PEST – Phishing Email Suspicion Test

In 'main.m':

Lines 20 -25 define the file store locations for PEST data:

a. Line 20 (outFile1) sets the file store directory to a specific location on the computer/cloud. Replace 'PATH_TO_OUT' with the desired location for data files
b. Line 24 (outFile2) sets the file store as the same root directory as the task itself – in 'data' folder

```
% cloud data store
outFile1 = strcat('PATH_TO_OUT/pestdata_', num2str(subjectID), '_',...
    str, '_', num2str(subjectAge), '_', subjectGender, '.dat');

% local data store
outFile2 = strcat(pwd, '/data/pestdata_', num2str(subjectID), '_',...
    str, '_', num2str(subjectAge), '_', subjectGender, '.dat');
```

The data is stored as a .dat output file. The file is organized as follows:

Each comma-separated value, in this order, indicates:

1. The subject response (on a scale of 1 – 4). **1** indicates 'Definitely Safe', **2** indicates 'Possibly Safe', **3** indicates 'Possibly Suspicious', and **4** indicates 'Definitely Suspicious'.
2. Reaction time (seconds)
3. Email category
4. Email type (i.e. weapon of influence)
5. Attachment (0 or 1). **0** indicates that the email had no attachment and a link, **1** indicates that the email had an attachment and no link
6. True email identifier. 'Sim_Scam/Sim_Safe' indicates that the subject responded to a simulated scam/safe email. 'Real_Scam/Real_Safe' indicates that the subject responded to a real scam/safe email
7. Unique email identifier - corresponds to email identifier in column B of the email spreadsheet (see below)
8. Score - subject score after each trial

In 'runTask.m':

Lines 7 - 10 indicate the number of emails to pull from each email set. 'emailSet1' corresponds to the simulated phishing emails, 'emailSet2' to the simulated safe emails, 'emailSet3' to the real phishing emails, and 'emailSet4' to the real safe emails. The program pulls 40 emails from each email set. To change the number of emails from each set:

1. Simulated Phish: Line 7 - change second argument of 'getEmails' to an integer <= 84
2. Simulated Safe: Line 8 - change second argument of 'getEmails' to an integer <= 84
3. Real Phish: Line 9 - change second argument of 'getEmails' to an integer <= 140
4. Real Safe: Line 10 - change second argument of 'getEmails' to an integer <= 40

In 'cycleEmails.m':

Line 14 sets the maximum score needed to complete the task. Subjects receives 2 points for correct email identification of high confidence (i.e. definitely scam or definitely safe), and 1 point for correct email identification of low confidence (i.e. possibly scam or possibly safe). Subjects also lose points for misidentification. They lose points in the same manner as they gain them. 2 points are lost for incorrect classification of high confidence, and 1 point is lost for incorrect classification of low confidence. The subject's score is displayed at the end of the task.

Direct inquiries to Ziad Hakim at **zhakim@ufl.edu**

The emails are stored in 4 excel spreadsheet files (.xlsx) in ‘resources/emails’ folder in the task’s root directory. If you want to change email content, you must modify these four files ‘real-phish.xlsx’, ‘real-safe.xlsx’, 'sim-phish.xlsx', and 'sim-safe.xlsx’. The spreadsheets are organized via the scheme:

- Each row corresponds to an email
- Column A – email subject line
- Column B – email identifier
- Column C – email body with link
- Column D – email category
- Column E – email type (i.e. weapon of influence)
- Column F – true email identifier (e.g. ‘Sim_Scam’, ‘Sim_Safe’, ‘Real_Scam’, ‘Real_Safe’)
- Column G – attachment name
- Column H – attachment preview text
- Column I – alternate email body with attachment
- Column J – sender name

Note: whether an email is displayed with attachments is computed randomly. It is critical, then, to have both versions of the email and the attachment name.

Similarly, the task instructions are stored in ‘resources/instructions.xlsx’ in the task’s root directory. If you wish to change any instruction for the task, you must modify this spreadsheet. The spreadsheet is organized via the scheme:

- Each row corresponds to an instruction screen
- Column A – instruction text/function call
- Column B – path of any images used in the instruction sequence. Images are stored in ‘resources/imgs’ in the tasks’ root directory

To change an instruction’s image: store the new image in ‘resources/imgs’ directory and update Column B of the ‘instructions.xlsx’ spreadsheet accordingly.

The screen may or may not be displaying the task in proper proportions. Fixing this is critical to proper task functioning. The task draws elements on the screen via:

1. The script ‘drawEmail.m’ draws the email window, email contents, attachment window, and buttons on the screen
2. The function call ‘getPixels’ sets the position of these items based on either a horizontal or vertical offset. A horizontal offset is indicated by ‘w’ in the first argument of the function call, and a vertical offset is indicated by ‘h’ in the first argument of the function call. Note that this function is not fully accurate.
3. Changing the second argument will reposition the object by its new offset value.
4. Fixing component proportions involves some trial and error, unless you know the dimensions of your display and can manipulate it efficiently.

Direct inquiries to Ziad Hakim at **zhakim@ufl.edu**