# How To: Counterbalance Blocks

## Pre-requisite: How to Randomise Blocks

Counterbalancing your blocks is really just an extension of the blocking scenario, except that you set the blocks to operate in a predetermined order rather than leaving PsychoPy to randomise them.

PsychoPy currently cannot handle the ordering for you - you need to decide how to create the order of presentation and how to assign participants.

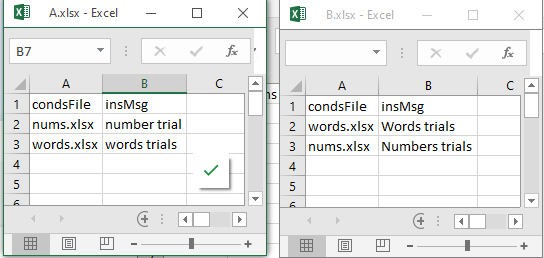
You need a conditions file per conditions order (e.g. A first -> B second Or B-> first A second), then determine which file will be used for this group and use that file in the blocks loop

Remember to set the blocks loop to be **sequential** rather than random to preserve the order you have set

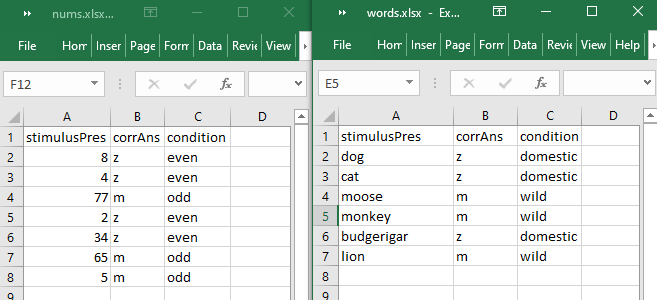
The easiest way is by hand at the start of the run for the participant.

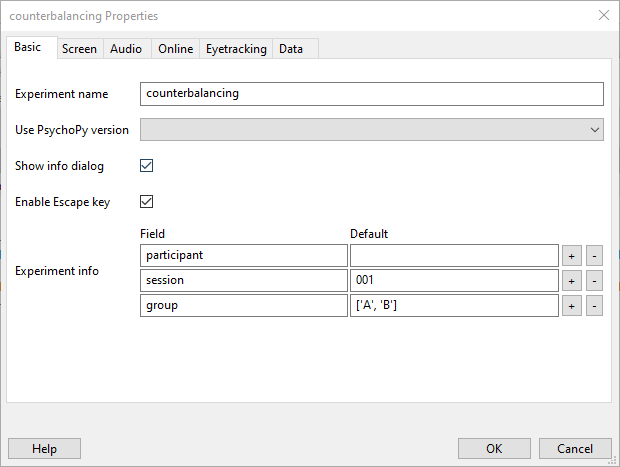
The steps are:

1. You will need to create several conditions files to be able to counterbalance effectively (see the screengrab below for my example
   1. One .xlsx file **per condition** to specify the order of presentation (in the example below these are named A.xlsx and B.xlsx)
      1. **A.xlsx** will present the nums.xlsx condition first, followed by the words.xlsx condition
      2. **B.xlsx** will present the words.xlsx condition first followed by the nums.xlsx condition
      3. Both A and B.xlsx will also store the instructions for the conditions

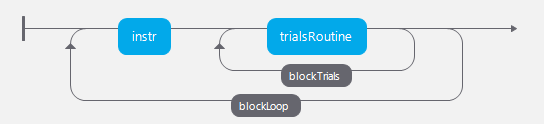


Type the instructions for each condition in the insMsg column

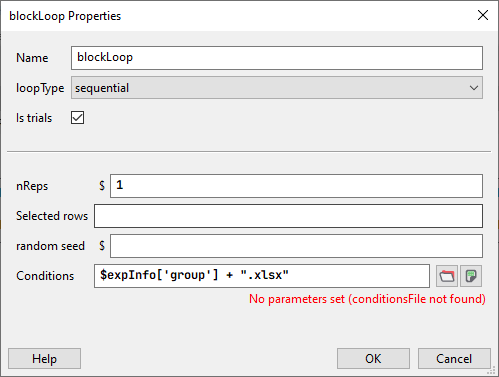
1. You will also need
   1. One .xlsx file **per stimulus set** (named nums.xlsx and words.xlsx)
   2. You may have more files depending on the conditions in your experiment
   3. If you are presenting the same stimulus type (e.g. text component) in each block, you should set up the conditions files using the same column headings as per below
2. In **Experiment Settings** add a new field called **group** and using square brackets enter the filenames for each of the conditions without the .xlsx extension (below I have used A, B as the conditions, but you could use the name of the conditions such as [congruent, incongruent, control] etc). Doing this creates a dropdown for participants to choose a condition



1. Below is the **Flow** for an experiment where there are two blocks **A and B**, however you cannot see them as individual blocks, as these will be controlled by the conditions files. The **trialsRoutine** needs a **Loop** to randomise the trials within each block. Call this **blockTrials.**
2. In the loop properties for the **blockTrials** you need to type **$condsFile** into the **Conditions** box



1. You also need an **instruction routine**, if you have differing instructions for each block you may need to modify the instruction text and again, you can do this with the conditions files. Place a loop around the instructions, and the loop that contains your trials. Call this **blockLoop**
2. In the **blockLoop properties** ensure that
   1. the **loopType** is **sequential**
   2. and type  **$expInfo['group'] + ".xlsx"** into the **Conditions** box



1. When you run the experiment, it is your responsibility to inform them which of the counterbalanced conditions they should choose from the **‘group’** when the demographic data pop up appears at the beginning of the experiment