muCap: Connecting FaceReader™ to z-Tree – Manual & Quick Start Guide

Leonard Doyle & David Schindler, University of Munich & Tilburg University
Last update: November 6th 2019

This document is intended to give you a quick and easy introduction into connecting FaceReader™ and z-Tree via muCap. All explanations are exemplified using the IT infrastructure at the economics laboratory of the University of Munich (MELESSA), but most steps should be similar at other laboratories.

In all examples, we use z-Tree for conducting the experiment, FaceReader™ for the analysis of facial data and three small tools from the muCap package to define events, to record videos of subjects' faces and to match the recordings with the experimental data: muConfig, muCap and muProjectCreator.

muConfig defines event markers by assigning individual color codes. muCap silently records both a webcam video of the subject's face and also previously defined markers which indicate important events in the experiment. muProjectCreator can later be used to integrate these markers and videos into a FaceReader™ project file for analysis. After analyzing, the data (including markers) can be exported and then evaluated in statistical software packages (e.g. STATA or R).

Note that the possible use of muCap is not limited to previously mentioned software packages. Most importantly keep in mind that **by using muCap you agree to cite the companion paper** (*Leonard Doyle & David Schindler (2019): µCap: connecting FaceReader™ to z-Tree. Journal of the Economic Science Association 5(1), pp. 136-141.*) in any scholarly publication or presentation. Details can be found in the License.txt accompanying this release package.

Content

How muCap works 2
Quick start guide 3
Detailed instructions: 5
Prerequisites: Determine points of interest 5
Example experiment: simple dice game 5
Let's get practical: Setting up a z-Tree experiment 6
Setting up the treatment 7
Creating a marker definition list (markers.csv) 10
Testing & conducting the experiment 11
Testing on your own computer 11
Testing in the lab and conducting the experiment 11
Guides and tips for the experiment 12
Using FaceReader™ 13
Creating the FaceReader™ project file 13
Using muProjectCreator to create project with markers 13
Analyzing and exporting the data in FaceReader™ 14
Evaluating the data 15
Appendix 16
Configuring muCap (mucap.conf) 16
Setting up the lab structure 16
muCap.exe starting options 17
Minimum system requirements 17
Known issues / tested configurations 17

How muCap works

muCap will create videos of participants' faces and at the same time create time stamps whenever a certain pixel on screen changes color. By creating small colored boxes in z-Tree and placing them on each participant's screen (at a specific position), the color change triggers markers to be recorded. This effectively allows muCap to link the exact timing of experimental events to the exact video frame of the video footage. For muCap to understand what colors to look for and what events to trigger, muConfig helps to create a proper configuration file. Using muProjectCreator, the recordings and time stamps can be imported into FaceReader™ easily.

The following quick start guide will help you to set up an experiment in a few steps. The remainder of the manual deals with more detailed explanations and known issues.

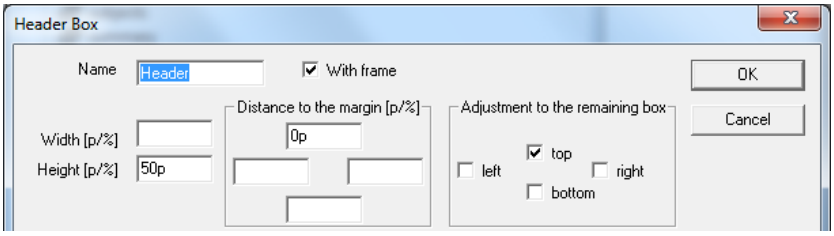
Quick start guide

Some steps in this quick start guide require knowledge about things only introduced later in this document. In case you don't understand a step, please go to the more detailed description in the respective section of this document. We provide a sample z-Tree file and sample muCap configuration files as part of the release package.

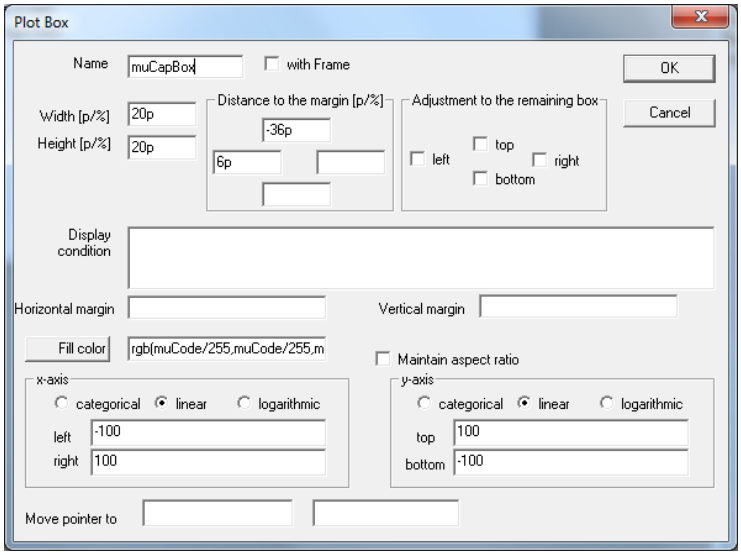
1. In z-Tree: Add a `subjects.do` program to the background:

```
muCode=240;
```

2. Reduce the height of the header box (in the background) to 50 pixels:



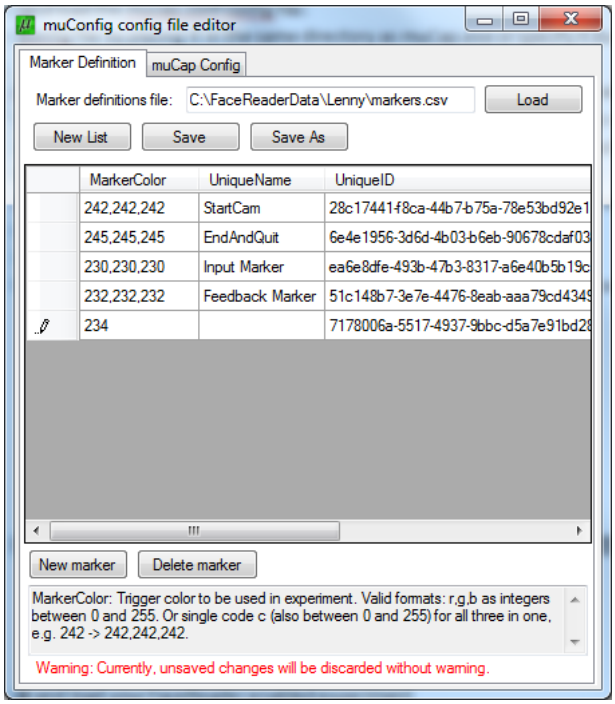
3. Add a plot box right after the header box. Set the background color to `rgb(muCode/255,muCode/255,muCode/255)` and set margins and size as in the image:



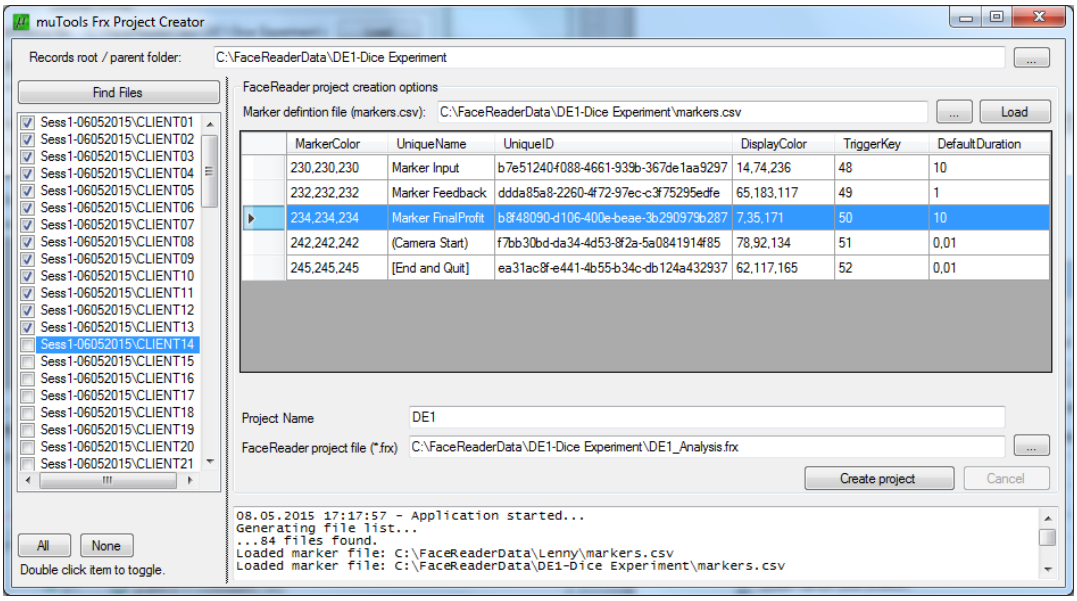
4. Whenever you want to have the box change its color execute a `subjects.do` program:

```
muCode=242;
```

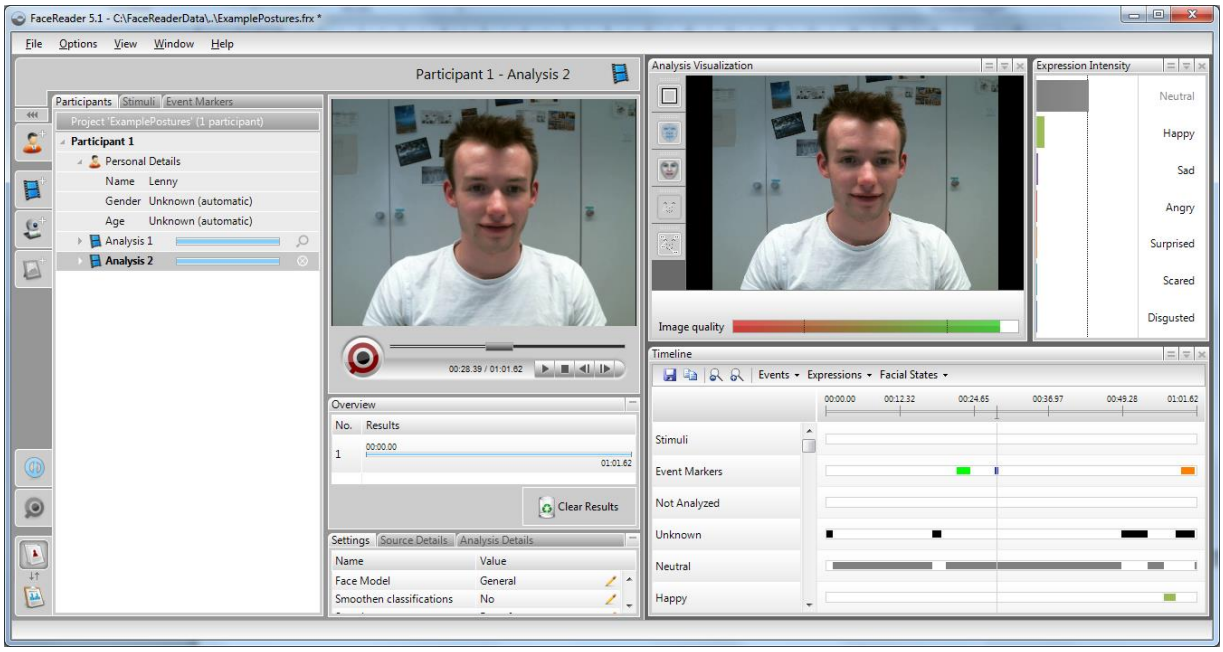
5. Note that `muCode=242` is reserved to start the recording and `muCode=245` to end it!
6. Run muConfig. Click "New List" to generate an empty list of markers. Add new markers by clicking on "New Marker". Enter all marker colors used in your z-Tree code (variable `muCode`) and assign unique names. When all markers have been defined, click "Save".



7. Make sure the configuration file is stored in the same folder as muCap and that this folder can be reached by all client computers. Start z-Tree on the server, z-Leaf on all clients, and muCap on all clients. Start your experiment!
8. Collect all video files and markers, and store them on the computer you are running FaceReader™ on.
9. Run muProjectCreator (on the computer you just stored the files on). Select the correct search path and click on “Find Files”. Select the path for the marker definition file and click “Load”. Enter a project name and click “Create project”.



10. Open FaceReader™ and load the project. Start the analysis. Done!



Detailed instructions:

Prerequisites: Determine points of interest

Before we get into detail with the actual coding and setup, we need to set up a list of important events we will want to analyze. muCap is only able to record **points in time**, for example subjects entering a certain stage, but it is not able to record time spans! For every event we want to analyze, we will have to estimate the duration of interest. For example: To obtain a baseline emotional level for each participant, we might seek to observe them for a few minutes before the actual emotion-inducing event (which might last only a few seconds) occurs.

To go from time points to time spans we have introduced the *default duration* of a marker. On import, each marker will be compiled to have exactly that length (or less if the next marker follows earlier).

Example experiment: simple dice game

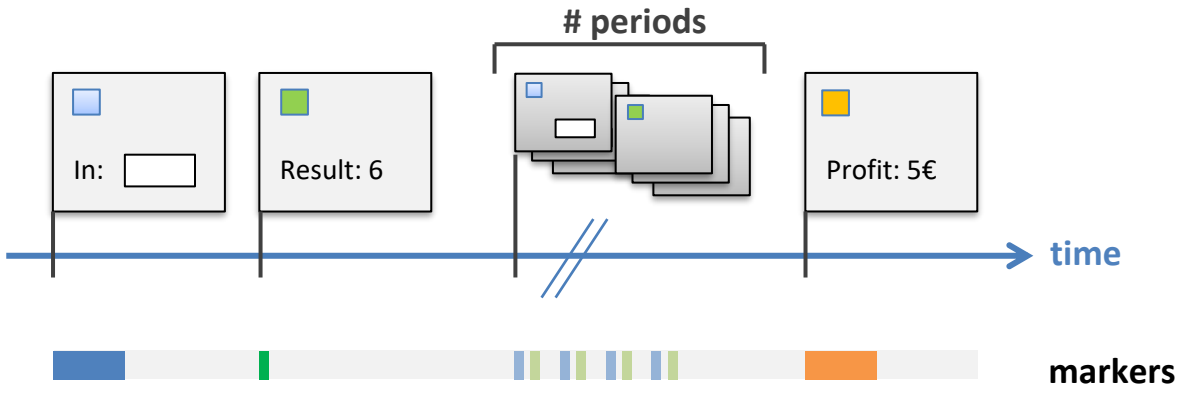
We will use an example experiment[1] to go through the steps necessary to use muCap. Imagine a simple experiment where the subject receives some profit if he guesses the result of a die throw correctly. For each period, there will be an input stage for the subject to enter his or her guess, and a feedback stage showing the throw and resulting profit. Finally, a short questionnaire and a total profit screen will appear. For our emotion analysis, we are interested in the following 3 events:

- Input stage: in each period, determine the subject's current emotional state during his or her decision. As this is not a sudden emotion, the subject might take a minimum of 10 seconds to input his decision. Therefore set the default duration to 10 seconds.
- Feedback stage: in each period, the subject will briefly react to the outcome. This emotion will not last more than 1 second.

[1] All mentioned example files are available as part of the release package.

- Final profit stage at the end of the experiment: The subject will already roughly know the outcome so we do not expect any sudden emotions, so we will plan to average this emotion over 10 seconds again.

To summarize, we decided on 3 different interesting event types throughout the experiment and know exactly when we want to trigger the start of these events and how long each will last. (If you wanted to analyze the input for each stage separately, or distinguish between positive and negative feedback, you would want to treat these as separate events! Otherwise you might have trouble to separate them later for data evaluation.)

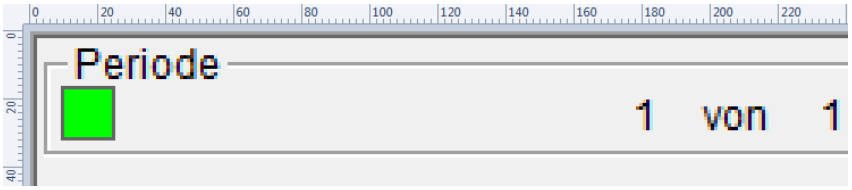


Sketch of how each color code is translated into a marker with duration

Next, we will set up z-Tree to trigger the camera recording (start/ stop) and event markers at the right time.

Let's get practical: Setting up a z-Tree experiment

z-Tree will communicate with muCap through **sending color codes** on the screen! To achieve this, we will add a small box to the z-Tree experiment which can change color. muCap will monitor the exact time of this change in color and interpret this change, e.g. as a marker or to start/ stop the recording.



View of the plot box that is used to send color codes to muCap for camera and event control

For our example experiment to interact with muCap, we will do the following:

- In the very first period, before anything interesting happens, send a color code telling muCap to start the recording.
- In every period, the Input stage will be indicated by one color code.
- In every period, the Feedback stage will be coded with a slightly different code.
- In the last period, the Final profit stage will be coded with a third color.
- At the end of the last period, send a final color code to tell muCap to finish recording and save.

Thus we will use the following color codes in our experiment:

Event	Color Code	Meaning
Start capture	242 (fixed definition)	Tell muCap to start recording camera and screen events **(!)** *Warning! It will take a couple of seconds until the webcam is up and running, do this well before the first event occurs!*
Input stage	230	Subject has to make an input
Feedback stage	232	Subject receives feedback of his profit in this round
Profit stage	234	Final payoff for the whole experiment is displayed
(Pause capture	*243 (fixed definition)*	*Pause to spare unnecessary video data, restart with start code)*
(End capture	*244 (fixed definition)*	*Tell muCap to end recording and save data)*
End capture and quit	245 (fixed definition)	As above, additionally exit muCap on client.

(i) *We are using RGB color definitions. Small variations in color will be detected by muCap but are barely visible to subjects.*

(i) *The pause and end code are listed here for completeness. There is also the option of recording multiple files per session, refer to the appendix for details.*

Now that we have decided on a set of marker types, we can begin implementing them. In z-Tree, we will define a plot box in the background header whose color we will keep updating through changing a variable. This plot box will serve as our interface between z-Tree and muCap.

Setting up the treatment

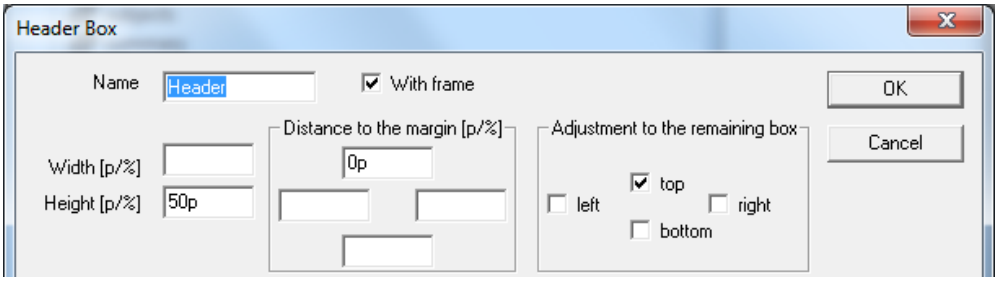
In the `Background` of each treatment file used, do the following:

- To initialize the variable used for color coding, add a program `subjects.do` with your variable definition:

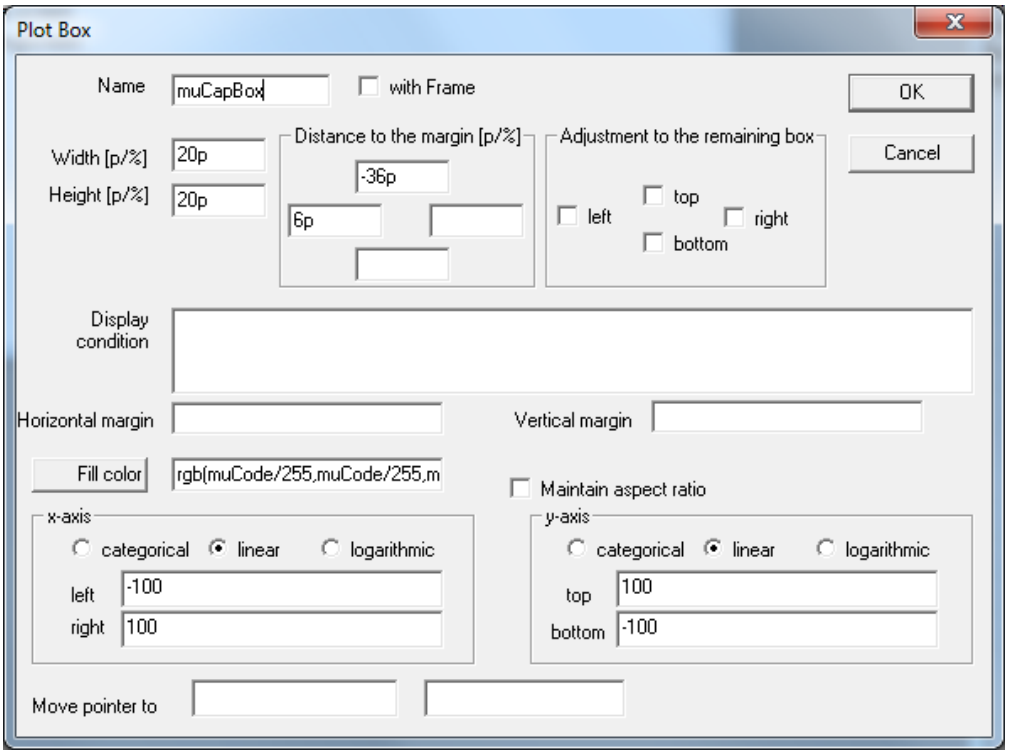
in: `subjects.do`

```
//initialize color code
muCode=240; //240 is dummy/ idle color
```

- Edit the `Header` box to use absolute rather than relative sizing. This will avoid a lot of problems with different screen sizes. Set the height of the header to 50px, leaving the width blank to adjust to the screen size as shown:



- After the `Header` box, add a new plot box, name it, set the background color to `rgb(muCode/255,muCode/255,muCode/255)` and set margins and size as in the image:



(i) *Note the negative top margin. Values: Width=Height=20p, Margin=6p left, -36p top, no frame*

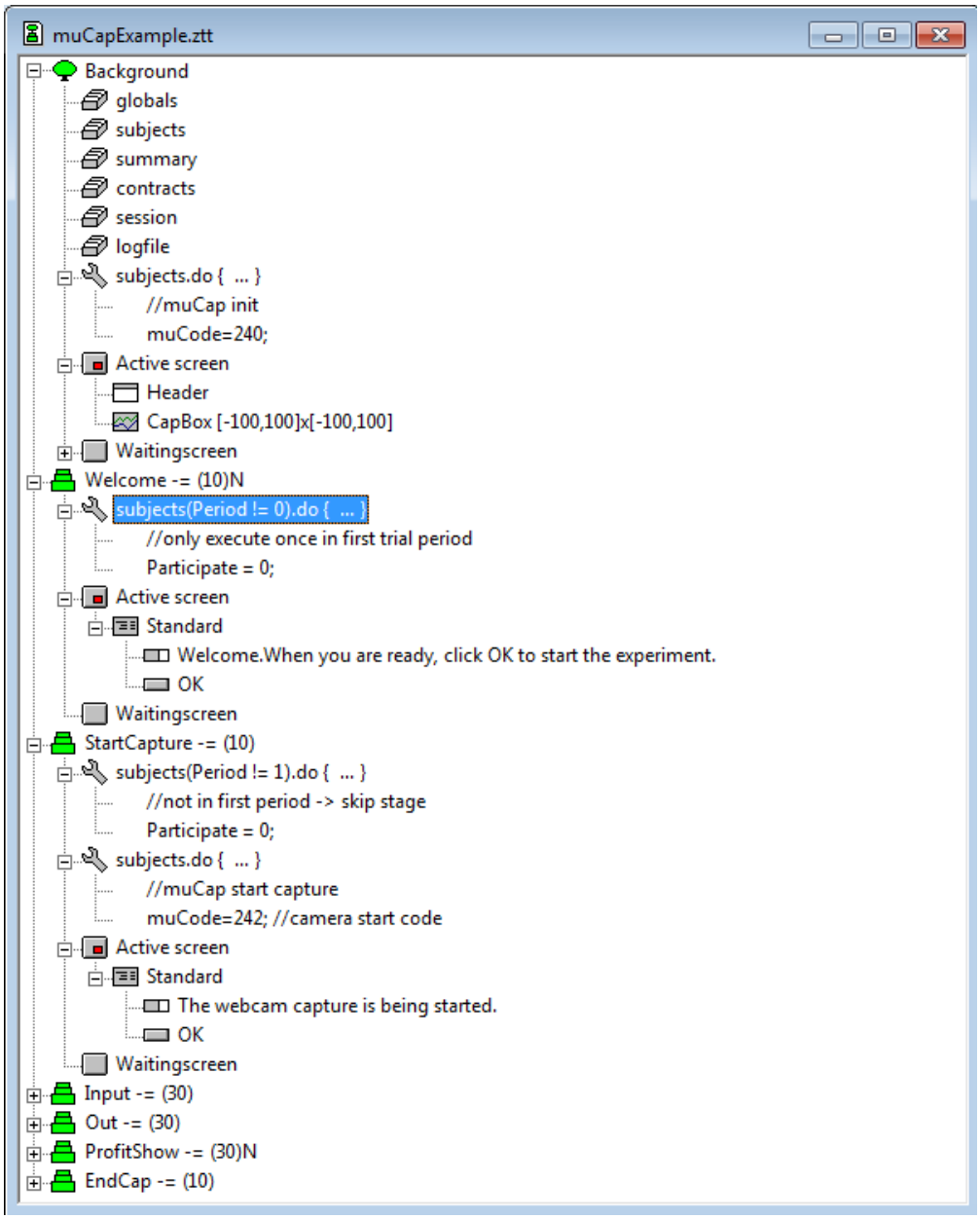
- In each stage when a marker change is supposed to be registered, add a `subjects.do` program with the following code:

in: `subjects.do`
`//define color code for this stage` `muCode=242; //code for starting capture`

(i) *Make sure to always update the color codes appropriately (i.e. to the corresponding stage).*

By putting the plot box in the background, it will show in every stage and we can change its color simply by setting `muCode` to values between 0 and 255 (this range is again credited to the use of RGB colors). The plot box will update immediately when the value changes. Set the variable inside a `subjects.do` program, so that the values are individual to each subject!

Great! We are done enabling our z-Tree project to command muCap. Your Treatment file should now look something like this:



Notes:

- muCap will be configured to monitor a pixel exactly inside of our plot box and therefore register any color change. However it does not know where the box is, it relies on us configuring it. So it is essential that you set up the plot box as described. In any case, test the experiment thoroughly!
- Unfortunately, at the moment there is no possibility to color-code intermediate waiting screens or the questionnaire. All of these will be interpreted as grey (the standard z-Tree background color, 240).
- Undefined color changes will be recorded and saved, but will then be dropped during import with muProjectCreator. If you want to keep them, just add them to the definition list as an own marker.
- The value `muCode` is stored in the subjects table and therefore reset every period. To keep the state from the previous period, use the following code (adapted from z-Tree manual):

in: `subjects.do` with Condition: `Period>1`

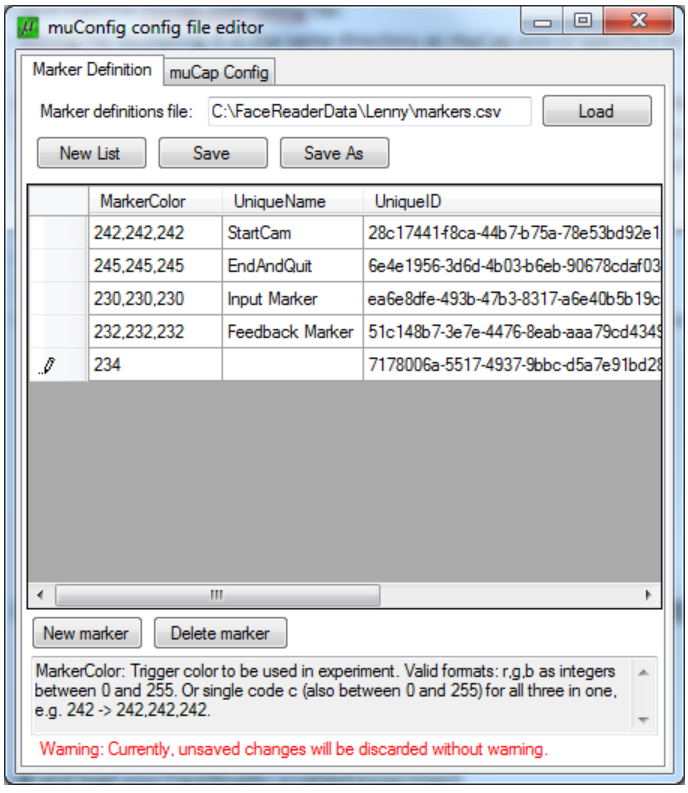
`//if not in first period, try to restore muCode Value from previous period.`
`// as this is done before boxes are executed, no color change will be registered by muCap`
`muCode = OLDsubjects.find(same(Subject),muCode);`

- Take care not to use colors which might be displayed in z-Leaf anyway, e.g. white (255), the background color (240) or black (0). Otherwise, muCap might register unwanted markers or not register wanted ones.
- If the box does not show and no events are registered by muCap, even though all values are set as described, you might have the "first boxes on top" option activated in the general parameters (access by double clicking on the background). This manual assumes this setting is off. If you rely on it, reconfigure the plot box accordingly.

Creating a marker definition list (markers.csv)

Before we can continue, we have to set up the marker list defining the color codes, names and default durations of each marker.

This is easily done using muConfig which is part of the muCap package. Once started, simply load an existing list or start from scratch. There is a small help box explaining the most important details for each item.



muConfig helps you set up the marker list we will need in the last steps

You can load any existing marker definitions file by clicking on "Load". "New List" creates an empty marker definition file where you can add markers "New marker" or remove markers "Delete marker". Make sure to

have the color and unique name defined for each marker. An example marker definitions file is part of the release package.

Testing & conducting the experiment

Our z-Tree project and configuration files are now complete. When testing on your own computer, if no configuration file is found in muCap's directory, it will prompt you for one or enter developer mode, in which you can test the event triggering features (and video if camera is present).

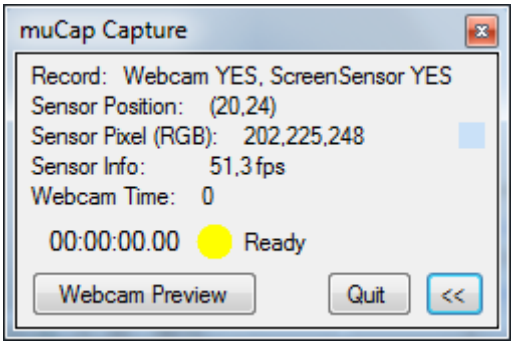
In general, you should first start z-Tree, then z-Leaf and finally muCap. (FaceReader™ and muProjectCreator will only be used after the experiment for analysis.)

Testing on your own computer

Steps:

- Start z-Tree and load your experiment
- Start z-Leaf

  (i) *Although multiple z-Leafs can be run on the test computer, currently only one instance of muCap can be used and configured on each computer. The z-Leaf window also has to be top left so the programmed pixel positions match the full screen values at which muCap will monitor the color box.*
- Start muCap.exe without a "mucap.conf" file in the same directory. muCap will then ask you for a configuration file or whether to load in developer mode. Select "No" for developer mode.
  - If you do need customized settings, refer to the appendix for configuration options.
- If loaded correctly in developer mode, a small window indicating muCap's current state and a Debug window with the recorded marker list should appear.



- If you want to record from a webcam, check the video settings using the "Webcam Preview" button in muCap (you should close the Preview window again before recording begins).
- Run the experiment.
- Observe the current color and registered changes in the muCap Debug window.
- At the end of the test, close muCap using the "Quit" color code or the button.
- You can now have a look at the files created by muCap. Depending on your configuration, you will usually find those in the same folder.

Testing in the lab and conducting the experiment

Firstly, before testing and especially conducting the experiment, the camera settings and positioning have to be checked carefully.

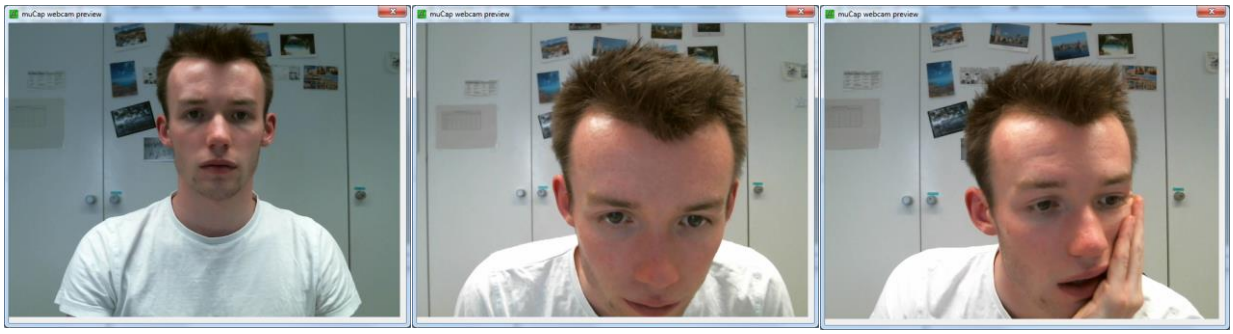
- Install the cameras on each client.

- Start z-Tree as usual and load your experiment.
- Run z-Leaf on each client.
- Run muCap on each client. muCap is configured to hide, it will only show errors and warnings during start-up so as not to disturb subject.
  If the previous experimenter forgot to delete the recorded video files, you will be asked whether you want to overwrite the existing files. Choose yes if you are sure the previous files can be discarded.
- Run the experiment by starting the z-Tree treatment on the experimenter computer.
  - As soon as the stage with the color box indicating a webcam start is reached on each client, the record light on each webcam will light up.
  - As soon as the end capture stage is reached, i.e. the stage with the color code for ending the capture, the webcam recording will stop.
- Finish the z-Tree experiment as usual and save the z-Tree files.
- If you used the capture end & quit color code, muCap will exit automatically. Otherwise, quit muCap manually.
  - The recorded video and markers have been saved to the local hard disk of each client.
- We now have to collect the recorded video and marker files and copy them to the computer running FaceReader™.
- You can now switch off all clients and the experimenter computer. For the analysis, we will only need the FaceReader™ computer.

Guides and tips for the experiment

Here are a few notes on some general good practices:

- Experience shows that a lot of participants will sit close to the screen. It is very hard to set up the webcam viewing angle nicely to accommodate different postures, play around with the adjustment during testing.
- Many participants will lean on their elbows when closer to the screen. Hands and objects obscuring even small parts of the face will make the analysis less reliable or impossible, so we advise to inform the participants not to lean towards the screen if possible. Nonetheless, expect a few failed scenes in every session.
- A possible solution is to use bigger fonts where possible.
- Another clever trick might be to have an input immediately before the interesting moment. This will ensure the subject's hands are not covering the face and the person is facing the screen. Ideally even require a two hand input (e.g. key press and mouse action) to guarantee suitable posture.



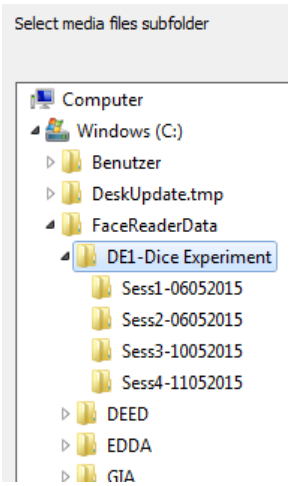
1: good, subject will not grow any taller, but might lean back etc.
2: especially people with bad eyesight move very close to the screen, which cuts off parts of the face
3: often, resting the face on the hand will be interpreted as a "happy" expression due to rise in mouth corners

Using FaceReader™

Congratulations! We successfully conducted the experiment and have saved the video and marker files to the FaceReader™ computer. Now it is time to feed them to FaceReader™ and start the analysis.

Creating the FaceReader™ project file

For our example project, assume we created the following file structure:

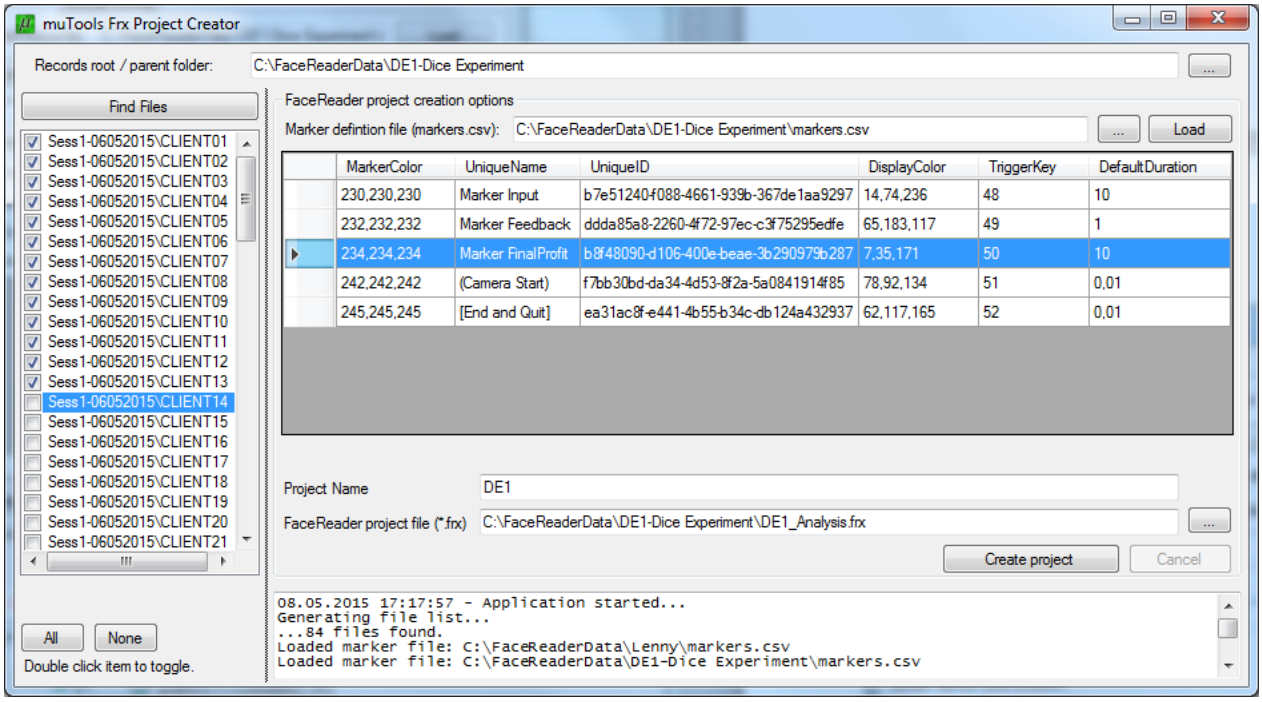


In each session subfolder, there are 24 (or the number of clients used) videos and marker files with matching names. In Munich, these would be called CLIENTxy.wmv and CLIENTxy.csv, respectively.

Using muProjectCreator to create project with markers

- Open muProjectCreator.
- Choose the search path where your videos are stored.
    - muProjectCreator will search on 2 levels of subfolders, so for example choose your experiment folder to search all sessions' subfolders.
- Click "Find Files" to search specified folder for video files.
    - Videos will also be added even if no matching markers are found. These will be imported with an empty marker list.
- Next, select the marker definitions file, e.g. "X:\Videos\markers.csv" and press "Load".
- Finally, choose a project title and file name for the FaceReader™ project.

- By clicking “Create project”, muProjectCreator will compile the file list into a FaceReader™ project, adding each video file as a new participant and patching in the defined markers according to the definitions loaded.



Analyzing and exporting the data in FaceReader™

This file can now be loaded normally in FaceReader™ for analysis.

- Open FaceReader™.
- Load the project.
- Verify the markers have been imported correctly and all the videos etc. are set up correctly.
- Start the batch analysis. This will be very time consuming, but fortunately will not require you to be there (If time permits, you can occasionally check on the current progress).
- Once the batch analysis has finished, save the project again.
- You can now choose “File→Export→Export all” to export the detailed logs of each participant’s analysis to separate files for use in statistical software packages.

Notes:

- Markers can still be edited after import inside FaceReader™. However this is not suitable for large changes as it has to be done individually for each video.
- In general, the detailed log will be the most interesting, the state log contains very few information.
- Do not attempt to export an “observer log”. It does not contain any more data and has proven to crash FaceReader™ on large projects on our machines.

Evaluating the data

The exported logs are essentially all you will need now from the experiment. They are tabulator separated tables containing columns for some basic video data, of course the emotions, the marker name for each point in time and some further information like head orientation etc.

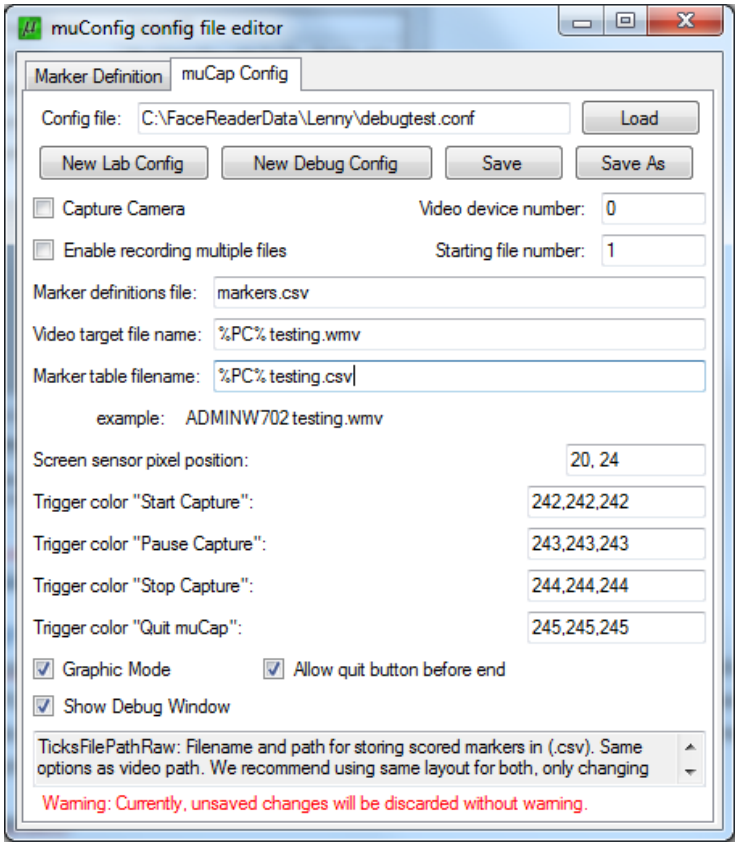
3	Face Model:	General									
4	Calibration:	None									
5	Start time:	09.08.2014 19:30									
6	Filename:	C:\CapVE\IDS\Videos\17.wmv									
7	Frame rate:	21,27659574									
8											
9	**Video Time**	**Neutral**	**Happy**	**Sad**	**Angry**	**Surprised**	**Scared**	**Disgusted**	**Valence**	**Gender**	**Age**
10	00:00:00.000	0.415685100	0.147391100	0.030830500	0.244982200	0.007255198	0.000004068	0.006697997	-0.097591100	Male	15 - 2
11	00:00:00.047	0.478993300	0.129147500	0.028212450	0.221082300	0.006676359	0.000005102	0.006060210	-0.091934740	Male	15 - 2
12	00:00:00.094	0.516971500	0.114278500	0.025889150	0.200508500	0.006243767	0.000007908	0.005496475	-0.086230060	Male	15 - 2
13	00:00:00.141	0.555554800	0.101860900	0.023653410	0.182730300	0.005833238	0.000007735	0.005241376	-0.080869310	Male	15 - 2
14	00:00:00.188	0.569408200	0.090506930	0.021919380	0.171533300	0.005841357	0.000009934	0.004768636	-0.081026380	Male	15 - 2
15	00:00:00.235	0.606298300	0.080164900	0.020257150	0.159431500	0.005733338	0.000009473	0.004410729	-0.079266570	Male	15 - 2
16	00:00:00.282	0.641050200	0.072655100	0.018566330	0.144559500	0.005794147	0.000009396	0.003960639	-0.071904350	Male	15 - 2
17	00:00:00.329	0.674788700	0.067316790	0.016508230	0.132911500	0.006298988	0.000008700	0.003548927	-0.065594730	Male	15 - 2
18	00:00:00.376	0.708363300	0.063354730	0.014637500	0.121681800	0.007182661	0.000008307	0.003149070	-0.058327030	Male	15 - 2
19	00:00:00.423	0.731022900	0.060954160	0.013307160	0.110703300	0.007149715	0.000007781	0.002867361	-0.049749180	Male	15 - 2
20	00:00:00.470	0.751341300	0.057700490	0.012519940	0.099504860	0.006881439	0.000007704	0.002652626	-0.041804370	Male	15 - 2
21	00:00:00.517	0.775274900	0.055087020	0.011447550	0.090970520	0.006981822	0.000007442	0.002395282	-0.035883500	Male	15 - 2
22	00:00:00.564	0.792350000	0.052858040	0.010755900	0.082579740	0.006723725	0.000007263	0.002249663	-0.029721700	Male	15 - 2
23	00:00:00.611	0.806322800	0.049383680	0.010029540	0.078688590	0.006880862	0.000007253	0.002082295	-0.029304910	Male	15 - 2
24	00:00:00.658	0.812697400	0.046565370	0.009960865	0.073168540	0.006634034	0.000007286	0.002000821	-0.026603170	Male	15 - 2
25	00:00:00.705	0.827488200	0.043284710	0.009398444	0.070388650	0.006904251	0.000007037	0.001861458	-0.027103950	Male	15 - 2
26	00:00:00.752	0.822859200	0.040734180	0.010528600	0.064867640	0.006513556	0.000007633	0.001886078	-0.024133460	Male	15 - 2
27	00:00:00.799	0.832280900	0.038546800	0.010298620	0.060815820	0.006427440	0.000007932	0.001765426	-0.022269030	Male	15 - 2

Appendix

In this section we provide some additional information on how to set up muCap in the lab and how to reconfigure it for custom needs.

Configuring muCap (mucap.conf)

In addition to creating marker definition files, muConfig can also be used to customize the behavior of muCap for different IT infrastructures. It features a quick help box explaining the most important details on each setting.



This manual so far explained only one recording per subject/ session and no pause functions. However, it is also possible to pause the video and restart quickly if you want to skip unnecessary parts. Since there is however a known bug with pausing and Logitech webcams, there is also the possibility to record multiple videos during one session. For this, multi record has to be enabled and the filenames have to contain the number wildcard.

Note:

- Both configuration files are plain text, so editing without muConfig is possible in theory. However this is at your own risk and probably not necessary for most cases.

Setting up the lab structure

At the University of Munich, we have a shared network drive that all clients (and experimenter computer) can access. We recommend a similar setup with the following file structure:

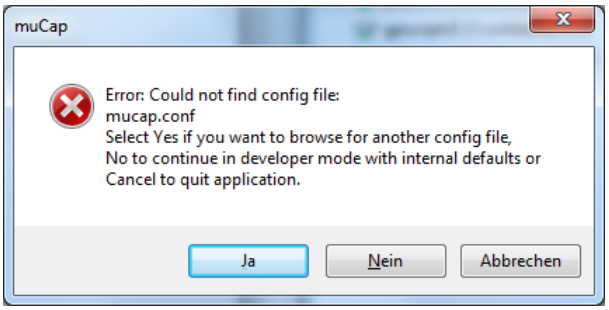
- One Server that can be accessed by all clients.
    - Place muCap.exe somewhere.

    - Place the mucap.conf in the same or a different folder.
        - If you use a different folder, you will have to use command line options to start with the correct configuration file
- On each client, create a folder for the recorded files, e.g. “C:\Videos\”
    - In the configuration file, configure both video path and marker list path (not marker definitions) accordingly.
    - Make sure the client user has write access to this folder.
    - Saving the videos locally reduces network traffic and increases reliability.

muCap.exe starting options

In Munich we start muCap on the clients using a remote control program (called LAS Manager) which can call muCap specifying a configuration file. The command line options to achieve this are:

- If none specified, muCap will look for “mucap.conf” in the same directory as its starting directory
- To specify a configuration file, use
    - “muCap.exe <xyz>.conf” or
    - “muCap.exe relative\path\xyz.conf” or
    - “muCap.exe C:\path\xyz.conf”
    - (as usual in windows paths with blanks have to be contained in quotes ””)
- If no configuration file is not found muCap will ask for options on startup:



Minimum system requirements

For muCap on each client:

- Windows XP, Vista, 7, 8
- 2GHz CPU
- 1GB Ram
- Depending on video length up to 1GB of free disk space
- .net Framework 2.0

For the FaceReader™ analysis PC:

- FaceReader™ 5 minimum requirements
- Additionally: .net Framework 4.5

Known issues / tested configurations

- The video pause option is not working for any Logitech webcam on 64-bit operating system. This is a known issue with Logitech drivers. The issue does not arise on 32-bit operating systems or with drivers other than Logitech. A future version of muCap might work around this issue. For now,

recording multiple video files works as a reasonable alternative (needs to be enabled in the configuration file).

- We tested muCap in the following environment: Windows 7 64-bit, Core2Duo Platform, .net Framework 3.5.
- Additionally we found the following problems in this environment: MS Surface Pro, Win 8, .net Framework 4: The timing seems to be imprecise and generates latencies of up to half a second in rare cares. Future versions will hopefully address this issue.
- Untested: Windows XP, but all technologies used should be compatible.