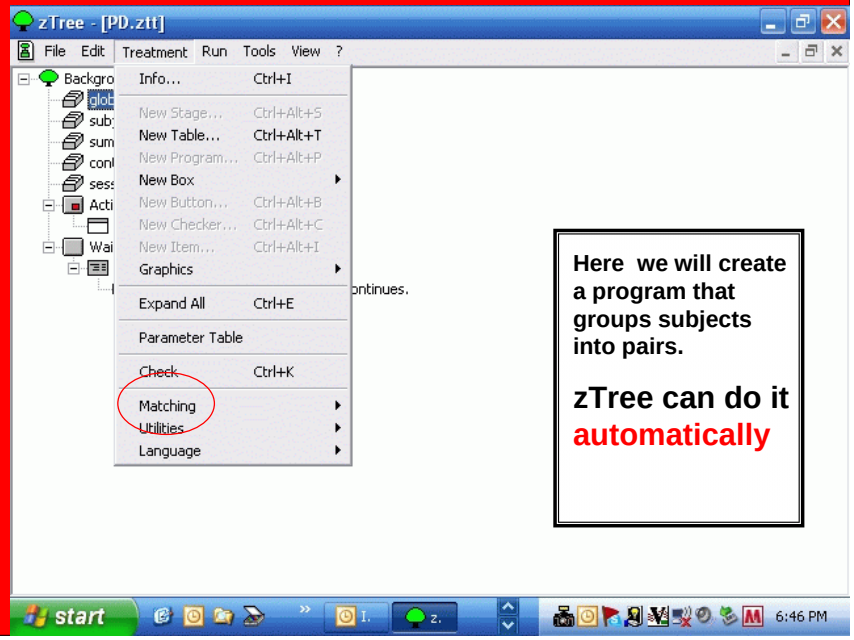
I DO NOT USE THIS



Here we will create a program that groups subjects into pairs.

zTree can do it automatically

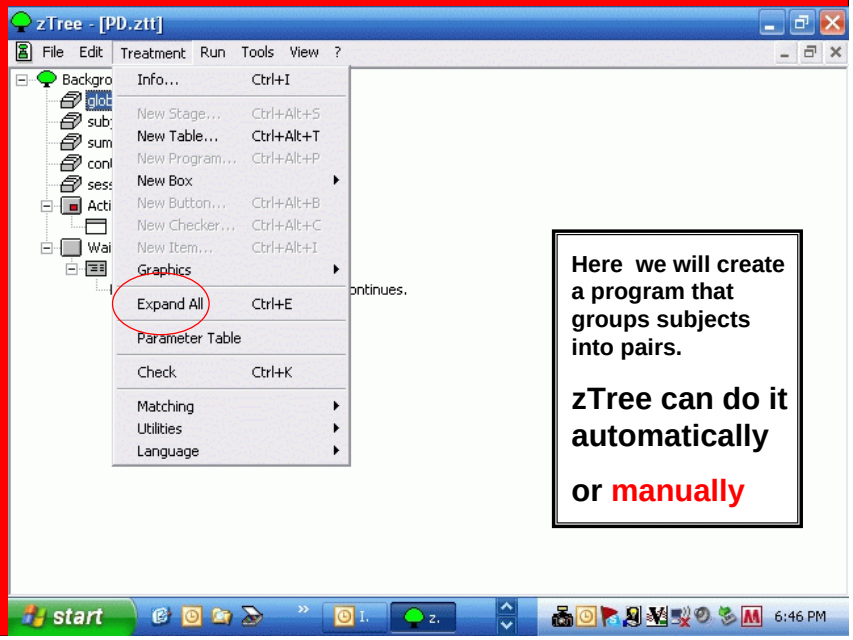
I DO NOT USE THIS



Here we will create a program that groups subjects into pairs.

zTree can do it automatically

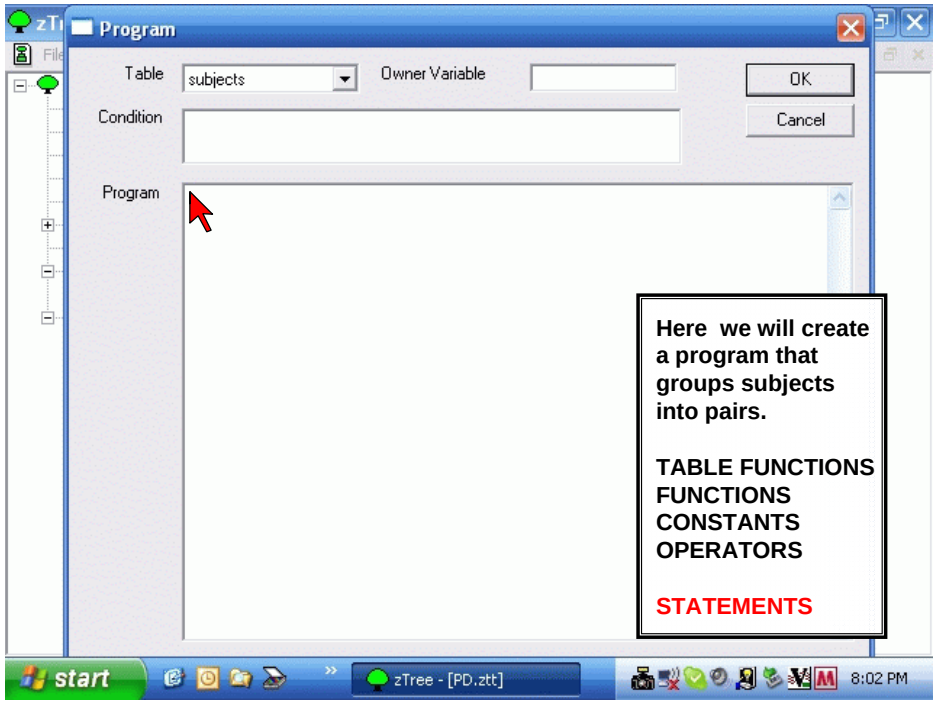
I DO NOT USE THIS



Here we will create a program that groups subjects into pairs.

zTree can do it automatically

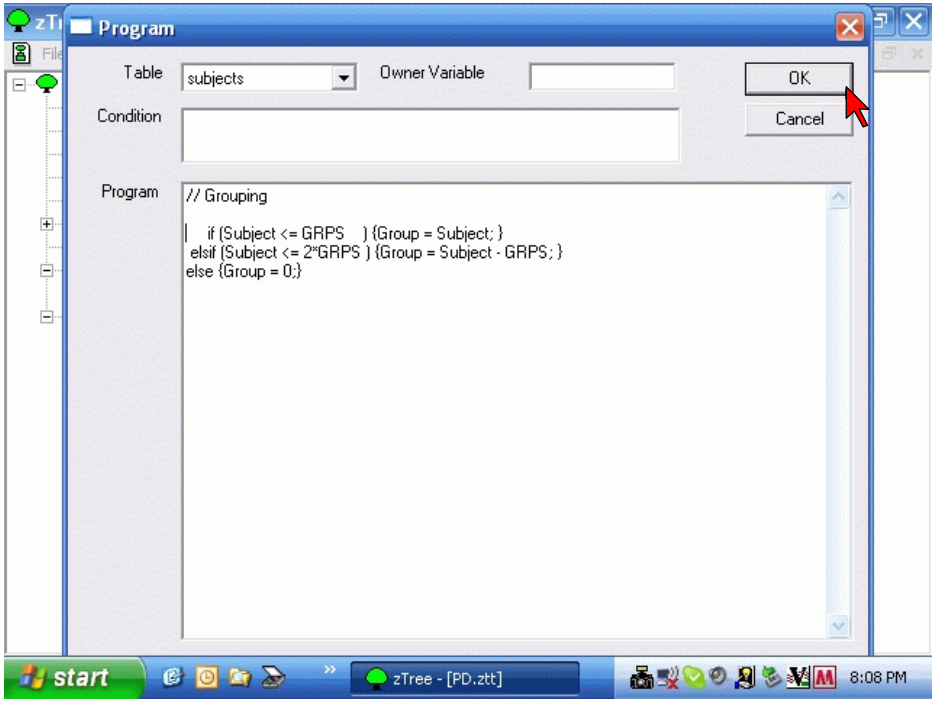
or manually

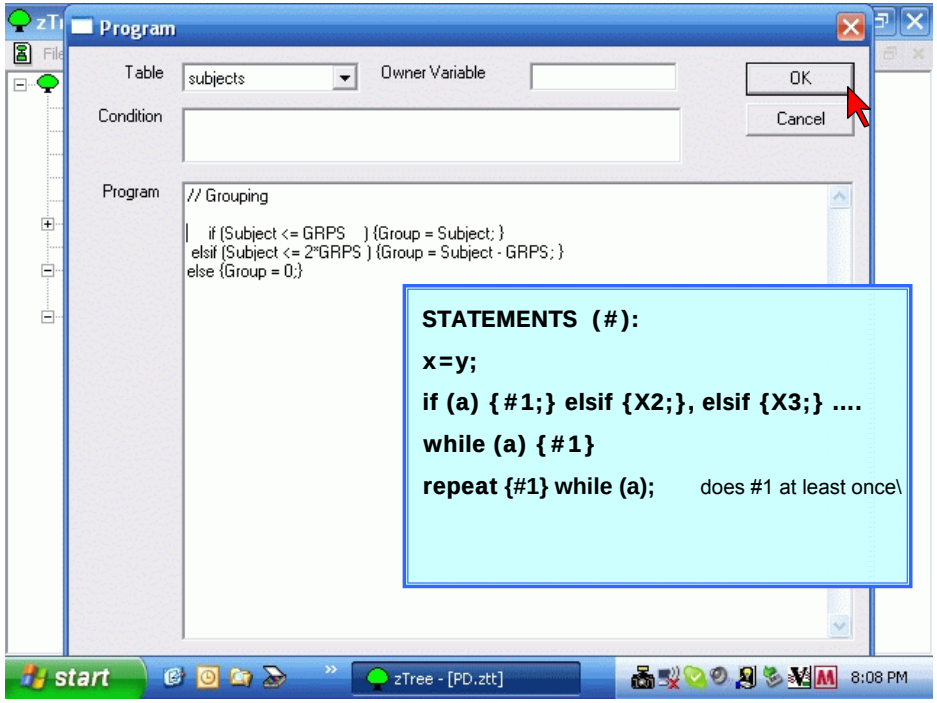


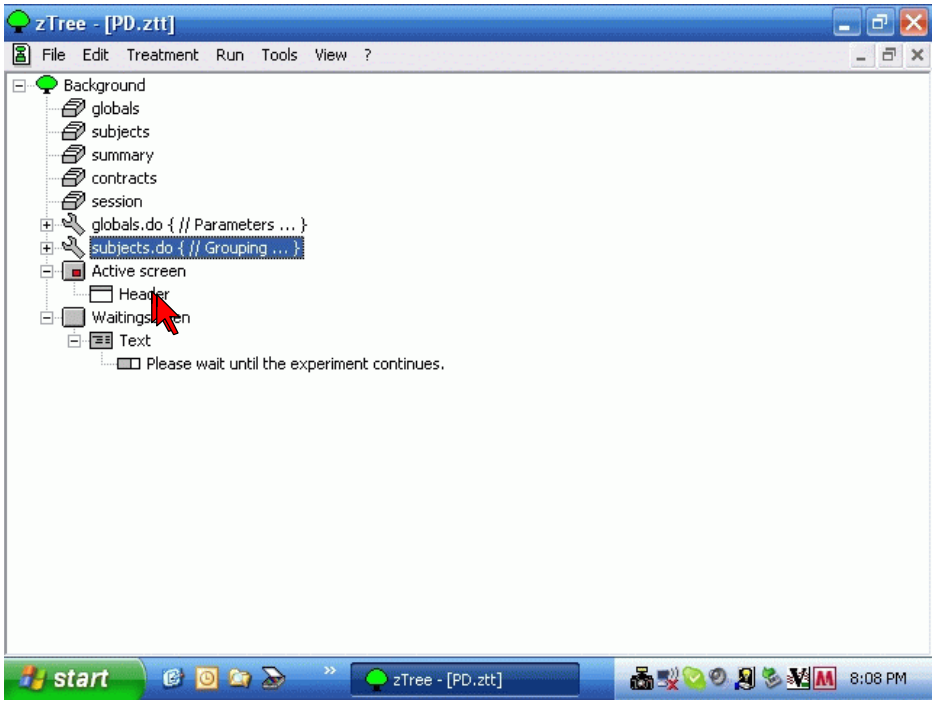
Here we will create a program that groups subjects into pairs.

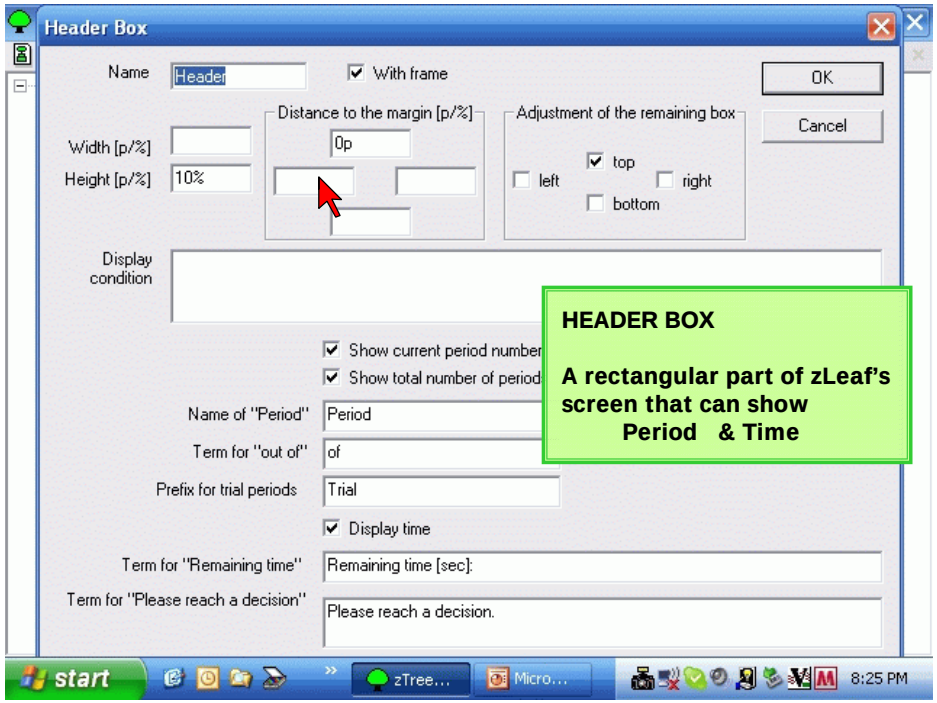
TABLE FUNCTIONS
FUNCTIONS
CONSTANTS
OPERATORS

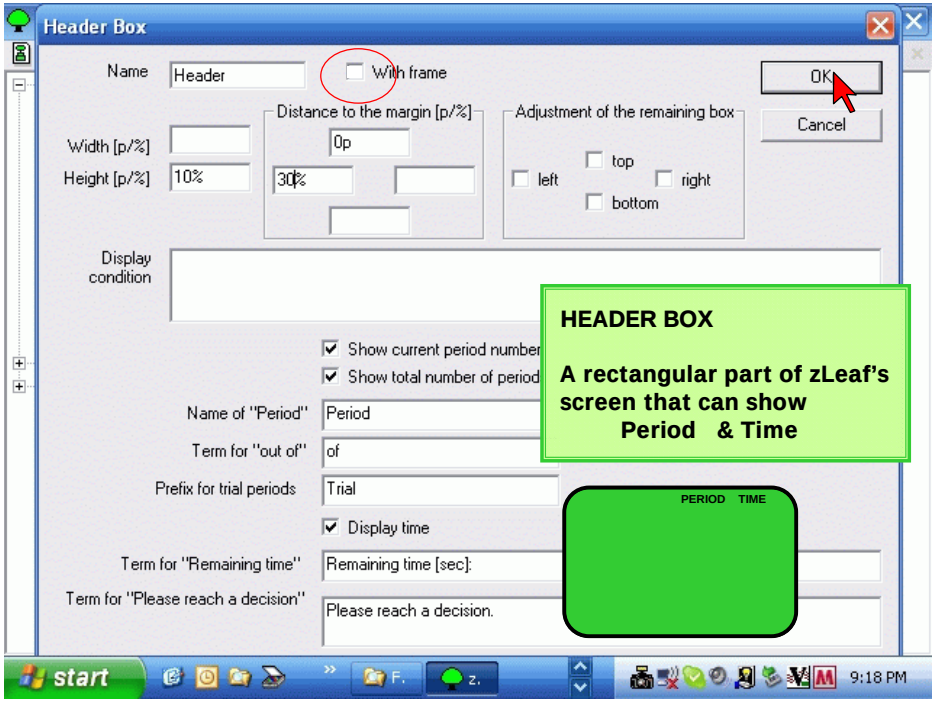
STATEMENTS


Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
// Grouping
if (Subject <= GRPS) {Group = Subject; }
elsif (Subject <= 2*GRPS) {Group = Subject - GRPS; }
else {Group = 0;}
start
zTree - [PD.ztt]
8:08 PM


Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
// Grouping
if (Subject <= GRPS) {Group = Subject; }
elsif (Subject <= 2*GRPS) {Group = Subject - GRPS; }
else {Group = 0;}
STATEMENTS (#):
x=y;
if (a) {#1;} elsif {X2;}, elsif {X3;} ....
while (a) {#1}
repeat {#1} while (a);
does #1 at least once\
start
zTree - [PD.ztt]
8:08 PM

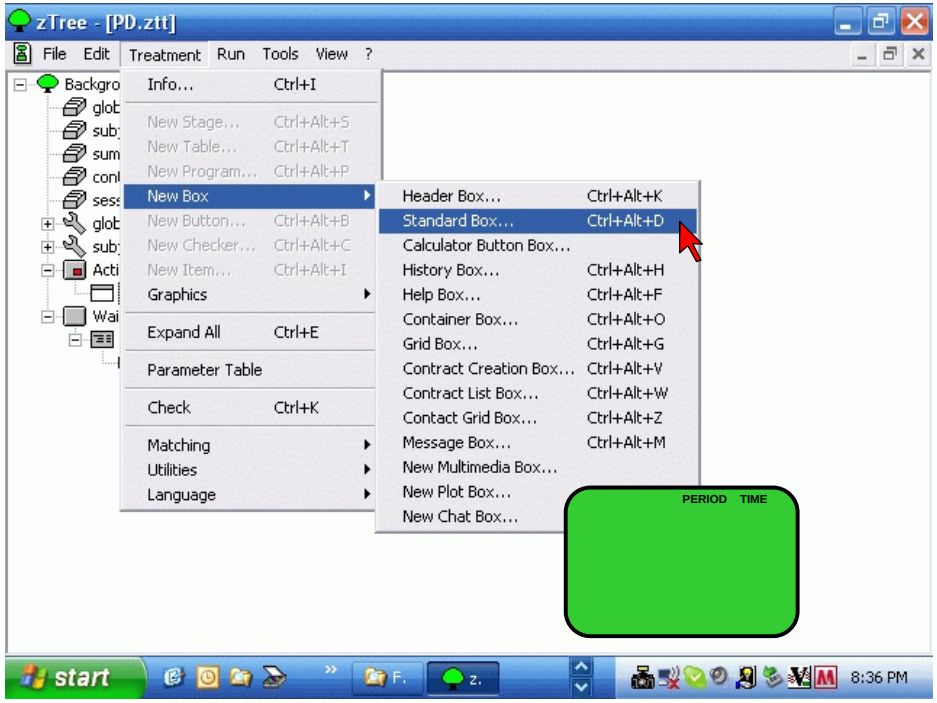

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { // Parameters ... }
subjects.do { // Grouping ... }
Active screen
Header
Waitingscreen
Text
Please wait until the experiment continues.
start
zTree - [PD.ztt]
8:08 PM

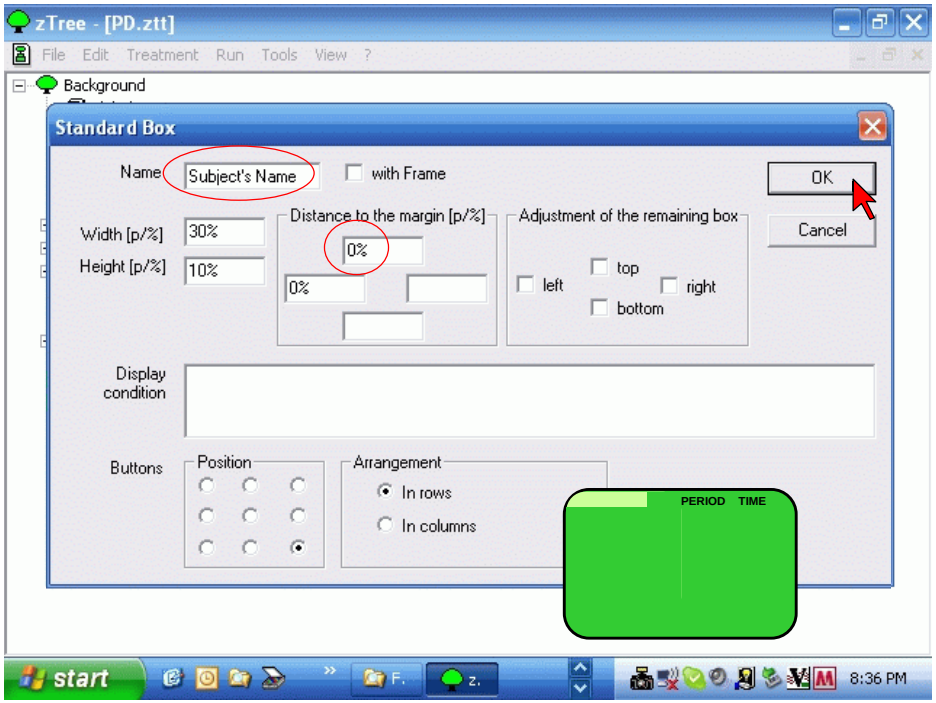

Header Box
Name
Header
With frame
OK
Cancel
Distance to the margin [p/%]
Adjustment of the remaining box
Width [p/%]
Height [p/%]
10%
0p
top
left
right
bottom
Display condition
Show current period number
Show total number of period
Name of "Period"
Period
Term for "out of"
of
Prefix for trial periods
Trial
Display time
Term for "Remaining time"
Remaining time [sec]:
Term for "Please reach a decision"
Please reach a decision.
HEADER BOX
A rectangular part of zLeaf's screen that can show Period & Time
start
zTree...
Micro...
8:25 PM

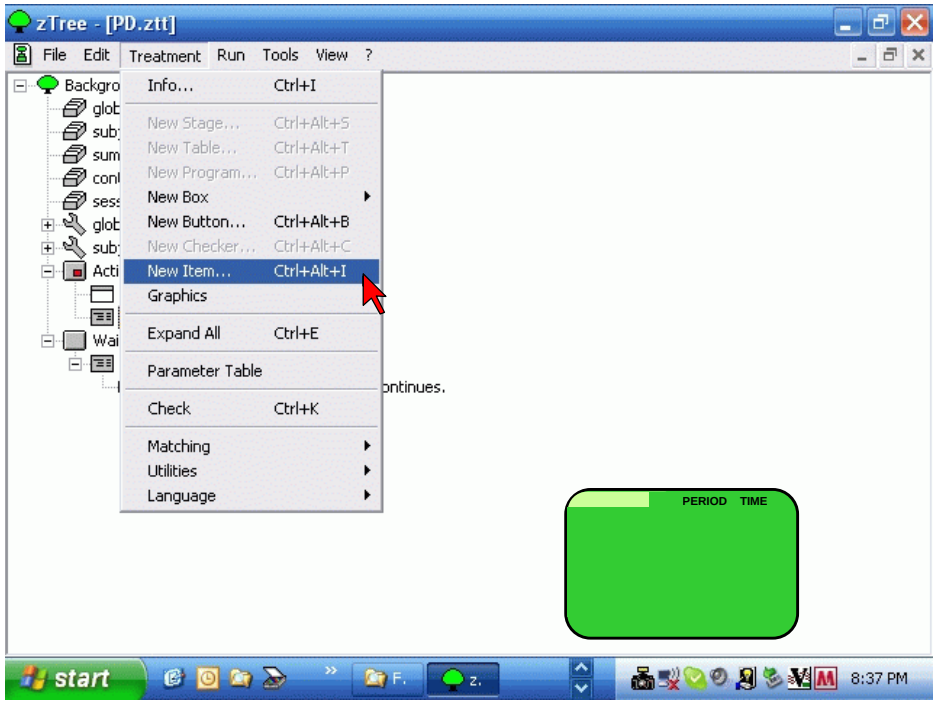


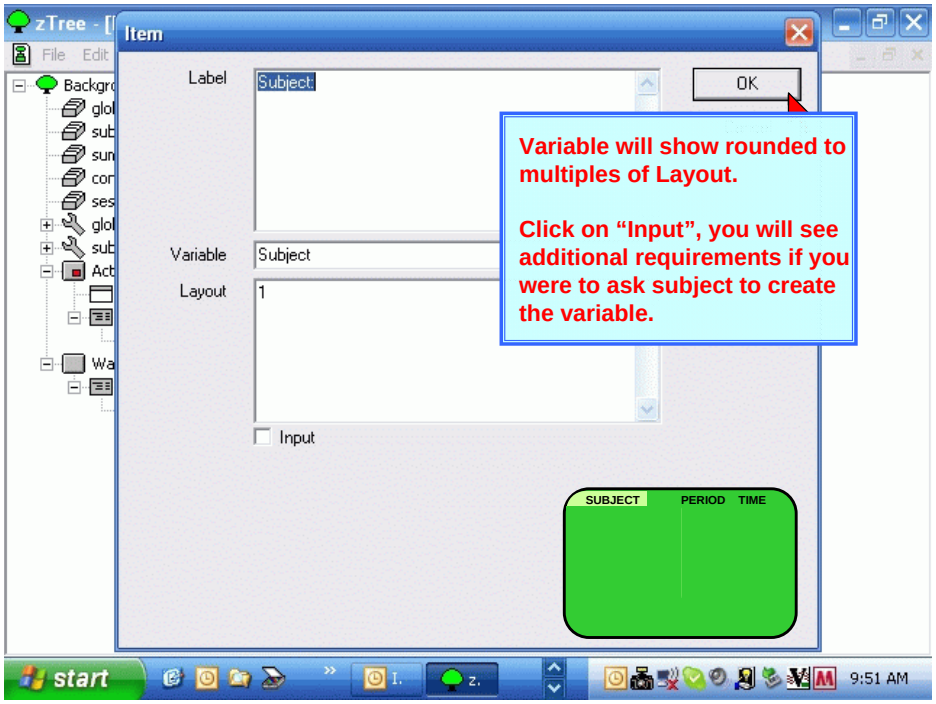
HEADER BOX

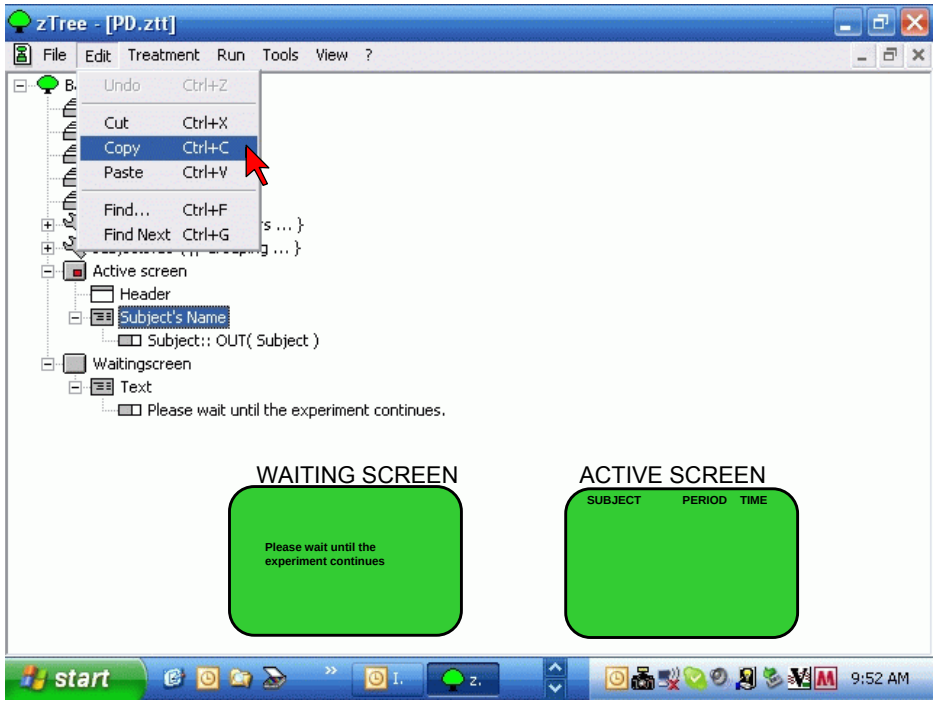
A rectangular part of zLeaf's screen that can show Period & Time

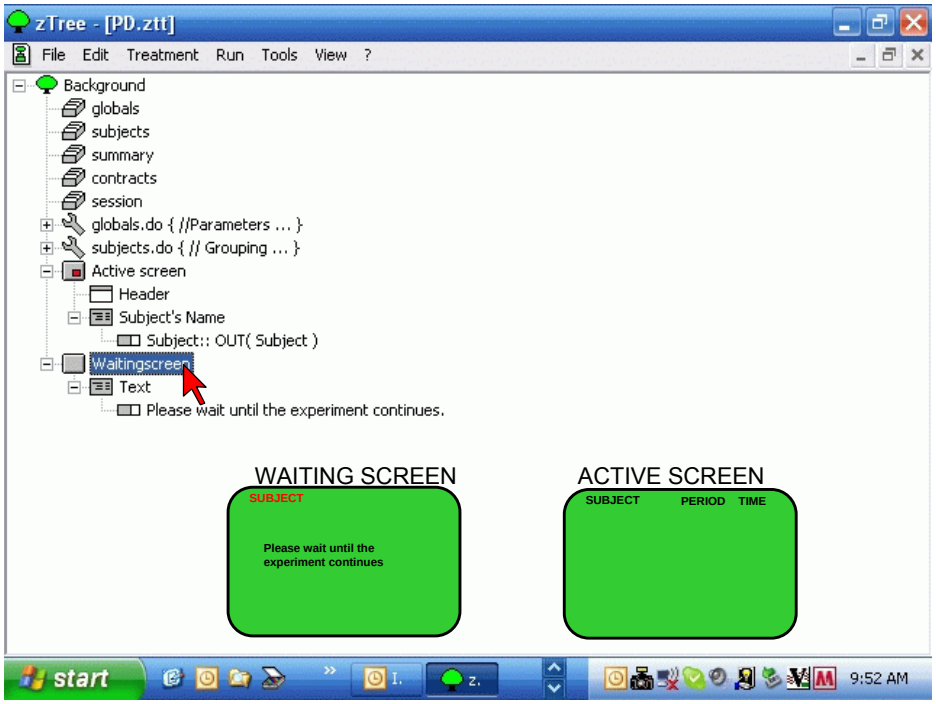


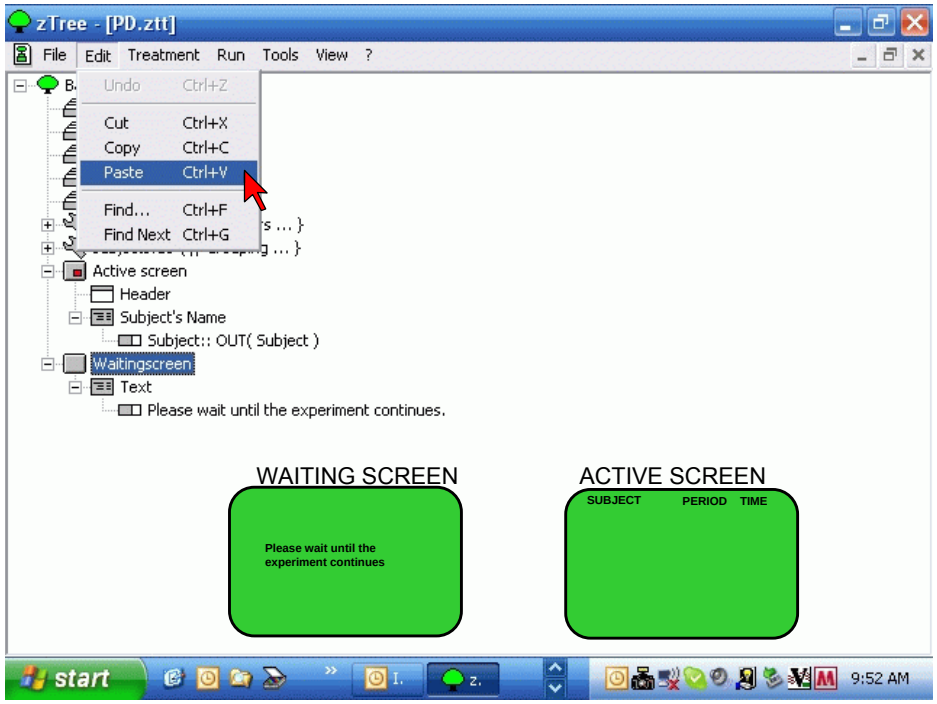


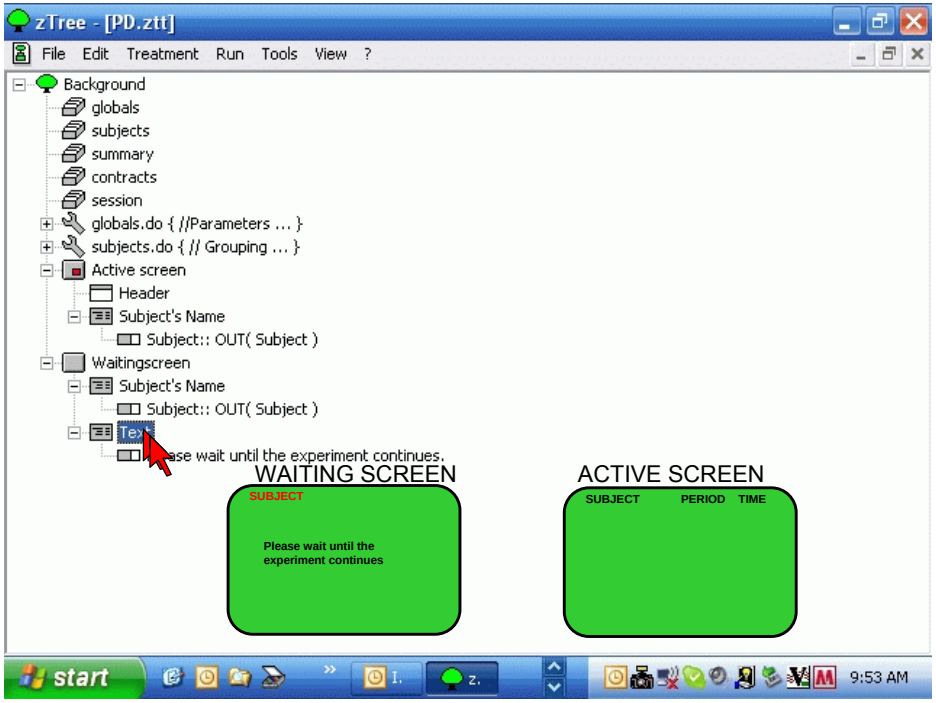


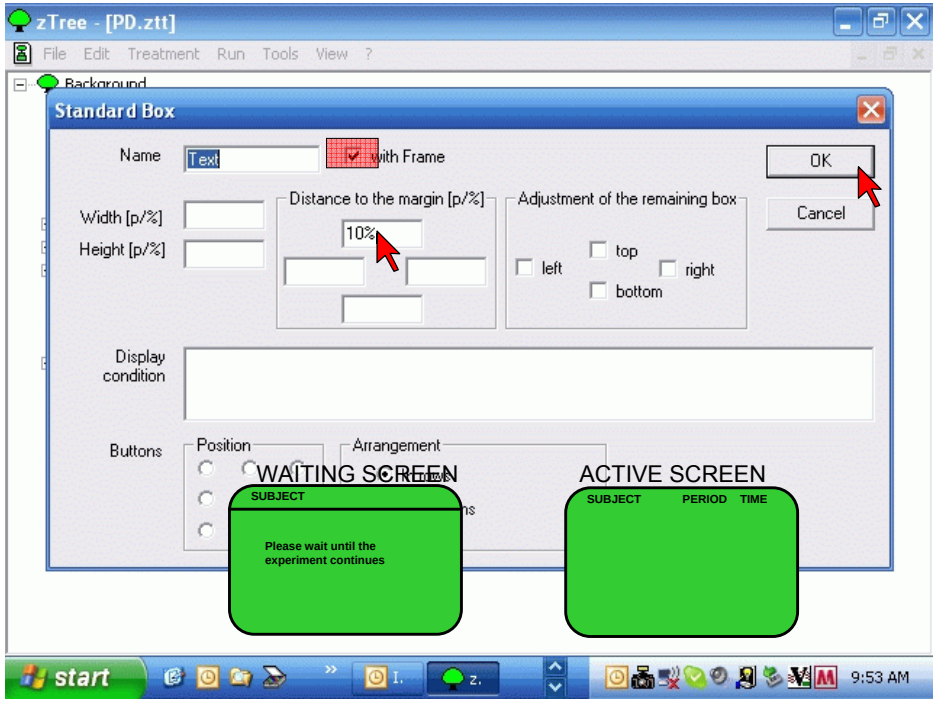

zTree - [
Item
Label
Subject
OK
Variable
Subject
Layout
1
Input
Variable will show rounded to multiples of Layout.
Click on "Input", you will see additional requirements if you were to ask subject to create the variable.
SUBJECT
PERIOD
TIME
start
9:51 AM

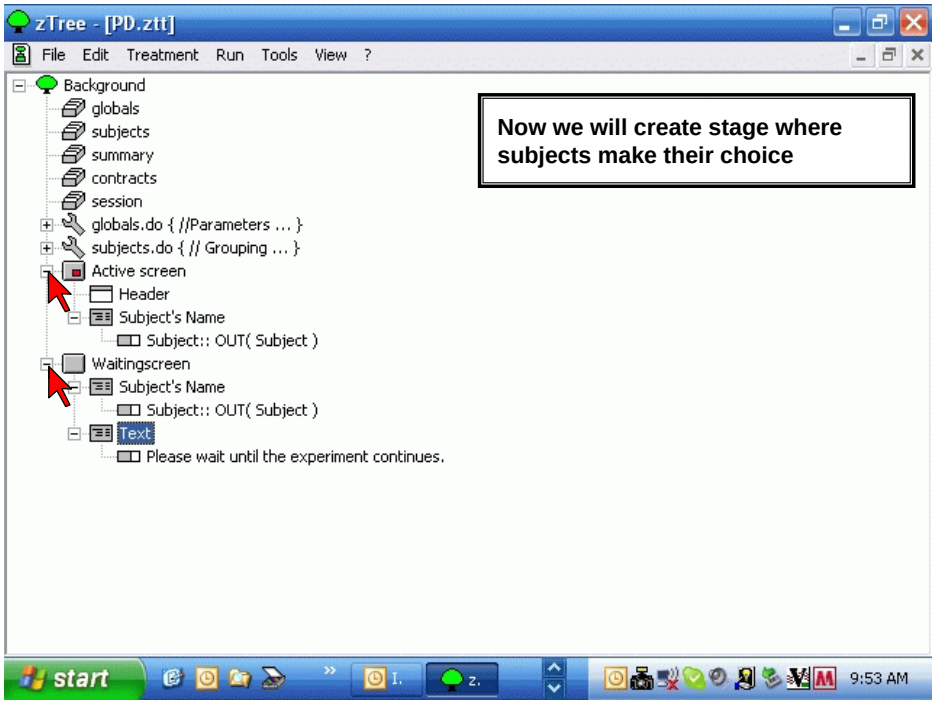

zTree - [PD.ztt]
File
Edit
Treatment
Run
Tools
View
?
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find... Ctrl+F
Find Next Ctrl+G
Active screen
Header
Subject's Name
Subject:: OUT(Subject)
Waitingscreen
Text
Please wait until the experiment continues.
WAITING SCREEN
Please wait until the experiment continues
ACTIVE SCREEN
SUBJECT
PERIOD
TIME
start
9:52 AM

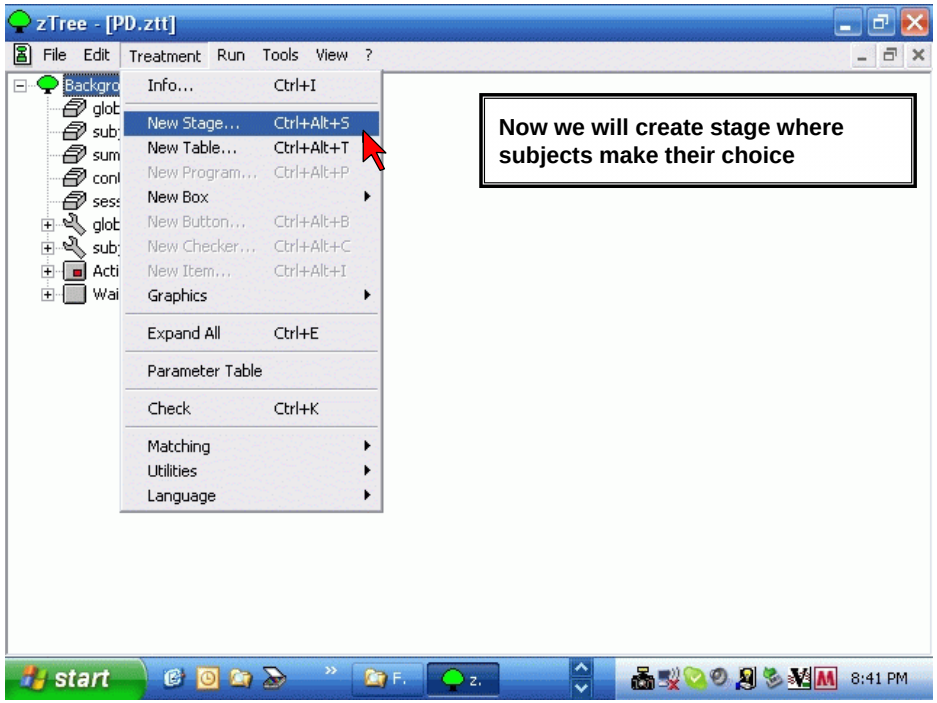

zTree - [PD.ztt]
File
Edit
Treatment
Run
Tools
View
?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Header
Subject's Name
Subject:: OUT(Subject)
Waitingscreen
Text
Please wait until the experiment continues.
WAITING SCREEN
SUBJECT
Please wait until the experiment continues
ACTIVE SCREEN
SUBJECT
PERIOD
TIME
start
9:52 AM

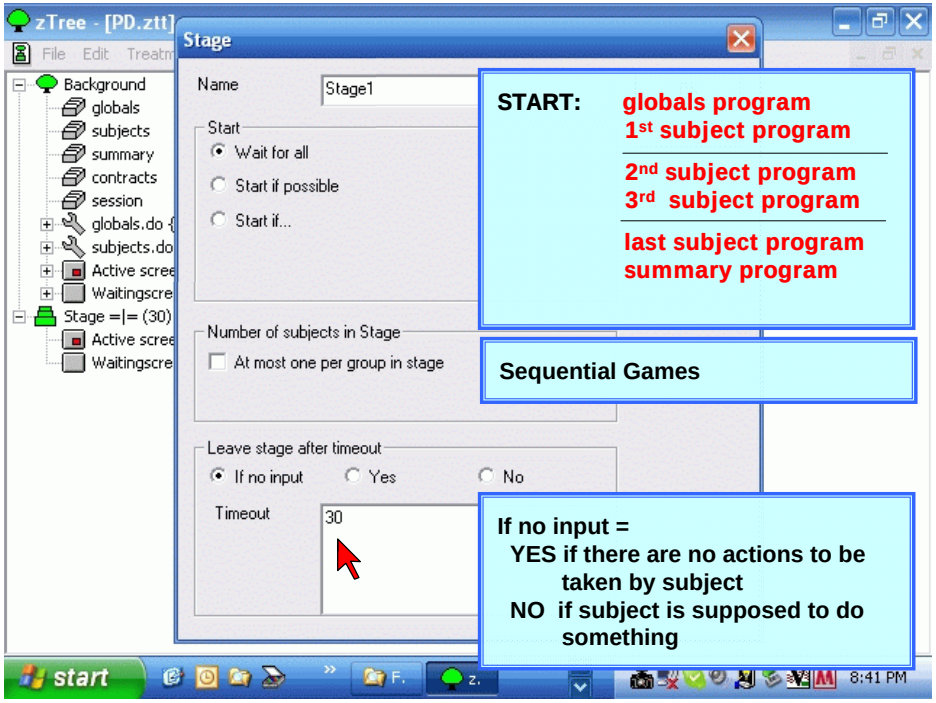

zTree - [PD.ztt]
File
Edit
Treatment
Run
Tools
View
?
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find... Ctrl+F
Find Next Ctrl+G
Active screen
Header
Subject's Name
Subject:: OUT(Subject)
Waitingscreen
Text
Please wait until the experiment continues.
WAITING SCREEN
Please wait until the experiment continues
ACTIVE SCREEN
SUBJECT
PERIOD
TIME
start
9:52 AM

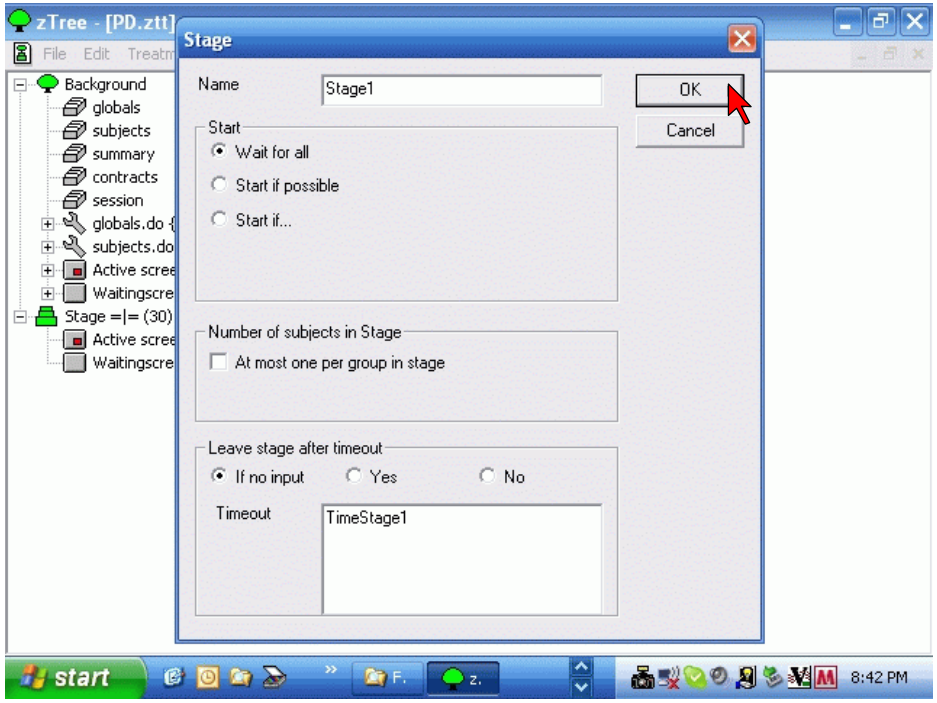

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Header
Subject's Name
Subject:: OUT(Subject)
Waitingscreen
Subject's Name
Subject:: OUT(Subject)
Text
Please wait until the experiment continues.
WAITING SCREEN
SUBJECT
Please wait until the experiment continues
ACTIVE SCREEN
SUBJECT PERIOD TIME
start
9:53 AM

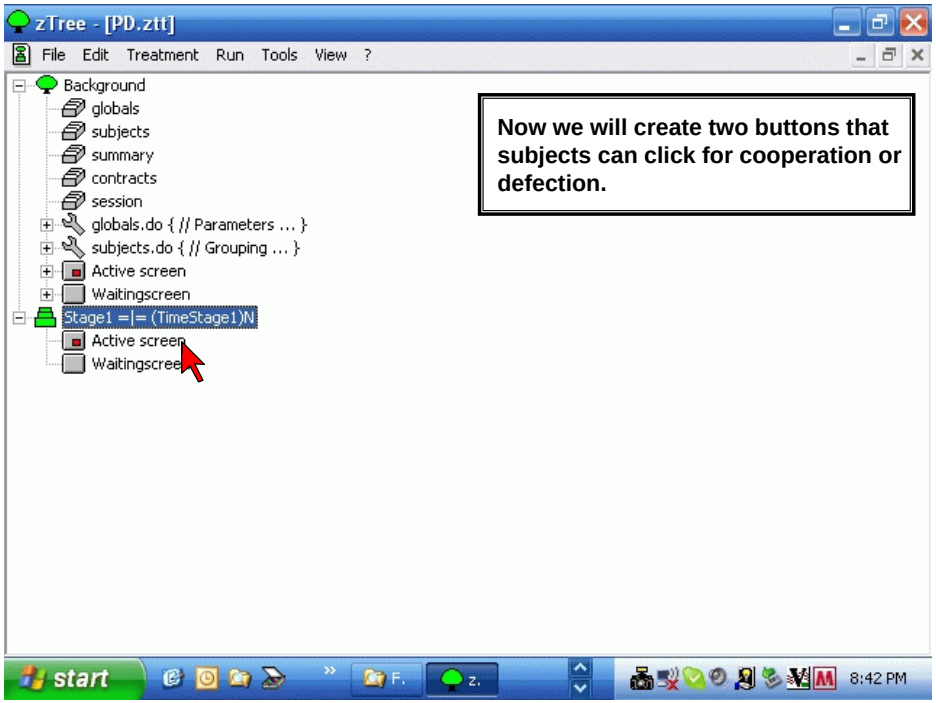

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Standard Box
Name
Text
with Frame
OK
Cancel
Width [p/%]
Height [p/%]
Distance to the margin [p/%]
10%
Adjustment of the remaining box
left
top
right
bottom
Display condition
Buttons
Position
Arrangement
WAITING SCREEN
SUBJECT
Please wait until the experiment continues
ACTIVE SCREEN
SUBJECT PERIOD TIME
start
9:53 AM

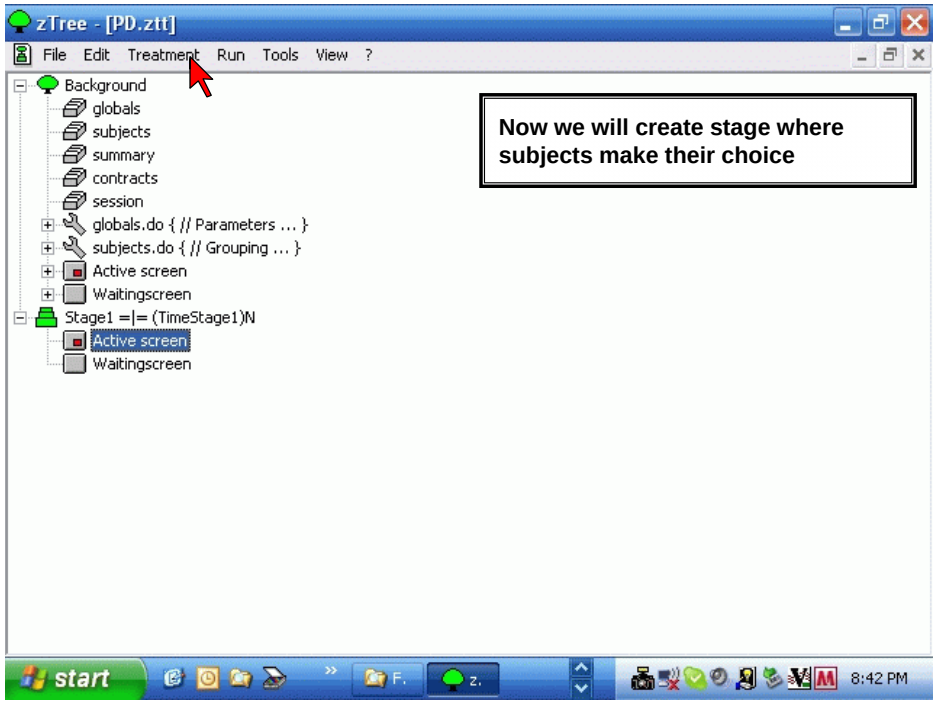

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Header
Subject's Name
Subject:: OUT(Subject)
Waitingscreen
Subject's Name
Subject:: OUT(Subject)
Text
Please wait until the experiment continues.
Now we will create stage where subjects make their choice
start
9:53 AM

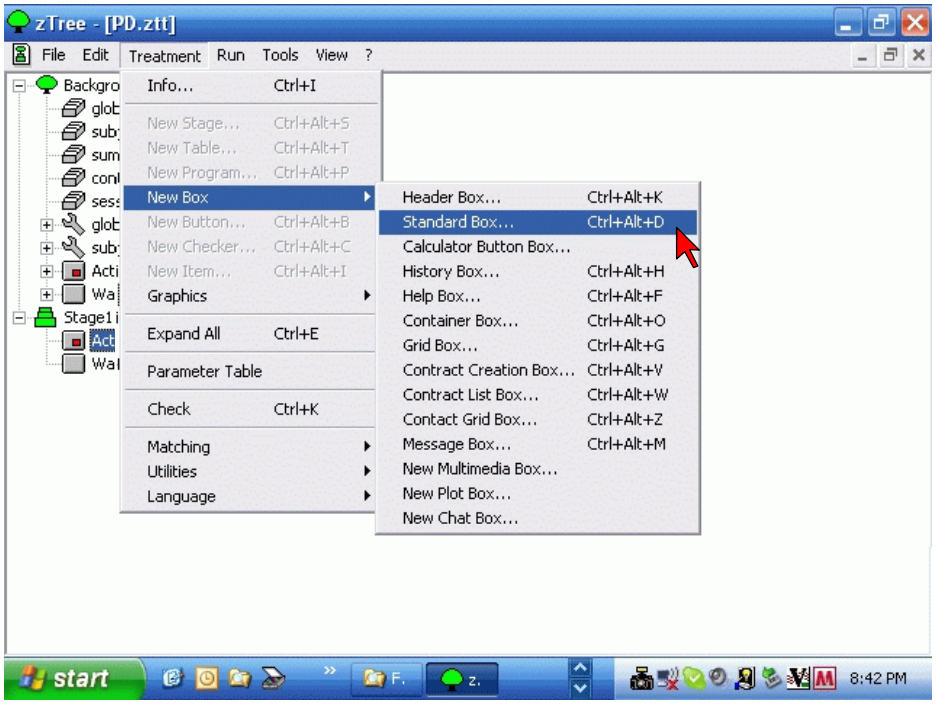

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Now we will create stage where subjects make their choice
start
8:41 PM

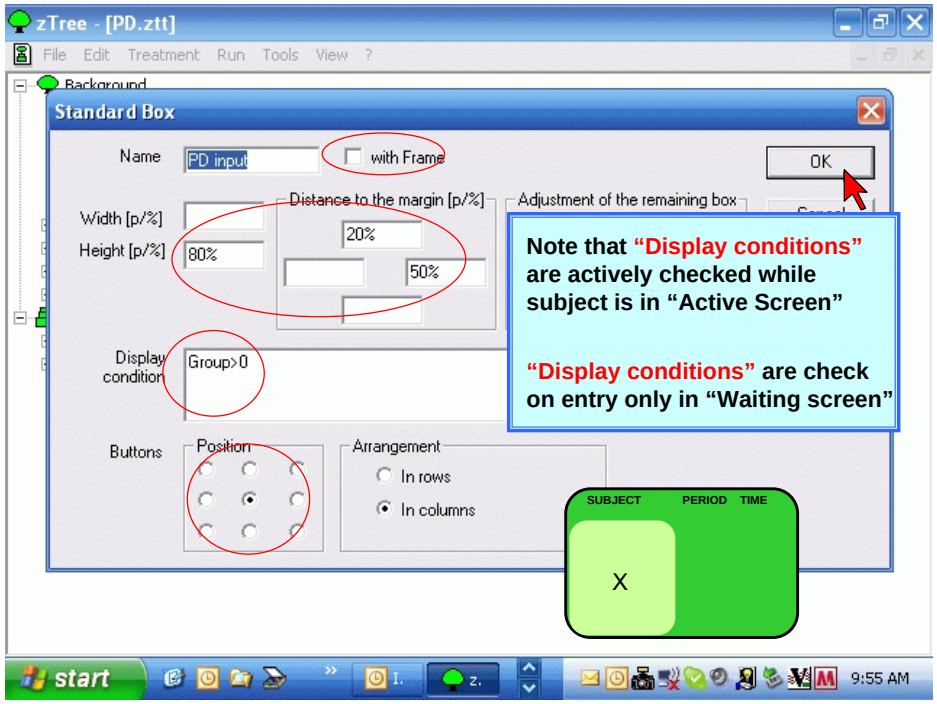

zTree - [PD.ztt]
Stage
Name
Stage1
Start
Wait for all
Start if possible
Start if...
Number of subjects in Stage
At most one per group in stage
Leave stage after timeout
If no input
Yes
No
Timeout
30
Background
globals
subjects
summary
contracts
session
Stage =|= (30)
start
8:41 PM
START:
globals program
1st subject program
2nd subject program
3rd subject program
last subject program
summary program
Sequential Games
If no input =
YES if there are no actions to be taken by subject
NO if subject is supposed to do something

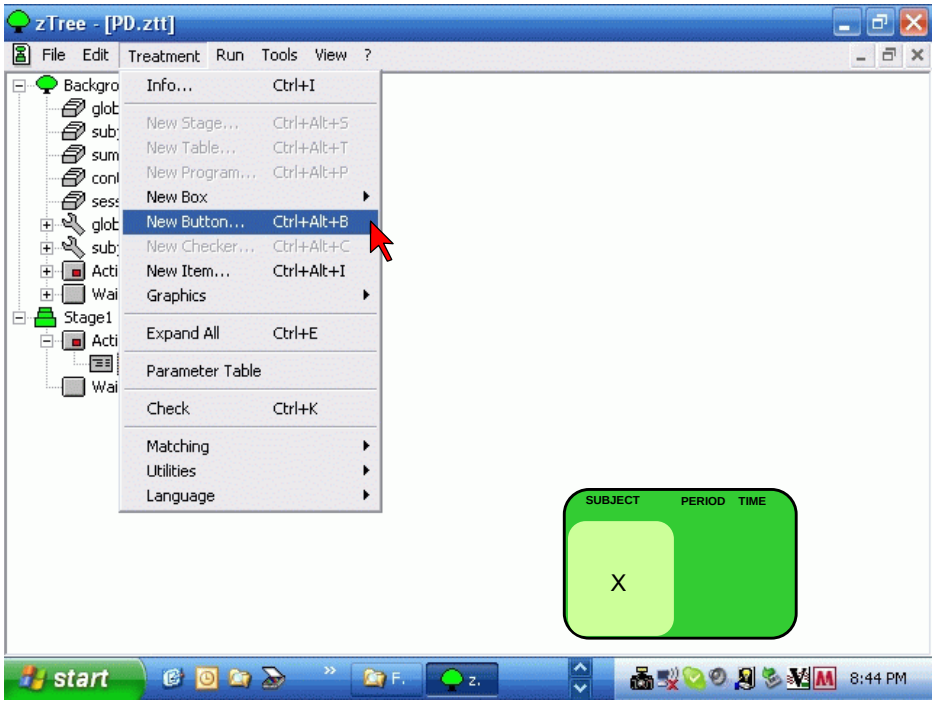

zTree - [PD.ztt]
Stage
Name
Stage1
OK
Cancel
Start
Wait for all
Start if possible
Start if...
Number of subjects in Stage
At most one per group in stage
Leave stage after timeout
If no input
Yes
No
Timeout
TimeStage1
Background
globals
subjects
summary
contracts
session
Stage =|= (30)
start
8:42 PM

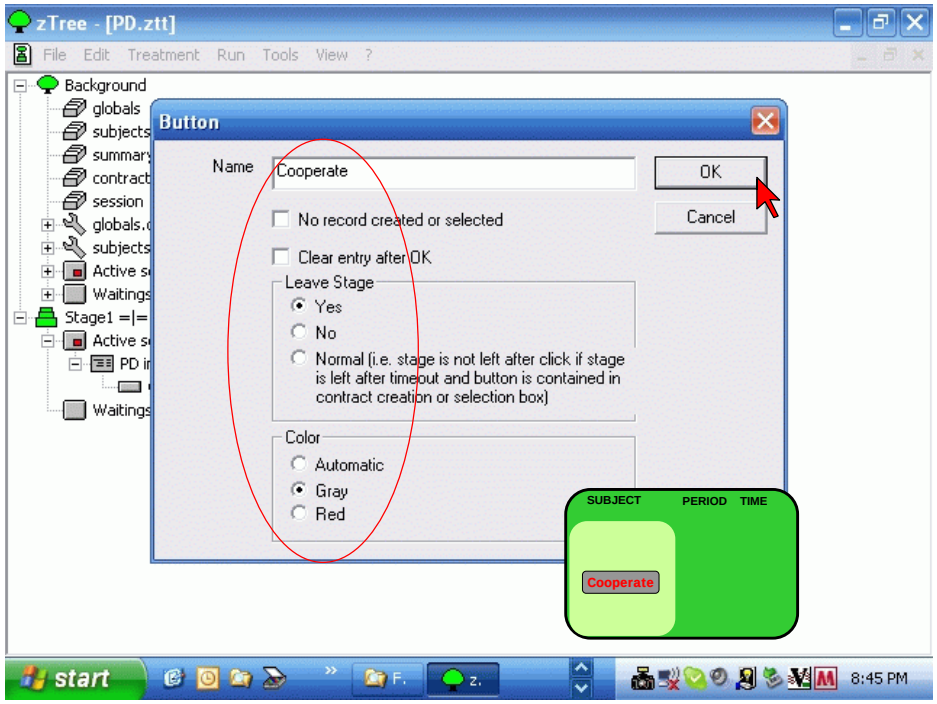

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { // Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
Active screen
Waitingscreen
start
8:42 PM
Now we will create two buttons that subjects can click for cooperation or defection.

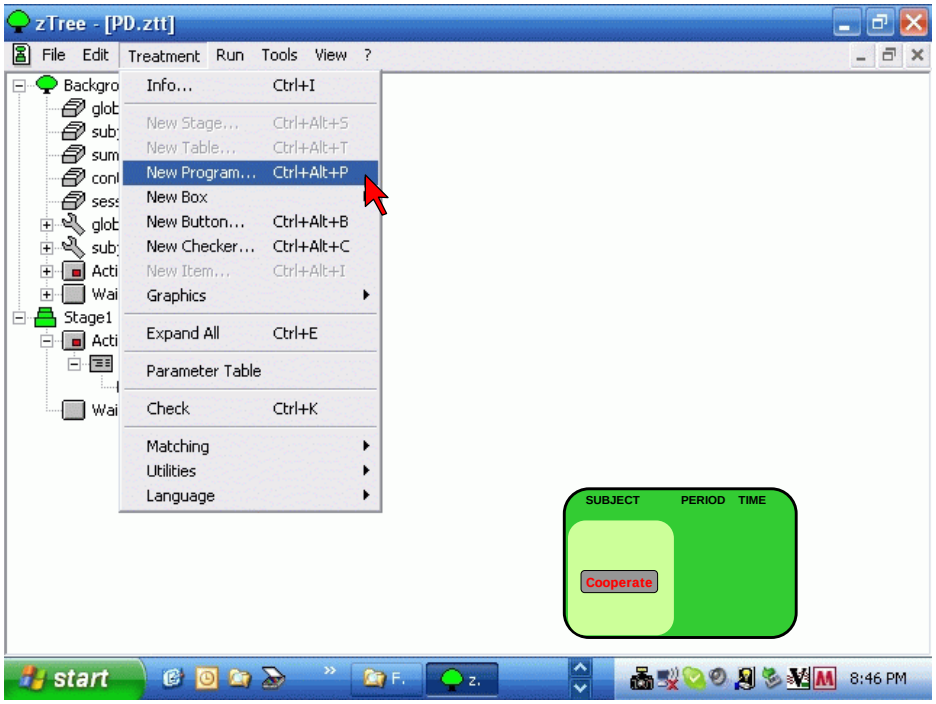

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { // Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
Active screen
Waitingscreen
start
8:42 PM
Now we will create stage where subjects make their choice

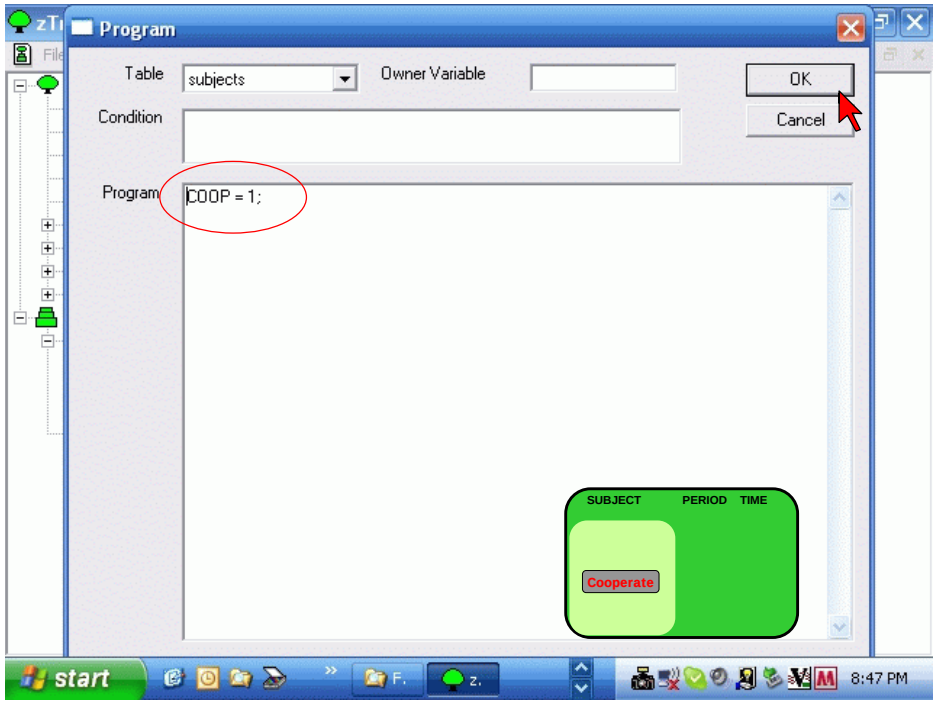

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Header Box... Ctrl+Alt+K
Standard Box... Ctrl+Alt+D
Calculator Button Box...
History Box... Ctrl+Alt+H
Help Box... Ctrl+Alt+F
Container Box... Ctrl+Alt+O
Grid Box... Ctrl+Alt+G
Contract Creation Box... Ctrl+Alt+V
Contract List Box... Ctrl+Alt+W
Contact Grid Box... Ctrl+Alt+Z
Message Box... Ctrl+Alt+M
New Multimedia Box...
New Plot Box...
New Chat Box...
start
8:42 PM

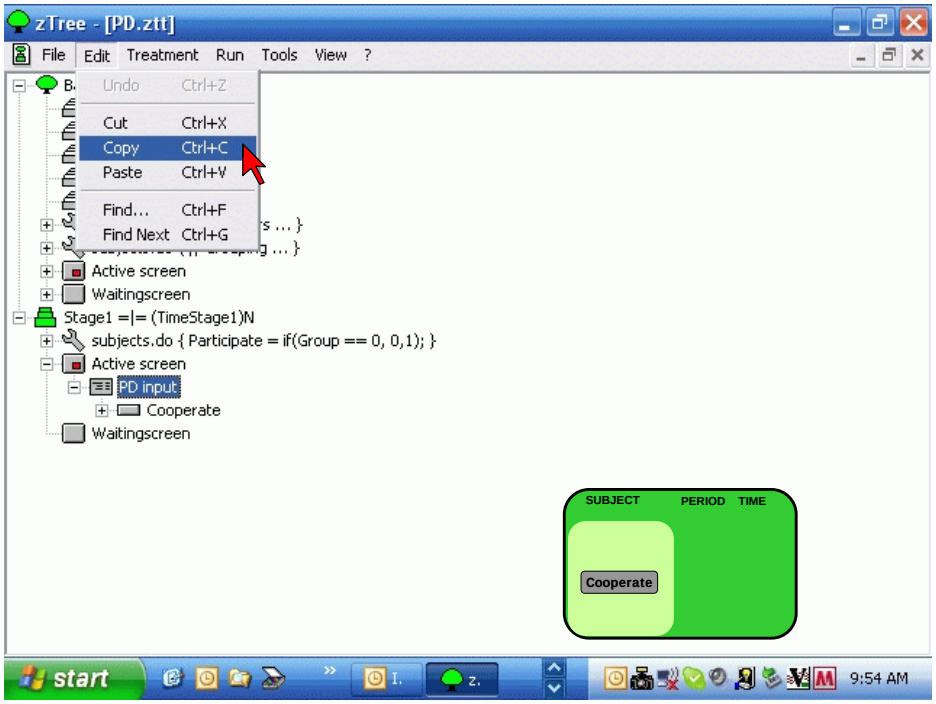

zTree - [PD.ztt]
Standard Box
Name
PD input
with Frame
OK
Width [p/%]
Height [p/%]
80%
Distance to the margin [p/%]
20%
50%
Adjustment of the remaining box
Display condition
Group>0
Buttons
Position
Arrangement
In rows
In columns
SUBJECT PERIOD TIME
X
Note that "Display conditions" are actively checked while subject is in "Active Screen"
"Display conditions" are check on entry only in "Waiting screen"
start
9:55 AM

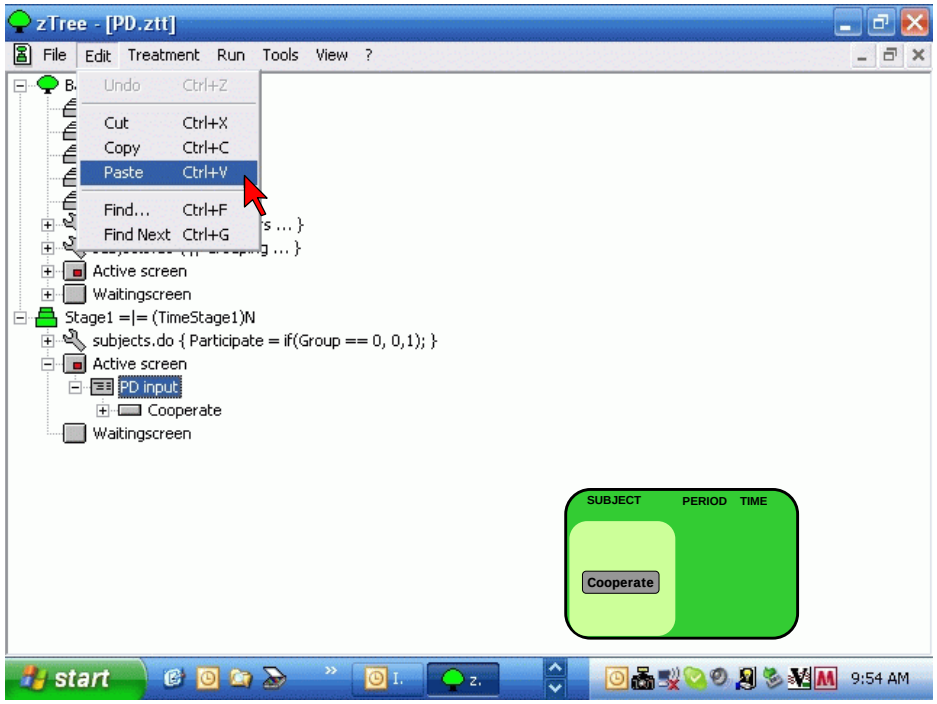

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
SUBJECT PERIOD TIME
X
start
8:44 PM

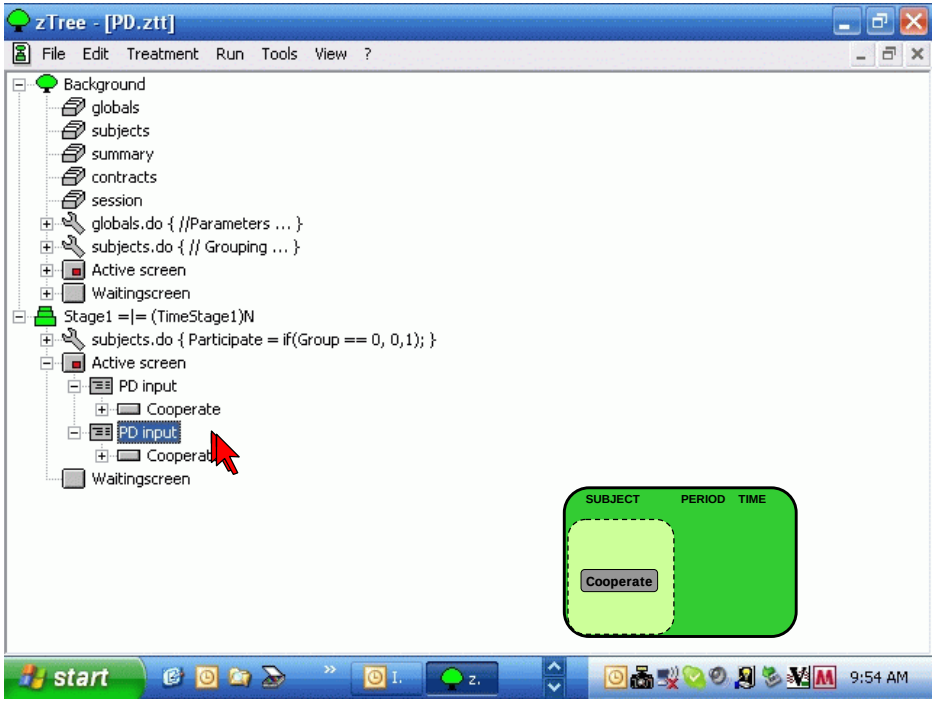

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
Button
Name
Cooperate
OK
Cancel
No record created or selected
Clear entry after OK
Leave Stage
Yes
No
Normal (i.e. stage is not left after click if stage is left after timeout and button is contained in contract creation or selection box)
Color
Automatic
Gray
Red
SUBJECT PERIOD TIME
Cooperate
start
8:45 PM

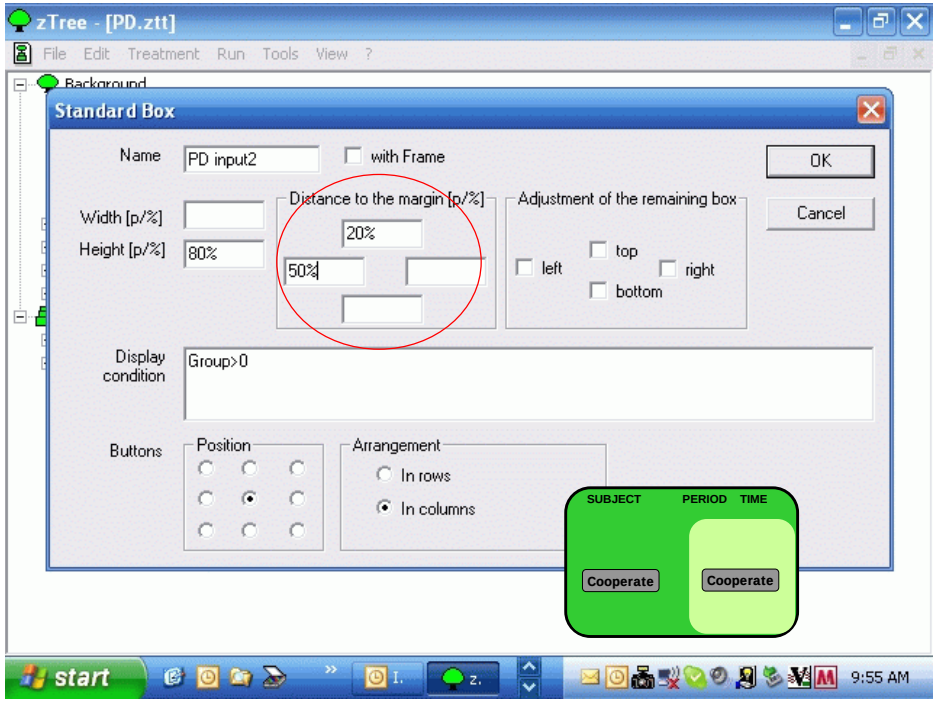

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Stage1
SUBJECT PERIOD TIME
Cooperate
start
8:46 PM

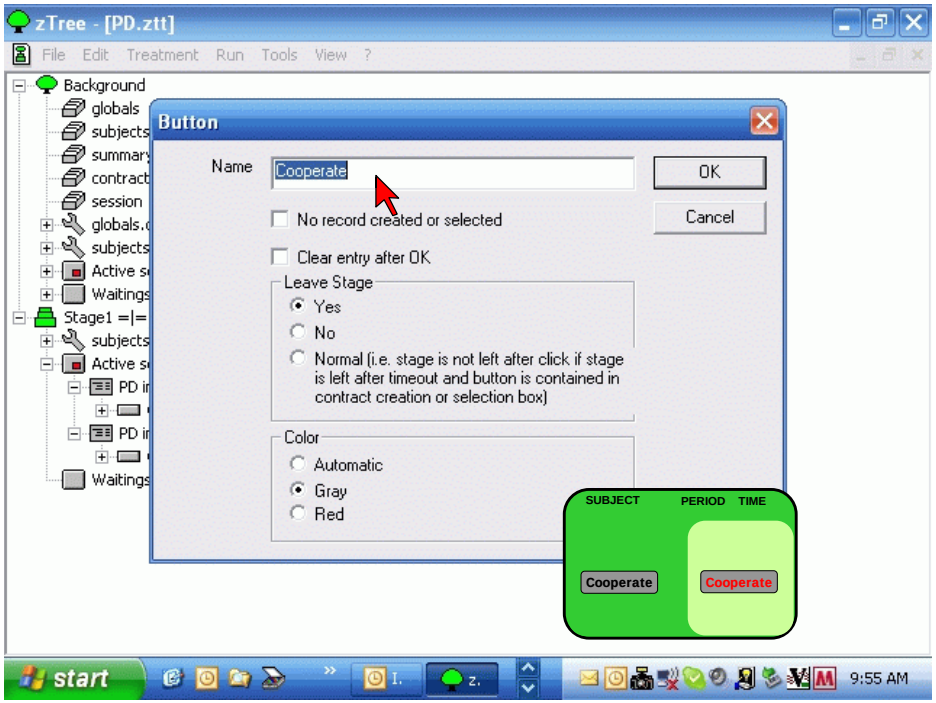

Program
Table subjects
Owner Variable
OK
Cancel
Condition
Program
COOP = 1;
SUBJECT PERIOD TIME
Cooperate
start
8:47 PM

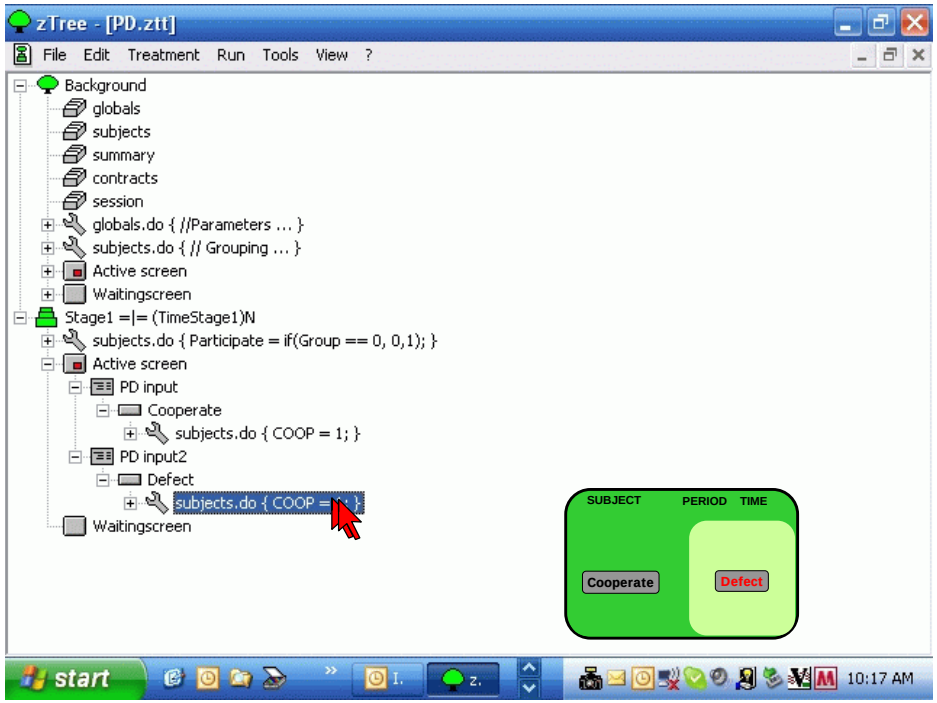

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find... Ctrl+F
Find Next Ctrl+G
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
subjects.do { Participate = if(Group == 0, 0,1); }
Active screen
PD input
Cooperate
Waitingscreen
SUBJECT PERIOD TIME
Cooperate
start
9:54 AM

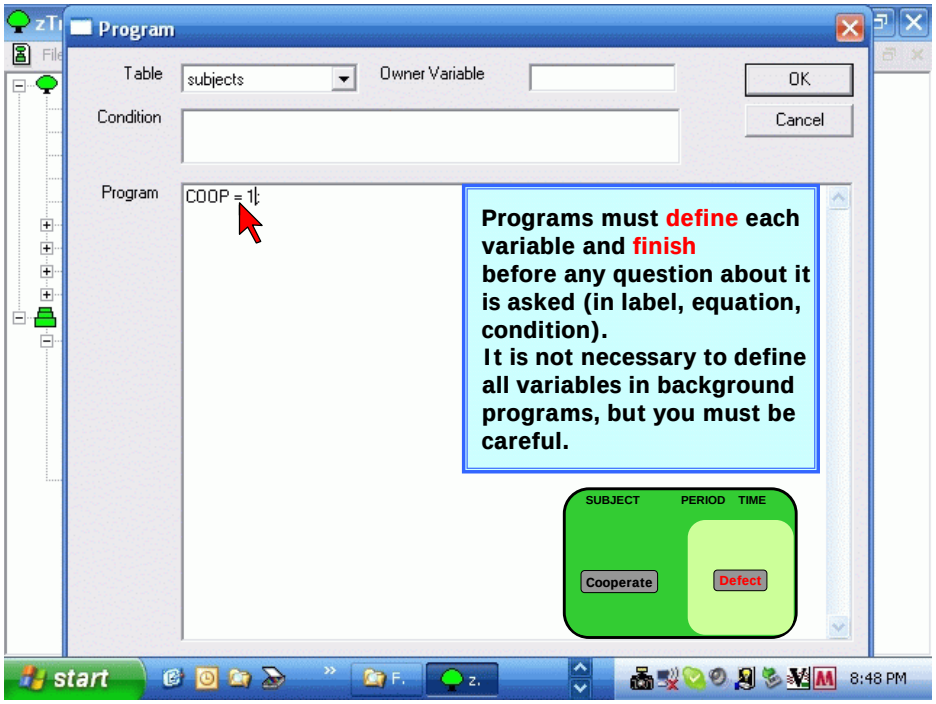

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Undo Ctrl+Z
Cut Ctrl+X
Copy Ctrl+C
Paste Ctrl+V
Find... Ctrl+F
Find Next Ctrl+G
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
subjects.do { Participate = if(Group == 0, 0,1); }
Active screen
PD input
Cooperate
Waitingscreen
SUBJECT PERIOD TIME
Cooperate
start
9:54 AM

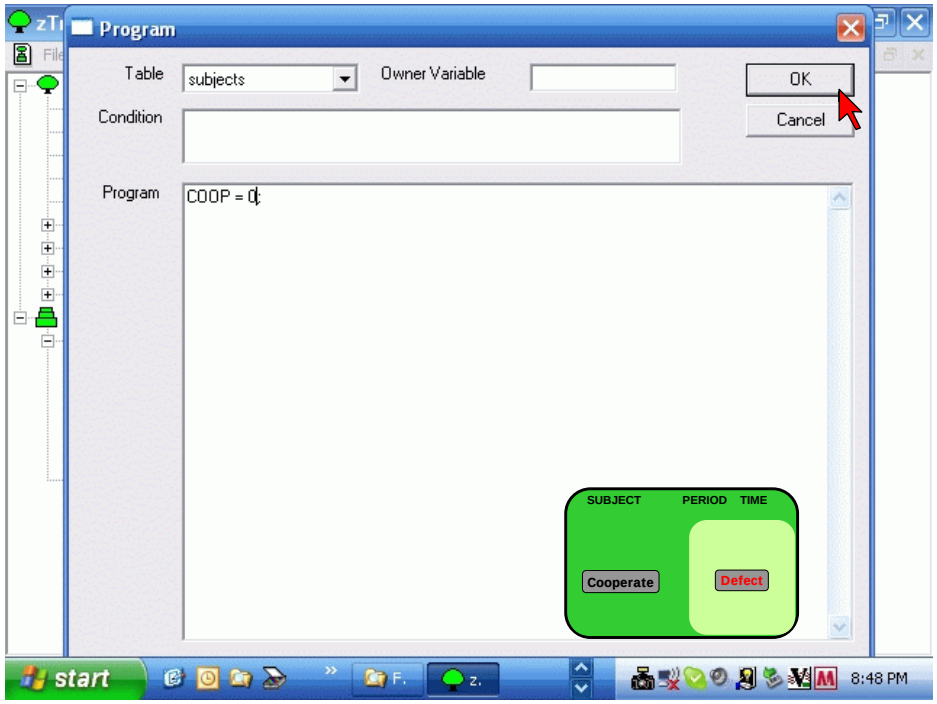

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
subjects.do { Participate = if(Group == 0, 0,1); }
Active screen
PD input
Cooperate
PD input
Cooperate
Waitingscreen
SUBJECT PERIOD TIME
Cooperate
start
9:54 AM

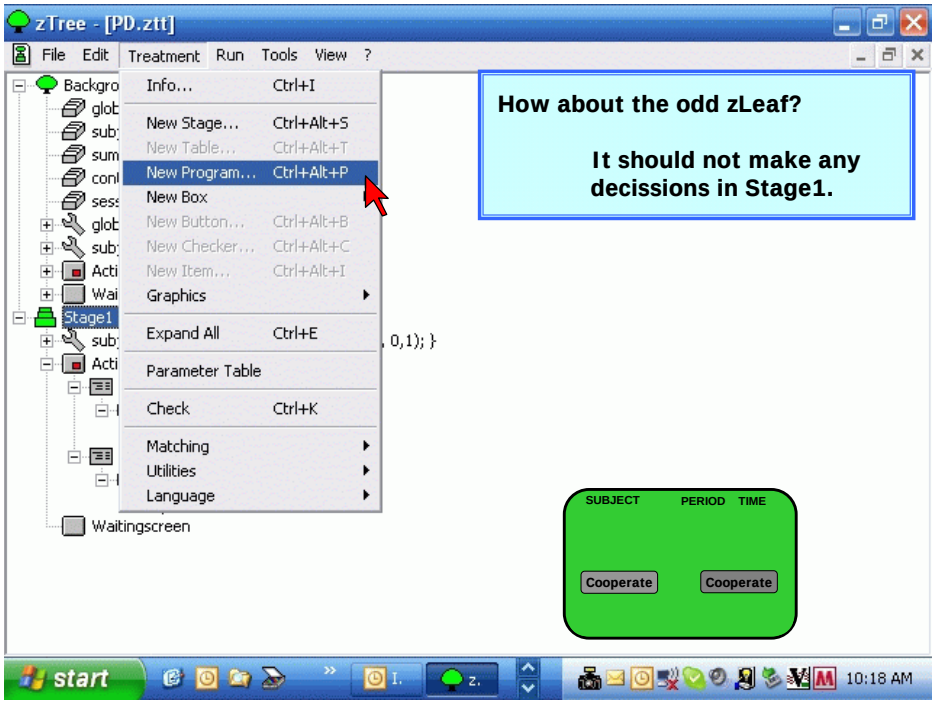

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Standard Box
Name
PD input2
with Frame
OK
Cancel
Width [p/%]
Height [p/%]
80%
Distance to the margin [p/%]
20%
50%
Adjustment of the remaining box
left
top
right
bottom
Display condition
Group>0
Buttons
Position
Arrangement
In rows
In columns
SUBJECT PERIOD TIME
Cooperate
Cooperate
start
9:55 AM

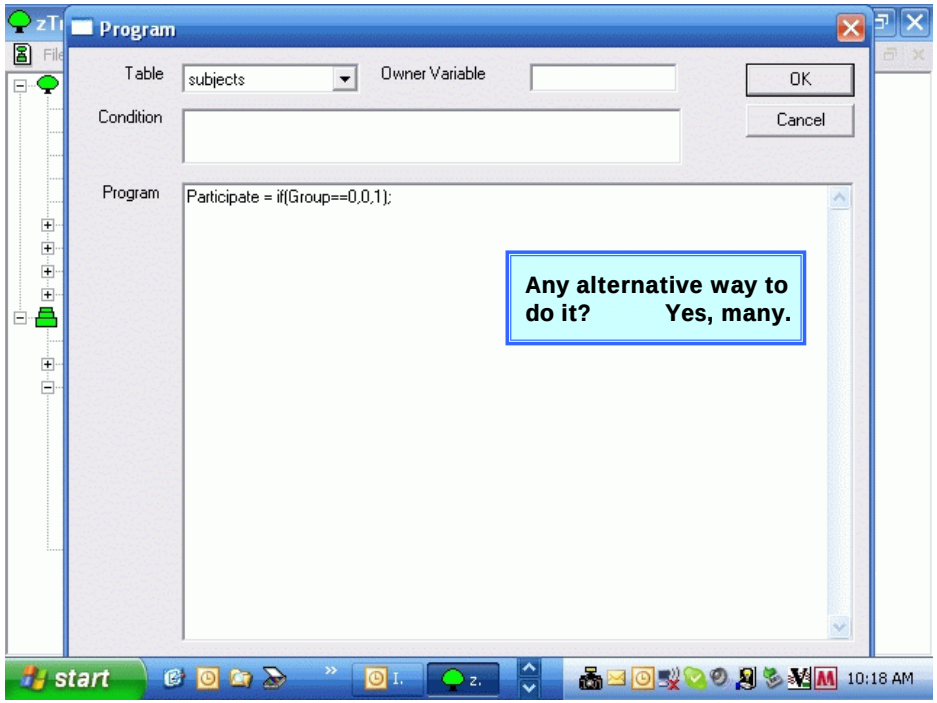

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Button
Name
Cooperate
OK
Cancel
No record created or selected
Clear entry after OK
Leave Stage
Yes
No
Normal (i.e. stage is not left after click if stage is left after timeout and button is contained in contract creation or selection box)
Color
Automatic
Gray
Red
SUBJECT PERIOD TIME
Cooperate
Cooperate
start
9:55 AM

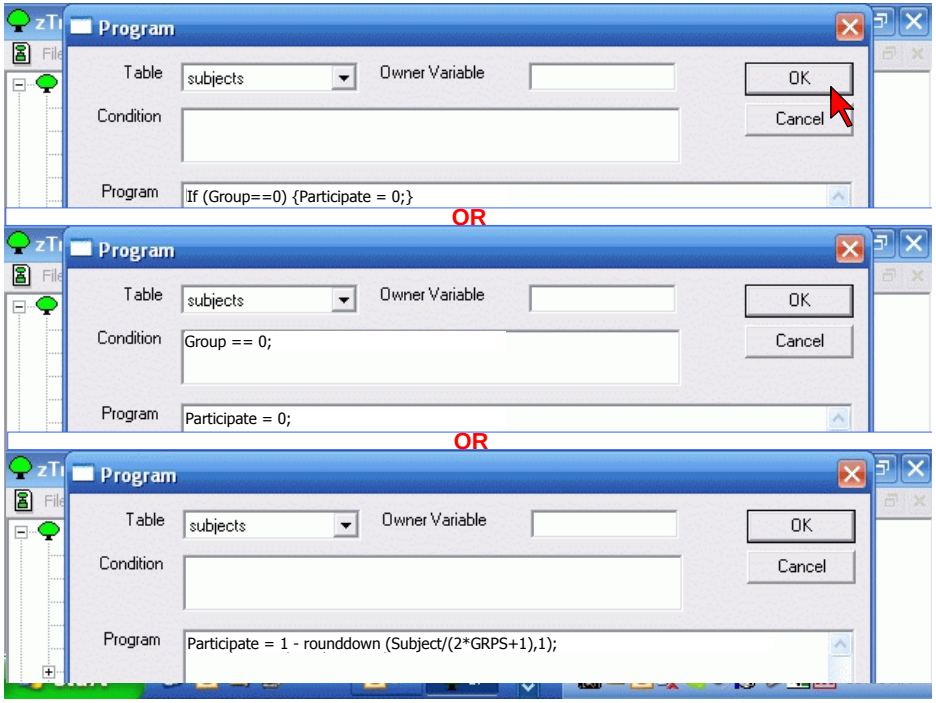

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
subjects.do { Participate = if(Group == 0, 0,1); }
Active screen
PD input
Cooperate
subjects.do { COOP = 1; }
PD input2
Defect
subjects.do { COOP = 0; }
Waitingscreen
SUBJECT PERIOD TIME
Cooperate
Defect
start
10:17 AM

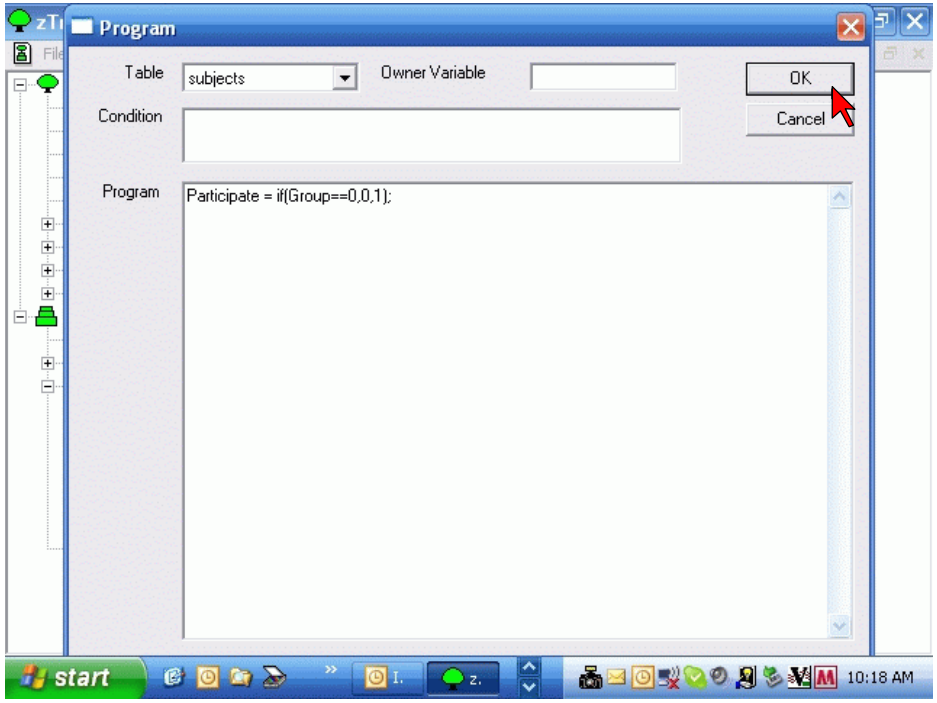

Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
COOP = 1;
Programs must define each variable and finish before any question about it is asked (in label, equation, condition).
It is not necessary to define all variables in background programs, but you must be careful.
SUBJECT
PERIOD
TIME
Cooperate
Defect
start
8:48 PM

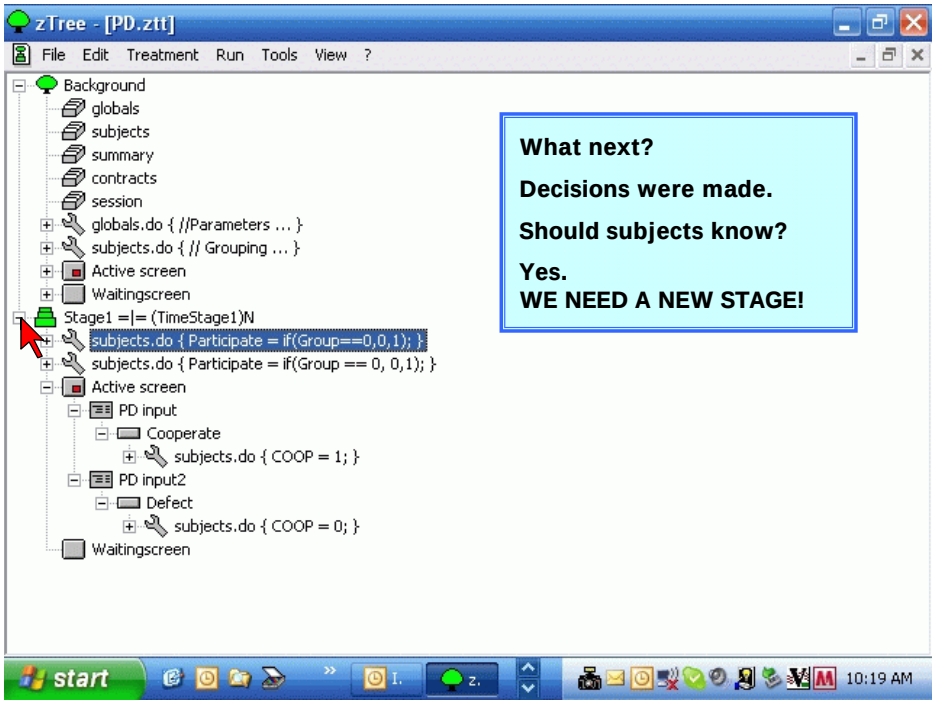

Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
COOP = 0;
SUBJECT
PERIOD
TIME
Cooperate
Defect
start
8:48 PM

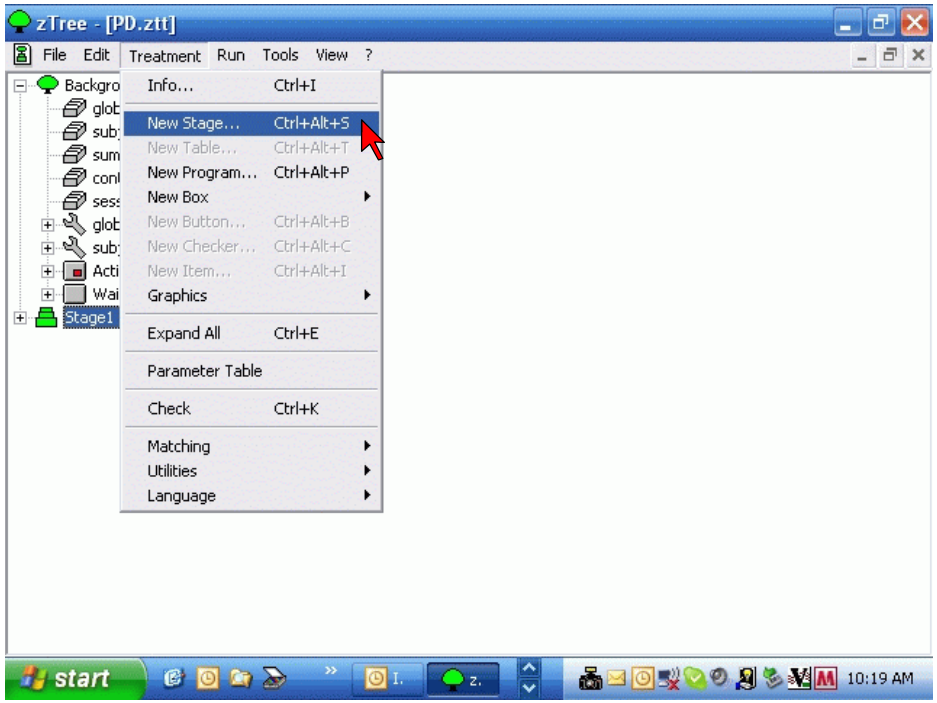

zTree - [PD.ztt]
File
Edit
Treatment
Run
Tools
View
?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Stage1
Waitingscreen
How about the odd zLeaf?
It should not make any decissions in Stage1.
SUBJECT
PERIOD
TIME
Cooperate
Cooperate
start
10:18 AM

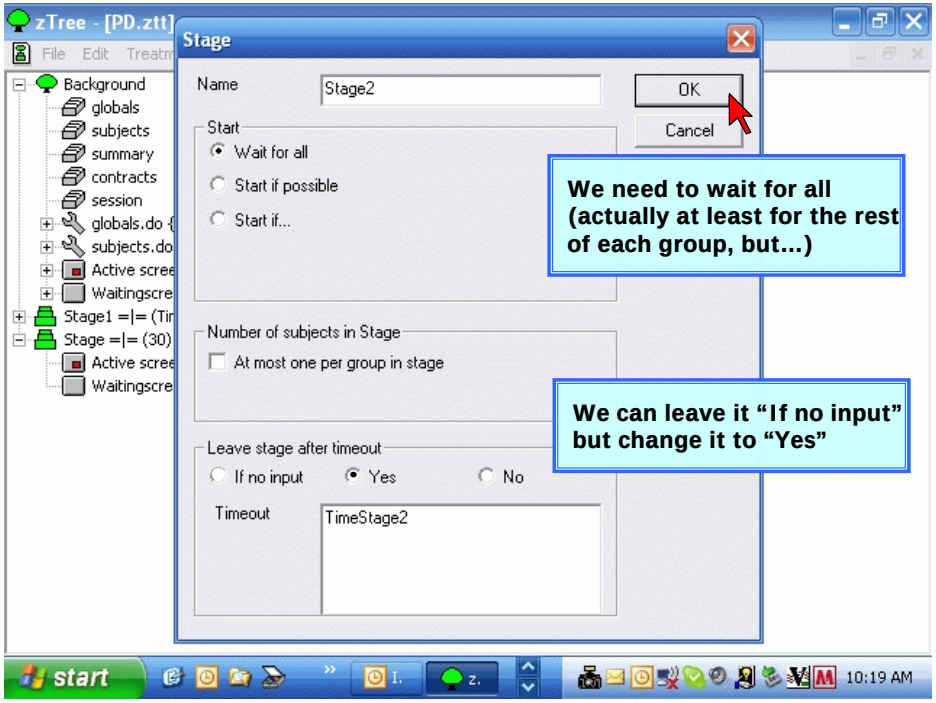

Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
Participate = if(Group==0,0,1);
Any alternative way to do it? Yes, many.
start
10:18 AM

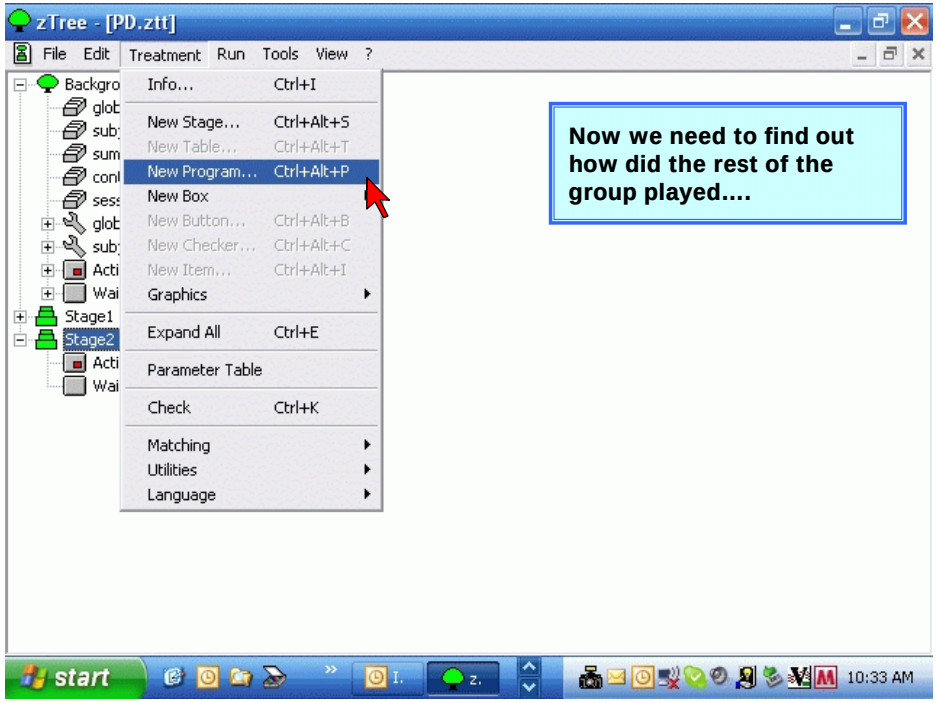

Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
If (Group==0) {Participate = 0;}
OR
Program
Table
subjects
Owner Variable
OK
Condition
Group == 0;
Cancel
Program
Participate = 0;
OR
Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
Participate = 1 - rounddown (Subject/(2*GRPS+1),1);

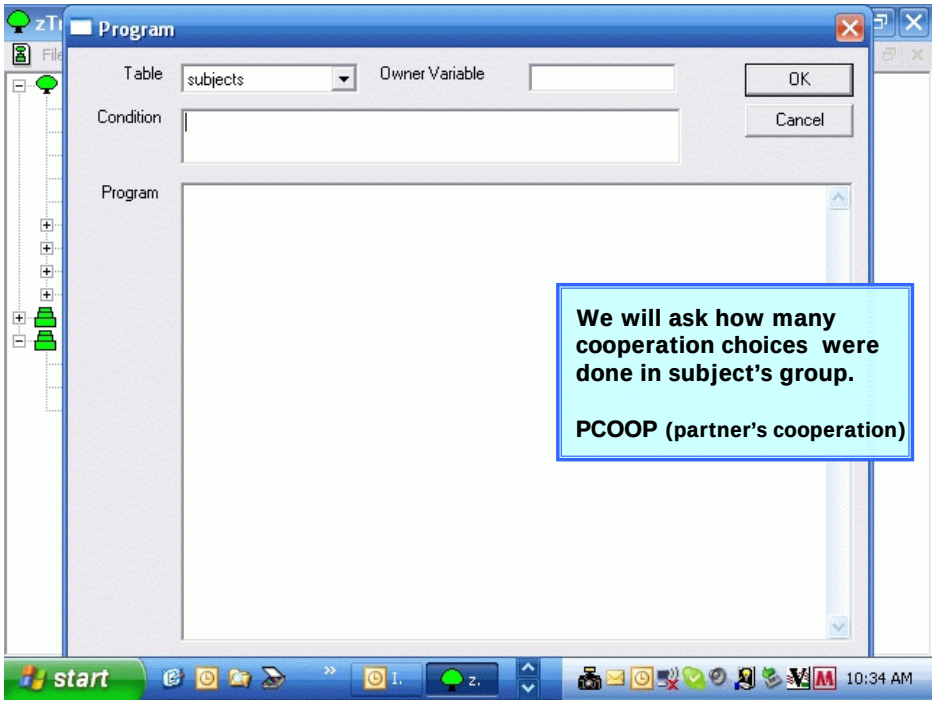

Program
Table
subjects
Owner Variable
OK
Condition
Cancel
Program
Participate = if(Group==0,0,1);
start
10:18 AM

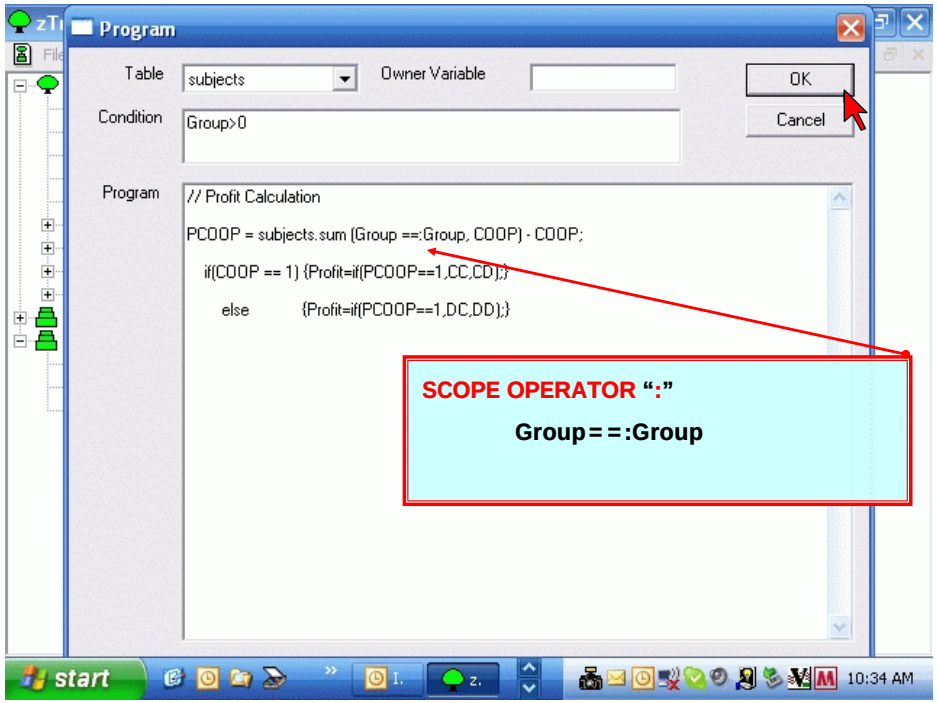

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
subjects.do { Participate = if(Group==0,0,1); }
subjects.do { Participate = if(Group == 0, 0,1); }
Active screen
PD input
Cooperate
subjects.do { COOP = 1; }
PD input2
Defect
subjects.do { COOP = 0; }
Waitingscreen
What next?
Decisions were made.
Should subjects know?
Yes.
WE NEED A NEW STAGE!
start
10:19 AM

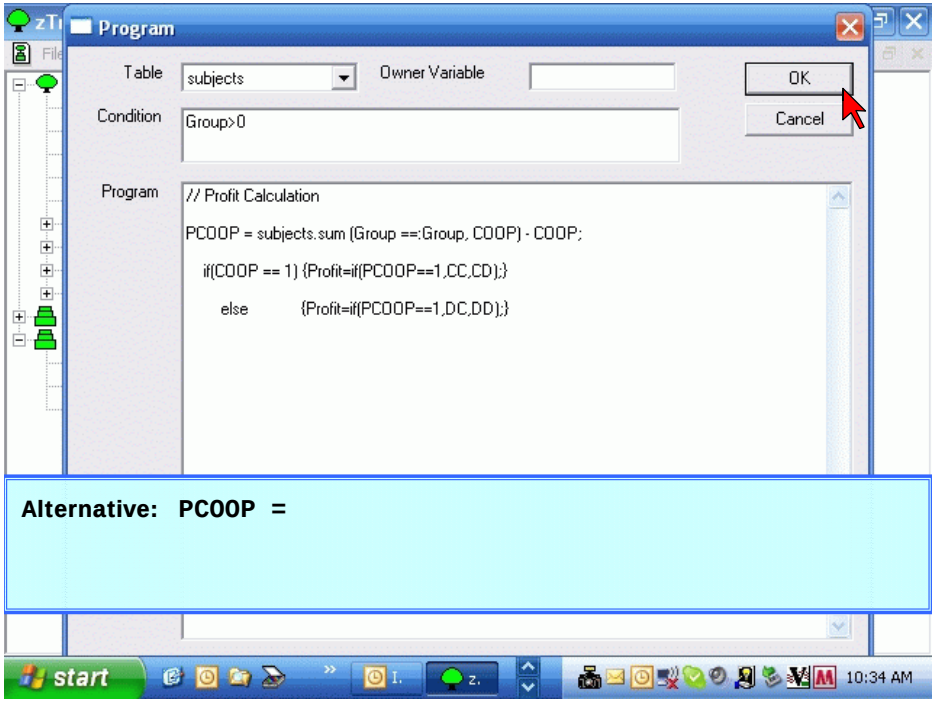

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
start
10:19 AM


zTree - [PD.ztt]
Stage
Name
Stage2
OK
Cancel
Start
Wait for all
Start if possible
Start if...
Number of subjects in Stage
At most one per group in stage
Leave stage after timeout
If no input
Yes
No
Timeout
TimeStage2
We need to wait for all (actually at least for the rest of each group, but...)
We can leave it "If no input" but change it to "Yes"
start
10:19 AM

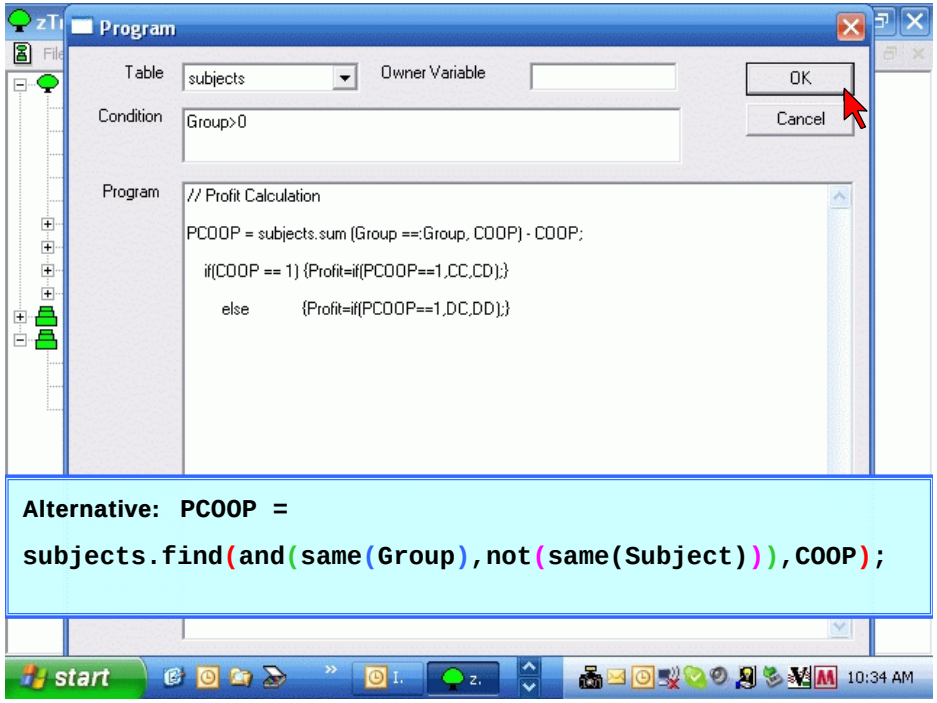

zTree - [PD.ztt]
File
Edit
Treatment
Run
Tools
View
?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Now we need to find out how did the rest of the group played....
start
10:33 AM


Program
Table
subjects
Owner Variable
OK
Cancel
Condition
Program
We will ask how many cooperation choices were done in subject's group.
PCOOP (partner's cooperation)
start
10:34 AM


Program
Table
subjects
Owner Variable
OK
Cancel
Condition
Group>0
Program
// Profit Calculation
PCOOP = subjects.sum (Group ==:Group, COOP) - COOP;
if(COOP == 1) {Profit=if(PCOOP==1,CC,CD);}
else {Profit=if(PCOOP==1,DC,DD);}
SCOPE OPERATOR ":"
Group==:Group
start
10:34 AM

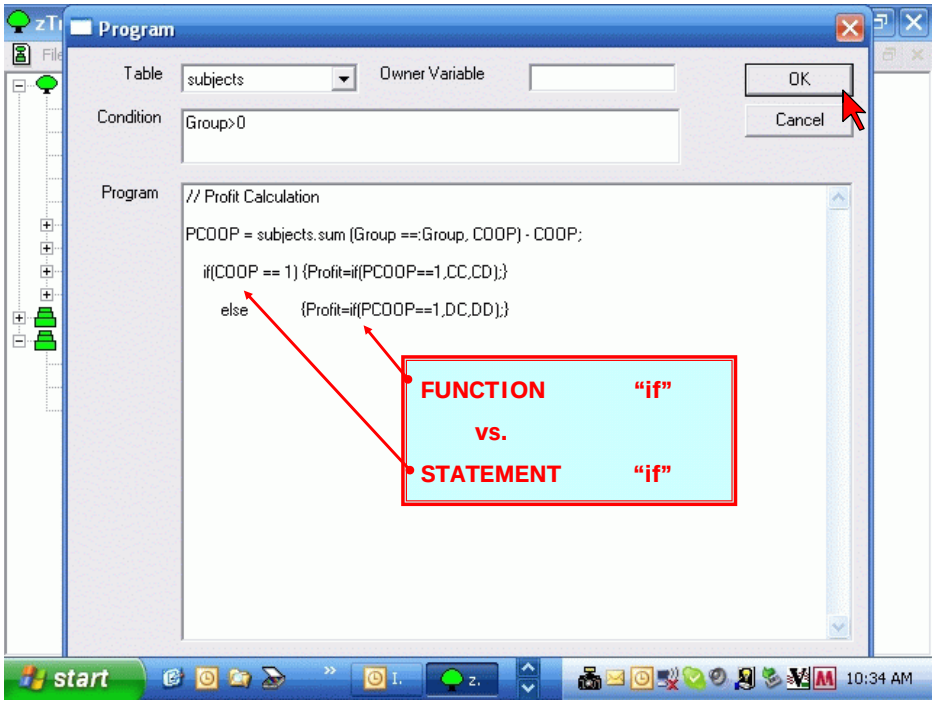


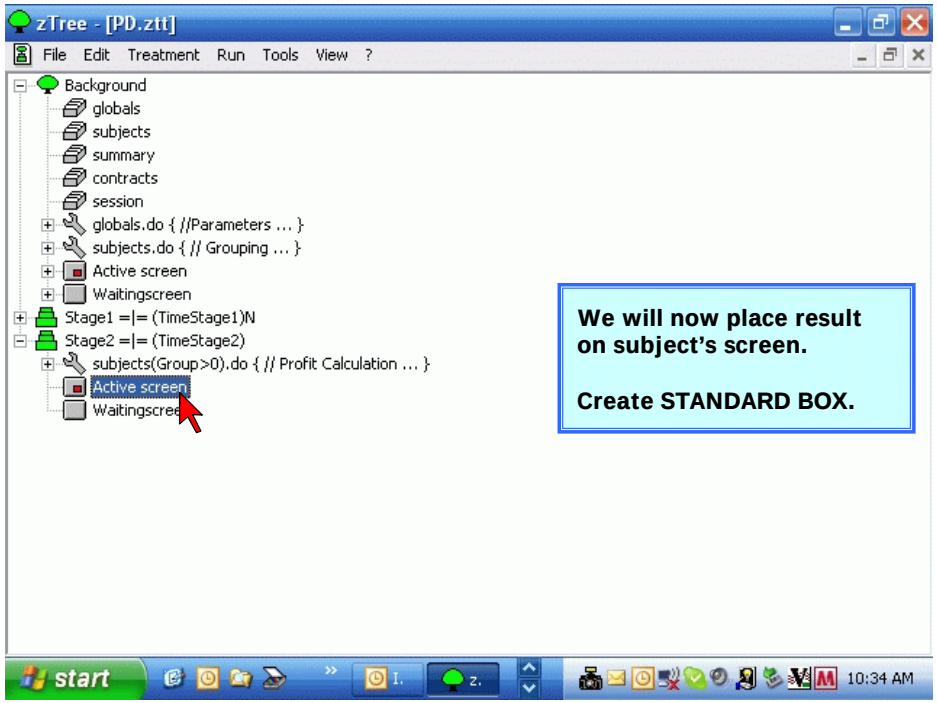
Alternative: PCOOP =



Alternative: PCOOP =

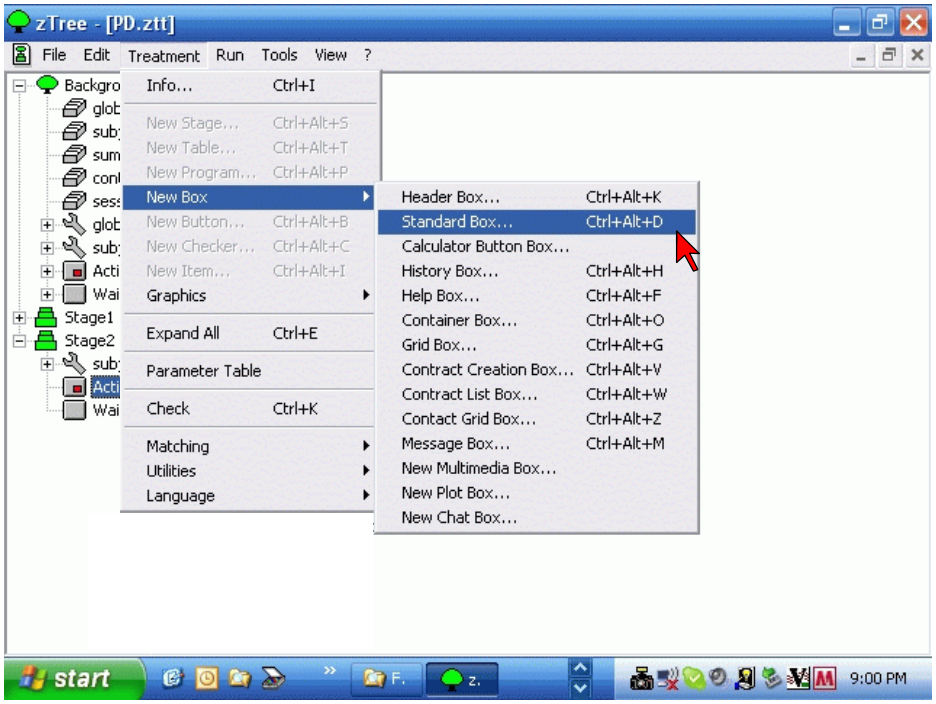
```
subjects.find(and(same(Group),not(same(Subject))),COOP);
```

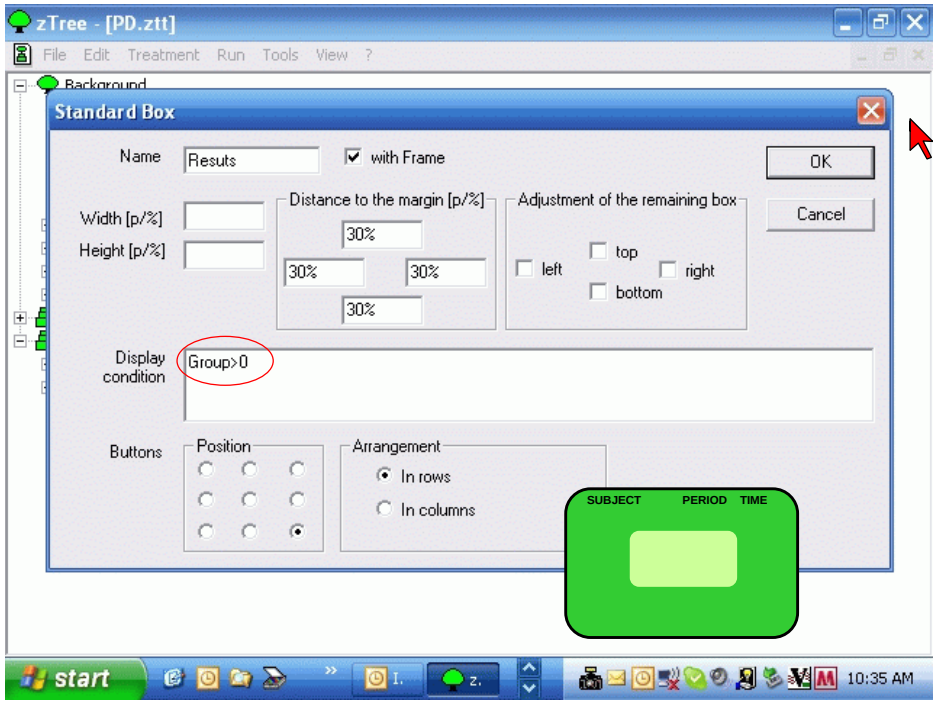


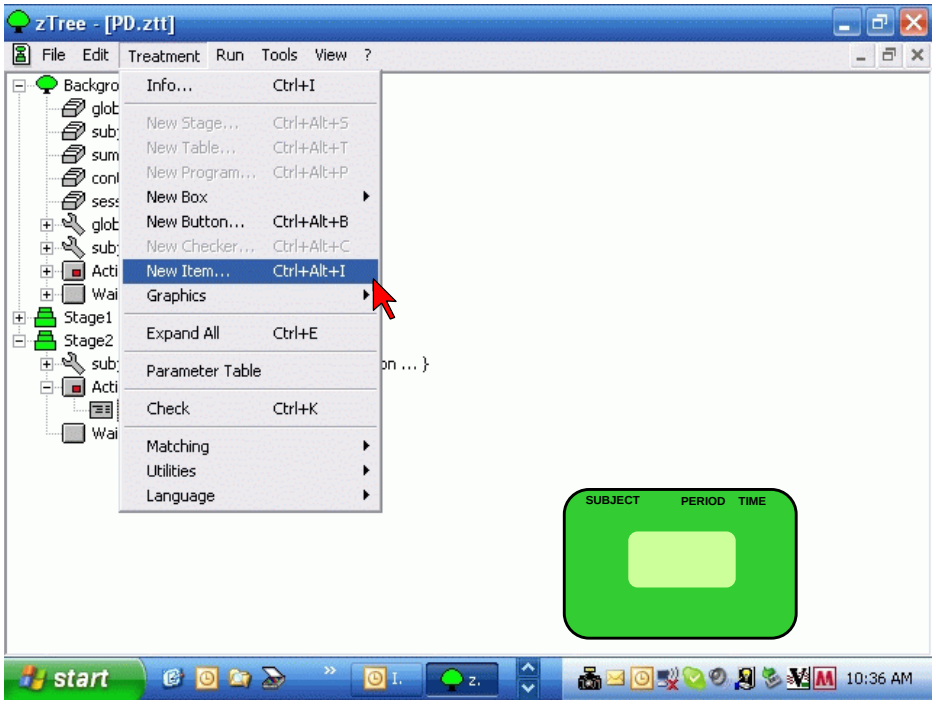


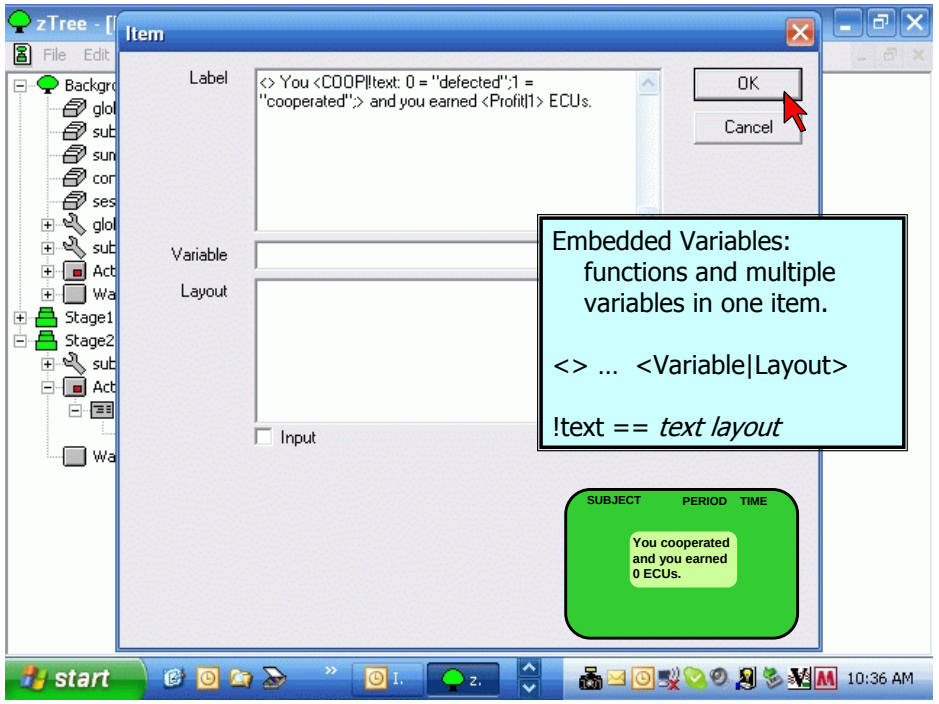
We will now place result on subject's screen.

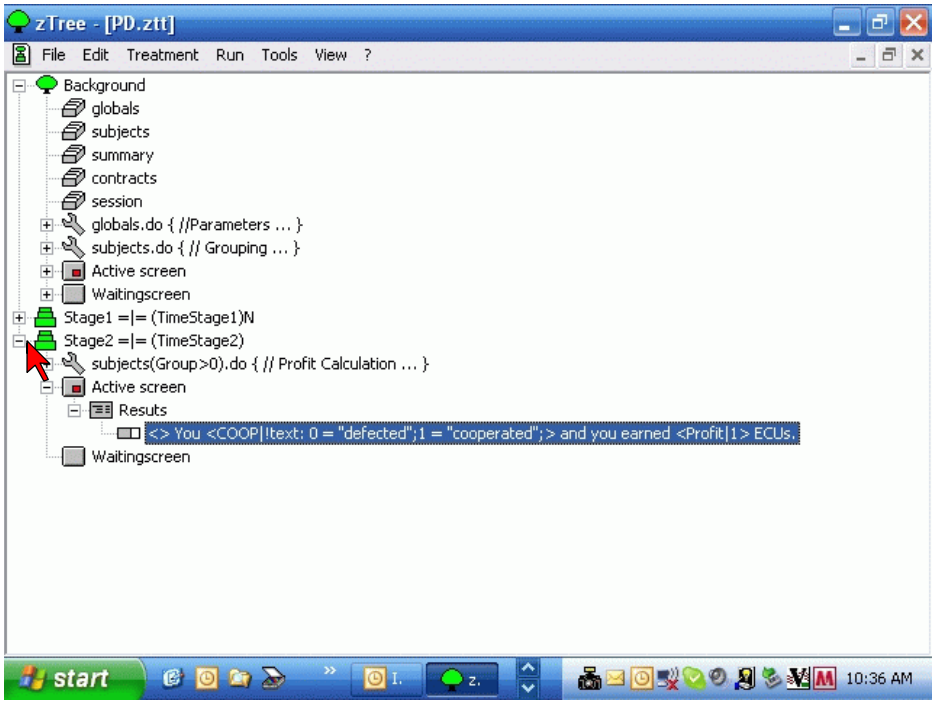
Create STANDARD BOX.

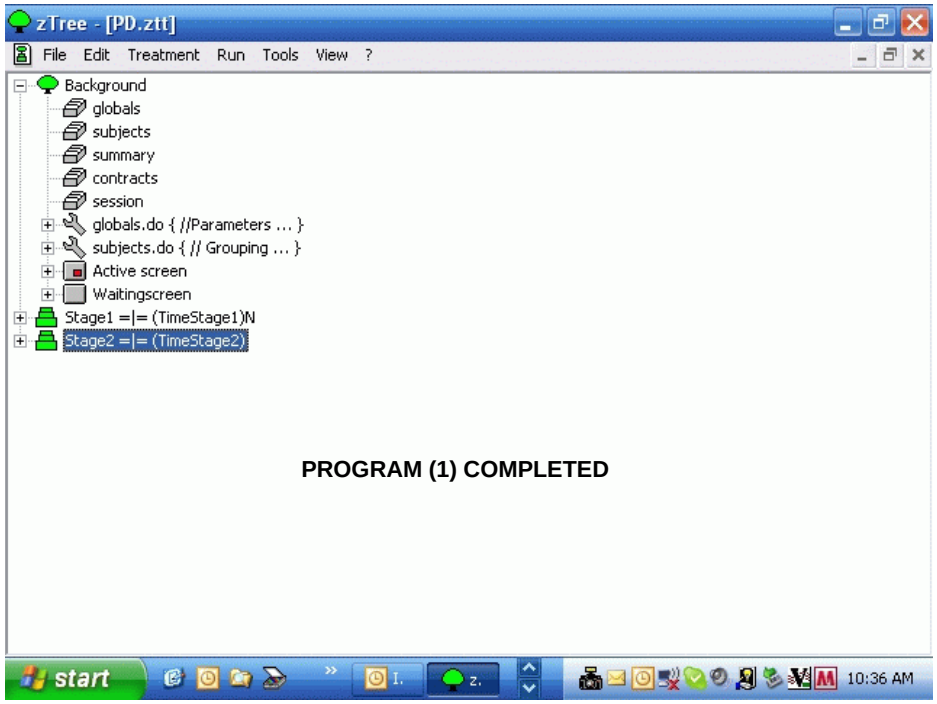

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
Header Box... Ctrl+Alt+K
Standard Box... Ctrl+Alt+D
Calculator Button Box...
History Box... Ctrl+Alt+H
Help Box... Ctrl+Alt+F
Container Box... Ctrl+Alt+O
Grid Box... Ctrl+Alt+G
Contract Creation Box... Ctrl+Alt+V
Contract List Box... Ctrl+Alt+W
Contact Grid Box... Ctrl+Alt+Z
Message Box... Ctrl+Alt+M
New Multimedia Box...
New Plot Box...
New Chat Box...
Stage1
Stage2
start
9:00 PM

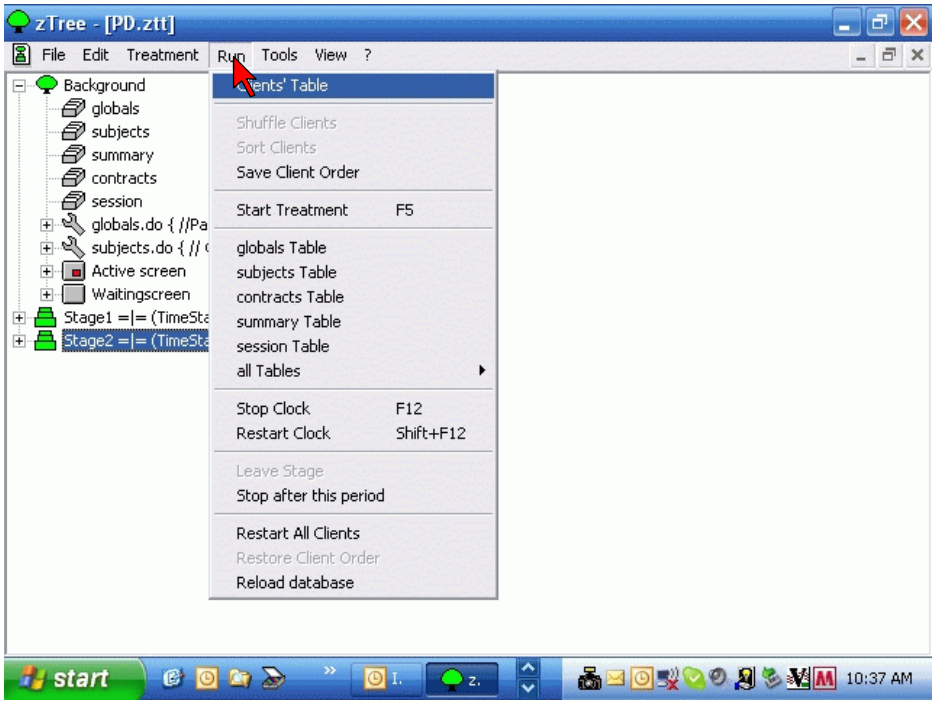

zTree - [PD.ztt]
Standard Box
Name
Resuts
with Frame
OK
Cancel
Width [p/%]
Height [p/%]
Distance to the margin [p/%]
30%
Adjustment of the remaining box
top
left
right
bottom
Display condition
Group>0
Buttons
Position
Arrangement
In rows
In columns
SUBJECT PERIOD TIME
start
10:35 AM

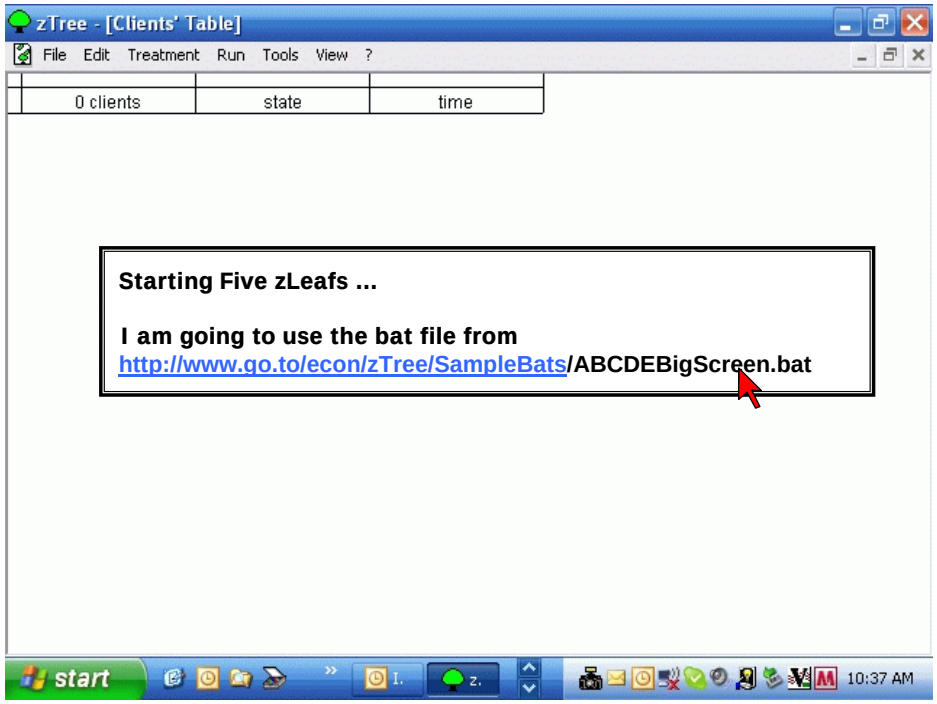

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Info... Ctrl+I
New Stage... Ctrl+Alt+S
New Table... Ctrl+Alt+T
New Program... Ctrl+Alt+P
New Box
New Button... Ctrl+Alt+B
New Checker... Ctrl+Alt+C
New Item... Ctrl+Alt+I
Graphics
Expand All Ctrl+E
Parameter Table
Check Ctrl+K
Matching
Utilities
Language
SUBJECT PERIOD TIME
start
10:36 AM

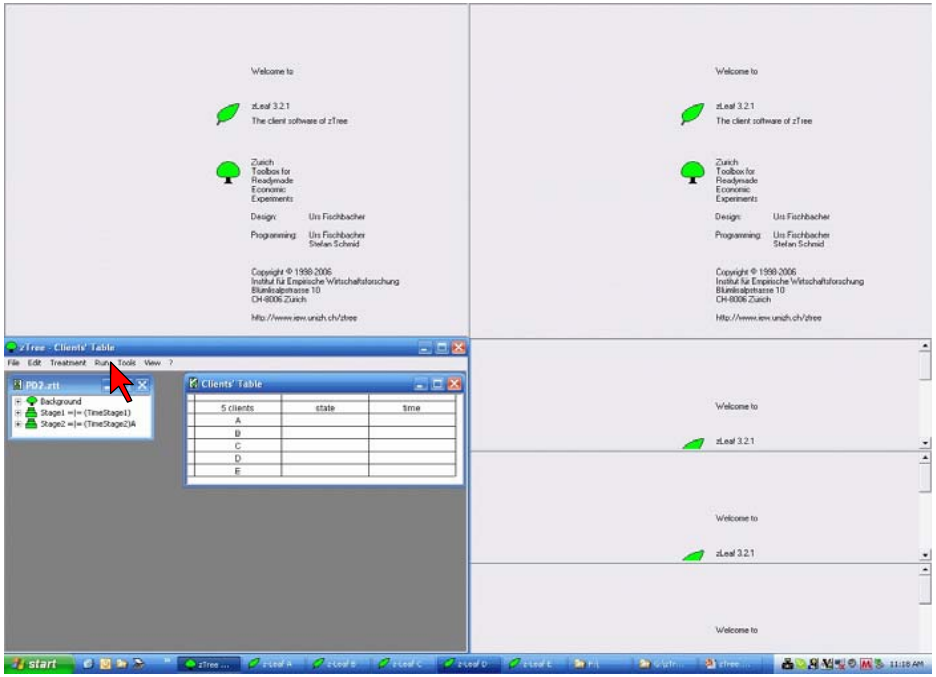

Item
Label
<> You <COOP|!text: 0 = "defected";1 = "cooperated"> and you earned <Profit|1> ECUs.
OK
Cancel
Variable
Layout
Input
Embedded Variables:
functions and multiple variables in one item.
<> ... <Variable|Layout>
!text == text layout
SUBJECT PERIOD TIME
You cooperated and you earned 0 ECUs.
start
10:36 AM

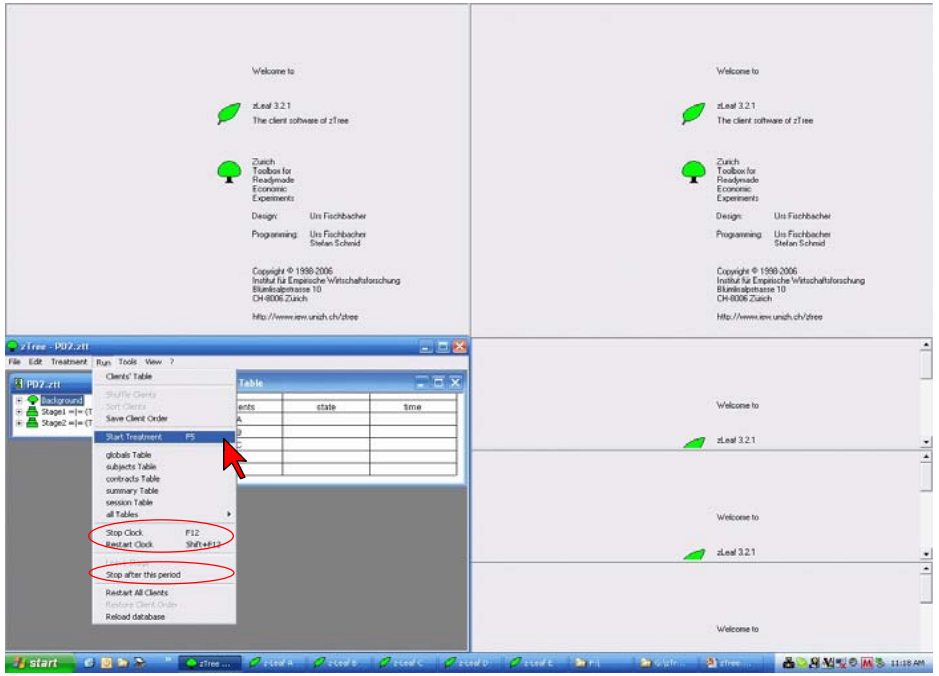

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
Stage2 =|= (TimeStage2)
subjects(Group>0).do { // Profit Calculation ... }
Active screen
Resuts
<> You <COOP|!text: 0 = "defected";1 = "cooperated";> and you earned <Profit|1> ECUs.
Waitingscreen
start
10:36 AM

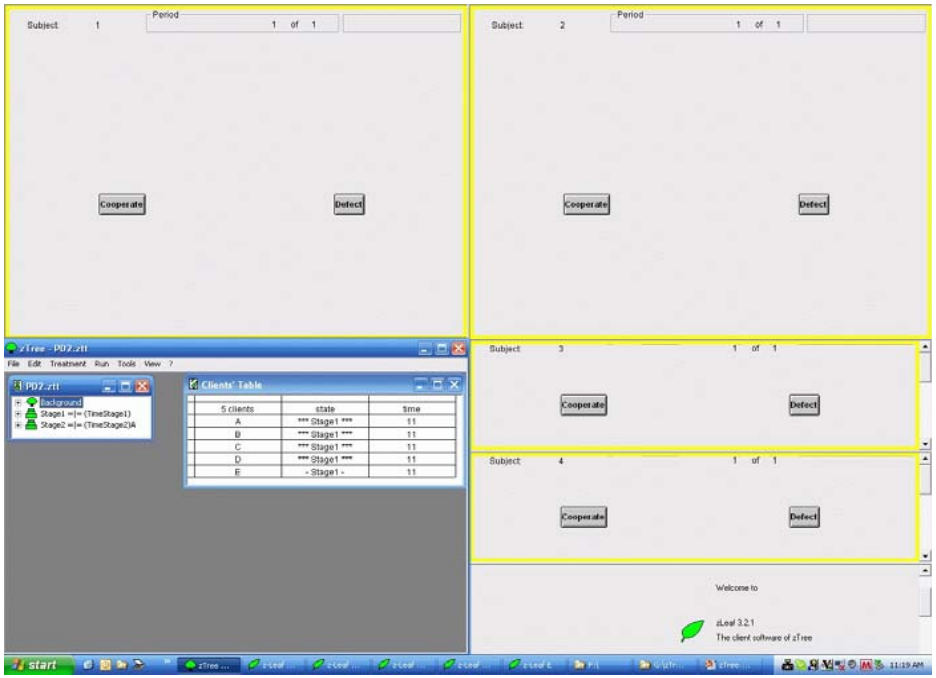

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
globals.do { //Parameters ... }
subjects.do { // Grouping ... }
Active screen
Waitingscreen
Stage1 =|= (TimeStage1)N
Stage2 =|= (TimeStage2)
PROGRAM (1) COMPLETED
start
10:36 AM

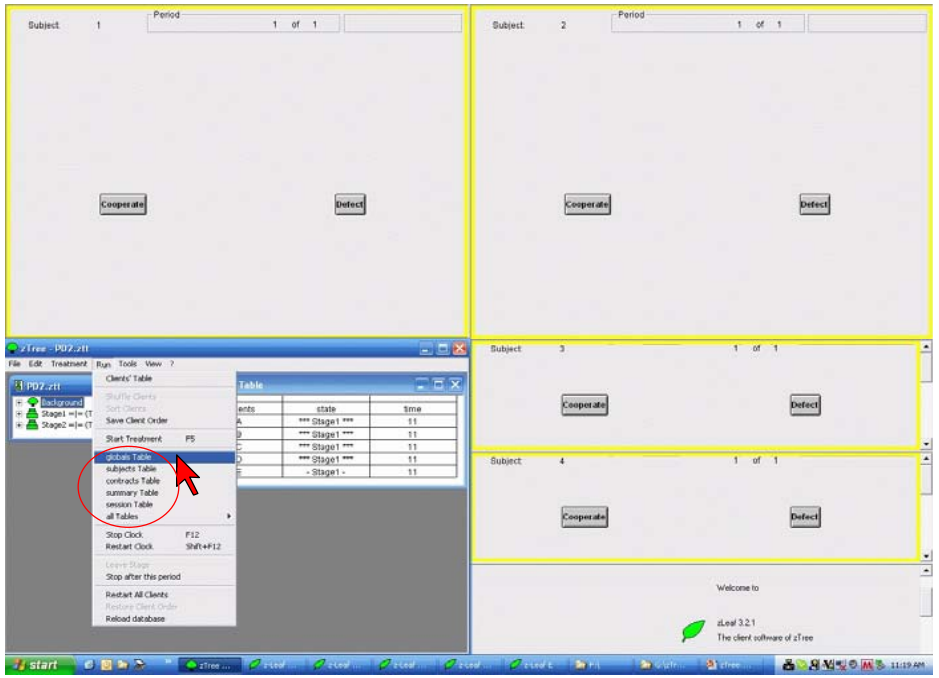

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
globals
subjects
summary
contracts
session
Active screen
Waitingscreen
Stage1 =|= (TimeSta
Stage2 =|= (TimeSta
Clients' Table
Shuffle Clients
Sort Clients
Save Client Order
Start Treatment F5
globals Table
subjects Table
contracts Table
summary Table
session Table
all Tables
Stop Clock F12
Restart Clock Shift+F12
Leave Stage
Stop after this period
Restart All Clients
Restore Client Order
Reload database
start
10:37 AM

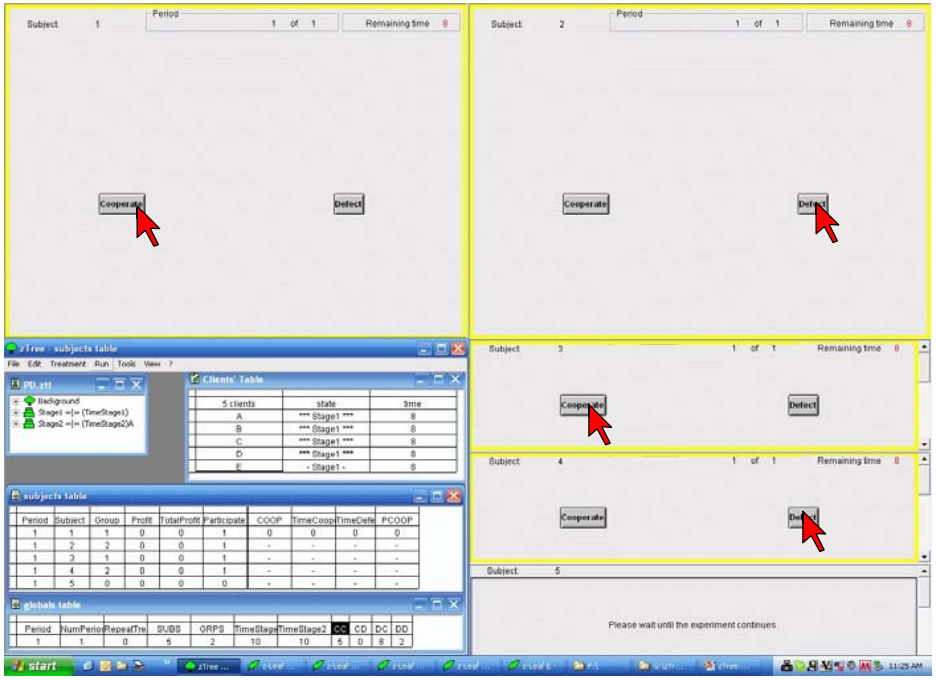

zTree - [Clients' Table]
File Edit Treatment Run Tools View ?
0 clients
state
time
Starting Five zLeafs ...
I am going to use the bat file from
http://www.go.to/econ/zTree/SampleBats/ABCDEBigScreen.bat
start
10:37 AM

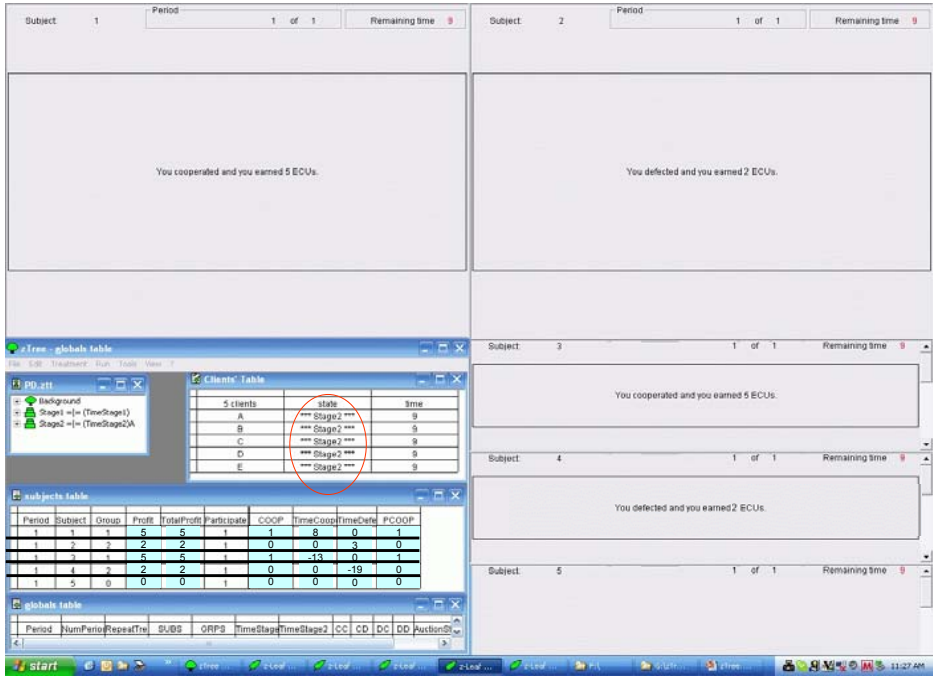

Welcome to
zLeaf 3.2.1
The client software of zTree
Zurich Toolbox for Readymade Economic Experiments
Design: Urs Fischbacher
Programming: Urs Fischbacher Stefan Schmid
Copyright © 1998-2006 Institut für Empirische Wirtschaftsforschung Blümlisalpstrasse 10 CH-8006 Zürich
http://www.iew.unizh.ch/ztree
zTree - Clients' Table
File Edit Treatment Run Tools View ?
PD2.zti
Background
Stage1 =|= (TimeStage1)
Stage2 =|= (TimeStage2)A
Clients' Table
5 clients
state
time
A
B
C
D
E
start

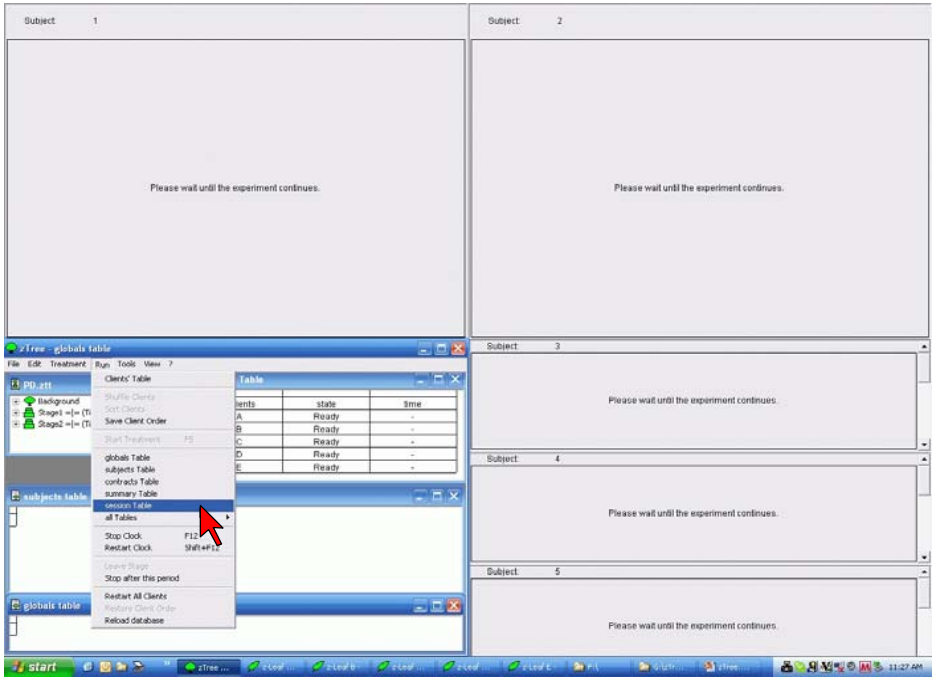

Welcome to
zLeaf 3.2.1
The client software of zTree
Zurich Toolbox for Readymade Economic Experiments
Design: Urs Fischbacher
Programming: Urs Fischbacher Stefan Schmid
Copyright © 1998-2006 Institut für Empirische Wirtschaftsforschung Blümlisalpstrasse 10 CH-8006 Zürich
http://www.iew.unizh.ch/ztree
zTree - PD2.zti
File Edit Treatment Run Tools View ?
Clients' Table
Shuffle Clients
Sort Clients
Save Client Order
Start Treatment F5
globals Table
subjects Table
contracts Table
summary Table
session Table
all Tables
Stop Clock F12
Restart Clock Shift+F12
Leave Stage
Stop after this period
Restart All Clients
Restore Client Order
Reload database
start

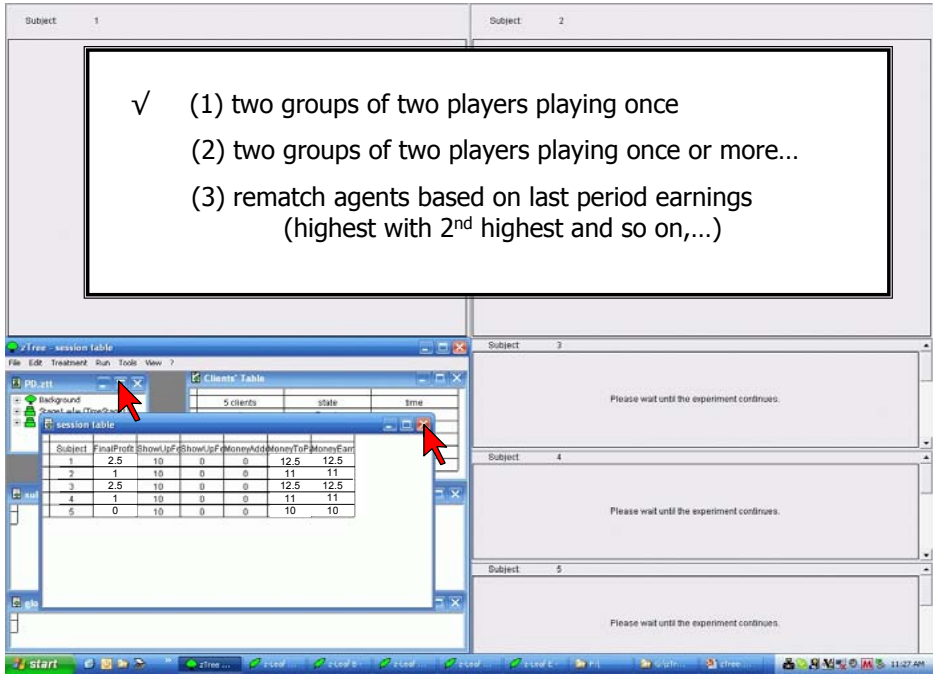

Subject 1
Period 1 of 1
Cooperate
Defect
Subject 2
Subject 3
Subject 4
zTree - PD2.zti
File Edit Treatment Run Tools View ?
Clients' Table
5 clients
state
time
A *** Stage1 *** 11
B *** Stage1 *** 11
C *** Stage1 *** 11
D *** Stage1 *** 11
E - Stage1 - 11
Welcome to
zLeaf 3.2.1
The client software of zTree
start


Subject 1
Period 1 of 1
Cooperate
Defect
Subject 2
Subject 3
Subject 4
zTree - PD2.zti
File Edit Treatment Run Tools View ?
Clients' Table
Shuffle Clients
Sort Clients
Save Client Order
Start Treatment F5
globals Table
subjects Table
contracts Table
summary Table
session Table
all Tables
Stop Clock F12
Restart Clock Shift+F12
Leave Stage
Stop after this period
Restart All Clients
Restore Client Order
Reload database
Welcome to
zLeaf 3.2.1
The client software of zTree
start

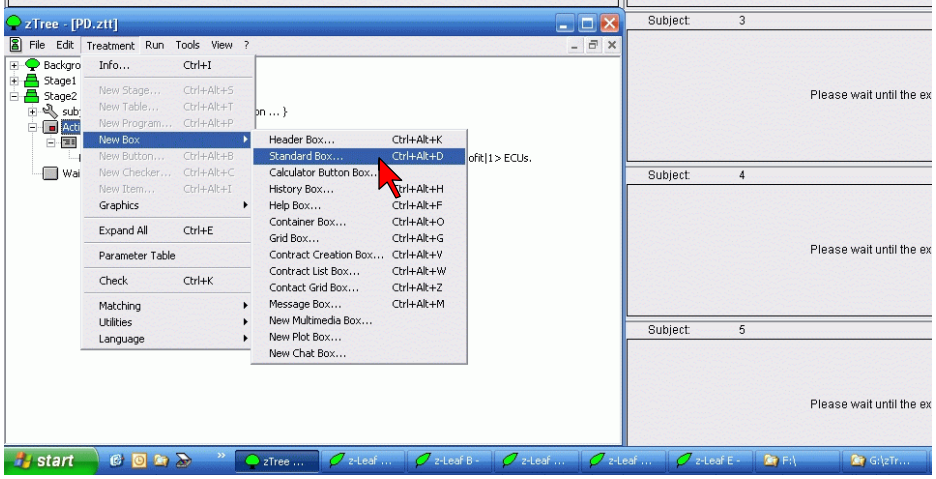

Subject 1
Period 1 of 1
Remaining time 8
Cooperate
Defect
Subject 2
Subject 3
Subject 4
Subject 5
Please wait until the experiment continues.
zTree - subjects table
Clients' Table
subjects table
globals table


You cooperated and you earned 5 ECUs.
You defected and you earned 2 ECUs.
zTree - globals table
Clients' Table
subjects table
globals table

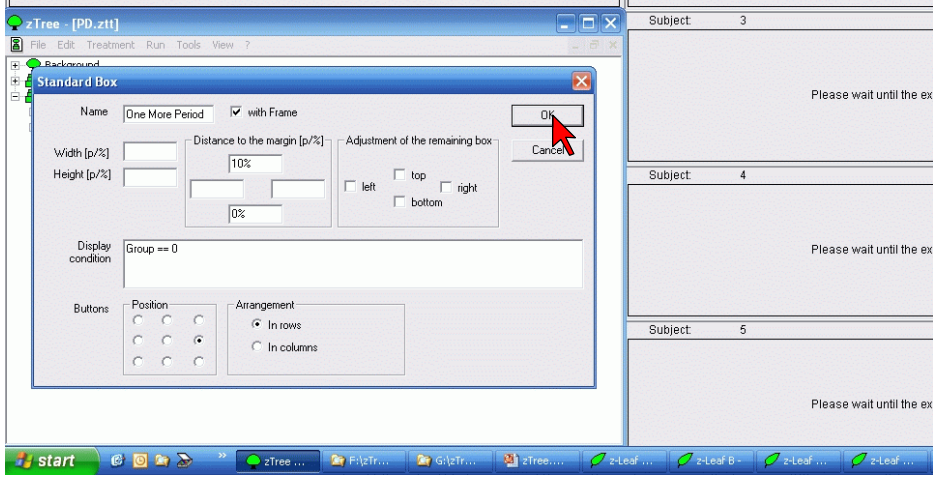

Please wait until the experiment continues.
zTree - globals table
Clients' Table
subjects table
globals table
session table


√ (1) two groups of two players playing once
(2) two groups of two players playing once or more…
(3) rematch agents based on last period earnings
(highest with 2nd highest and so on,…)
zTree - session table
Clients' Table
session table
Please wait until the experiment continues.

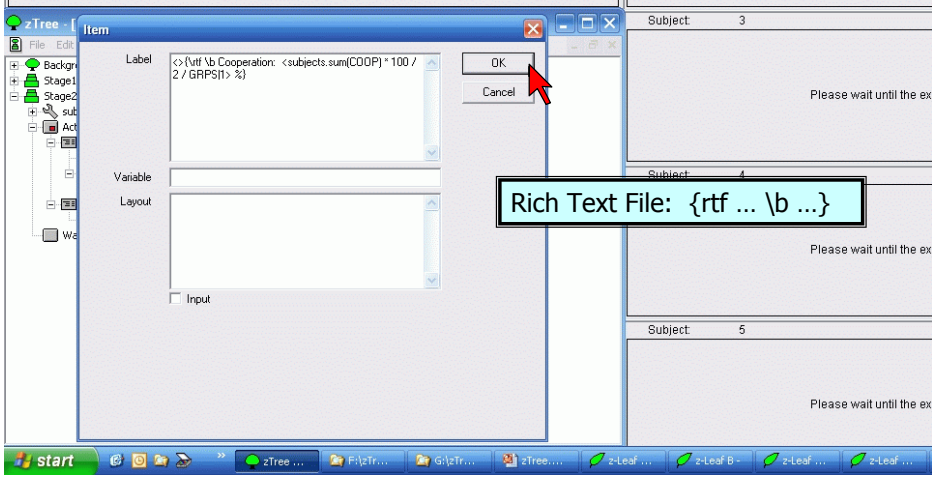
Add a box for group==0 zLeaf in Stage2



Add a box for group==0 zLeaf in Stage2

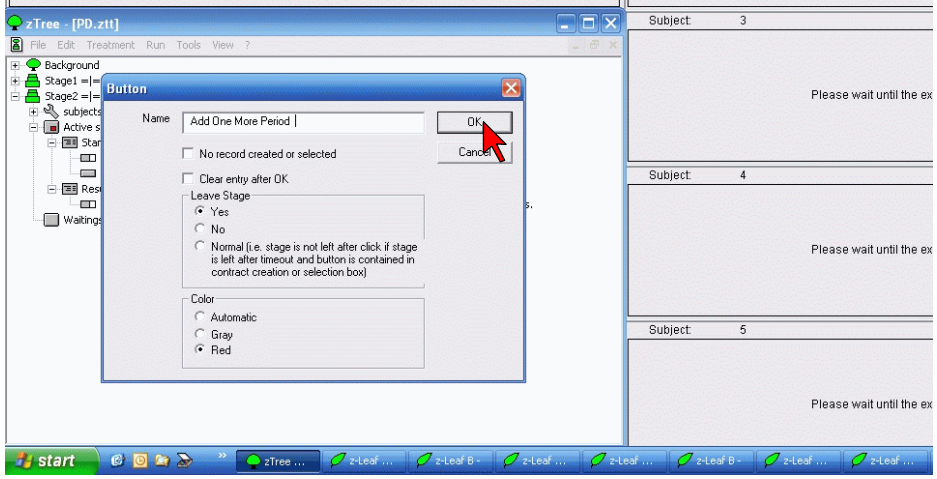


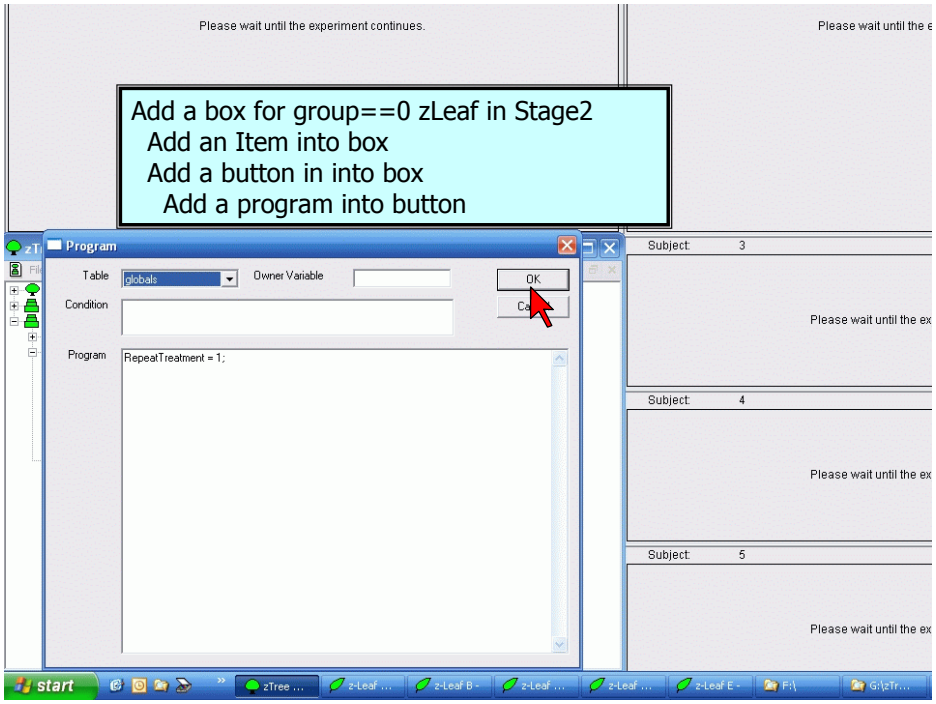
Add a box for group==0 zLeaf in Stage2
Add an Item into box

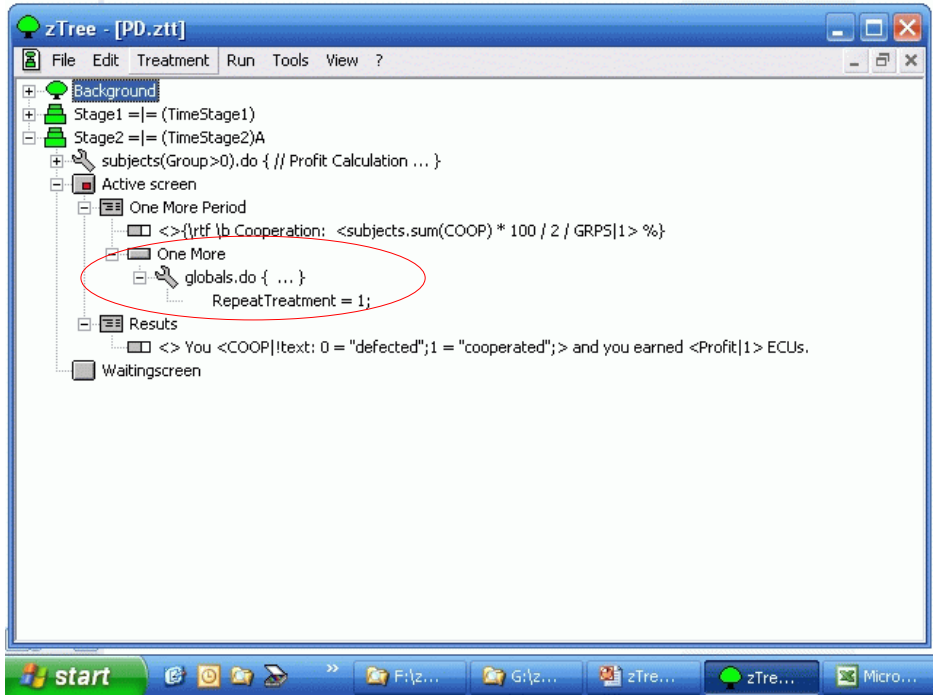


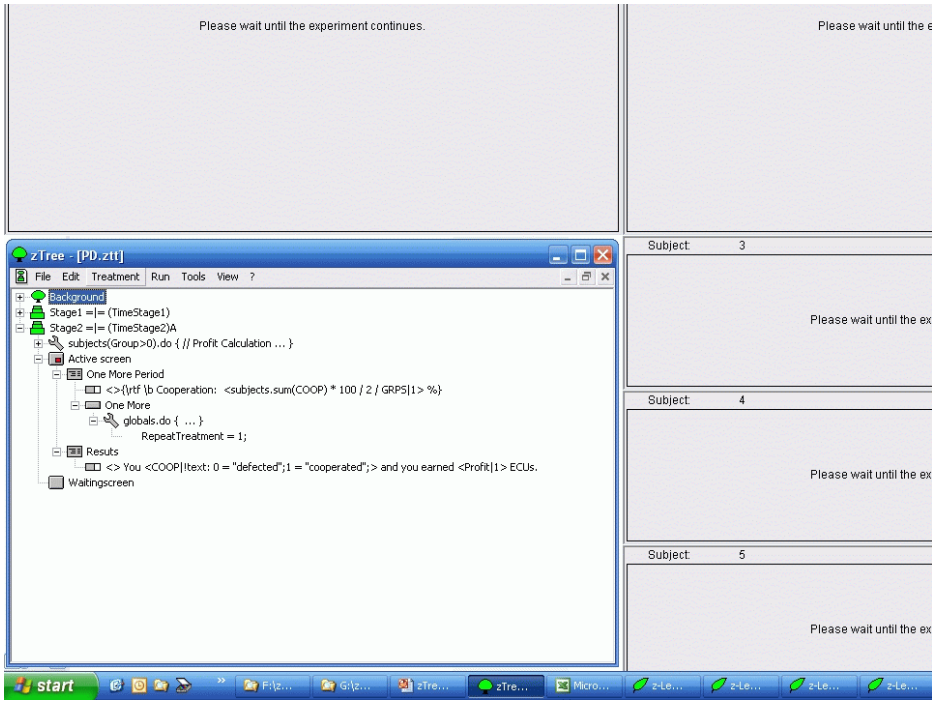
Rich Text File: {rtf ... \b ...}

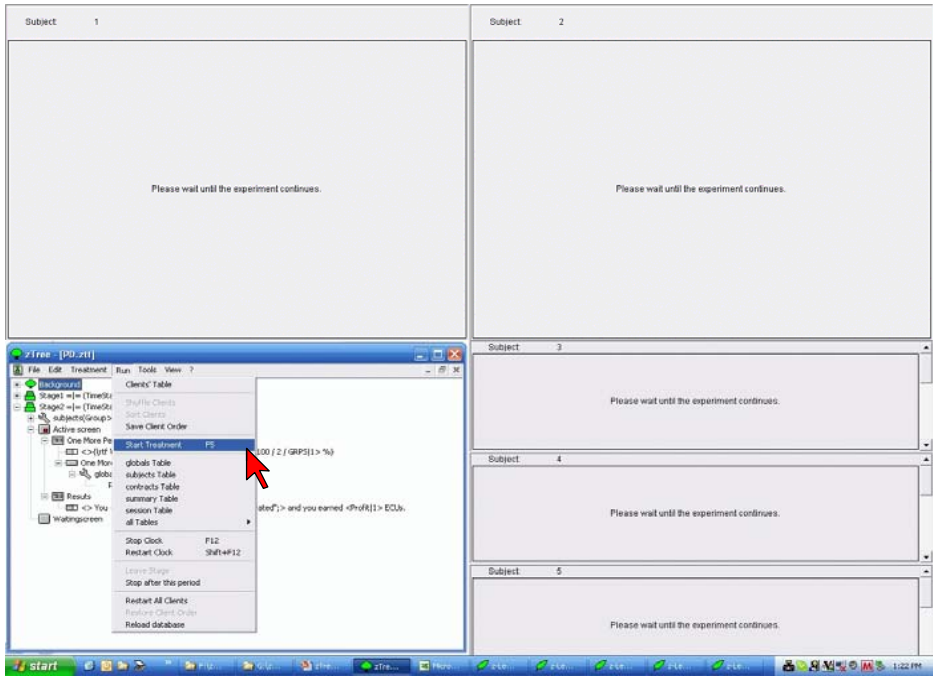
Add a box for group==0 zLeaf in Stage2
Add an Item into box
Add a button in into box

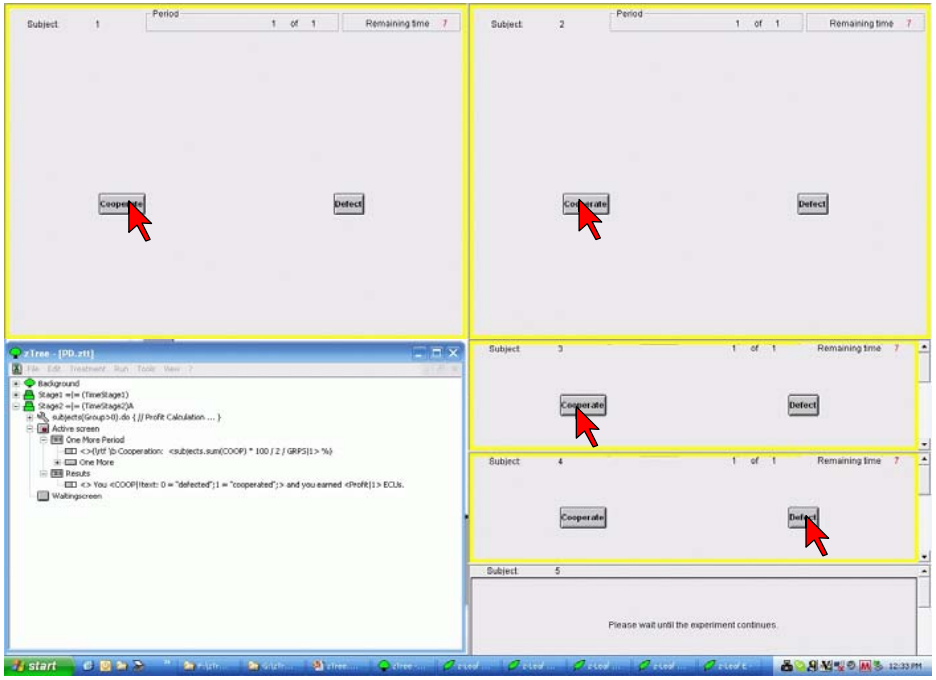


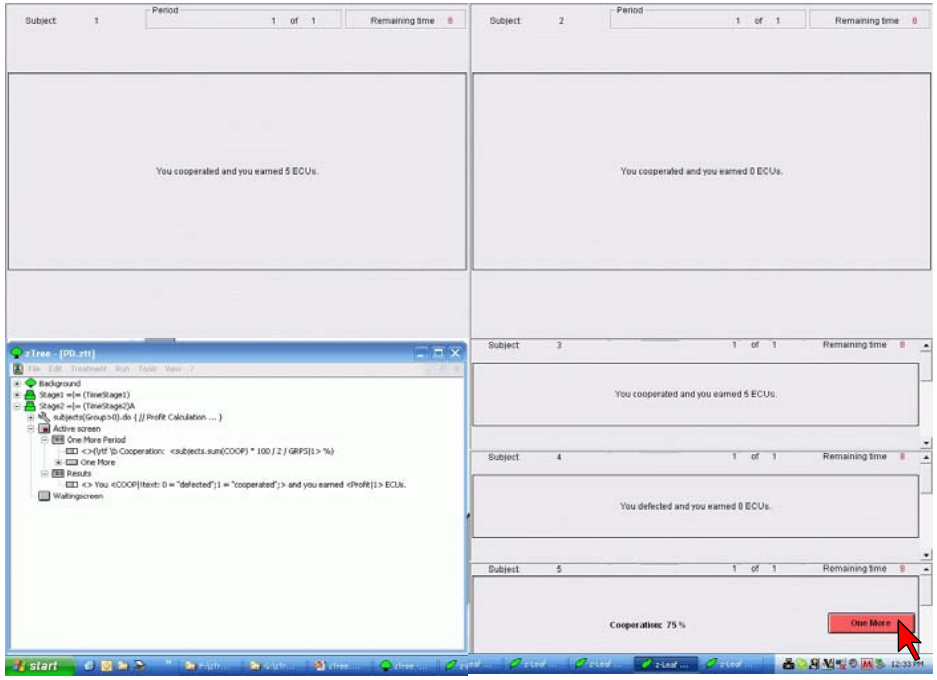

Please wait until the experiment continues.
Add a box for group==0 zLeaf in Stage2
Add an Item into box
Add a button in into box
Add a program into button
Program
Table
globals
Owner Variable
OK
Condition
Program
RepeatTreatment = 1;
Subject: 3
Subject: 4
Subject: 5

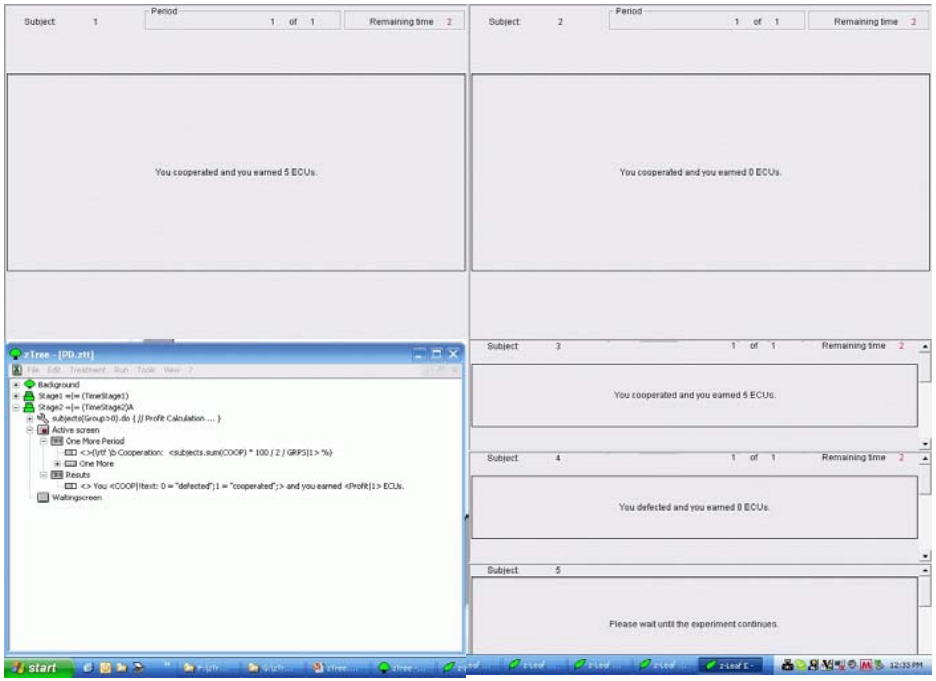

zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
Stage1 =|= (TimeStage1)
Stage2 =|= (TimeStage2)A
subjects(Group>0).do { // Profit Calculation ... }
Active screen
One More Period
<>{\rtf \b Cooperation: <subjects.sum(COOP) * 100 / 2 / GRPS|1> %}
One More
globals.do { ... }
RepeatTreatment = 1;
Resuts
<> You <COOP|!text: 0 = "defected";1 = "cooperated";> and you earned <Profit|1> ECUs.
Waitingscreen

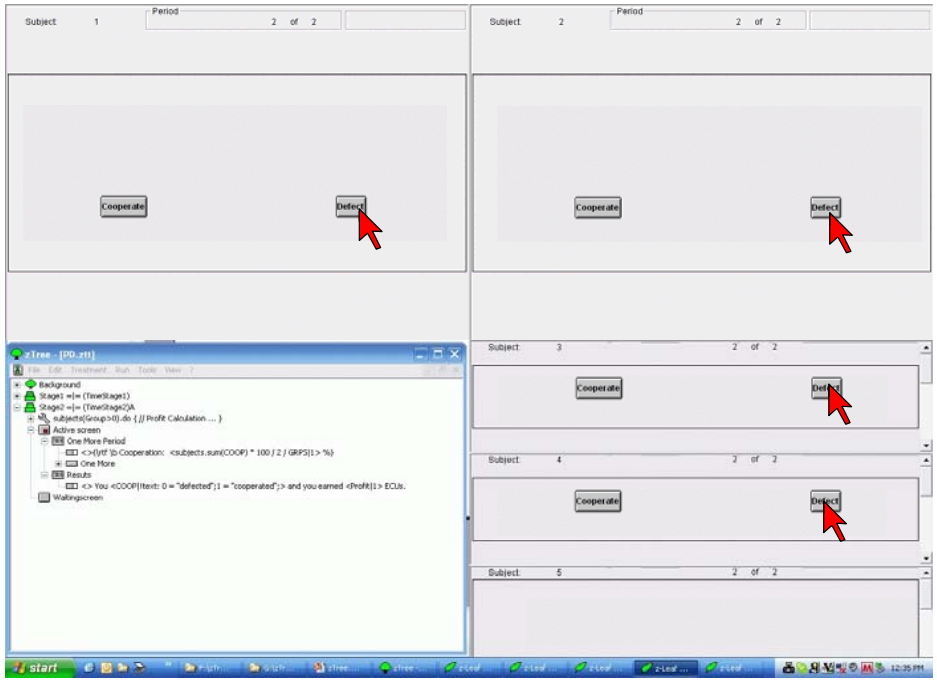

Please wait until the experiment continues.
zTree - [PD.ztt]
File Edit Treatment Run Tools View ?
Background
Stage1 =|= (TimeStage1)
Stage2 =|= (TimeStage2)A
subjects(Group>0).do { // Profit Calculation ... }
Active screen
One More Period
<>{\rtf \b Cooperation: <subjects.sum(COOP) * 100 / 2 / GRPS|1> %}
One More
globals.do { ... }
RepeatTreatment = 1;
Resuts
<> You <COOP|!text: 0 = "defected";1 = "cooperated";> and you earned <Profit|1> ECUs.
Waitingscreen
Subject: 3
Subject: 4
Subject: 5

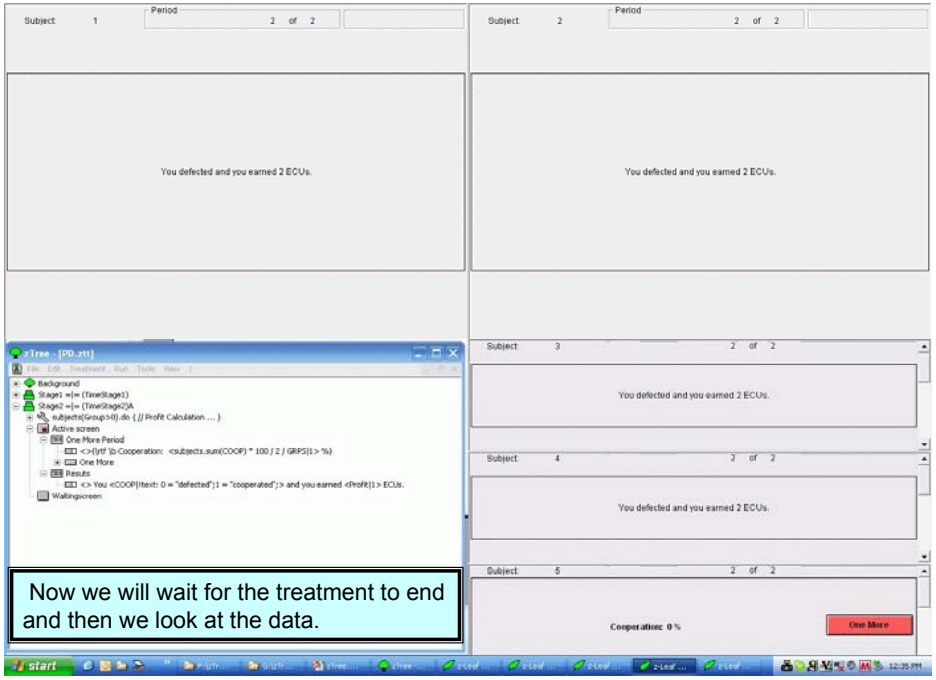

Subject: 1
Subject: 2
Please wait until the experiment continues.
Subject: 3
Subject: 4
Subject: 5

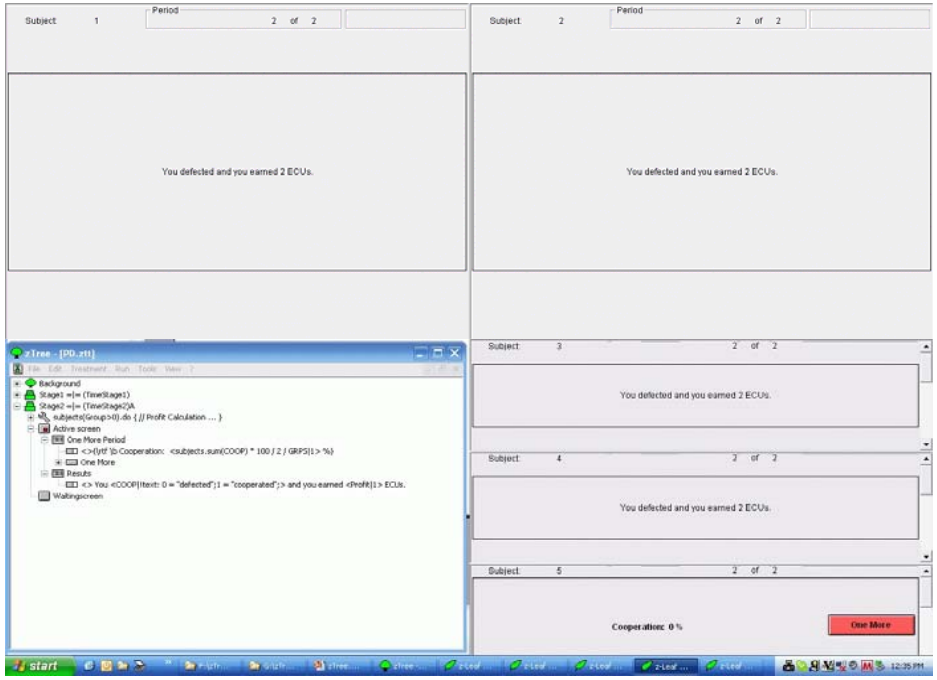

Subject 1
Period 1 of 1
Remaining time 7
Cooperate
Defect
Subject 2
Period 1 of 1
Remaining time 7
Cooperate
Defect
zTree - [PD.ztt]
Subject 3
1 of 1
Remaining time 7
Cooperate
Defect
Subject 4
1 of 1
Remaining time 7
Cooperate
Defect
Subject 5
Please wait until the experiment continues.

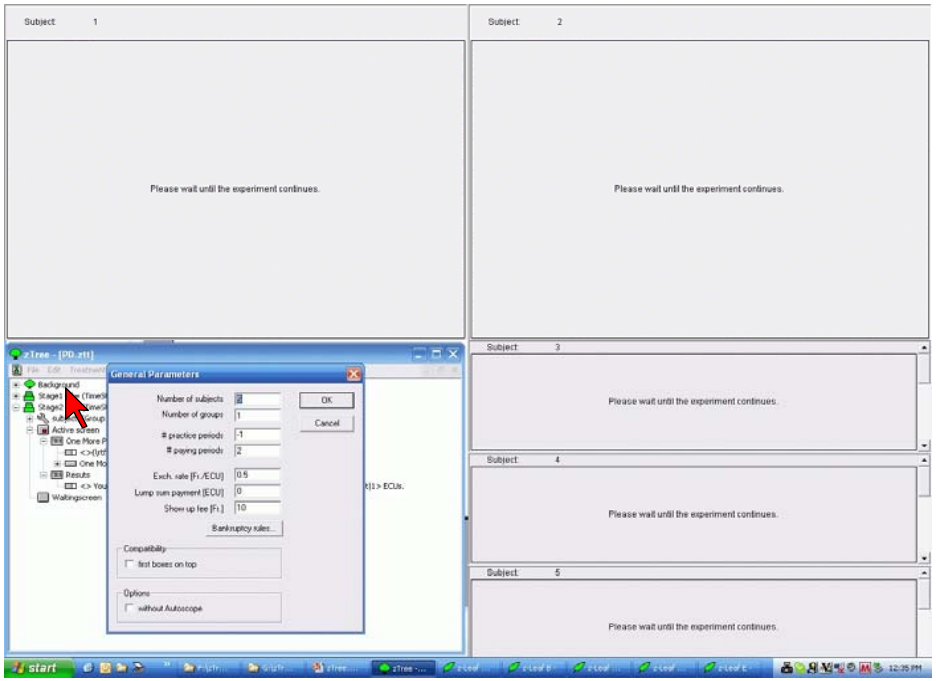

Subject 1
Period 1 of 1
Remaining time 8
You cooperated and you earned 5 ECUs.
Subject 2
Period 1 of 1
Remaining time 8
You cooperated and you earned 0 ECUs.
zTree - [PD.ztt]
Subject 3
1 of 1
Remaining time 8
You cooperated and you earned 5 ECUs.
Subject 4
1 of 1
Remaining time 8
You defected and you earned 0 ECUs.
Subject 5
1 of 1
Remaining time 8
Cooperation: 75 %
One More

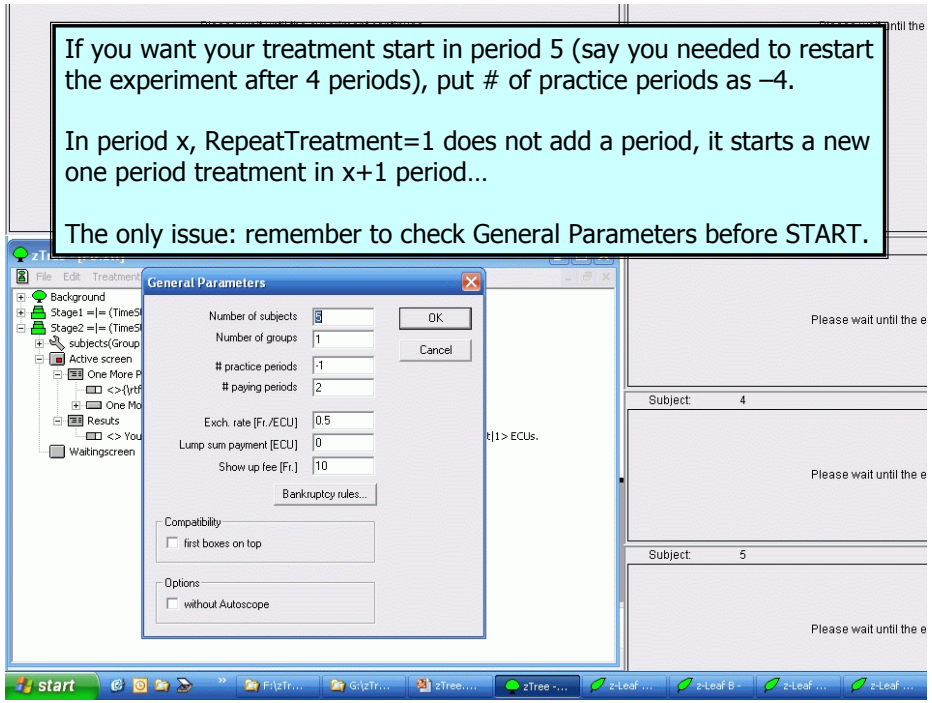

Subject 1
Period 1 of 1
Remaining time 2
You cooperated and you earned 5 ECUs.
Subject 2
Period 1 of 1
Remaining time 2
You cooperated and you earned 0 ECUs.
zTree - [PD.ztt]
Subject 3
1 of 1
Remaining time 2
You cooperated and you earned 5 ECUs.
Subject 4
1 of 1
Remaining time 2
You defected and you earned 0 ECUs.
Subject 5
Please wait until the experiment continues.

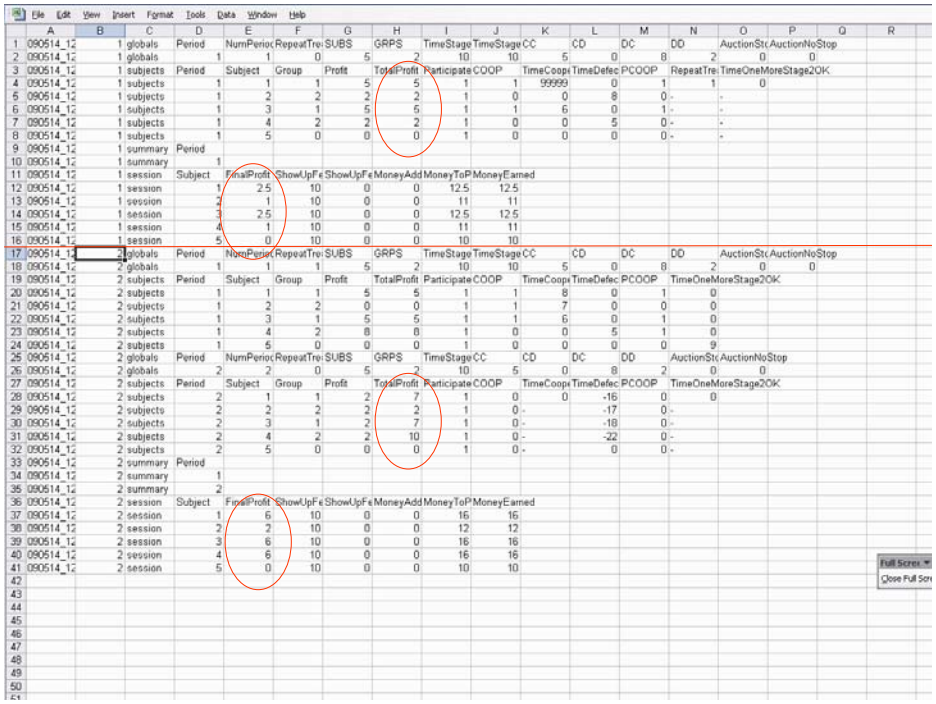

Subject 1
Period 2 of 2
Cooperate
Defect
Subject 2
Period 2 of 2
Cooperate
Defect
zTree - [PD.ztt]
Subject 3
2 of 2
Cooperate
Defect
Subject 4
2 of 2
Cooperate
Defect
Subject 5
2 of 2

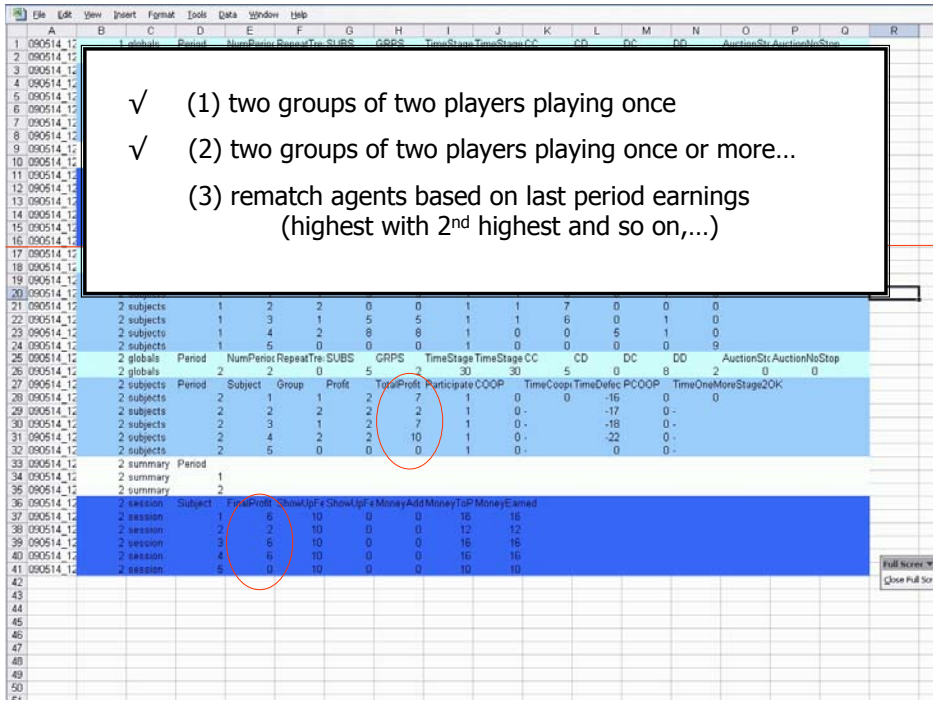

Subject 1
Period 2 of 2
You defected and you earned 2 ECUs.
Subject 2
Period 2 of 2
You defected and you earned 2 ECUs.
Subject 3
2 of 2
You defected and you earned 2 ECUs.
Subject 4
2 of 2
You defected and you earned 2 ECUs.
Subject 5
2 of 2
Cooperation: 0 %
One More
Now we will wait for the treatment to end and then we look at the data.


Subject 1
Period 2 of 2
You defected and you earned 2 ECUs.
Subject 2
Period 2 of 2
You defected and you earned 2 ECUs.
Subject 3
2 of 2
You defected and you earned 2 ECUs.
Subject 4
2 of 2
You defected and you earned 2 ECUs.
Subject 5
2 of 2
Cooperation: 0 %
One More


Subject 1
Please wait until the experiment continues.
Subject 2
Please wait until the experiment continues.
Subject 3
Please wait until the experiment continues.
Subject 4
Please wait until the experiment continues.
Subject 5
Please wait until the experiment continues.
General Parameters
Number of subjects
Number of groups
practice periods
paying periods
Exch. rate [Fr./ECU]
Lump sum payment [ECU]
Show up fee [Fr.]
OK
Cancel
Bankruptcy rules...
Compatibility
first boxes on top
Options
without Autoscope


If you want your treatment start in period 5 (say you needed to restart the experiment after 4 periods), put # of practice periods as –4.
In period x, RepeatTreatment=1 does not add a period, it starts a new one period treatment in x+1 period…
The only issue: remember to check General Parameters before START.
General Parameters
Number of subjects
Number of groups
practice periods
paying periods
Exch. rate [Fr./ECU]
Lump sum payment [ECU]
Show up fee [Fr.]
OK
Cancel
Bankruptcy rules...
Compatibility
first boxes on top
Options
without Autoscope
Subject 4
Subject 5

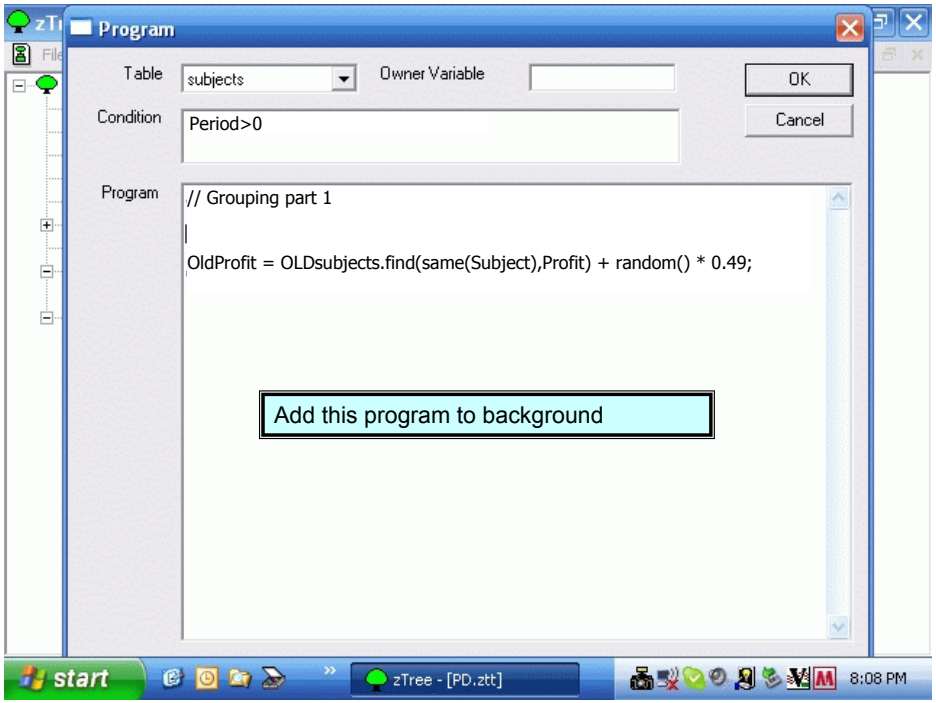


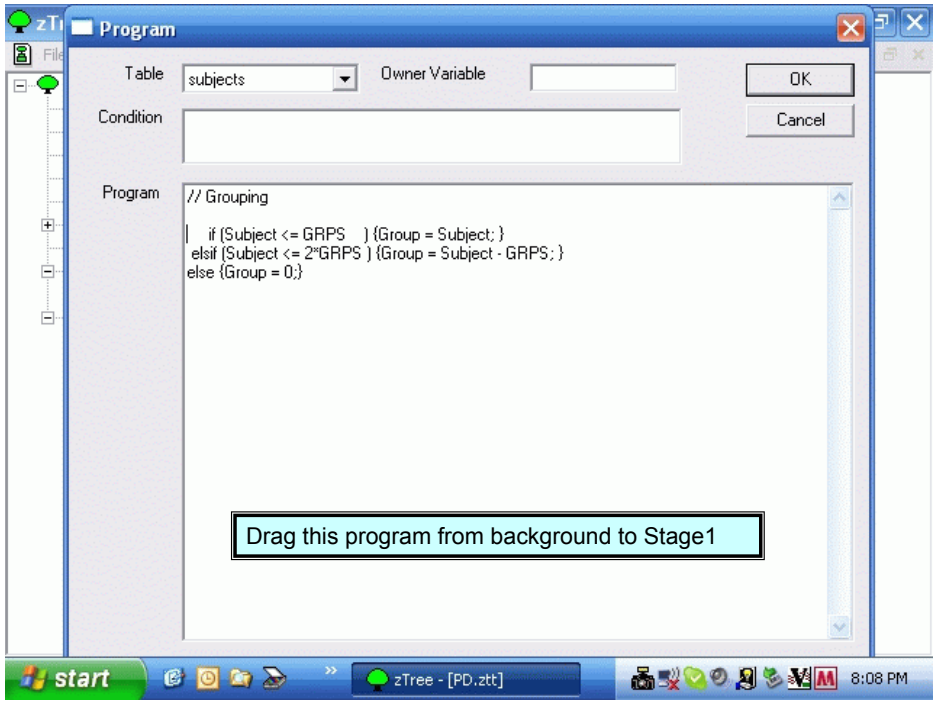


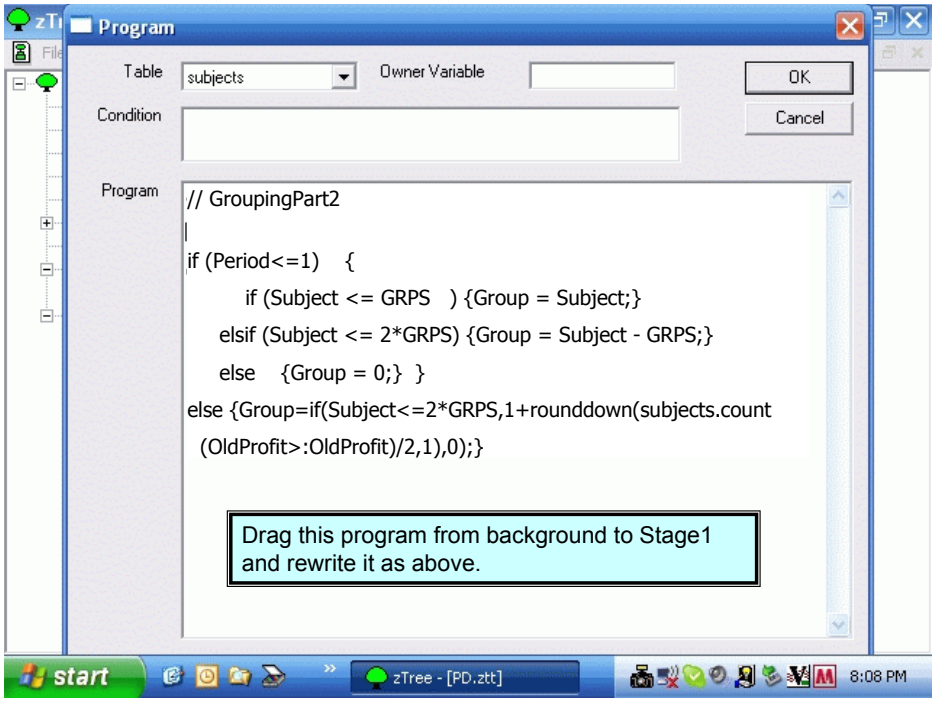
√ (1) two groups of two players playing once

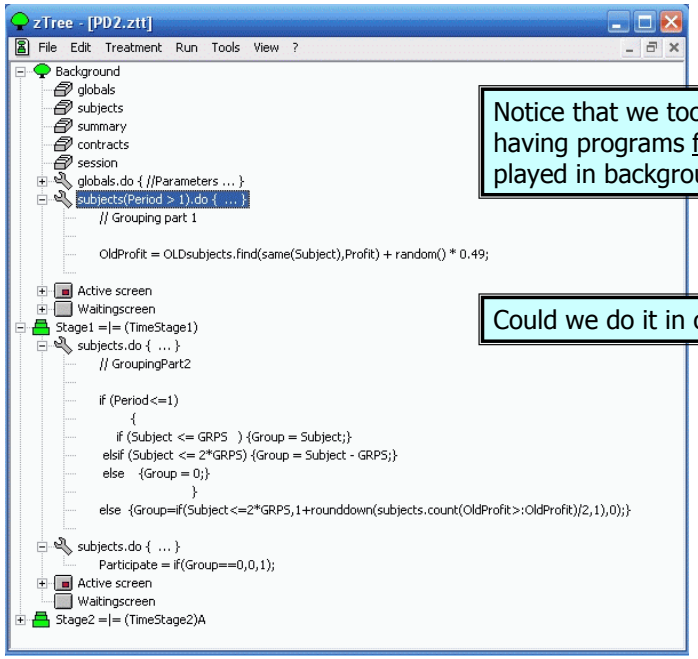
√ (2) two groups of two players playing once or more…

(3) rematch agents based on last period earnings
(highest with 2nd highest and so on,…)









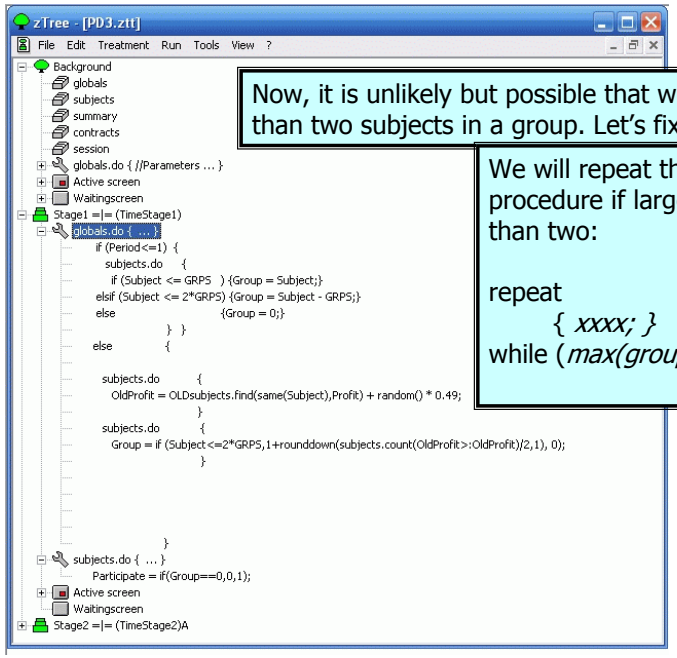
STATEMENTS (#): (continued)

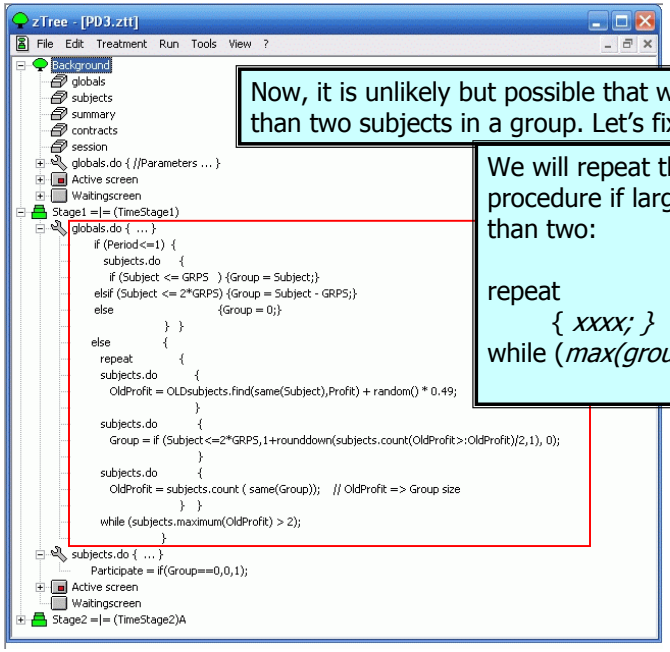
T.do {#1;} **does statement in a table T**

example:

```
subjects.do {Group =Group +1;
             if (Group>GRPS) {Group = 1;}}
```

T.new {#1;} **creates a new record in table T (contracts or yours table)**
(e.g. new offer is placed in auction)





Now, it is unlikely but possible that we will have more than two subjects in a group. Let's fix that...

We will repeat the whole grouping procedure if largest group is more than two:

repeat
{ *xxxx;* }
while (*max(group) too big*)

```
Stage1 =|= (TimeStage1)
globals.do { ... }
    if (Period<=1) {
      subjects.do    {
        if (Subject <= GRPS  ) {Group = Subject;}
    elsif (Subject <= 2*GRPS) {Group = Subject - GRPS;}
    else                      {Group = 0;}
                  } }
  else            {
    repeat          {
    subjects.do        {
      OldProfit = OLDsubjects.find(same(Subject),Profit) + random() * 0.49;
                       }
    subjects.do        {
      Group = if (Subject<=2*GRPS,1+rounddown(subjects.count(OldProfit>:OldProfit)/2,1), 0);
                       }
    subjects.do        {
      OldProfit = subjects.count ( same(Group));   // OldProfit => Group size
                     } }
    while (subjects.maximum(OldProfit) > 2);
                }
subjects.do { ... }
```

STATEMENTS (#): (continued)

T.do {#1;} **does statement in a table T**

example:

subjects.do {Group =Group +1;
if (Group>GRPS) {Group = 1;}}

T.new {#1;} **creates a new record in table T (contracts)**
(e.g. new offer in auction)

----

array x[n]; **creates an array of vars: x[1], x[2],..x[n]**

----

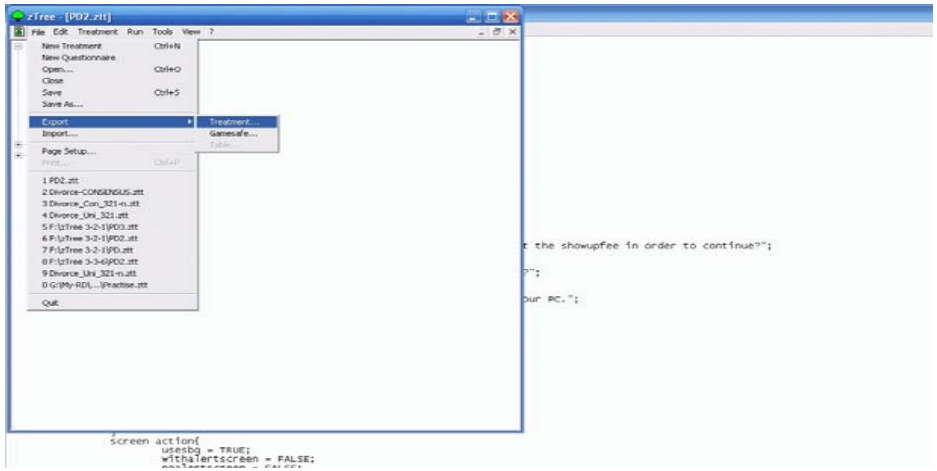
later (n) do {#1;} **does #1 n second later (if n>0)**

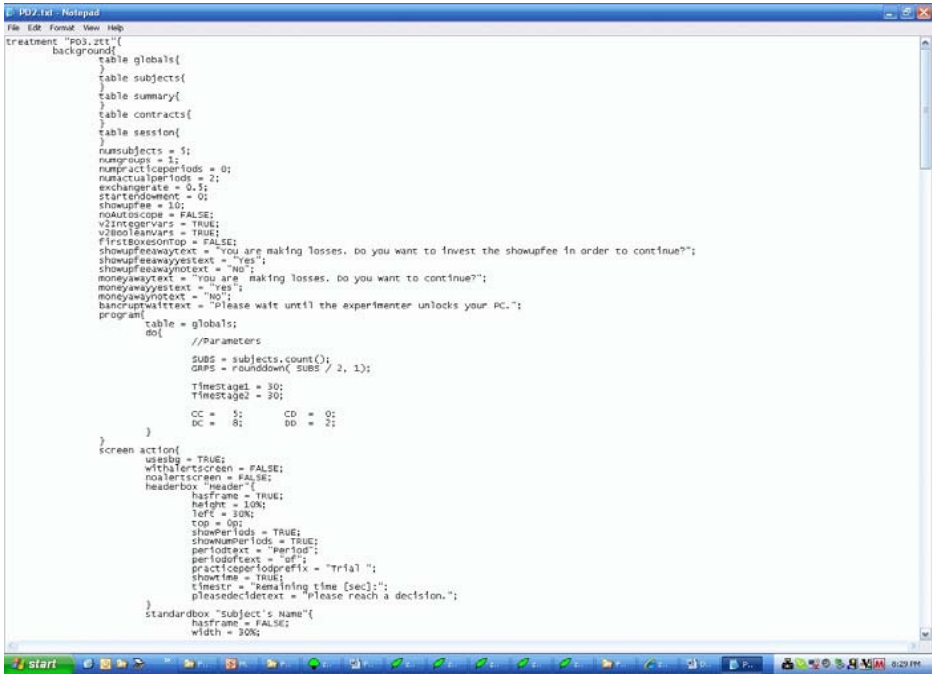
later (n) repeat {#1;} **does #1 n second later (if n>0)**
refreshes value of n and repeats

How big is the Program File?

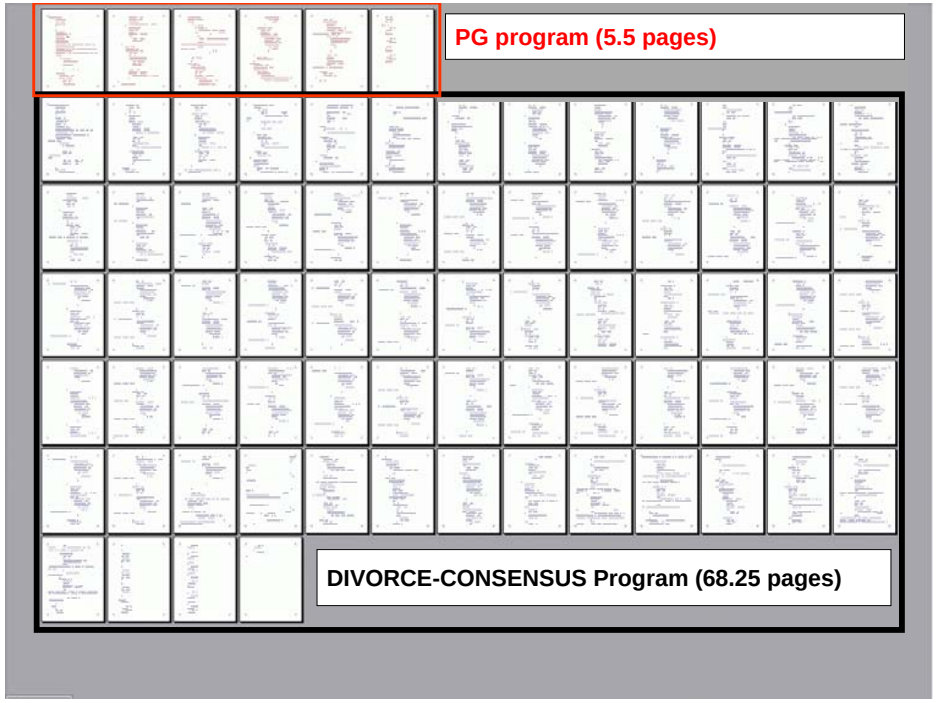
Very small.

You can export it into a text (importing helped with transfer from older versions of zTree Programs.)



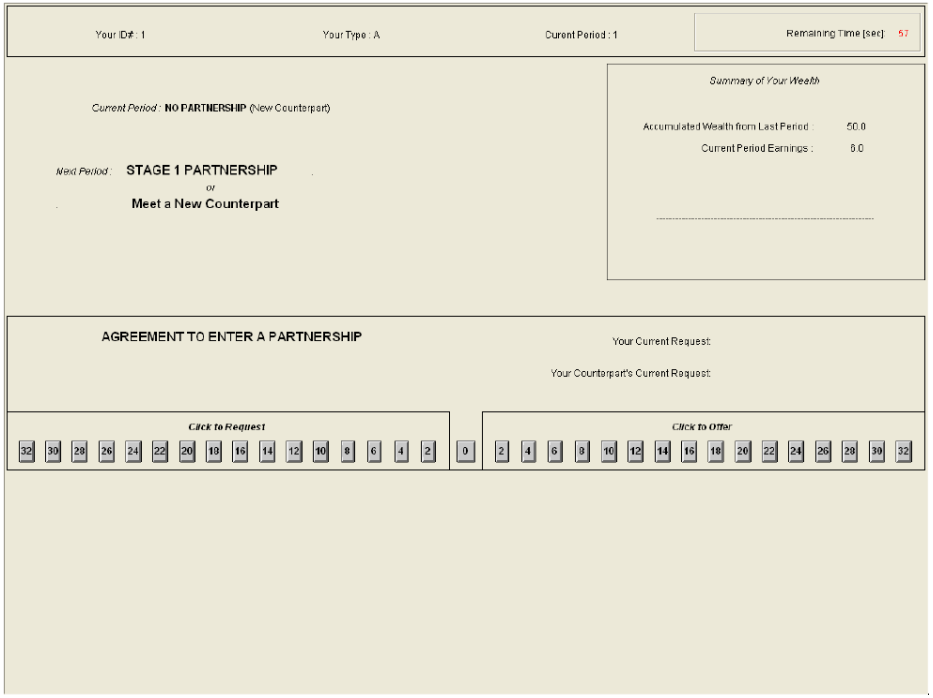


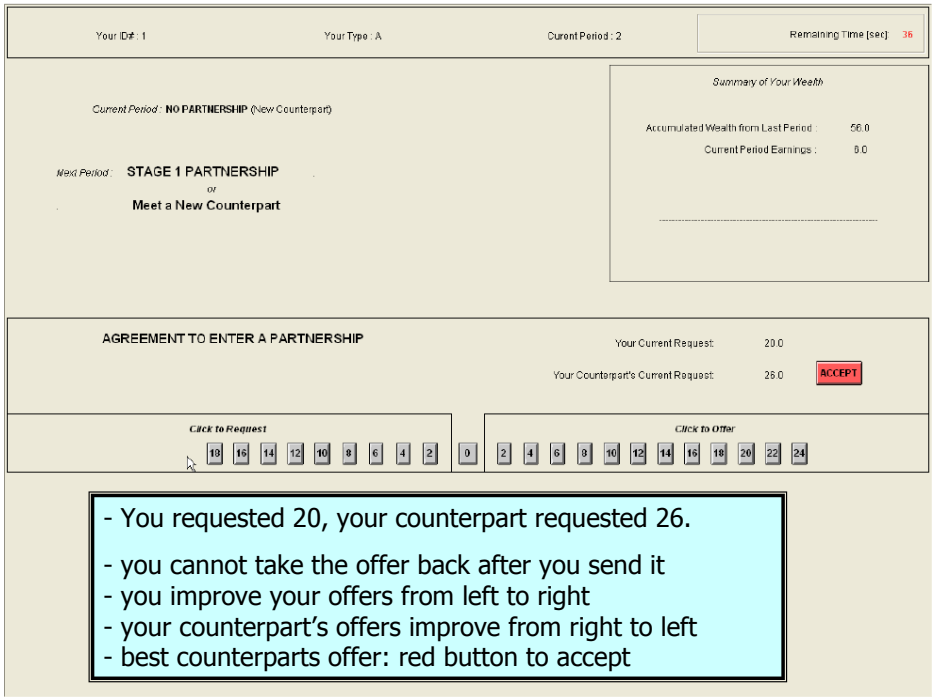
```
treatment "P03.ztt"{
        background{
                table globals{
                }
                table subjects{
                }
                table summary{
                }
                table contracts{
                }
                table session{
                }
                numsubjects = 5;
                numgroups = 1;
                numpracticeperiods = 0;
                numactualperiods = 2;
                exchangerate = 0.5;
                starteendowment = 0;
                showupfee = 10;
                noAutoscope = FALSE;
                v2integervars = TRUE;
                v2booleanvars = TRUE;
                firstBoxesonTop = FALSE;
                showupfeeawaytext = "You are making losses. Do you want to invest the showupfee in order to continue?";
                showupfeeawayyestext = "Yes";
                showupfeeawaynotext = "No";
                moneyawaytext = "You are  making losses. Do you want to continue?";
                moneyawayyestext = "Yes";
                moneyawaynotext = "No";
                bancruptwaittext = "Please wait until the experimenter unlocks your PC.";
                program{
                        table = globals;
                        do{
                                //Parameters

                                SUBS = subjects.count();
                                GRPS = roundedown( SUBS / 2, 1);

                                TimeStage1 = 30;
                                TimeStage2 = 30;

                                CC =   5;       CD  =  0;
                                DC =   8;       DD  =  2;
                        }
                }
                screen action{
                        usebg = TRUE;
                        withalertscreen = FALSE;
                        noalertscreen = FALSE;
                        headerbox "header"{
                                hasframe = TRUE;
                                height = 10%;
                                left = 30%;
                                top = 0p;
                                showPeriods = TRUE;
                                showNumPeriods = TRUE;
                                periodtext = "Period";
                                periodoftext = " of";
                                practiceperiodprefix = "Trial ";
                                showtime = TRUE;
                                timestr = "Remaining time [sec]:";
                                pleasedecidetext = "Please reach a decision.";
                        }
                        standardbox "subject's Name"{
                                hasframe = FALSE;
                                width = 30%;
```



Marriage Game Design

- Two types of subjects (A & B) are are randomly paired with each other.
- Task:
  - Their task in period 1 (also every time when they are matched with new counterparts in later periods) is to decide if they want to <u>enter</u> a partnership starting from next period.
  - Once a partnership is formed, the task in each period is to decide if they want to <u>stay together</u> with the same partner for at least one more period.
- Entering, continuing and terminating a partnership can be facilitated by negotiable transfer payments.
  - Either party can offer/request some payment to/from his/her counterpart





- You requested 20, your counterpart requested 26.
- you cannot take the offer back after you send it
- you improve your offers from left to right
- your counterpart's offers improve from right to left
- best counterparts offer: red button to accept

Marriage Game Design

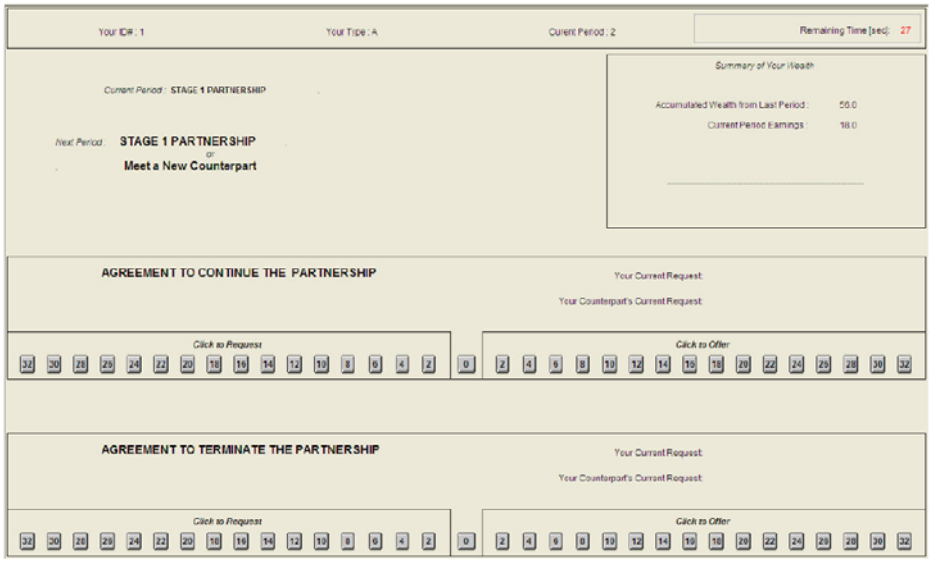
- **The "unattached" (singles)**

  Agreement => form a marriage
  No agreement => matched with new counterparts next period.

- **The "attached" (married)**

  Agreement => continue together
  (possibly under different payoffs)
  Agreement => to divorce (=new counterparts)
  No agreement=>
  Unilateral Divorce Law: matched with new counterparts next period.
  Consensus Divorce Law: stay together

- **10% discounting of future utilities**



Marriage Game Design

- Partnership deteriorates
  (with exogenous probability 2/9)
  Stage 1 is better than Stage 2
  Stage 2 is better than Stage 3
- Tensions increases…

Balanced (Unbalanced) Payoff Structure

	A	B
Single	6	6
Stage 1	18	10
Stage 2	3(15)	15(3)
Stage 3	4	4

Marriage Game Design

- Partnership deteriorates
  (with exogenous probability 2/9)
  Stage 1 is better than Stage 2
  Stage 2 is better than Stage 3
- Tensions increases…

- **10% Discounting**
  (die ….
  …and reborn as a single
  in a new life [same game])

- **… 10% =>**
  with 11 or more pairs we could guarantee that (divorced) agents would be matched to different people.

Balanced (Unbalanced)
Payoff Structure

	A	B
Single	6	6
Stage 1	18	10
Stage 2	3(15)	15(3)
Stage 3	4	4

Marriage Game Design

- **10% =>**
  with 11 or more pairs we could guarantee that (single) agents would be matched to different counterparts.

TWO SINGLE PEOPLE => **This pair dies with 10% prob.**
1^+ out of remaining 10^+ pairs dies as well

ZERO OR MORE THAN TWO SINGLE PEOPLE => **Two pairs die with 50%+ probability**

Marriage Game Design

PHASE ONE (35 periods)
(NEW PEOPLE BORN)

- **10% =>**
  with 11 or more pairs we could guarantee that (single) agents would be matched to different counterparts.

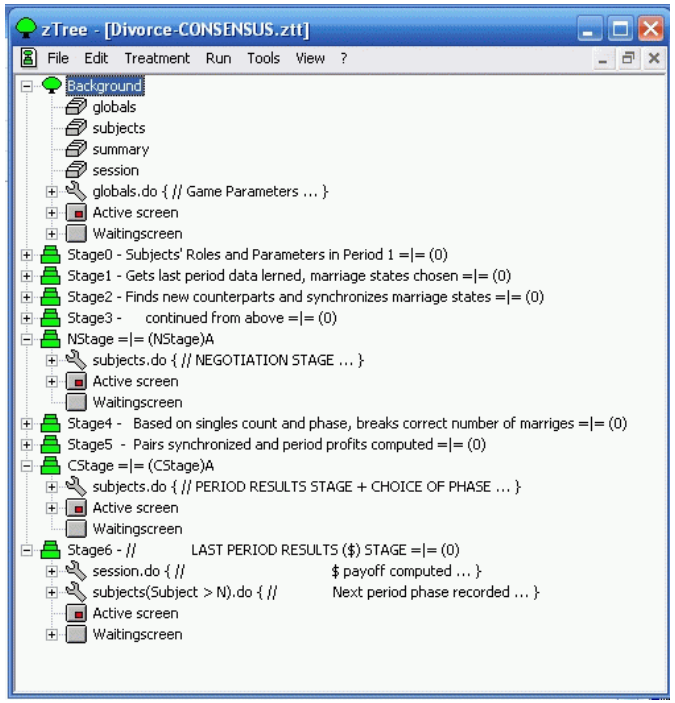
TWO SINGLE PEOPLE => **This pair dies with 10% prob.**
1^+ out of remaining 10^+ pairs dies as well

ZERO OR MORE THAN TWO SINGLE PEOPLE => **Two pairs die with 50%+ probability**

PHASE TWO (GAME ENDS)

TWO SINGLE PEOPLE => **This pair dies with 10% prob.**
1^+ out of remaining 10^+ pairs dies as well

ZERO OR MORE THAN TWO SINGLE PEOPLE => **All pairs die with 10%+ probability, GAME ENDS**



Each period, matching took several seconds to compute.

zLeafs can only wait so long before they crash.

Here the program was cut into pieces and six stages with empty screens we added.

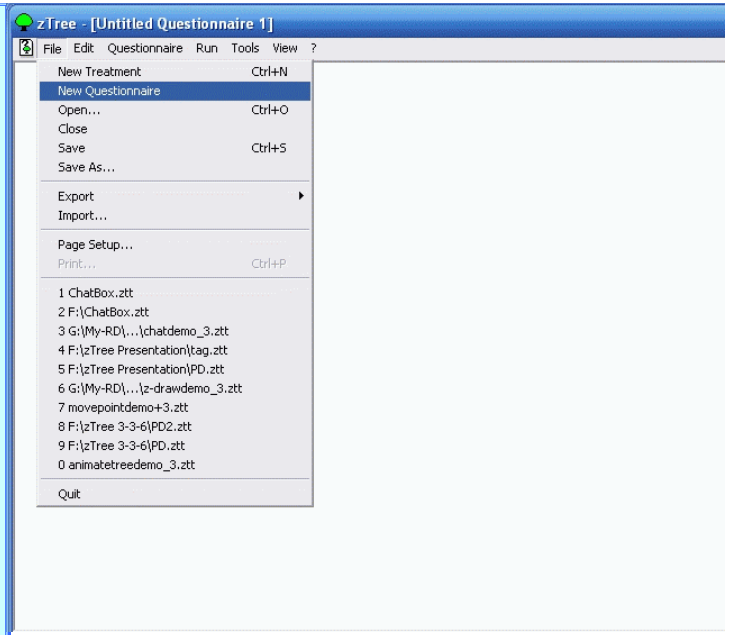
Marriage Game Design

Outline

1. Introduction
2. Simple Program (PD)
3. **Simple Questionnaire**
4. Design
   (From Standard Box to Contracts, Chat and Graphics)
5. More Tips and Tricks and Examples
6. VRPD Game

Questionnaires

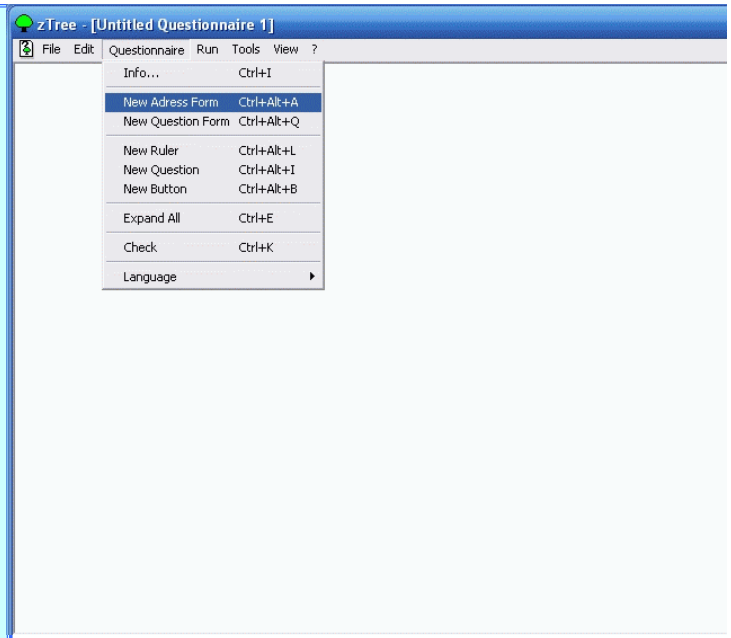
- You can run a questionnaire at the end of an experiment
- Questionnaires can be simple, just names and emails, or involve more complicated surveys



Questionnaires

New Address Form

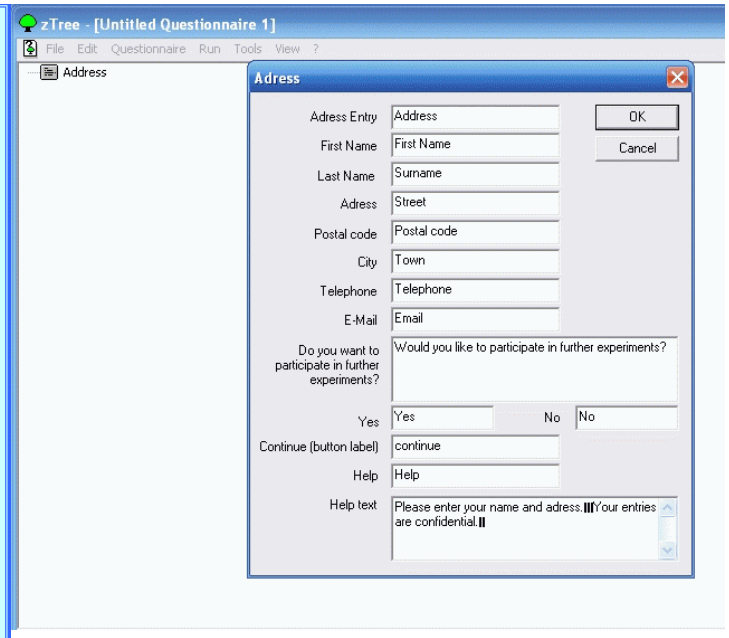
Double-click on "Address" and
- delete what you do not need
- change to what you do need



Questionnaires

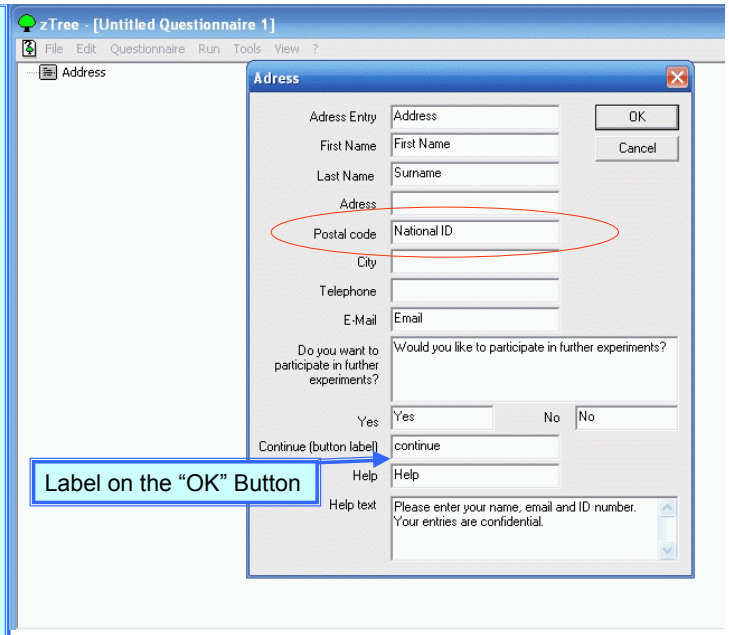
New Address Form

Double-click on "Address" and
- delete what you do not need
- change to what you do need



Questionnaires

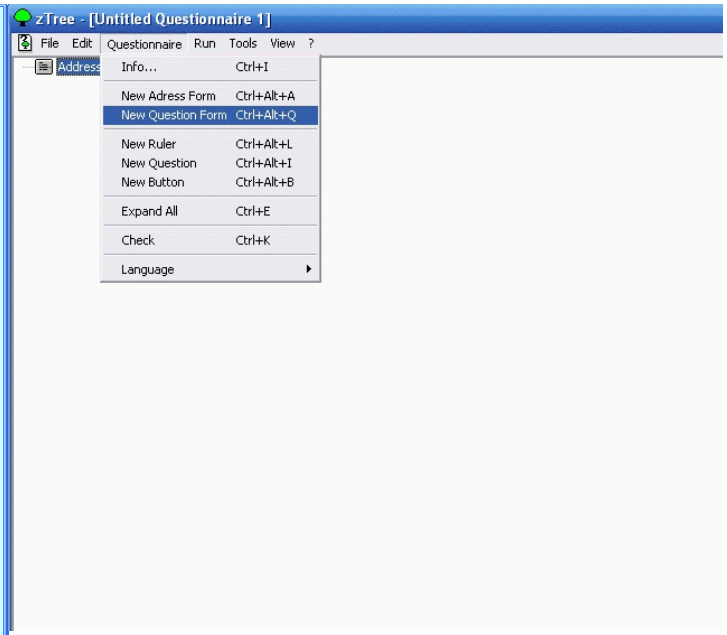
What you enter into the blank will correspond to what the header on the answer blank will read; for example, you can use the "Postal Code" blank to gather information about National ID



Questionnaires

Select "Adress", then go to **Questionnaire→ New Question Form**

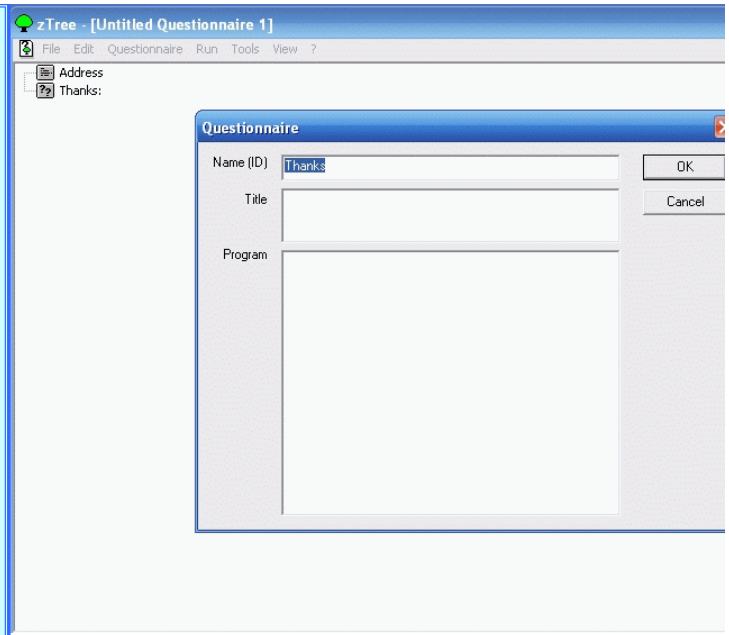
You can add items to this question form, just like you do with a box in a normal treatment



Questionnaires

Select "Adress", then go to **Questionnaire→ New Question Form**

You can add items to this question form, just like you do with a box in a normal treatment

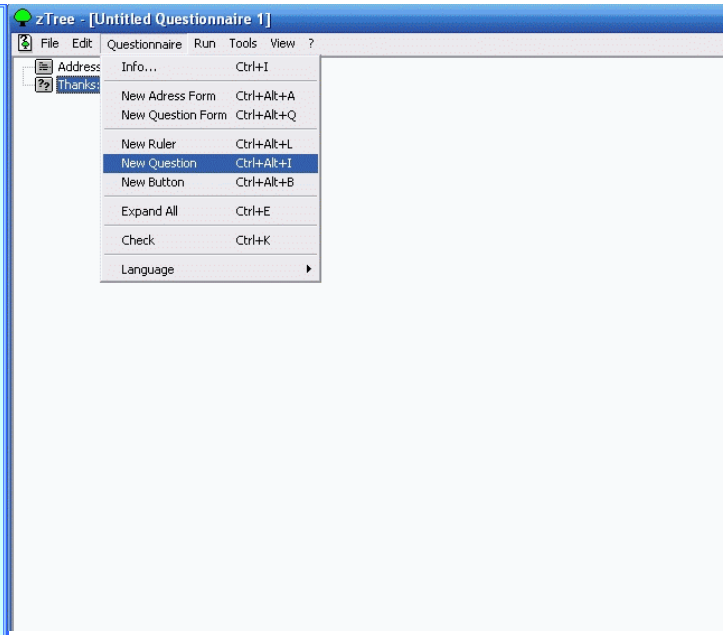


Questionnaires

Beside questions, typical information to display might include:

The variable FinalProfit – total earnings from the experiment, not including the show-up fee

The variable ShowUpFee
Create items (with labels) to display these variables

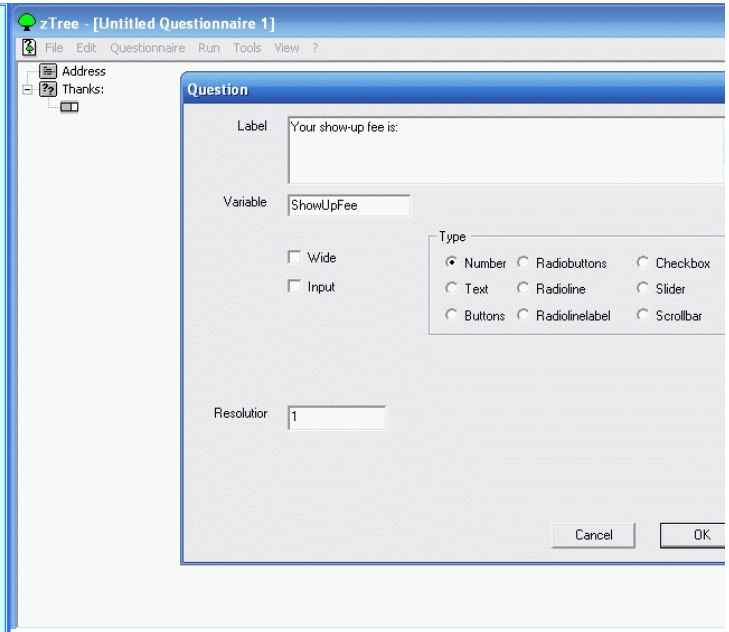


Questionnaires

Beside questions, typical information to display might include:

The variable FinalProfit – total earnings from the experiment, not including the show-up fee

The variable ShowUpFee
Create items (with labels) to display these variables

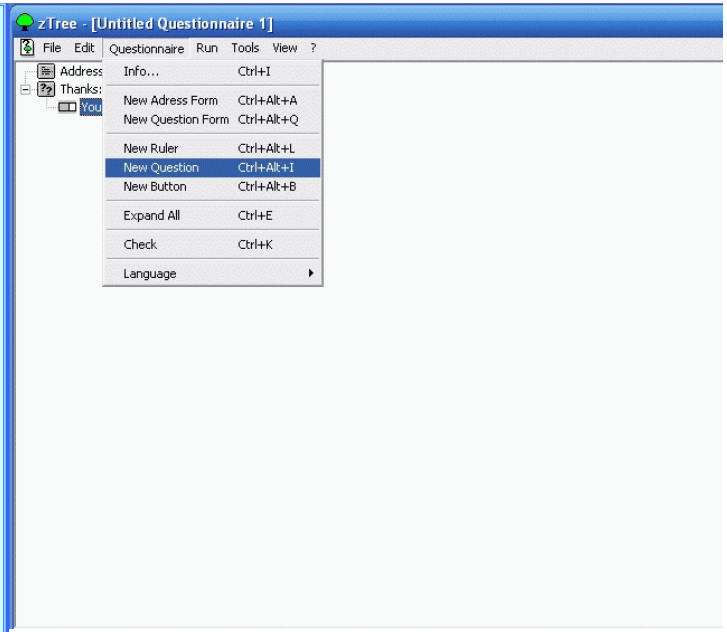


Questionnaires

Beside questions, typical information to display might include:

The variable FinalProfit – total earnings from the experiment, not including the show-up fee

The variable ShowUpFee
Create items (with labels) to display these variables

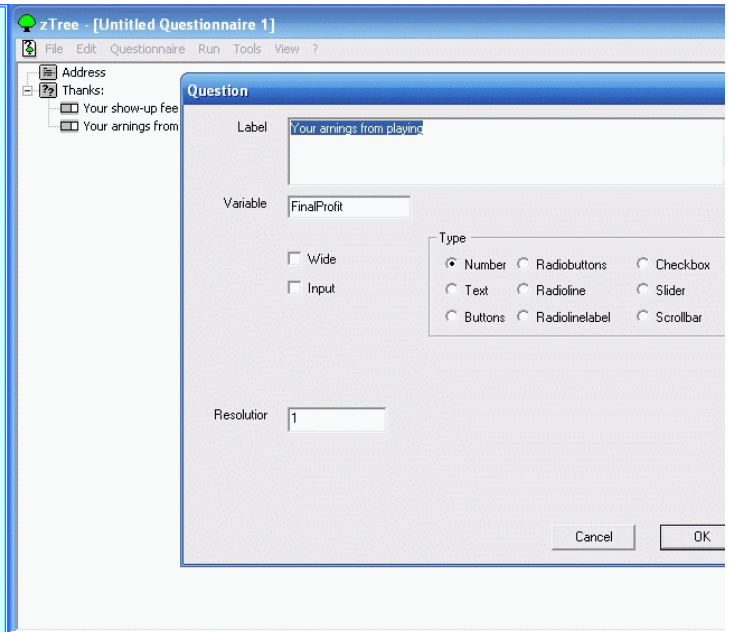


Questionnaires

Beside questions, typical information to display might include:

The variable FinalProfit – total earnings from the experiment, not including the show-up fee

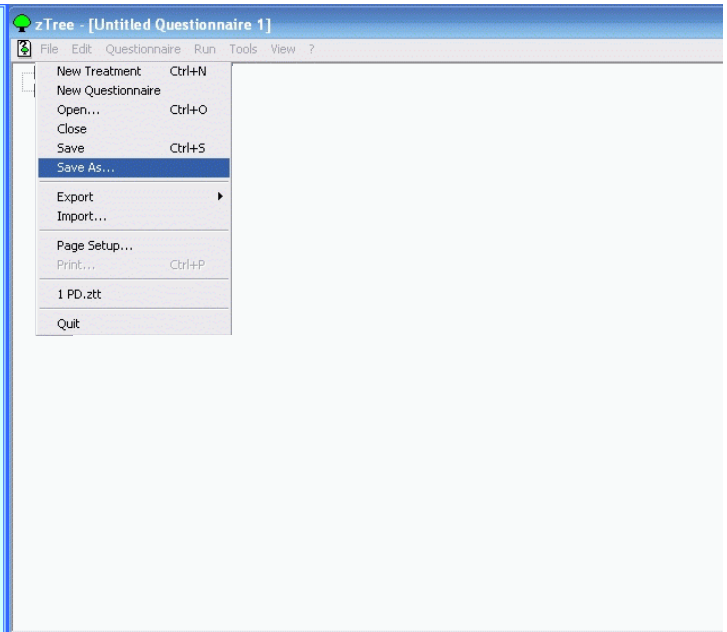
The variable ShowUpFee
Create items (with labels) to display these variables



Questionnaires

Save it (Short.ztq)

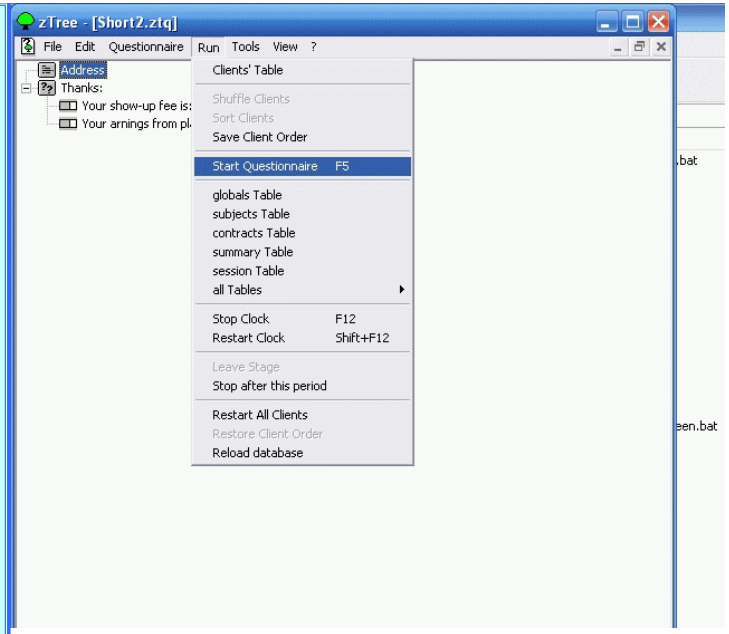
Run it

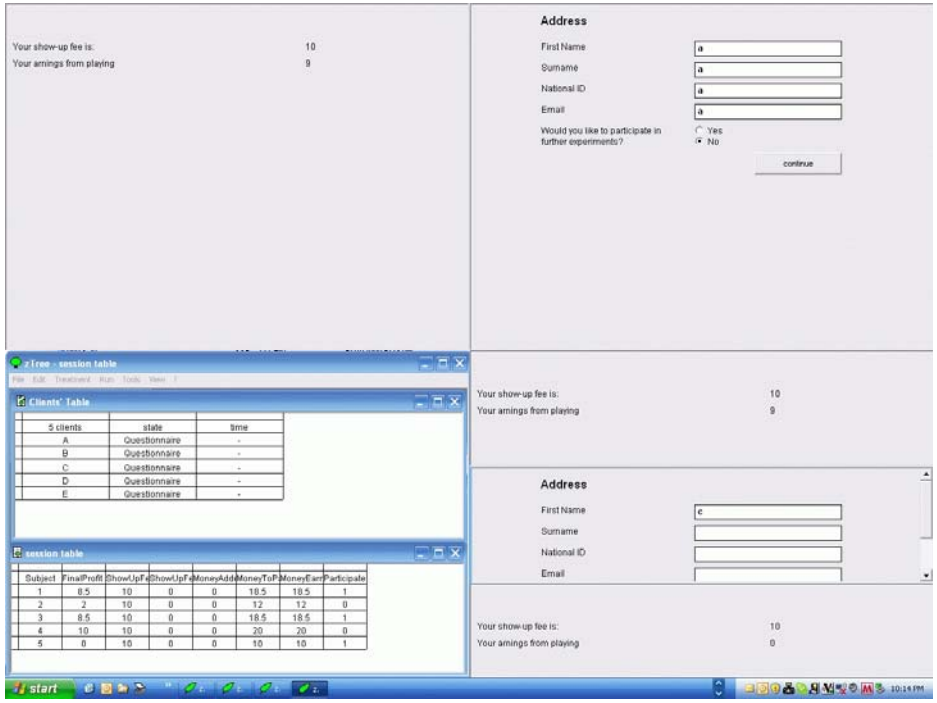


Questionnaires

Save it (Short.ztq)

Run it





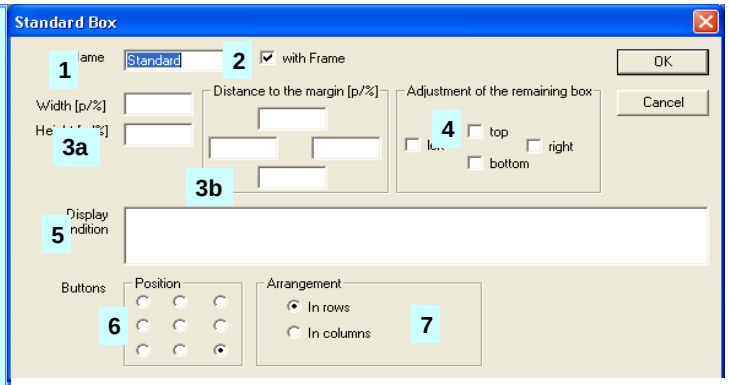
Outline

1. Introduction
2. Creating a Simple Program (PD)
3. Creating a Simple Questionnaire
4. **More on Design**

   (From Standard Box to Contracts, Chat and Graphics)
5. More Tips and Tricks and Examples
6. VRPD Game

DESIGN

Standard Box



1. **Label of the box (not shown to subjects)**
2. **Frame (with it, a box overlaps older graphics)**
3. **Size of the box, in points or percent of the remaining screen**
   Distance away from the (remaining) screen edge in points or percent
4. **Adjustment of the remaining box (whether to "cut off" the screen above, below, to the left, or to the right of the current box**
5. **Display Condition (if present, Boolean expression that must be true in order for box to be shown)**
6. **Button Position (where to place buttons in this box)**
7. **Arrangement (how to arrange buttons)**

DESIGN

Standard Box Item

Item

Label: This text will be displayed to the subjects

Variable: This variable will be displayed as output

Layout: Here you indicate how many decimal places

☐ Input

OK

Cancel

DESIGN

Standard Box Item

Item

Label: Your endowment is:

Variable: Endowment

Layout: 1

Embedded Variables:

```
<>Your income is < profit |!text: 0="small"; 80 = "large";>
```

- Tells program that there will be embedded variables in this text
- Regular text
- Variable name
- Embedded Variable
- Layout

DESIGN

Standard Box Item

Item

Label: <> Your endowment is <Endowment|1> francs, you <Endowment|!text: 0="lost a lot of money"; 100="did great">

Layout	output variable
2	6
!radio: 1 = "86.8"; 24 = "102.8";	86.8 / 102.8
!radioline: 0="zero";5="five"; 6;	zero … five
!slider: 0 ="A"; 100= "B"; 101;	A … B
!scrollbar: 0="L";100= "R";101;	L … R
!checkbox:1="check me";	check me
!text: 2 = "two"; 3 = "three"; 5 = "five"; 7 = "seven"; 11 = "eleven";	seven
!button: 1 = "accept"; 0 = "reject";	accept

DESIGN

Standard Box Item

Item

Label: <> Your endowment is <Endowment|1> francs you <Endowment|!text: 0="lost a lot of money"; 100="did great">

Variable:

Layout:

WARNING the sentence is placed in and if later the number changes and does not fit then only some digits are shown…

☐ Input

OK

Cancel

DESIGN

Standard Box Item

Item

Label: <> Your endowment is <Endowment|1> francs <Endowment|!text: 0="lost a lot of money"; 100="did {rtf \b great }">

OK | Cancel

Variable

Layout

WARNING: The sentence is placed in and if later the number changes and does not fit then only some digits are shown…

☐ Input

{rtf_} allows for:	\fsN	\cfN	\sub	\super
\b	\b0	\i	\i0	\u	\u0
\tab	\par	\bullet	\line	~~\strike~~	
\ql		\qc		\qr

DESIGN

Standard Box Item

Item

Label: Choose an amount to contribute:

OK | Cancel

Variable: Contribution

Layout: 1

☑ Input

Minimum: 0

Maximum: 20

☐ Show value (value of variable or default)

☐ Empty allowed

Default: 0

DESIGN

Standard Box Item

Item

Label

WARNING: Variable is recorded in a table only after a regular button is hit

OK | Cancel

Layout	input variable
2	6
!radio: 1 = "86.8"; 24 = "102.8";	86.8 / 102.8
!radioline: 0="zero";5="five"; 6;	zero … five
!slider: 0 ="A"; 100= "B"; 101;	A … B
!scrollbar: 0="L";100= "R";101;	L … R
!checkbox:1="check me";	check me
!text: 2 = "two"; 3 = "three"; 5 = "five"; 7 = "seven"; 11 = "eleven";	seven
!button: 1 = "accept"; 0 = "reject";	accept / reject

DESIGN

Standard Box Item

Button

Buttons cannot go before items.

You have to use buttons to record data in tables

You can use buttons to exit stages or execute programs

Button

Name: OK

OK | Cancel

☐ No record created or selected

☐ Clear entry after OK

Leave Stage
- ◉ Yes
- ○ No
- ○ Normal (i.e. stage is not left after click if stage is left after timeout and button is contained in contract creation or selection box)

Color
- ◉ Automatic
- ○ Gray
- ○ Red

Text without variables are displayed centered

shows variable a 17

shows variable b 9

text in label column without variable

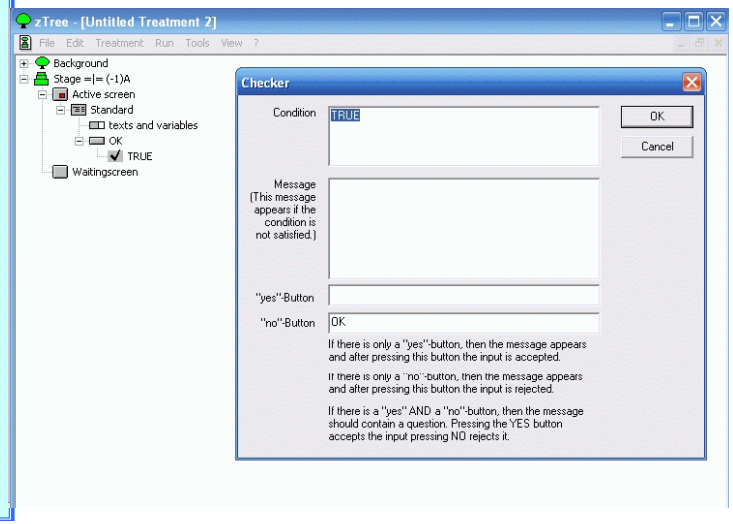
please enter a value for c

OK

- Standard
  - Text without variables are displayed centered
  - shows variable a: OUT(a)
  - shows variable b: OUT(b)
  - text in label column without variable: OUT(_)
  - please enter a value for c: IN(c)
  - OK

DESIGN

Standard Box
- Item
  - Checker
- Button
  - Checker



DESIGN

Standard Box
- Item
  - Checker
- Button
  - Checker
- *Box*

Container Box
(Standard Box for Boxes only)

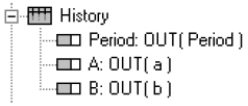
Standard Box	Contract Creation Box
Container Box	Contract List Box
History Box	Contract Grid Box
Grid Box)	Message Box
Calculator Box	Multimedia Box
Help Box	Chat Box
Header Box	Plot Box

DESIGN

Standard Box
- Item
  - Checker
- Button
  - Checker
- *Box*

Container Box

History Box



Period	A	B
3	36	12
4	45	24
5	51	21
6	37	11
7	63	28

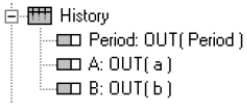
DESIGN

Standard Box
- Item
  - Checker
- Button
  - Checker
- *Box*

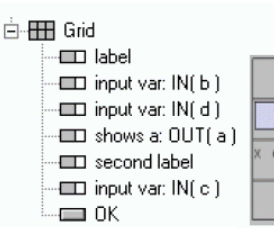
Container Box

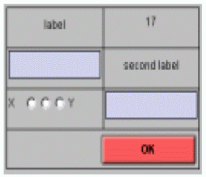
History Box

Grid Box *(table)*

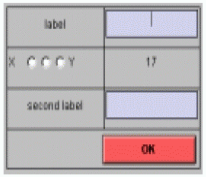


Period	A	B
3	36	12
4	45	24
5	51	21
6	37	11
7	63	28





column by column



row by row

DESIGN

Standard Box
- Item
  - Checker
- Button
  - Checker
- *Box*

Container Box

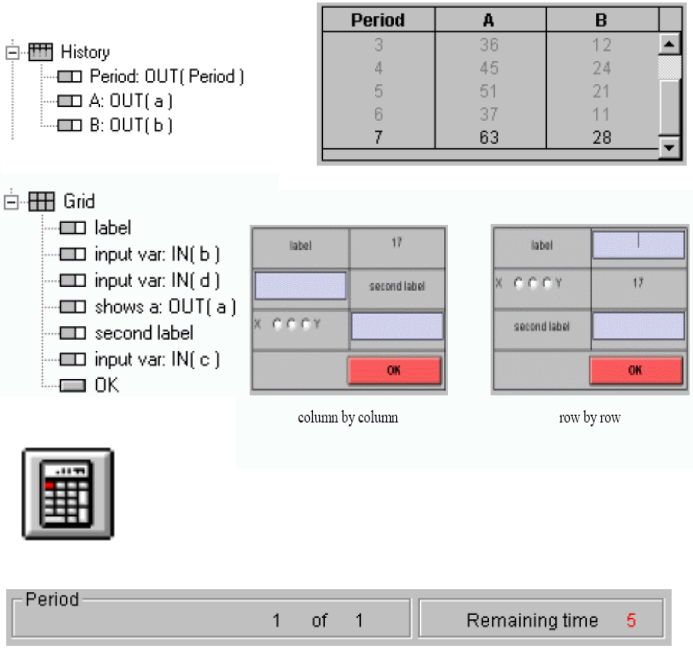
History Box

Grid Box *(table)*

Calculator Box

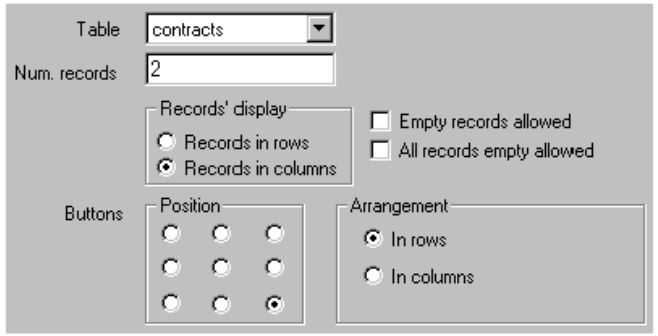
Help Box *(text)*

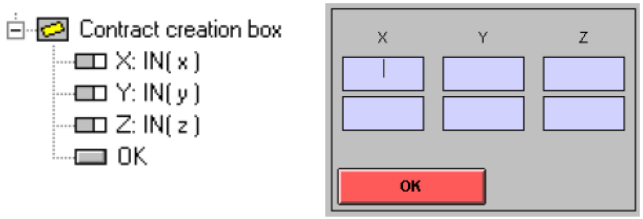
Header Box



DESIGN

Contract Creation Box



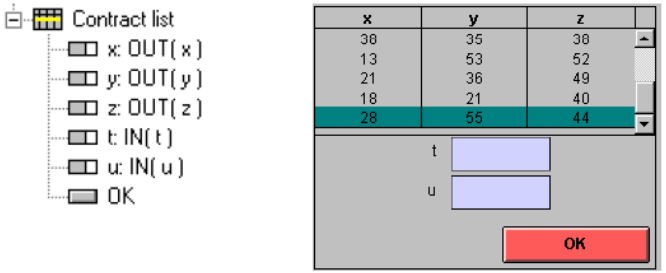


box in z-Leaf

DESIGN

Contract Creation Box

Contract List Box

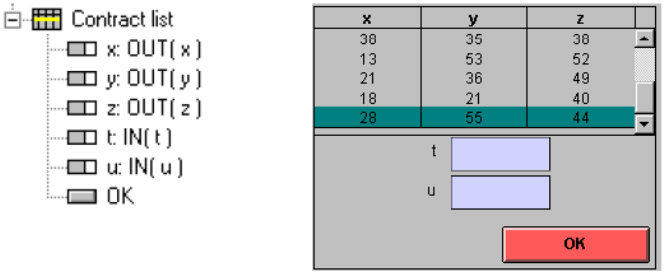


The button executes program that may change the selected record.

DESIGN

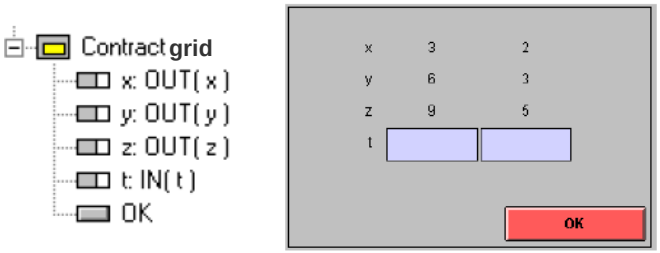
Contract Creation Box

Contract List Box



The button executes program that may change the selected record.

Contract Grid Box



Records can be in columns or in rows.

DESIGN

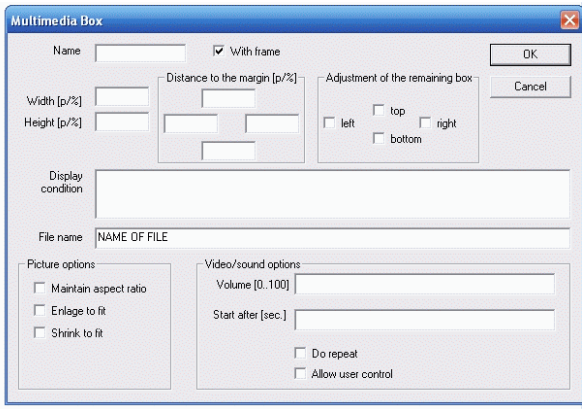
Message Box

Multimedia Box

You can place at most one message box per screen. If zTree sends a message (say you entered a wrong value to Item), it will appear in the box and subject must click OK to continue. This way, only one OK is needed for multiple meassages.

Image: jpg, gif, png, bmp.
Movie: mpg, avi.
Sound: wav, mp3.

Do not play movie or sound in waiting screen



DESIGN

Message Box

Multimedia Box

Chat Box

You can place at most one message box per screen. If zTree sends a message (say you entered a wrong value to Item), it will appear in the box and subject must click OK to continue. This way, only one OK is needed for multiple meassages.

Image: jpg, gif, png, bmp.
Movie: mpg, avi.
Sound: wav, mp3.

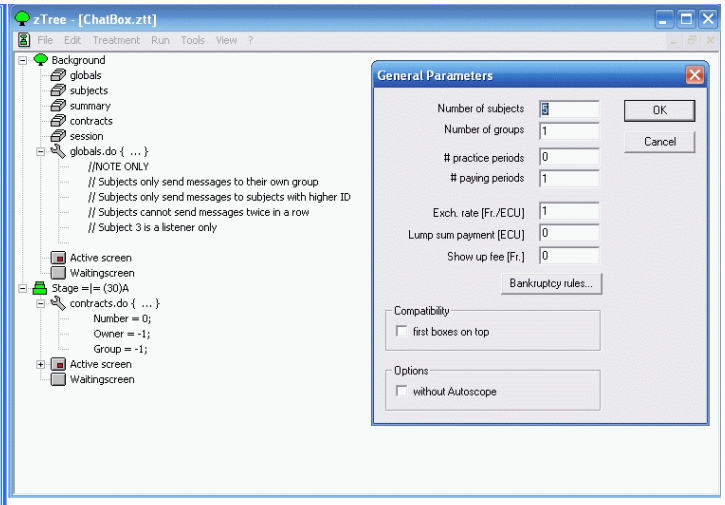
Allows subjects to send text messages to each other. The following example program will illustrate its use:

We will create a new program such that:

// Subjects only send messages to their own group
// Subjects only send messages to subjects with higher ID
// Subjects cannot send messages twice in a row
// Subject 3 is a listener only

DESIGN

Chat Box

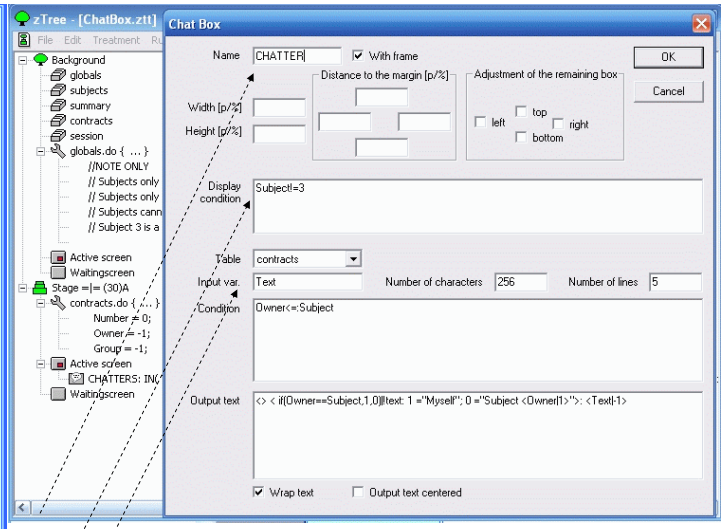


(1) Add the first stage

(2) Add a "program" with contracts table there:
Number = 0;
Owner = -1;
Group = -1;

DESIGN

Chat Box



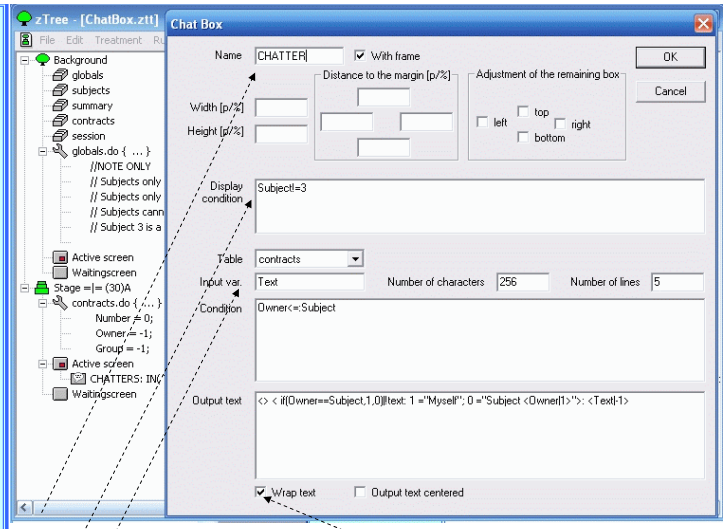
(3) In Stage's Active Screen Create New Chat Box: CHATTER

(4) Subject!=3 (3 is only a listener, this is not his window)

(5) Name Input Variable (=Text), limit characters to 256 (so excel can remember) and allow for 5 lines of text input.

DESIGN

Chat Box



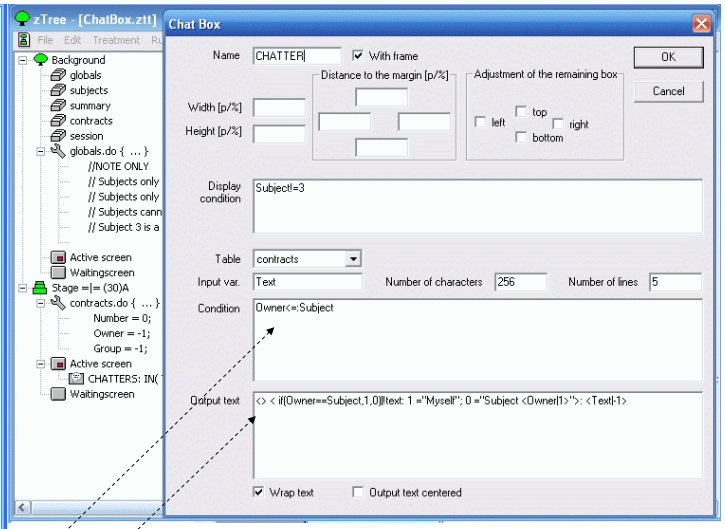
(3) In Stage's Active Screen Create New Chat Box: CHATTER

(4) Subject!=3 **(3 is only a listener, this is not his window)**

(5)Input Var (e.g. Text), limit characters to 256 (excel can remember) and allow for 5 lines of text input and wrap.

DESIGN

Chat Box

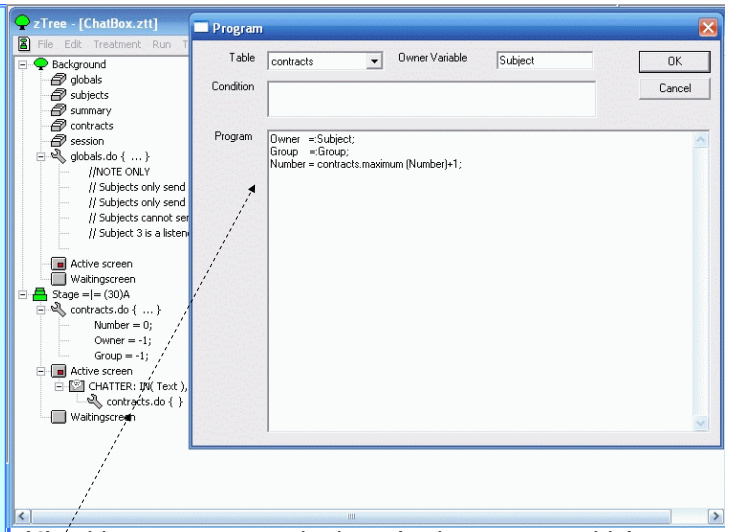


(6) Owner<=Subject will limit observed messages to those sent by yourself and subject with higher ID number

(7) **<> < if(Owner == Subject, 1, 0)|!text:**
1 ="Myself";
0 ="Subject <Owner|1>">: <Text|-1>

DESIGN

Chat Box

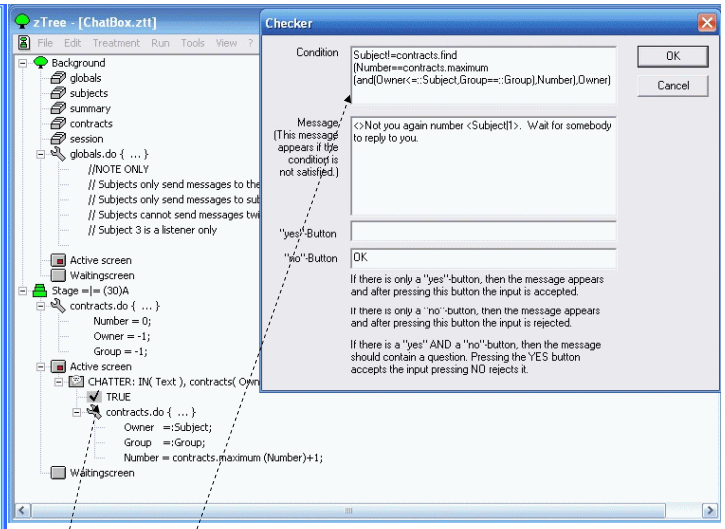


(6) Add a program into the box (with contracts table)

Owner =:Subject;
Group =:Group;
Number = contracts.maximum (Number)+1;

DESIGN

Chat Box

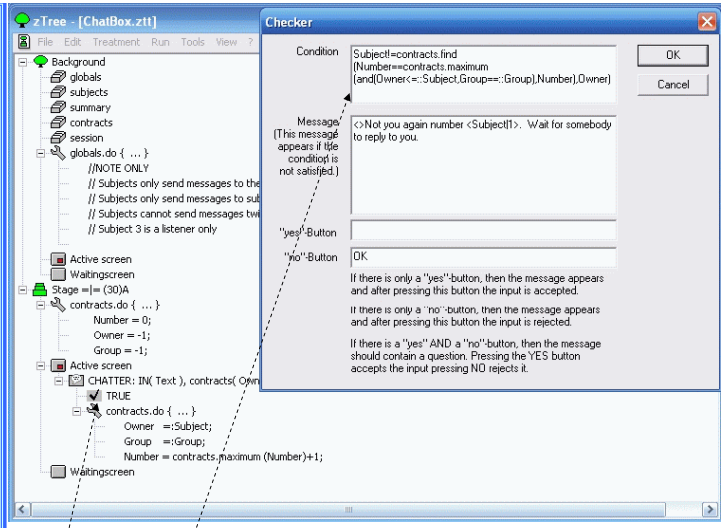


(6) Add a checker into the box (so subjects know why…)

(7) **Find the owner of the latest contract from same group with larger ID:** Subject!=contracts.find(Number==
contracts.maximum(and(Owner<=::Subject,Group==::Group)
Number), Owner)

DESIGN

Chat Box

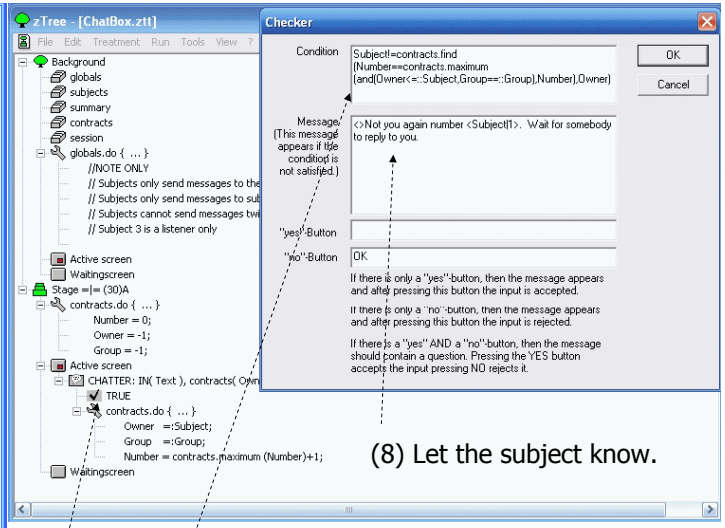


(6) Add a checker into the box (so subjects know why...)

(7) **Find the owner of the latest contract from same group with larger ID:** Subject!=contracts.find(Number==contracts.maximum(and(Owner<=::Subject,Group==::Group) Number), Owner)

DESIGN

Chat Box



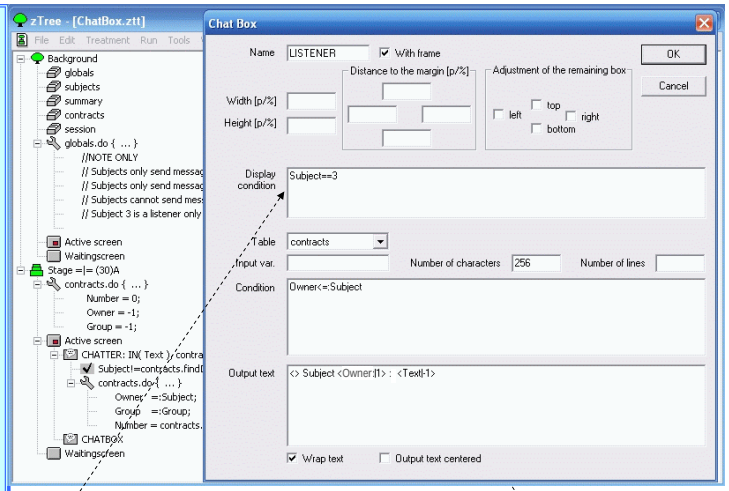
(8) Let the subject know.

(6) Add a checker into the box (so subjects know why...)

(7) **Find the owner of the latest contract from same group with larger ID:** Subject!=contracts.find(Number==contracts.maximum(and(Owner<=::Subject,Group==::Group) Number), Owner)

DESIGN

Chat Box

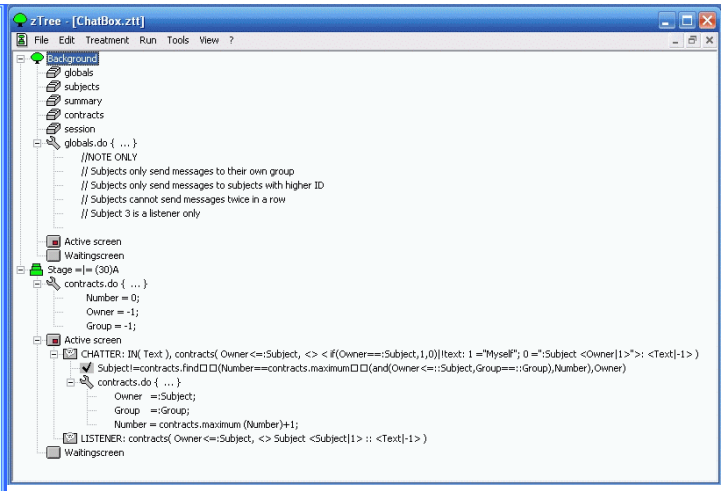


(9) Finally add a listening chat for subject S3

<> Subject <Owner|1> : <Text|-1>

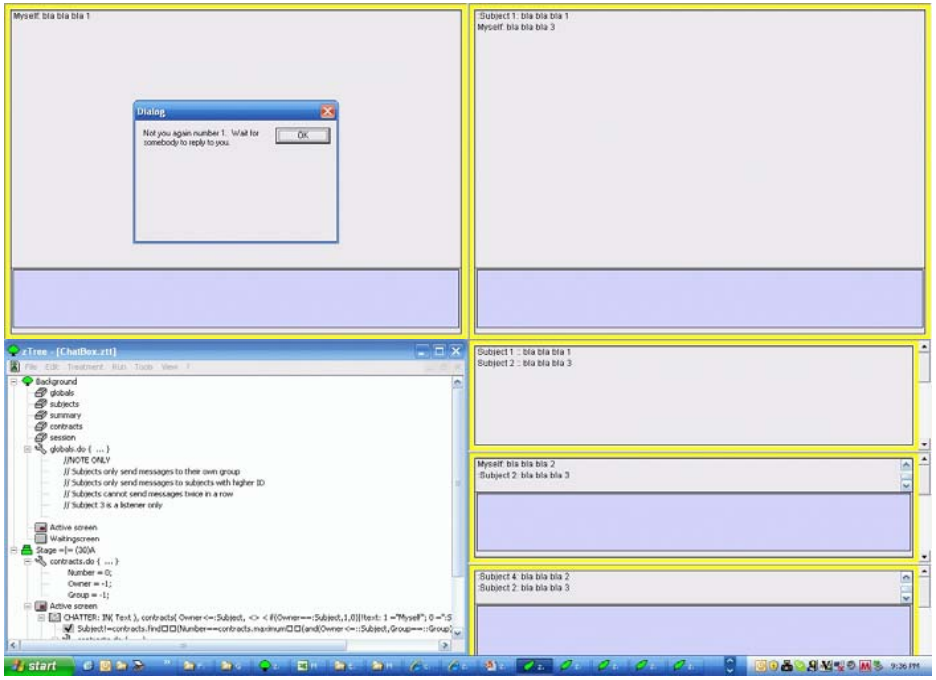
DESIGN

Chat Box

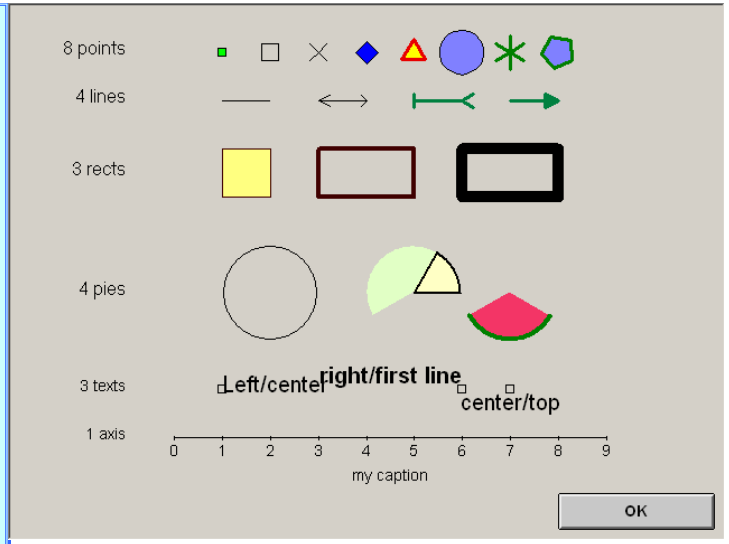


Program Completed. Ready to be tried. Note that

- S1 can only speak once
- S2 can at most speak twice **(before and after S1)**
- **S3 only listens**
- **S4** can at most speak 4x and S5 at most 8x



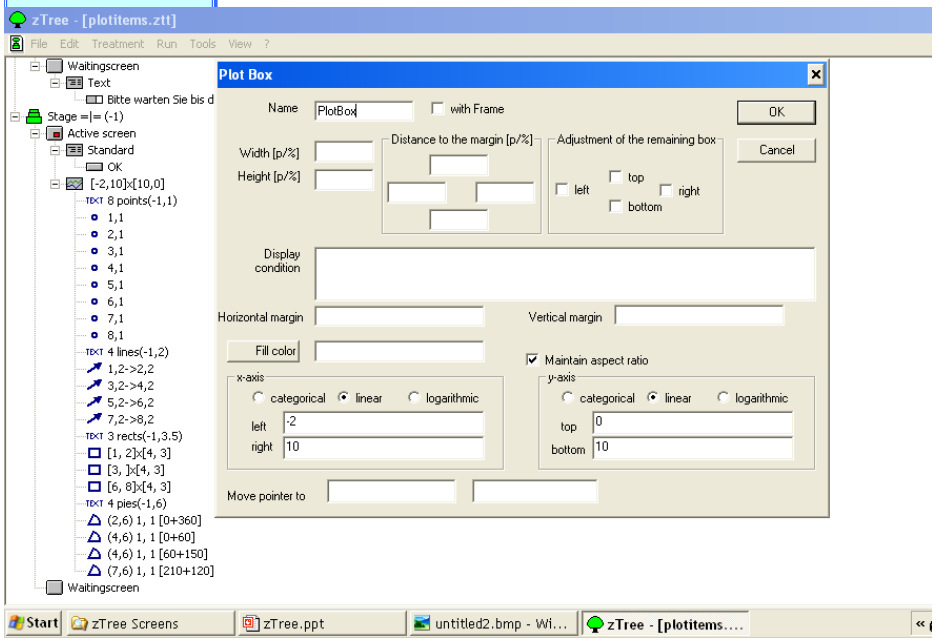
PLOT Box



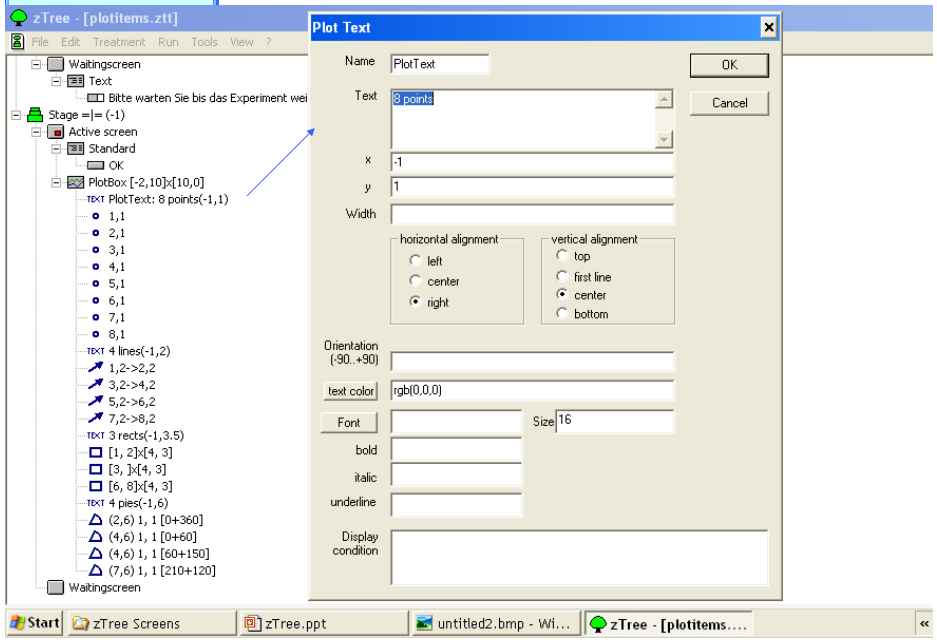
plotitems.ztt : Sample vector graphics items

draw.ztt : Manipulate graphics items

PLOT Box



PLOT Box



PLOT Box

zTree - [plotitems.ztt]

File Edit Treatment Run Tools View ?

- Waitingscreen
  - Text
    - Bitte warten Sie bis das Experiment weiter geht.
- Stage =|= (-1)
  - Active screen
    - Standard
      - OK
    - PlotBox [-2,10]x[10,0]
      - PlotText: 8 points(-1,1)
      - 1,1
      - 2,1
      - 3,1
      - 4,1
      - 5,1
      - 6,1
      - 7,1
      - PlotPoint: 8,1
      - 4 lines(-1,2)
      - 1,2->2,2
      - 3,2->4,2
      - 5,2->6,2
      - 7,2->8,2
      - 3 rects(-1,3.5)
      - [1, 2]x[4, 3]
      - [3,]x[4, 3]
      - [6, 8]x[4, 3]
      - 4 pies(-1,6)
      - (2,6) 1, 1 [0+360]
      - (4,6) 1, 1 [0+60]
      - (4,6) 1, 1 [60+150]
      - (7,6) 1, 1 [210+120]
- Waitingscreen

Plot Point

Name	PlotPoint
x	8
y	1
is a star (not polygon)	
Size	15
Num vertices	5
Start at (angle from x)	-30
Line color	rgb(0.00,0.50,0.00)
Line width	3
Fill color	rgb(0.50,0.50,1.00)
Display condition	

OK Cancel

8 points

4 lines

Start zTree Screens

PLOT Box

zTree - [plotitems.ztt]

File Edit Treatment Run Tools View ?

- Waitingscreen
  - Text
    - Bitte warten Sie bis das Experiment weiter geht.
- Stage =|= (-1)
  - Active screen
    - Standard
      - OK
    - PlotBox [-2,10]x[10,0]
      - PlotText: 8 points(-1,1)
      - 1,1
      - 2,1
      - 3,1
      - 4,1
      - 5,1
      - 6,1
      - 7,1
      - PlotPoint: 8,1
      - 4 lines(-1,2)
      - 1,2->2,2
      - 3,2->4,2
      - 5,2->6,2
      - PlotLine: 7,2->8,2
      - 3 rects(-1,3.5)
      - [1, 2]x[4, 3]
      - [3,]x[4, 3]
      - [6, 8]x[4, 3]
      - 4 pies(-1,6)
      - (2,6) 1, 1 [0+360]
      - (4,6) 1, 1 [0+60]
      - (4,6) 1, 1 [60+150]
      - (7,6) 1, 1 [210+120]
- Waitingscreen

Plot Line

	Point 1	Point 2
Name	PlotLine	
x	7	8
y	2	2
arrow size		12
arrow width	12	
Arrow is closed		
line color	rgb(0.00,0.50,0.25)	
line width	3	
Display condition		

OK Cancel

8 points

4 lines

Start zTree Screens

PLOT Box

zTree - [plotitems.ztt]

File Edit Treatment Run Tools View ?

- Waitingscreen
  - Text
    - Bitte warten Sie bis das Experiment weiter geht.
- Stage =|= (-1)
  - Active screen
    - Standard
      - OK
    - PlotBox [-2,10]x[10,0]
      - PlotText: 8 points(-1,1)
      - 1,1
      - 2,1
      - 3,1
      - 4,1
      - 5,1
      - 6,1
      - 7,1
      - PlotPoint: 8,1
      - 4 lines(-1,2)
      - 1,2->2,2
      - 3,2->4,2
      - 5,2->6,2
      - PlotLine: 7,2->8,2
      - 3 rects(-1,3.5)
      - [1, 2]x[4, 3]
      - [3,]x[4, 3]
      - PlotRectangle: [6, 8]x[4, 3]
      - 4 pies(-1,6)
      - (2,6) 1, 1 [0+360]
      - (4,6) 1, 1 [0+60]
      - (4,6) 1, 1 [60+150]
      - (7,6) 1, 1 [210+120]
- Waitingscreen

Plot Rect

Name	PlotRectangle
x	
y	
width	
height	
Position	3
	6
	8
	4
line color	rgb(0.00,0.00,0.00)
line width	10
fill color	
Display condition	

OK Cancel

3 rects

Start zTree Screens

PLOT Box

4 pies

zTree - [plotitems.ztt]

File Edit Treatment Run Tools

- Waitingscreen
  - Text
    - Bitte warten Sie bis das Experiment w
- Stage =|= (-1)
  - Active screen
    - Standard
      - OK
    - PlotBox [-2,10]x[10,0]
      - PlotText: 8 points(-1,1)
      - 1,1
      - 2,1
      - 3,1
      - 4,1
      - 5,1
      - 6,1
      - 7,1
      - PlotPoint: 8,1
      - 4 lines(-1,2)
      - 1,2->2,2
      - 3,2->4,2
      - 5,2->6,2
      - PlotLine: 7,2->8,2
      - 3 rects(-1,3.5)
      - [1, 2]x[4, 3]
      - [3,]x[4, 3]
      - PlotRectangle: [6, 8]x[4, 3]
      - 4 pies(-1,6)
      - (2,6) 1, 1 [0+360]
      - (4,6) 1, 1 [0+60]
      - (4,6) 1, 1 [60+150]
      - PlotPie: (7,6) 1, 1 [210+120]
- Waitingscreen

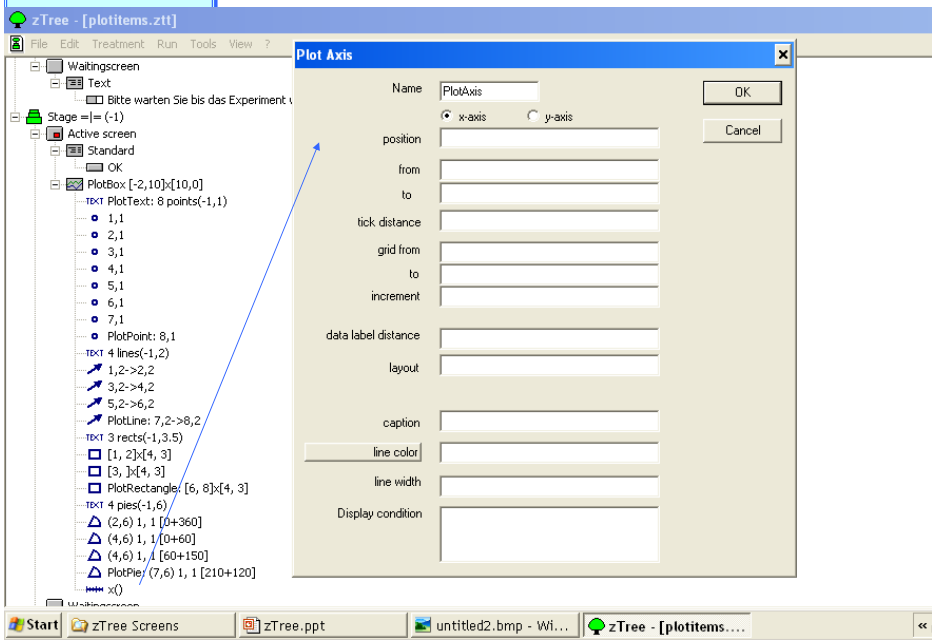
Plot Pie

Name	PlotPie
center x	7
center y	6
radius x	1
radius y	1
start angle [°]	210
angle [°]	120
line color	rgb(0.00,0.50,0.00)
line width	4
do not draw lines to center	
fill color	rgb(0.95,0.21,0.40)
Display condition	

OK Cancel

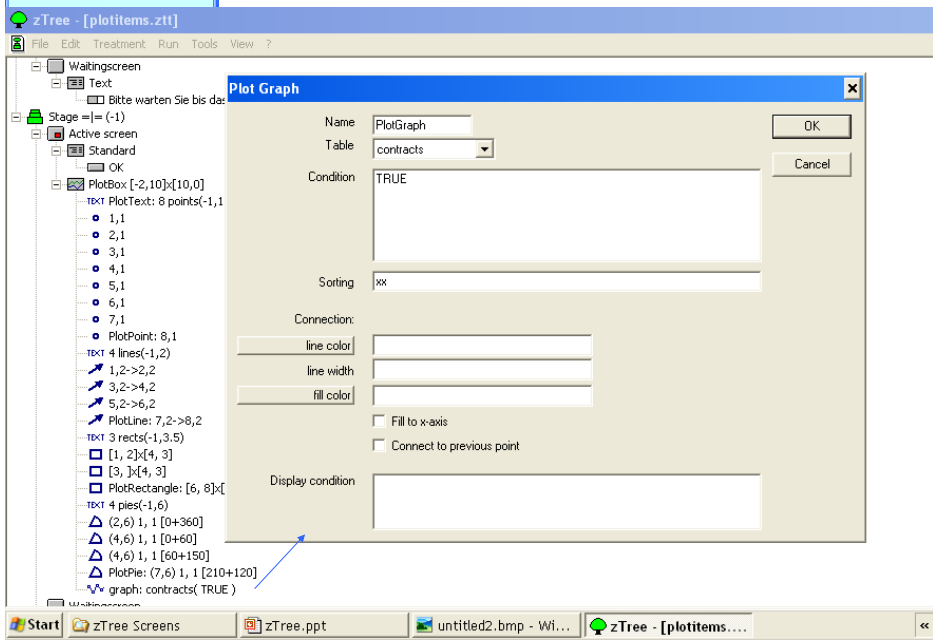
Start zTree Screens zTree.ppt untitled2.bmp - Wi... zTree - [plotitems....

PLOT Box



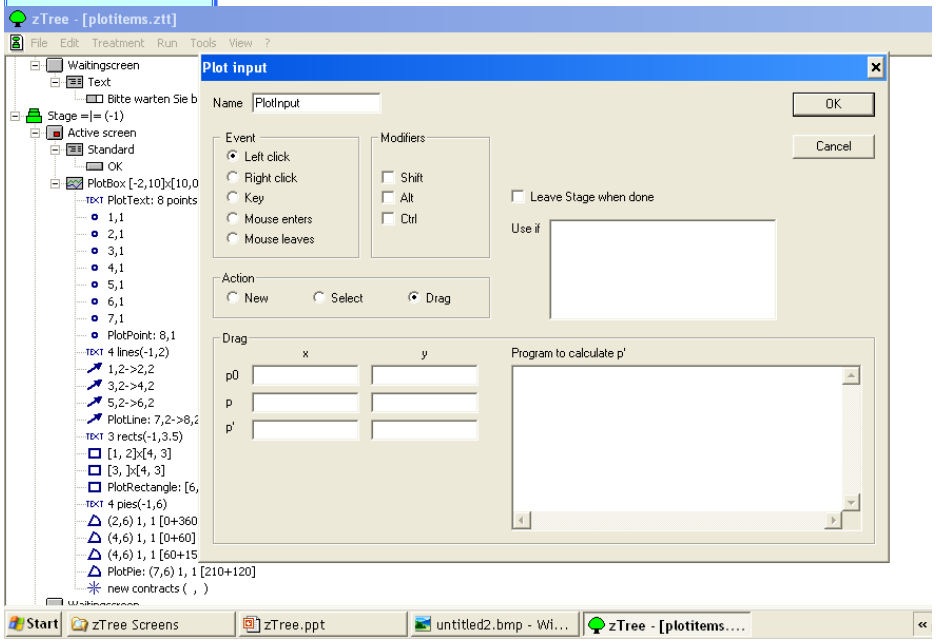
PLOT Box

PLOT GRAPH



PLOT Box

PLOT INPUT



PLOT Box

PLOT INPUT

Plot inputs can be placed into **plot boxes** and **plot items**. They can contain checkers' and programs'.

In principle, plot input works in the following way.

- Click at a new position:
  - The subject clicks at a screen position.
  - The coordinates of the position are stored in the subjects table or in a new contract of the contracts table or another user defined table.
  - The programs in the plot input item are executed in the record where the data is stored.
- Select an object
  - The subjects click at an objects that contains a plot input item that treats the selection.
  - The consequence of the selection is determined by the program contained in the plot input. The program ins executed in the subjects table if it is an **plot item** in a **plot box**. If the plot input is in a **plot item** contained in a **plot graph** then the program is executed in the record representing this object in the **plot graph**. If **plot graphs** nested, the **scope environment**' represents the objects.
- Drag an object
  - The subject clicks at an objects that contains a plot input item that treats the dragging.
  - z-Leaf makes an movable object, which can be dragged.
  - When the object is released, the horizontal and vertical distance are stored in variables. The programmer of the treatment has to provide a program that updated the variables is a way that the data represents the new position.
  - The updated data is transmitted to z-Leaf and draws the object at the new position.

PLOT Box

PLOT INPUT

Plot inputs can be placed into **plot boxes** and **plot items**. They can contain **checkers**' and **programs**'.

In principle, plot input works in the following way.

- Click at a new position:
  - The subject clicks at a screen position.
  - The coordinates of the position are stored in the subjects table or in a new contract of the contracts table or another user defined table.
  - The programs in the plot input item are executed in the record where the data is stored.
- Select an object
  - The subjects click at an objects that contains a plot input item that treats the selection.
  - The consequence of the selection is determined by the program contained in the plot input. The program ins executed in the subjects table if it is an **plot item** in a **plot box**. If the plot input is in a **plot item** contained in a **plot graph** then the program is executed in the record representing this object in the **plot graph**. If **plot graphs** nested, the **scope environment**' represents the objects.
- Drag an object
  - The subject clicks at an objects that contains a plot input item that treats the dragging.
  - z-Leaf makes an movable object, which can be dragged.
  - When the object is released, the horizontal and vertical distance are stored in variables. The programmer of the treatment has to provide a program that updated the variables is a way that the data represents the new position.
  - The updated data is transmitted to z-Leaf and draws the object at the new position.

PLOT Box

PLOT INPUT

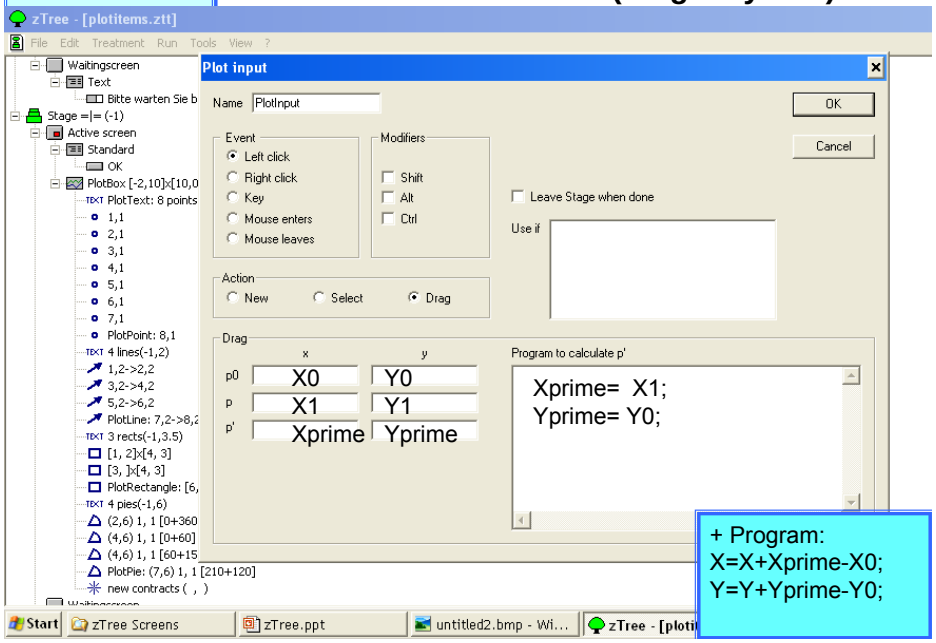
Plot inputs can be placed into **plot boxes** and **plot items**. They can contain **checkers**' and **programs**'.

In principle, plot input works in the following way.

- Click at a new position:
  - The subject clicks at a screen position.
  - The coordinates of the position are stored in the subjects table or in a new contract of the contracts table or another user defined table.
  - The programs in the plot input item are executed in the record where the data is stored.
- Select an object
  - The subjects click at an objects that contains a plot input item that treats the selection.
  - The consequence of the selection is determined by the program contained in the plot input. The program ins executed in the subjects table if it is an **plot item** in a **plot box**. If the plot input is in a **plot item** contained in a **plot graph** then the program is executed in the record representing this object in the **plot graph**. If **plot graphs** nested, the **scope environment**' represents the objects.
- Drag an object
  - The subject clicks at an objects that contains a plot input item that treats the dragging.
  - z-Leaf makes an movable object, which can be dragged.
  - When the object is released, the horizontal and vertical distance are stored in variables. The programmer of the treatment has to provide a program that updated the variables is a way that the data represents the new position.
  - The updated data is transmitted to z-Leaf and draws the object at the new position.

PLOT Box

PLOT INPUT (drag only <-->)



+ Program:
X=X+Xprime-X0;
Y=Y+Yprime-Y0;

Outline

1. Introduction
2. Creating a Simple Program (PD)
3. Creating a Simple Questionnaire
4. More on Design
   (From Standard Box to Contracts, Chat and Graphics)
5. **More Tips and Tricks and Examples**
6. VRPD Game

Storing Information From Previous Periods

- zTree remembers the value of globals and subjects variables from one prior period prior only:

  These previous values are stored in a table called OLD*table*; so, the previous variables in the subjects table can be called from the table called OLDsubjects

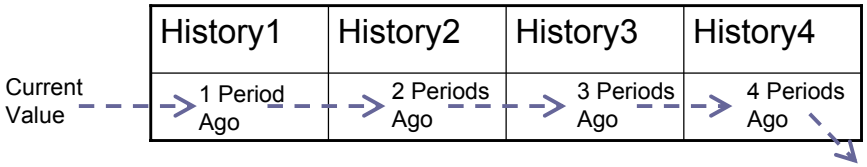
- Longer histories?
  - Use summary or session's table
  - Record a sequence of pat periods

Storing Information From Previous Periods

- Put the following at the beginning of the treatment:

```
i=1;
while (i <maxi) do {History1[i]=OLDsubjects.find(same(Subject),Current[i]);
                    History2[i]=OLDsubjects.find(same(Subject),History1[i]);
                    History3[i]=OLDsubjects.find(same(Subject),History2[i]);
                    History4[i]=OLDsubjects.find(same(Subject),History3[i]);
                    i=i+1}
```

// Current[i] = *CurrentValues*;

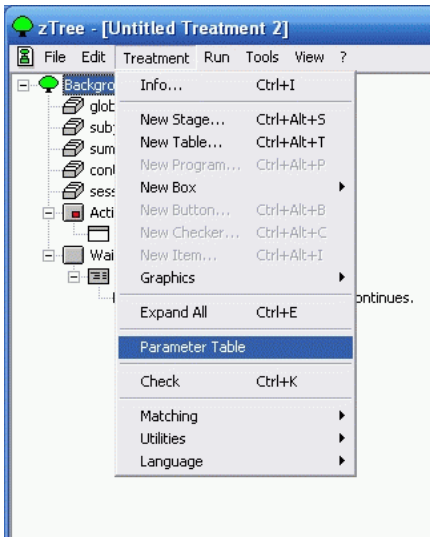


Storing Information From Previous Treatments

- All variables must be defined (given values) in programs before they can be read. The exceptions are session variables that are mentioned in Background.
  Click on Session table, insert sessions variable name (that was use in previous treatment) in the bottom box. Variable will be remember the value from the previous treatment.

Pauses and Popup Windows Between Periods

- Go to Treatments => Parameters Table and click on a chosen period. Then you can pause the session by sending a prompt (you can also change parameters for the chosen period)



Pauses and Popup Windows Between Periods