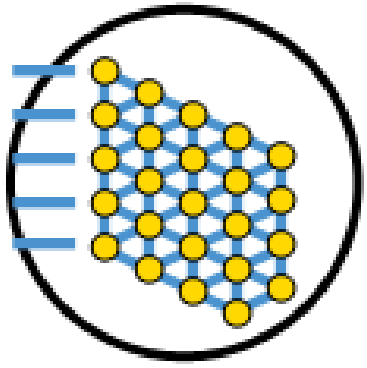
E-Prime

Extensions



Net Station 2.0

User Manual

For Research and Education Only

Psychology Software Tools, Inc.
311 23rd Street Extension, Suite 200
Sharpsburg, PA 15215-2821
Phone: 412.449.0078
Fax: 412.449.0079
E-mail: info@pstnet.com

PST-100952

E-Prime Extensions for Net Station 2.0 User Manual
PST-100954
Rev 2

Psychology Software Tools, Inc.
311 23rd Street Extension, Suite 200
Sharpsburg, PA 15215-2821
Phone: 412-449-0078
Fax: 412-449-0079
E-mail: info@pstnet.com
Web: www.pstnet.com

For questions or comments regarding this manual or installation assistance:
Please e-mail us at support@pstnet.com or visit us at https://support.pstnet.com

This manual describes the installation procedure for the E-Prime Extensions for Net Station. Please review the manual completely and thoroughly before beginning the system installation.

The E-Prime Extensions for Net Station (PST-100952) is for research and educational purposes only.

Table of Contents

Chapter 1: Getting Started Guide 6
1.1 Installation Instructions 6
1.2 E-Prime Extensions for Net Station 2.0 Overview 6
1.3 Software Installation 6
1.4 Network Configuration 10
1.5 Troubleshooting 16
1.6 Product Service and Support 19
1.7 Resources 19
Chapter 2: How to Create Your Own Net Station Experiment 20
2.1 Overview 20
2.2 Programming Methods 21
Tutorial 1: **Adding Net Station Support to an E-Prime Experiment 23**
Tutorial 2: **Adding User Defined Keys 48**
Tutorial 3: **Adding a Practice Block 54**
Tutorial 4: **Timing Test 67**
Chapter 3: Net Station PackageCall Reference 71
3.1 Introduction 71
Chapter 4: Contact Information 85

Chapter 1: **Getting Started Guide**

1.1 Installation Instructions

E-Prime Extensions for Net Stations 2.0 is compatible with E-Prime 2.0 Professional Service Pack 1 (SP1) only. See the following Knowledge Base article: KB 5346 INFO: Which version of E-Prime should I use with E-Prime Extensions? and E-Prime 2.0 Requirements for additional information.

For a summary of E-Prime's new features, review the *E-Prime New Features/Reference Guide* that can be accessed through the Start menu: Start\All Programs\E-Prime 2.0\Documentation, or through the Help menu in E-Studio: Help > Documentation > New Features Reference Guide.

1.2 E-Prime Extensions for Net Station 2.0 Overview

E-Prime Extensions for Net Station 2.0 is a set of software routines that allow communication between Net Station and E-Prime during the run of an experiment. Also included is a set of sample experiments that can be run directly, or used as a basis from which to create new experiments[1].

1.3 Software Installation

Before continuing, be sure that you have administrative rights to install this software on the computer. If you do not have administrative rights, you will be unable to install E-Prime Extensions for Net Station 2.0. If you are unsure of your administrative privileges, contact your System Administrator. The installation will update the default E-Prime installation with the files required for use with Net Station.

⚠ **NOTE:** *This manual assumes that you are installing the software on a computer configured to use the Mac NTP synchronized clock. If you are developing an experiment on a computer that is not connected to Net Station, you will want to select the E-Prime Primary Real-time Clock instead of the SNTP clock. This is done in the 'Timing' tab of the Experiment Object. Be certain to set the clock back to the SNTP clock in the experiment before testing or running the experiment on a PC connected to Net Station to ensure good timing results. See the Knowledge Base articles: KB 5468 FEATURE: SNTP Clock configuration options can now be accessed from the Experiment Object and KB 5612 FEATURE: E-Prime SNTP Realtime Clock for additional information.*

Install E-Prime 2.0

1) ***Uninstall*** any **previous version** of **E-Prime** and **E-Prime Extensions for Net Station 2.0** using the **Programs and Features** Control Panel in Windows 7 or the **Add/Remove** Control Panel in Windows XP. The **E-Prime Extensions for Net Station 2.0** ***requires*** a copy of **E-Prime 2.0 Professional** or newer to be installed on the machine.
2) ***Insert*** the **E-Prime 2.0 CD** into your **CD-ROM** drive.
   ⚠ **NOTE:** *The CD that came with your purchase may not have the most current version of E-Prime or the Net Station extensions. For the most recent updates, contact EGI (supportteam@egi.com) or PST (https://support.pstnet.com) and download the recommended versions.*
3) If installing from the CD, the **installation** should ***automatically launch***.
   ⚠ **NOTE:** *If it does not, you may use Windows Explorer to browse the CD and launch the Setup.exe file in the main folder.*
4) ***Follow*** the **prompts** in the **installation program** to ***provide*** any **required information**. (e.g. User Name, Institution, Serial Number, etc).
   ⚠ **NOTE:** *Installation of E-Prime requires a valid E-Prime License (verified through the E-Prime HASP USB hardware key and Serial Number).*
   ⚠ **NOTE:** *In order to use PST online Product Service and Support you will need to install your software with a valid Serial Number.*

[1] Please note this manual is intended for use with the TCP/IP (Ethernet) and NTP based Experimental Control Interface methods. If you have issues regarding the legacy serial communications or earlier synchronization methods, please contact EGI.

Installing E-Prime Extensions for Net Station 2.0

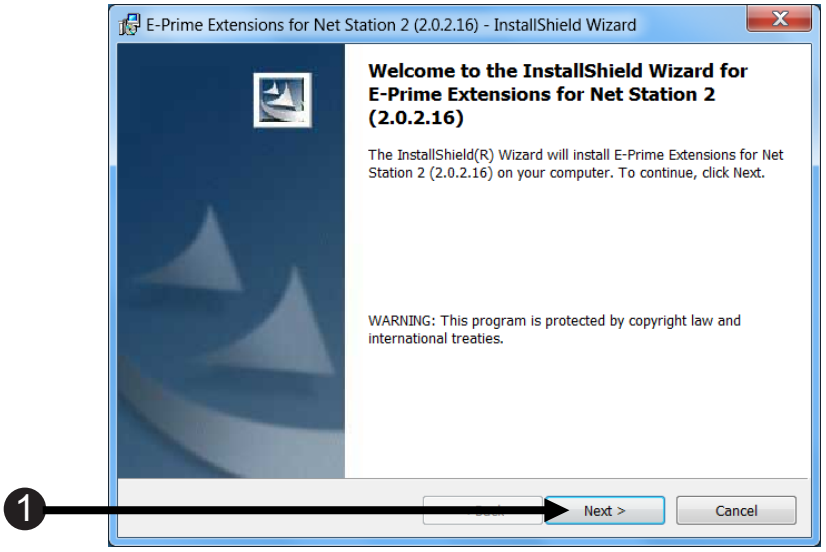
Before continuing, be sure that you have administrative rights to install this software on the computer. If you do not have administrative rights, you will be unable to install E-Prime Extensions for Net Station 2.0 (EENS). If you are unsure of your administrative privileges, contact your System Administrator.

If you have an E-Prime system purchased from Electrical Geodesics (EGI), contact EGI before performing any installations to be certain you have compatible and current versions of E-Prime and E-Prime Extensions for Net Station (EENS). E-Prime and related software must be installed using an administrative account as pre configured on E-Prime systems purchased from EGI.

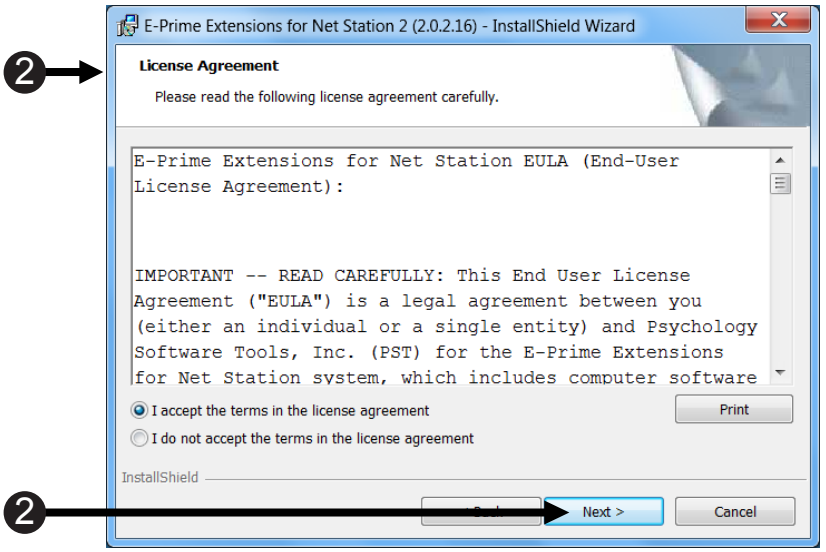
⚠ **NOTE:** *E-Prime Extensions for Net Station 2.0 is compatible with E-Prime 2.0 Professional only.*

⚠ **NOTE:** *The version number on the following screen grabs may not correspond to the version number on your software.*

1) ***Insert*** the **Net Station installation CD** into your **CD-ROM drive** or ***double click*** on the downloaded **EENS installer**.
Click the **Next** button to continue installation.



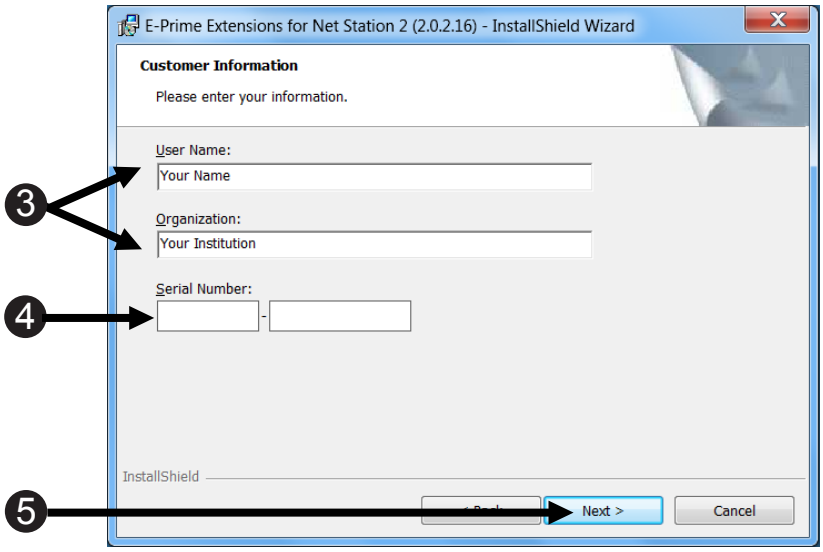
2) ***Please read*** the **License Agreement** and make sure that you ***agree completely*** with the terms and conditions described in the **agreement** before proceeding. Once you have read the agreement, ***click*** **Next** to proceed with the installation.



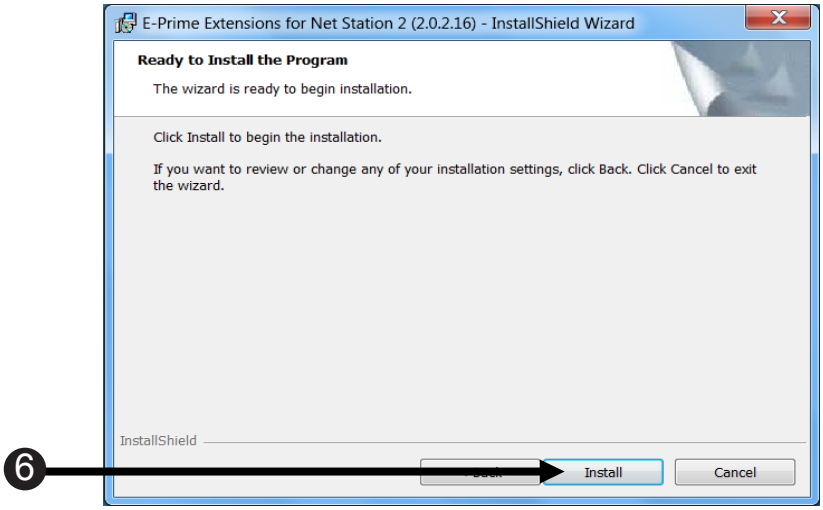
3) ***Specify*** **Your Name** and **Your Institution** or check with your system administrator for appropriate information.

4) The **Serial Number field** must be ***complete*** to obtain access to online **Product Service and Support**.

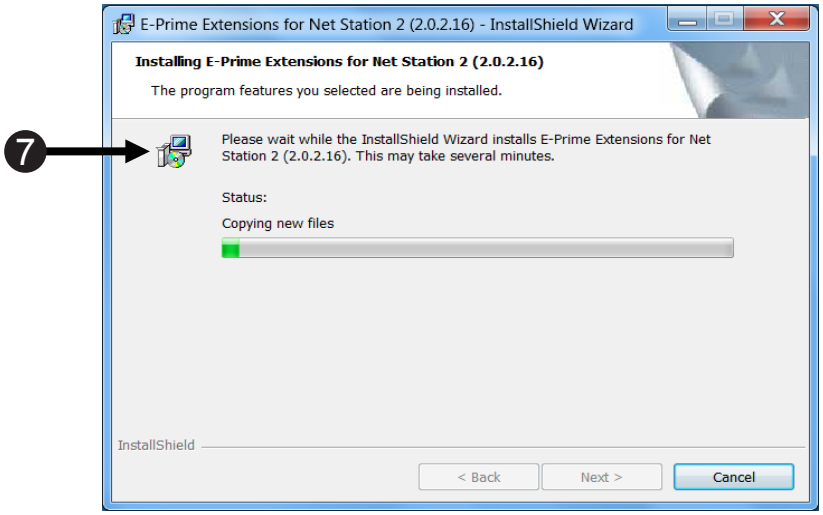
5) ***Click*** **Next** to begin transferring files to your computer.



6) ***Click*** **Install** to continue the installation.

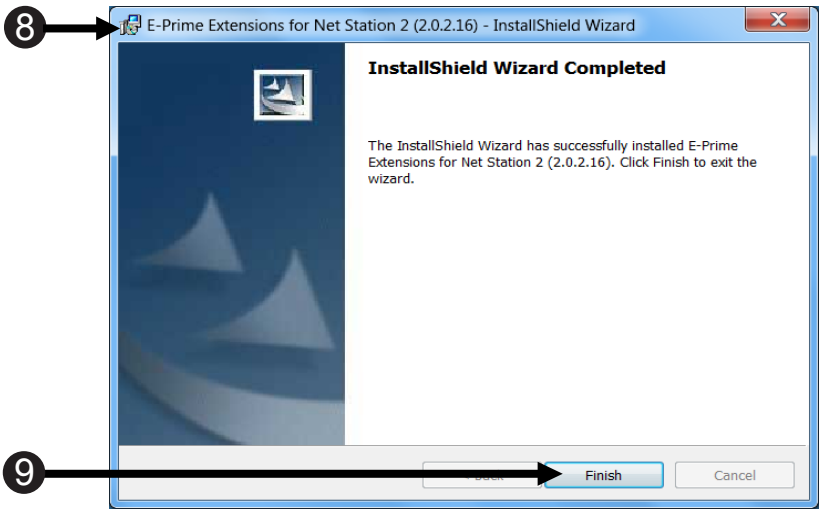


7) ***Wait*** while the installer configures the software.



8) If **E-Prime Extensions for Net Station 2.0** is installed properly, you will see the following window.

9) ***Click*** **Finish** to complete installation.



1.3 Software Installation (continued)

Finding the "My Experiments" Folder

E-Prime 2.0 is compatible with Microsoft Windows XP, Vista and 7. If installing EENS, it is recommended that only Windows 7 be used. If another operating system is required due to requirements at your facility, please contact EGI (supportteam@egi.com). You will frequently be working within the "My Experiments" folder, because this folder is the default location to store new experiments created with E-Studio. E-Prime creates the "My Experiments" folder in your personal documents folder on your PC. This folder also contains the "Samples" folder, which stores the sample experiments that are documented in Appendix B of the E-Prime User's Guide, and the "Tutorials" folder, which stores the E-Studio files that are documented in the E-Prime Getting Started Guide.

The table below shows the default paths to your personal documents folder. Note that the path on your particular machine may have been modified by your administrator:

Operating System	Path to your personal documents folder ("My Documents" or "Documents")
XP	<drive>\Documents and Settings\<user name>\My Documents\
Vista	<drive>\Users\<user name>\Documents\
Windows 7	<drive>\Users\<user name>\My Documents

When the E-Prime documentation directs you to the "My Experiments" folder, it does not include the full path to the folder. Instead, the documentation refers to "…My Experiments", where the "…" indicates the full path up to your personal documents folder. When you see this notation in the documentation, replace the "…" with the path to your personal documents folder.

1.4 Network Configuration

TCP/IP Configuration:

This document covers the configuration of EGI's NA400 amplifier. For information on configuring the NA300 amplifier, contact EGI (supportteam@egi.com). Interactions between E-Prime and Net Station use TCP/IP communications. There are two separate components that require distinct connections to Net Station software: ECI and NTP. The Experimental Control Interface (ECI) uses TCP/IP to control Net Station recording operations and to send event information to Net Station. This connection is between E-Prime and Net Station directly. Network Time Protocol (NTP) is an industry standard mechanism for communicating clock and timing information between computers or devices. Net Station software employs this mechanism to provide timing information to E-Prime so that events in E-Prime are synchronized with the EEG data collected by EGI's amplifiers.

To configure ECI, set the IP address in the EENS Init PackageCall as described in the tutorials in this manual. To configure NTP, you can edit the file SNTPClockExtension.ini by using the Timing tab of the Experiment Object properties dialog.

***NA400 NTP Configuration*:**
The NA400 uses Ethernet to communicate data that is collected by the Amp to Net Station running on a Mac computer. The system clock on the amplifier provides the NTP service to E-Prime. An Ethernet switch is used to connect the Net Station computer, the E-Prime computer and the NA400 amplifier together in a local network. This local network should be isolated from any other networks or the internet. To connect the system to external networks for file transfer or support purposes, use the second Ethernet port on the Mac desktop system or the wireless connection on the Mac laptop.

NA400 ECI Configuration:
For the NA400, follow these steps to set a static IP address on the Mac (OS 10.8):

1) ***Launch* System Preferences** from the Apple menu.

2) ***Click*** on the **Network** icon.

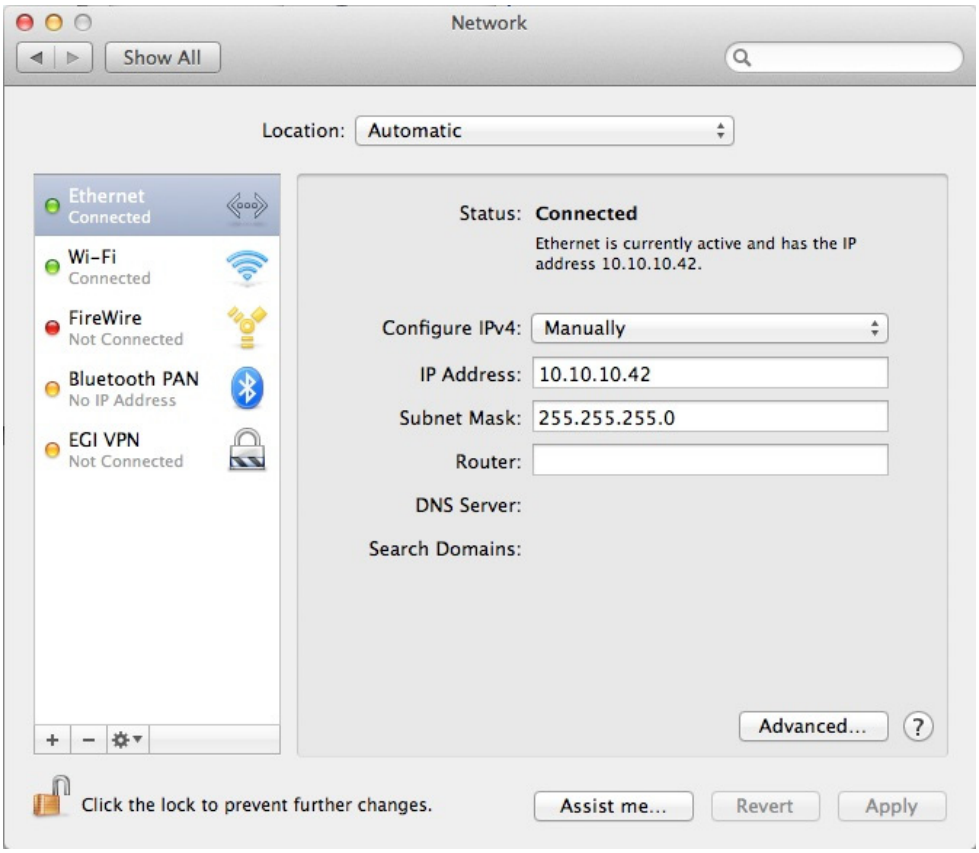
3) To create a unique network location, ***select*** the **location** or ***create*** a **new location** from the **"Location:"** pop up menu or ***edit*** the **default settings** for the location **"Automatic"**.

4) ***Select*** "**Ethernet**", from the left panel. If there is more than one Ethernet option, ***select*** the **one** for the **Ethernet port** to which you have **connected** the **E-Prime computer**.

5) ***Select*** "**Manually**" from the "**Configure IPv4:"** pop up menu to the right.

6) ***Type*** your **Macintosh's static IP address** into the "IP Address text field.
For the NA400, the factory IP setting is 10.10.10.42.

7) ***Type* 255.255.255.0** into the **subnet mask field text box**.



Set a static IP address on your PC (Windows):

1) ***Open*** the **Local Area Connection Status** control panel by going to **Start** > **Control Panel** > **Network and Sharing Center**> **Local Area Connection**.

2) ***Click*** the **Properties** button to ***open*** the **Local Area Connection Properties** panel.

3) ***Select*** "**Internet Protocol (TCP/IPv4)**" from the list, and ***click*** the **Properties** button to ***open*** the **Internet Protocol** (TCP/IP) Properties panel.

4) ***Select*** the "**Use the following IP Address**" radio button.

5) ***Type*** your **PC's static IP address** (e.g., 10.10.10.43) into the "**IP address**" text field.

6) ***Type*** **255.255.255.0** into the **subnet mask text field**.

7) ***Click*** the **OK** button to ***save*** your selection, and ***close*** the **Local Area Connection Properties** panel.

Running a Test Experiment:

Once you have completed the software and hardware installation, and have configured the TCP/IP settings on your Macintosh and PC computers, you should run a test experiment. In this section, you will use the NSBasicRT.es2 sample experiment provided with the E-Prime Extensions for Net Station 2.0 installer. This test run will also familiarize you with the procedures for setting up other experiments for TCP/IP ECI communications.

Launch the NSBasicRT.es2 sample experiment:

1) ***Launch*** the **E-Studio application** from the Windows Start menu by ***selecting*** **Start** > **Programs** > **E-Prime** > **E-Studio**.

2) After **E-Studio** launches, ***dismiss*** the "**New E-Prime 2.0 Experiment**" template dialog.

3) ***Use*** the **Open File** dialog to ***navigate*** to appropriate folder, and ***load*** the **NSBasicRT.es2** experiment. The default file path is:

…\My Experiments\Net Station\Samples\NSBasicRT

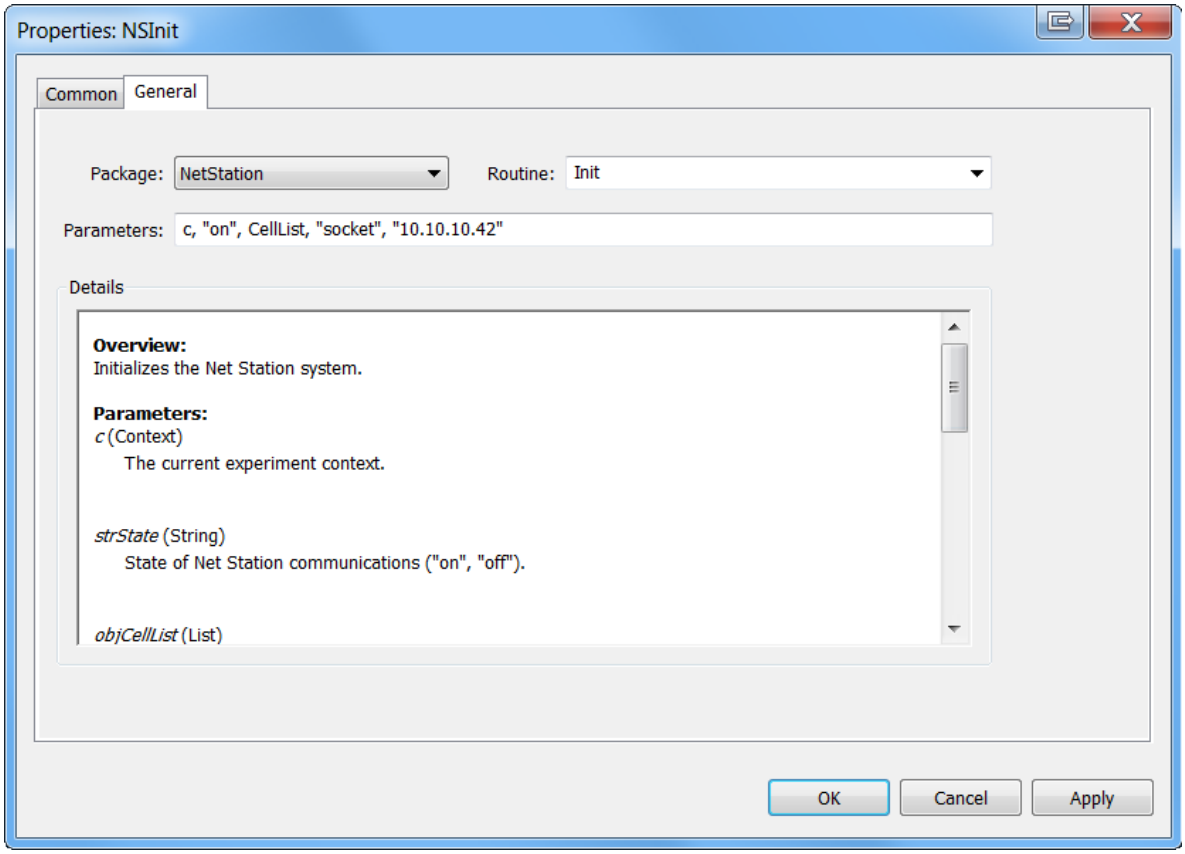
4) You may also ***launch*** any **E-Studio** experiment by ***double-clicking*** on the experiment file in Windows Explorer.

Specify the NSInit parameters:

1) Once you have ***loaded*** the **NSBasicRT.es2** experiment, you must set it up to ***communicate*** using the **static IP address** of the **Net Station Macintosh**. ***Double-click*** on the **icon** "**NSInit**" in the **Structure Window**.

2) In the "**Parameters**" text field, ***add*** "**socket**" and the **static IP address** ("XXX.XXX.XXX.XXX") of your **Net Station Macintosh**. For example, assuming the Mac IP address is 10.10.10.42, you would enter:

c, "on", CellList, "socket", "10.10.10.42"

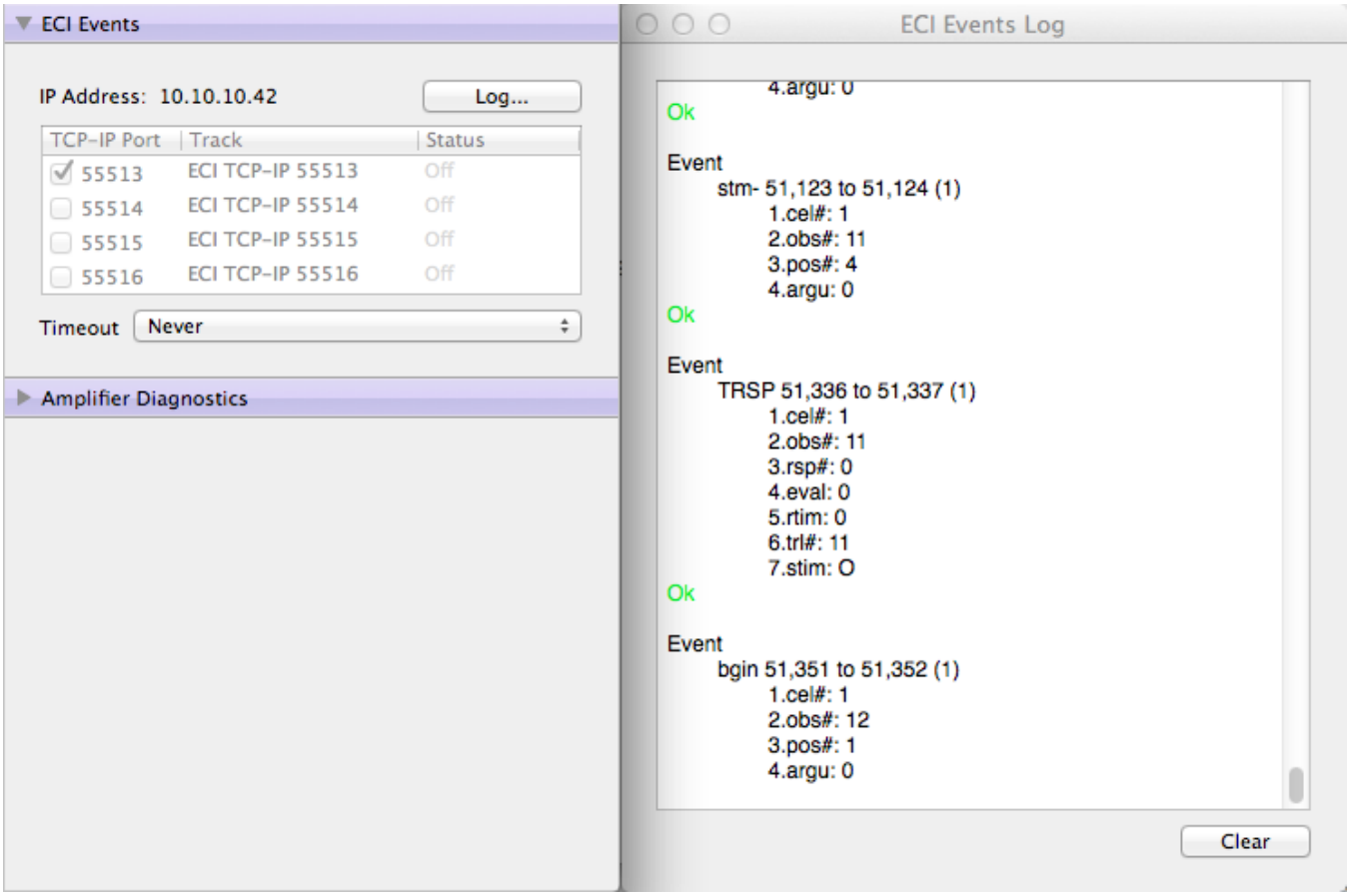


3) ***Click*** the **OK** button to ***save*** your changes and ***close*** the **NSInit** object.

Launch Net Station:

1) ***Follow*** the **instructions** provided by EGI to ***set up*** Net Station for acquisition. Please ***use*** the default ECI port, **55513**, for this test run.

2) In the **ECI panel**, ***click*** the **Log** button to ***open*** the **Log panel**.



4) Start acquisition on the Mac. This will start the NTP services in Net Station that allow E-Prime to synchronize with the EGI amplifier.

Run the experiment:

1) ***Select*** **E-Run** > ***Run*** from the **application menu** to ***generate*** and ***run*** the **experiment** (alternatively you can press F7, or click the Run button).

2) ***After*** the **experiment** begins ***executing***, it will present message boxes prompting you to ***enter*** **StartUp Info**. ***Accept*** the **default values** or ***provide*** a **valid value** for each prompt.

3) If the experiment and Net Station have established communications successfully, the experiment will proceed to the experiment instructions and should execute to completion.

   During the execution of the experiment trials, Net Station will display the messages being sent from E-Prime within the ECI Log panel and the “ECI TCP/IP 55513” Event Track. It is recommended that you keep these visible during testing for reference.

4) If you have trouble running the test experiment, refer to **1.5 Troubleshooting,** (Page 16) and **1.6 Product Service and Support,** (Page 19) sections at the end of this chapter before continuing.

   After you have successfully run NSBasicRT.es2 and have verified that both systems are working properly, you should proceed to Chapter 2, which provides a set of Tutorials that are designed to walk you through the process of adding Net Station support to an existing E-Prime experiment.

⚠ **NOTE:** *Prior to collecting any real data with E-Prime and Net Station, it is strongly recommended that you review the critical timing information contained in* ***Chapter 2, Tutorial 4: Timing Test****, (Page 67).*

Alternate NSInit Parameters:

1) E-Prime will use default TCP port 55513 if none is specified in the “Parameters” text field. If you want to use a different port, you may specify it by entering 55514, 55515, or 55516 after the IP address, separated by a colon (“XXX.XXX.XXX.XXX:NNNNN”). For example:

c, “on”, CellList, “socket”, “10.10.10.42:55514”

2) Net Station will default to port 55513. If you want to use a different port, you many specify it in the Configuration settings of the Net Station ECI panel. You may select multiple ports if you wish, and Net Station will automatically activate the port on which E-Prime sends events during each experimental run.

The NETSTATION.INI file

If you prefer, you may use the NETSTATION.INI file to specify global communication parameters so that there is no need to edit the NSInit package in individual experiments. If no local communication parameters are found in the NSInit package at experiment runtime, E-Prime will automatically search NETSTATION.INI for global parameters to use. Local settings in an experiment's NSInit package will always override any global settings in the NETSTATION.INI file.

Please note that changing these values will put the system into a configuration other than that recommended by EGI. Please contact EGI (supportteam@egi.com) before modifying these settings.

Specify Socket communication parameters in NETSTATION.INI:

1) Navigate to the NETSTATION.INI file. The default file path is:

C:\WINDOWS\NETSTATION.INI

2) ***Right-click*** on the file and ***select*** "**Properties**". In the General tab, ***verify*** that the "**Read-only**" box is unchecked so that you may ***edit*** the file.

3) Under the "**[Communications]**" header, indicate that you want to use the **Socket (TCP/IP)** method by ***entering***:

Method="socket"

4) Now, ***specify*** the **IP address** of your **Net Station Macintosh** computer ("XXX.XXX.XXX.XXX"). For example:

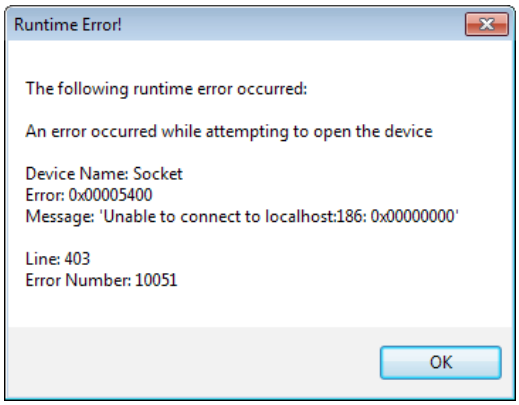
MethodParams="10.10.10.42"

5) ***Save*** and ***close*** the **NETSTATION.INI** file.

1.5 Troubleshooting

If you have difficulty running an experiment, take note of any error messages that you see and try some of the suggestions below. If you still have issues, please refer to the **1.6 Product Service and Support,** (Page 19) section at the end of this chapter for contact information.

Communication cannot be verified:



If your E-Prime experiment and Net Station fail to establish TCP/IP (socket) ECI communication check the following:

1) ***Verify*** that all **Ethernet cables** are properly ***connected*** between the **Net Station Mac** and the **E-Prime PC** and that the optical network cable is connected between the NA400 and the optical network port on the switch.

2) ***Review*** the **Parameters** of the "**NSInit**" object in your E-Prime experiment.
Refer to **1.4 Network Configuration,** (Page 10) of this chapter, for an example of NSInit Parameters.

3) ***Verify*** that the **Net Station** and **E-Prime** have the same **TCP Port *selected*** for ECI communication.

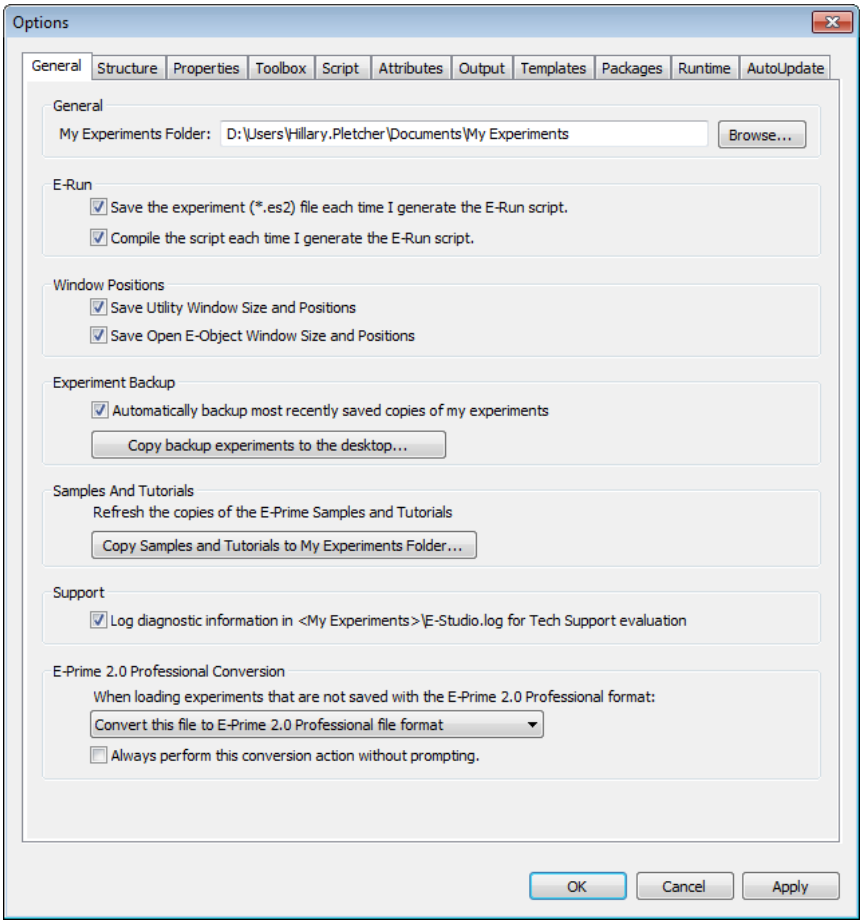
4) ***Verify*** that you have set static IP addresses on your PC and Macintosh computers.

5) If you have active firewalls on your PC and/or Macintosh, ***verify*** that you have ***allowed*** **exceptions for the TCP ports** you are trying to use for E-Prime and Net Station ECI communications.

Cannot locate E-Prime or Net Station Sample experiments and Tutorials:

Sample experiments and tutorials should be copied to each users ...\My Experiments folder after installation. If you cannot find the Samples and Tutorial experiments or if you just wish to refresh the copies you already have you may force E-Studio to recopy them for you.

1) ***Open*** **E-Studio** and ***select*** **Tools** > **Options**… from the application menu bar.



2) ***Click*** on the "**Copy Samples and Tutorials to ...\My Experiments Folder…**" button at the bottom of the Options dialog.

3) If there are sample files already there you will be prompted to overwrite them.

1.6 Product Service and Support

Psychology Software Tools, Inc. provides technical support for E-Prime via the PST web site. In order to receive technical support, you must register online at

https://support.pstnet.com

Registration requires a valid E-Prime serial number and e-mail address. At the support site, you will also find a Knowledge Base including release notes and a compilation of frequently asked questions. The support site also includes E-Prime sample paradigms that are available for you to download.

Electrical Geodesics, Inc. provides technical support for E-Prime Extensions for Net Station 2.0. To receive technical support, contact EGI at supportteam@egi.com.

Chapter 2: **How to Create Your Own Net Station Experiment**

2.1 Overview

This chapter will illustrate the creation of an E-Prime/Net Station experiment using one of the sample paradigms provided with the E-Prime Extensions for Net Station 2.0 installation (EENS), the SEFixedPositionAOI.es2 paradigm. It will also introduce and explain some programming methods that E-Prime uses.

In order to complete the tutorials, you will need a computer with E-Prime and the EENS software already installed. If you have not installed E-Prime, please complete the installation before continuing. If you have not installed EENS, please refer to the installation section in this manual, **Chapter 1: Getting Started Guide**, (Page 6).

While this document includes some basic introduction to programming concepts specific to using E-Prime with Net Station, the tutorials assume you are familiar with using E-Prime to build behavioral experiments. If you are new to using E-Prime, it is suggested that you work through all of the tutorials included in the *E-Prime 2.0 Getting Started Guide* and *E-Prime 2.0 User's Guide* prior to beginning this tutorial.

2.2 Programming Methods

Programming Methods Overview

The E-Prime Extensions for Net Station 2.0 (EENS) combines two technologies to create an integrated programming environment: PackageCalls and InLine Script. The typical EENS experiment will include use of both technologies. A brief overview of each is provided below.

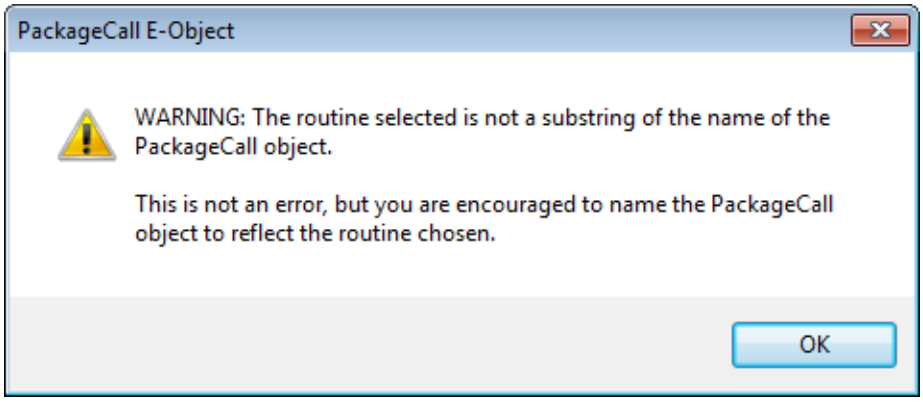
Use of PackageCalls in the EENS

PackageCalls are pre-written, cohesive sets of E-Basic routines grouped together in a single file that can be maintained externally. These libraries are included in the EENS installation. The EENS PackageCalls will allow you to automatically perform various functions e.g., hardware initialization and data output formatting via graphical representations in the Structure window in E-Studio. Some of the EENS PackageCalls will require you to set or modify parameters to enable them to work with your experiment. **Chapter 3: Net Station PackageCall Reference**, (Page 71) provides a listing of available routines along with descriptions of how to use them.

PackageCall Method

Routines contained within a package file are typically called by dragging a PackageCall from the E-Studio Toolbox and dropping it into a Procedure Object at the location within the experiment where the call is to be invoked.

When using the PackageCall, it is strongly recommended that you rename the object to reflect the specific Package File and Routines which are being referenced within the object. If you do not follow this recommendation, E-Prime will issue the following warning:



It is also a common convention, when calling routines within the Net Station Package File, to create a name for each object by abbreviating “Net Station” as “SE” and then concatenating the name of the routine being called. For example, a PackageCall that is configured to call the “Open” routine would be named “SEOpen”.

InLine Script Method

An InLine is an object used to insert user-defined script into an experiment at a specific point. The object is placed at the desired location in the script. The InLine method can be used to call PackageCalls as well. However, unless there is a need to make a series of PackageCalls in sequence or mingle PackageCalls with additional E-Basic script, this calling method is not typically recommended.

If you are making direct PackageCalls via E-Basic script, please refer to **Chapter 3: Net Station PackageCall Reference**, (Page 71) and **Tutorial 2: Adding User Defined Keys,** (Page 48). These sections of the manual provide PackageCall parameters and functions as examples of the script commonly required to successfully invoke each routine.

⚠ **NOTE:** *Users must have the EENS fully and correctly installed prior to using the Net Station PackageCalls by either method.*

Resolution

Experiment video resolution must match the world configuration specified in Net Station. Please refer to the Net Station documentation that came with your Net Station System.

Configuring Net Station System

Please refer to the Net Station documentation.

⚠ **NOTE:** *When using the E-Prime Extensions for Net Station 2.0, E-Prime and the Net Station system, you must have a two computer setup, in order to collect tracking data. E-Prime and the E-Prime Extensions for Net Station must be installed one computer and the Net Station system must be installed on another system. Then the two computers must be connected with a network hub or switch in order to communicate with each other.*

Tutorial 1: **Adding Net Station Support to an E-Prime Experiment**

Summary:

Net Station enabled experiments can be run locally from E-Studio during development and testing (i.e. without having the Net Station hardware connected). You should fully test your experiment locally before deploying it to the experiment presentation computer connected to your Net Station hardware. Note that while many Net Station related errors can be detected while testing locally, some errors will only be detected or generated when the actual Net Station hardware is in use. Thus, you should fully test your experiment with the Net Station hardware connected and active prior to scheduling actual participants.

In order to test without EGI hardware and software connected, you will need to use the E-Prime realtime clock instead of the SNTP clock, which requires that an NA400 be attached to your system. This is done in the Timing tab of the Experiment Object properties panel. You will also need to turn off ECI communications by changing the parameter in the NSInit object from “on” to “off” as shown in **Task 5: Add a Net Station PackageCall to initialize the system,** (Page 28).

Note that it is critical that these settings be set back to ‘SNTP clock’ and ECI communications “on” in order to get events sent to Net Station with correct timing.

See the Knowledge Base articles: KB 5468 FEATURE: SNTP Clock configuration options can now be accessed from the Experiment Object and KB 5612 FEATURE: E-Prime SNTP Realtime Clock for additional information.

Goal:

This tutorial illustrates how to add support for Net Station into the PictureRT sample experiment that is included with E-Prime.

Overview of Tasks:

- Load the PictureRT sample experiment and resave it as NSPictureRT
- Add the Net Station Package File to the Experiment Object
- Enable all of the Startup Info prompts that are required by Net Station
- Add Net Station PackageCalls to initialize/uninitialize the experiment at the Session level
- Add Net Station PackageCalls to start/stop recording at the Block level
- Add Net Station PackageCalls to initialize/uninitialize each trial
- Add Net Station PackageCalls to send RESP, Trial, and TRSP events to Net Station
- Assign unique four character codes to each event of interest within the trial sequence
- Add and configure the CellList
- Assign CellNumbers to individual trials
- Verify the overall experiment structure and run the experiment

Recommended readings:

It is recommended that you have read and worked through the tutorials included in the *E-Prime Getting Starting Guide* before beginning this tutorial. Prior to collecting critical data, it is recommended that you also read Chapter 4: Critical Timing, in the *E-Prime User’s Guide* as well as **Chapter 2, Tutorial 4: Timing Test,** (Page 67)**.**

Estimated Tutorial Time: 30-45 minutes

Task 1: Open the PictureRT.es2 experiment in E-Studio

Locate the E-Studio icon in the Start > Programs > E-Prime menu and launch the application by selecting it. Load the PictureRT.es2 sample experiment.

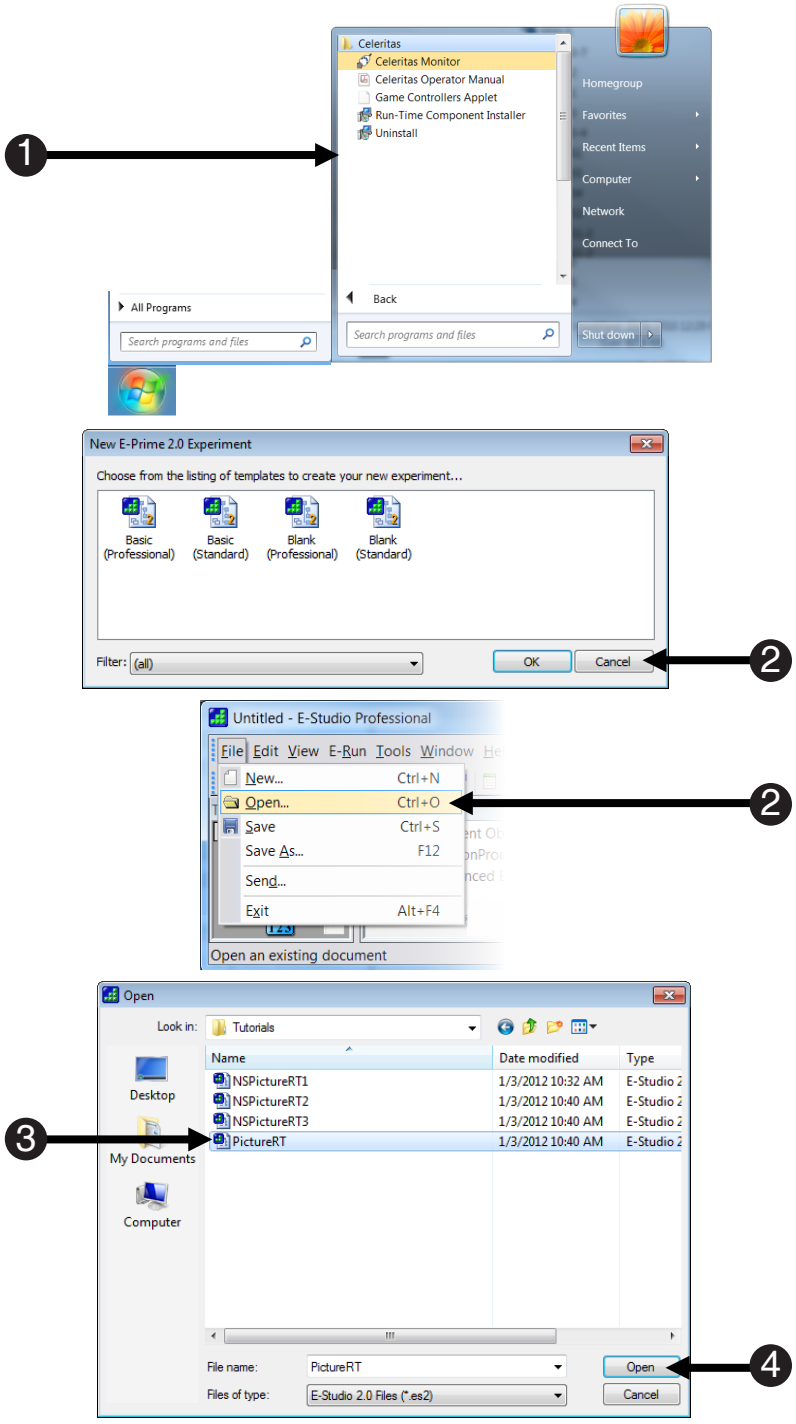
The E-Studio application is installed as part of the typical E-Prime installation. This application is used to create, modify, and test experiments within E-Prime. Open the E-Studio application, navigate to the \My Experiments\Samples\PictureRT, and load the PictureRT.es2 sample experiment.

1) ***Click*** on the Windows **Start** menu, ***select*** **Programs**, and then ***select*** **E-Prime**. From the menu, ***click*** on **E-Studio** to launch the application.

2) ***Click*** the **Cancel** button. ***Select*** **Open** from the **File** menu**.**

3) ***Navigate*** to the **"...Samples\ PictureRT"** folder to load the experiment.

4) ***Select*** the **"PictureRT.es2"** file and then ***click*** the **Open** button to load the experiment into E-Studio.



Task 2: Save the experiment under a new name

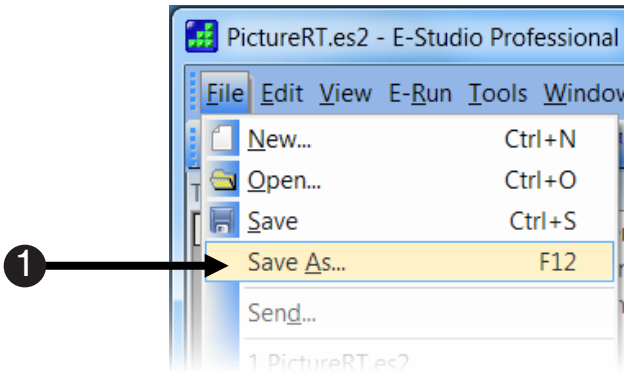
Save the PictureRT.es2 experiment in the same folder under the new name "NSPictureRT.es2".

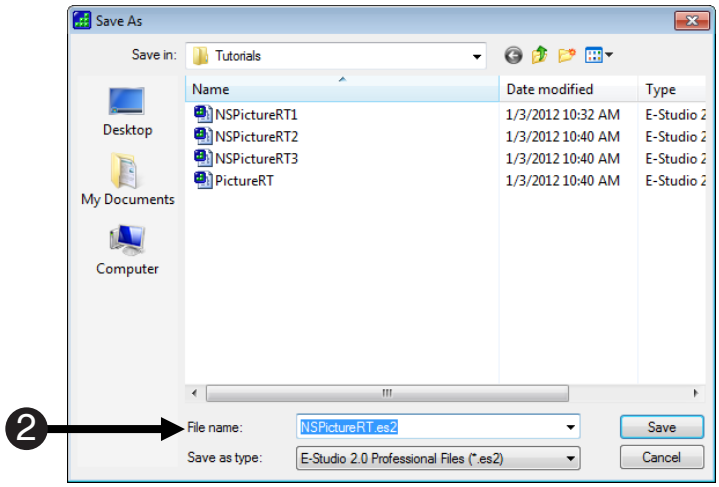
The PictureRT.es2 sample experiment requires bitmap (i.e. RedCar.bmp, BlueCar.bmp) resources that exist in the current folder. Rename the experiment but be sure to save it in the same folder ("...\Samples\PictureRT") so that the existing bitmap resource references within the experiment will remain valid and can be reused.

1) ***Select*** **Save As...** from the **File** menu**.**

2) ***Type*** "**NSPictureRT.es2**" as the new name in the **Filename** field.

3) ***Click*** the **Save** button.





Task 3: Add the Net Station Package File to the experiment

Open the Properties dialog for the Experiment Object and use the Packages tab to add the Net Station Package File to the experiment.

Package Files in E-Prime are cohesive sets of E-Basic routines that are grouped together. In order to gain access to the routines within a package file you must first add the package file to the experiment. Package files can be added to an experiment using the Packages tab of the Experiment Object Properties dialog. The routines that are used to communicate with the Net Station hardware and software at runtime are contained within the Net Station Package File.

1) ***Double-click*** the **Experiment** object at the top of the tree in the **Structure** window.

2) ***Click*** on the **Packages** tab of the **Experiment Object Properties** dialog.

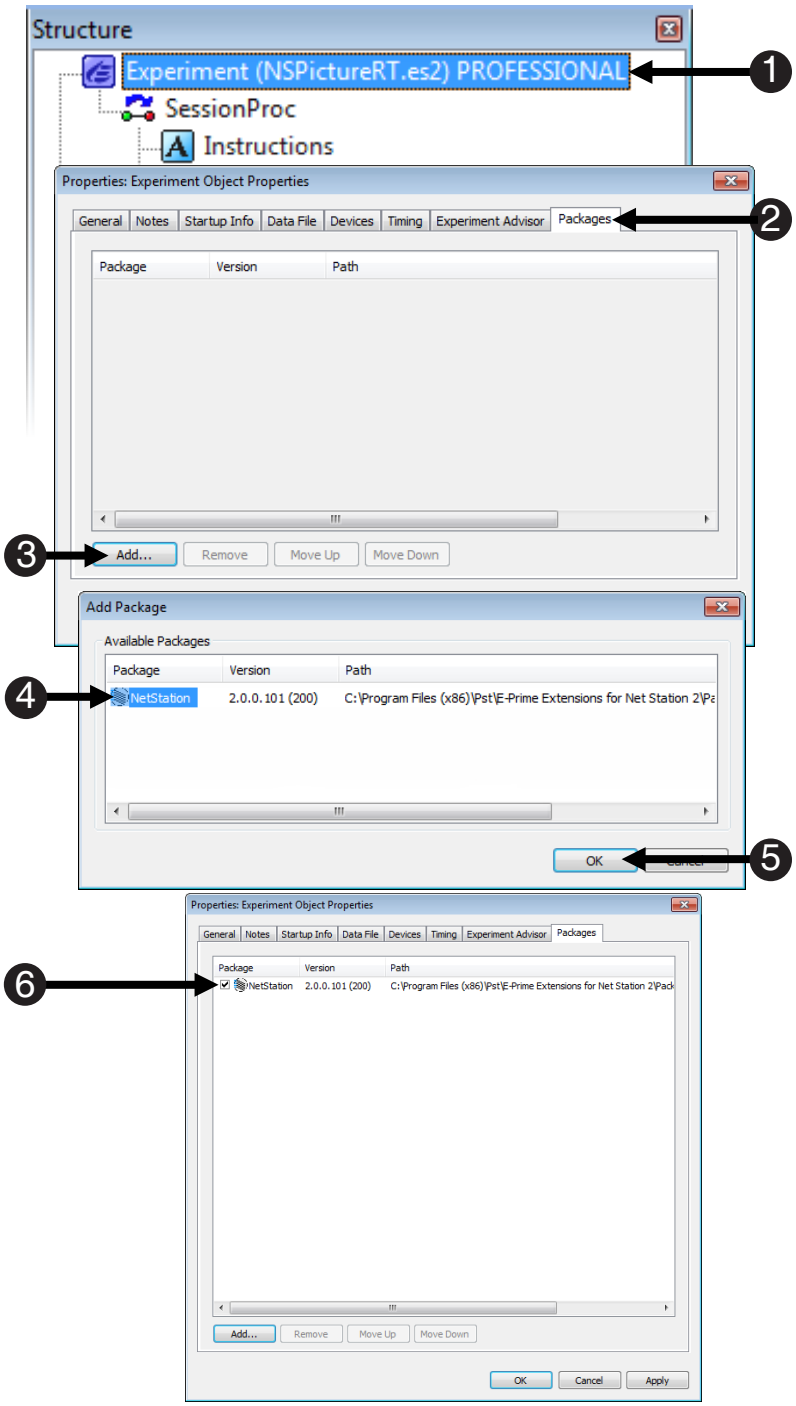
3) ***Click*** the **Add...** button.

4) ***Select*** the **Net Station** package file in the **Add Package** dialog.

5) ***Click*** the **OK** button to dismiss the dialog.

6) ***Verify*** the **Net Station** package file has been added and is checked (**<u>Do not</u>** yet *click* **OK** or dismiss the dialog).

⚠ **NOTE:** *The Package File version number that is displayed by E-Studio reflects the version of the Net Station Package File that is currently installed on your machine and may not match the picture.*



Task 4: Configure the experiment Startup Info

Configure the experiment to prompt for the Startup Info parameters that are required by Net Station.

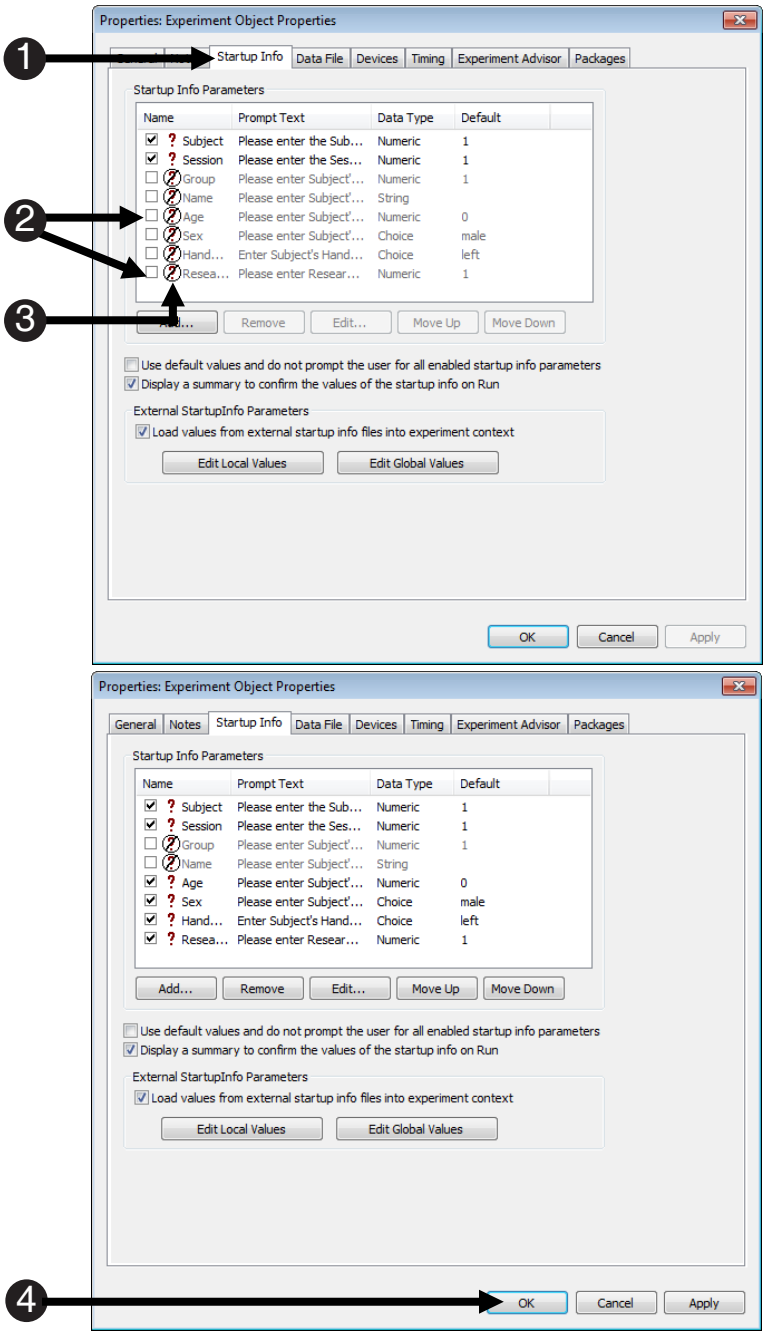
Net Station requires that experiments provide information related to Subject, Session, Age, Sex, Handedness, and ResearcherID. Failure to configure E-Prime to send these parameters will cause a runtime failure of the experiment. Subject and Session are already enabled in E-Prime by default, but all of the other parameters must be added to the experiment using the Startup Info tab of the Experiment Object Properties dialog. It is typical to prompt the researcher for this information at the beginning of the experiment. If you do not prompt for the information at runtime, you must set the Default value of each parameter to the value you want to use.

1) ***Click*** the **Startup Info** tab on the **Experiment Object Properties** dialog.

2) ***Click*** the **checkbox** associated with each of the **Age**, **Sex**, **Handedness**, and **ResearcherID** parameters to include these parameters in the experiment.

3) ***Click*** the  icon beside each of the same parameters to enable prompting for the parameter at runtime.

4) ***Click*** the **OK** button to accept the changes and dismiss the dialog.



Task 5: Add a Net Station PackageCall to initialize the system

Add a PackageCall at the beginning of the SessionProc to initialize the system for use with Net Station. Name the object "NSInit".

The Net Station package file must be initialized at the start of the experiment by making a call to the NetStation_Init routine. To call a routine in a package file you can either make the call directly using E-Basic script within an Inline object, or (more preferably) drag a PackageCall object from the E-Studio Toolbox, drop it at the desired location within the experiment, and edit its properties. When using the PackageCall object it is strongly recommended that you rename the object to reflect the specific package file and routine that is being referenced within the object. When creating experiments for use with Net Station it is a common convention to create a name for each PackageCall object by abbreviating Net Station as "NS" and then concatenating the name of the routine being called, e.g. the object which is used in this task to call the "NetStation_Init" routine is named "NSInit".

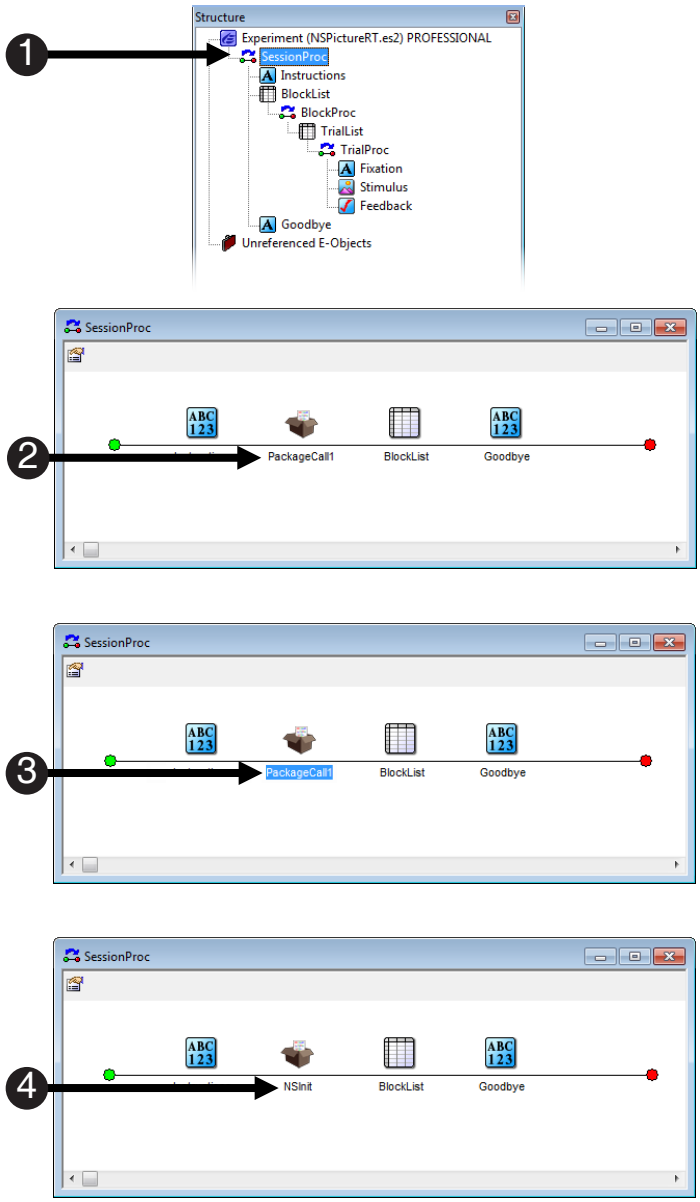
1) ***Double-click*** the **SessionProc** object to open it in the workspace.

2) ***Drag*** a new **PackageCall** object from the **Toolbox** and ***drop*** it as the first object in the **SessionProc** procedure. The object will be given a default name of **PackageCall1**.

3) ***Click*** on the **PackageCall1** object to select it then ***press*** **F2** to rename the object.

⚠ **NOTE:** *You may alternatively right click on the object and select Rename from the context menu.*

4) ***Type*** "**NSInit**" as the new object name and then ***press*** **Enter** to accept the change.



Task 5 (continued): Add a Net Station PackageCall to initialize the system

Configure the NSInit PackageCall object to call the Init routine of the Net Station package file.

The Properties dialog of the PackageCall object to specifies which Package File and Routine are to be called in each instance. After you select a Package within the interface, the Routine dropdown list will be populated with all of the routines contained within the package that you can call via a PackageCall object. After you select a Routine, the Parameters field will be set to the default parameters and the Description field will be filled with the text that the package file author included for the selected routine. You must edit the parameters to specify socket support and the IP address that you are currently using. You can refer to the Description field for information about each parameter in the list (any parameter in double quotes indicates string data). When the NSInit call is made in the experiment, it will cause Net Station to start recording for a brief period during which SESS and CELL events are being sent; containing global information about the experiment. In some cases, a file without recording breaks is preferable. To have E-Prime continue recording after the NSInit call, add the optional boolean parameter as described in the Details text box.

5) ***Double-click*** the **NSInit PackageCall** object within the **SessionProc** to display its **Properties** dialog.

6) ***Select*** **Net Station** from the **Package** dropdown list.

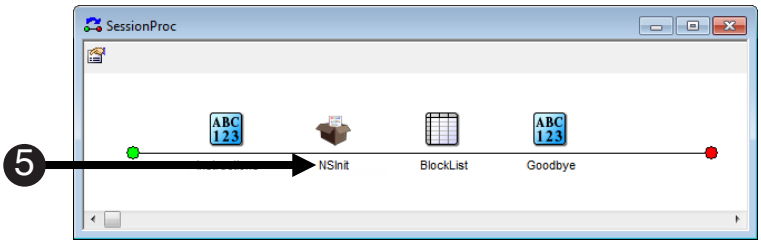
7) ***Select*** **Init** from the **Routine** dropdown list.

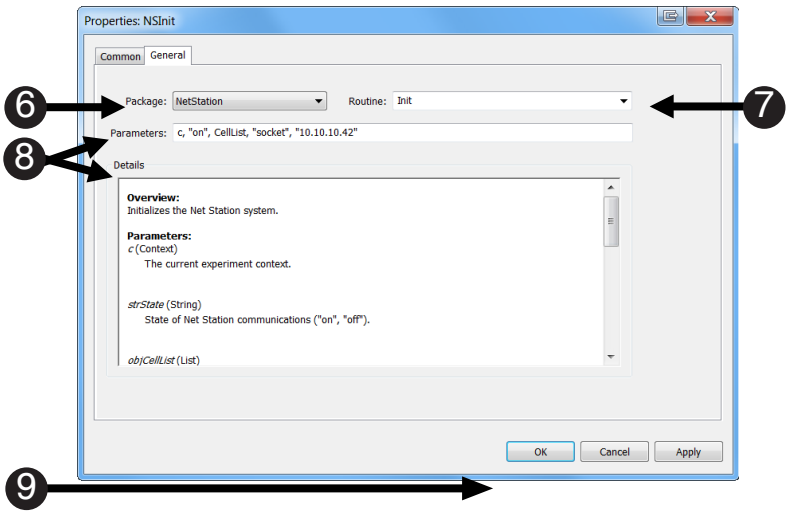
8) ***Review*** the parameters listed in the **Parameters** and **Details** fields. ***Update*** the Parameters to specify "**socket**" communications and correct IP address.

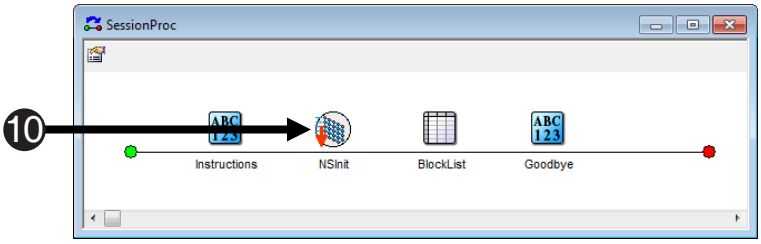
⚠ **NOTE:** *The recommended parameters are: c, "on", CellList, "socket", "10.10.10.42"*

9) ***Click*** the **OK** button to accept the changes and dismiss the dialog.

10) If the **Package file *defines*** an **icon** for the routine it will ***replace*** the **default PackageCall icon**, otherwise the icon associated with the entire package file itself will be used.







Task 6: Add a Net Station PackageCall to uninitialize the system

Add a PackageCall at the end of the SessionProc to uninitialize the system by calling the Uninit routine within the Net Station package file. Rename the object "NSUninit".

At the end of a Net Station, enabled experiment the system should be uninitialized. The NetStation_Uninit call will send a message to Net Station to indicate that the experiment is now complete. This call is typically placed at the end of your SessionProc. After you have placed and configured the object notice that the icons for the NSInit and NSUninit objects are a matched pair, e.g. they each show a small arrow pointing in opposite directions that is intended to indicate the start and end of an experiment level.

1) ***Drag*** a new **PackageCall** object from the **Toolbox** and ***drop*** it as the last object in the **SessionProc** procedure.

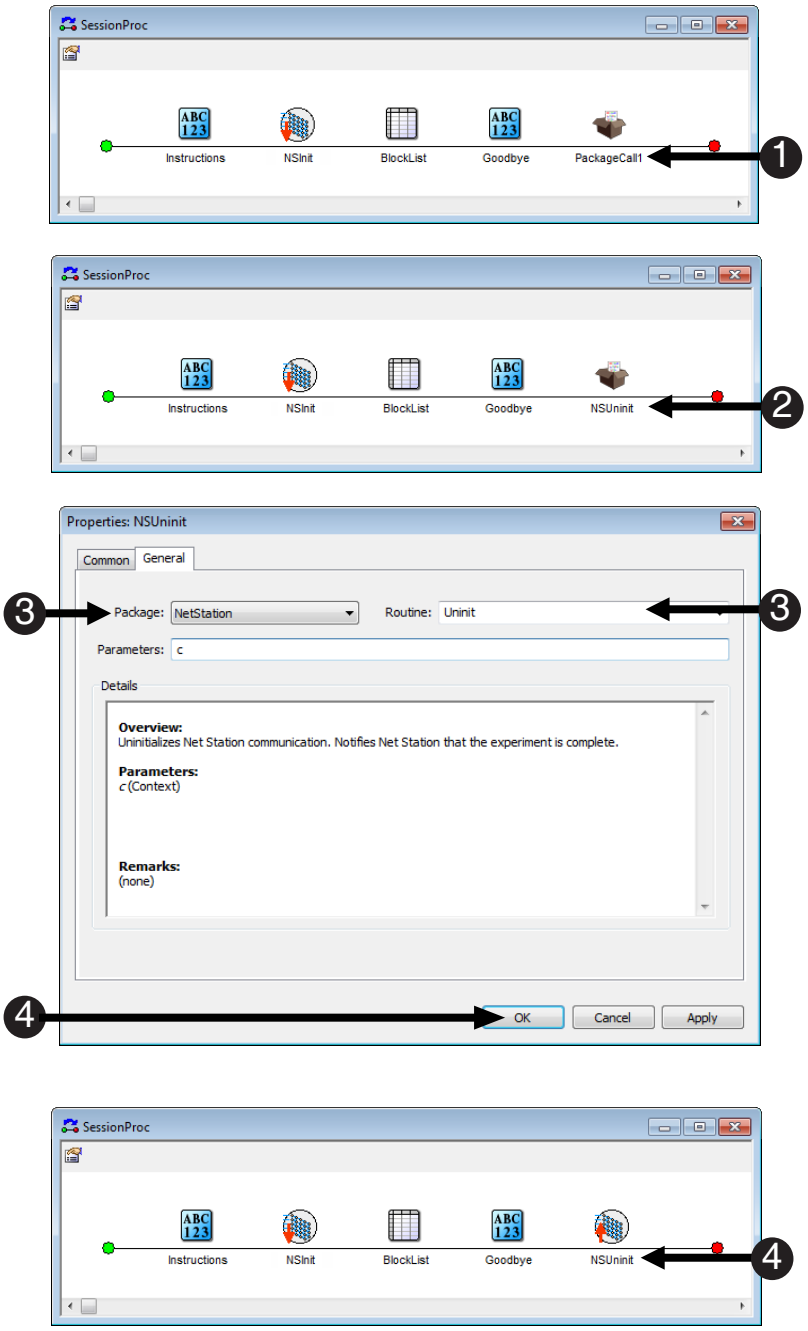
2) ***Click*** on the **PackageCall1** object to select it then ***press*** **F2** to rename the object. ***Type*** "**NSUninit**" as the new object name and then ***press*** **Enter** to accept the change.

3) ***Double-click*** the **NSUninit** object to open its **Properties** dialog. Set the **Package** field to **Net Station** and the **Routine** field to Uninit.

4) ***Click*** the **OK** button to accept the changes and ***verify*** the object has been added to the end of the **SessionProc**.

⚠ **NOTE:** *The icons for the NSInit and NSUninit objects are a matched pair.*

⚠ **NOTE:** *Calling NSUninit will shut down the recording session in Net Station. To delay this recording shut down, place an E-Object (such as a TextDisplay object, that is displayed until dismissed by a keystroke) so that the Net Station file will not be closed until the object is dismissed.*



Task 7: Add a Net Station PackageCall to start recording

Add a PackageCall at the beginning of the BlockProc to direct Net Station to start recording. Name the object "NSStartRecording".

When you want Net Station to begin, recording you must direct it to do so by making a call to the NetStation_StartRecording routine. You may start and stop recording as many times as you would like within your experiment. Typically, you will start recording at the beginning of a block of trials and stop recording at the end of the block. Open the BlockProc Procedure and add a new PackageCall object at the beginning of the Procedure (i.e. immediately before the TrialList object). Rename this object as "NSStartRecording" and set its properties so that it calls the NetStation_StartRecording routine.

1) ***Double-click*** the **BlockProc** object to open it in the workspace.

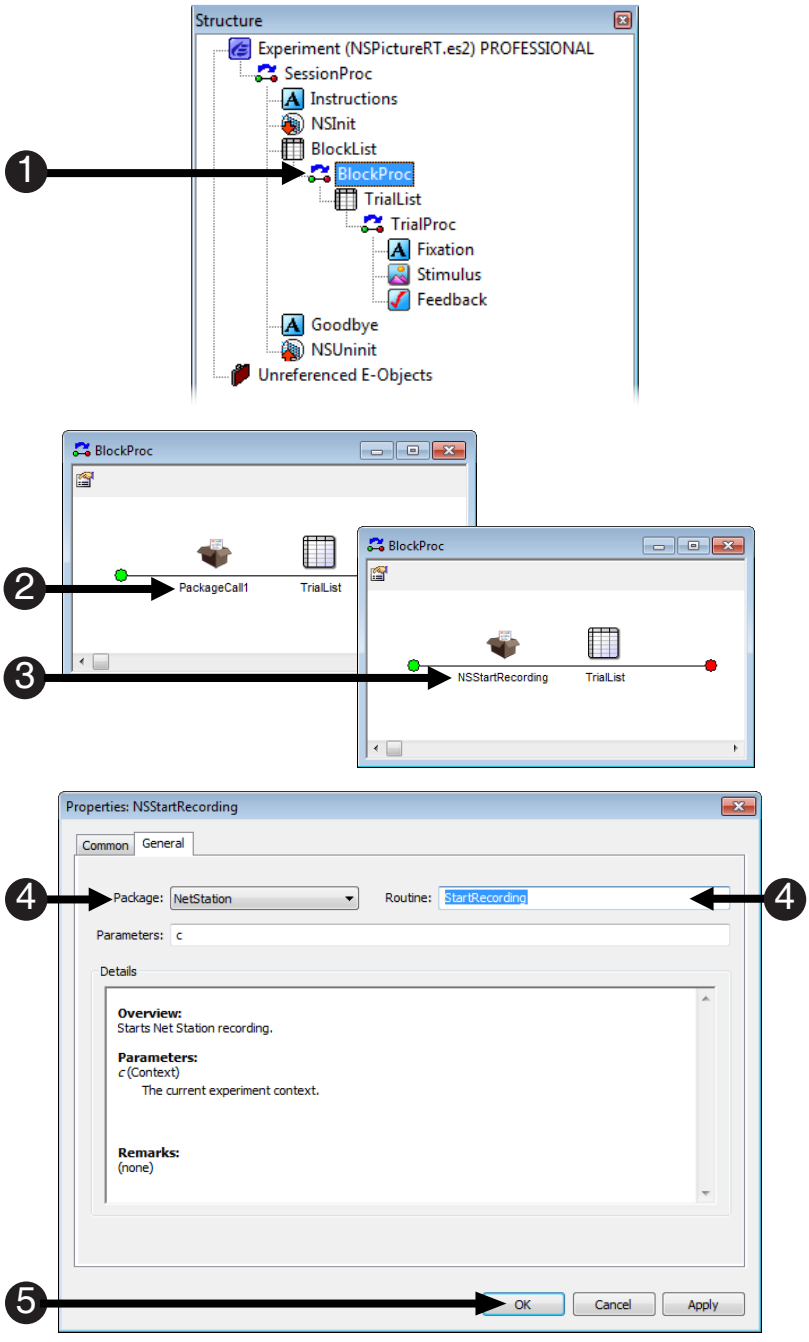
2) ***Drag*** a new **PackageCall object** from the **Toolbox** and ***drop*** it as the first object in the **BlockProc** procedure.

3) ***Click*** on the new object to select it then ***press*** **F2** and ***rename*** the object as "**NSStartRecording**".

4) ***Double-click*** on the **NSStartRecording** object to open its dialog then set the **Package** field to **Net Station** and the **Routine** field to **StartRecording**.

5) ***Click*** the **OK** button to accept the changes and dismiss the dialog.

⚠ **NOTE:** *Due to the processing time required to start and stop recording within Net Station, starting and stopping recording for each trial is not recommended.*



Task 8: Add a Net Station PackageCall to stop recording

Add a PackageCall at the beginning of the BlockProc to direct Net Station to start recording. Name the object "NSStartRecording".

When you want, Net Station to stop recording you must direct it to do so by making a call to the NetStation_StopRecording routine. Add a new PackageCall object at the end of the BlockProc Procedure (i.e. immediately after the TrialList object). Rename this object as "NSStopRecording" and set its properties so that it calls the NetStation_StopRecording routine.

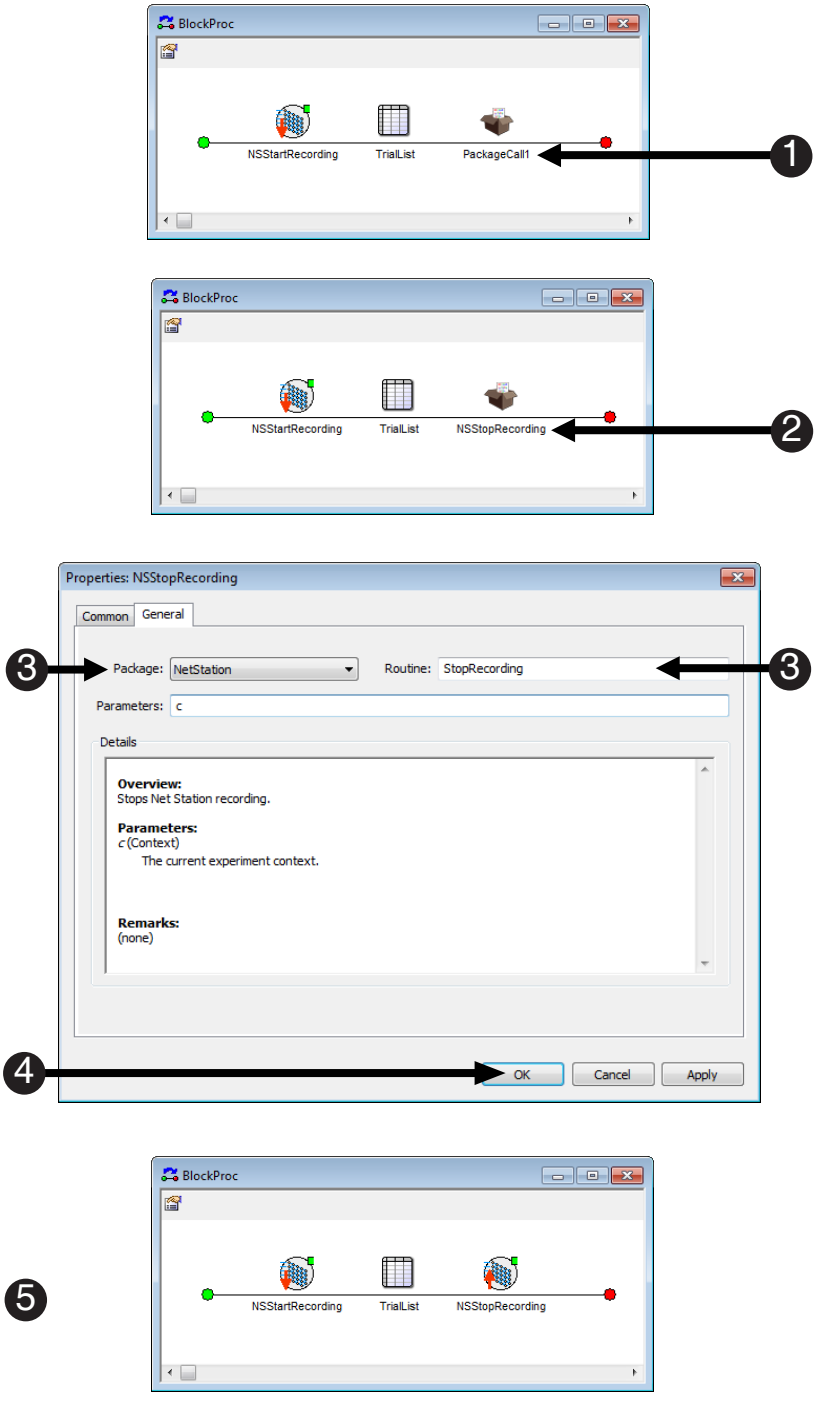
1) ***Drag*** a new **PackageCall** object from the **Toolbox** and ***drop*** it as the last object in the **BlockProc** procedure.

2) ***Click*** on the new object to select it then ***press*** **F2** and ***rename*** the object as "**NSStopRecording**".

3) ***Double-click*** on the **NSStopRecording** object to open its **Properties** dialog then set the **Package** field to **Net Station** and the **Routine** field to **StopRecording**.

4) ***Click*** the **OK** button to accept the changes and dismiss the dialog.

5) ***Verify*** that your **BlockProc** Procedure matches the picture.



Task 9: Add a Net Station PackageCall to initialize the trial

Add a PackageCall at the beginning of the TrialProc to initialize each trial. Name the object "NSTrialBegin".

Net Station must be notified of the start of each trial. Open the TrialProc Procedure and add a new PackageCall object at the beginning of the Procedure (i.e. immediately before the Fixation object). Rename this object as "NSTrialBegin" and set its properties so that it calls the NetStation_TrialBegin routine.

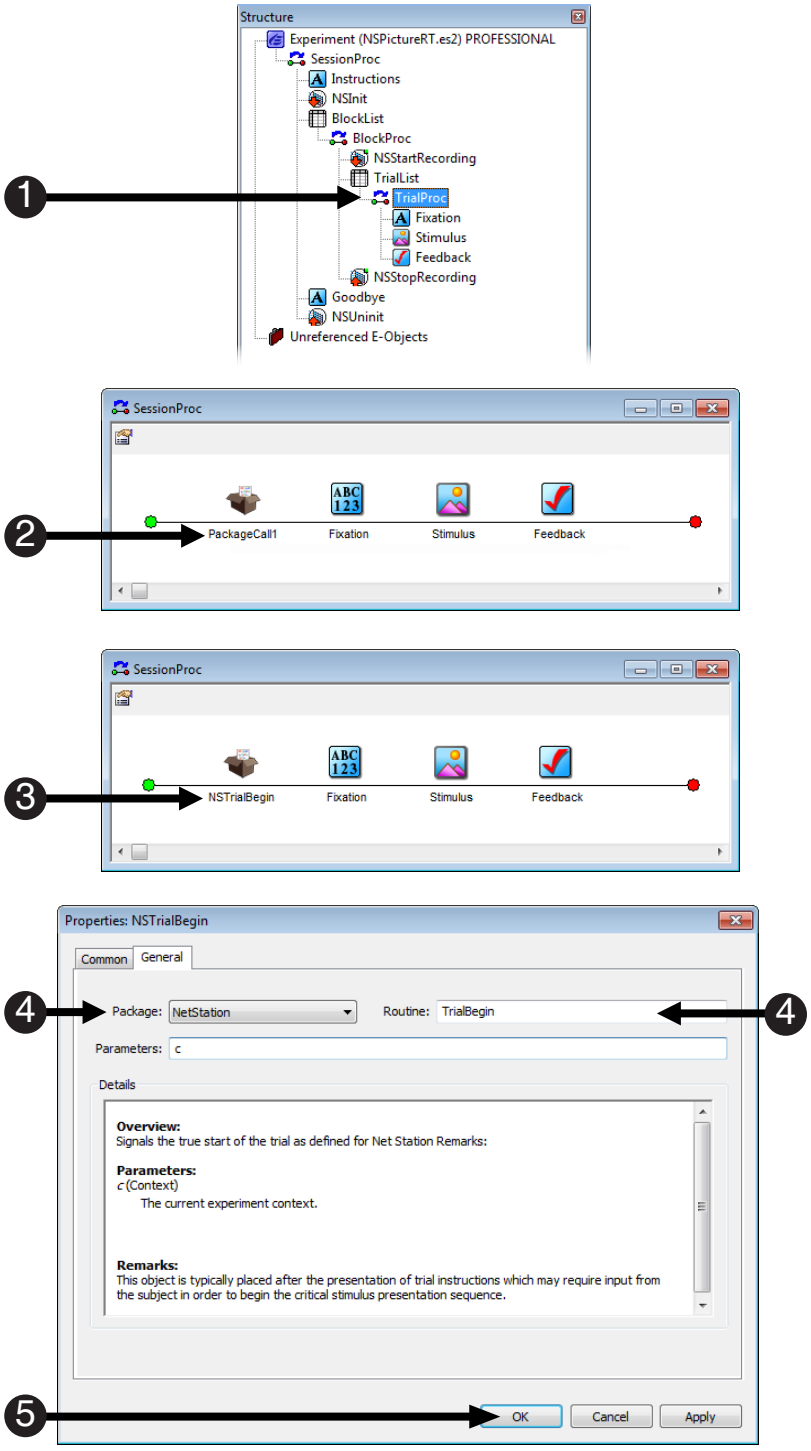
1) ***Double-click*** the **TrialProc** object to open it in the workspace.

2) ***Drag*** a new **PackageCall** object from the **Toolbox** and ***drop*** it as the first object in the **TrialProc** procedure.

3) ***Click*** on the new object to select it then ***press*** **F2** and ***rename*** the object as "**NSTrialBegin**".

4) ***Double-click*** on the **NSTrialBegin** object to open its **Properties** dialog then set the **Package field** to **Net Station** and the **Routine** field to **TrialBegin**.

5) ***Click*** the **OK** button to ***accept*** the changes and ***dismiss*** the dialog.



Task 10: Add a Net Station PackageCall to uninitialize the trial

Add a PackageCall at the end of the TrialProc to uninitialize each trial. Name the object "NSTrialEnd".

Although not required, Net Station is typically notified of the end of each trial. Add a new PackageCall object at the end of the Procedure (i.e. immediately after the Feedback object). Rename this object as "NSTrialEnd" and set its properties so that it calls the NetStation_TrialEnd routine.

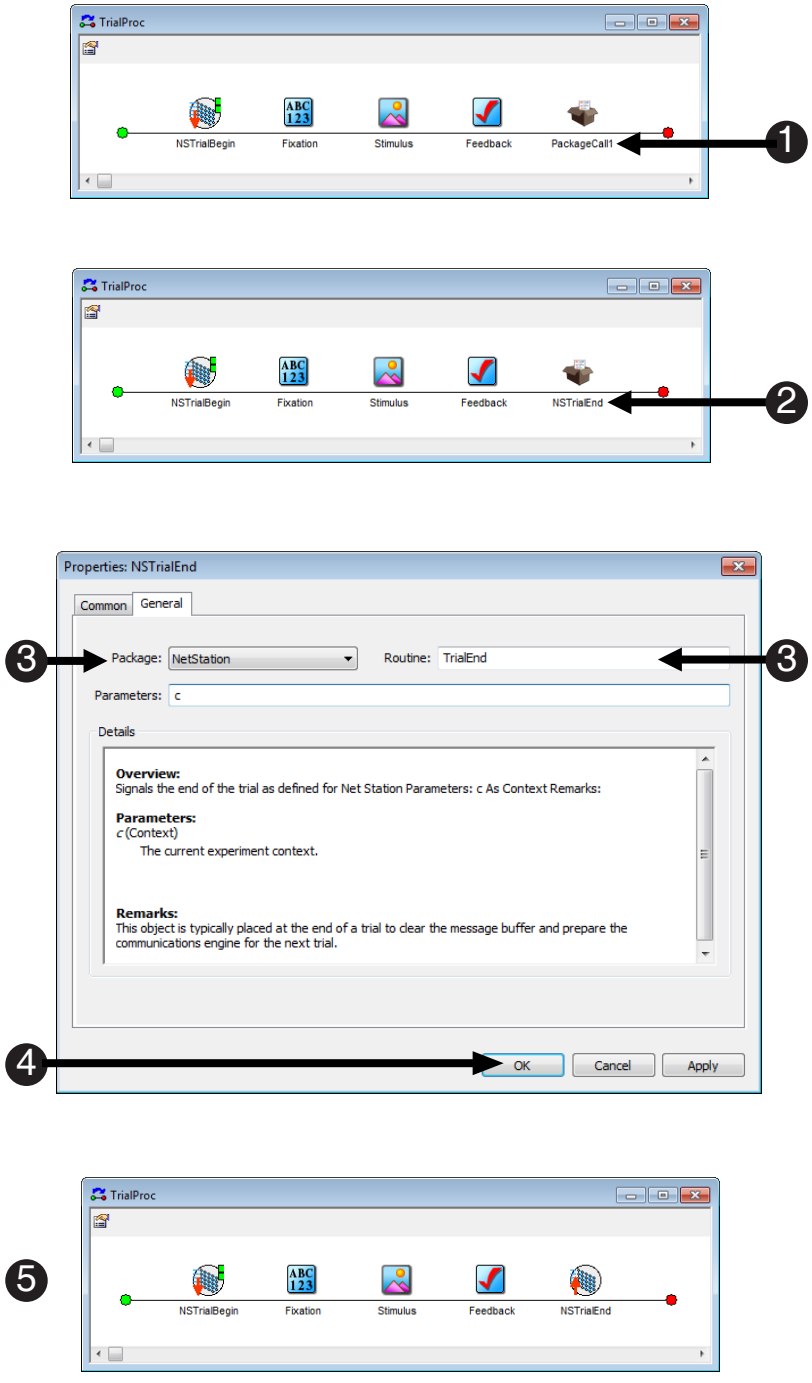
1) ***Drag*** a new **PackageCall** object from the **Toolbox** and ***drop*** it as the last object in the **TrialProc** procedure.

2) ***Click*** on the new object to select it then ***press*** **F2** and ***rename*** the object as "**NSTrialEnd**".

3) ***Double-click*** on the **NSTrialEnd** object to open its **Properties** dialog then set the **Package** field to **Net Station** and the **Routine** field to **TrialEnd**.

4) ***Click*** the **OK** button to ***accept*** the changes and ***dismiss*** the dialog.

5) ***Verify*** that your **TrialProc** Procedure matches the picture.



Task 11: Add script to send trial events to Net Station

Add Inline objects after the Feedback object. Name the object "NSSendTrialEvents". Add script to the Inline to make direct PackageCalls to send the required trial event information to Net Station.

Net Station requires that it be notified of all critical trial events (e.g. stimulus events presented or participant responses that were accepted during the trial). While you may use a series of PackageCall objects to complete this task, it is often easier to just use a single Inline object and a few lines of E-Basic script to make a series of direct calls to the required package routines. Add a new Inline object after the Feedback object on the TrialProc. Rename this object as "NSSendTrialEvents" and open the object in the workspace so that you can type in the required script.

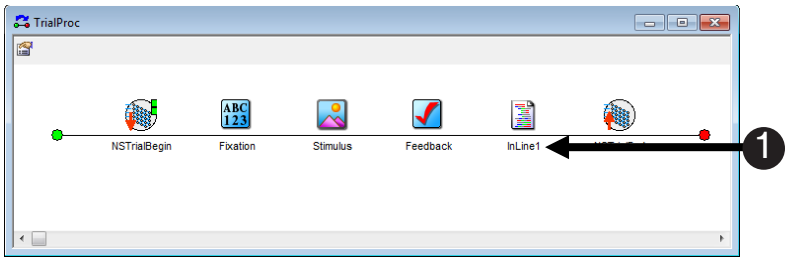
Net Station requires that it be notified of any participant response that was collected during each trial. You can do this by making a direct call to the NetStation_SendRespEvent routine for each response collected during the trial. This routine requires that you pass in a reference to the object within the trial sequence that collected the response (e.g. the Stimulus object in this example). It is important to make sure that the allowable response interval within each trial has completed before making this call (e.g. if you are using PreRelease on objects within your trial sequence and a participant responds near the very end of the response interval). You could potentially send invalid or incomplete response data to Net Station (see Chapter 4: Critical Timing, *E-Prime User's Guide*, for details on how to avoid this problem.)

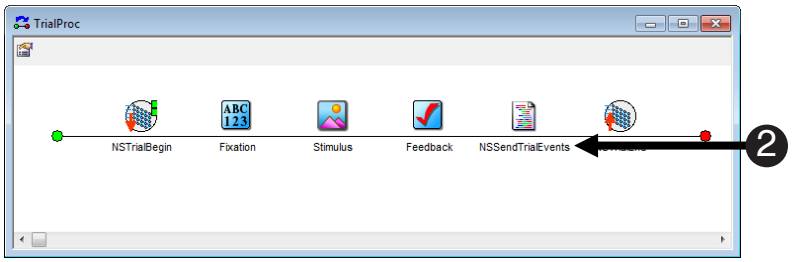
1) ***Drag*** an **InLine** object from the **Toolbox** and ***drop*** it after the **Feedback** object in the **TrialProc** procedure.

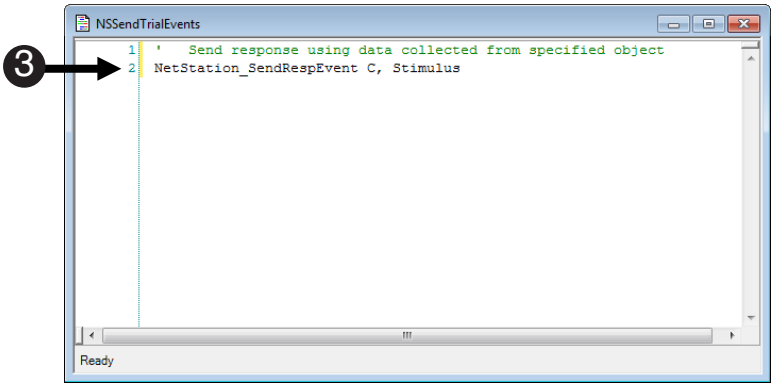
2) ***Click*** on the new object to select it then ***press*** **F2** and ***rename*** the object as "**NSSendTrialEvents**".

3) ***Double-click*** on the **NSSendTrialEvents** object to ***open*** it in the workspace and then ***type*** in the **script** exactly as shown to add a direct call to the **NetStation_SendRespEvent** routine.

⚠ **NOTE:** *It is recommended that you always add a comment line to document each direct package file call.*







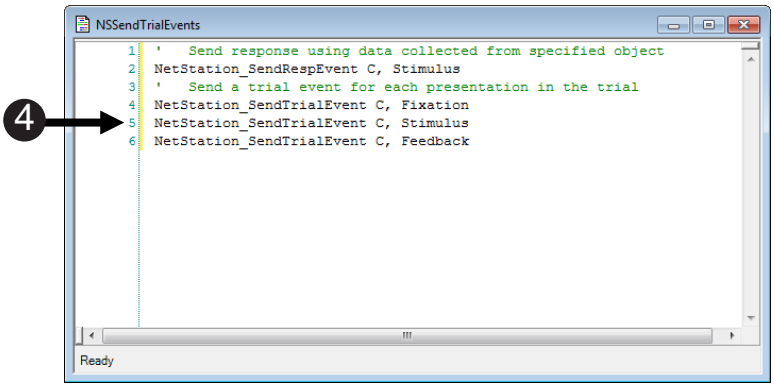
Task 11 (continued): Add script to send trial events to Net Station

Add script to the Inline to make direct PackageCalls to send the required trial event information to Net Station.

Prior to the end of each trial, Net Station requires that it be sent all timing information related to any critical stimulus presentation that occurred within the trial. Append additional script to the NSSendTrialEvents Inline object to make a direct call to the NetStation_SendTrialEvent routine for each object of interest in the trial sequence of your experiment (e.g. in this experiment we are interested in the Fixation, Stimulus, and Feedback objects because they directly alter the visual display).

Note the syntax of the NetStation_SendTrialEvent subroutine call. Its required parameters are the current experiment Context (c) and a reference to the object that generated the current event.

4) ***Type*** in the **script** exactly as shown to add a call to the **NetStation_SendRespEvent** routine.



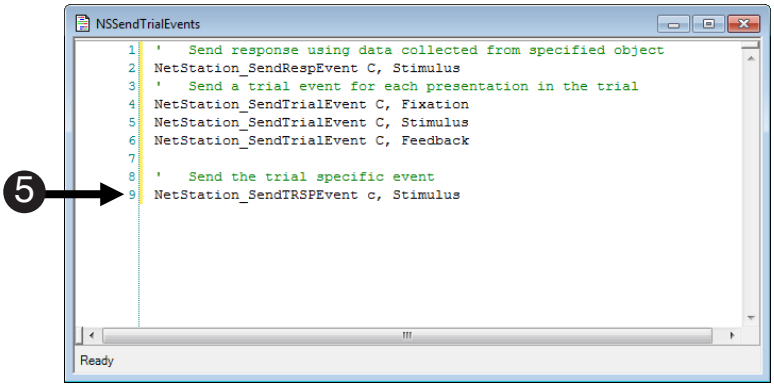
Task 11 (continued): Add script to send trial events to Net Station

Add script to the Inline to make direct PackageCalls to send the required trial event information to Net Station.

Net Station requires that a TRial SPecific (TRSP) event be sent prior to the end of each trial so that it can be notified of behavioral data related to the critical stimulus and response. Append an additional line of script to the NSSendTrialEvents Inline object to make a direct call to the NetStation_ SendTRSPEvent routine.

Note the syntax of the NetStation_SendTRSPEvent subroutine call. Its required parameters are the current experiment Context (c) and a reference to the object that collected the participant's response (e.g. the Stimulus object in this example). The routine also includes an additional optional parameter (KeyList) that is not currently used in this example. The optional KeyList parameter may be used to specify a reference to a user defined List object within the experiment that has been configured to describe additional key data that should be associated with the current trial. If the KeyList parameter is not specified then no additional key data will be sent to Net Station as part of the TRSP event. Adding user defined keys using a KeyList will be covered later in Tutorial 2.

5) ***Type*** in the **script** exactly as shown to add a call to the **NetStation_SendTRSPEvent** routine.



Task 12: Assign a unique four character code to each object of interest

Assign a unique four-character code to each stimulus presentation object or event of interest within the trial sequence. Set the Tag property of the Fixation object to "fix+".

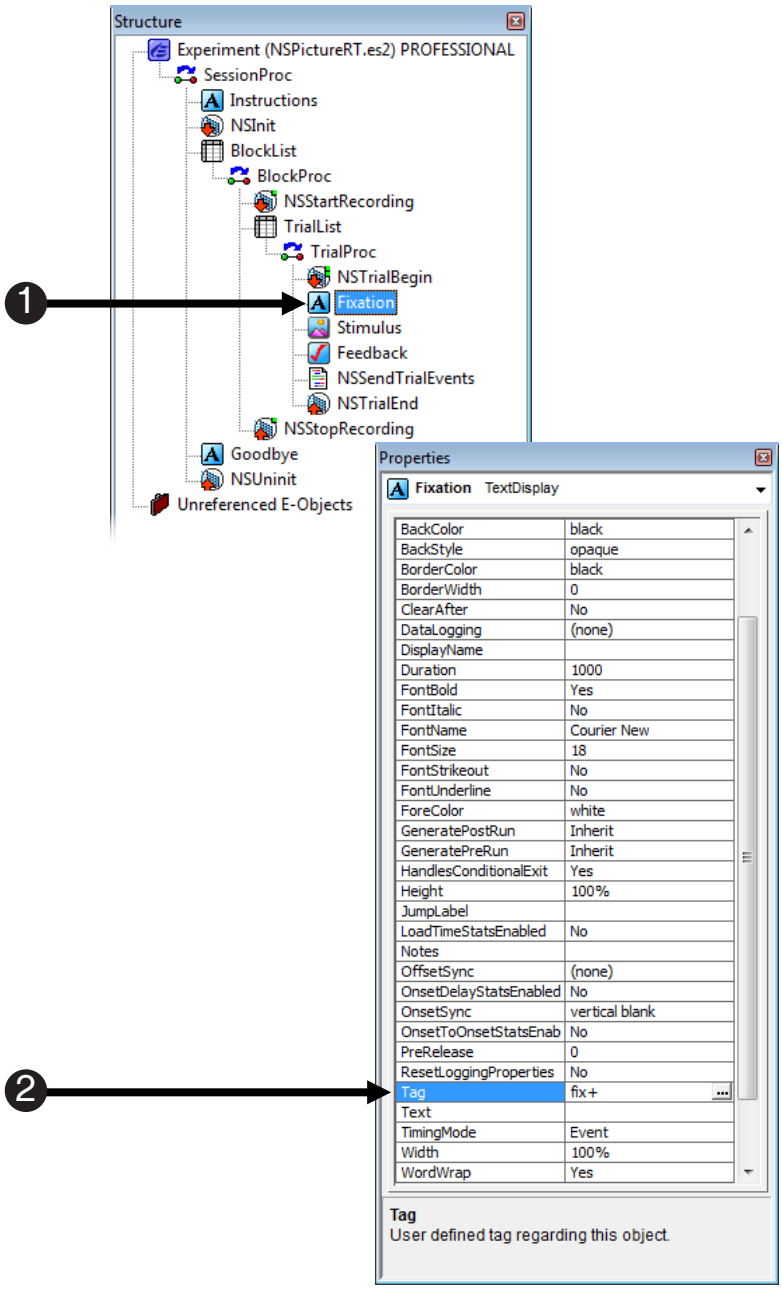
Net Station requires that each event of interest within a trial sequence be identified via a unique four-character code. Within a Net Station enabled experiment, you assign these unique codes to objects by setting each object's "Tag" property. At runtime, routines within the Net Station Package File will examine the assigned Tag property of an object in order to associate data sent to Net Station with a particular object. In this example experiment, you need to set the Tag property of the Fixation, Stimulus, and Feedback objects to "fix+", "stm+", and "stm-" respectively. Note that the editing of an object's Tag property can only be done through the Properties window within E-Studio so you should verify that window is currently viewable before beginning this task.

1) ***Click*** on the **Fixation** object in the **Structure** window to select it and display its properties in the **Properties** window.

⚠ **NOTE:** *To view the Properties window choose View > Properties from the application menu bar.*

2) ***Click*** in the **edit box** beside the **Tag** property and ***type*** "**fix+**" then ***press*** the **Enter** key to accept the change.

⚠ **NOTE:** *Assigned codes must be exactly four characters in length or an error will be generated when the experiment is run with Net Station.*



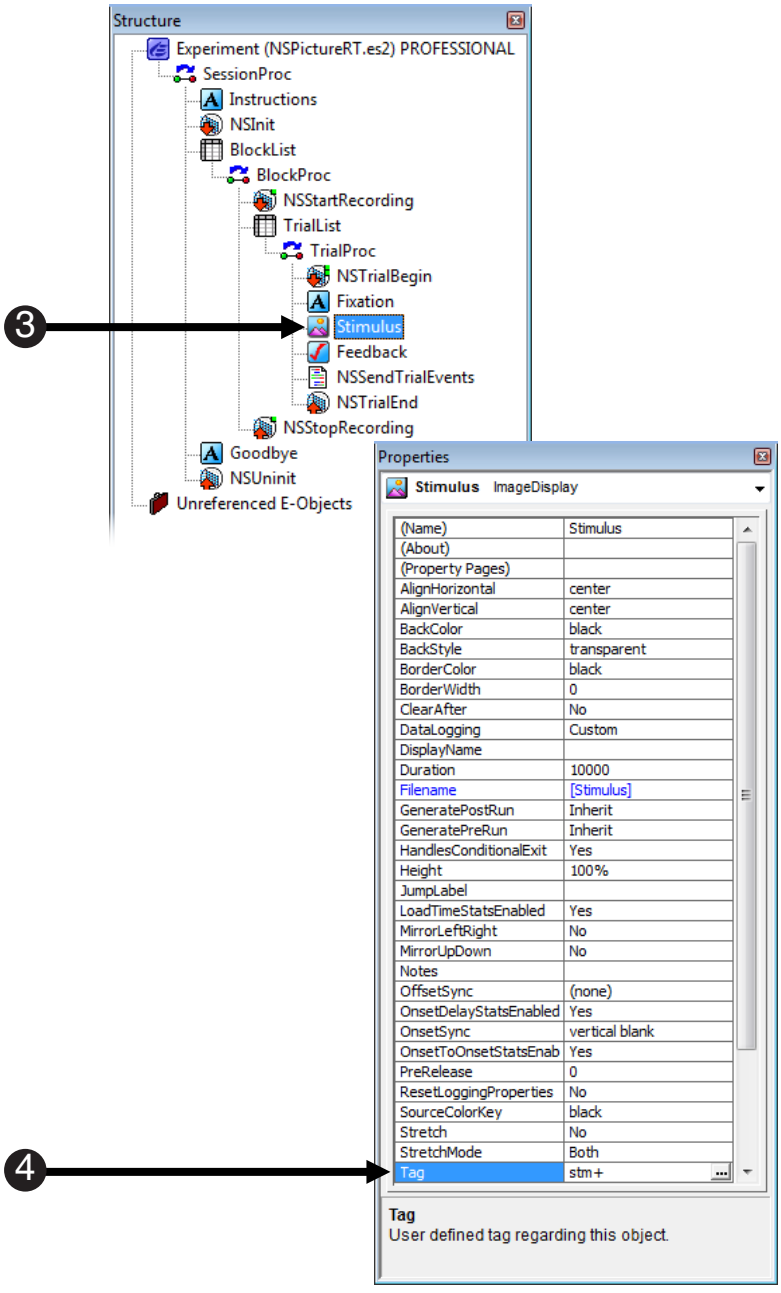
Task 12 (continued): Assign a unique four character code to each object of interest

Set the Tag property of the Stimulus object to "stm+".

In this example, the Stimulus object is responsible for presenting the critical stimulus and for enabling and accepting a response from the participant. Select the Stimulus object and set its Tag property to the four-character code of "stm+" to indicate that it will be the object that presents the critical stimulus.

3) ***Click*** on the **Stimulus** object in the **Structure** window to select it and display its properties in the **Properties Window**.

4) ***Click*** in the **edit box** beside the **Tag** property and ***type*** "**stm+**" then ***press*** the **Enter** key to accept the change.



Task 12 (continued): Assign a unique four character code to each object of interest

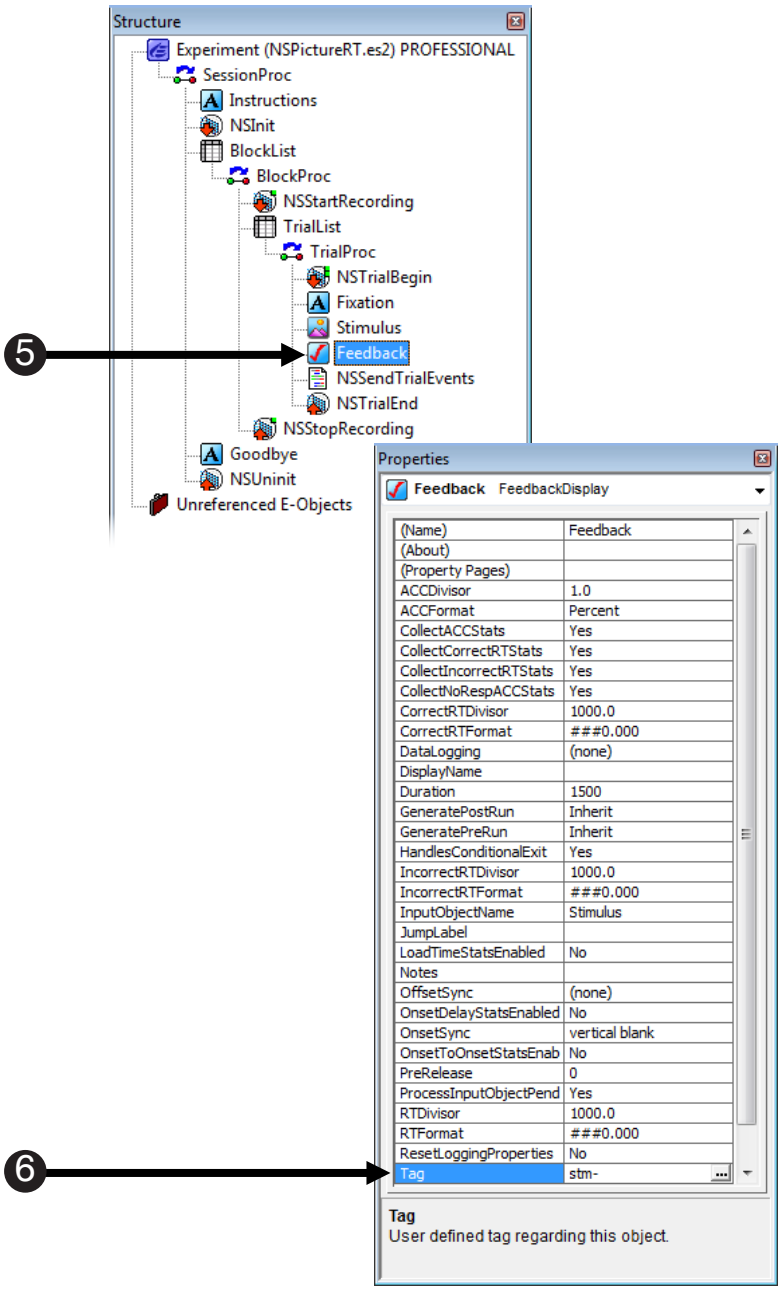
Set the Tag property of the Feedback object to "stm-".

In this example the Feedback object is configured to run after the participant has responded (or alternatively after the Stimulus object has timed out). When the Feedback object runs it clears the screen and thus removes/terminates the display of the critical stimulus. Select the Feedback object and set its Tag property to the four-character code of "stm-" to indicate that it is the object that signals the end of the critical stimulus.

5) ***Click*** on the **Feedback** object in the **Structure** window to select it and display its properties in the **Properties Window**.

6) ***Click*** in the **edit box** beside the **Tag** property and ***type*** "**stm-**" then ***press*** the **Enter** key to accept the change.

⚠ **NOTE:** *It is a common convention in Net Station enabled experiments to append a "-" to the end of an object's assigned tag to indicate the end/offset of an event.*



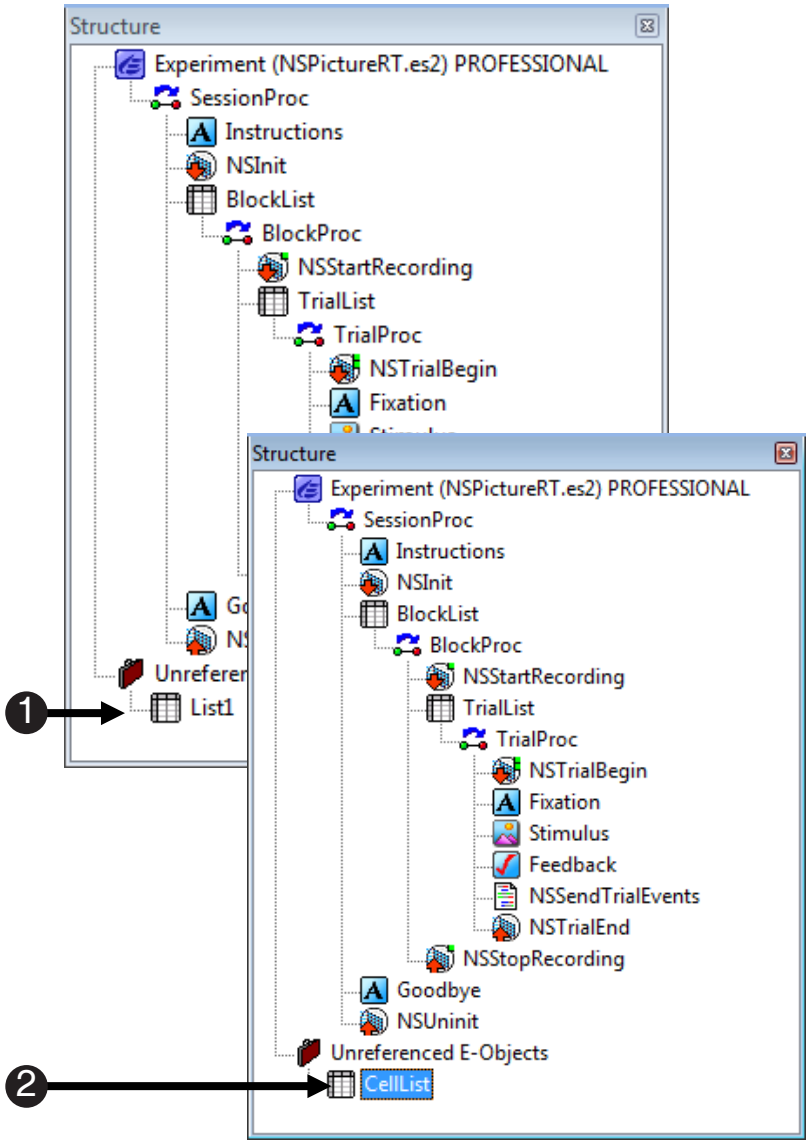
Task 13: Add a CellList object

Add a new List object to the Unreferenced E-Objects branch of the tree. Name the object "CellList".

Net Station identifies conditions or "cells" within an experiment by referencing user assigned Cell Numbers and Cell Labels. The routines within the Net Station Package File lookup information for these cells by examining the attributes of a named Cell List contained within the experiment. A Cell List is just a List object that has been created in the experiment and configured with a specified set of named attributes. Create a new List object in the Unreferenced E-Objects branch of the Structure window and rename it as "CellList".

1) ***Drag*** a new **List** object from the **Toolbox** and ***drop*** it on the **Unreferenced E-Object** branch of the **Structure** window.

2) ***Click*** on the new **List1** object to select it then ***press*** **F2** and ***rename*** the object as "**CellList**".



Task 14: Add the required CellNumber and CellLabel attributes to the CellList

Open the CellList object in the workspace and add two new attributes and one new level.

The CellList must minimally contain two named attributes – “CellNumber” and “CellLabel”. Open the CellList object in the workspace and add two new attributes. Since there are two cells defined and used in this experiment, you must also add an additional level/row to the CellList.

1) ***Double-click*** the **CellList** object to open it in the workspace.

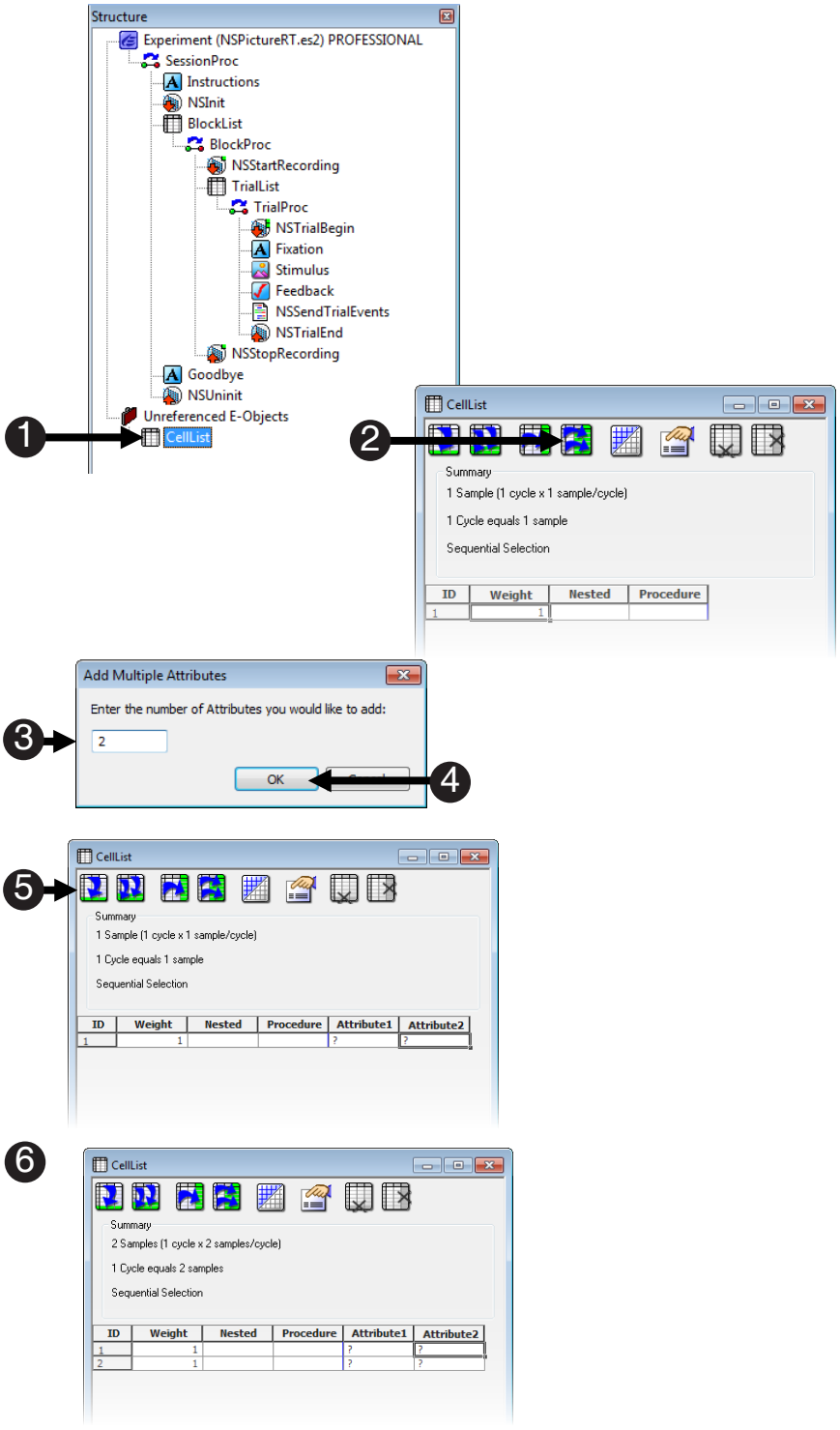
2) ***Click*** the **Add Multiple Attributes** button.

3) ***Type*** “**2**” in the **Add Multiple Attributes** dialog.

4) ***Click*** the **OK** button to add two new attributes to the **CellList**.

5) ***Click*** the **Add Level** button to add an additional level/row to the **CellList**.

6) ***Verify*** that your **CellList** matches the picture.



Task 14 (continued): Add the required CellNumber and CellLabel attributes to the CellList

Rename the new attributes to CellNumber and CellLabel then edit the attribute values to define the cells used within the experiment.

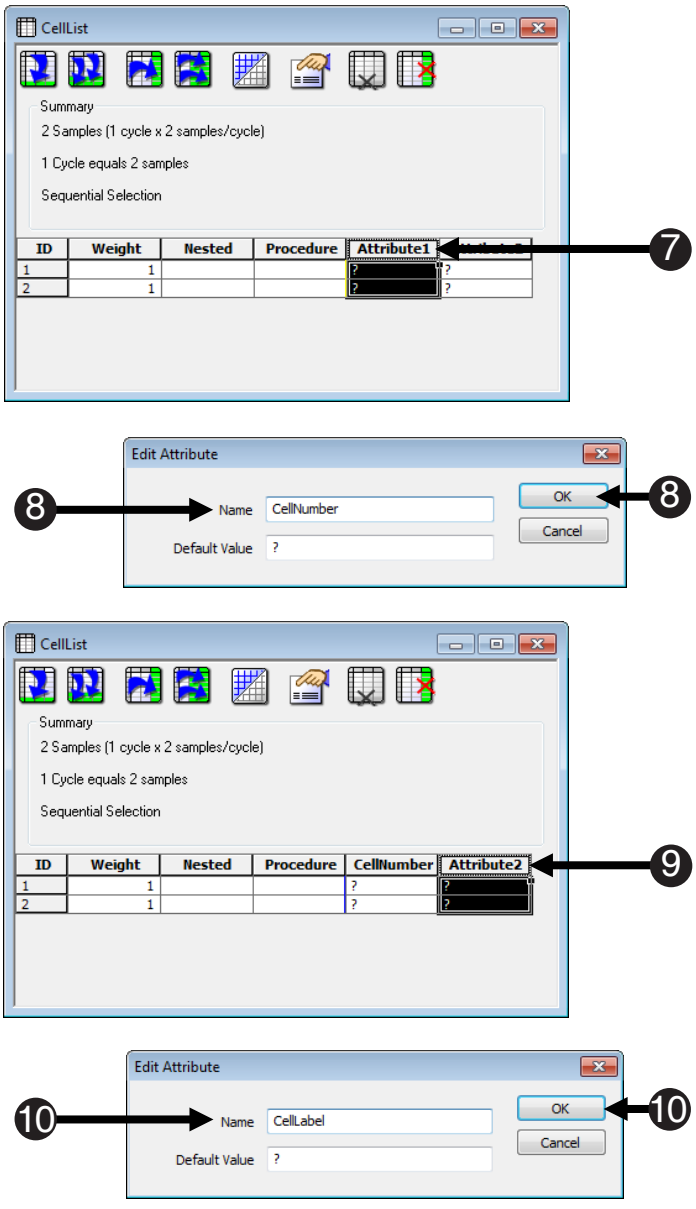
The routines within the Net Station Package File expect the CellList to contain attributes named "CellNumber" and "CellLabel". Rename each attribute respectively. If you misspell or forget to include one of these attributes, the Net Station Package File will generate an error when you attempt to run the experiment. Values for the CellNumber attribute must be numeric and must be unique within the experiment. You may currently define up to 255 unique cells within an experiment. Values for the CellLabel can be any string you choose to identify and describe the cell (i.e. CellLabels do not have to be four character codes).

7) ***Double-click*** the header of the **Attribute1** column to display the **Edit Attribute** dialog.

8) ***Type*** "**CellNumber**" in the **Name** field of the **Edit Attribute** dialog and ***click*** the **OK** button to accept the change.

9) ***Double-click*** the header of the **Attribute2** column to display the **Edit Attribute** dialog.

10) ***Type*** "**CellLabel**" in the **Name** field of the **Edit Attribute** dialog and ***click*** the **OK** button to accept the change.

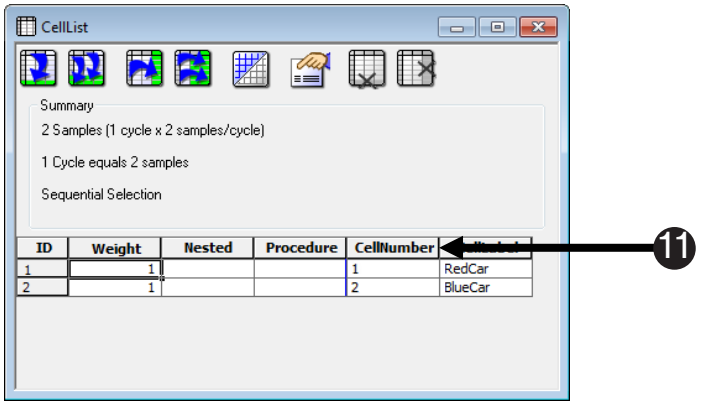


Task 14 (continued): Add the required CellNumber and CellLabel attributes to the CellList

Rename the new attributes to CellNumber and CellLabel then edit the attribute values to define the cells used within the experiment.

11) ***Click*** in each cell under the **CellNumber** and **CellLabel** attributes and ***edit*** the values to match those shown.

⚠ **NOTE:** *It is recommended that you assign CellNumbers sequentially in order to maintain compatibility with most prior releases of Net Station.*



Task 15: Add the required CellNumber attribute to the TrialList

Add a CellNumber attribute to the TrialList. Assign values to the CellNumber attribute to identify the specific cells represented in the TrialList.

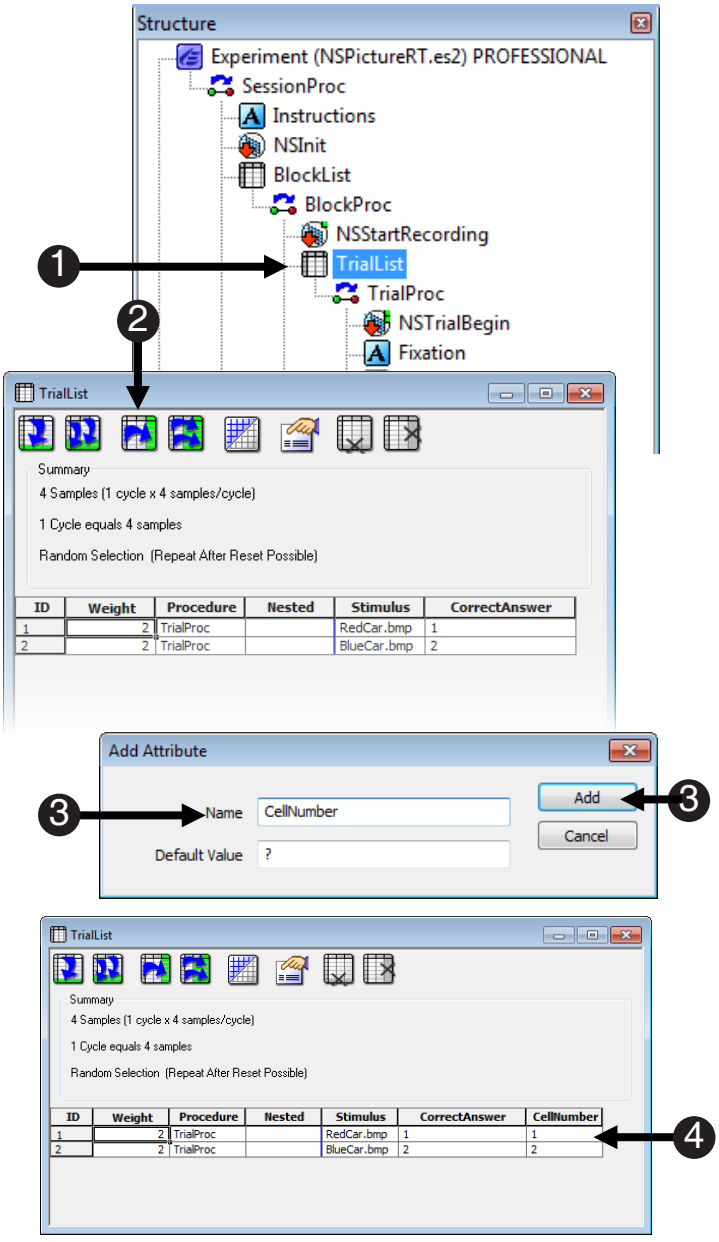
While you may define many different types of cells within an experiment, each individual trial must be associated with a single specific cell when it is run. You associate a trial with a particular cell by adding a CellNumber attribute to your trial level list (e.g. the TrialList object in this example) and then setting its values to identify the CellNumber of appropriate cell that is related to each trial represented in the list. Open the TrialList object in the workspace, create a new CellNumber attribute, and assign its values. Note that there is no need to also create or include a CellLabel attribute on the trial level list (e.g. since the CellNumber is identified for each trial, the Net Station Package File can retrieve the corresponding CellLabel as needed from the CellList that was defined earlier.)

1) ***Double-click*** the **TrialList** object to open it in the workspace.

2) ***Click*** the ***Add Attribute*** button to add a new attribute to the **TrialList**.

3) ***Type*** "**CellNumber**" in the **Name** field of the **Add Attribute** dialog and ***click*** the **Add** button to create the new attribute.

4) ***Click*** in each cell under the **CellNumber** attribute and ***edit*** the **values** to match those shown in the picture.



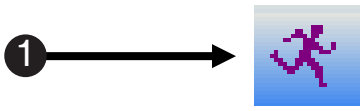
Task 16: Run the experiment

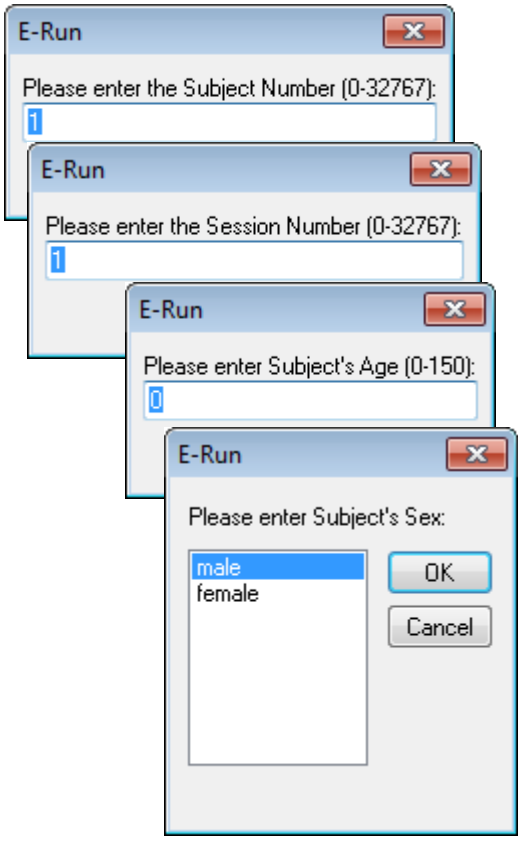
Run the experiment to verify that the appropriate prompts are presented and no runtime errors are generated.

Your experiment should now be enabled for use with Net Station! Once you have launched the experiment you will be presented with a series of initialization prompts. Press Enter or click OK to accept the default values during testing. Read and respond to any instructions presented then observe the stimulus presentation sequence to verify the experiment is functioning correctly and that no errors are generated. If you have, Net Station connected and active you should also use the Experiment Control Status Panel within the Workbench to review and verify the valid communications of Net Station related events.

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

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





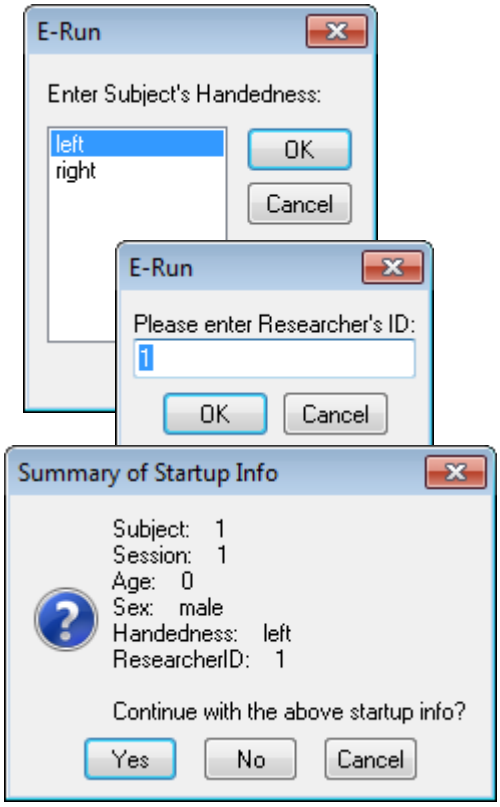
Task 16 (continued): Run the experiment

Run the experiment to verify that the appropriate prompts are presented and no runtime errors are generated.

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

⚠ **NOTE:** *When testing without the Net Station hardware connected you will receive a message during initialization indicating that Net Station communications will be disabled. While this is the expected functionality when testing locally, if this message occurs with the Net Station hardware connected and active, it indicates a communications problem that must be resolved.*

4) ***Save*** the **experiment**.



Tutorial 2: **Adding User Defined Keys**

Summary:

Net Station allows you to send additional trial information to Net Station when sending the Trial Specifications or TRSP event. Any attribute that is defined during the trial can be sent. To enable this feature you create and define a new “KeyList” within your experiment and add a reference to the KeyList in each call to the Net Station Package routine that is responsible for sending the TRSP event to Net Station. At runtime, the current value of all user-defined key’s attributes, defined in the KeyList, will be retrieved from the experiment Context and sent to Net Station as part of the TRSP event.

Goal:

This tutorial illustrates how to add a user defined KeyList to the NSPictureRT experiment created in Tutorial 1. The KeyList will include one new key (named “stim”) that will send to Net Station a reference to the name of the actual stimulus that was presented on each trial.

Overview of Tasks:

- Add a KeyList to the Unreferenced E-Objects branch of the Structure window.
- Configure the KeyList object to add the required attributes and user-defined keys.
- Add a reference to the KeyList object in the NetStation_SendTRSPEvent call.
- Verify the overall experiment structure and run the experiment.

Estimated Tutorial Time: 10-20 minutes

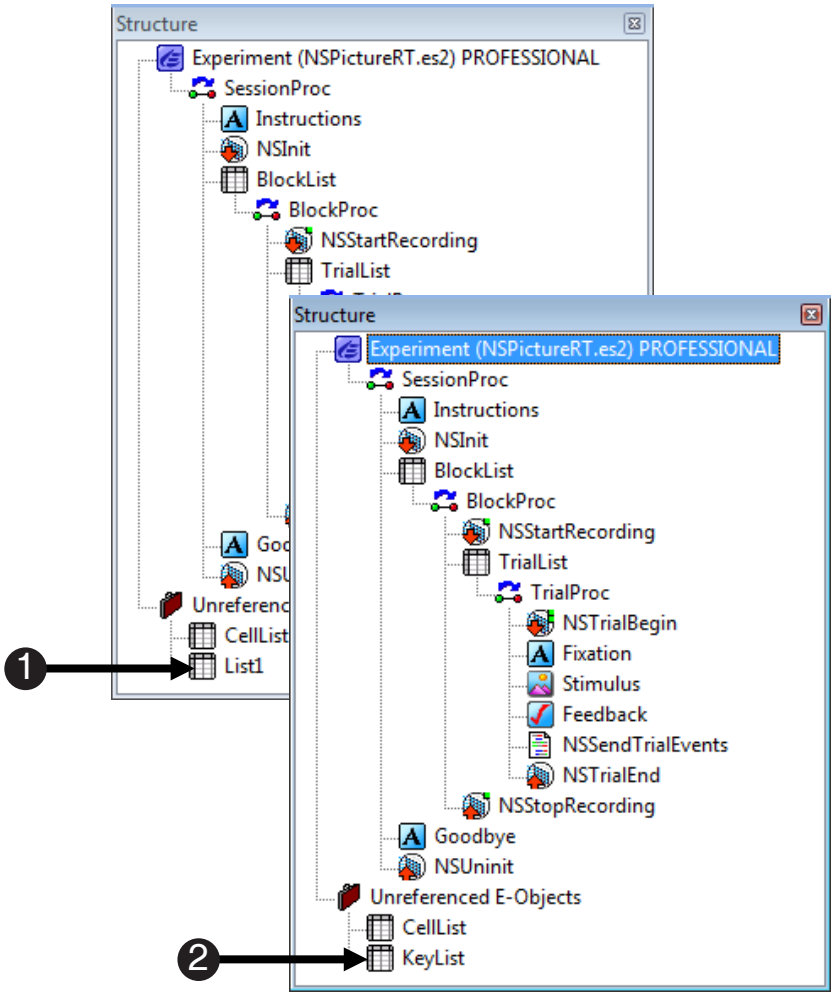
Task 1: Add a KeyList object

Add a new List object to the Unreferenced E-Objects branch of the Structure window. Name the object "KeyList".

A Net Station enabled experiment identifies user-defined keys by the use of one or more "Key Lists" defined within the experiment. The routines within the Net Station Package File use these lists at runtime to lookup information (e.g. name, ID, data type, value) related to each user defined key.
A KeyList is just a List object that has been created in the experiment and configured with a specified set of named attributes. Create a new List object in the Unreferenced E-Objects branch of the Structure window and rename it as "KeyList".

1) ***Open*** **NSPictureRT.es2** and ***drag*** a new **List** object from the **Toolbox** and ***drop*** it on the **Unreferenced E-Objects** branch of the **Structure** window.

2) ***Click*** on the new **List1** object to ***select*** it then ***press*** **F2** to ***rename*** the object as "**KeyList**".



Task 2: Add the required KeyName, KeyID, and KeyDataType attributes to the KeyList

Open the KeyList object in the workspace and add three new attributes.

Each Key List that is defined in an experiment must contain the three named attributes of “KeyName”, “KeyID”, and “KeyDataType”. Open the KeyList object in the workspace and add three new attributes.

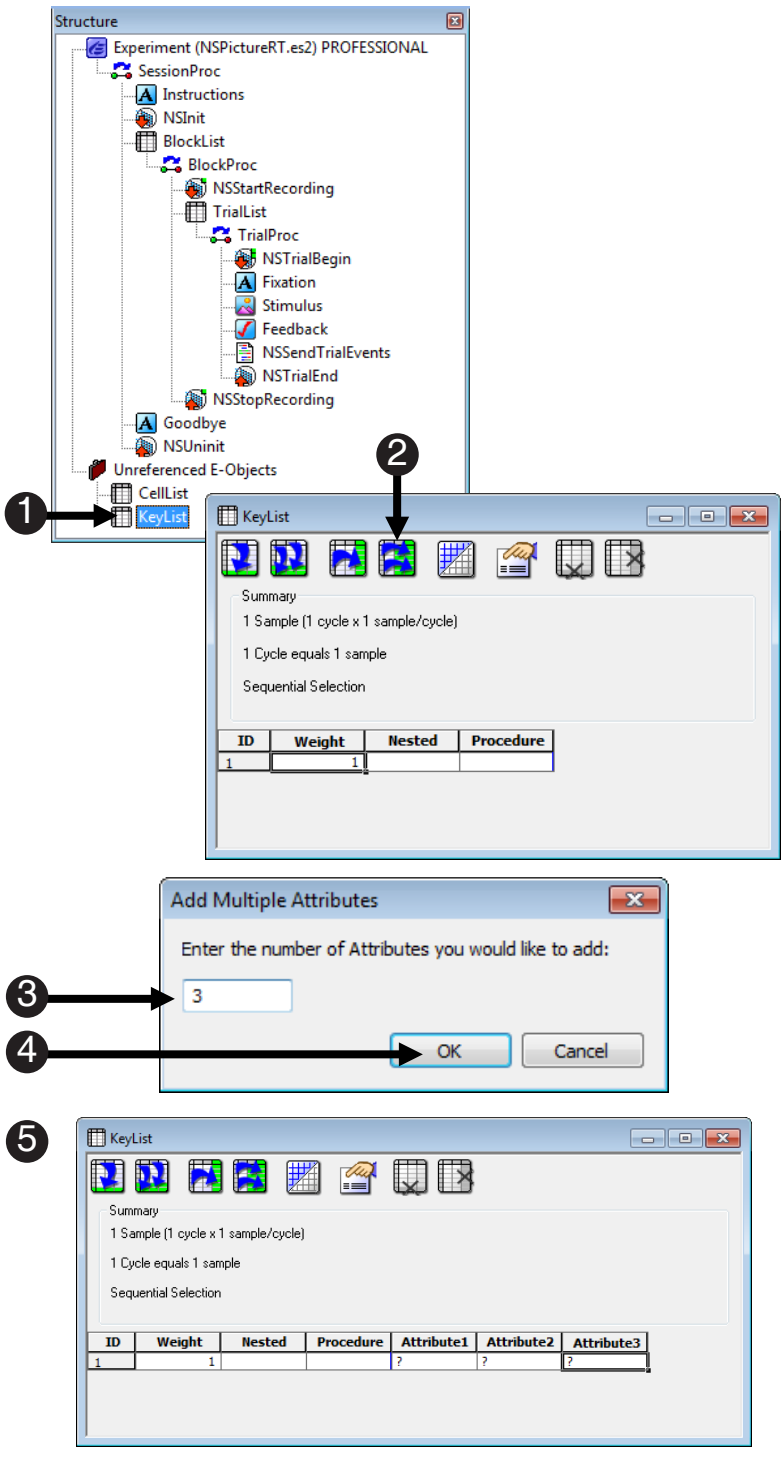
1) ***Double-click*** the **KeyList** object to open it in the workspace.

2) ***Click*** the **Add Multiple Attributes** button.

3) ***Type*** “**3**” in the **Add Multiple Attributes** dialog.

4) ***Click*** the **OK** button to ***add*** **three** new attributes to the KeyList.

5) ***Verify*** that your **KeyList** matches the picture.



Task 2 (continued): Add the required KeyName, KeyID, and KeyDataType attributes to the KeyList

Rename the new attributes to KeyName, KeyID, and KeyDataType then edit the attribute values to create a new user defined key.

The routines within the Net Station Package File expect each List object that is used as a Key List to contain a set of attributes named “KeyName”, “KeyID”, and “KeyDataType”. Rename each attribute respectively. If you misspell or forget to include one of these attributes the Net Station Package File will generate an error when you attempt to run the experiment. The value of KeyName must be a string. The value of KeyID must be a four-character code. The value of KeyDataType must be one of the following codes: “shor”, “long”, “bool”, or “TEXT”. If the value of KeyDataType is left blank, it will default to “shor”.

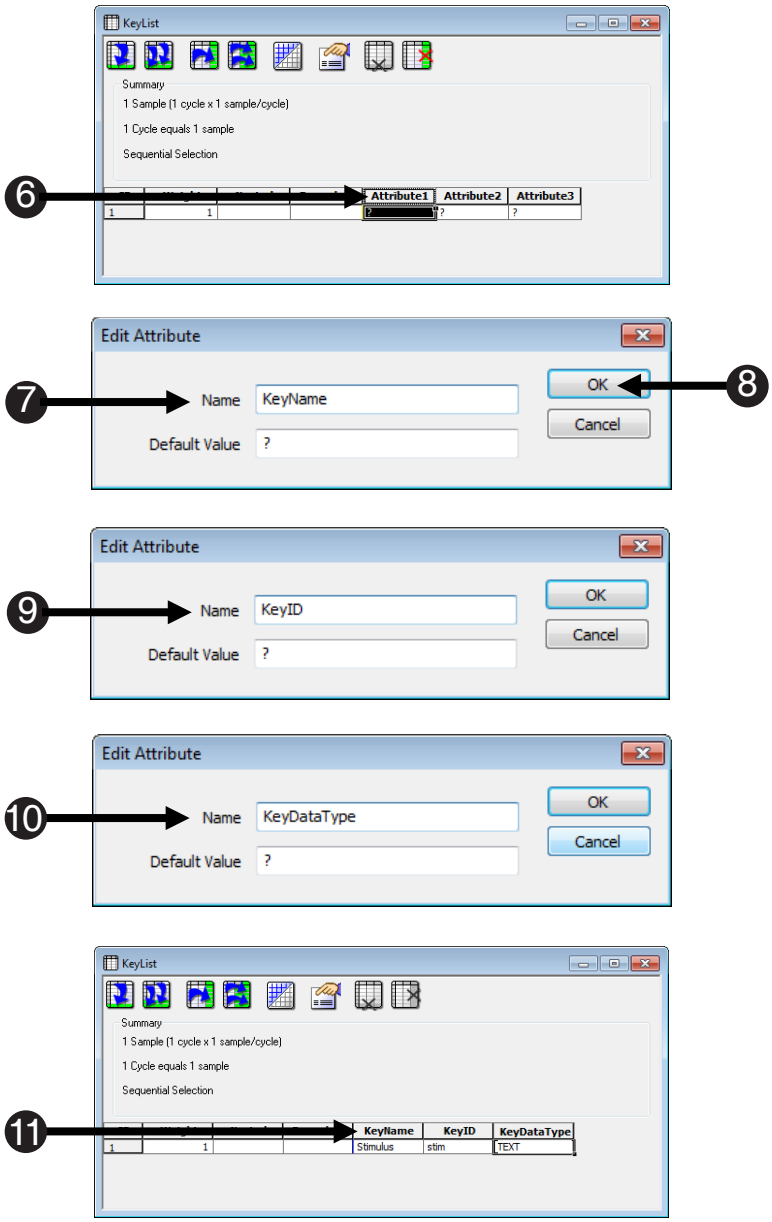
6) ***Double-click*** the header of the **Attribute1** column to display the **Edit Attribute** dialog.

7) ***Type*** “**KeyName**” in the **Name** field of the **Edit Attribute** dialog.

8) ***Click*** the **OK** button to ***accept*** the change.

9) ***Repeat*** steps 1-3 above to ***rename*** **Attribute2** and **Attribute3** to **KeyID** and **KeyDataType** respectively.

10) ***Click*** in each cell under the **KeyName**, **KeyID**, and **KeyDataType** attributes and ***edit*** the values to match those shown.



Task 3: Add a reference to the KeyList

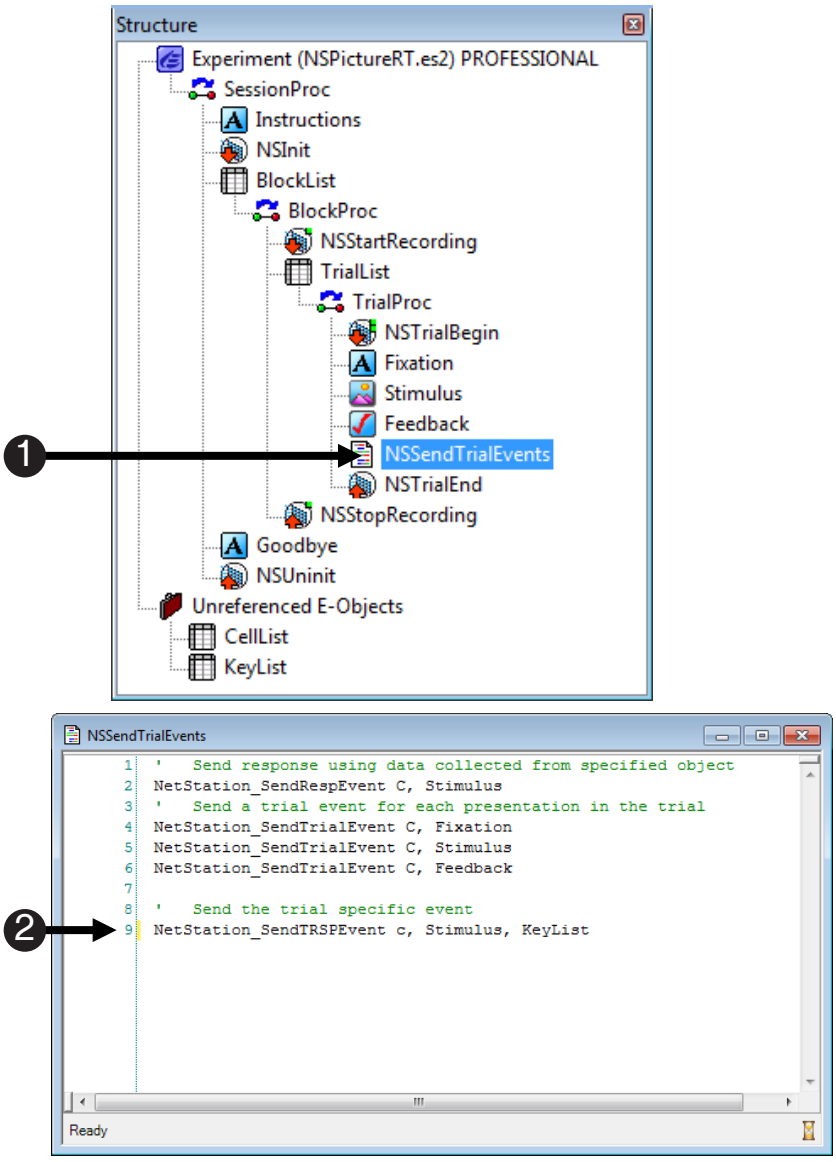
Add script to the NSSendTrialEvents Inline to append the KeyList to the package file routine that sends the TRial SPecific (TRSP) event to Net Station.

The NetStation_SendTRSPEvent includes an optional parameter that is used to specify a reference to a list of user-defined keys. While it is acceptable to define multiple Key Lists within an experiment, typically only one list is used and only one KeyList can be sent for a given call to NetStation_SendTRSPEvent. Each Key List that is defined must be configured to fully describe the additional key data that is to be associated with the current trial.

Add the optional parameter to the end of the NetStation_SendTRSPEvent parameters list to associate the KeyList object with the TRSP event. At runtime, the NetStation_SendTRSPEvent routine will process the KeyList by retrieving each KeyName, looking up its current value within the experiment Context, and then appending all of the key information to the end of the TRSP event. If a specified KeyName cannot be found in the context, a runtime error will be generated.

1) ***Double-click*** on the **NSSendTrialEvents** inline object to open the dialog box.

2) ***Type*** in the **script** exactly as shown to add a reference to the **KeyList** as the last parameter of the **NetStation_SendTRSPEvent** routine.



Task 4: Run the experiment

Run the experiment to verify that the new key is being sent to Net Station and that no runtime errors are generated.

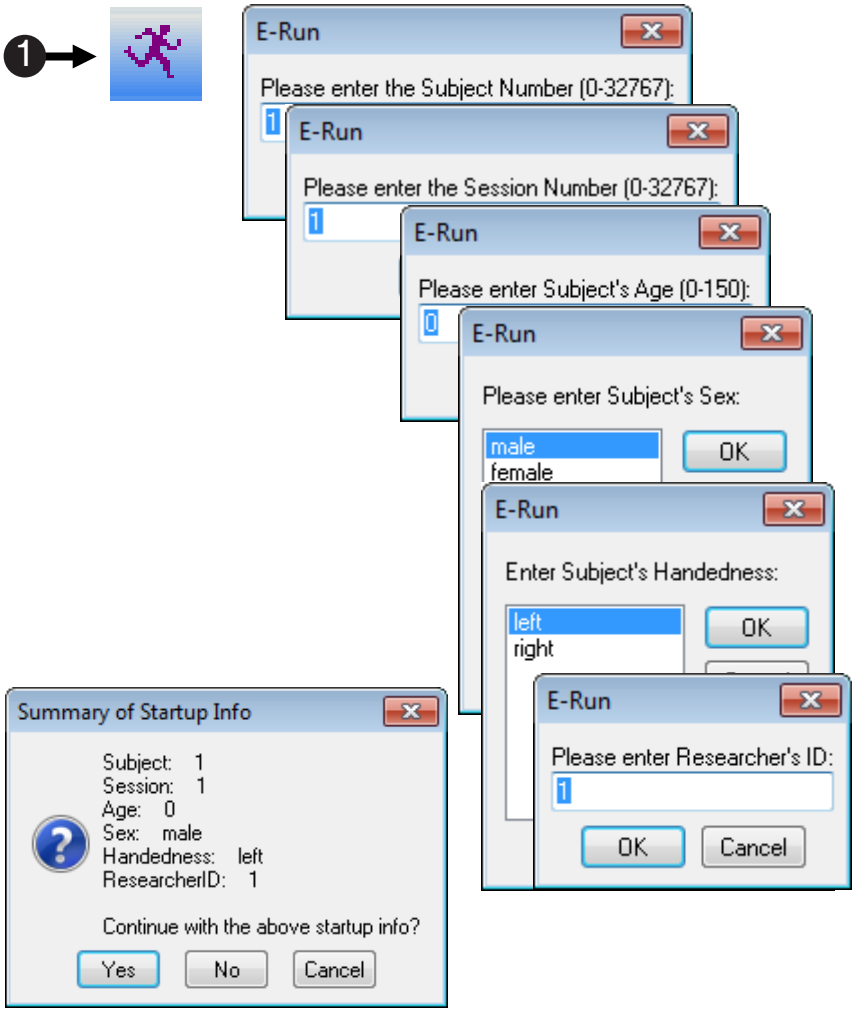
Your experiment should now be configured to send additional keys from the KeyList. Once you have launched the experiment you will be presented with a series of initialization prompts. Press Enter or click OK to accept the default values during testing. Read and respond to any instructions presented then observe the stimulus presentation sequence to verify the experiment is functioning correctly and that no errors are generated. You can verify that the events are being sent by looking at the event track in Net Station and the ECI log window, which is launched from the ECI controls on the left panel of controls in Net Station Acquisition. Events should be visible in both locations.

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

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

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

4) ***Save*** the **experiment**.



Tutorial 3: **Adding a Practice Block**

Summary:

It is common for the design and implementation of an experiment to evolve over time during its useful life cycle (e.g., number of trials is increased/decreased as new stimuli are added/deleted, new experimental conditions/cells are introduced, participant friendly breaks are introduced, etc.) Many experiments introduce blocks of practice trials after a basic version of the experiment is functioning correctly. Within a Net Station enabled experiment,introducing a block of practice trials often theoretically introduces a new set of cells that need to be reflected to Net Station (e.g. practice related data can be ignored during critical data analysis). This process typically involves adding or restructuring entries in the Cell List and assigning new Cell Numbers to the appropriate trials.

Goal:

This tutorial illustrates how to add a new block of practice trials into the NSPictureRT sample experiment created during Tutorial 1 and Tutorial 2. You will edit the experiment's Cell List and assign new Cell Numbers and Cell Labels to the trials of the practice block.

Overview of Tasks:

- Add two new cells/rows to the CellList
- Add a new PracticeBlockProc procedure to the BlockList object
- Add the required Net Station PackageCalls to the PracticeBlockProc
- Add and configure a new PracticeTrialList object to represent the new practice trials
- Add instruction displays to begin and end the block of practice trials
- Verify the overall experiment structure and run the experiment

Recommended readings:

It is required that you have read through and completed at least Tutorial 1 before beginning this tutorial. It is recommended that you have also completed Tutorial 2.

Estimated Tutorial Time: 15-30 minutes

Task 1: Add new cells to the CellList

Add two new cells to the CellList then edit the attribute values to define the new practice cells used within the experiment.

Adding a block of practice trials to the experiment will conceptually create two new cells (which will be name “redcar_prac” and “bluecar_prac” in this example). Add two new rows to the CellList and assign the CellNumber and CellLabel attributes accordingly to define the two new cells. The values for the CellNumber attribute must be numeric and must be unique within the experiment. You may currently define up to 255 unique cells within an experiment. It is recommended that when you add new cells to the CellList you assign new CellNumbers sequentially (e.g. to assure compatibility with any previously collected data). Values for the CellLabel can be any string you choose to identify and describe the cell (i.e. CellLabels do not have to be four character codes).

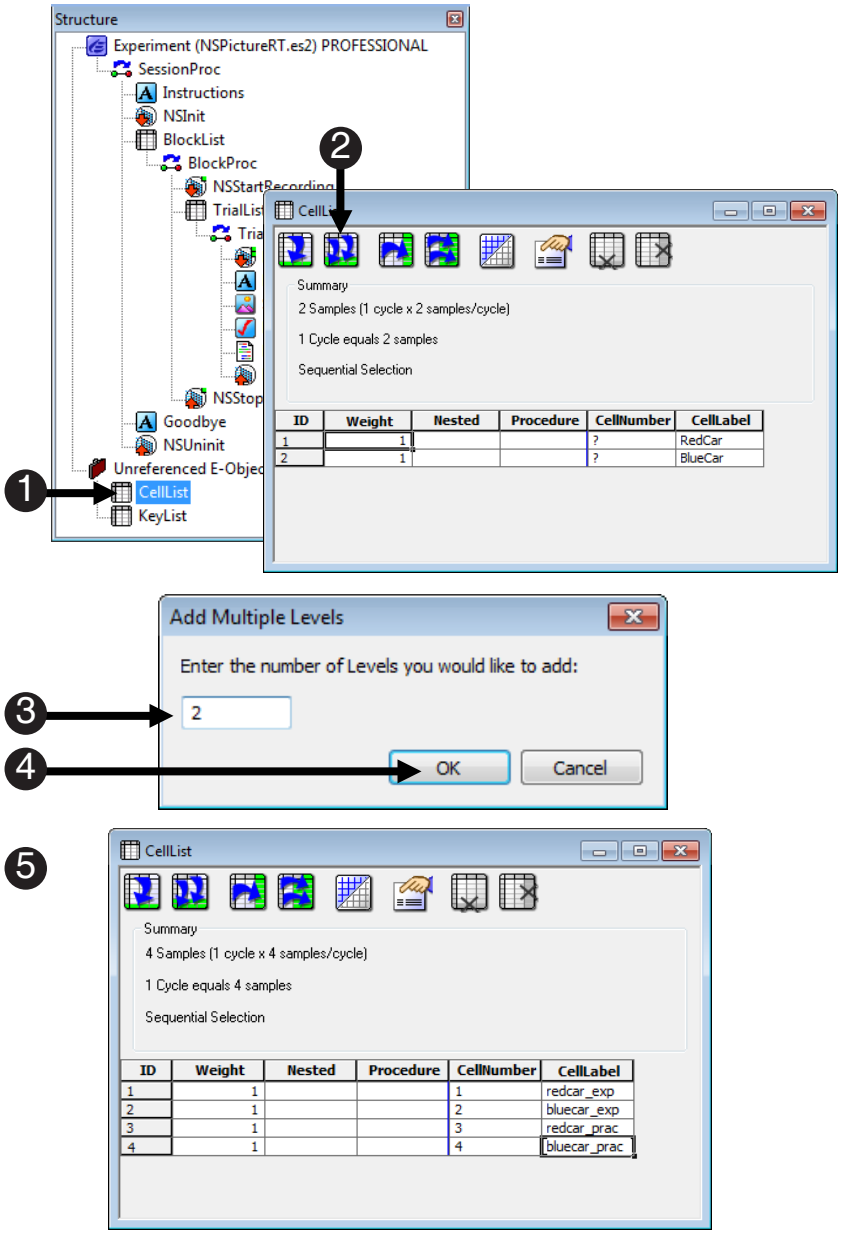
1) ***Open*** **NSPictureRT.es2** and ***double-click*** the **CellList** object to open it in the workspace.

2) ***Click*** the **Add Multiple Levels** button.

3) ***Type*** “2” in the **Add Multiple Levels** dialog.

4) ***Click*** the **OK** button to ***add*** **two new levels/rows** to the **CellList**.

5) ***Click*** in each cell under the **CellNumber** and **CellLabel** attributes and ***edit*** the values to match those shown.



Task 2: Edit the BlockList to add a PracticeBlockProc Procedure

Add a new row to the BlockList object and create a new PracticeBlockProc Procedure to represent the practice block.

Create a new level/row in the BlockList to represent the new block of practice trials.
Type “PracticeBlockProc” under the Procedure attribute to create a new Procedure object that will be used to execute the new block. Allow E-Studio to create the new PracticeBlockProc object and set the existing PracticeMode attribute to “Yes” to log an indication that this block is a block of practice trials.

1) ***Double-click*** the **BlockList** object to open it in the workspace.

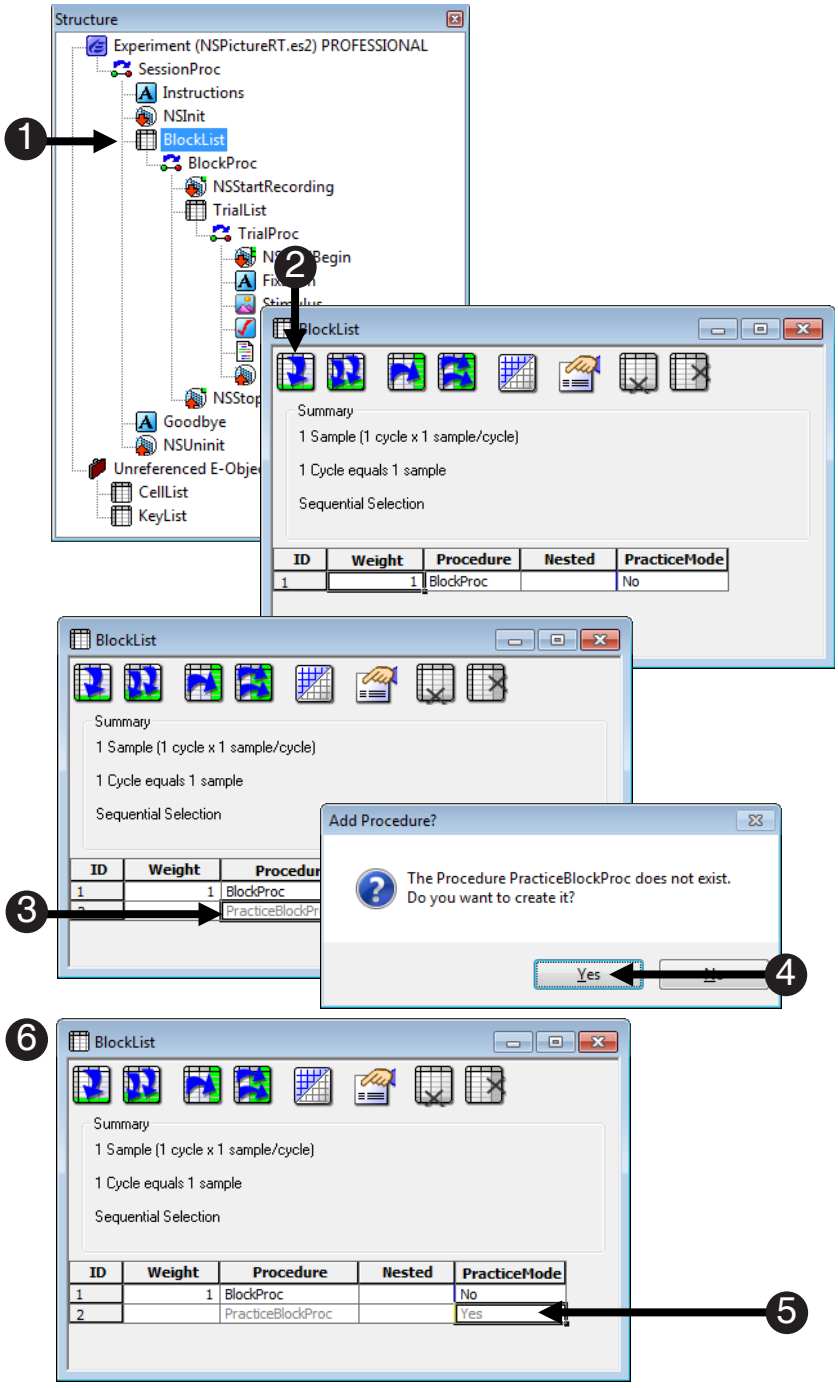
2) ***Click*** the **Add Level** button to add a new level/row to the **BlockList**.

3) ***Click*** in the cell under the **Procedure** attribute in the second row, ***type*** **PracticeBlockProc**,and then ***press*** **Enter**.

4) ***Click*** the **Yes** button to create a new **PracticeBlockProc** Procedure in the experiment.

5) ***Click*** in the cell under the **PracticeMode** attribute in the second row, ***type*** **Yes**, and ***press*** **Enter**.

6) ***Verify*** that your **BlockList** matches the picture.



Task 2 (continued): Edit the BlockList to add a PracticeBlockProc Procedure

Reorder the levels/rows of the BlockList so that the practice block is run first.

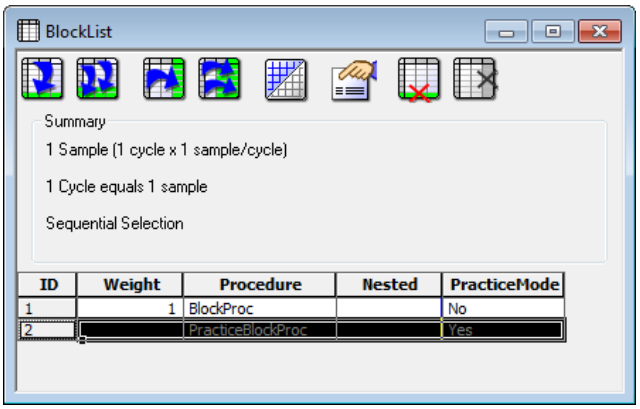
When a List object is configured to select exemplars sequentially, the order of the rows directly determines the sampling order. Click the row header for the second row and drag it above the first row so that the block of practice is run first followed by the block of real trials. If you want to run multiple blocks of practice or real trials, you can edit the cells under Weight attribute as desired.

7) ***Click*** the **ID** row header to ***select*** the **second row**.

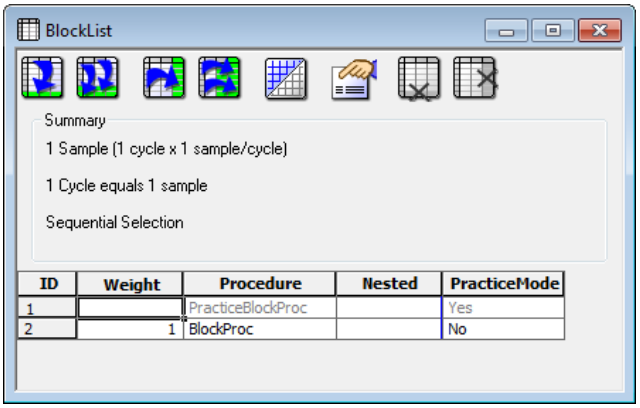
8) ***Drag*** the **second row** above the **first row** to ***reorder*** the blocks then ***verify*** that your **BlockList** matches the picture.

⚠ **NOTE:** *During the drag operation, you will see a red line indicating where the selected row will be moved to when it is dropped.*

❶



❷



Task 3: Specify the PracticeBlockProc

Add the required Net Station PackageCalls to the beginning and ending of the PracticeBlockProc.

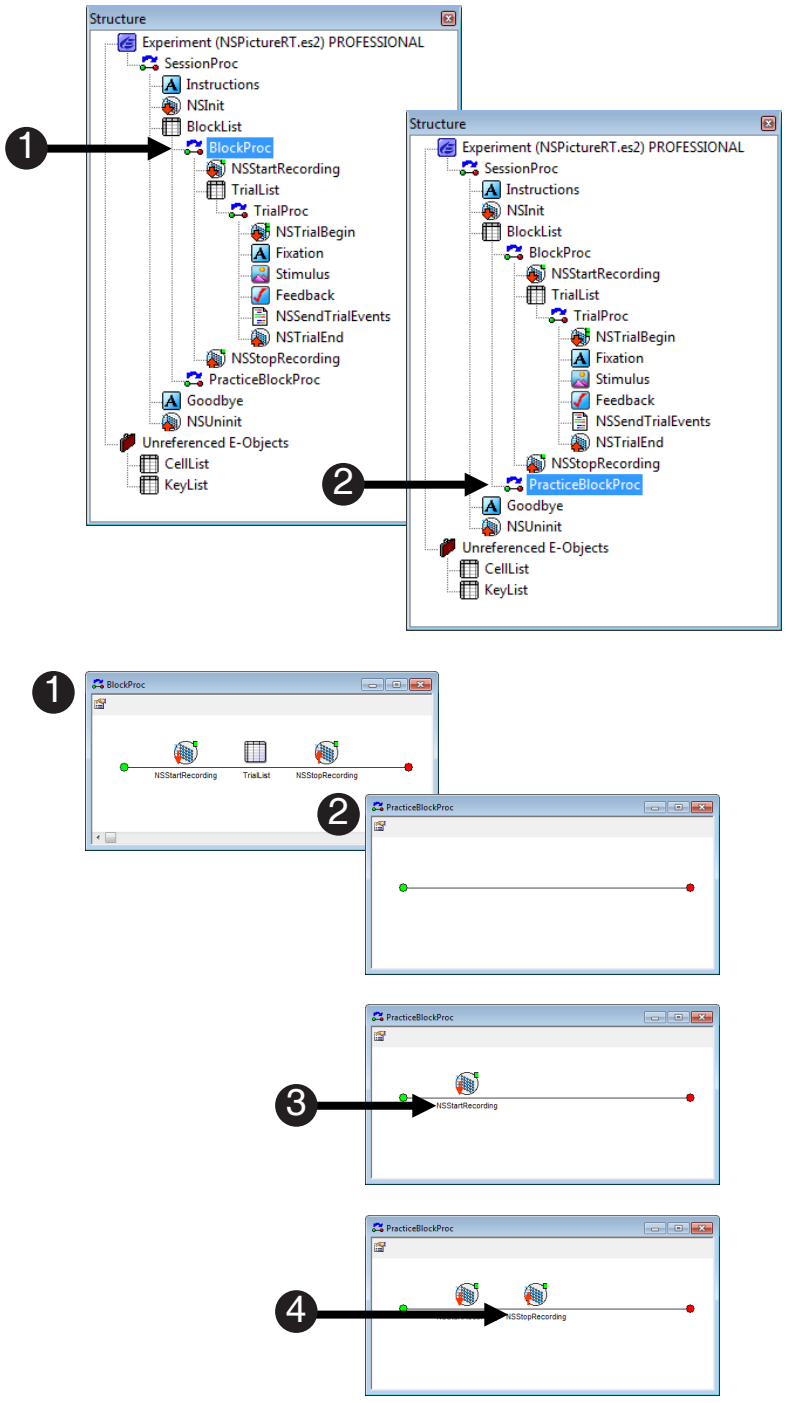
Configure the PracticeBlockProc to start Net Station recording at the beginning of the practice block and stop Net Station recording at the end of the practice block. You can reuse the NSStartRecording and NSStopRecording objects that are already being called from the original BlockProc by dragging and dropping them on PracticeBlockProc object.

1) ***Double-click*** the **BlockProc** object to open it in the workspace.

2) ***Double-click*** the **PracticeBlockProc** object to open it in the workspace.

3) ***Drag*** the **NSStartRecording** object from the **BlockProc** and drop it in the **PracticeBlockProc**.

4) ***Drag*** the **NSStopRecording** object from the **BlockProc** and ***drop*** it as the last object in the **PracticeBlockProc**.



Task 4: Create and configure the PracticeTrialList

Create a new List to represent the practice trials by making a copy of the TrialList object.

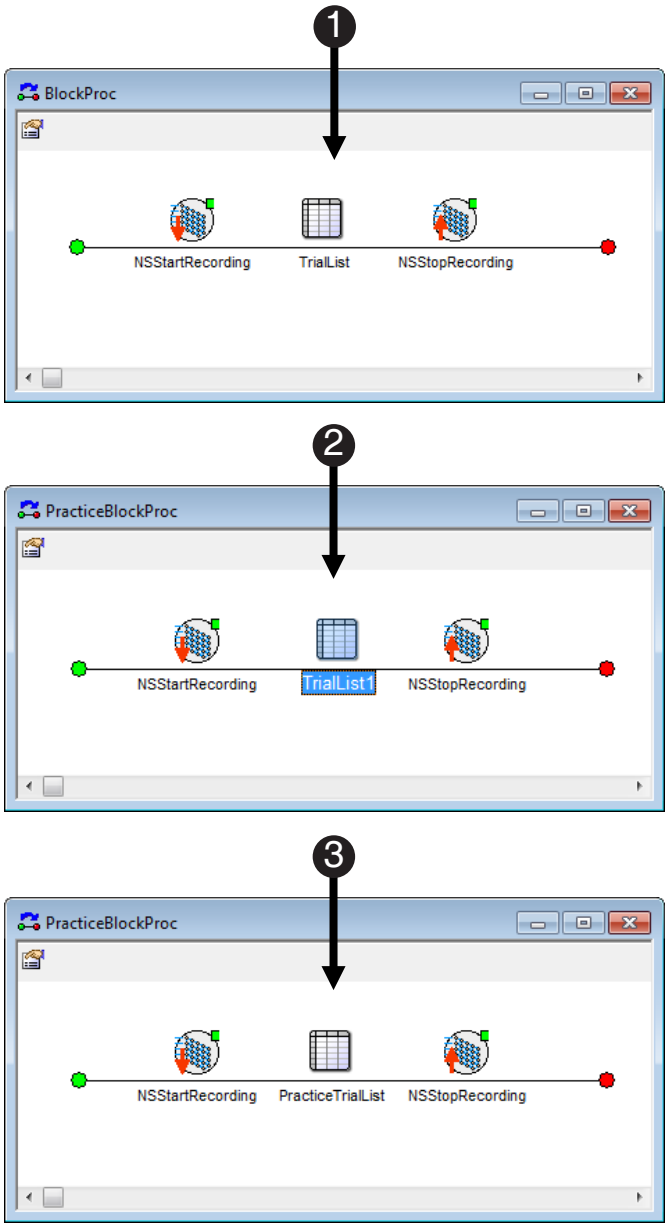
The new block of practice trials will require a unique List at the trial level since it will need to reference CellNumbers that are different from those used in the real trials. Since the existing TrialList object is already configured appropriately for use with Net Station you can copy it and rename it to quickly create a PracticeTrialList for use in the PracticeBlockProc. To make a copy of an existing object you select the object and then drag it into position while holding down the Ctrl key. When a copy operation is in progress the mouse cursor will include a plus sign (+). Be sure to keep the Ctrl key held down until after you release the mouse button.

1) ***Click*** the **TrialList** object to select it within the **BlockProc**.

2) ***Hold down*** the **Ctrl** key, ***drag*** the **TrialList** object over the **PracticeBlockProc**, and ***drop*** it between the **NSStartRecording** and **NSStopRecording** objects.

⚠ **NOTE:** *Dragging an object while holding down the Ctrl key will make a new copy of the object. The mouse cursor will include a plus sign (+) when a copy operation will occur. Notice that the new object that is created is named TrialList1.*

3) ***Click*** on the **TrialList1** object to ***select*** it then ***press*** **F2** and ***rename*** the object as "**PracticeTrialList**".



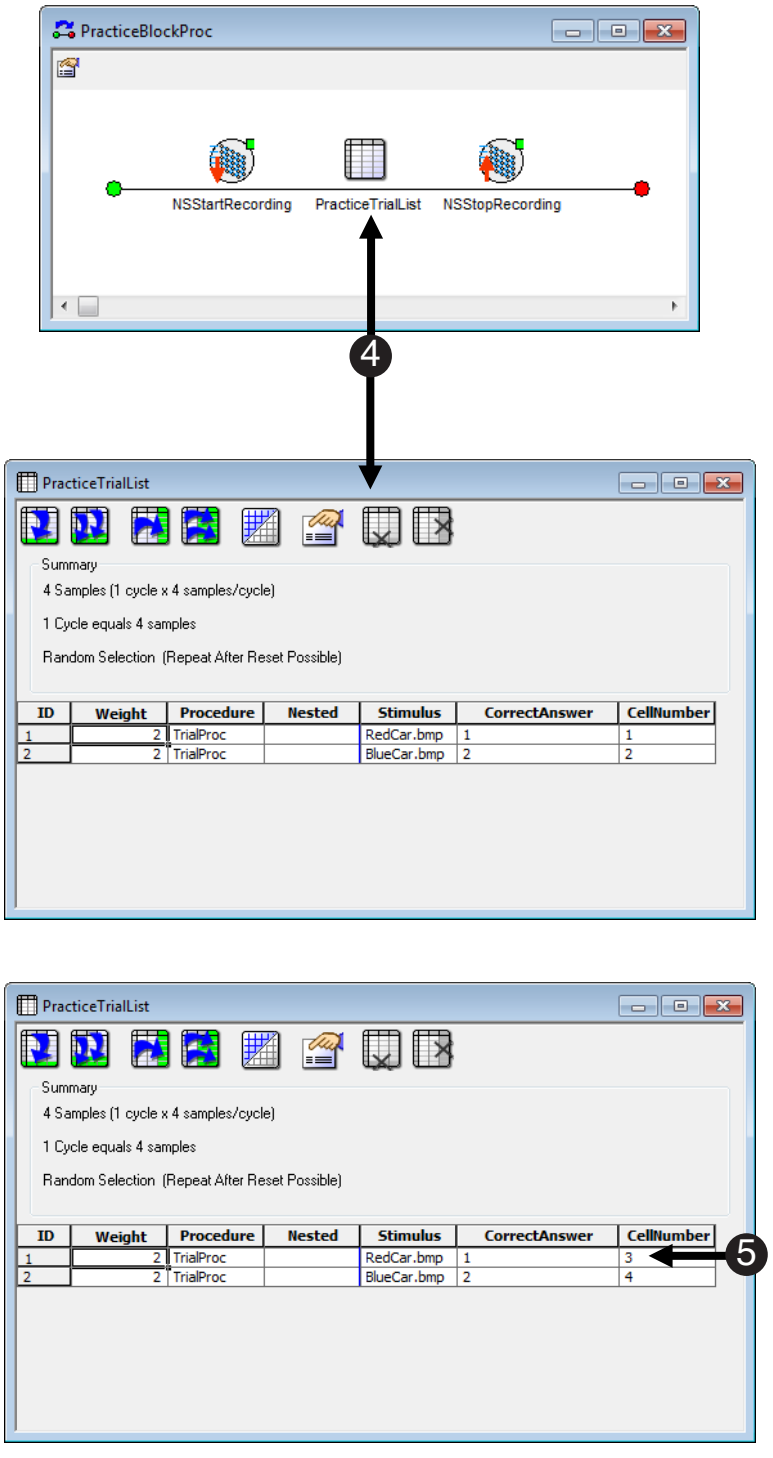
Task 4 (continued): Create and configure the PracticeTrialList

Assign the CellNumbers required for the practice trials.

Because the PracticeTrialList was created by copying the existing TrialList the CellNumbers are incorrect for use in the practice block. The practice trials must reference the new cells that were created in the CellList (i.e., “redcar_prac” and “bluecar_prac”). Edit the cells under the CellNumber attribute to reference the new cell numbers of 3 and 4 respectively.

4) ***Double click*** the **PracticeTrialList** object to ***open*** it in the workspace.

5) ***Click*** in the **cells** under the **CellNumber** attribute and ***edit*** the **values** to match those in the picture.



Task 5: Add an instruction display to begin practice

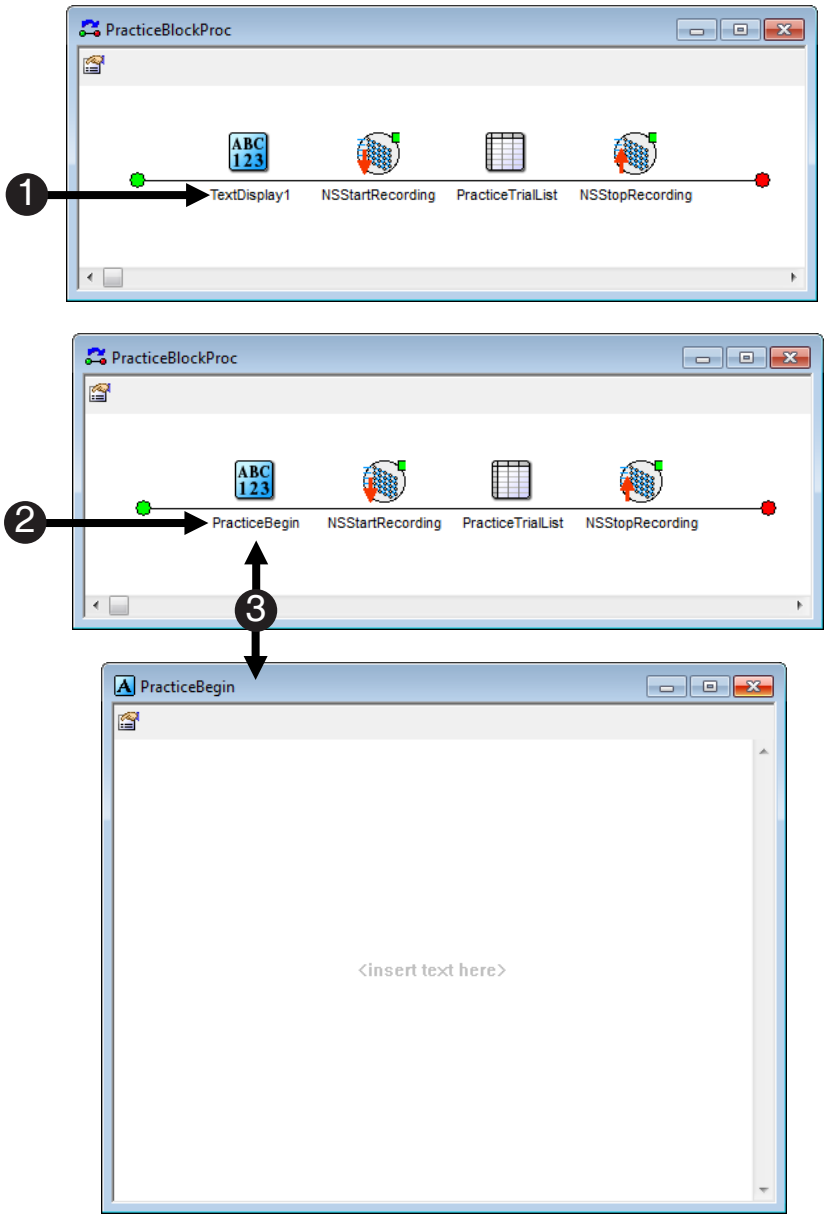
Add a new TextDisplay at the beginning of the practice block to orient the participant.

Create a new TextDisplay object at the very beginning of the PracticeBlockProc (before Net Station recording begins) to notify the participant that they are about to begin a block of practice trials. Note, if you want to combine text, images, and sound to present instructions within your own experiments it is recommended that you use a Slide object instead of a TextDisplay object.

1) ***Drag*** a new **TextDisplay** object from the **Toolbox** and ***drop*** it at the beginning of the **PracticeBlockProc**.

2) ***Click*** on the **TextDisplay1** object to ***select*** it then ***press*** **F2** and ***rename*** the object as "**PracticeBegin**".

3) ***Double-click*** the **PracticeBegin** object to ***open*** it in the **workspace**.



Task 5 (continued): Add an instruction display to begin practice

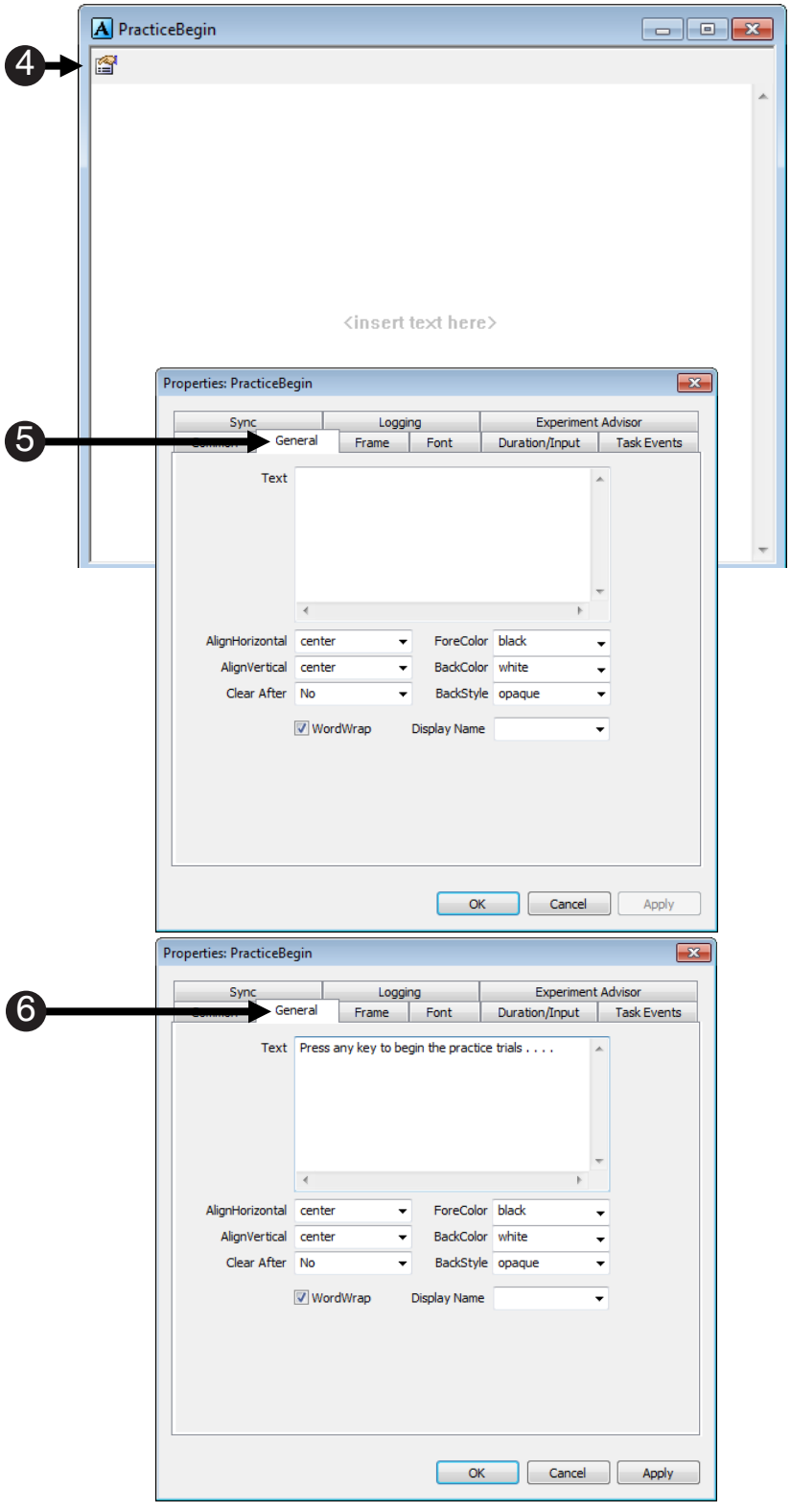
Type the text that you want to present to the participant.

Open the Properties dialog for the PracticeBegin object and type in the text that you want to have presented to the participant prior to the start of the practice block.

4) ***Click*** the **Properties** button to display the **Properties** dialog for the **PracticeBegin** object.

5) ***Click*** on the **General** tab to ***select*** it.

6) ***Click*** in the **Text** field and ***type*** "**Press any key to begin the practice trials**..."



Task 5 (continued): Add an instruction display to begin practice

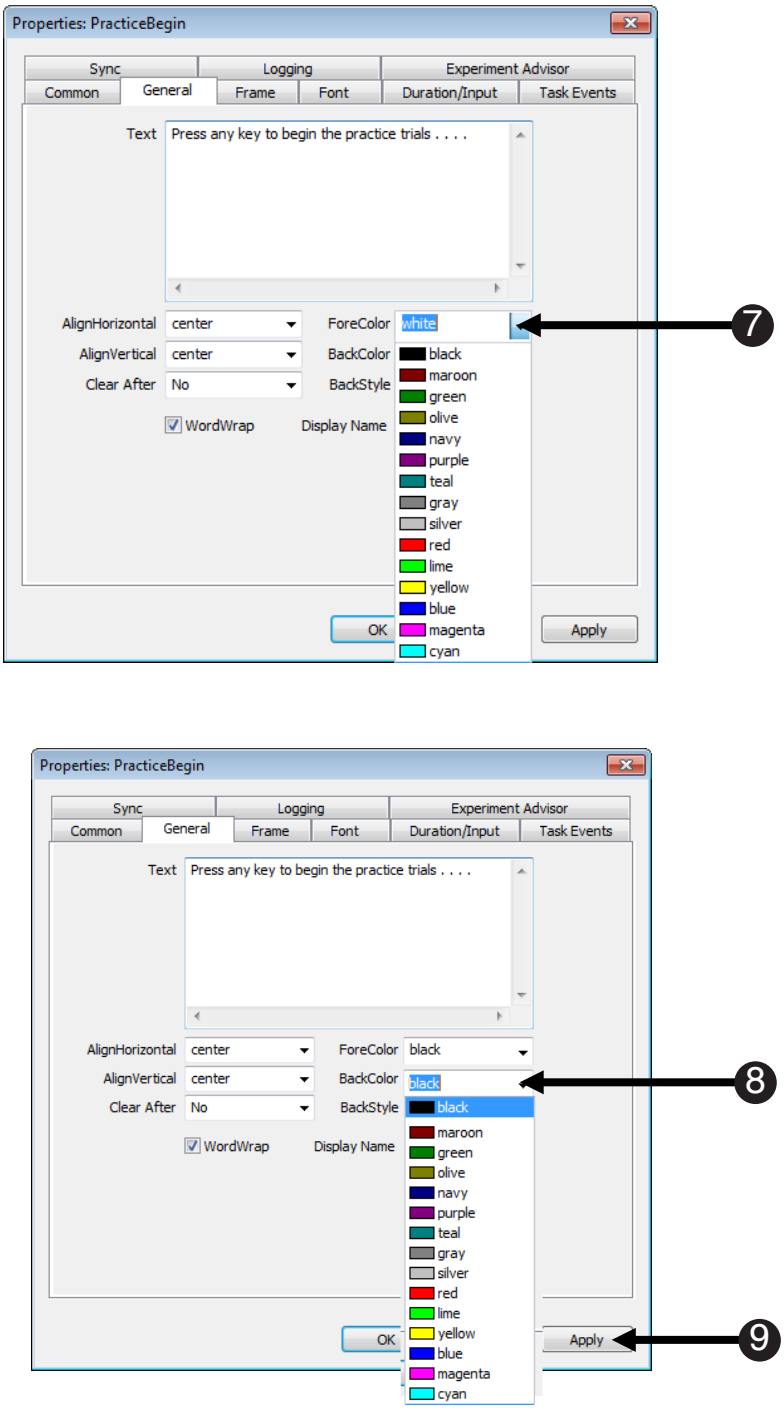
Set the foreground and background colors for the instruction display.

The PracticeBegin object is currently configured to present black text on a white background however, other displays within the experiment are configured to present white text on a black background. Change the foreground and background properties of the PracticeBegin object to match those that are used in the rest of the experiment.

7) ***Click*** the **dropdown** beside the **ForeColor** field and ***select*** "**white**" as the text foreground color.

8) ***Click*** the **dropdown** beside the BackColor field and ***select*** "**black**" as the text background color.

9) ***Click*** the **Apply** button to ***save*** the changes to this tab.



Task 5 (continued): Add an instruction display to begin practice

Configure the PracticeBegin object to wait until the participant presses a key to continue.

Set the properties of the PracticeBegin object so that it presents the instructions to the participant and then waits indefinitely until the key is pressed to terminate the display and begin the block of practice trials. Specifically, set the Duration property to "(infinite)" and then add a new Input Mask, which references the Keyboard device. The default settings for the various Response Options are acceptable and do not need to be edited further.

11) ***Click*** the **Duration/Input** tab to select it.

12) ***Click*** the **dropdown** beside the **Duration** field select "**(infinite)**" as the duration of the object.

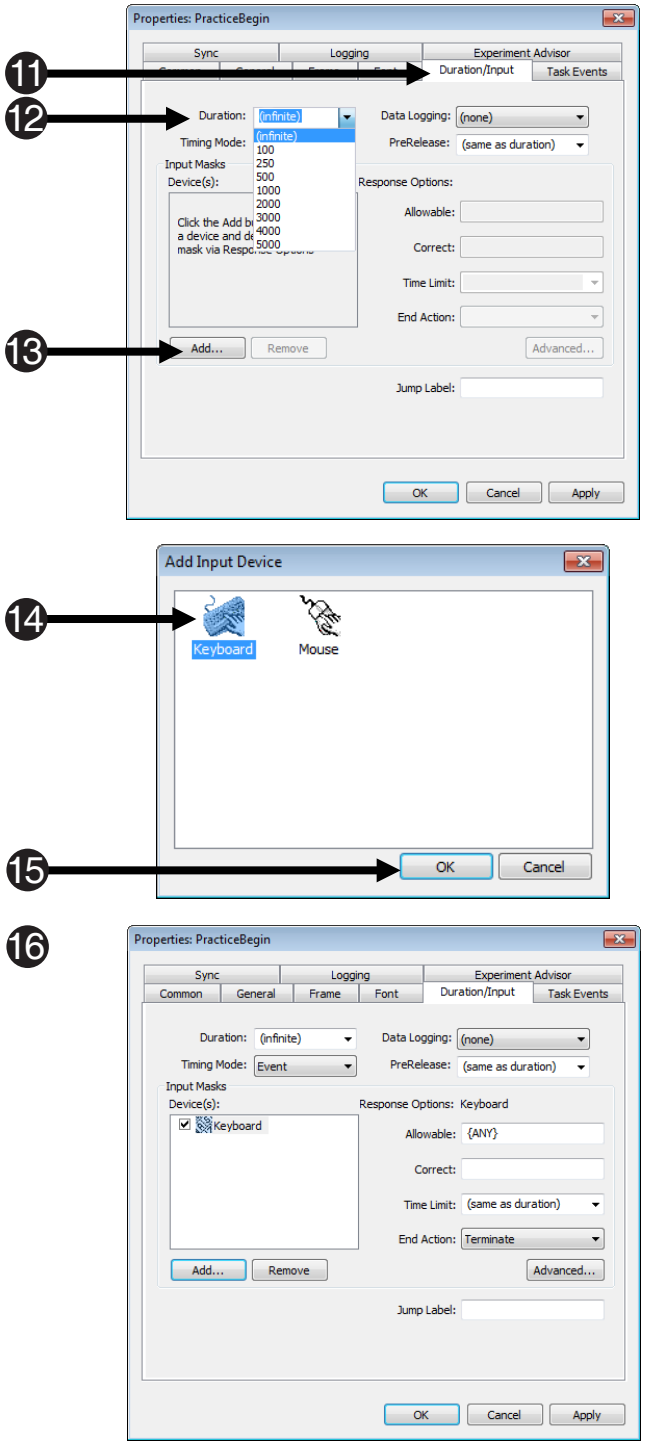
13) ***Click*** the **Add...** button.

14) ***Click*** on the **Keyboard** device to ***select*** it.

15) ***Click*** the **OK** button to ***add*** a new input mask for the **Keyboard** device.

⚠ **NOTE:** *By default, the input mask will allow any key press to terminate the input.*

16) ***Verify*** that your **Duration/Input** tab matches the picture then ***click*** the **OK** button to ***accept*** **all changes**.



Task 6: Add an instruction display to end practice

Add a new TextDisplay at the end of the practice block to notify the participant that the practice trials are complete.

Create a new TextDisplay object to notify the participant that the practice trials have been completed and the real trials are about to begin. Since the PracticeBegin object is already configured appropriately for use in the experiment (e.g. foreground color, background color, and wait for a key press to continue), it can be copied, renamed and edited for this purpose. Click on the PracticeBegin object, hold down the Ctrl key, and copy it to the very end of the PracticeBlockProc. Rename the new object as “PracticeEnd” and then open it and edit the text message it presents to appropriately notify the participant.

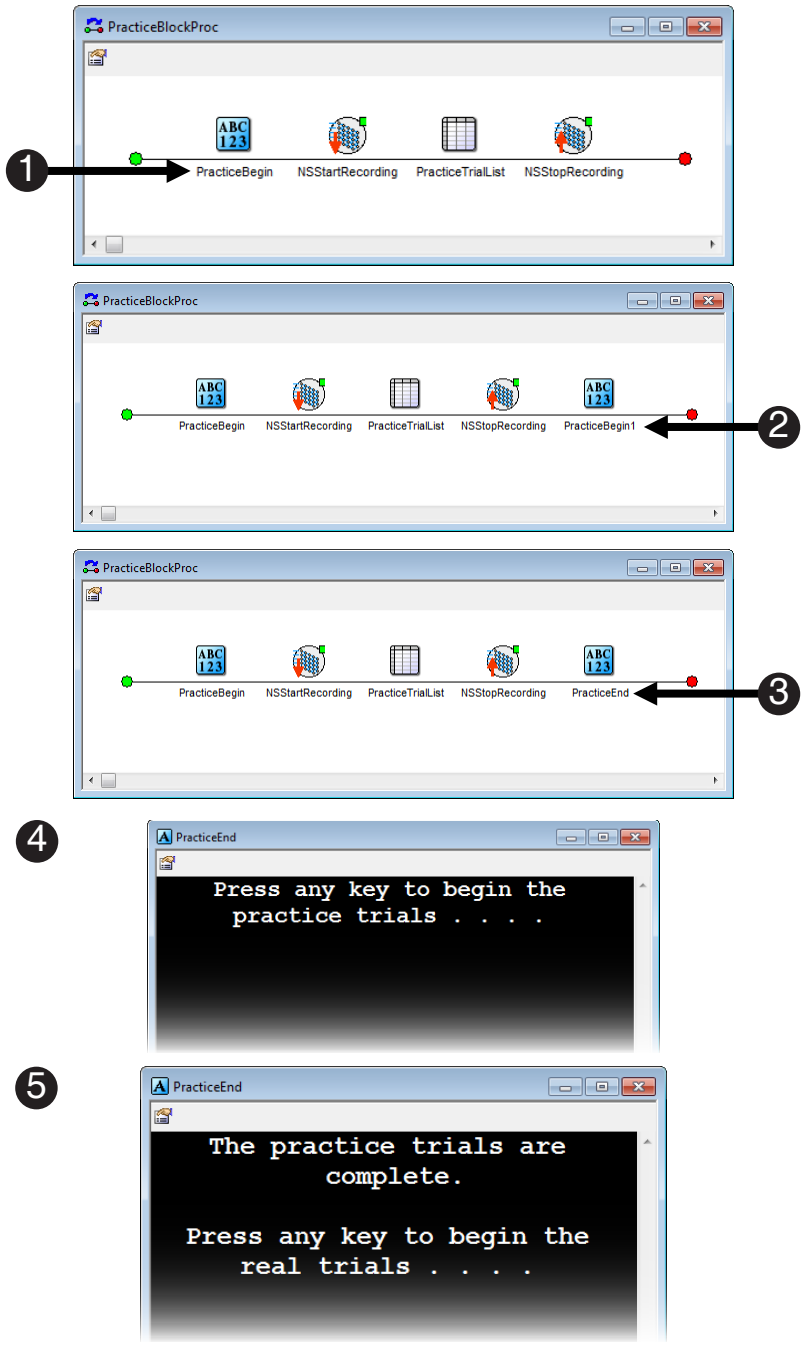
1) ***Click*** the **PracticeBegin** object to ***select*** it within the **PracticeBlockProc**.

2) ***Hold down*** the **Ctrl** key, ***drag*** the **PracticeBegin** and ***drop*** it at the end of the **PracticeBlockProc**.
NOTE: *Notice that the new object is named PracticeBegin1.*

3) ***Click*** on the **PracticeBegin1** object to select it then ***press*** **F2** and ***rename*** the **object** as “**PracticeEnd**”.

4) ***Double-click*** the **PracticeEnd** object to open it in the workspace.

5) ***Edit*** the **text** to ***match*** the **picture**.



Task 7: Run the experiment

Run the experiment to verify that the appropriate prompts are presented and no runtime errors are generated.

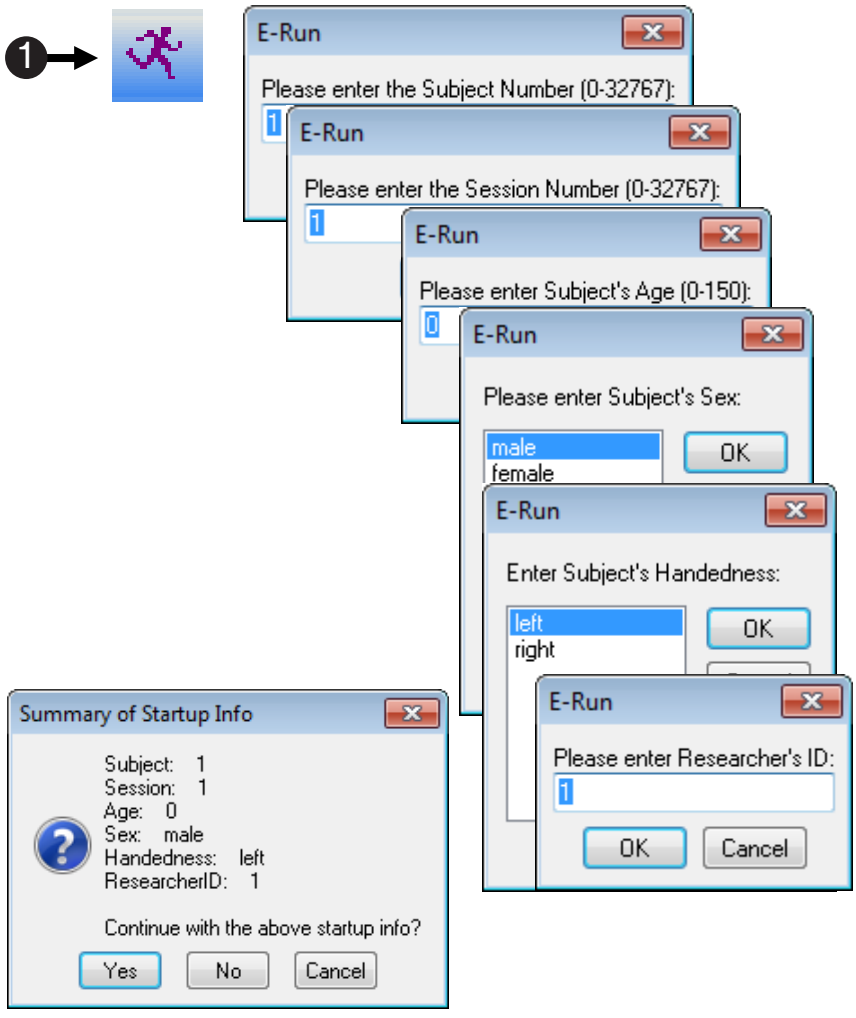
Your experiment should now be designed to present a block of practice trials prior to beginning the real trials. Once you have launched the experiment you will be presented with a series of initialization prompts. Press Enter or click OK to accept the default values during testing. Read and respond to any instructions presented then observe the stimulus presentation sequence to verify the experiment is functioning correctly and that no errors are generated.

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

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

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

4) ***Save*** the **experiment**.



Tutorial 4: **Timing Test**

NTP timing method is very efficient and reliable, but it is always prudent to verify that your new system is functioning within specification before you begin collecting any real data. Running the sample NSTimingTest.es2 experiment provided with the E-Prime Extensions for Net Station 2.0 installation will help you check your system's visual timing accuracy. You should also repeat this process to verify timing in each new experiment you create, prior to running participants. For more information on timing tests for various media, contact EGI (supportteam@egi.com).

Once you record test data using the NSTimingTest.es2 experiment and your EGI AV Tester, you can review the timing results by using the EGI Event Timing Tester application provided with Net Station.

The Net Amps amplifiers have a digital input plug in the back and an adapter that can connect with a DB9 cable. This DB9 cable, called the Digital Input or DIN cable, allows DIN events output from the AV Tester device to be recorded in the Net Station file.

Please refer to the Net Station, System, and AV Tester manuals that came with your EGI system for more information on how to use those EGI products, or contact EGI Support. If you have questions regarding how to set up and run experiments in E-Prime, review section **1.4 Network Configuration,** (Page 10).

Run the NSTimingTest.es2 sample experiment:

1) Once you have set up your AV Tester equipment and amplifier according to the EGI manuals, ***launch*** Net Station Acquisition on the Macintosh and configure the ECI controls to record the correct digital input (DIN) value. You can also open the ECI log window from this control to verify proper conveyance of events from E-Prime. You should have the ECI Log and the ECI panels open. Use default port 55513.

2) If you are running a visual timing test, make sure that you have enabled ***only*** DIN 3 in the ECI Controls. For an auditory experiment, select DIN2 only. This will eliminate any unneeded information and provide a clean DIN event in the recording file.

⚠ **NOTE:** *The three AV Tester DIN outputs differ in sensitivity. Under most circumstances, DIN 3 is appropriate for visual stimuli, while DIN 2 is better for auditory stimuli. Some monitors, such as CRT displays, may require using DIN 1. Some experimentation may be necessary.*

3) ***Turn on*** **Streaming in Net Station Acquisition**.

4) On the PC, launch the appropriate timing test for the media you are using in your experiment. This tutorial uses the E-Prime sample experiment "NSTimingTest.es2" located at ...\My Experiments\Net Station\Samples\NSTimingTest. For additional timing test experiments for use with different media, contact EGI (supportteam@egi.com).

5) ***Specify*** "**socket**" and your Macintosh computer's IP address ("e.g., "10.10.10.42") in the NSInit parameters.

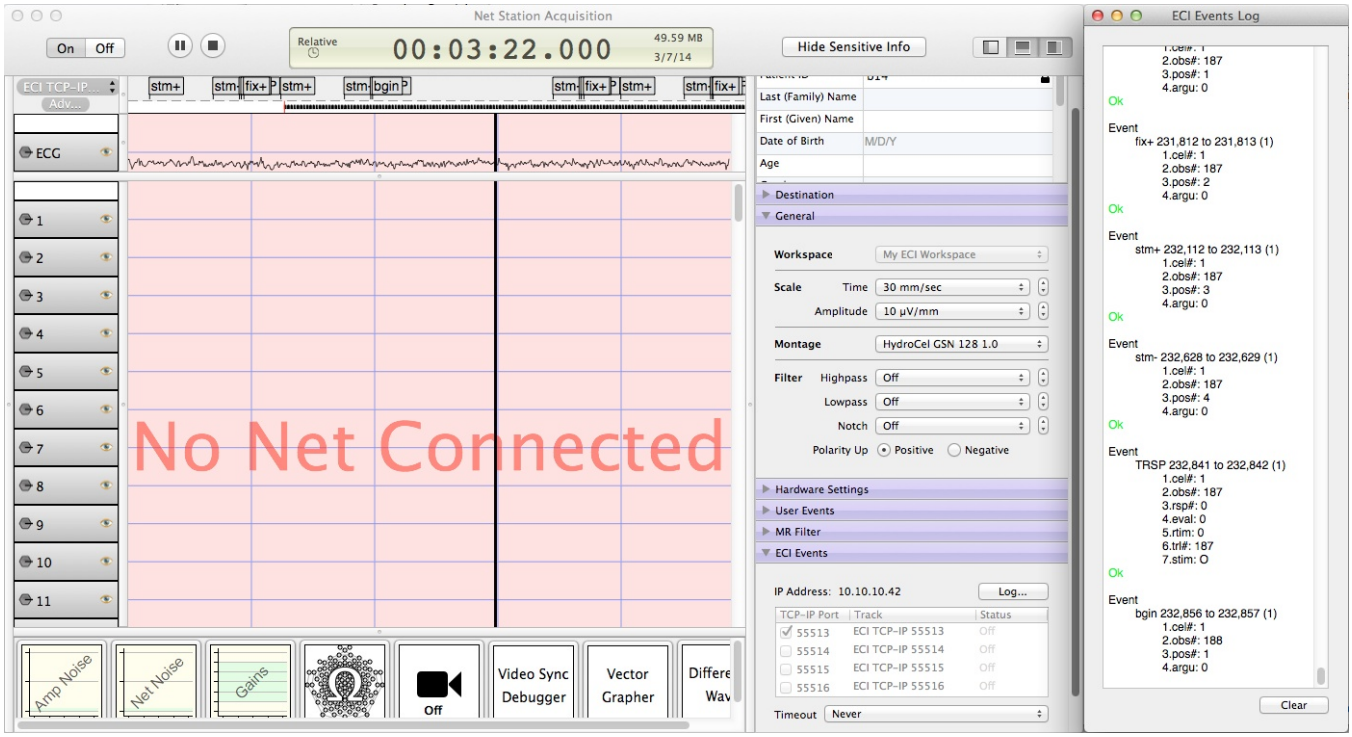
6) ***Select*** **E-Run** > **Run** from the application menu (or click the Run button) to generate and begin running the experiment.

7) ***After*** the entering **StartUp Info** in the message boxes, ***line up*** the **AV Tester Optical Sensor** with the white Stimulus square as prompted by the experiment. Make sure that the **sensor** is ***even*** and as ***close as possible*** to the **monitor screen, but avoid touching the display with the sensor**.

8) ***Continue running*** the **experiment**. You can monitor events in **Net Station Acquisition's ECI Event Track and ECI Event Log** as the experiment progresses.

9) The **NSTimingTest.es2** experiment ***flashes*** a white Stimulus rectangle on the screen every second for one hour. Each Stimulus flash corresponds with a "stim+" event.

With the Net Amps amplifiers, the AV Tester causes DIN events to be placed in the recording each time a flash of the Stimulus rectangle occurs, which can be compared to the events E-Prime sends through the ECI to verify accurate timing of events.



After you have successfully run NSTimingTest.es2, you can quit Net Station Acquisition and review the timing results with the Event Timing Tester included in your Net Station installation.

How to review a DIN file with Event Timing Tester

If you used the Net Amps for your testing and recorded DIN events in your file, you can use the EGI Event Timing Tester application to review your timing results.

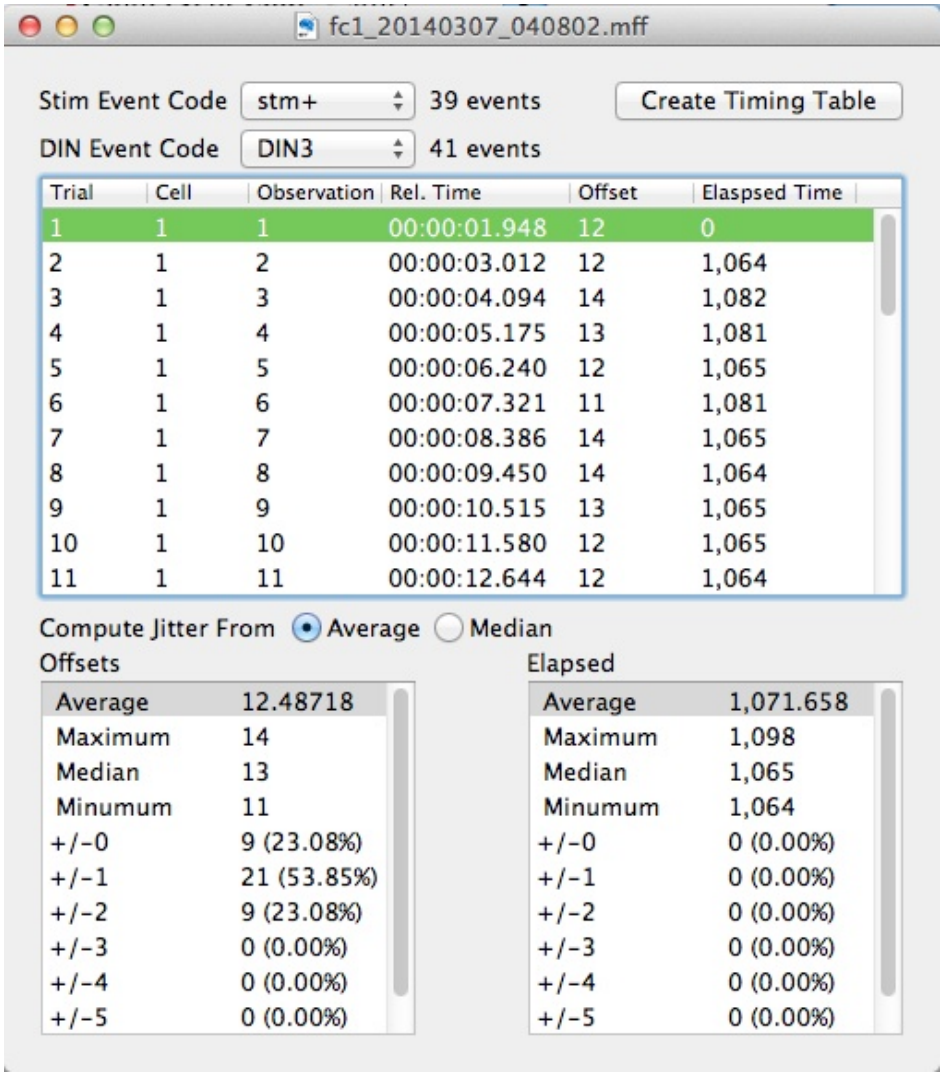
1) On the Net Station Macintosh, Launch the EGI Event Timing Tester (ETT) application. This application can be found in the Net Station application folder.

2) ***Choose*** "**Open**" from the **ETT File menu**, and ***select*** your **timing test recording** using the file browser dialog. **ETT** will ***load*** event information from your file and present a File panel.

3) From the "**Stim Event Code**" pop-up menu, ***select*** "**stm+**". This is the stimulus event from the NSTimingTest.es2 experiment.

4) From the "**DIN Event Code**" pop-up menu, ***select*** "**DIN1**", or whichever **DIN code** you used in Net Station.

5) ***Click*** the "**Create Timing Table**" button to ***generate*** a table of event information. Each row represents a single trial (flash of the Stimulus rectangle) from the NSTimingTest.es2 experiment. The columns contain specific details about each trial.



6) ***Review*** the column "**Rel. Time**" **(Relative Time)**. This is the time, in milliseconds, from the beginning of the experiment to each stm+ event.

7) ***Compare*** the "**Rel. Time**" column to the "**Elapsed Time**" column. Elapsed Time is the time duration, in milliseconds, of each trial. The Elapsed Time should equal the difference between the Relative Time of the current and previous trials. In this experiment, that would be roughly **1000** ms.

8) Most importantly, ***review*** the "**Offset**" column. The Offset value listed for each trial is the number of milliseconds between the stim+ and DIN events in Net Station.

   The Offset value represents the difference between the timestamp E-Prime assigned to the stim+ event it sent to Net Station, and the timestamp Net Station assigned the DIN event it created when the AV Tester detected the flash of the Stimulus rectangle.

9) Since E-Prime and Net Station both get time from the amplifier clock, the Offset for all trials should not vary by more than a millisecond or two for the entire recording. Look carefully through the ETT table for your test file and verify that this is true.

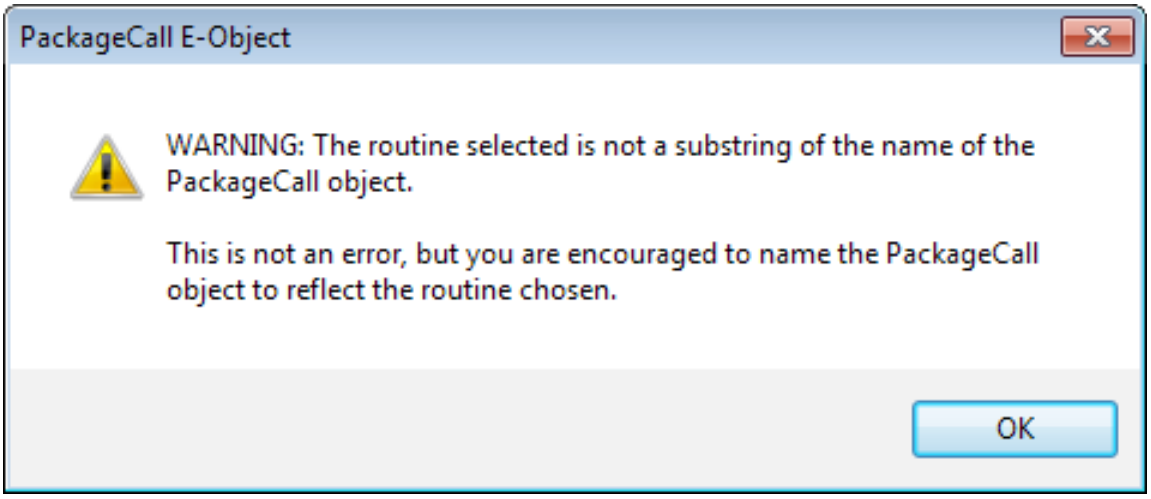
Chapter 3: **Net Station PackageCall Reference**

3.1 Introduction

This section details the set of routines that are available within the Net Station Package File and how to properly call them.

PackageCall Method
Routines contained within a package file are typically called by dragging a PackageCall object from the E-Studio Toolbox, and dropping it into a Procedure object at the location within the experiment where the call is to be invoked.

When using the PackageCall object it is strongly recommended that you rename the object to reflect the specific Package File and Routine that are being referenced within the object. Newer version of E-Prime will issue the following warning if you have not followed this recommendation.



It is also a common convention, when calling routines within the Net Station package file, to create a name for each object by abbreviating “Net Station” as “NS” and then concatenating on the name of the routine being called. For example, a PackageCall object that is configured to call the “NetStation_Init” routine would be named “NSInit”.

Inline Script Method
It is also possible to directly call a package file routine by writing E-Basic script within an Inline object that is placed at the desired location within the experiment. However, unless there is a need to make, a series of PackageCalls in sequence, or mingle PackageCalls with additional E-Basic script; this calling method is not typically recommended.

If you are making direct PackageCalls via E-Basic script, please refer to the following reference pages, which document each Net Station package routine and provide examples of the script commonly required to successfully invoke each routine.

NetStation_Init

Description

Initializes the Net Station system. This routine attempts to configure the Net Station Package File and ready it for use. It will verify that the required Startup Info has been collected (i.e. Age, Sex, Handedness, etc.) and attempt to establish communications with Net Station. If valid communications are established, recording will be initiated and Net Station and E-Prime will be synchronized. The specified Cell List will then be processed and information about each cell used in the experiment will be sent to Net Station. After this information is sent to Net Station, recording will be then be stopped again.

Syntax

```
NetStation_Init c, strState, objCellList[,varCommMethod]
[,varCommMethodParams][,varStopRecording][,vHandlePreRelease]
```

Default Parameters

```
c, "on", CellList
```

Parameters

c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

strState As String

The requested state of Net Station communications ("on", "off"). When "on" is specified, the system will attempt to establish communications with Net Station and a non-fatal error will be produced if the attempt fails. When "off" is specified, the system will not attempt to establish communications with Net Station and no errors will be generated. This mode is useful for testing your experiment in the absence of Net Station.

objCellList As List

The Cell List to use for this experiment. Specifies a List object, which contains attributes (minimally CellNumber and CellLabel) which describe and identify the cells that may be used within the experiment.

Optional varCommMethod As Variant

The requested Net Station communications method (currently, only "socket" is supported.). If not specified the communications method parameters will be read from the Method setting in the NETSTATION.INI file.

Optional CommMethodParams As Variant

A comma delimited string of parameters associated with the Net Station communications method. If not specified the communications method parameters will be read from the MethodParams setting in the NETSTATION.INI file.

When strCommMethod="socket", use this parameter to specify the IP address and port of the remote Net Station computer, e.g. "10.10.10.42". If no port number is specified the default port expected by Net Station will be used. If you want to use a different port number then append a colon and the port number to the IP address string, e.g. "10.10.10.42:55513".

NetStation_Init continued

Optional varStopRecording As Variant

Indicates if Net Station should stop recording immediately after being initialized (defaults to True). This can be set to false to remove any recording breaks between this call and the rest of the recorded file in the case that a single epoch file is preferred.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks

Station session. This initialization process consists of the following tasks: Gather information about the experiment session, i.e., researcher ID, participant age. If Net Station communication will occur, the communication parameters are setup and communication is initialized.

Example

```
‘ Initialize the system for active socket communications on IP
‘ address 10.10.10.42
NetStation_Init c, “on”, CellList, “socket”, “10.10.10.42”

‘ Turn off active socket communications on IP address 10.10.10.42.
‘ Useful when testing Net Station experiments on a lab experiment
‘ development machine that is not connected to the Net Station
‘ hardware.
NetStation_Init c, “off”, CellList, “socket”, “10.10.10.42”

‘ Initialize the system using the communication parameters specified
in
‘ the NETSTATION.INI file.
NetStation_Init c, “on”, CellList
```

NetStation_SendRespEvent

Description

Sends a RESP event to the Net Station system. Net Station requires that it be notified of any participant response of interest that was collected during each trial. This routine obtains the relevant response information from an object that is passed as the second parameter. This parameter must be a reference to the object within the trial sequence that was responsible for collecting the response. Only one TRSP event should be sent for each trial.

It is important to verify that the allowable response interval within each trial has fully completed before calling this routine. For example, if PreRelease is in use by objects contained within the trial sequence and a participant responds near the very end of the response interval there is the potential for invalid or incomplete response data to be sent to Net Station (see Chapter 4: Critical Timing, *E-Prime User's Guide*, for details on how to avoid this problem.)

If the routine determines that there was no response collected then the routine will return immediately and no response related information will be sent to Net Station.

Syntax

```
NetStation_SendRespEvent c, objRteObject [,varEventID][,vHandlePreRelease]
```

Default Parameters

```
c, RteObject
```

Parameters

c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

objRteObject As RteObject

A reference to either the stimulus object that collected the response or a ResponseData object. ResponseData objects are obtained programmatically from the Responses collection that is a property of each InputMask defined for the stimulus object (see InputMask. Responses in the E-Basic Help for more information).

Optional varEventID As Variant

An optional four character code to be used associated with the response event. If the parameter is specified its value must be exactly four characters in length or a runtime error will be generated. If the parameter is not specified it will default to 'resp'.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

NetStation_SendRespEvent continued

Remarks

Writes response event to Net Station.

Example

```
‘ Send response information collected by an object named Stimulus
NetStation_SendRespEvent c, Stimulus

‘ Send response information collected by an object named Stimulus but
‘ use “rsp1” as the id for the response
NetStation_SendRespEvent c, Stimulus, “rsp1”

‘ Send response information for each of several responses collected by
‘ an object named Stimulus and assign a unique id to each response.
Dim n As Long
For n = 1 To Stimulus.InputMasks(1).Responses.Count
   NetStation_SendRespEvent c, Stimulus.InputMasks(1).Responses(n),_
        “rsp” & n
Next n
```

NetStation_SendTrialEvent

Description

Sends a Trial event to the Net Station system. Prior to the end of each trial, Net Station requires that it be sent all relevant timing information related to any critical stimulus presentation or object/event of interest that occurred during the trial.

The routine examines the specified object's "Tag" property to obtain a unique ID for the current trial event. The object's Tag property must be set to a unique four-character code. If the ID code is not exactly four characters in length a runtime error will be generated.

The routine includes two optional parameters that can be used to associate additional user defined data with the current trial event.

Syntax

```
NetStation_SendTrialEvent c, objTrialEvent [,varKeyValue][,varKeyName]
[,vHandlePreRelease]
```

Default Parameters

```
c, TrialEvent
```

Parameters

c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

objTrialEvent As RteRunnableObject

A reference to an object in the trial sequence. The object's Tag property must be set to a unique four character code or a runtime error will be generated.

Optional varKeyValue As Variant

Optional user defined data that is to be associated with the trial event. If a value is specified it must be numeric (integer). If a value is not specified it will default to 0.

Optional varKeyName As Variant

Optional user defined four-character code to associate with the nKeyValue. If the parameter is specified its value must be exactly four characters in length or a runtime error will be generated. If the parameter is not specified it will default to 'argu'.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

NetStation_SendTrialEvent continued

Remarks

ObjTrialEvent MUST be set to a valid E-Prime object in the presentation sequence. Assumes that the object's Tag property is set to a unique four character code. Writes the trial event to Net Station.

Example

```
‘ Send trial event information for the object that presents the
‘ fixation
NetStation_SendTrialEvent c, Fixation

‘ Send trial event information for the object that presents the
‘ critical stimulus and send an addition user defined key and data
‘ obtained from an attribute that exists in the experiment context
NetStation_SendTrialEvent c, Stimulus, c.GetAttrib (“data”), “data”
```

NetStation_SendTRSPEvent

Description

Sends a TRial SPecific (TRSP) event to the Net Station system. Net Station requires that a TRSP event be sent prior to the end of each trial so that it is notified of the behavioral data related to the critical stimulus and response.

The routine includes an optional parameter that may be used to specify a reference to a user defined List object within the experiment which has been configured to describe additional key data that should be associated with the current trial. If a List object is referenced, it must be specified to minimally contain the named attributes of KeyID, KeyName and KeyDataType. The routine will process the KeyList by retrieving each KeyName, looking up its current value in the experiment Context, and then appending all of the key information to the end of the TRSP event.

Syntax

```
NetStation_SendTRSPEvent c, objStimulus [,varKeyList][vHandlePreRelease]
```

Default Parameters

```
c, Stimulus
```

Parameters

c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

objStimulus As RteRunnableInputObject

A reference to the object in the trial sequence that is responsible for presenting the critical stimulus and optionally collecting the response data. The specified object's Tag property must be set to a unique four-character code within the trial sequence or a runtime error will be generated.

Optional varKeyList As Variant

An optional Key List to use for this trial. Specifies a List object that contains attributes which describe a set of user defined keys that are to be sent to Net Station and associated with the current trial. If the parameter is not specified then no additional keys will be sent to Net Station. If the parameter is specified, the KeyList object is expected to minimally contain the named attributes of KeyID, KeyName and KeyDataType. If a specified KeyName cannot be found in the context a runtime error will be generated. The value of the KeyName attribute must a string. The value of the KeyID attribute must be a four character code. The value of the KeyDataType attribute must be one of the following codes: “shor”, “long”, “bool” or “TEXT”. If the value of KeyDataType is left blank it will default to “shor”.

NetStation_SendTRSPEvent continued

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks

ObjStimulus MUST be set to a valid E-Prime object capable of collecting a response. ObjKeyList (if specified) MUST be a List object that contains attributes KeyName, KeyID and KeyDataType. Sends TRSP (Trial specific) event to Net Station.

Example

```
' Send the TRSP event assuming that the object named "Stimulus" is
' responsible for presenting the critical stimulus and collecting
' the response.
NetStation_SendTRSPEvent c, Stimulus

' Send the TRSP event as above but also include the additional
' user defined keys that are listed in a List object named KeyList
NetStation_SendTRSPEvent c, Stimulus, KeyList
```

NetStation_StartRecording

Description
Start the Net Station system recording. Recording is typically started at the beginning the experiment or after any extended rest or break periods within the execution of the experiment have occurred.

Syntax
```
NetStation_StartRecording c [,vHandlePreRelease]
```

Default Parameters
```
c
```

Parameters
c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks
None

Example
```
‘ Start the Net Station system recording.
NetStation_StartRecording c
```

NetStation_StopRecording

Description
Stop the Net Station system recording. Recording is typically stopped at the end of a block of trials or immediately prior to any extended rest or break periods, that may occur within the experiment.

Syntax
```
NetStation_StopRecording c [,vHandlePreRelease]
```

Default Parameters
```
c
```

Parameters
c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks
None

Example
```
‘ Stop the Net Station system recording
NetStation_StopRecording c
```

NetStation_TrialBegin

Description
Notifies the Net Station system of the start of a new trial. This call is typically executed at the very beginning of each trial procedure or after the presentation of any trial level instructions which may require input from the participant in order to begin the critical stimulus presentation sequence.

When this routine is called, it will determine the currently active cell by searching the experiment context for an attribute named “CellNumber”. Once the current cell is determined the routine uses this information to increment the number of observation for the current cell as well as the current Net Station trial number.

Syntax
```
NetStation_TrialBegin c
```

Default Parameters
```
c
```

Parameters
c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Remarks
This object is typically placed after the presentation of trial instructions which may require input from the participant in order to begin the critical stimulus presentation sequence. Initializes some global variables and sends a “begin” message to Net Station.

Example
```
‘ Notify Net Station of the start of the current trial
NetStation_TrialBegin c
```

NetStation_TrialEnd

Description
The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Syntax
```
NetStation_TrialEnd  c [,vHandlePreRelease]
```

Default Parameters
```
c
```

Parameters
c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks
This object is typically placed at the end of a trial to clear the message buffer and prepare the communications engine for the next trial. Calls NetStation_Wait to clear the message buffer.

Example
```
‘ Notify Net Station of the end of the trial
NetStation_TrialEnd c
```

NetStation_Uninit

Description
Uninitializes the Net Station system. This call is typically executed at the end of the experiment (e.g. at the end of the session procedure) and is responsible for coordinating the termination of communications with Net Station.

Syntax
```
NetStation_Uninit  c [,vHandlePreRelease]
```

Default Parameters
```
c
```

Parameters
c As Context

The current experiment context. The routines in the Net Station Package File may optionally use the experiment context to retrieve the current values of attributes stored in the context. The context is typically the first parameter of every package file routine.

Optional vHandlePreRelease As Variant

An optional parameter that specifies if the HandlePreRelease Routine should be called prior to the execution of this Routine's script. If not specified this will default to, "True".

Remarks
Sends a "experiment complete" message to Net Station.

Example
```
‘ Uninitialize the Net Station system
NetStation_Uninit c
```

Chapter 4: **Contact Information**

For additional information or support

PSYCHOLOGY SOFTWARE TOOLS
Solutions for Research, Assessment, and Education

Contact us at
Psychology Software Tools, Inc
311 23rd Street Extension, Suite 200
Sharpsburg, PA 15215-2821
Phone: 412-449-0078
Fax: 412-449-0079
www.pstnet.com

For E-Prime product support and technical issues:
Please e-mail us at support@pstnet.com or visit us at https://support.pstnet.com

For Net Station product support and technical issues,
contact EGI at supportteam@egi.com