1.1. Writing to the Tab Delimited .gazedata File in EET 3.1

Summary:

E-Prime provides extensive capabilities for collecting, reporting, and viewing data. In most designs, E-Prime is used to collect and output one data row per trial. In eye tracking experiments, the goal is to collect and output data for every eye gaze sample (of which there may be several hundred to thousands per trial depending on the trial duration). Because of this, EET paradigms are enabled to employ an additional data collection and output technique which results in the creation of a Tab Delimited text file. This file contains data rows which are a combination of eye gaze data received by E-Prime from the Tobii Pro Eye Tracker and any additional data added by the user via E-Prime.

The type and amount of data you choose to collect is determined by your experimental design and analysis choices. To allow maximum flexibility, the methods are performed in script sections of the experiment (as opposed to PackageCalls) to expose both the content and the format of output files to manipulation by the user.

NOTE: *The following tasks can be performed in any EET 3.1 experiment where a .gazedata file is needed since the User Script and SaveGazeData InLine is written generically. However, you can also use and modify the experiment file attached to SaveGazeData in E-Prime Extensions for Tobii Pro 3.1.*

Overview of Tasks:

- Open the experiment you wish to add gaze data to.
- Declare a Global User Defined Data Type and name it UserEyeGazeDataType.
- Add a new member to the end of the UserEyeGazeData Type data structure.
- Edit the script to append a tab character and new column label.
- Append an ebTab line for the UserEyeGazeDataType.
- Create the SaveGazeData InLine.
- Modify the SaveGazeData InLine to use the HitTest method to track AOIs.
- Verify the overall experiment structure and run the experiment.
- Open the .gazedata file.
- Familiarize yourself with the basic structure of the .gazedata file.
- Open the .gazedata file for TETFixedPositionAOI to verify the data written to the file.
- Look at the specific sections of script from the SaveGazeData InLine and the TETFixedPositionAOI UserScript side by side with the columns of the .gazedata file the script created.
- Learn how the .gazedata file is created.

Recommended readings:

Before beginning the E-Prime Extensions for Tobii Pro Tutorial, we recommend that you have read the information we offer on scripting:

Article 23286: Extending Experiments with Script
Article 22869: Getting Started with Writing Script
Article 22879: User Script Window
Article 22710: InLine Object

Estimated Tutorial Time:

15-20 minutes

NOTE: *The typical E-Prime data file (.edat3) generated by your experiment is always available to you (unless you take steps to specify otherwise). For most eye tracking studies, the output file described in this section is the data most used in your data analysis (as opposed to the .edat3 file). It is up to the experimenter to specify the column ordering of this output file and to verify that the correct data is available for the analysis which the experimenter expects to perform on the data.*

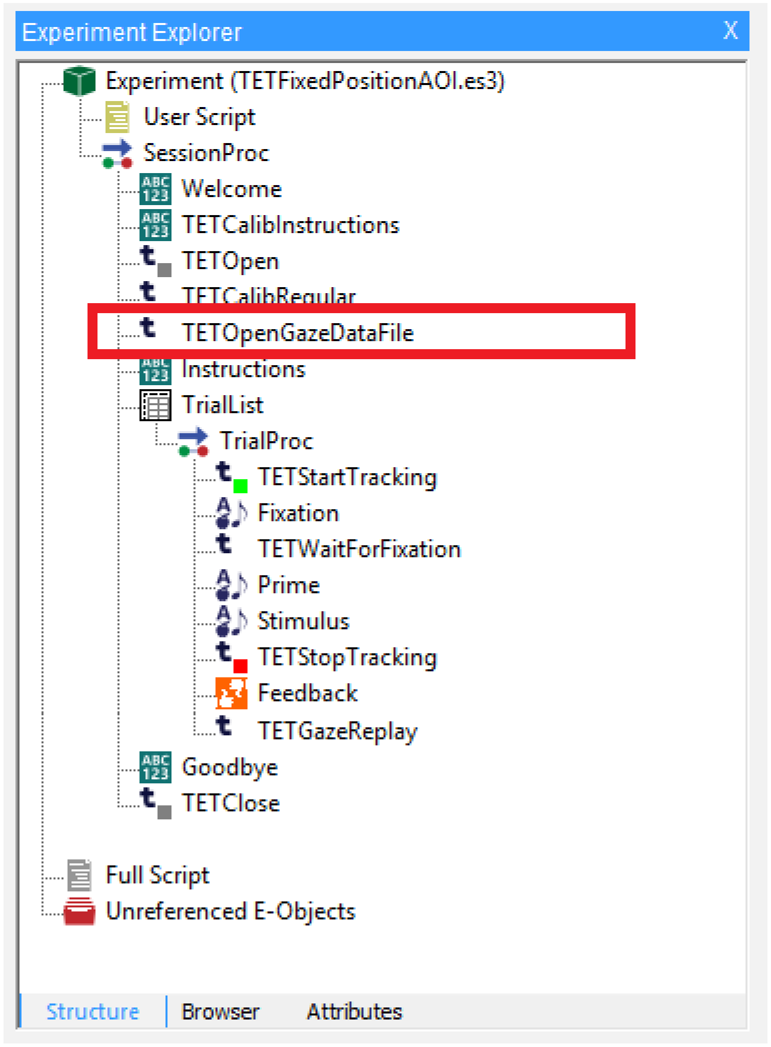
NOTE: *What data should you collect in your .gazedata file? As you will see, the starting design material provided includes the typical non-paradigm specific data you are likely to collect. It does not, however, include paradigm specific data which might be critical to your study. The subsequent tasks show you how to add this additional data. During this tutorial, script is added to the experiment that manipulates the format of the output file. Additionally, the HitTest method to discern the Areas of Interest (AOI) we are tracking in the experiment is discussed. AOIs are defined areas on the screen which represent critical objects or regions. AOIs are typically used in analysis or active runtime processing in E-Prime.*

Task 1: Add the TETOpenGazeDataFile PackageCall to Create and Open the Tab Delimited .gazedata File

Add a PackageCall to the SessionProc that opens the .gazedata file.

The **TETOpenGazeDataFile** PackageCall needs to be added to an experiment to open the Tab Delimited .gazedata file that is created (in addition to the .edat3 file) once an experiment has completed.

1) ***Drag*** the shortcut **TETOpenGazeDataFile** object from the **Tobii Pro Toolbox** and ***drop*** it where you want the .gazedata file opened (typically the SessionProc after Calibration).



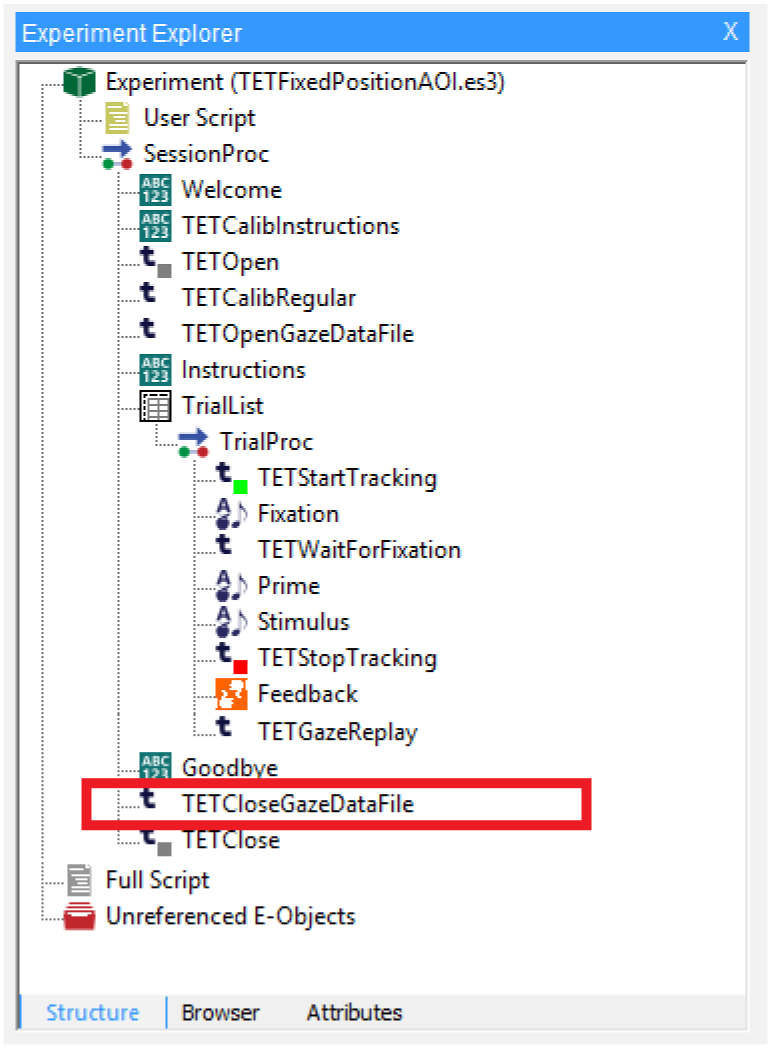
```
Experiment Explorer
Experiment (TETFixedPositionAOI.es3)
    User Script
    SessionProc
        Welcome
        TETCalibInstructions
        TETOpen
        TETCalibRegular
        TETOpenGazeDataFile
        Instructions
        TrialList
            TrialProc
                TETStartTracking
                Fixation
                TETWaitForFixation
                Prime
                Stimulus
                TETStopTracking
                Feedback
                TETGazeReplay
        Goodbye
        TETClose
Full Script
Unreferenced E-Objects
```

Task 2: Add the TETCloseGazeDataFile PackageCall to End Data Collection

Add a PackageCall at the end of the SessionProc that closes the. gazedata file.

At the end of the experiment, **TETCloseGazeDataFile** needs added to close the .gazedata file that was previously opened. This must occur whenever you are no longer saving any new .gazedata values in the file.

1) ***Drag*** the shortcut **TETCloseGazeDataFile** object from the **Tobii Pro Toolbox** and ***drop*** it after the **Goodbye Object** in the **SessionProc.**



Task 3: Copy the TETUserScript.txt into the E-Prime User Script

Open the TETUserScript.txt file, copy the contents, and paste them into the User Script tab in E-Studio.

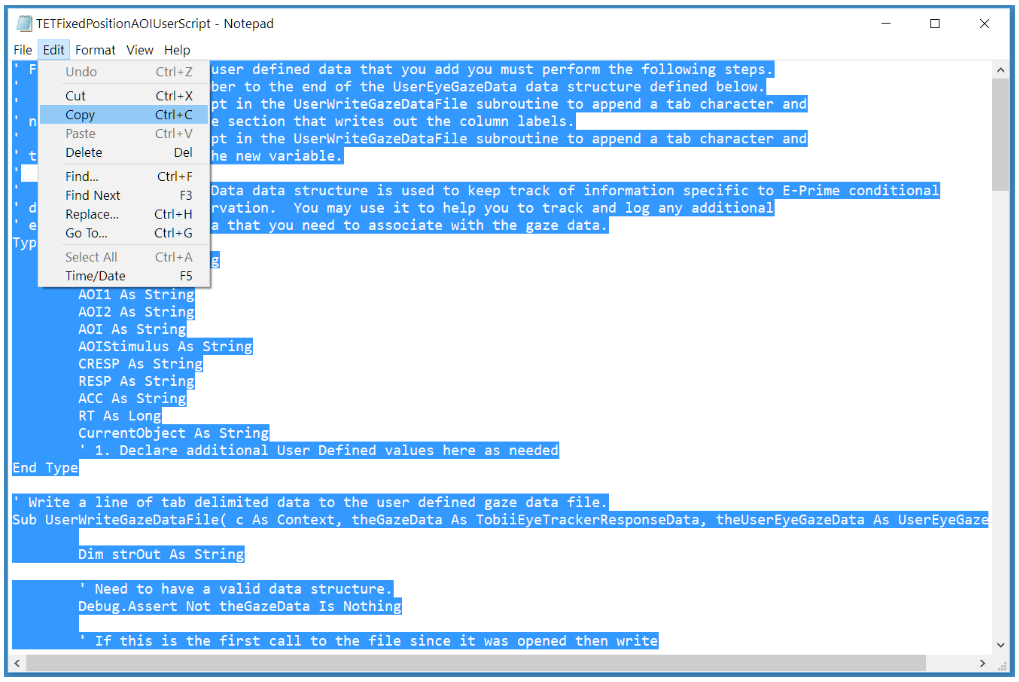
The **UserEyeGazeData** Type data structure is used to keep track of information specific to E-Prime conditional data per eye gaze observation. You may use it to help you to track and log any additional experiment related data that you need to associate with the gaze data. The quick and efficient way to get started is to copy the entire contents of the TETUserScript.txt included in *SaveGazeData in E-Prime Extensions for Tobii Pro 3.1* and paste it into the User Script window.

The script provided in TETUserScript.txt and TETFixedPositionAOISave GazeData.txt can be copied to every experiment that you create. They were written to be exported easily and contain commonly used variables. For each new experiment you create, you may need to alter some of the variables to be specific to that experiment.

1) ***Double click*** the **TETUserScript.txt** file to open the file in your default text editor application (e.g., Notepad).

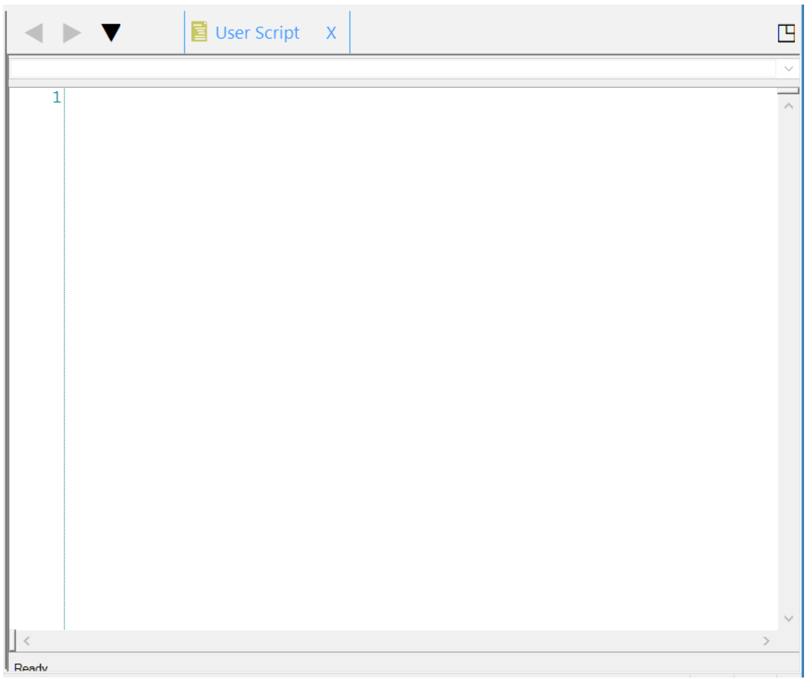
2) ***Select*** the **Edit** > **Select All (Ctrl+A)** to highlight the contents of the file.

3) ***Select*** **Edit** > **Copy** (**Ctrl+C**) to copy the contents of the file.

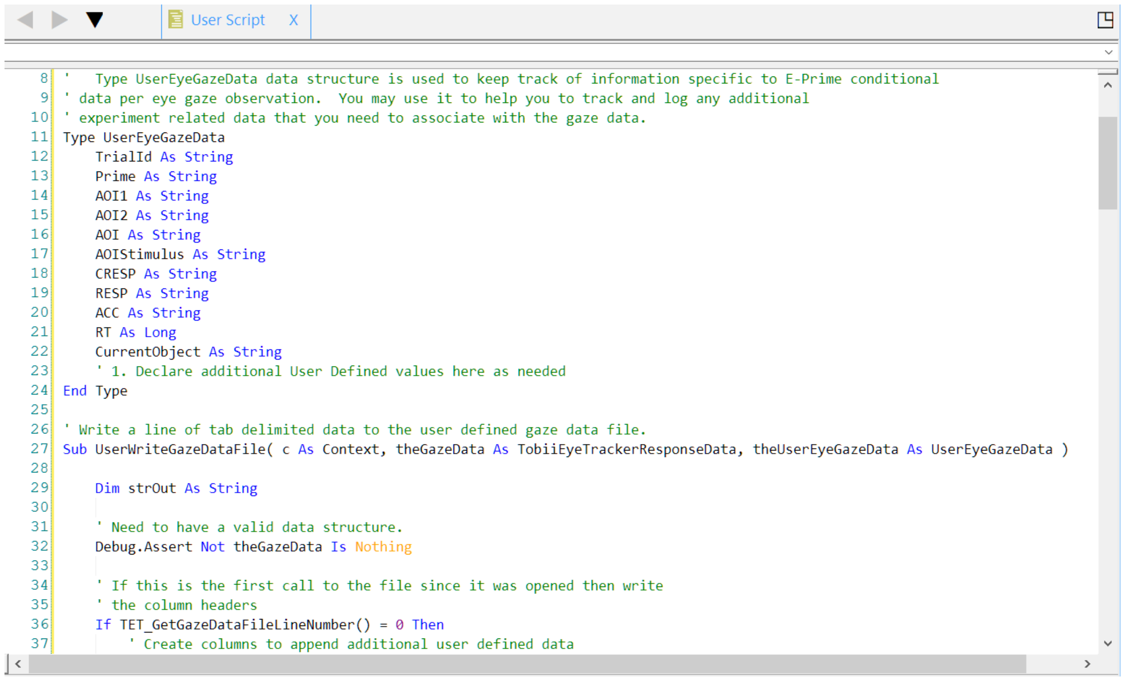


```
' F                        user defined data that you add you must perform the following steps.
'                          ber to the end of the UserEyeGazeData data structure defined below.
'                          pt in the UserWriteGazeDataFile subroutine to append a tab character and
' n                        e section that writes out the column labels.
'                          pt in the UserWriteGazeDataFile subroutine to append a tab character and
' t                        he new variable.
'
'                          Data data structure is used to keep track of information specific to E-Prime conditional
' d                        rvation.  You may use it to help you to track and log any additional
' e                        a that you need to associate with the gaze data.
Typ
                           g

        AOI1 As String
        AOI2 As String
        AOI As String
        AOIStimulus As String
        CRESP As String
        RESP As String
        ACC As String
        RT As Long
        CurrentObject As String
        ' 1. Declare additional User Defined values here as needed
End Type

' Write a line of tab delimited data to the user defined gaze data file.
Sub UserWriteGazeDataFile( c As Context, theGazeData As TobiiEyeTrackerResponseData, theUserEyeGazeData As UserEyeGaze

        Dim strOut As String

        ' Need to have a valid data structure.
        Debug.Assert Not theGazeData Is Nothing

        ' If this is the first call to the file since it was opened then write
```

4) ***Open*** the **User Script** window (Alt+7 to open) in **E-Studio**.



5) ***Paste*** the contents of the file into the **User Script** and then ***save*** the experiment file.



```
'   Type UserEyeGazeData data structure is used to keep track of information specific to E-Prime conditional
' data per eye gaze observation.  You may use it to help you to track and log any additional
' experiment related data that you need to associate with the gaze data.
Type UserEyeGazeData
    TrialId As String
    Prime As String
    AOI1 As String
    AOI2 As String
    AOI As String
    AOIStimulus As String
    CRESP As String
    RESP As String
    ACC As String
    RT As Long
    CurrentObject As String
    ' 1. Declare additional User Defined values here as needed
End Type

' Write a line of tab delimited data to the user defined gaze data file.
Sub UserWriteGazeDataFile( c As Context, theGazeData As TobiiEyeTrackerResponseData, theUserEyeGazeData As UserEyeGazeData )

    Dim strOut As String

    ' Need to have a valid data structure.
    Debug.Assert Not theGazeData Is Nothing

    ' If this is the first call to the file since it was opened then write
    ' the column headers
    If TET_GetGazeDataFileLineNumber() = 0 Then
        ' Create columns to append additional user defined data
```

Task 4: Understanding the UserEyeGazeData Type

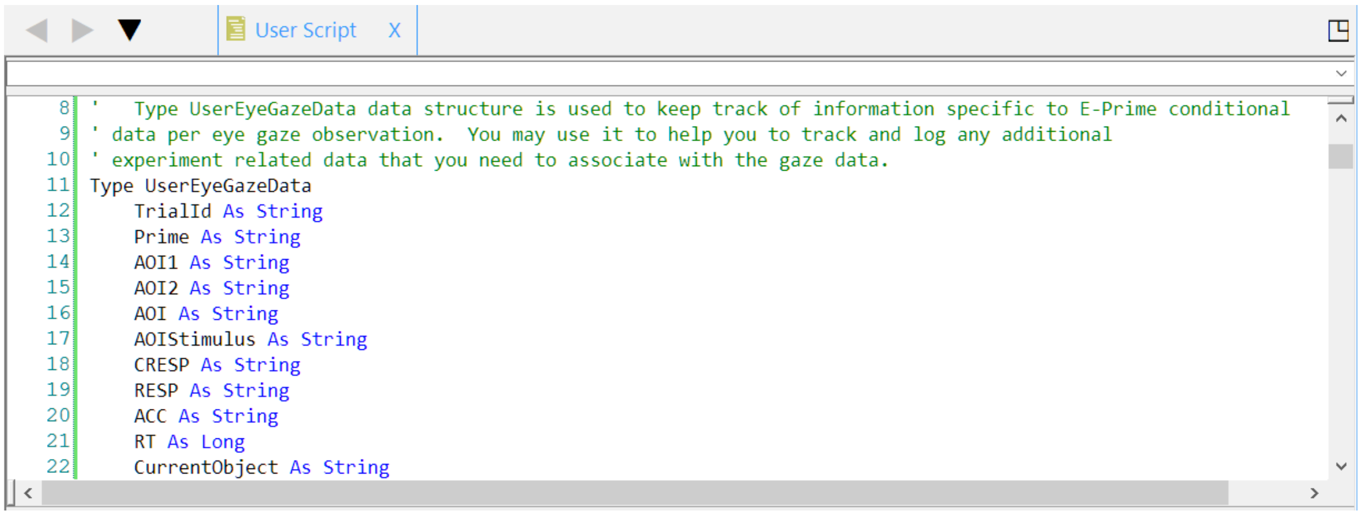
Examine the User Script.

The **UserEyeGazeData** Type is used to store any experimental data from E-Prime that is to be associated with the Tobii Pro eye gaze data in analysis. E-Prime provides access to a wealth of experimental and contextual data. Anything related to experimental design, stimulus presentation, user responses, timing audit results, and user calculated data can be captured and contextualized with eye gaze data. For example, if your experiment is displaying an image on a colored background, you may wish to log the color of the background. For a full range of options available to you, please refer to:

E-Prime Command Reference

Before proceeding, become familiar with the script you just pasted into the User Script window. The variables provided represent typical data collected in most studies. For readability, we recommend any additional User values be defined at the bottom of the Type structure. The order is not important if it is consistent ordering throughout the steps which follow. In some designs it may not be necessary to log the Prime, or the RT, etc. If you have determined that you do not need them in analysis, you may remove the values that are not necessary to your data file (so long as you make sure to delete them in the next two steps).

1) ***Locate*** the **Type UserEyeGazeData** in the User Script (Line 11).



```
 8  '   Type UserEyeGazeData data structure is used to keep track of information specific to E-Prime conditional
 9  ' data per eye gaze observation.  You may use it to help you to track and log any additional
10  ' experiment related data that you need to associate with the gaze data.
11  Type UserEyeGazeData
12      TrialId As String
13      Prime As String
14      AOI1 As String
15      AOI2 As String
16      AOI As String
17      AOIStimulus As String
18      CRESP As String
19      RESP As String
20      ACC As String
21      RT As Long
22      CurrentObject As String
```

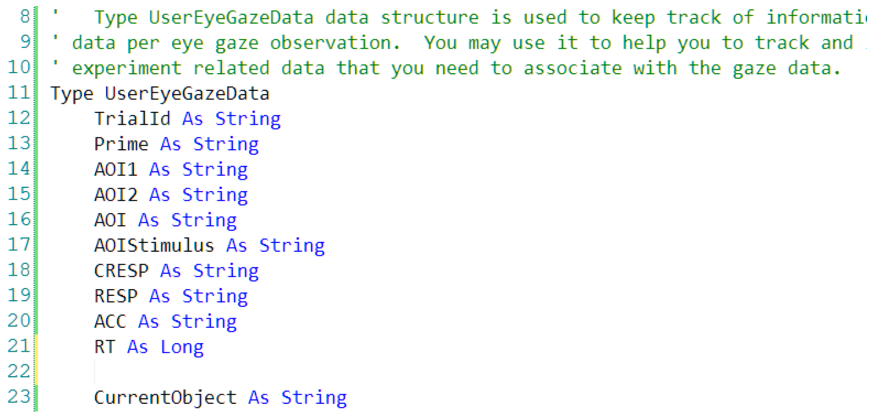
NOTE: *The numbers located to the left of the script window tell you what line you are at in the script. You can use this to find specific places within the script. For example, Type UserEyeGazeData is located at Line 11.*

Task 5: Add a New Member to the End of the UserEyeGazeData Type

Edit the script at the end of the UserEyeGazeData Type and declare the variable BackColor as a string.

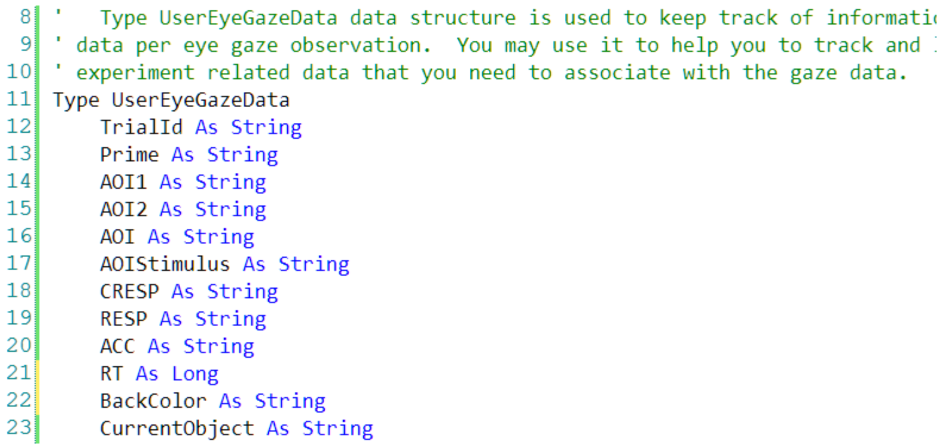
In this step, you will add the background color of the slide as a variable related to the context of the stimulus presentation. E-Prime determines the value for this variable and associates it with every eye gaze sample acquired while an instance of the slide was being presented.

1) ***Position*** your cursor after the **g** on the line that reads "**RT As Long**" (Line 21). ***Press*** **Enter.**



```
 8  '   Type UserEyeGazeData data structure is used to keep track of informati
 9  ' data per eye gaze observation.  You may use it to help you to track and
10  ' experiment related data that you need to associate with the gaze data.
11  Type UserEyeGazeData
12      TrialId As String
13      Prime As String
14      AOI1 As String
15      AOI2 As String
16      AOI As String
17      AOIStimulus As String
18      CRESP As String
19      RESP As String
20      ACC As String
21      RT As Long
22      
23      CurrentObject As String
```

2) ***Type*** "**BackColor As String**" (Line 22).



```
 8  '   Type UserEyeGazeData data structure is used to keep track of informati
 9  ' data per eye gaze observation.  You may use it to help you to track and
10  ' experiment related data that you need to associate with the gaze data.
11  Type UserEyeGazeData
12      TrialId As String
13      Prime As String
14      AOI1 As String
15      AOI2 As String
16      AOI As String
17      AOIStimulus As String
18      CRESP As String
19      RESP As String
20      ACC As String
21      RT As Long
22      BackColor As String
23      CurrentObject As String
```

Task 6: Edit User Script to Append a Column for the BackColor Variable

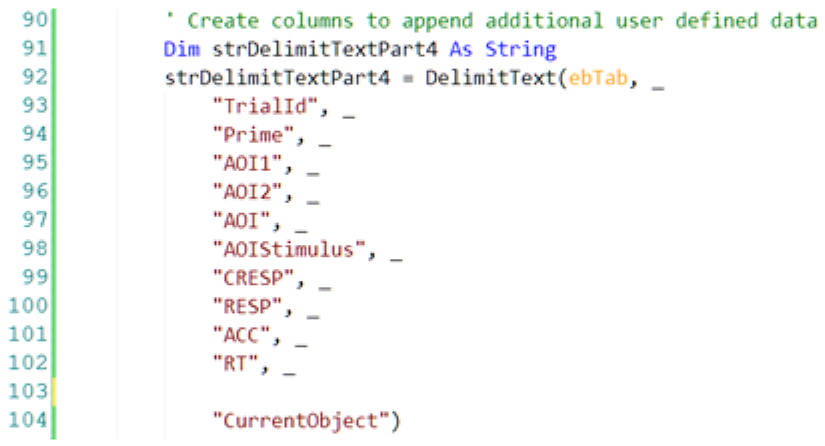
Edit the script labeled 'Create columns to append additional user defined data to label the BackColor column.

To add a column to the .gazedata file for the BackColor variable, you need to add a line for each of the new data values you created in the previous step. This can be done in a section of the script labeled "Create columns to append additional user defined data." The ebTab command is a constant that is used to create columns in the data file by creating a tab. Using the proper syntax, you can list the name of the variables you want to create columns for. For readability purposes, keep the order consistent.

1) ***Locate*** 'Create columns to append additional user defined data in the ebTab group in the **User Script** (Line 90).

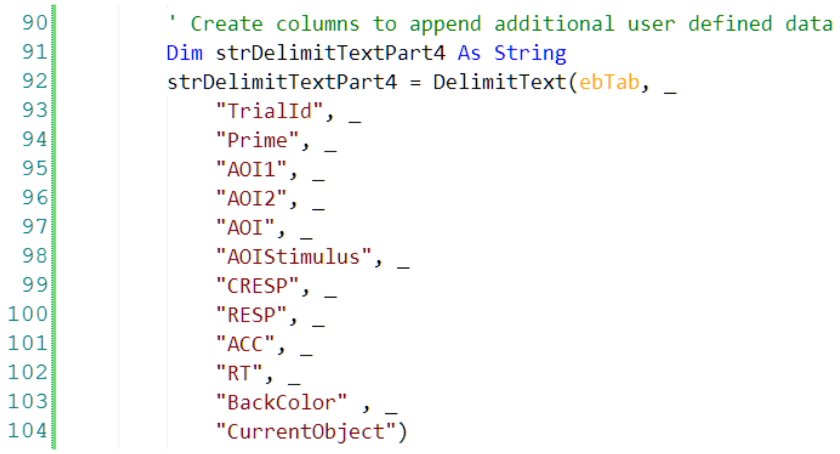
2) ***Position*** your cursor after the _ (underscore) on the line that reads **"RT", _** (Line 102).

3) ***Press* Enter**.



```
90          ' Create columns to append additional user defined data
91          Dim strDelimitTextPart4 As String
92          strDelimitTextPart4 = DelimitText(ebTab, _
93              "TrialId", _
94              "Prime", _
95              "AOI1", _
96              "AOI2", _
97              "AOI", _
98              "AOIStimulus", _
99              "CRESP", _
100             "RESP", _
101             "ACC", _
102             "RT", _
103
104             "CurrentObject")
```

4) ***Type* "BackColor", _**" (Line 103).



```
90          ' Create columns to append additional user defined data
91          Dim strDelimitTextPart4 As String
92          strDelimitTextPart4 = DelimitText(ebTab, _
93              "TrialId", _
94              "Prime", _
95              "AOI1", _
96              "AOI2", _
97              "AOI", _
98              "AOIStimulus", _
99              "CRESP", _
100             "RESP", _
101             "ACC", _
102             "RT", _
103             "BackColor" , _
104             "CurrentObject")
```

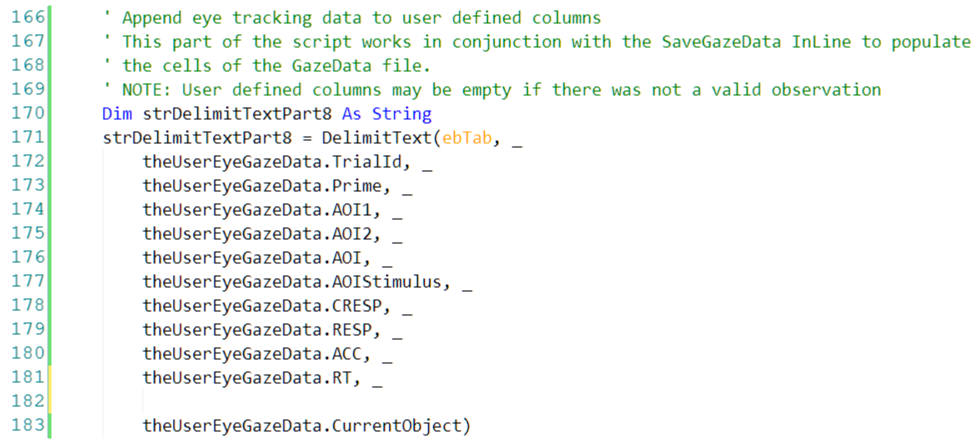
Task 7: Edit User Script to Append Value of BackColor

Edit the script labeled 'Append eye tracking data to user defined columns in the UserEyeGazeData Group to include the value of the BackColor variable to the data file.

You need to add an ebTab line in theUserEyeGazeData group for each of the new data values you created in the previous step. This uses the SaveGazeData InLine (that is created in a future step) to populate the values in the .gazedata file. The order of the variables is very important and needs to match the order of the columns from the previous step, as the ebTab command is only able to populate the cells in the order that it is given. The order of the columns listed in the "Create columns to append additional user defined data" section and the order of the variables in the "Append eye tracking data to user defined columns" need to correspond so the data lines up properly. This ensures that the correct data is populated under the correct column.

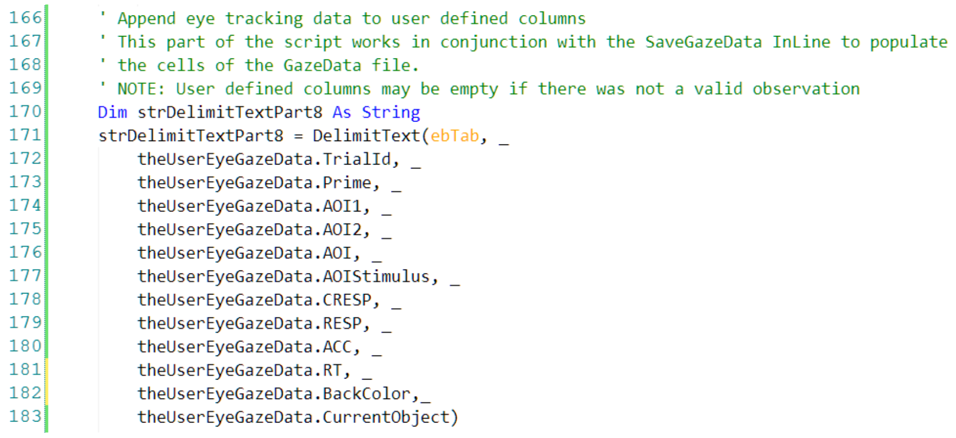
1) ***Locate*** ' Append eye tracking data to user defined columns near the end of the **User Script** (Line 166).

2) ***Position*** your cursor after the _ (underscore) on the line that reads **theUserEyeGazeData.RT , _** (Line 181)**. *Press* Enter.**



```
166      ' Append eye tracking data to user defined columns
167      ' This part of the script works in conjunction with the SaveGazeData InLine to populate
168      ' the cells of the GazeData file.
169      ' NOTE: User defined columns may be empty if there was not a valid observation
170      Dim strDelimitTextPart8 As String
171      strDelimitTextPart8 = DelimitText(ebTab, _
172          theUserEyeGazeData.TrialId, _
173          theUserEyeGazeData.Prime, _
174          theUserEyeGazeData.AOI1, _
175          theUserEyeGazeData.AOI2, _
176          theUserEyeGazeData.AOI, _
177          theUserEyeGazeData.AOIStimulus, _
178          theUserEyeGazeData.CRESP, _
179          theUserEyeGazeData.RESP, _
180          theUserEyeGazeData.ACC, _
181          theUserEyeGazeData.RT, _
182
183          theUserEyeGazeData.CurrentObject)
```

3) ***Type*** **theUserEyeGazeData.BackColor , _** (Line 182).



```
166      ' Append eye tracking data to user defined columns
167      ' This part of the script works in conjunction with the SaveGazeData InLine to populate
168      ' the cells of the GazeData file.
169      ' NOTE: User defined columns may be empty if there was not a valid observation
170      Dim strDelimitTextPart8 As String
171      strDelimitTextPart8 = DelimitText(ebTab, _
172          theUserEyeGazeData.TrialId, _
173          theUserEyeGazeData.Prime, _
174          theUserEyeGazeData.AOI1, _
175          theUserEyeGazeData.AOI2, _
176          theUserEyeGazeData.AOI, _
177          theUserEyeGazeData.AOIStimulus, _
178          theUserEyeGazeData.CRESP, _
179          theUserEyeGazeData.RESP, _
180          theUserEyeGazeData.ACC, _
181          theUserEyeGazeData.RT, _
182          theUserEyeGazeData.BackColor,_
183          theUserEyeGazeData.CurrentObject)
```

Task 8: Add an InLine Object to the TrialProc

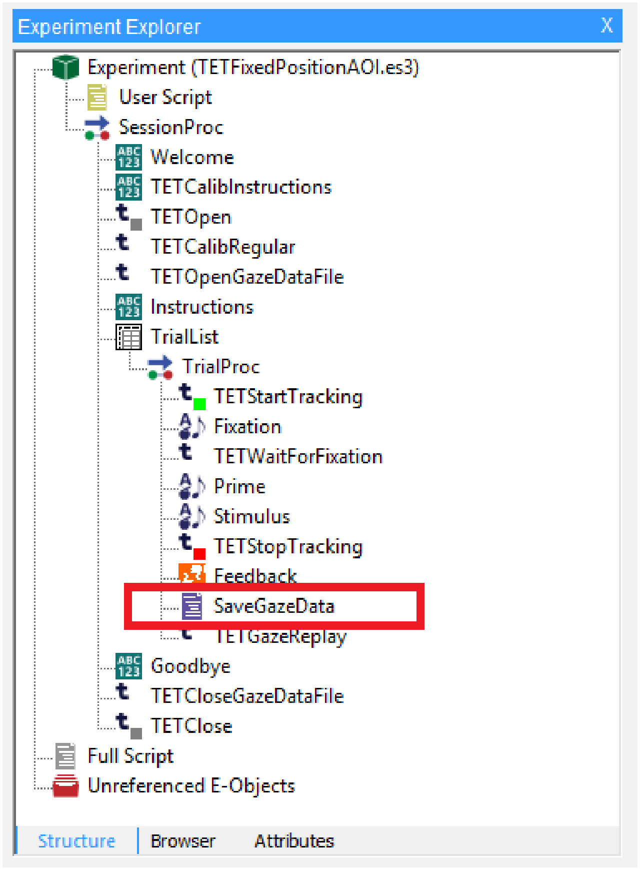
Add an InLine Object to the TrialProc and rename it to SaveGazeData.

The **SaveGazeData** InLine Object is used to collect the eye tracking data and works in conjunction with the User Script to write the data to a tab delimited .gazedata file. The .gazedata file can be viewed via Excel, or a similar spreadsheet application. The first step in creating the SaveGazeData InLine is to add an InLine Object to the experiment after you have stopped collecting eye tracking data. In this experiment, the InLine should follow the TETStopTracking PackageCall.

NOTE: *The script provided in TETUserScript.txt and TETSaveGazeData.txt can be copied to every EET 3.1 experiment that you create. They were written to be exported easily and contain commonly used variables. For each new experiment you create, you may need to alter some of the variables or script to be specific to that experiment.*

1) ***Drag*** a new **InLine Object** from the E-Prime Toolbox and ***drop*** it where you want the gaze data saved (typically after the TETStopTracking PackageCall).

2) ***Rename*** (F2) the object to **SaveGazeData**. ***Press*** **Enter** to accept the change.



3) ***Double click*** **SaveGazeData** to open it in the workspace.

NOTE: *In EET 3.2., the PackageCall TETGazeDataSave needs to be placed after TETGazeReplay.* In EET 3.1, the SaveGazeData InLine can occur before TETGazeReplay occurs.

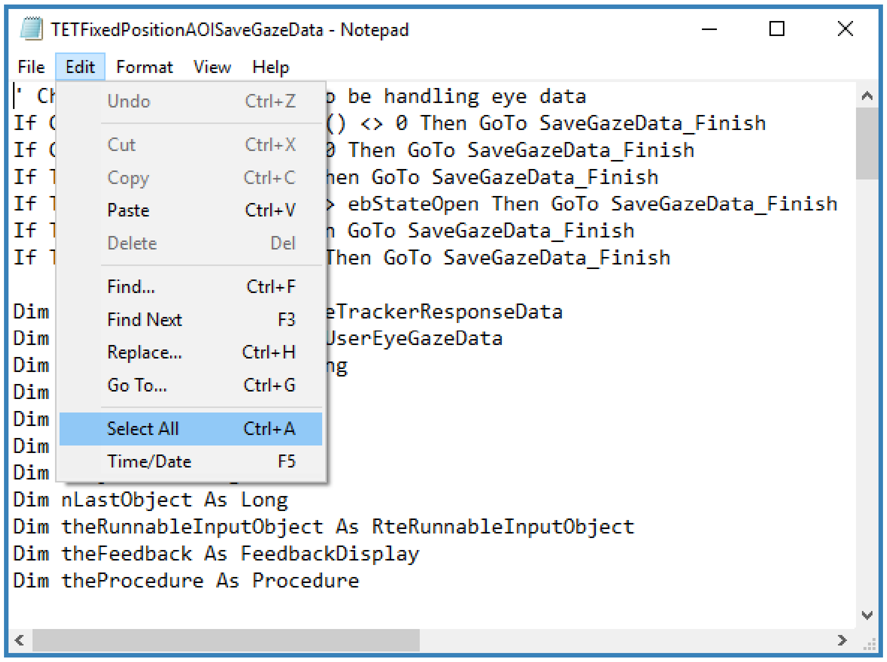
Task 9: Copy Script from TETSaveGazeData.txt to SaveGazeData InLine

Open the TETSaveGazeData.txt file, copy the contents, and paste them into the SaveGazeData InLine in E-Studio.

Creating the .gazedata file varies greatly from experiment to experiment depending on what experimental variables you want to include with each gaze data sample. The quick and efficient way to get started is to copy the entire contents of the TETUserScript.txt included in [*SaveGazeData in E-Prime Extensions for Tobii Pro 3.1*]() and paste it into the SaveGazeData InLine object.

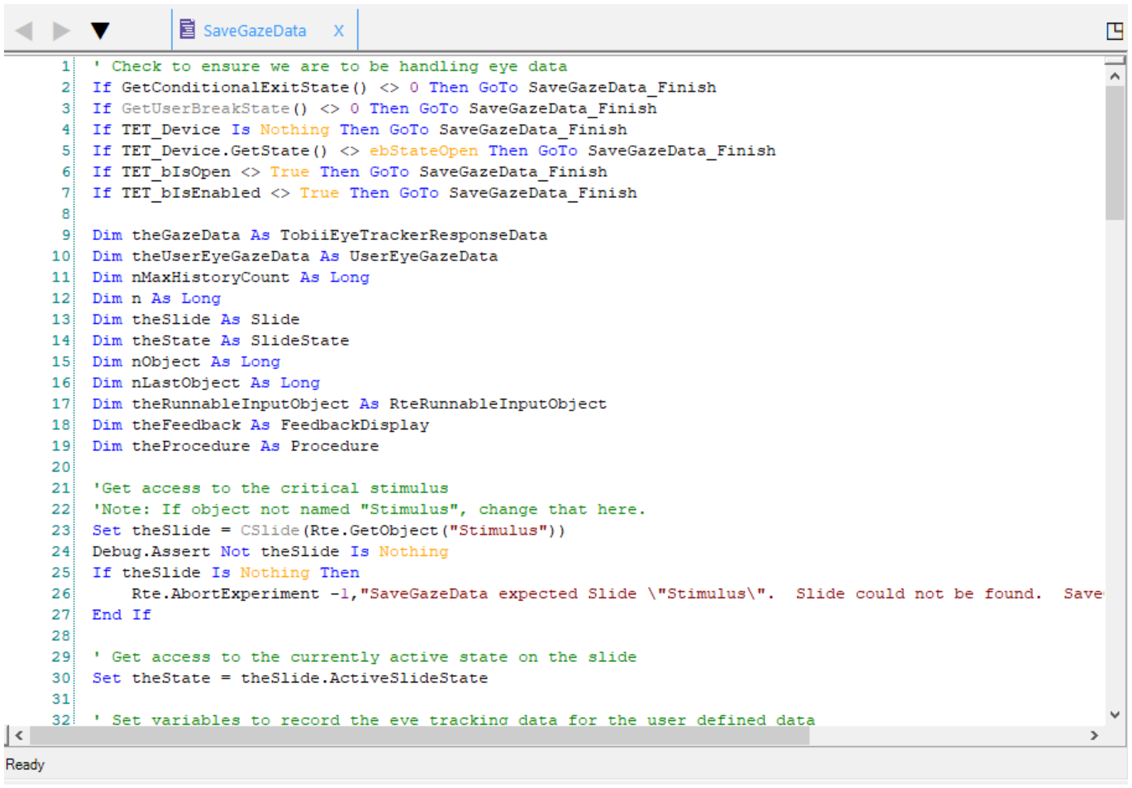
1) ***Double click*** the **TETSaveGazeData.txt** file to open the file.

2) ***Select*** the **Edit** > **Select All (Ctrl+A)** to highlight the contents of the file. **Edit** > **Copy (Ctrl+C)** to copy the contents of the file.



```
' Ch                       o be handling eye data
If (                       () <> 0 Then GoTo SaveGazeData_Finish
If (                       0 Then GoTo SaveGazeData_Finish
If T                       hen GoTo SaveGazeData_Finish
If T                       > ebStateOpen Then GoTo SaveGazeData_Finish
If T                       n GoTo SaveGazeData_Finish
If T                       Then GoTo SaveGazeData_Finish

Dim                        eTrackerResponseData
Dim                        UserEyeGazeData
Dim                        ng
Dim
Dim
Dim
Dim
Dim nLastObject As Long
Dim theRunnableInputObject As RteRunnableInputObject
Dim theFeedback As FeedbackDisplay
Dim theProcedure As Procedure
```

3) ***Open*** **SaveGazeData** in **E-Studio**, ***select*** **Edit** > **Paste (Ctrl+V)** to ***paste*** the contents of the file in **E-Studio**.



SaveGazeData X

```
' Check to ensure we are to be handling eye data
If GetConditionalExitState() <> 0 Then GoTo SaveGazeData_Finish
If GetUserBreakState() <> 0 Then GoTo SaveGazeData_Finish
If TET_Device Is Nothing Then GoTo SaveGazeData_Finish
If TET_Device.GetState() <> ebStateOpen Then GoTo SaveGazeData_Finish
If TET_bIsOpen <> True Then GoTo SaveGazeData_Finish
If TET_bIsEnabled <> True Then GoTo SaveGazeData_Finish

Dim theGazeData As TobiiEyeTrackerResponseData
Dim theUserEyeGazeData As UserEyeGazeData
Dim nMaxHistoryCount As Long
Dim n As Long
Dim theSlide As Slide
Dim theState As SlideState
Dim nObject As Long
Dim nLastObject As Long
Dim theRunnableInputObject As RteRunnableInputObject
Dim theFeedback As FeedbackDisplay
Dim theProcedure As Procedure

'Get access to the critical stimulus
'Note: If object not named "Stimulus", change that here.
Set theSlide = CSlide(Rte.GetObject("Stimulus"))
Debug.Assert Not theSlide Is Nothing
If theSlide Is Nothing Then
    Rte.AbortExperiment -1,"SaveGazeData expected Slide \"Stimulus\".  Slide could not be found.  Save
End If

' Get access to the currently active state on the slide
Set theState = theSlide.ActiveSlideState

' Set variables to record the eye tracking data for the user defined data
```

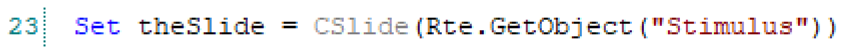
Ready

Task 10: View and Edit the SaveGazeData InLine

View the content that you just pasted into the SaveGazeData InLine and confirm theSlide is referencing the correct Slide Object.

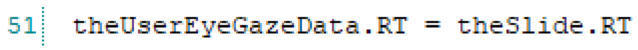
The contents of SaveGazeData need modified once they are pasted into the InLine Object. The script that was copied in the previous steps was not paradigm specific, so it could be imported into other experiments. For each new experiment you need to check that the variables are set correctly. The first variable we need to check is the Slide variable. This variable needs to correspond to the object that displays your critical stimulus. In this experiment, the critical stimulus (i.e., the Slide named Stimulus) is the object that displays the images to the left and right visual fields.

1) ***Confirm*** **Set theSlide = CSlide (Rte.Getobject("Stimulus"))**



```
Set theSlide = CSlide(Rte.GetObject("Stimulus"))
```

2) ***Position*** your cursor at the end of the line that reads **"theUserEyeGazeData.RT = theSlide.RT".**

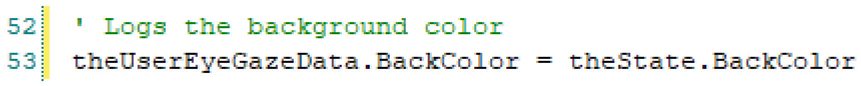


```
theUserEyeGazeData.RT = theSlide.RT
```

3) ***Press*** **Enter**. ***Type*** 'Logs the background color.

   NOTE: *It is important to start the line with a single quote; this indicates that the characters which follow the quote are a comment and not E-Basic script.*

4) ***Press*** **Enter**. ***Type*** **theUserEyeGazeData. BackColor = theState.BackColor**
   The background color of the Slide Object will now be written out to the .gazedata file.



```
' Logs the background color
theUserEyeGazeData.BackColor = theState.BackColor
```

Task 11: Understand theUserEyeGazeData Type Variable Equivalents

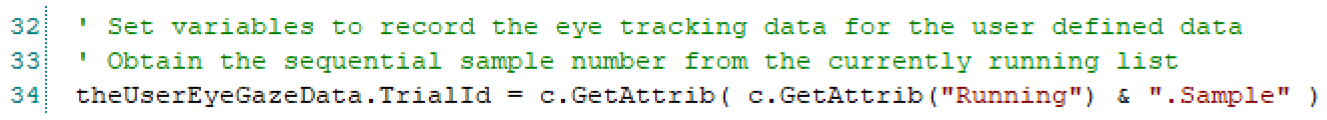
Understand how the variables in theUserEyeGazeData type correspond to other objects in the experiment.

The next section of script reviews the variables that are used to assign a unique ID to the trial and collect other information about the trial. AOI is defined as an Area of Interest. Look over the variable descriptions below to get a basic understanding of how they function in the script. This differs experiment to experiment depending on the variable names and the data that is being saved in the .gazedata file.

NOTE: *The methods employed in the SaveGazeData InLine are not designed to be executed when E-Prime is performing tasks related to data collection and stimulus presentation. In most experimental designs this is performed between trials. In this tutorial, samples from the eye tracker are stored in memory until the end of the trial, at which point this script is executed.*

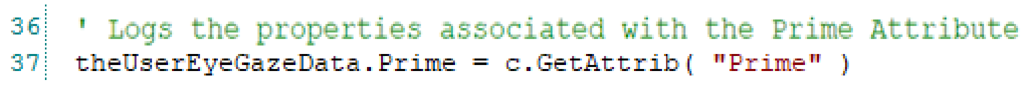
Once the trial is over, the SaveGazeData InLine processes all the eye gaze data collected since TET_StartTracking was called. The SaveGazeData InLine then performs any calculations required to fill the variables pertaining to the current trial. This is accomplished by a loop that runs through each of the acquired .gazedata points and logs the appropriate E-Prime related data.

1) **theUserEyeGazeData.TrialId:** Obtains the sequential sample number from the currently running list.



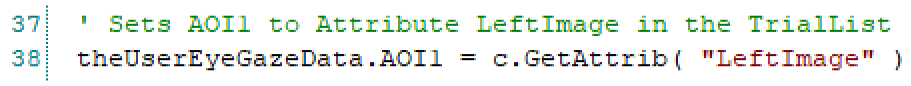
```
' Set variables to record the eye tracking data for the user defined data
' Obtain the sequential sample number from the currently running list
theUserEyeGazeData.TrialId = c.GetAttrib( c.GetAttrib("Running") & ".Sample" )
```

2) **theUserEyeGazeData.Prime:** Logs the properties associated with the Prime Attribute.



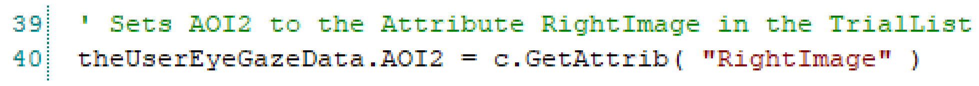
```
' Logs the properties associated with the Prime Attribute
theUserEyeGazeData.Prime = c.GetAttrib( "Prime" )
```

3) **theUserEyeGazeData.AOI1:** Sets AOI1 to Attribute LeftImage in the TrialList.



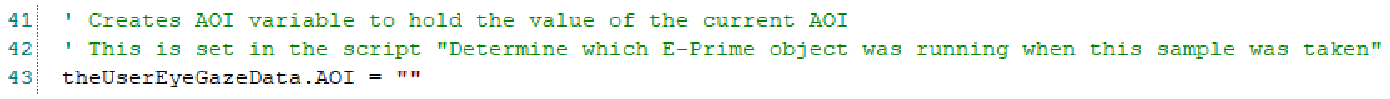
```
' Sets AOI1 to Attribute LeftImage in the TrialList
theUserEyeGazeData.AOI1 = c.GetAttrib( "LeftImage" )
```

4) **theUserEyeGazeData.AOI2:** Sets AOI2 to the Attribute RightImage in the TrialList.



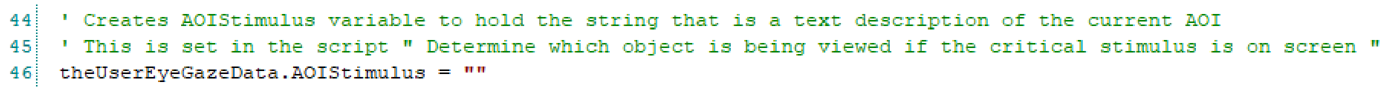
```
' Sets AOI2 to the Attribute RightImage in the TrialList
theUserEyeGazeData.AOI2 = c.GetAttrib( "RightImage" )
```

5) **theUserEyeGazeData.AOI:** Creates AOI variable to hold the value of the current AOI. This is set in the script “Determine which E-Prime object was running when this sample was taken”.



```
' Creates AOI variable to hold the value of the current AOI
' This is set in the script "Determine which E-Prime object was running when this sample was taken"
theUserEyeGazeData.AOI = ""
```

6) **theUserEyeGazeData.AOIStimulus:** Creates AOIStimulus variable to hold the string that is a text description of the current AOI. This is set in the script “Determine which object is being viewed if the critical stimulus is on screen.



```
' Creates AOIStimulus variable to hold the string that is a text description of the current AOI
' This is set in the script " Determine which object is being viewed if the critical stimulus is on screen "
theUserEyeGazeData.AOIStimulus = ""
```

Task 12: Determining the objects on Screen during each Gaze Data Sample

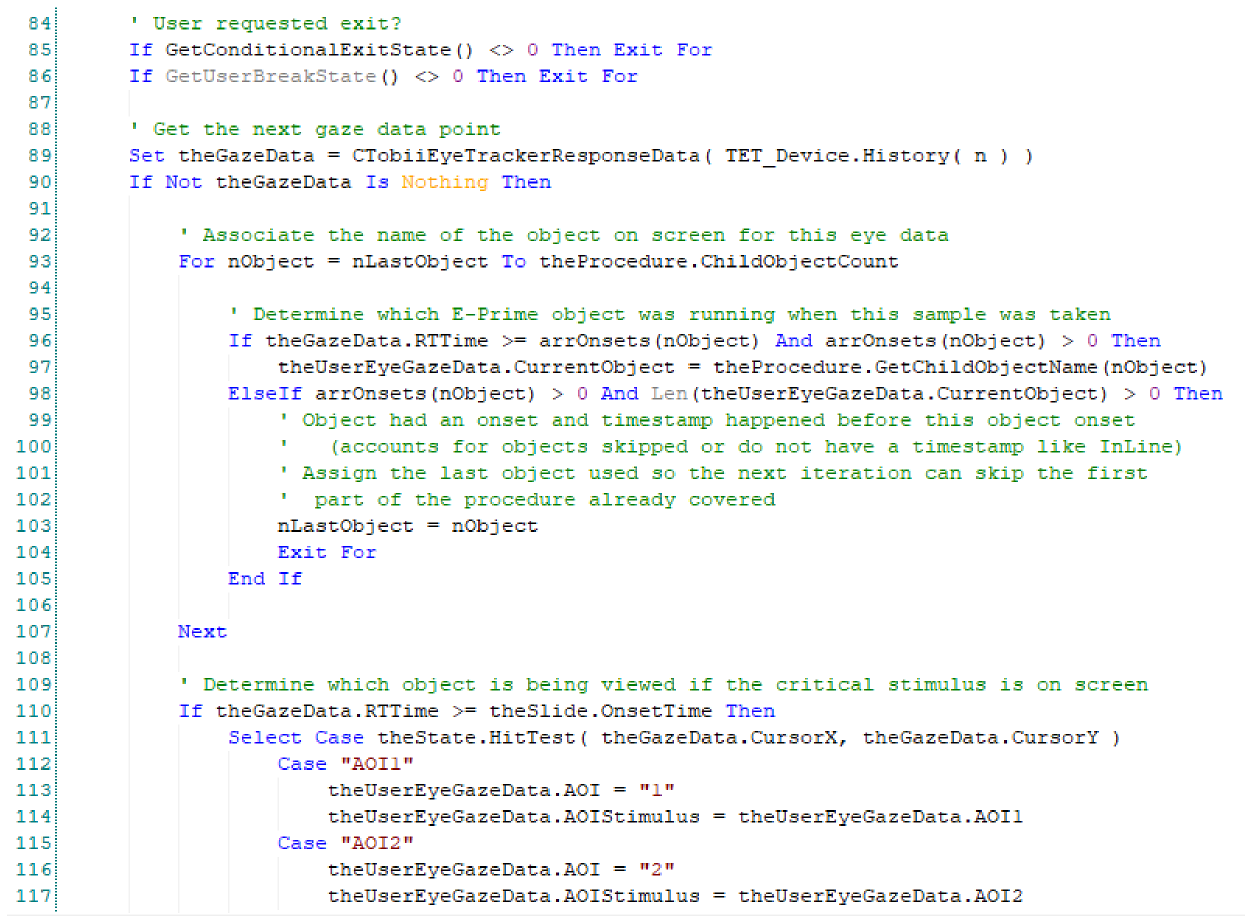
Use the .OnsetTime property to log the time at which the object was placed on the screen.

The screen image below shows how E-Basic script associates eye tracking data with the E-Prime object that is currently displayed on the screen. The RTTime property of the data structure named theGazeData provides a timestamp of when the current .gazedata point was delivered to E-Prime. This value is then compared to the time at which each object in the TrialProc began to execute to determine which object was being displayed when the .gazedata point was recorded.

The SaveGazeData InLine Object contains script which first loops through all the objects in the TrialProc up to the Feedback Object and retrieves their OnsetTime property. The script shown below shows the comparison of this value to the RTTime.

Both sections of script are written generically: if you add, remove, or rename any object on the TrialProc, up to the Feedback Object, this script still operates as intended. You do not need to modify any of this script even if you modify the objects on the TrialProc.

1) ***Compare*** the eye gaze data time **(theGazeData.RTTime)** to the object's onset time **(arrOnsets(nobject)**)".



```
    ' User requested exit?
    If GetConditionalExitState() <> 0 Then Exit For
    If GetUserBreakState() <> 0 Then Exit For

    ' Get the next gaze data point
    Set theGazeData = CTobiiEyeTrackerResponseData( TET_Device.History( n ) )
    If Not theGazeData Is Nothing Then

        ' Associate the name of the object on screen for this eye data
        For nObject = nLastObject To theProcedure.ChildObjectCount

            ' Determine which E-Prime object was running when this sample was taken
            If theGazeData.RTTime >= arrOnsets(nObject) And arrOnsets(nObject) > 0 Then
                theUserEyeGazeData.CurrentObject = theProcedure.GetChildObjectName(nObject)
            ElseIf arrOnsets(nObject) > 0 And Len(theUserEyeGazeData.CurrentObject) > 0 Then
                ' Object had an onset and timestamp happened before this object onset
                '   (accounts for objects skipped or do not have a timestamp like InLine)
                ' Assign the last object used so the next iteration can skip the first
                '  part of the procedure already covered
                nLastObject = nObject
                Exit For
            End If

        Next

        ' Determine which object is being viewed if the critical stimulus is on screen
        If theGazeData.RTTime >= theSlide.OnsetTime Then
            Select Case theState.HitTest( theGazeData.CursorX, theGazeData.CursorY )
                Case "AOI1"
                    theUserEyeGazeData.AOI = "1"
                    theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI1
                Case "AOI2"
                    theUserEyeGazeData.AOI = "2"
                    theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI2
```

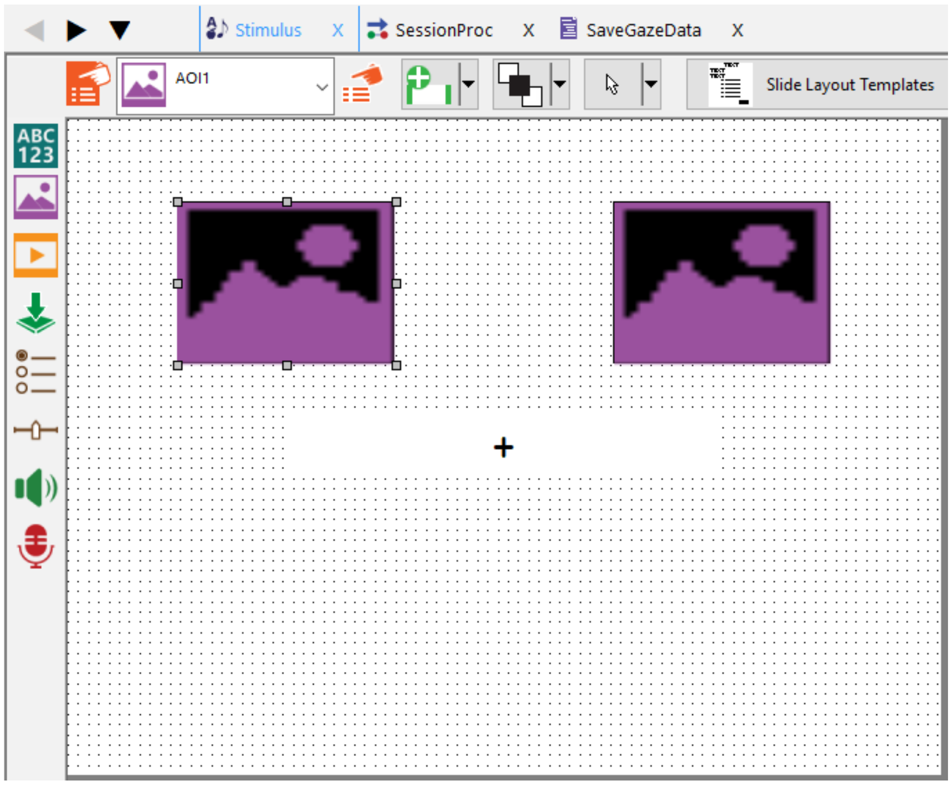
Task 13: Declare AOIs as Slide Sub-Objects

Associate Area of Interest (AOI) objects with eye gaze data.

Recall that we previously defined an **Area of Interest** (AOIs) as areas on the screen which represent critical objects or regions. AOIs are typically used to assist in analysis or in active runtime processing in E-Prime. Now we need to populate the AOI columns we declared in the User Script with data. In an earlier task (*Task 10: View and Edit the SaveGazeData InLine*) we defined the critical stimulus as the Stimulus Object. The Stimulus Object contains three Sub-Objects; Fixation, AOI1, and AOI2.

We have already associated the AOI1 label to the images presented on the left side of the screen and the AOI2 label to the images presented on the right side of the screen in the script when the experiment was originally created.

1) ***Select* AOI1**. When AOI1 is selected from the dropdown list, the image associated with the name, AOI1, becomes indicated with gray boxes.



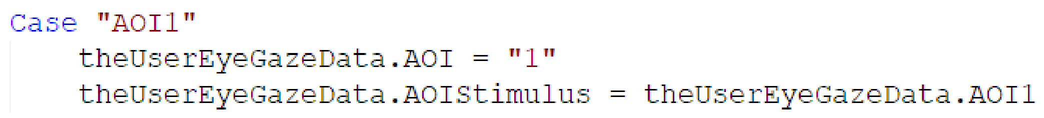
Task 14: The HitTest Method

Use the HitTest Method to determine which Sub-Object a given .gazedata point is within.

The HitTest Method determines the on-screen location of the cursor or mouse pointer on a Slide Object by using the X and Y coordinates. The .gazedata point can be recorded with this method by substituting the .CursorX and .CursorY. This allows you to determine if the current point was within the given Sub-Object by referencing the name of the Slide Sub-Object, e.g., AOI1, AOI2, Fixation. The "Else" case logs a null value for the AOI if the given .gazedata point is not within any of the AOIs. You need to add to the script below to include any of the possible AOIs which you present in addition to removing the AOIs that do not exist in your experiment.

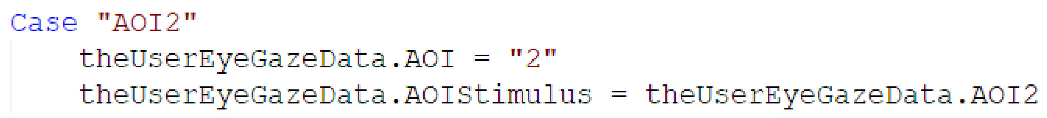
1) Command line that executes **HitTest**.

2) **Case "AOI1"**: ***Defines*** the **criteria** for the **1st AOI**.
   Assigns the **current AOI** to **1** and ***populates*** **AOIStimulus** with a **text description** of AOI1.



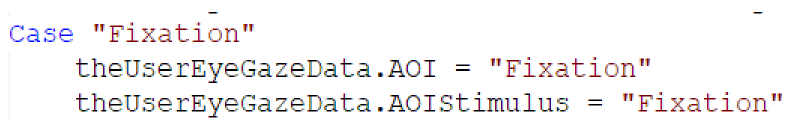
```
Case "AOI1"
    theUserEyeGazeData.AOI = "1"
    theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI1
```

3) **Case "AOI2"**: ***Defines*** the **criteria** for the **2nd AOI**.
   Assigns the **current AOI** to **2** and ***populates*** **AOIStimulus** with a **text description** of AOI2.



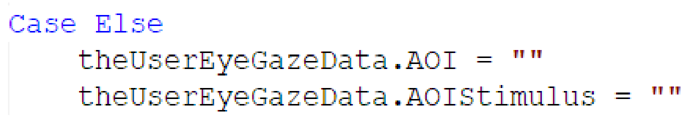
```
Case "AOI2"
    theUserEyeGazeData.AOI = "2"
    theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI2
```

4) **Case "Fixation"**: ***Defines*** the **criteria** for the **Fixation** condition.
   Assigns the **current AOI** to **Fixation** and ***populates*** **AOIStimulus** with the text **"Fixation"**.



```
Case "Fixation"
    theUserEyeGazeData.AOI = "Fixation"
    theUserEyeGazeData.AOIStimulus = "Fixation"
```

5) **Case "Else"**: ***Defines*** the **criteria** for the **AOI null** condition.
   Assigns the **current AOI** to an **empty string** and ***populates*** **AOIStimulus** with an **empty string** (makes it blank.)



```
Case Else
    theUserEyeGazeData.AOI = ""
    theUserEyeGazeData.AOIStimulus = ""
```

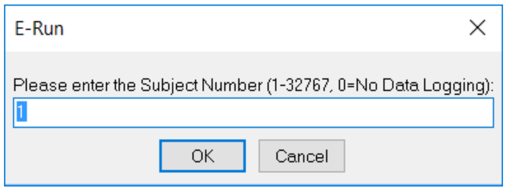
Task 15: Run the Experiment

Run the experiment to verify that the eye tracker is working and to ensure the .gazedata file is written.

Now you can run the experiment to generate a .gazedata file.

1) ***Click*** the **Run icon** on the toolbar to generate and run the experiment locally.

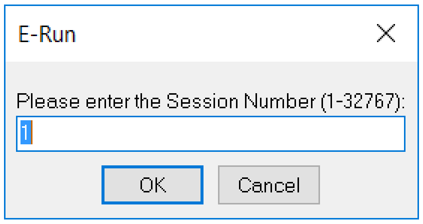
2) ***Press* Enter** to ***accept*** the **default values** for each of the initialization prompts presented.



E-Run

Please enter the Subject Number (1-32767, 0=No Data Logging):

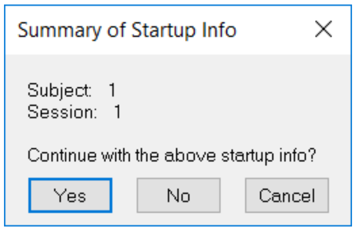
1



E-Run

Please enter the Session Number (1-32767):

1

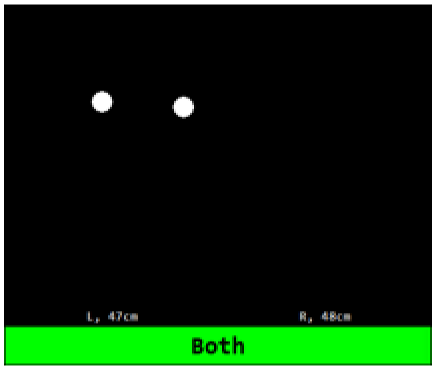


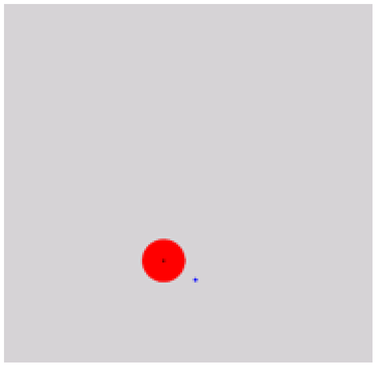
Summary of Startup Info

Subject: 1
Session: 1

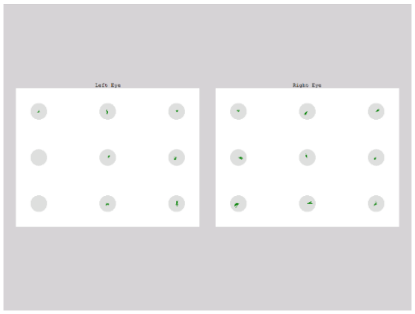
Continue with the above startup info?

3) ***Follow*** the **prompts on screen** to ***complete*** the **calibration** sequence.





4) ***Accept*** the **calibration** and ***perform*** the **experiment**.



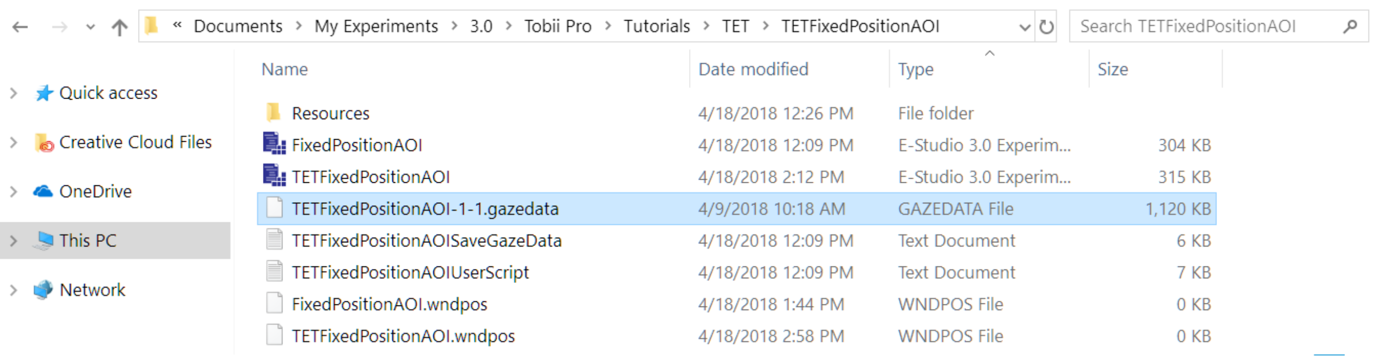
5) ***Observe*** the **stimulus presentation sequence** to ***verify*** the **experiment** is ***functioning*** correctly and ***completes*** with **no errors** being generated.

Task 16: Open the Eye .gazedata File

Verify the .gazedata file exists and can be opened.

The .gazedata file contains information about the stimulus presentation (e.g., what was on screen and when it was on screen, the eye tracking data, where the eyes were looking). This file facilitates data analysis. After the experiment is created, it is important that you check this file thoroughly to make sure all the components you need for your data analysis are included in the file before you begin running study participants. In this tutorial, you simply verify that the file exists and can be opened. In the next tutorial, we explain the gazedata output in detail. A filename is created in the form of DataFile.BaseName. DataFile.BaseName defaults to [ExperimentName]-[Subject#]-[Session#]. In this example the file is named TETFixedPositionAOI-1-1.gazedata (where Subject = 1, Session = 1).

1) ***Navigate*** to **your experiment folder**.



« Documents › My Experiments › 3.0 › Tobii Pro › Tutorials › TET › TETFixedPositionAOI

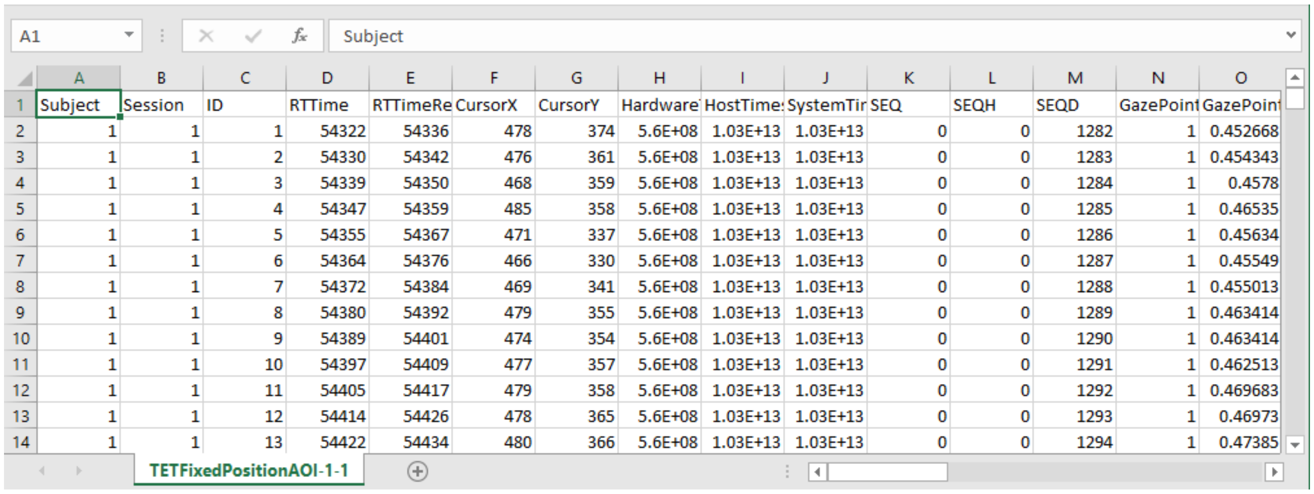
Search TETFixedPositionAOI

Name	Date modified	Type	Size
Resources	4/18/2018 12:26 PM	File folder	
FixedPositionAOI	4/18/2018 12:09 PM	E-Studio 3.0 Experim...	304 KB
TETFixedPositionAOI	4/18/2018 2:12 PM	E-Studio 3.0 Experim...	315 KB
TETFixedPositionAOI-1-1.gazedata	4/9/2018 10:18 AM	GAZEDATA File	1,120 KB
TETFixedPositionAOISaveGazeData	4/18/2018 12:09 PM	Text Document	6 KB
TETFixedPositionAOIUserScript	4/18/2018 12:09 PM	Text Document	7 KB
FixedPositionAOI.wndpos	4/18/2018 1:44 PM	WNDPOS File	0 KB
TETFixedPositionAOI.wndpos	4/18/2018 2:58 PM	WNDPOS File	0 KB

2) ***Right click*** the **TETFixedPositionAOI-1-1.gazedata file**. ***Select*** **Open With > Microsoft Office Excel**. If prompted, accept all defaults. ***Click*** **Finish**.

NOTE: *The file name varies depending on what subject number and session number you chose to run the experiment under.*

3) ***Verify*** that the **file** opens and looks like the image below.



A1 | Subject

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
1	Subject	Session	ID	RTTime	RTTimeRe	CursorX	CursorY	Hardware	HostTime:	SystemTir	SEQ	SEQH	SEQD	GazePoint	GazePoin
2	1	1	1	54322	54336	478	374	5.6E+08	1.03E+13	1.03E+13	0	0	1282	1	0.452668
3	1	1	2	54330	54342	476	361	5.6E+08	1.03E+13	1.03E+13	0	0	1283	1	0.454343
4	1	1	3	54339	54350	468	359	5.6E+08	1.03E+13	1.03E+13	0	0	1284	1	0.4578
5	1	1	4	54347	54359	485	358	5.6E+08	1.03E+13	1.03E+13	0	0	1285	1	0.46535
6	1	1	5	54355	54367	471	337	5.6E+08	1.03E+13	1.03E+13	0	0	1286	1	0.45634
7	1	1	6	54364	54376	466	330	5.6E+08	1.03E+13	1.03E+13	0	0	1287	1	0.45549
8	1	1	7	54372	54384	469	341	5.6E+08	1.03E+13	1.03E+13	0	0	1288	1	0.455013
9	1	1	8	54380	54392	479	355	5.6E+08	1.03E+13	1.03E+13	0	0	1289	1	0.463414
10	1	1	9	54389	54401	474	354	5.6E+08	1.03E+13	1.03E+13	0	0	1290	1	0.463414
11	1	1	10	54397	54409	477	357	5.6E+08	1.03E+13	1.03E+13	0	0	1291	1	0.462513
12	1	1	11	54405	54417	479	358	5.6E+08	1.03E+13	1.03E+13	0	0	1292	1	0.469683
13	1	1	12	54414	54426	478	365	5.6E+08	1.03E+13	1.03E+13	0	0	1293	1	0.46973
14	1	1	13	54422	54434	480	366	5.6E+08	1.03E+13	1.03E+13	0	0	1294	1	0.47385

Task 17: Introduction to .gazedata File

Understand the general output from the .gazedata file.

The .gazedata file contains information about eye movements from the Tobii Pro Eye Tracker and information about the stimulus presentation from E-Prime. The data consists of general information regarding the eye gaze data that is continuously collected as well as user-defined data. This data is compiled in one file to make analysis easier. The table below summarizes the general information included in the .gazedata file. User-defined data will be discussed later in this tutorial. Take some time to familiarize yourself with the basic output and the information that it provides. This way you can determine what you need to add to the data file in the way of user defined columns.

.gazedata File	
Column	***Definition***
Subject	The Subject number (from E-Prime Startup Info).
Session	The Session number (from E-Prime Startup Info).
ID	The counter of the current numbered .gazedata point acquired.
RTTime	The timestamp value based on E-Prime's clock. Returns the time stamp of when it currently is in the experiment.
RTTimeReceived	The E-Prime time when the application received the eye tracker packet.
CursorX	The X axis pixel location across all displays that corresponds to where the gaze is located on the screen and accounts for screen resolution (width).
CursorY	The Y axis pixel location across all displays that corresponds to where the gaze is located on the screen and accounts for screen resolution (height).
HardwareTimestamp	The complete device timestamp provided within the eye tracker sample (reported in microseconds).
HostTimestamp	The eye tracker system time when the application receives the gaze data (reported in microseconds).
SystemTimestamp	The complete system timestamp provided within the eye tracker sample (reported in microseconds).
SEQ	The sequence of the response within the InputMask.
SEQH	The sequence of the response provided by the InputHistoryManager.
SEQD	The sequence of the response provided by the device
GazePointValidityLeftEye	The validity of the left eye gaze point data.
GazePointPositionDisplayXLeftEye	The horizontal position (X) of the left eye gaze point on the active display (0, 0 is the upper left corner and 1, 1 is the lower right corner).
GazePointPositionDisplayYLeftEye	The vertical position (Y) of the left eye gaze point on the display (0, 0 is the upper left corner and 1, 1 is the lower right corner).
GazePointValidityRightEye	The validity of the right eye gaze point data.
GazePointPositionDisplayXRightEye	The horizontal position (X) of the right eye gaze point on the display (0, 0 is the upper left corner and 1, 1 is the lower right corner).
GazePointPositionDisplayYRightEye	The vertical position (Y) of the right eye gaze point on the display (0, 0 is the upper left corner and 1, 1 is the lower right corner).
GazePointPositionUserXLeftEye	The horizontal gaze point position (X) in the user coordinate system for the left eye (the x-axis points horizontally towards the user's right).
GazePointPositionUserYLeftEye	The vertical gaze point position (Y) in the user coordinate system for the left eye (the y-axis points vertically towards the ceiling).
GazePointPositionUserZLeftEye	The gaze point position from the user to the eye tracker (Z) in the user coordinate system for the left eye (the z-axis points towards the user).
GazePointPositionUserXRightEye	The horizontal gaze point position (X) in the user coordinate system for the right eye (the x-axis points horizontally towards the user's right).

.gazedata File continued	
Column	***Definition***
GazePointPositionUserYRightEye	The vertical gaze point position (Y) in the user coordinate system for the right eye (the y-axis points vertically towards the ceiling).
GazePointPositionUserZRightEye	The gaze point position from the user to the eye tracker (Z) in the user coordinate system for the right eye (the z-axis points towards the user).
GazeOriginValidityLeftEye	The validity of the left eye gaze origin data.
GazeOriginPositionUserXLeftEye	The horizontal gaze origin position (X) in the user coordinate system for the left eye (the x-axis points horizontally towards the user’s right).
GazeOriginPositionUserYLeftEye	The vertical gaze origin position (Y) in the user coordinate system for the left eye (the y-axis points vertically towards the ceiling).
GazeOriginPositionUserZLeftEye	The gaze origin position from the user to the eye tracker (Z) in the user coordinate system for the left eye (the z-axis points towards the user).
GazeOriginValidityRightEye	The validity of the right eye gaze origin data.
GazeOriginPositionUserXRightEye	The horizontal gaze origin position (X) in the user coordinate system for the right eye (the x-axis points horizontally towards the user’s right).
GazeOriginPositionUserYRightEye	The vertical gaze origin position (Y) in the user coordinate system for the right eye (the y-axis points vertically towards the ceiling).
GazeOriginPositionUserZRightEye	The gaze origin position from the user to the eye tracker (Z) in the user coordinate system for the right eye (the z-axis points towards the user).
GazeOriginPositionTrackBoxXLeftEye	The horizontal normalized gaze origin position (X) in the track box coordinate system for the left eye (the x-axis points horizontally towards the user’s left).
GazeOriginPositionTrackBoxYLeftEye	The vertical normalized gaze origin position (Y) in the track box coordinate system for the left eye (the y-axis points vertically towards the ground).
GazeOriginPositionTrackBoxZLeftEye	The normalized gaze origin position from the user to the eye tracker (Z) in the track box coordinate system for the left eye (the z-axis points towards the user).
GazeOriginPositionTrackBoxXRightEye	The horizontal normalized gaze origin position (X) in the track box coordinate system for the right eye (the x-axis points horizontally towards the user’s left).
GazeOriginPositionTrackBoxYRightEye	The vertical normalized gaze origin position (Y) in the track box coordinate system for the right eye (the y-axis points vertically towards ground).
GazeOriginPositionTrackBoxZRightEye	The normalized gaze origin position from the user to the eye tracker (Z) in the track box coordinate system for the right eye (the z-axis points towards the user).

For further information on the different coordinate systems used in gathering data for .gazedata files, please refer to:

Coordinate System in Tobii Eye Tracking

Task 18: Validity Ratings

Familiarize yourself with the validity ratings for the .gazedata output.

The rows corresponding to the columns listed on the previous page contain the values of the output data. In some cases, the values need no explanation because they are timestamps or millimeters. In other cases, the data values are single digit numbers. When this is the case it is necessary to know the meaning assigned to the numeric value. The table below defines the meaning of the number.

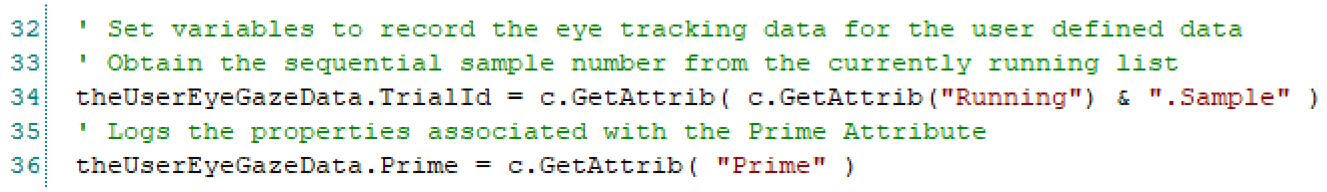
Validity Ratings	
0	0 indicates that valid data was not recorded on this sample and eye. The corresponding gaze data has values of -1 that indicate a valid observation was not made.
1	1 indicates that valid data was observed for this sample and eye. The corresponding gaze data has appropriate values (e.g., 0.222436338) and -1 is not reported.

Task 19: Examine the Structure and Content of the File

View the User-defined variables in the .gazedata file

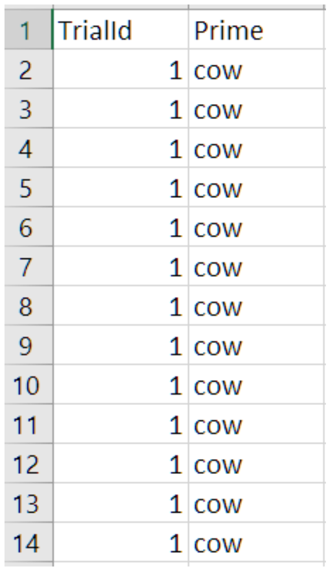
Data is collected at the rate of your Tobii Pro Eye Tracker Device. When a sample is timestamped, a row is written to the .gazedata file populating the columns. The next steps in the tutorial explains the user defined columns in the .gazedata file and highlights the script used to create them. If you do not have the .gazedata file open, see *Task 16: Open the Eye .gazedata File*, for instructions on how to navigate to and open the .gazedata file.

1) **TrialID Column**: Script used to create assign variable. (location: SaveGazeData InLine):
theUserEyeGazeData.TrialId = c.GetAttrib(c.GetAttrib ("Running") & ".Sample")
Purpose: Unique identifier used to keep track of the trial number.



```
' Set variables to record the eye tracking data for the user defined data
' Obtain the sequential sample number from the currently running list
theUserEyeGazeData.TrialId = c.GetAttrib( c.GetAttrib("Running") & ".Sample" )
' Logs the properties associated with the Prime Attribute
theUserEyeGazeData.Prime = c.GetAttrib( "Prime" )
```

2) The **TrialID column *increases*** for each trial. The **trial** is labeled **1**.



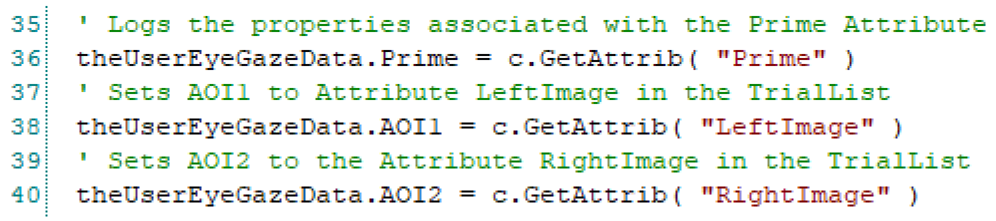
1	TrialId	Prime
2	1	cow
3	1	cow
4	1	cow
5	1	cow
6	1	cow
7	1	cow
8	1	cow
9	1	cow
10	1	cow
11	1	cow
12	1	cow
13	1	cow
14	1	cow

3) **Prime Column**: Script used to assign variable. (location: SaveGazeData InLine):
theUserEyeGazeData.Prime = c.GetAttrib("Prime")
Purpose: Logs properties associated with Prime.

4) The **Prime column *shows*** what sound file was played when the Prime Object was on screen. The **Prime** was **cow**.

5) **AOI1 Column**: Script used to assign variable. (location: SaveGazeData InLine):

theUserEyeGazeData.AOI1 = c.GetAttrib("LeftImage")
Purpose: Sets the LeftImage Attribute (TrialList) as AOI1.

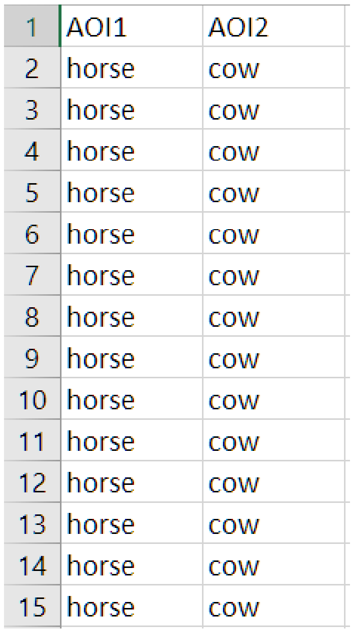


```
' Logs the properties associated with the Prime Attribute
theUserEyeGazeData.Prime = c.GetAttrib( "Prime" )
' Sets AOI1 to Attribute LeftImage in the TrialList
theUserEyeGazeData.AOI1 = c.GetAttrib( "LeftImage" )
' Sets AOI2 to the Attribute RightImage in the TrialList
theUserEyeGazeData.AOI2 = c.GetAttrib( "RightImage" )
```

6) The **AOI1** is ***assigned*** the **LeftImage attribute** from the TrialList. The **AOI1 column *shows*** what image was displayed on the left side of the screen on a trial by trial basis. In this example, the LeftImage is a **horse**.

7) **AOI2 Column**: Script used to assign variable. (location: SaveGazeData InLine):
theUserEyeGazeData.AOI2 = c.GetAttrib("RightImage")
Purpose: Sets the RightImage Attribute (TrialList) as AOI2.

8) The **AOI2** is ***assigned*** the **RightImage attribute** from the TrialList. The **AOI2 column *shows*** what image was displayed on the right side of the screen on a trial by trial basis. In this example, the RightImage is a **cow**.



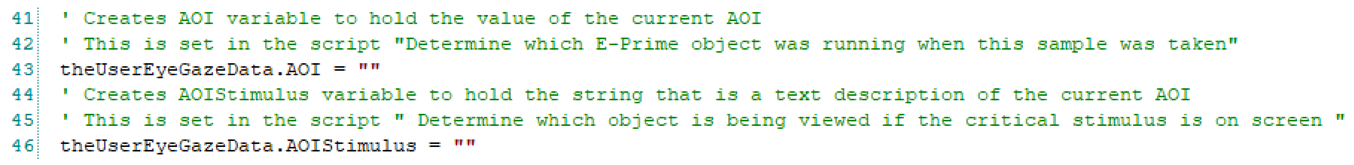
1	AOI1	AOI2
2	horse	cow
3	horse	cow
4	horse	cow
5	horse	cow
6	horse	cow
7	horse	cow
8	horse	cow
9	horse	cow
10	horse	cow
11	horse	cow
12	horse	cow
13	horse	cow
14	horse	cow
15	horse	cow

Task 20: Understand the AOI and AOIStimulus Variables

Learn how the AOI and AOIStimulus variables are created and what their purpose is.

The AOI and AOIStimulus variables are special variables that hold information about what was on screen at a certain point in time. This information is determined in the SaveGazeData InLine within the experiment. The first image below shows the variables that indicate which AOI the participant is currently viewing, and the name corresponding to that AOI. Then the script goes on to determine which case is true (discussed on next page).

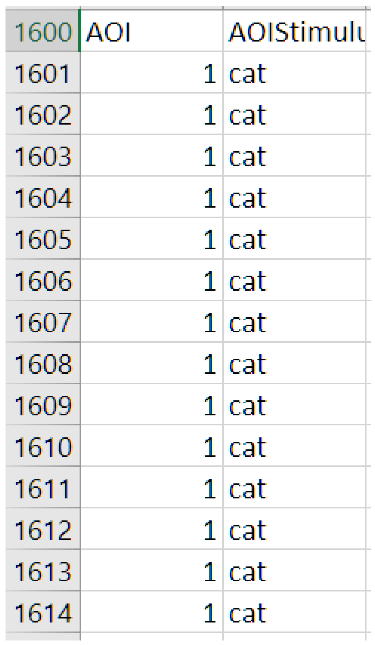
1) The variables that indicate which AOI the participant is currently viewing, and the name corresponding to that AOI.



```
' Creates AOI variable to hold the value of the current AOI
' This is set in the script "Determine which E-Prime object was running when this sample was taken"
theUserEyeGazeData.AOI = ""
' Creates AOIStimulus variable to hold the string that is a text description of the current AOI
' This is set in the script " Determine which object is being viewed if the critical stimulus is on screen "
theUserEyeGazeData.AOIStimulus = ""
```

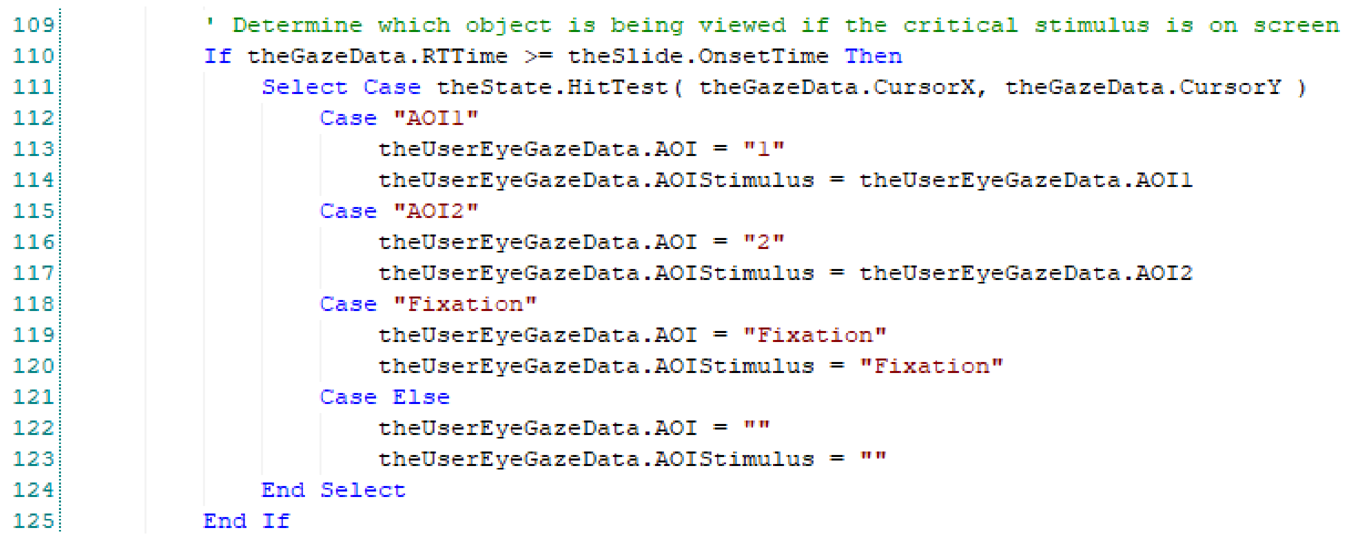
2) **AOI Column**: Script used to create variable. (location: SaveGazeData InLine):
theUserEyeGazeData.AOI = ""
Purpose: Creates AOI variable to hold the value of the AOI the participant is currently viewing.

3) **AOI Stimulus Column**: Script used to create variable (location: SaveGazeData InLine):
theUserEyeGazeData.AOIStimulus = ""
Purpose: Creates AOIStimulus variable to hold the string that is a text description of the current AOI.



1600	AOI	AOIStimulu
1601	1	cat
1602	1	cat
1603	1	cat
1604	1	cat
1605	1	cat
1606	1	cat
1607	1	cat
1608	1	cat
1609	1	cat
1610	1	cat
1611	1	cat
1612	1	cat
1613	1	cat
1614	1	cat

4) Case “AOI1” is ***executed*** when the **hit test coordinates *correspond*** with the location of the **Stimulus Slide Sub-Object** named AOI1.



```
' Determine which object is being viewed if the critical stimulus is on screen
If theGazeData.RTTime >= theSlide.OnsetTime Then
    Select Case theState.HitTest( theGazeData.CursorX, theGazeData.CursorY )
        Case "AOI1"
            theUserEyeGazeData.AOI = "1"
            theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI1
        Case "AOI2"
            theUserEyeGazeData.AOI = "2"
            theUserEyeGazeData.AOIStimulus = theUserEyeGazeData.AOI2
        Case "Fixation"
            theUserEyeGazeData.AOI = "Fixation"
            theUserEyeGazeData.AOIStimulus = "Fixation"
        Case Else
            theUserEyeGazeData.AOI = ""
            theUserEyeGazeData.AOIStimulus = ""
    End Select
End If
```

5) Case “AOI2” is ***executed*** when the **hit test coordinates *correspond*** with the location of the **Stimulus Slide Sub-Object** named AOI2.

6) Case “Fixation” is ***executed*** when the **hit test coordinates *correspond*** with the location of the **Stimulus Slide Sub-Object** named Fixation.

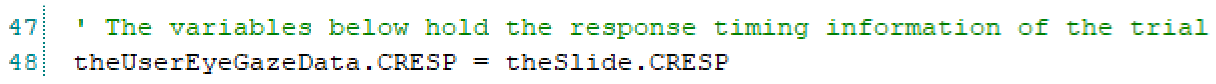
7) Case Else is ***executed*** when the **hit test coordinates *correspond*** with a location within the **Stimulus Slide** that does not have any **Sub-Object** associated with it.

Task 21: Understand the Remaining Variables in the .gazedata File

Learn how the remaining variables in the .gazedata file are created and what their purpose is.

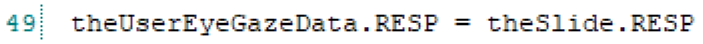
The variables CRESP, RESP, ACC, and RT tell us information about accuracy, the response made, and reaction time. This is important information to have when doing the analysis.

1) **CRESP Column**: Script used to create variable. (location: SaveGazeData InLine):
   theUserEyeGazeData.CRESP = Stimulus.CRESP
   Purpose: Records correct response to Stimulus Object for the trial.



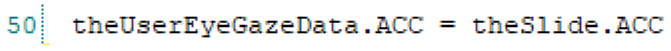
```
' The variables below hold the response timing information of the trial
theUserEyeGazeData.CRESP = theSlide.CRESP
```

2) **RESP Column**: Script used to create variable. (location: SaveGazeData InLine):
   theUserEyeGazeData.RESP = Stimulus.RESP
   Purpose: Records correct response to Stimulus Object for the trial.



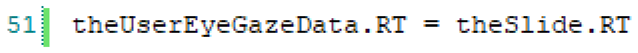
```
theUserEyeGazeData.RESP = theSlide.RESP
```

3) **ACC Column**: Script used to create variable. (location: SaveGazeData InLine):
   theUserEyeGazeData.ACC = Stimulus.ACC
   Purpose: Scores accuracy of response to Stimulus Object for the trial.



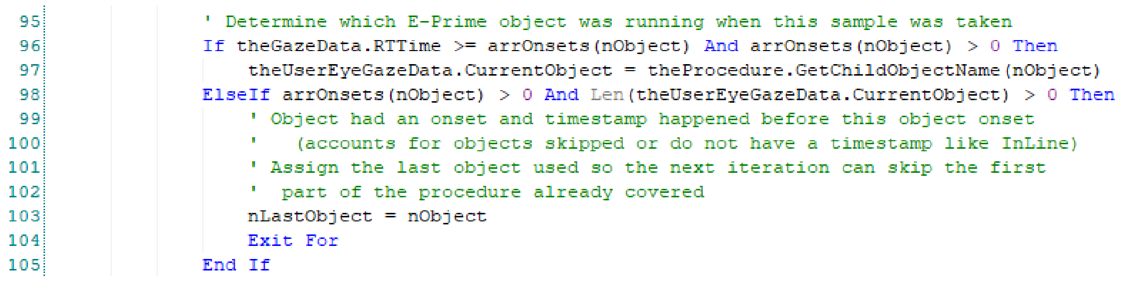
```
theUserEyeGazeData.ACC = theSlide.ACC
```

4) **RT Column**: Script used to create variable. (location: SaveGazeData InLine):
   theUserEyeGazeData.RT = Stimulus.RT
   Purpose: Records time of response to Stimulus Object for the trial.



```
theUserEyeGazeData.RT = theSlide.RT
```

5) **Currentobject Column**: Script used to create variable. (location: SaveGazeData InLine):
   theUserEyeGazeData.Currentobject = “”
   Purpose: Records the current object for the gaze point.



```
' Determine which E-Prime object was running when this sample was taken
If theGazeData.RTTime >= arrOnsets(nObject) And arrOnsets(nObject) > 0 Then
    theUserEyeGazeData.CurrentObject = theProcedure.GetChildObjectName(nObject)
ElseIf arrOnsets(nObject) > 0 And Len(theUserEyeGazeData.CurrentObject) > 0 Then
    ' Object had an onset and timestamp happened before this object onset
    '   (accounts for objects skipped or do not have a timestamp like InLine)
    ' Assign the last object used so the next iteration can skip the first
    '  part of the procedure already covered
    nLastObject = nObject
    Exit For
End If
```

NOTE: *The script in the image above determines the value for this variable via the .OnsetTime Property.*