Instructions for listening tests using Web Audio Evaluation Tool

Brecht De Man

These instructions are about use of the Web Audio Evaluation Tool [1] with the APE interface [2] on Windows and Mac OS X platforms.

Contents

1 Installation and set up

Download the folder and unzip in a location of your choice.

1.1 Contents

The folder should contain the following elements:

Main folder:

- `ape.css, core.css, graphics.css`, structure.css: style files (edit to change appearance)
- `ape.js`: JavaScript file for APE-style interface [2]
- `core.js`: JavaScript file with core functionality
- `index.html`: webpage where interface should appear
- `jquery-2.1.4.js`: jQuery JavaScript Library
- `pythonServer.py`: webserver for running tests locally
- `pythonServer-legacy.py`: webserver with limited functionality (no automatic storing of output XML files)

Output files (/saves/)

- The output XML files of tests will be stored here by default by the `pythonServer.py` script.

Auxiliary scripts (/scripts/)

- Helpful Python scripts for extraction and visualisation of data.

Test data (/test-data/)

- The audio and setup files necessary for running the test. Specifically, `_subjects` contains the setup files which are selected sequentially, and the other folders contain the audio for each 'song'.

1.2 Browser

As Microsoft Internet Explorer doesn't support the Web Audio API [1], you will need another browser like Google Chrome, Safari or Firefox (all three are tested and confirmed to

[1] `http://caniuse.com/#feat=audio-api`

work).

The tool is platform-independent and works in any browser that supports the Web Audio API. It does not require any specific, proprietary software. However, in case the tool is hosted locally (i.e. you are not hosting it on an actual webserver) you will need Python, which is a free programming language - see the next paragraph.

1.3 Python 2.7

On Windows, Python 2.7 is not generally preinstalled and therefore has to be downloaded[2] and installed to be able to run scripts such as the local webserver, necessary if the tool is hosted locally.

On Mac OS X, Python comes preinstalled.

2 Listening test

2.1 Start local webserver

If the test is hosted locally, you will need to run the local webserver provided with this tool.

2.1.1 Windows

Simply double click the Python script `pythonServer.py` in the folder you downloaded.

You may see a warning like the one in Figure 1. Click 'Allow access'.

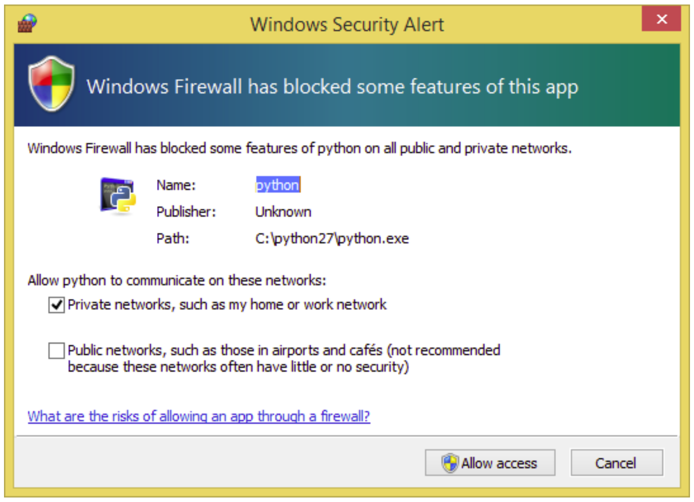
The process should now start, in the Command prompt that opens - see Figure 2.

You can leave this running throughout the different experiments (i.e. leave the Command Prompt open).

2.1.2 Mac OS X

Open the Terminal (find it in **Applications/Terminal** or via Spotlight), and go to the folder you downloaded. To do this, type `cd [folder]`, where `[folder]` is the folder

[2] `https://www.python.org/downloads/windows/`



Windows Firewall has blocked some features of this app

Windows Firewall has blocked some features of python on all public and private networks.

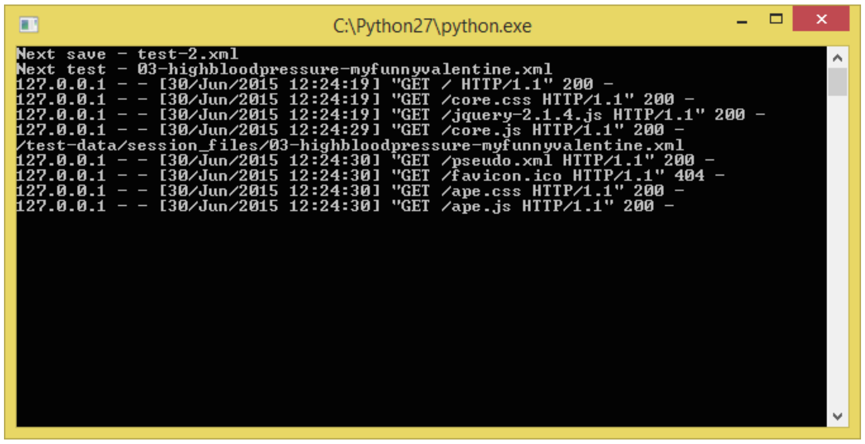
Name: python
Publisher: Unknown
Path: C:\python27\python.exe

Allow python to communicate on these networks:

- [✓] Private networks, such as my home or work network
- [ ] Public networks, such as those in airports and cafés (not recommended because these networks often have little or no security)

What are the risks of allowing an app through a firewall?

Figure 1: Windows: Potential warning message when executing `pythonServer.py`.



```
Next save - test-2.xml
Next test - 03-highbloodpressure-myfunnyvalentine.xml
127.0.0.1 - - [30/Jun/2015 12:24:19] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2015 12:24:19] "GET /core.css HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2015 12:24:19] "GET /jquery-2.1.4.js HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2015 12:24:29] "GET /core.js HTTP/1.1" 200 -
/test-data/session_files/03-highbloodpressure-myfunnyvalentine.xml
127.0.0.1 - - [30/Jun/2015 12:24:30] "GET /pseudo.xml HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2015 12:24:30] "GET /favicon.ico HTTP/1.1" 404 -
127.0.0.1 - - [30/Jun/2015 12:24:30] "GET /ape.css HTTP/1.1" 200 -
127.0.0.1 - - [30/Jun/2015 12:24:30] "GET /ape.js HTTP/1.1" 200 -
```

Figure 2: Windows: The Command Prompt after running `pythonServer.py` and opening the corresponding website.

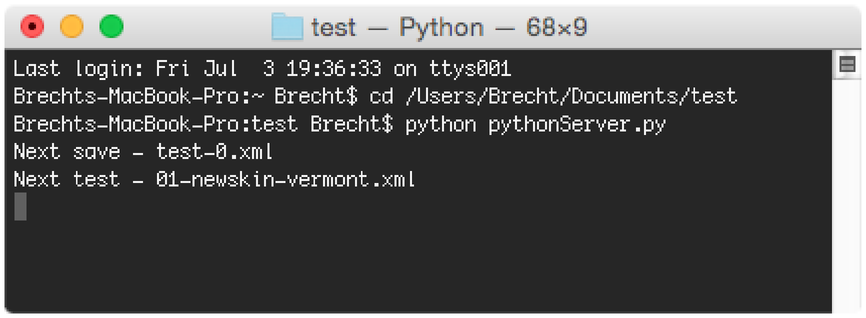
where to find the `pythonServer.py` script you downloaded. For instance, if the location is `/Users/John/Documents/test/`, then type

```
cd /Users/John/Documents/test/
```

Then hit enter and run the Python script by typing

```
python pythonServer.py
```

and hit enter again. See also Figure 3.



```
Last login: Fri Jul  3 19:36:33 on ttys001
Brechts-MacBook-Pro:~ Brecht$ cd /Users/Brecht/Documents/test
Brechts-MacBook-Pro:test Brecht$ python pythonServer.py
Next save - test-0.xml
Next test - 01-newskin-vermont.xml
```

Figure 3: Mac OS X: Potential warning message when executing `pythonServer.py`.

You can leave this running throughout the different experiments (i.e. leave the Terminal open).

2.2 Sample rate

Depending on how the experiment is set up, audio is resampled automatically (the Web Audio default) or the sample rate is enforced. In the latter case, you will need to make sure that the sample rate of the system is equal to the sample rate of these audio files. For this reason, all audio files in the experiment will have to have the same sample rate.

To change the sample rate in Mac OS X, go to **Applications/Utilities/Audio MIDI Setup** or find this application with Spotlight. Then select the output of the audio interface you are using and change the 'Format' to the appropriate number. Also make sure the bit depth and channel count are as desired. If you are using an external audio interface, you may have to go to the preference pane of that device to change the sample rate.

To change the sample rate in Windows, right-click on the speaker icon in the lower-right corner of your desktop and choose 'Playback devices'. Right-click the appropriate playback device and click 'Properties'. Click the 'Advanced' tab and verify or change the sample rate under 'Default Format'.

Always make sure that all other digital equipment in the playback chain (clock, audio interface, digital-to-analog converter, ...) is set to this same sample rate.

For this experiment, the sample rate is enforced and should be 96 kHz.

2.3 Setting up a participant

2.3.1 Instructions

Before each test, show the instruction video[3] on how to use the test, and print the instructions below and make sure it is available. Make sure to ask whether the participant has any questions upon seeing and/or reading the instructions.

- You will be asked for your name ("John") and location (room identifier).
- An interface will appear, where you are asked to
  - click green markers to play the different mixes;
  - drag the markers on a scale to reflect your preference for the mixes;
  - comment on these mixes, using text boxes with corresponding numbers (in your **native language**);
  - optionally comment on all mixes together, or on the song, in 'General comments'.
- You are asked for your personal, honest opinion. Feel free to use the full range of the scale to convey your opinion of the various mixes. Don?t be afraid to be harsh and direct.
- The markers appear at random positions at first (which means some markers may hide behind others).
- The interface can take a few seconds to start playback, but switching between mixes should be instantaneous.
- This is a research experiment, so please forgive us if things go wrong. Let us know immediately and we will fix it or restart the test.
- When the test is finished (after all songs have been evaluated), just call the experimenter, do NOT close the window.
- After the test, please fill out our survey about your background, experience and feedback on the test.

[3]`https://www.youtube.com/watch?v=aGhmn7agCLM`

- By participating, you consent to us using all collected data for research. Unless asked explicitly, all data will be anonymised when shared.

2.3.2 The test

To start the test, open the browser and type

`localhost:8000`

and hit enter. The test should start (see Figure 4).

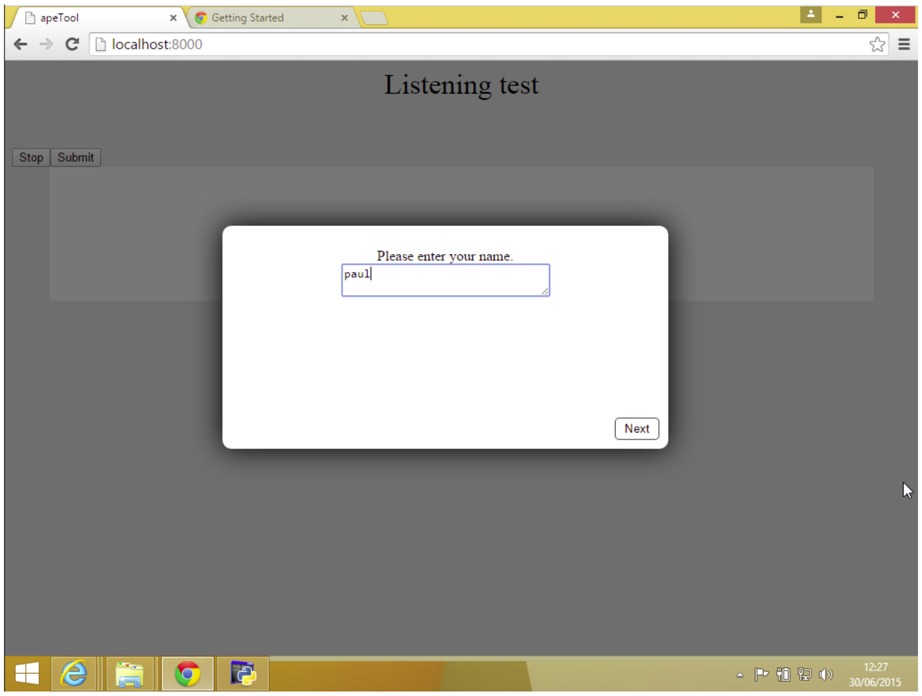


Figure 4: The start of the test in Google Chrome on Windows 7.

If at any point in the test the participant reports weird behaviour or an error of some kind, or the test needs to be interrupted, please notify the experimenter and/or refer to Section 2.4.

When the test is over (the subject should see a message to that effect, and click 'Submit' one last time), the output XML file containing all collected data should have appeared in 'saves/'. The names of these files are 'test-0.xml', 'test-1.xml', etc., in ascending order. The Terminal or Command prompt running the local web server will display the following file name. If such a file did not appear, please again refer to Section 2.4.

It is advised that you back up these results as often as possible, as a loss of this data means that the time and effort spent by the subject(s) has been in vain. Save the results to an external or network drive, and/or send them to the experimenter regularly.

To start the test again for a new participant, you do not need to close the browser or shut down the Terminal or Command Prompt. Simply refresh the page or go to `localhost:8000` again.

2.3.3 Survey

The tool allows for embedded questions before and after each page, and before and after the whole test. If these do <u>not</u> include survey questions (about the participant's background, demographic information, and so on) make sure to ask the participant to complete the survey immediately after the test. Above anything else, this decreases the likelihood that the survey goes forgotten and the experimenters do not receive the data in time.

2.4 Troubleshooting

Thanks to feedback from using the interface in experiments by the authors and others, many bugs have been caught and fatal crashes due to the interface (provided it is set up properly by the user) seem to be a thing of the past. However, if things do go wrong or the test needs to be interrupted for whatever reason, all data is not lost. In a normal scenario, the test needs to be completed until the end (the final 'Submit'), at which point the output XML is stored in the `saves/`. If this stage is not reached, open the JavaScript Console (see below for how to find it) and type

```
createProjectSave()
```

and hit enter. This will open a pop-up window with a hyperlink that reads 'Save File'; click it and an XML file with results until that point should be stored in your download folder. Alternatively, a lot of data can be read from the same console, in which the tool prints a lot of debug information. Specifically:

- the randomisation of pages and fragments are logged;
- any time a slider is played, its ID and the time stamp (in seconds since the start of the test) are displayed;
- any time a slider is dragged and dropped, the location where it is dropped including the time stamp are shown;
- any comments and pre- or post-test questions and their answers are logged as well.

You can select all this and save into a text file, so that none of this data is lost. You may to choose to do this even when a test was successful as an extra precaution.

2.4.1 Opening the JavaScript Console

- In Google Chrome, the JavaScript Console can be found in **View>Developer>JavaScript Console**, or via the keyboard shortcut Cmd + Alt + J (Mac OS X).
- In Safari, the JavaScript Console can be found in **Develop>Show Error Console**, or via the keyboard shortcut Cmd + Alt + C (Mac OS X). Note that for the Developer menu to be visible, you have to go to Preferences (Cmd + ,) and enable 'Show Develop menu in menu bar' in the 'Advanced' tab.
- In Firefox, go to **Tools>Web Developer>Web Console**, or hit Cmd + Alt + K.

References

[1] N. Jillings, D. Moffat, B. De Man, and J. D. Reiss, "Web Audio Evaluation Tool: A browser-based listening test environment," in *12th Sound and Music Computing Conference (accepted)*, June 2015.

[2] B. De Man and J. D. Reiss, "APE: Audio Perceptual Evaluation toolbox for MATLAB," in *136th Convention of the Audio Engineering Society*, April 2014.