

E-Studio

*Graphical Design
Environment*

E-Basic

*Full Scripting
Language*

E-Run

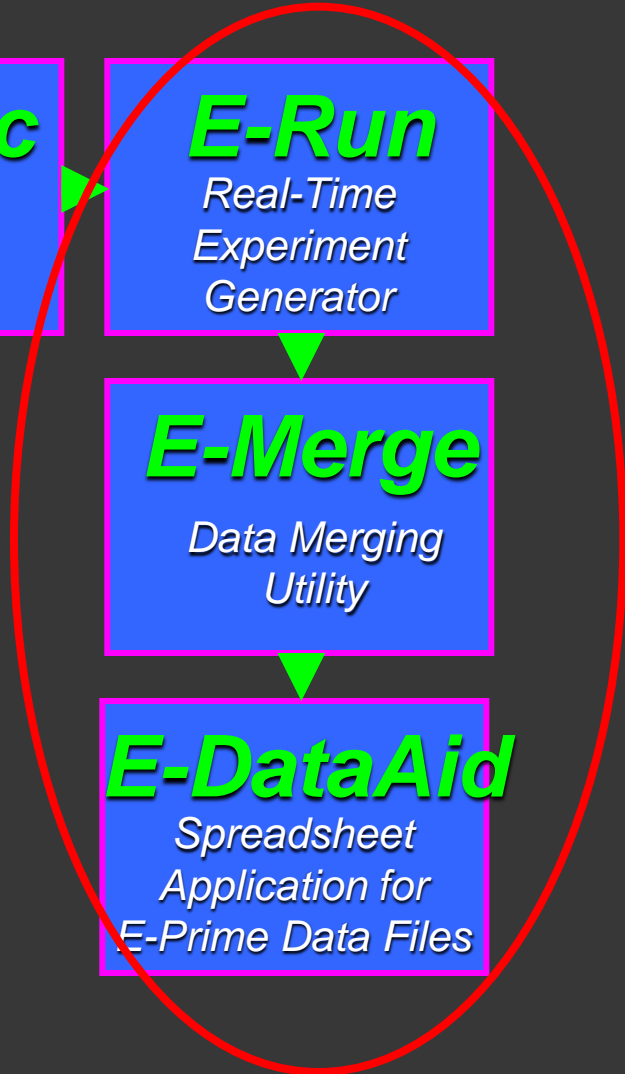
*Real-Time
Experiment
Generator*

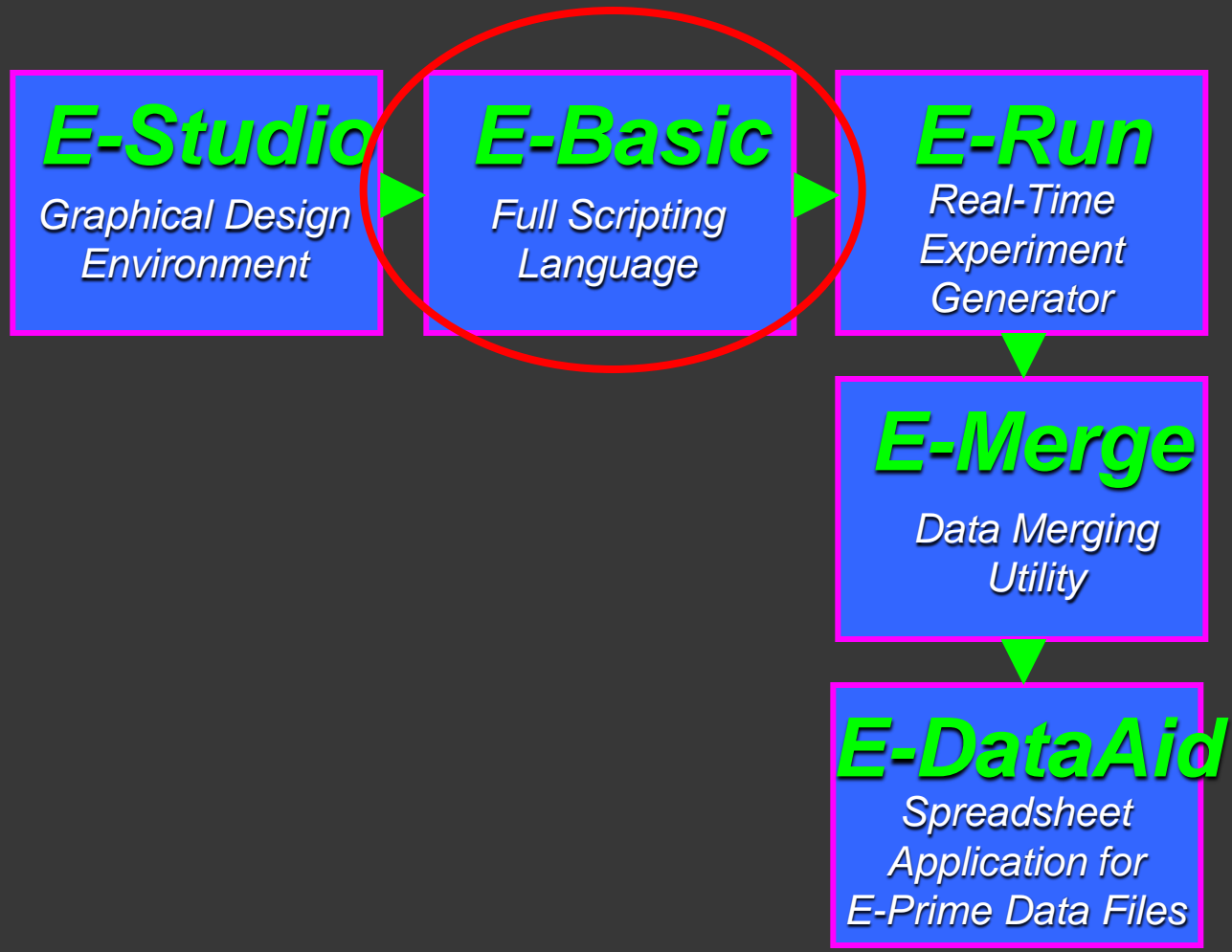
E-Merge

*Data Merging
Utility*

E-DataAid

*Spreadsheet
Application for
E-Prime Data Files*





Overzicht

1. E-Basic

1. Visual Basic
2. Oefening 4
3. Andere voorbeelden
4. Opmerking

2. Timing issues

1. Timing modes
2. Refresh rates
3. Delays
4. Tips

1. E-Basic

Waarom?

- ◎ Wat niet geïmplementeerd is in E-Studio kan je toch programmeren
- ◎ En ja, er zijn veel dingen NIET geïmplementeerd...
- ◎ Zelfs eenvoudige...

InLine

Geen probleem:

Je kan script toevoegen d.m.v. een InlineObject

The screenshot displays a software development environment with three main panels:

- Toolbox:** A vertical list of icons on the left. The 'InLine' icon, which depicts a document with a script, is circled in red.
- Structure:** A tree view in the top-left showing a project hierarchy: 'Experiment (Inline.es)' contains 'SessionProc', which includes 'InLine1', 'DesignList', and 'TrialProc'. 'TrialProc' further contains 'Infixation', 'arrow', and 'stim'. An 'End' node is also present, along with 'Unreferenced E-Objects'.
- Diagram:** A horizontal flow diagram on the right. It features a green dot on the left, followed by three icons: a document labeled 'InLine1', a grid labeled 'DesignList', and a monitor labeled 'End'. A red dashed arrow points from the circled 'InLine' icon in the toolbox to the 'InLine1' icon in the diagram.

Below the diagram, the text *plaats inline ~ doel* is written in red, indicating the placement of the inline object relative to the goal.

1.1. Visual Basic

- ◎ 'leesbare' code
- ◎ Elk object heeft lijst van eigenschappen & methodes
 - Vb. *InstructiesExp.text* (object.property)
 - Vb. *InstructiesExp.run* (object.method)

```
InstructiesExp.Name = "InstructiesExp"  
InstructiesExp.Tag = ""  
  
Set InstructiesExpEchoClients = New EchoClientCollection  
  
InitTextDisplayDefaults InstructiesExp  
  
InstructiesExp.Text = "Welkom in dit experiment.\n\nIn dit experiment krijg je telkens een fix  
"Oud taak om zo snel mogelijk te reageren op de KLEUR van de cirkel.\n\nDruk op de  
  
InstructiesExp.Duration = CLng("-1")  
InstructiesExp.TimingMode = ebTimingModeEvent  
InstructiesExp.PreRelease = Val("0")  
  
InstructiesExp.OnsetSync = 1  
InstructiesExp.OffsetSync = 0  
  
Set BlockList = New List  
BlockList.Name = "BlockList"  
BlockList.Tag = ""
```

1.1. Visual Basic

◎ Properties

- Retrieve property: `c.GetAttrib` ("name"),
vb. `c.GetAttrib` ("Stimulus.ACC")
- Modify property: `c.SetAttrib` "name", value,
vb. `c.SetAttrib` "aantalfout", *afout*

```
Fixatie.Run

Stimulus.Filename = c.GetAttrib("stimulus")
Stimulus.Load
Stimulus.AlignHorizontal = c.GetAttrib("pos")

Stimulus.InputMasks.Reset

StimulusEchoClients.RemoveAll
Stimulus.InputMasks.Add Keyboard.CreateInputMask("jff", c.GetAttrib("correctresp"),

Stimulus.Run
c.SetAttrib "Stimulus.OnsetDelay", Stimulus.OnsetDelay
c.SetAttrib "Stimulus.OnsetTime", Stimulus.OnsetTime
c.SetAttrib "Stimulus.DurationError", Stimulus.DurationError
c.SetAttrib "Stimulus.RTTime", Stimulus.RTTime
c.SetAttrib "Stimulus.ACC", Stimulus.ACC
c.SetAttrib "Stimulus.RT", Stimulus.RT
c.SetAttrib "Stimulus.RESP", Stimulus.RESP
```

1.1. Visual Basic

◎ Methods

- Commando's, vb. run
- Functies, return van waarde, vb. mean

```
Case "Incorrect"  
  
    Set Feedback_SlideText = CSlideText(Feedback.States.Item("Incorrect").Objects(1))  
    Feedback_SlideText.Text = "" & _  
    Format$(Feedback.ACCStats.Mean / Feedback.ACCDivisor), Feedback.ACCFormat) & _  
    " Average Percent Correct"  
    Set Feedback_SlideText = Nothing  
  
    Set Feedback_SlideText = CSlideText(Feedback.States.Item("Incorrect").Objects(2))  
    Feedback_SlideText.Text = "" & _  
    Format$(Stimulus.RT / Feedback.RTDivisor), Feedback.RTFormat) & _  
    " Seconds Response Time"  
    Set Feedback_SlideText = Nothing  
  
    Set Feedback_SlideText = CSlideText(Feedback.States.Item("Incorrect").Objects(3))  
    Set Feedback_SlideText = Nothing  
Case "NoResponse"  
  
    Set Feedback_SlideText = CSlideText(Feedback.States.Item("NoResponse").Objects(1))  
    Set Feedback_SlideText = Nothing  
Case "Pending"  
  
End Select
```

Feedback.Run

1.1. Visual Basic

- Comments: '
- Conditional statements
vb. If... Then
End If

```
If Stimulus.ACC = 1 Then
    'Set the ActiveState to Correct
    Feedback.ActiveState = "Correct"

    'Add an observation to the accuracy stats
    Feedback.AccStats.AddObservation Stimulus.ACC

    'Add an observation to the response time stats
    ' unless the user did not respond and the author
    ' does not want us to add the no response RT
    If Len(Stimulus.RESP) > 0 Then
        Feedback.RTStats.AddObservation Stimulus.RT
        Feedback.CorrectRTStats.AddObservation Stimulus.RT
    End If
Else
    'Is it incorrect or no response?
    If Len(Stimulus.RESP) > 0 Then
        'Set the ActiveState to Incorrect
        Feedback.ActiveState = "Incorrect"

        'Set the accuracy stats
```

1.1. Visual Basic

- MsgBox “ ”

The screenshot shows the Visual Basic IDE with the following components:

- Structure Pane:** Shows the project hierarchy: Experiment (msgbox.es) > SessionProc > InLine1. There are also Unreferenced E-Objects listed.
- Properties Pane:** Shows the properties for the selected 'InLine1' component. The 'Code' property is set to `MsgBox "Take a break, click OK to continue"`.
- SessionProc Component:** A diagram showing a green dot connected to a red dot by a line, with a document icon and the label 'InLine1' in the center.
- InLine1 Code Editor:** A window showing the code for the 'InLine1' component: `MsgBox "Take a break, click OK to continue"`.
- E-Run Window:** A separate window titled 'E-Run' that displays the output of the code: a message box with the text 'Take a break, click OK to continue' and an 'OK' button.

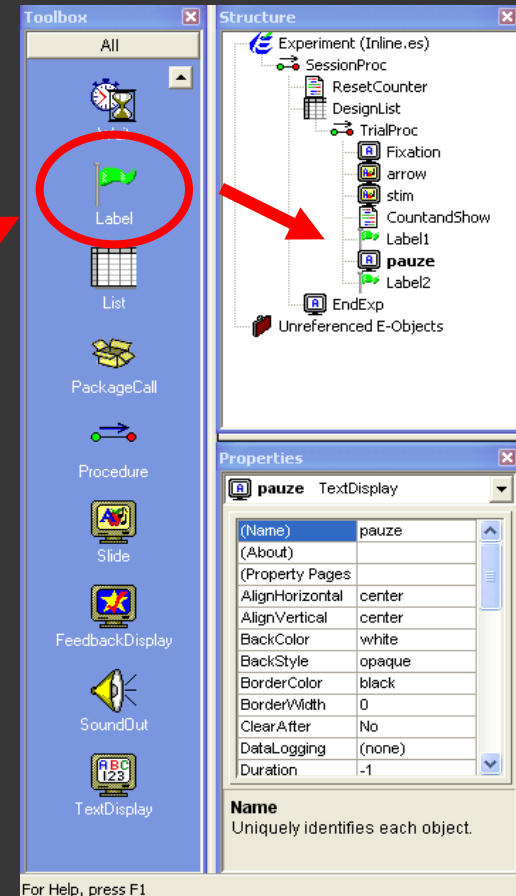
A red arrow points from the code editor window to the 'E-Run' window, indicating the execution of the code.

1.1. Visual Basic

- Loops, vb. do...loop until

- GoTo **Label**

Als je achteruit of
vooruit in je experiment
wilt springen



For Help, press F1

1.2. Oefening 4 (teller)

Doel: 40 trials, pauze na elke 10 trials met teller

Werk verder op Oefening3.es

1. Verhoog het aantal trials tot 40 en selection = random

The screenshot shows the DesignList software interface. The main window displays a summary of 56 samples (7 cycles x 8 samples/cycle) and a table of trials. The Properties dialog box is open, showing the Selection tab. The 'Exit List' section is highlighted with a red circle, and the 'After 5 cycles (40 samples)' option is selected. A red arrow points to the number '5' in the 'After 5 cycles (40 samples)' option.

| ID | Weight | Nested | Procedure | Condition | Stimulus | StateName | Probe |
|----|--------|--------|-----------|-------------------|----------|-----------|----------------|
| 1 | 1 | | TrialProc | poscomp_arrcomp | left | left | arrowleft.bmp |
| 2 | 1 | | TrialProc | posincomp_arrcomp | left | right | arrowleft.bmp |
| 3 | | | | | | | arrowleft.bmp |
| 4 | | | | | | | arrowleft.bmp |
| 5 | | | | | | | arrowright.bmp |
| 6 | | | | | | | arrowright.bmp |
| 7 | | | | | | | arrowright.bmp |
| 8 | | | | | | | arrowright.bmp |

Variabele 'teller'

- ◎ **Declareren** = naam en type bepalen
 - Lokaal
 - Binnen een procedure
 - > Enkel in deze procedure toegankelijk
 - Globaal
 - In User tab van script
 - > Over ganse experiment toegankelijk
- ◎ **Initialiseren** = beginwaarde toekennen

2. Teller declareren

The screenshot shows the E-Studio interface. The 'View' menu is open, with 'Script' selected. A red arrow points from the 'View' menu item to the 'Script' option. Another red arrow points from the 'Script' option to the code editor. The code editor shows the declaration: `Dim teller as Long`. A red circle highlights the 'User' button in the bottom right corner of the code editor. A text box explains the components of the declaration: **Dim**: declareren van nieuwe variabele, **teller**: naam van de nieuwe variabele, **as Long**: geheel getal.

View E-Run Tools Window

- Attributes Alt+1
- Browser Alt+2
- Output Alt+3
- Properties Alt+4
- Script** Alt+5
- Structure Alt+6
- Toolbox Alt+7

Full Screen

Toolbar

Status Bar

ResetCounter InLine

| | |
|------------------|--------------|
| (Name) | ResetCounter |
| (About) | |
| (Property Pages) | |
| Code | teller = 0 |
| Notes | |
| Tag | |

`Dim teller as Long`

User

Ln 1, Col 19

<http://www.pstnet.com>

(Als je script niet te zien krijgt, moet je 't eerst nog eens genereren)

3. Teller initialiseren

The screenshot displays the E-Studio environment for 'Exercise4Slide.es'. The **Toolbox** on the left contains various objects, with **ResetCounter** circled in red. A red arrow points from this icon to the **ResetCounter** object in the **SessionProc** window. The **Structure** window shows the hierarchy: Experiment (Exercise4Slide.es) > SessionProc > ResetCounter. The **Properties** window for the selected **ResetCounter** object shows the **Code** property set to `teller = 0`. The **Script** window shows the declaration `Dim teller as Long`. The status bar at the bottom right indicates 'Ln 1, Col 19'.

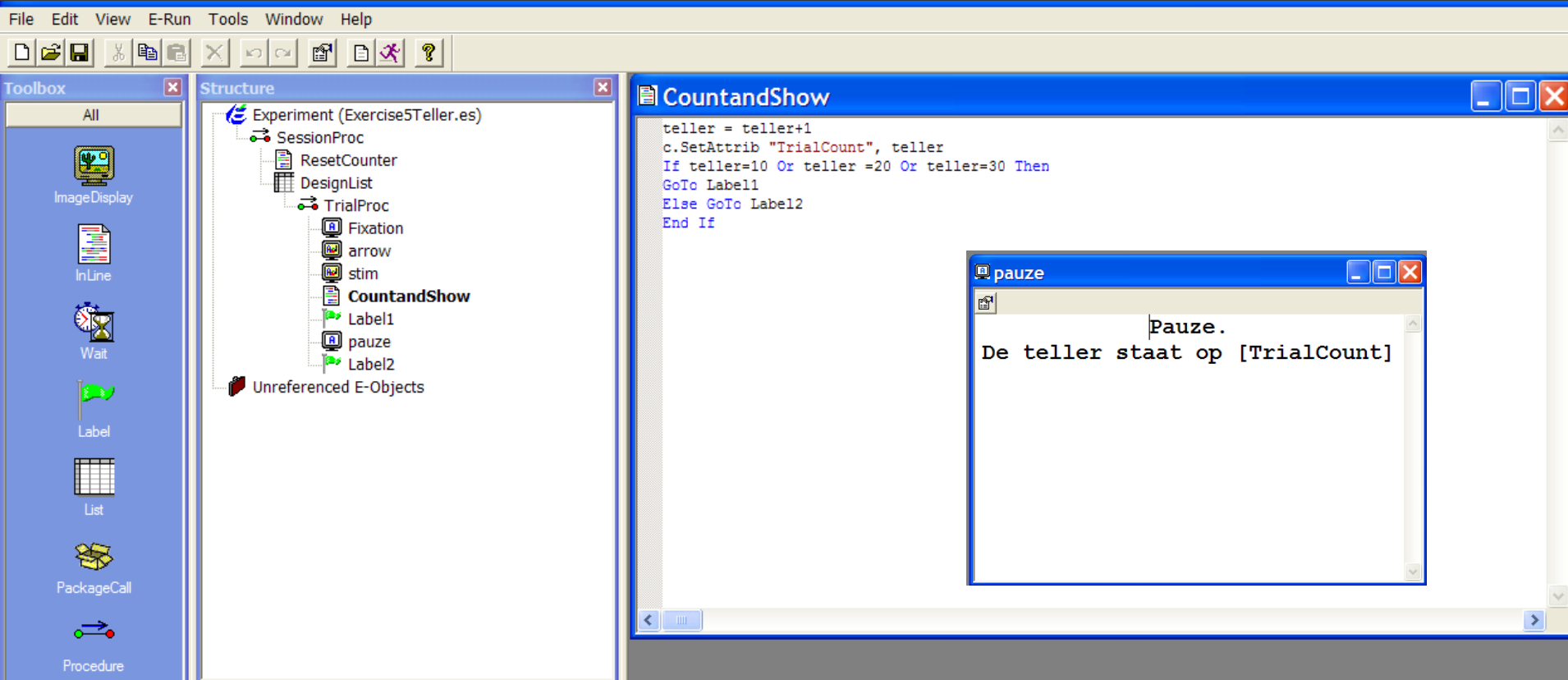
| (Name) | ResetCounter |
|------------------|--------------|
| (About) | |
| (Property Pages) | |
| Code | teller = 0 |
| Notes | |
| Tag | |

```
Dim teller as Long
```

Ln 1, Col 19

<http://www.pstnet.com>

4. InLine “Count and Show” en Pauze



The screenshot displays a software development environment with three main windows:

- Toolbox:** A vertical panel on the left containing various object icons such as ImageDisplay, InLine, Wait, Label, List, PackageCall, and Procedure.
- Structure:** A tree view showing the hierarchy of objects. The 'CountandShow' object is highlighted under the 'TrialProc' object.
- CountandShow:** A code editor window containing the following code:

```
teller = teller+1
c.SetAttrib "TrialCount", teller
If teller=10 Or teller =20 Or teller=30 Then
GoTo Label1
Else GoTo Label2
End If
```
- pauze:** A text display window containing the text:

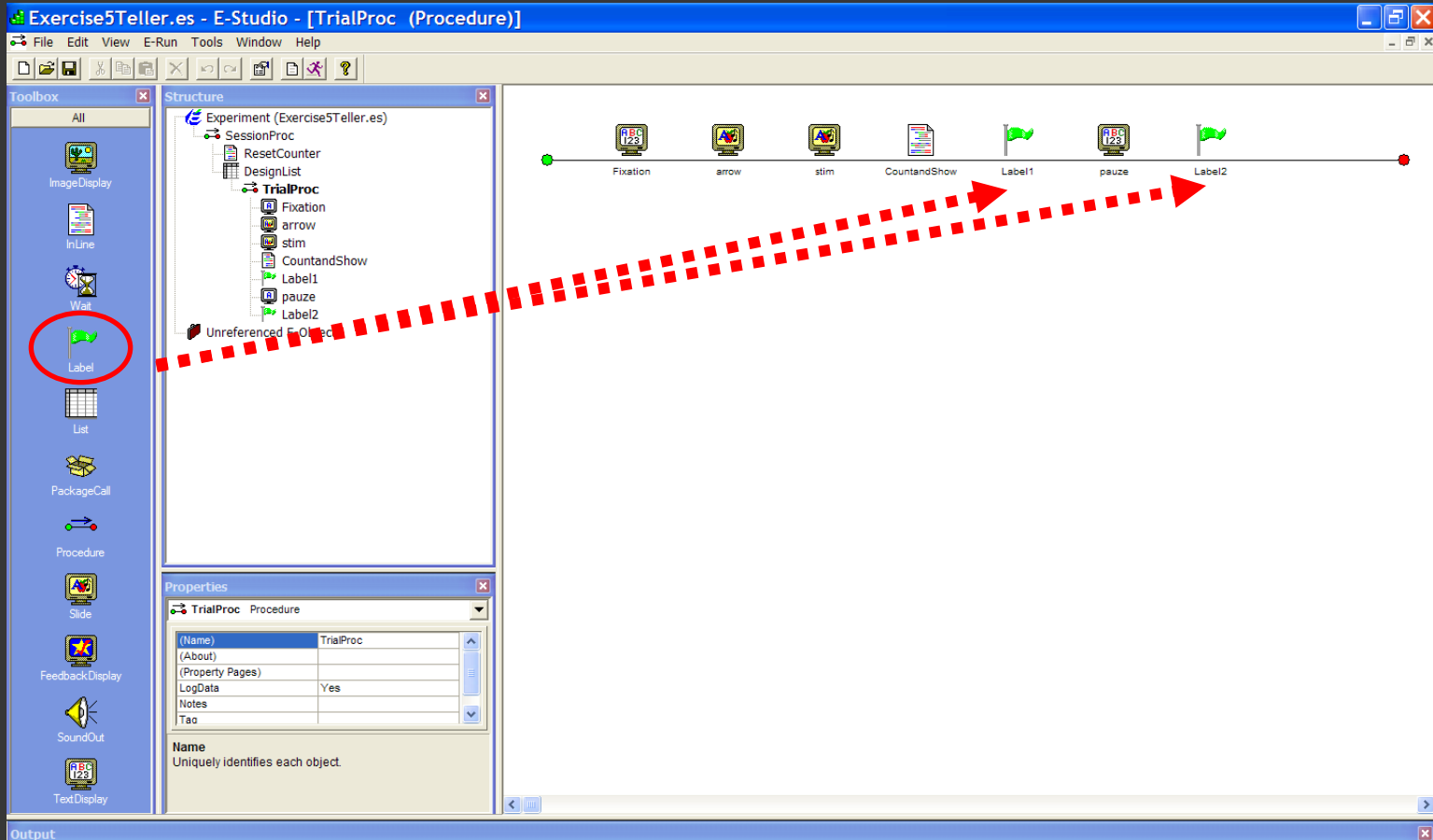
```
Pauze.
De teller staat op [TrialCount]
```

Voeg een InLine Object toe, noem het 'CountandShow'

Voeg een TextDisplay Object toe, noem het 'Pauze'

Schrijf er de juiste tekst in...

5. Labels invoegen



Voeg twee labels in
Noem ze Label1 en Label2

Ziet alles er zo uit? Goed zo! Save ... Generate ... Run!

The screenshot displays the E-Studio interface with several windows:

- CountandShow**: A script window containing the following code:

```
teller = teller + 1  
c.SetAttrib "TrialCount", teller  
If teller = 10 Or teller = 20 Or teller = 30 Then  
GoTo Label1  
Else GoTo Label2  
End If
```

The line `c.SetAttrib "TrialCount", teller` is highlighted with a red box.
- pauze**: A text display window showing the text: `Pauze.
De teller staat op [TrialCount]`. A red arrow points from this window to the Properties dialog.
- Properties: pauze**: A dialog box for configuring the 'pauze' object. The 'Input Masks' section is expanded, showing 'Keyboard' selected with a checkmark. The 'Response Options' are set to 'Keyboard', and the 'Allowable' is set to '{SPACE}'. Other settings include 'Duration: (infinite)', 'Timing Mode: Event', and 'End Action: Terminate'.
- Structure**: A tree view showing the trial procedure hierarchy. The 'pauze' object is circled in red.
- Properties**: A window showing the properties of the selected 'pauze' object, such as '(Name) pauze', 'AlignHorizontal center', and 'Duration -1'.
- Script**: A window containing the code `Dim teller as long`.

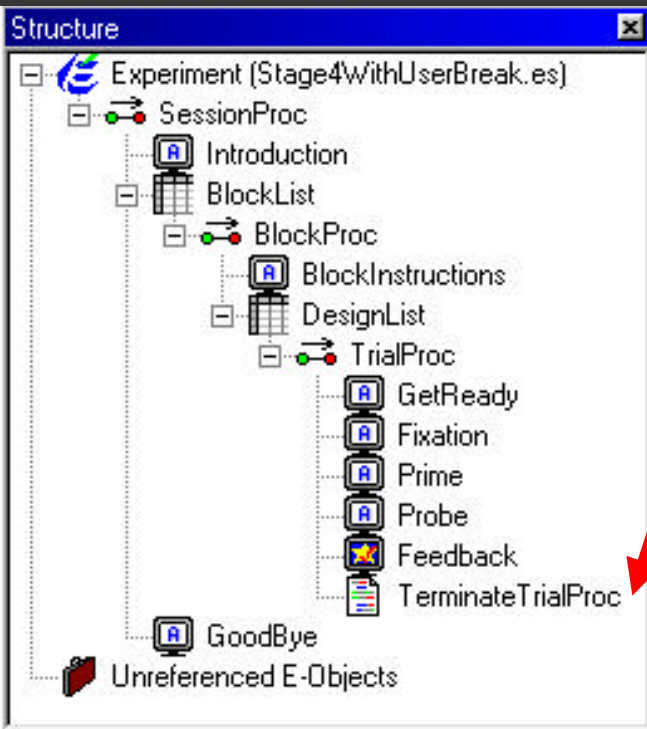
Red annotations highlight the `c.SetAttrib` line in the script and the 'pauze' object in the structure tree, with a red arrow pointing to the Properties dialog.

TrialCount moet opgehaald worden in pauze object, dus als attribuut definiëren

1.3. Andere voorbeelden

- *Vervroegd stoppen van experiment*
- Methode 1
 - Ctrl + Alt + Shift
 - .txt file → E-Recovery → .edat file
- Methode 2
 - Ctrl + Shift
 - .edat file

1.3. Andere voorbeelden



```
TerminateTrialProc  
  
' Terminate block if user break  
If (GetUserBreakState <> 0) Then      'Check for user break  
    DesignList.Terminate  
End If  
  
SetUserBreakState 0                  'Reset user break to end
```

- Als je Ctrl + Shift indrukt
- > GetUserBreakState wordt 1
 - > DesignList stopt
 - > GoodBye, Einde experiment

1.3. Andere voorbeelden

- *Bepaalde trials herhalen*

Op basis van de antwoorden van de pp vul je een lijst aan.
Enkel de stimuli in déze lijst worden op het einde herhaald.

- fout beantwoorde trials
- invalide trials (bijvoorbeeld als voicekey niet reageerde)

The screenshot displays a software interface with a 'Structure' window on the left and two 'RerunList' configuration windows on the right. The 'Structure' window shows a hierarchical tree of trial components. Three red arrows point from the 'RerunList' windows to specific components in the structure: 'InitErrorCount', 'RerunList', and 'RerunVal'.

The top 'RerunList' window shows a summary and a table:

Summary
1 Sample (1 cycle x 1 sample/cycle)
1 Cycle equals 1 sample
Random Selection

| ID | Weight | Nested | Procedure | CorrectAn... | left | right |
|----|--------|--------|-----------|--------------|------|-------|
| 1 | 1 | | RerunProc | ? | ? | ? |

The bottom 'RerunList' window shows an identical summary and table.

The 'InitErrorCount' window contains the following text:

```
'ErrorCount is used to keep track of the number of trials  
'on which the response is incorrect. ValidCount is used to  
'keep track of the invalid trials. The total is then  
'used to assign the number of samples to be run to repeat  
'error and invalid trials.  
g_nErrorCount = 0  
g_nValidCount = 0
```

The image shows a software development environment with three overlapping code windows. On the left is a tree view of a project structure. The top window, titled "TrackErrors", contains the following code:

```
'On incorrect trials, write the current trial info to the
'RerunList object, which is run after TrialList.
If resp.ACC = 0 Then
    g_nErrorCount = g_nErrorCount + 1
    If g_nErrorCount > 1 Then
        RerunList.AddLevel g_nErrorCount
    End If
RerunList.SetWeight g_nErrorCount, 1
RerunList.SetProc g_nErrorCount, "RerunProc"
RerunList.SetAttrib g_nErrorCount, "left", c.GetAttrib("left")
RerunList.SetAttrib g_nErrorCount, "right", c.GetAttrib("right")
RerunList.SetAttrib g_nErrorCount, "CorrectAnswer", c.GetAttrib("CorrectAnswer")
End if
```

The middle window, titled "TrackValid", contains the following code:

```
'On invalid trials, write the current trial info to the
'RerunVal object, which is run after RerunList.
If ValidityVK.RESP = 0 Then
    g_nValidCount = g_nValidCount + 1
    If g_nValidCount > 1 Then
        RerunVal.AddLevel g_nValidCount
    End If
RerunVal.SetWeight g_nValidCount, 1
RerunVal.SetProc g_nValidCount, "RerunProc"
RerunVal.SetAttrib g_nValidCount, "left", c.GetAttrib("left")
RerunVal.SetAttrib g_nValidCount, "right", c.GetAttrib("right")
RerunVal.SetAttrib g_nValidCount, "CorrectAnswer", c.GetAttrib("CorrectAnswer")
End if
```

The bottom window, titled "SetRerunList", contains the following code:

```
'If errors occur, run RerunList
'Set number of samples from ErrorCount
If g_nErrorCount > 0 Then
    Set RerunList.TerminateCondition = Cycles(1)
    Set RerunList.ResetCondition = Samples(g_nErrorCount)
    RerunList.Reset
    If g_nValidCount > 0 Then
        Set RerunVal.TerminateCondition = Cycles(1)
        Set RerunVal.ResetCondition = Samples(g_nValidCount)
        RerunVal.Reset
    Else
        End If
Else
    'No error trials to run !!!
    GoTo EndOfBlock
End If
```

Red arrows indicate the following connections:

- From the "TrackErrors" window to the "TrackValid" window.
- From the "TrackValid" window to the "SetRerunList" window.
- From the "SetRerunList" window to the "TrackErrors" window.

The "SetRerunList" window has a red box around the "No error trials to run !!!" message.

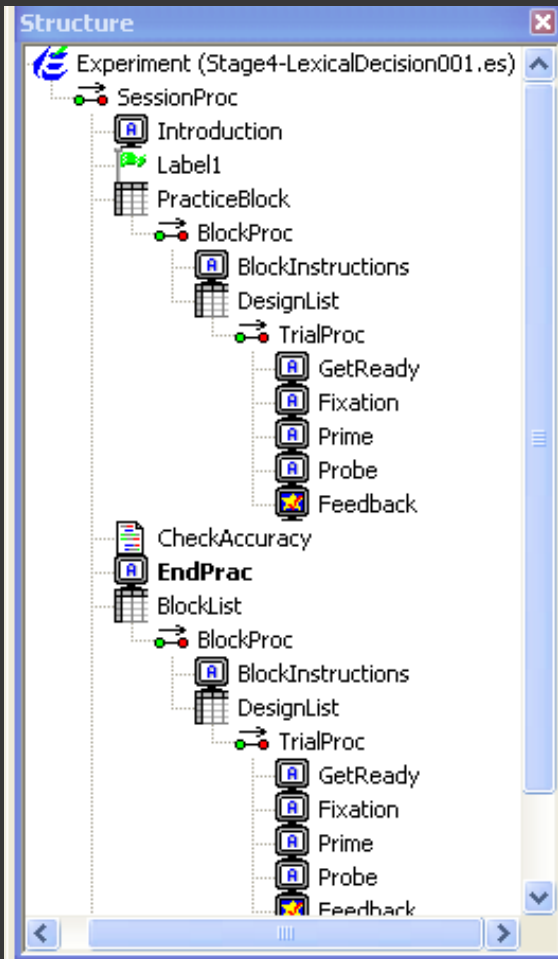
1.3. Andere voorbeelden

- *Oefenblok herhalen als pp het niet goed genoeg doet*

The screenshot displays the software interface for designing an experiment. On the left, the 'Structure' pane shows a hierarchical tree of components: SessionProc, Introduction, PracticeBlock, BlockProc, BlockInstructions, DesignList, TrialProc, GetReady, Fixation, Prime, Probe, Feedback, CheckAccuracy, BlockList, BlockProc, BlockInstructions, DesignList, and TrialProc. The 'CheckAccuracy' component is highlighted. Below the structure pane is the 'Properties' pane for the selected 'CheckAccuracy' object, showing its name and a description: 'Uniquely identifies each object.'

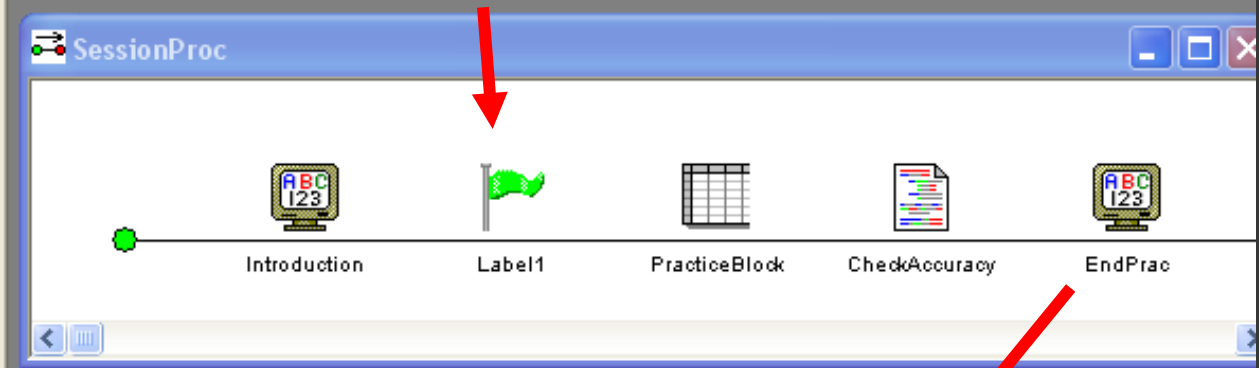
On the right, a flowchart illustrates the experimental sequence: Introduction (computer icon), PracticeBlock (grid icon), CheckAccuracy (document icon), BlockList (grid icon), and GoodBye (computer icon). A red arrow points from the 'CheckAccuracy' component in the Structure pane to the 'CheckAccuracy' block in the flowchart. Another red arrow points from the 'CheckAccuracy' block in the flowchart to a code editor window.

```
If Feedback.ACCStats.Mean > .80 Then
    EndPrac.Text = "Your accuracy for the practice trials was " & _
        Feedback.ACCStats.Mean*100 & "%" & _
        "\n\nGet ready for the real trials"
Else
    EndPrac.Text = "You did not achieve the required 80% accuracy" & _
        "\n\nYou must repeat the practice trials"
    Feedback.ACCStats.Reset
    EndPrac.Run
    GoTo Labell
End If
```



Properties window for 'EndPrac' (TextDisplay). The table below shows the properties:

| Property | Value |
|----------|---------|
| (Name) | EndPrac |
| (About) | |



EndPrac window showing a text input field. The text '<insert text here>' is displayed in the field. A red arrow points to this text. Below the window, the text 'tekst wordt at runtime ingevuld' is written in red.

tekst wordt at runtime ingevuld

1.4. Opmerking

- ◎ InLine lost niet alles op
- ◎ Vooral (restricties opleggen aan) randomisatie kan problemen opleveren
- ◎ Vb. gelijkende trials niet na elkaar
- ◎ Het kan (zie samples op site E-Prime) maar enkel met klein aantal stimuli (Do...Loop)

Oplossing

Extern gegenereerde lijst inlezen:

- LoadMethod File
- *.txt
 - TAB
 - \0
 - attribute headers moeten erin staan

The screenshot displays the E-Prime software interface with several windows open:

- Structure:** Shows a hierarchical tree of the experiment. The 'TrialList1' object is highlighted under 'BlockProc1'.
- Properties:** Shows the configuration for 'TrialList1 List'. The 'LoadMethod' is set to 'File' and the 'Filename' is 'listblok1.txt'.
- TrialList1:** A summary window showing a table of trial data. The table has columns: ID, weight, nested, Procedure, stimulus, pos, correctresp, type. The data rows are:

| ID | weight | nested | Procedure | stimulus | pos | correctresp | type |
|----|--------|--------|-----------|-----------------|-------|-------------|------|
| 1 | | | TrialProc | redcircle.bmp | left | f | c |
| 2 | 1 | 1 | TrialProc | redcircle.bmp | right | f | i |
| 3 | 1 | 1 | TrialProc | greencircle.bmp | left | f | i |
| 4 | 1 | 1 | TrialProc | greencircle.bmp | right | f | c |
- listblok1.txt - Notepad:** Shows the raw text of the file. The text is:

```
weight  nested  Procedure  stimulus  pos  correctresp  type
1          1          TrialProc  greencircle.bmp  left  f          i
1          1          TrialProc  redcircle.bmp   left  f          c
1          1          TrialProc  greencircle.bmp  right f          i
1          1          TrialProc  redcircle.bmp   right f          i
```

Overzicht

1. E-Basic

1. Visual Basic
2. Oefening 4
3. Andere voorbeelden
4. Opmerking

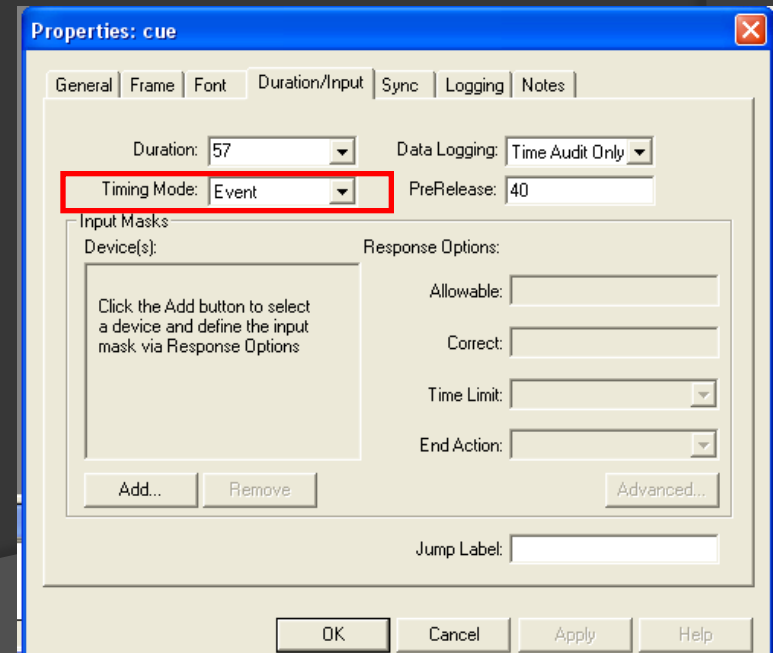
2. Timing issues

1. Timing modes
2. Refresh rates
3. Delays
4. Tips

2. Timing issues

2.1 Timing modes

- **Event:** zorgt voor vaste aanbiedingsduur (*Default*)
- **Cumulative:** zorgt voor vast interstimulusinterval



2. Timing issues

2.2. Refresh rates

*uiteindelijke
aanbiedingstijd
=
bedoelde
aanbiedingstijd?*

$$\begin{aligned}\text{Refresh duration} &= 1 / \text{refreshrate sec} && \text{60Hz} \\ &= 1 / 60 \text{ sec} \\ &= \mathbf{16,67} \text{ ms}\end{aligned}$$

Moeilijk mee te werken!

Duur van de stimulus = veelvoud van refresh duration

Als je de stimulus 60 ms wilt aanbieden...wordt dit **66.67 ms**

Oplossing: Verander refreshrate van je computer

2. Timing issues

*uiteindelijke
aanbiedingstijd
=
bedoelde
aanbiedingstijd?*

2.2. Refresh rates

$$\begin{aligned}\text{Refresh duration} &= 1 / \text{refreshrate sec} \\ &= 1 / 100 \text{ sec} \\ &= 10 \text{ ms}\end{aligned}$$

100Hz

Makkelijk mee te werken

Duur van de stimulus = veelvoud van refresh duration

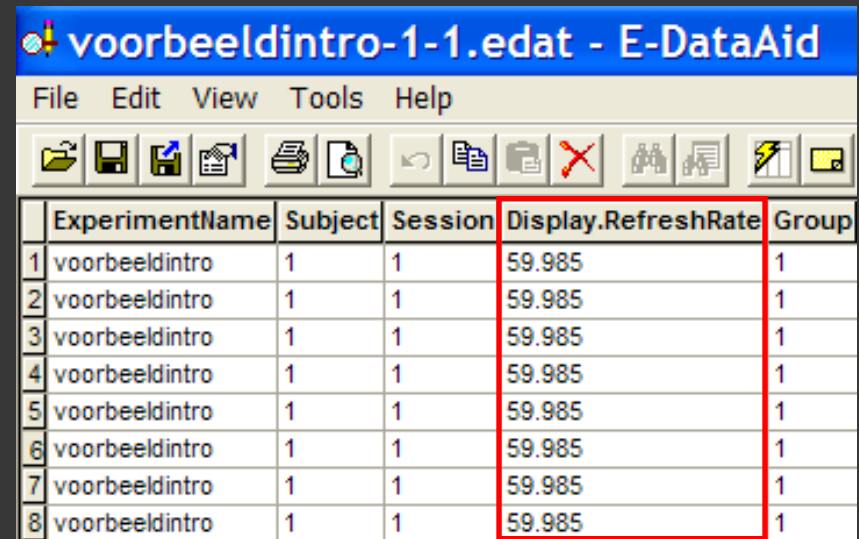
Dus bijvoorbeeld

$$4 \times 10 = 40 \text{ ms}$$

$$5 \times 10 = 50 \text{ ms}$$

$$6 \times 10 = 60 \text{ ms} \dots$$

2. Timing issues



Screenshot of the E-DataAid software interface. The window title is "voorbeeldintro-1-1.edat - E-DataAid". The menu bar includes File, Edit, View, Tools, and Help. The toolbar contains various icons for file operations and data manipulation. Below the toolbar is a table with the following data:

| | ExperimentName | Subject | Session | Display.RefreshRate | Group |
|---|----------------|---------|---------|---------------------|-------|
| 1 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 2 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 3 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 4 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 5 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 6 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 7 | voorbeeldintro | 1 | 1 | 59.985 | 1 |
| 8 | voorbeeldintro | 1 | 1 | 59.985 | 1 |

Hoe weet ik de refreshrate?

⇒ Check op je PC (let op, soms fout bij Windows)

⇒ Check in .edat file

⇒ Meer info op de website

<http://expsy.ugent.be/intern/eprimeFAQ.htm>

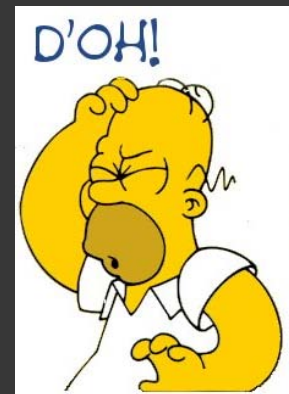
2. Timing issues

2.3. Besturingssysteem zorgt voor delays

Er gebeurt meer dan je denkt

- ophalen picture
 - voorbereiden picture
 - picture op scherm zetten
 - picture duration = **200 ms**
- } 105 ms

-> Totale duur voor volgende picture op het scherm komt is **305 ms**

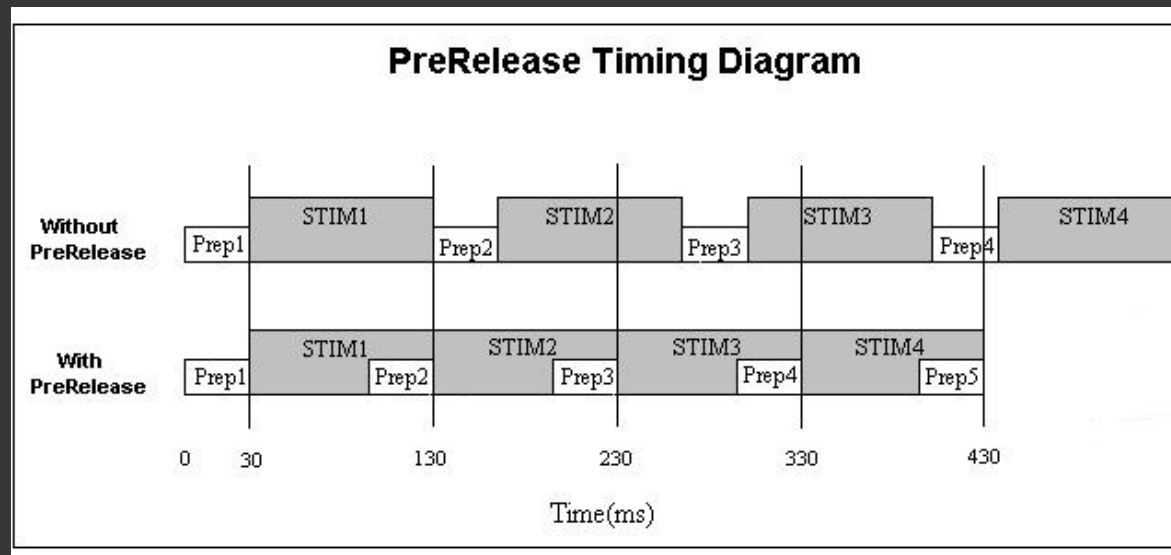


2. Timing issues

2.3. Besturingsysteem zorgt voor delays

Oplossing: **PreRelease**

Tijdens presentatie van huidige stimulus wordt de volgende stimulus al voorbereid



2. Timing issues

2.3. Besturingssysteem zorgt voor delays

Oplossing: **PreRelease**

- Voorbereidingstijd $<$ Display Time
Bijvoorbeeld 100ms $<$ 200 ms
- Algemeen is een PreRelease van 100 à 200 ms ruim voldoende

Structure

- probe
- fb1
- klar2
- InLine1
- blocklist
- blockproc
 - fixation
 - cue
 - mask
 - isi
 - probe
 - fb2
 - InLine2
- thanx
- Unreferenced E-Objects

Properties

fixation TextDisplay

| (Name) | fixation |
|------------------|----------|
| (About) | |
| (Property Pages) | |
| AlignHorizontal | center |
| AlignVertical | center |
| BackColor | white |
| BackStyle | |

Name
Uniquely identifies each object.

fixation

#

cue

[cue1] [cue2]

Properties: fixation

General | Frame | Font | Duration/Input | Sync | Logging | Notes

Duration: 390 Data Logging: Time Audit Only

Timing Mode: Event PreRelease: 100

Input Masks

Device(s): Response Options:

Allowable:

Correct:

Time Limit:

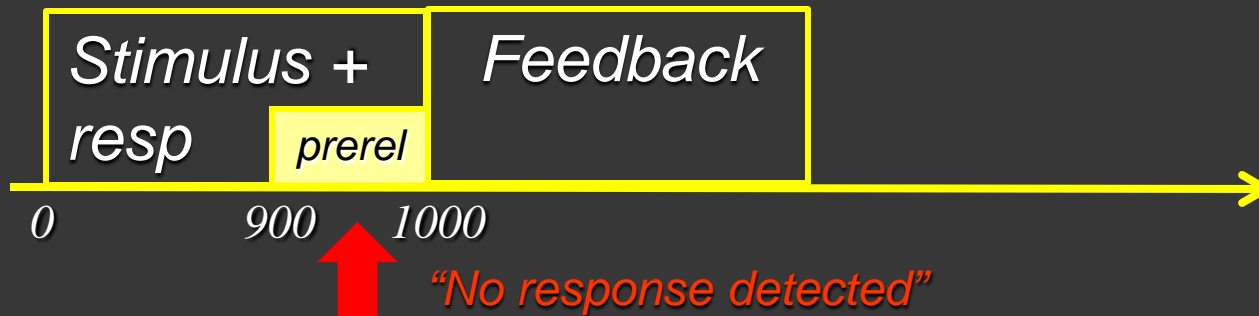
End Action:

Jump Label:

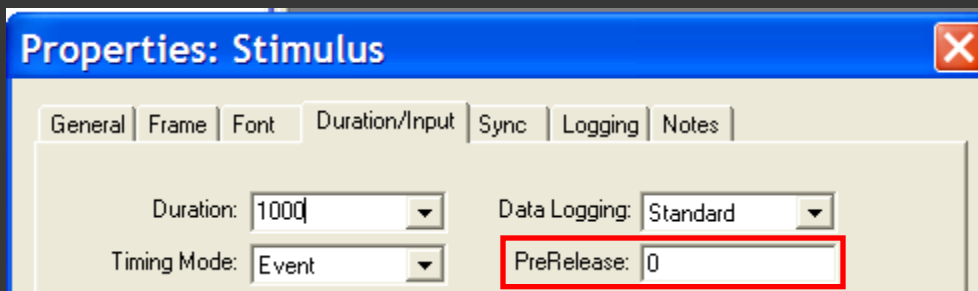
Buttons: Add... Remove Advanced... OK Cancel Apply Help

Géén PreRelease

- ⊙ Vóór een FeedbackObject



-> Oplossing: Zet PreRelease op 0 ms bij het object voorafgaand aan FeedbackObject

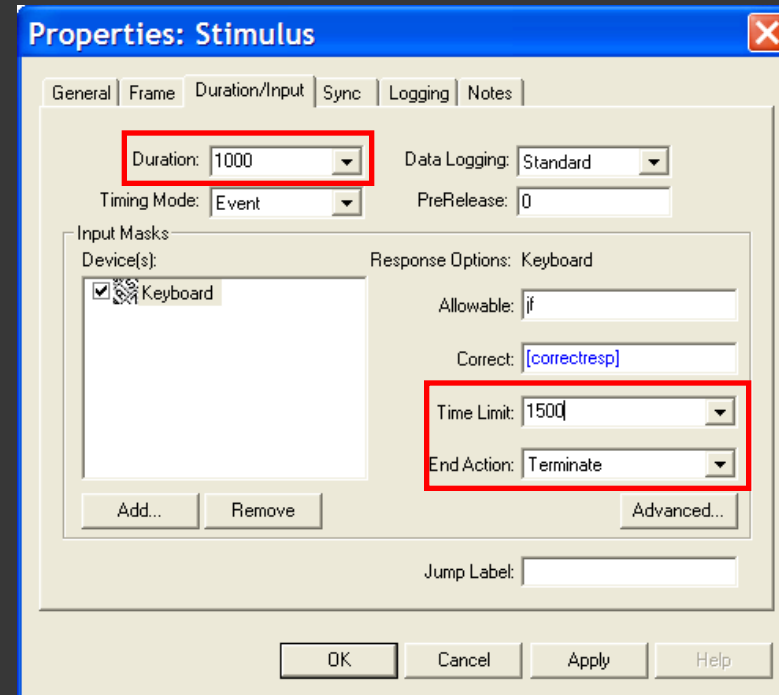


(Default is 0)

Géén PreRelease

⊙ Bij object dat respons registreert

- 1) als responstijd > stimulusduur
- 2) als End Action = Terminate



2. Timing issues

2.4 Tips

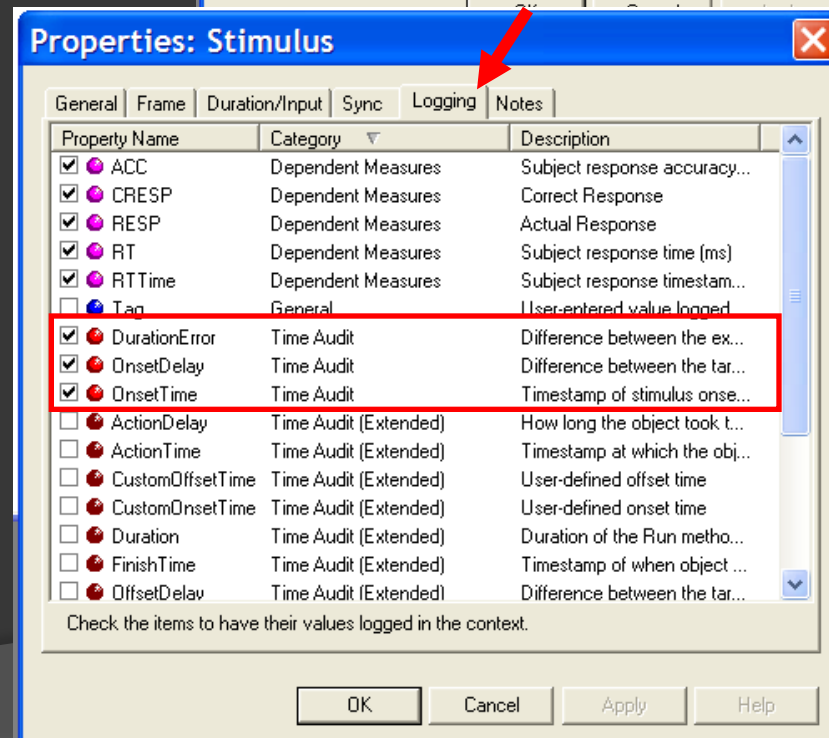
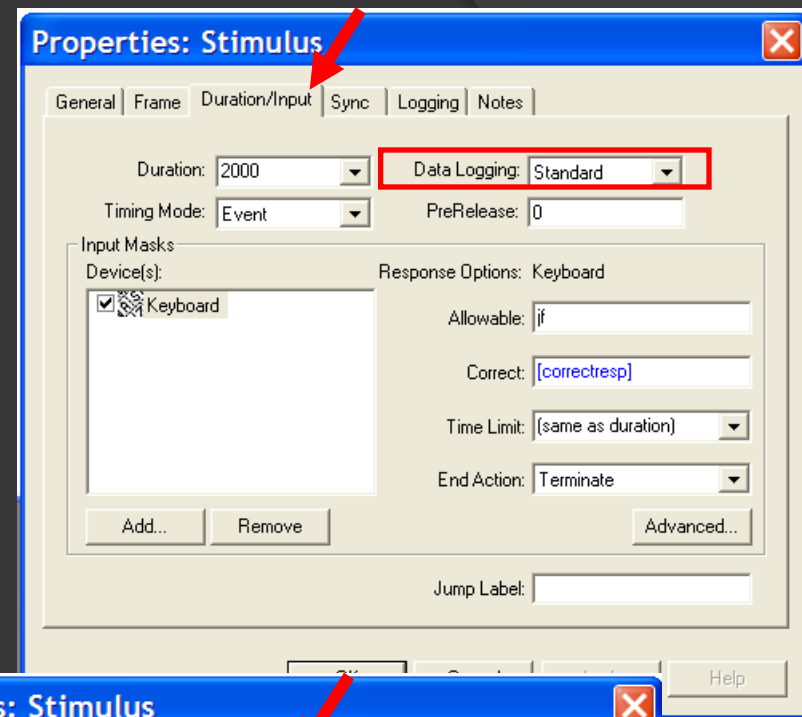
- Logging
- ~~Achtergrondprogramma's~~
- Refresh Clock Test

www.pstnet.com

-> Support

-> Download

-> Misc



Test

- ◎ 19/10 om 10u in PC klas 1
- ◎ Experiment van nul programmeren
- ◎ Oplossingen extra's online
<http://users.ugent.be/~iimbo/Teaching.htm>