

NeoWin Interface Designer

Complete Plugin Scripting Tutorial

Build · Export · Test — using the NWIDCompanion plugin

— includes Page Enter/Exit Events, GoSub Action wiring, Master Pages, Canvas Border, and Multi-Modal UI

What this tutorial covers

Each test walks you through the complete workflow:

1. **Build** — create controls in the NWID Designer, set their properties and wire their events
2. **Export** — generate the runtime HTML and subs file
3. **Set up VNW** — first test only: configure your publication
4. **Script** — write the plugin commands in your VNW subroutines
5. **Test** — run it and verify the expected result

Requirements

- NWID Designer open and running
- NWIDCompanion plugin installed in VisualNEO Win
- A VNW publication with one ezEdge rectangle named Rectangle1

Revised Edition v10

How to Read This Tutorial

Each test is divided into numbered phases. The color of the phase bar tells you where you are working:

1 Build in Designer
2 Export
3 Set Up VNW
4 Script + Test

Designer Interface Overview

Area	Description
Control Palette	Left panel — click a control type to select it, then click on the canvas to place it.
Canvas	Center — your layout. Click to select, drag to move, drag corners to resize.
Properties Panel	Right panel — set the ID, caption, and all other settings. The Events tab is inside here.
Events Tab	Inside the Properties Panel — wire events to VNW subroutines using GoSub actions.
Page Tabs	Bottom bar — add pages with the + button. Click a tab to switch. Master page tabs show a ■ prefix.
Master Pages	■ NEW — shared background layers visible on every assigned page.
Canvas Border	NEW — decorative border around the page canvas. Found in Canvas Properties → Border group.
Export Button	Top toolbar — generates runtime.html and runtime_subs.txt.

Wiring Events to VNW Subroutines

All event wiring is done through JS Actions in the Events tab:

- 1 Select the control and open the Events tab in the Properties Panel.
- 2 Click ■ Add to add a new action row.
- 3 Set the Event dropdown to the event you want (e.g. Left_Click, Value_Changed).
- 4 Set the Action dropdown to GoSub.
- 5 Type your VNW subroutine name in the Value field (e.g. OnTestClick).

VNW Publication Setup — Interface Tester

NWID includes a ready-made VNW publication called Interface Tester. Rectangle1 is already placed and configured. A file picker runs automatically when you launch it.

1 Build in Designer

Export from NWID — Click the Export button. NWID generates two files: runtime.html and runtime_subs.txt.

2 Export

Open Interface Tester — Open Interface Tester.pub in VisualNEO Win.

3 Set Up VNW

Paste the subroutines — Open the Subroutine Action in VNW. Delete existing content. Copy runtime_subs.txt and paste it in.

4 Script + Test

Run and pick your HTML — Run the publication. The file picker appears — navigate to your export folder and select runtime.html.

Note: Repeat all four steps every time you re-export from NWID. Your plugin commands go inside the stub subroutines at the top of the pasted subs content. [NWID.Rect] is already set to "Rectangle1" in the generated subs.

The 8 Plugin Actions — Quick Reference

Action	Parameters	Purpose
NWID_Set	"Rectangle1" "ctrlId" "property" "value"	Set any property on any control
NWID_Get	"Rectangle1" "ctrlId" "property" "[ResultVar]"	Read a property. Result in [NWID.PropValue]
NWID_Visibility	"Rectangle1" "ctrlId" "Show Hide Enable Disable Toggle"	Show, hide, enable, disable or toggle
NWID_Navigate	"Rectangle1" "GoToPage NextPage PrevPage Reload" "pageName"	Navigate between pages
NWID_Items	"Rectangle1" "ctrlId" "AddItem ClearItems SetText" "text"	Manage ComboBox/ListBox items
NWID_GoSub	"subName"	Call any VNW subroutine by name
NWID_AddRow	"Rectangle1" "tableId" "cell1,cell2,..."	Append a row to a Table
NWID_ClearRows	"Rectangle1" "tableId"	Remove all rows from a Table

Setting properties — two methods:

Plugin command:

```
NWID_Set "Rectangle1" "label:lbl_status" "caption" "Hello!"
```

SetProp pattern:

```
SetVar "[lbl_status.caption]" "Hello!"
SetVar "[SetObjectProperty]" "lbl_status.caption"
GoSub "SetProp"
```

Note: Per-control variables are set automatically on every event. See next page for the full reference.

Automatic Control Variables

Set automatically in VNW before every subroutine call — no code needed

When any control fires an event, NWID sets per-control VNW variables using the control's own ID.

Example — Button named btn_test fires Left_Click:

```
:OnTestClick
  MessageBox "Test" "ID=[btn_test.name] Event=[btn_test.event]"
Return
```

Variables available for every control:

Variable Pattern	Example (btn_test)	Contains
[ctrl_id.name]	[btn_test.name]	Control ID string — "btn_test"
[ctrl_id.event]	[btn_test.event]	Event that fired — e.g. "Left_Click"
[ctrl_id.sub]	[btn_test.sub]	Primary subroutine name — "OnTestClick"
[ctrl_id.prop]	[btn_test.prop]	Primary property name — "caption"

Primary value variable per control type:

Control Type	Auto Variable	Value	Notes
Button	[btn_id.caption]	Caption text	
Label	[lbl_id.caption]	Caption text	
GroupBox	[grp_id.caption]	Caption text	
TextInput	[txt_id.value]	Current text	~300ms debounce
TextArea	[txa_id.value]	Current text	~300ms debounce
CheckBox	[chk_id.checked]	"true" / "false"	
Switch	[sw_id.checked]	"1" / "0"	
RadioButton	[rdo_id.checked]	"true" / "false"	each has own variable
Slider	[sld_id.value]	Numeric string	
ComboBox	[cbo_id.value]	Selected label text	
ListBox	[lst_id.value]	Selected item text	
ButtonBar	[bar_id.value]	Clicked button label	
TabControl	[tab_id.value]	Active tab index	
ProgressBar	[pb_id.value]	Numeric (0–100)	
Image	[img_id.filename]	Image file path	
Table	[tbl_id.row]	Clicked row index	also [tbl_id.cell]
Table	[tbl_id.col1] [tbl_id.col2] ...	Auto-parsed row columns	[tbl_id.colCount] = column count

Tip: Shared subroutine: Each control sets its own variables. Check each one: If "[rdo_a.checked]" "=" "true"

Page & Application Events

Startup · Shutdown · Page Enter · Page Exit — auto-generated stubs

NWID generates ready-to-fill subroutine stubs for every lifecycle event. These fire automatically. No wiring required — just add your logic above the Return.

Lifecycle order:

:NWID_OnReady → :Home_OnEnter → navigate → :Home_OnExit → :Settings_OnEnter → ... → :OnAppExit

Subroutine	When it fires	Key variables
:NWID_OnReady	Once — runtime fully loaded	[NWID.CanvasW] [NWID.CanvasH]
:Home_OnEnter	Every time Home becomes active	[NWID.PageName] [NWID.PageIndex]
:Home_OnExit	Just before leaving Home	[NWID.PageName] [NWID.PageIndex]
:NWID_OnPageChanged	After every page change	[NWID.PageName] [NWID.PageIndex]
:NWID_OnPropValue	When NWID_Get returns	[NWID.PropName] [NWID.PropValue]
:OnAppExit	Before WebView2 closes	—

Page Enter / Exit — what NWID generates for you

```
:Home_OnEnter
. Fires every time the Home page becomes active
. [NWID.PageName] = Home
Return

:Home_OnExit
. Fires just before leaving the Home page
Return
```

Tip: Page Enter vs NWID_OnPageChanged: Use :PageName_OnEnter for page-specific setup. Use :NWID_OnPageChanged for logic that applies to all page transitions.

Tip: OnReady vs OnEnter: :NWID_OnReady fires once at launch. :PageName_OnEnter fires every time that page is displayed, including the first time.

TEST 01 Basic Controls

Button — Confirm the Event Chain

1 Build in Designer

- 1 Open NWID — start NWID Designer.
- 2 Add Button — click Button in the palette, then click the canvas.
- 3 Set ID — in Properties: ID = btn_test
- 4 Set Caption — Caption = Test Connection
- 5 Wire event — Events tab: click ■ Add → Event: Left_Click → Action: GoSub → Value: OnTestClick

Properties Panel — what to set for each control

Control ID	Property	Value
btn_test	ID	btn_test
btn_test	Caption	Test Connection

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_test	Left_Click	GoSub	OnTestClick

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[btn_test.name]	btn_test — the ID of the button
[btn_test.event]	Left_Click
[btn_test.caption]	Caption text
[btn_test.sub]	OnTestClick
[btn_test.prop]	caption

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```

Step 1 – Verify the event chain:
:OnTestClick
AlertBox "Connection Test" "Name=[btn_test.name] Event=[btn_test.event]"
Return

Step 2 – Set a property:
:OnTestClick
NWID_Set "Rectangle1" "button:btn_test" "caption" "Clicked!"

```

```
Return
```

Alternative — using SetProp:

```
:OnTestClick  
SetVar "[btn_test.caption]" "Clicked!"  
SetVar "[SetObjectProperty]" "btn_test.caption"  
GoSub "SetProp"  
Return
```

Expected results:

- Step 1: Click the button — MessageBox shows Name=btn_test Event=Left_Click
- Step 2: Click the button — caption changes to "Clicked!"
- Confirms the full chain: browser click → NWID message → VNW subroutine

Note: This test has no complex logic — it proves the connection works and demonstrates both ways to set a property.

■ **PASS** ■ **FAIL** Notes: _____

TEST 02 Basic Controls

Label — Change Caption with NWID_Set

1 Build in Designer

- 1 Add Label — ID = lbl_status | Caption = Waiting...
- 2 Add Button — ID = btn_go | Caption = Click Me
- 3 Wire event — Events tab on btn_go: ■ Add → Event: Left_Click → Action: GoSub → Value: OnGoClick

Properties Panel — what to set for each control

Control ID	Property	Value
lbl_status	ID / Caption	lbl_status / Waiting...
btn_go	ID / Caption	btn_go / Click Me

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_go	Left_Click	GoSub	OnGoClick

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnGoClick
NWID_Set "Rectangle1" "label:lbl_status" "caption" "Button was clicked!"
Return
```

Alternative — using SetProp:

```
:OnGoClick
SetVar "[lbl_status.caption]" "Button was clicked!"
SetVar "[SetObjectProperty]" "lbl_status.caption"
GoSub "SetProp"
Return
```

Expected results:

- Click the button — label changes from "Waiting..." to "Button was clicked!" immediately

Note: NWID_Set takes four parameters: rectangle name, control ID, property name, new value. The property name for a label's text is "caption".

Note: Both methods produce the same result. The plugin command is more concise; the SetProp pattern uses the same variable names shown in Variable Bindings.

■ **PASS** ■ **FAIL** Notes: _____

TEST 03 Basic Controls

TextInput — Read Value with NWID_Get

1 Build in Designer

- 1 Add TextInput — ID = txt_name | Placeholder = Enter your name
- 2 Add Button — ID = btn_submit | Caption = Submit
- 3 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnSubmitClick
- 4 Add Label — ID = lbl_output | Caption = (leave empty)

Properties Panel — what to set for each control

Control ID	Property	Value
txt_name	ID / Placeholder	txt_name / Enter your name
btn_submit	ID / Caption	btn_submit / Submit
lbl_output	ID / Caption	lbl_output / (empty)

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_submit	Left_Click	GoSub	OnSubmitClick

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[NWID.PropValue]	Contains the value returned by NWID_Get — arrives in :NWID_OnPropValue

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnSubmitClick
NWID_Get "Rectangle1" "textInput:txt_name" "value" "[MyName]"
Return

:NWID_OnPropValue
NWID_Set "Rectangle1" "label:lbl_output" "caption" "Hello, [NWID.PropValue]!"
Return
```

Expected results:

- Type your name and click Submit — label shows Hello, [your name]!

Note: NWID_Get does not return a value directly. It fires :NWID_OnPropValue when the result arrives.

Note: :NWID_OnPropValue is pre-generated — it already appears in your exported subs file. Add logic inside the existing stub.

■ **PASS** ■ **FAIL** Notes: _____

TEST 04 Basic Controls

TextArea — Live Count and Set Content

1 Build in Designer

- 1 Add TextArea — ID = txa_notes | Placeholder = Type your notes here... | Height ~120px
- 2 Wire event — ■ Add → Event: Value_Changed → Action: GoSub → Value: OnNotesChanged
- 3 Add Button — ID = btn_clear | Caption = Clear
- 4 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnClearNotes
- 5 Add Label — ID = lbl_count | Caption = 0 characters

Properties Panel — what to set for each control

Control ID	Property	Value
txa_notes	ID / Placeholder / Height	txa_notes / Type your notes here... / ~120px
btn_clear	ID / Caption	btn_clear / Clear
lbl_count	ID / Caption	lbl_count / 0 characters

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
txa_notes	Value_Changed	GoSub	OnNotesChanged
btn_clear	Left_Click	GoSub	OnClearNotes

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[txa_notes.value]	Full current text of the TextArea — updated ~300ms after typing pauses

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnNotesChanged
StrLen "[txa_notes.value]" "[CharCount]"
NWID_Set "Rectangle1" "label:lbl_count" "caption" "[CharCount] characters"
Return

:OnClearNotes
NWID_Set "Rectangle1" "textArea:txa_notes" "value" ""
NWID_Set "Rectangle1" "label:lbl_count" "caption" "0 characters"
Return
```

Alternative — using SetProp:

```
:OnNotesChanged
StrLen "[txa_notes.value]" "[CharCount]"
SetVar "[lbl_count.caption]" "[CharCount] characters"
SetVar "[SetObjectProperty]" "lbl_count.caption"
GoSub "SetProp"
Return

:OnClearNotes
SetVar "[txa_notes.value]" ""
SetVar "[SetObjectProperty]" "txa_notes.value"
GoSub "SetProp"
SetVar "[lbl_count.caption]" "0 characters"
SetVar "[SetObjectProperty]" "lbl_count.caption"
GoSub "SetProp"
Return
```

Expected results:

- Type in the TextArea — character count updates after you pause typing (~300ms)
- Click Clear — TextArea empties and count resets to 0

■ **PASS** ■ **FAIL** Notes: _____

TEST 05 Basic Controls

CheckBox — True / False Branch

1 Build in Designer

- 1 Add CheckBox — ID = chk_agree | Caption = I agree to the terms | Default = unchecked
- 2 Wire event — ■ Add → Event: Value_Changed → Action: GoSub → Value: OnAgreeChanged
- 3 Add Button — ID = btn_proceed | Caption = Proceed
- 4 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnProceedClick
- 5 Add Label — ID = lbl_status | Caption = (empty)

Properties Panel — what to set for each control

Control ID	Property	Value
chk_agree	ID / Caption / Checked	chk_agree / I agree to the terms / false
btn_proceed	ID / Caption	btn_proceed / Proceed
lbl_status	ID / Caption	lbl_status / (empty)

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
chk_agree	Value_Changed	GoSub	OnAgreeChanged
btn_proceed	Left_Click	GoSub	OnProceedClick

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[chk_agree.checked]	"true" when checked, "false" when unchecked

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnAgreeChanged
If "[chk_agree.checked]" "=" "true"
NWID_Set "Rectangle1" "label:lbl_status" "caption" "Agreed"
Else
NWID_Set "Rectangle1" "label:lbl_status" "caption" "Not agreed"
EndIf
Return

:OnProceedClick
NWID_Get "Rectangle1" "checkBox:chk_agree" "checked" "[ChkState]"
Return
```

```
:NWID_OnPropValue
If "[NWID.PropValue]" "=" "true"
  MsgBox "OK" "Proceeding!"
Else
  MsgBox "Error" "Please check the box first."
EndIf
Return
```

Alternative — using SetProp:

```
:OnAgreeChanged
If "[chk_agree.checked]" "=" "true"
  SetVar "[lbl_status.caption]" "Agreed"
Else
  SetVar "[lbl_status.caption]" "Not agreed"
EndIf
SetVar "[SetObjectProperty]" "lbl_status.caption"
GoSub "SetProp"
Return
```

Expected results:

- Check the box — label shows "Agreed" immediately
- Click Proceed while unchecked → error dialog
- Click Proceed while checked → OK dialog

■ **PASS** ■ **FAIL** Notes: _____

TEST 06 Basic Controls

RadioButton Group

1 Build in Designer

- 1 Add 3 RadioButtons, one below the other
- 2 rdo_a — ID = rdo_a | Caption = Option A | Group = mygroup
- 3 rdo_b — ID = rdo_b | Caption = Option B | Group = mygroup
- 4 rdo_c — ID = rdo_c | Caption = Option C | Group = mygroup
- 5 Wire all three —  Add → Event: Value_Changed → Action: GoSub → Value: OnOptionPicked
- 6 Add Label — ID = lbl_choice | Caption = Nothing selected

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
rdo_a	Value_Changed	GoSub	OnOptionPicked
rdo_b	Value_Changed	GoSub	OnOptionPicked
rdo_c	Value_Changed	GoSub	OnOptionPicked

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[rdo_a.checked] / [rdo_b.checked] / [rdo_c.checked]	"true" for the selected button; others get "false"

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnOptionPicked
If "[rdo_a.checked]" "=" "true"
NWID_Set "Rectangle1" "label:lbl_choice" "caption" "Selected: rdo_a"
EndIf
If "[rdo_b.checked]" "=" "true"
NWID_Set "Rectangle1" "label:lbl_choice" "caption" "Selected: rdo_b"
EndIf
If "[rdo_c.checked]" "=" "true"
NWID_Set "Rectangle1" "label:lbl_choice" "caption" "Selected: rdo_c"
EndIf
Return
```

Alternative — using SetProp:

```
:OnOptionPicked
If "[rdo_a.checked]" "=" "true"
```



```
SetVar "[lbl_choice.caption]" "Selected: rdo_a"  
EndIf  
If "[rdo_b.checked]" "=" "true"  
SetVar "[lbl_choice.caption]" "Selected: rdo_b"  
EndIf  
If "[rdo_c.checked]" "=" "true"  
SetVar "[lbl_choice.caption]" "Selected: rdo_c"  
EndIf  
SetVar "[SetObjectProperty]" "lbl_choice.caption"  
GoSub "SetProp"  
Return
```

Expected results:

- Click Option A — label shows Selected: rdo_a
- Click Option B — Option A deselects automatically

Note: All three share the same Group name — that makes them mutually exclusive.

■ **PASS** ■ **FAIL** Notes: _____

TEST 07 Basic Controls

Switch Toggle

1 Build in Designer

- 1 Add Switch — ID = sw_power | Label On = ON | Label Off = OFF | Default = off
- 2 Wire event — ■ Add → Event: Value_Changed → Action: GoSub → Value: OnPowerToggle
- 3 Add Label — ID = lbl_power | Caption = Power is OFF

Properties Panel — what to set for each control

Control ID	Property	Value
sw_power	ID / Label On / Label Off / Default	sw_power / ON / OFF / off
lbl_power	Caption	Power is OFF

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
sw_power	Value_Changed	GoSub	OnPowerToggle

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[sw_power.checked]	"1" when switched ON, "0" when switched OFF — not "true"/"false" like CheckBox

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnPowerToggle
If "[sw_power.checked]" "=" "1"
NWID_Set "Rectangle1" "label:lbl_power" "caption" "Power is ON"
Else
NWID_Set "Rectangle1" "label:lbl_power" "caption" "Power is OFF"
EndIf
Return
```

Alternative — using SetProp:

```
:OnPowerToggle
If "[sw_power.checked]" "=" "1"
SetVar "[lbl_power.caption]" "Power is ON"
Else
SetVar "[lbl_power.caption]" "Power is OFF"
EndIf
SetVar "[SetObjectProperty]" "lbl_power.caption"
```

```
GoSub "SetProp"  
Return
```

Expected results:

- Toggle ON — label shows "Power is ON"
- Toggle OFF — label shows "Power is OFF"

Note: Switch uses "1" and "0" — CheckBox uses "true" and "false".

■ **PASS** ■ **FAIL** Notes: _____

TEST 08 Selection Controls

Slider — Live Value Updates

1 Build in Designer

- 1 Add Slider — ID = sld_vol | Min = 0 | Max = 100 | Step = 1 | Default = 50
- 2 Wire event — ■ Add → Event: Value_Changing → Action: GoSub → Value: OnVolumeChange
- 3 Add Button — ID = btn_set75 | Caption = Set to 75
- 4 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnSetVol75
- 5 Add Label — ID = lbl_vol | Caption = Volume: 50

Properties Panel — what to set for each control

Control ID	Property	Value
sld_vol	ID / Min / Max / Step / Default	sld_vol / 0 / 100 / 1 / 50
btn_set75	ID / Caption	btn_set75 / Set to 75
lbl_vol	Caption	Volume: 50

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
sld_vol	Value_Changing	GoSub	OnVolumeChange
btn_set75	Left_Click	GoSub	OnSetVol75

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[sld_vol.value]	Current slider position as a string — updates on every drag movement

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnVolumeChange
NWID_Set "Rectangle1" "label:lbl_vol" "caption" "Volume: [sld_vol.value]"
Return

:OnSetVol75
NWID_Set "Rectangle1" "slider:sld_vol" "value" "75"
NWID_Set "Rectangle1" "label:lbl_vol" "caption" "Volume: 75"
Return
```

Alternative — using SetProp:

```
:OnVolumeChange
SetVar "[lbl_vol.caption]" "Volume: [sld_vol.value]"
SetVar "[SetObjectProperty]" "lbl_vol.caption"
GoSub "SetProp"
Return

:OnSetVol75
SetVar "[sld_vol.value]" "75"
SetVar "[SetObjectProperty]" "sld_vol.value"
GoSub "SetProp"
SetVar "[lbl_vol.caption]" "Volume: 75"
SetVar "[SetObjectProperty]" "lbl_vol.caption"
GoSub "SetProp"
Return
```

Expected results:

- Drag the slider — label updates continuously
- Click "Set to 75" — slider thumb jumps to 75

■ **PASS** ■ **FAIL** Notes: _____

TEST 09 Selection Controls

ComboBox — Populate and Read

1 Build in Designer

- 1 Add ComboBox — ID = cbo_color | Placeholder = Choose a color
- 2 Wire event — ■ Add → Event: Sel_Changed → Action: GoSub → Value: OnColorPicked
- 3 Add Button — ID = btn_load | Caption = Load Colors
- 4 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnLoadColors
- 5 Add Label — ID = lbl_chosen | Caption = Nothing chosen

Properties Panel — what to set for each control

Control ID	Property	Value
cbo_color	ID / Placeholder	cbo_color / Choose a color
btn_load	ID / Caption	btn_load / Load Colors
lbl_chosen	Caption	Nothing chosen

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
cbo_color	Sel_Changed	GoSub	OnColorPicked
btn_load	Left_Click	GoSub	OnLoadColors

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[cbo_color.value]	Text label of the selected item — e.g. "Blue"

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnLoadColors
NWID_Set "Rectangle1" "comboBox:cbo_color" "items" "Red|Green|Blue|Yellow|Purple"
Return

:OnColorPicked
NWID_Set "Rectangle1" "label:lbl_chosen" "caption" "Chosen: [cbo_color.value]"
Return
```

Alternative — using SetProp:

```
:OnLoadColors
SetVar "[cbo_color.items]" "Red|Green|Blue|Yellow|Purple"
```

```
SetVar "[SetObjectProperty]" "cbo_color.items"  
GoSub "SetProp"  
Return  
  
:OnColorPicked  
SetVar "[lbl_chosen.caption]" "Chosen: [cbo_color.value]"  
SetVar "[SetObjectProperty]" "lbl_chosen.caption"  
GoSub "SetProp"  
Return
```

Expected results:

- Click Load Colors — dropdown has 5 items
- Select Blue — label shows Chosen: Blue

■ **PASS** ■ **FAIL** Notes: _____

TEST 10 Selection Controls

ListBox — Populate, Select, and Check by Code

1 Build in Designer

- 1 Add ListBox — ID = lst_fruits | Height ~150px | Check Box Mode = on
- 2 Wire event — ■ Add → Event: Sel_Changed → Action: GoSub → Value: OnFruitPicked
- 3 Add Button — ID = btn_load | Caption = Load Items
- 4 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnLoadFruits
- 5 Add Button — ID = btn_check | Caption = Check Cherry
- 6 Events tab on btn_check — ■ Add → Event: Left_Click → Action: Set Variable → Variable: [lst_fruits.checkedItems] → Value: Cherry
- 7 Add Label — ID = lbl_picked | Caption = (empty)

Properties Panel — what to set for each control

Control ID	Property	Value
lst_fruits	ID / Check Box Mode / Height	lst_fruits / On / ~150px
btn_load	ID / Caption	btn_load / Load Items
btn_check	ID / Caption / Events	btn_check / Check Cherry / Set Variable: [lst_fruits.checkedItems] = Cherry
lbl_picked	Caption	(empty)

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
lst_fruits	Sel_Changed	GoSub	OnFruitPicked
btn_load	Left_Click	GoSub	OnLoadFruits

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[lst_fruits.value]	Text of the item the user clicked

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnLoadFruits
NWID_Set "Rectangle1" "listBox:lst_fruits" "items" "Apple|Banana|Cherry|Date|Elderberry"
Return

:OnFruitPicked
```



```
NWID_Set "Rectangle1" "label:lbl_picked" "caption" "Picked: [lst_fruits.value]"  
Return
```

Expected results:

- Click Load Items — list fills with 5 items
- Click Cherry — label shows "Picked: Cherry"
- Click Check Cherry — Cherry checkbox checks by code

Note: The Set Variable action in the Events tab drives the control directly — no VNW subroutine needed for btn_check.

■ **PASS** ■ **FAIL** Notes: _____

TEST 11 Display Controls

ProgressBar

1 Build in Designer

- 1 Add ProgressBar — ID = pb_load | Min = 0 | Max = 100 | Show Label = on
- 2 Add Button 1 — ID = btn_half | Caption = 50% — wire Left_Click → OnHalf
- 3 Add Button 2 — ID = btn_full | Caption = 100% — wire Left_Click → OnFull
- 4 Add Button 3 — ID = btn_spin | Caption = Spin — wire Left_Click → OnSpin

Properties Panel — what to set for each control

Control ID	Property	Value
pb_load	ID / Min / Max / Show Label	pb_load / 0 / 100 / on
btn_half	ID / Caption	btn_half / 50%
btn_full	ID / Caption	btn_full / 100%
btn_spin	ID / Caption	btn_spin / Spin

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_half	Left_Click	GoSub	OnHalf
btn_full	Left_Click	GoSub	OnFull
btn_spin	Left_Click	GoSub	OnSpin

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnHalf
NWID_Set "Rectangle1" "progressBar:pb_load" "value" "50"
Return

:OnFull
NWID_Set "Rectangle1" "progressBar:pb_load" "value" "100"
Return

:OnSpin
NWID_Set "Rectangle1" "progressBar:pb_load" "indeterminate" "true"
Return
```

Expected results:

- Click 50% — bar fills halfway

- Click 100% — bar fills completely
- Click Spin — animated indeterminate mode

■ **PASS** ■ **FAIL** Notes: _____

TEST 12 Display Controls

Image — Load from File Path

1 Build in Designer

1 Add Image — ID = img_main | Fit = contain

2 Add Button — ID = btn_load | Caption = Load Image — wire Left_Click → OnLoadImage

Properties Panel — what to set for each control

Control ID	Property	Value
img_main	ID / Fit	img_main / contain
btn_load	ID / Caption	btn_load / Load Image

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_load	Left_Click	GoSub	OnLoadImage

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnLoadImage
. Replace photo.jpg with a real image in your publication folder
NWID_Set "Rectangle1" "image:img_main" "filename" "[PubDir]photo.jpg"
Return
```

Alternative — using SetProp:

```
:OnLoadImage
SetVar "[img_main.filename]" "[PubDir]photo.jpg"
SetVar "[SetObjectProperty]" "img_main.filename"
GoSub "SetProp"
Return
```

Expected results:

- Click Load Image — image appears in the control

Note: [PubDir] is a VNW built-in variable that expands to your publication folder path.

■ **PASS** ■ **FAIL** Notes: _____

TEST 13 Display Controls

Rectangle / Oval — Change Color

1 Build in Designer

- 1 Add Rectangle — ID = rect_1 | set any default fill color
- 2 Add Button 1 — ID = btn_red | Caption = Red — wire Left_Click → OnColorRed
- 3 Add Button 2 — ID = btn_blue | Caption = Blue — wire Left_Click → OnColorBlue
- 4 Add Button 3 — ID = btn_grad | Caption = Gradient — wire Left_Click → OnColorGrad

Properties Panel — what to set for each control

Control ID	Property	Value
rect_1	ID / Fill Color	rect_1 / any default
btn_red	ID / Caption	btn_red / Red
btn_blue	ID / Caption	btn_blue / Blue
btn_grad	ID / Caption	btn_grad / Gradient

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_red	Left_Click	GoSub	OnColorRed
btn_blue	Left_Click	GoSub	OnColorBlue
btn_grad	Left_Click	GoSub	OnColorGrad

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnColorRed
NWID_Set "Rectangle1" "rectangle:rect_1" "style.background" "#cc2244"
Return

:OnColorBlue
NWID_Set "Rectangle1" "rectangle:rect_1" "style.background" "#2244cc"
Return

:OnColorGrad
NWID_Set "Rectangle1" "rectangle:rect_1" "style.background"
"linear-gradient(135deg,#cc2244,#2244cc)"
Return
```

Alternative — using SetProp:

```
:OnColorRed
SetVar "[rect_1.style.background]" "#cc2244"
SetVar "[SetObjectProperty]" "rect_1.style.background"
GoSub "SetProp"
Return
```

Expected results:

- Click Red — shape turns red
- Click Gradient — red-to-blue diagonal gradient

Note: "style.background" accepts any CSS color: hex, rgba, named colors, or gradients.

■ **PASS** ■ **FAIL** Notes: _____

TEST 14 Layout Controls

TabControl + Containers — Tab Switching with Visibility

1 Build in Designer

- 1 Add TabControl — ID = tab_main | Tabs = General|Advanced|About
- 2 Wire event — Events tab: ■ Add → Event: Tab_Changed → Action: GoSub → Value: OnTabChanged
- 3 Add Container 1 — ID = ctr_general — position over the TabControl content area
- 4 Inside ctr_general: add a Label — Caption = This is the General tab
- 5 Right-click ctr_general → Hide on Canvas
- 6 Add Container 2 — ID = ctr_advanced — same area
- 7 Inside ctr_advanced: add a Label — Caption = Advanced settings here
- 8 Right-click ctr_advanced → Hide on Canvas
- 9 Add Container 3 — ID = ctr_about — same area
- 10 Inside ctr_about: add a Label — Caption = About this application
- 11 Right-click → Show All Hidden
- 12 Add Button — ID = btn_advanced | Caption = Go to Advanced — wire Left_Click → OnGoAdvanced

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
tab_main	Tab_Changed	GoSub	OnTabChanged
btn_advanced	Left_Click	GoSub	OnGoAdvanced

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[tab_main.value]	Zero-based index of the active tab: "0", "1", or "2"
[tab_main.name]	tab_main
[tab_main.event]	Tab_Changed

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnTabChanged
If "[tab_main.value]" "=" "0"
NWID_Visibility "Rectangle1" "container:ctr_general" "Show"
NWID_Visibility "Rectangle1" "container:ctr_advanced" "Hide"
NWID_Visibility "Rectangle1" "container:ctr_about" "Hide"
EndIf
```

```
If "[tab_main.value]" "=" "1"
NWID_Visibility "Rectangle1" "container:ctr_general" "Hide"
NWID_Visibility "Rectangle1" "container:ctr_advanced" "Show"
NWID_Visibility "Rectangle1" "container:ctr_about" "Hide"
EndIf
If "[tab_main.value]" "=" "2"
NWID_Visibility "Rectangle1" "container:ctr_general" "Hide"
NWID_Visibility "Rectangle1" "container:ctr_advanced" "Hide"
NWID_Visibility "Rectangle1" "container:ctr_about" "Show"
EndIf
Return

:OnGoAdvanced
NWID_Set "Rectangle1" "tabControl:tab_main" "selectedTab" "1"
Return
```

Expected results:

- Click General tab — ctr_general visible, others hidden
- Click Advanced tab — ctr_advanced visible
- Click Go to Advanced button — tab switches programmatically

Note: Each container acts as a "tab panel." Use NWID_Set with "selectedTab" and a zero-based index to switch tabs by code.

Tip: All three containers must overlap in the same position. Use Hide on Canvas after finishing each container.

■ **PASS** ■ **FAIL** Notes: _____

TEST 15 Data Controls

Table — Load Rows and Handle Row Click

1 Build in Designer

- 1 Add Table — ID = tbl_data | Columns: Name|Age|City (set in Props tab)
- 2 Wire event — ■ Add → Event: Row_Click → Action: GoSub → Value: OnRowClick
- 3 Add Button — ID = btn_load | Caption = Load Data — wire Left_Click → OnLoadData
- 4 Add Label — ID = lbl_row | Caption = Click a row

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
tbl_data	Row_Click	GoSub	OnRowClick
btn_load	Left_Click	GoSub	OnLoadData

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[tbl_data.row]	Zero-based index of the clicked row
[tbl_data.cell]	Text content of the specific cell clicked

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnLoadData
. Rows are pipe-separated. Cells within each row are comma-separated.
NWID_Set "Rectangle1" "table:tbl_data" "rows" "Alice,32,London|Bob,28,Paris|Carol,45,Berlin"
Return

:OnRowClick
NWID_Set "Rectangle1" "label:lbl_row" "caption" "Row [tbl_data.row] : [tbl_data.cell]"
Return
```

Expected results:

- Click Load Data — table fills with three rows
- Click any cell — label updates to show row index and clicked cell text

■ **PASS** ■ **FAIL** Notes: _____

TEST 16 Basic Controls

ButtonBar — Navigation / Action Bar

1 Build in Designer

- 1 Add ButtonBar — click Button Bar in the palette, then click the canvas.
- 2 Set ID = bar_nav | Items = Home|About|Contact
- 3 Use the  icon picker to prefix icons, e.g. Home: Home|About: About|Contact: Contact
- 4 Orientation = Horizontal | Style = Connected
- 5 Wire per-item events — Events tab: click  Add for each row.
- 6 Add Label — ID = lbl_section | Caption = (empty)

Properties Panel — what to set for each control

Control ID	Property	Value
bar_nav	ID / Items	bar_nav / Home About Contact
bar_nav	Orientation / Style	Horizontal / Connected
lbl_section	ID / Caption	lbl_section / (empty)

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
bar_nav	Item_Click Home	GoSub	OnNavHome
bar_nav	Item_Click About	GoSub	OnNavAbout
bar_nav	Item_Click Contact	GoSub	OnNavContact

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[bar_nav.value]	Label of the clicked button — e.g. "Home"
[bar_nav.name]	Control ID — bar_nav
[bar_nav.event]	Item_Click

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnNavHome
NWID_Set "Rectangle1" "label:lbl_section" "caption" "Welcome Home"
Return

:OnNavAbout
```

```
NWID_Set "Rectangle1" "label:lbl_section" "caption" "About Us"
Return

:OnNavContact
NWID_Set "Rectangle1" "label:lbl_section" "caption" "Contact Us"
Return
```

Alternative — using SetProp:

```
:OnNavHome
SetVar "[lbl_section.caption]" "Welcome Home"
SetVar "[SetObjectProperty]" "lbl_section.caption"
GoSub "SetProp"
Return
```

Expected results:

- Click Home — "Welcome Home"
- [bar_nav.value] contains the last clicked button label

Note: Leave Item as "— all items —" to fire the same GoSub for every button. You can mix both patterns.

Catch-all pattern:

```
:OnNavClick
NWID_Set "Rectangle1" "label:lbl_section" "caption" "Section: [bar_nav.value]"
Return
```

■ **PASS** ■ **FAIL** Notes: _____

TEST 17 Navigation

Multi-Page Navigation

1 Build in Designer

- 1 Name Page 1 "Home" — double-click the page tab
- 2 Add on P1 — lbl_home (Caption=This is the Home page), btn_tosettings (Caption=Go to Settings)
- 3 Wire btn_tosettings — ■ Add → Event: Left_Click → Action: GoSub → Value: OnGoSettings
- 4 Add Page 2 — click +, name it "Settings"
- 5 Add on P2 — lbl_settings (Caption=This is the Settings page), btn_home (Caption=Go Home)
- 6 Wire btn_home — ■ Add → Event: Left_Click → Action: GoSub → Value: OnGoHome

Properties Panel — what to set for each control

Control ID	Property	Value
btn_tosettings	ID / Caption / Page	btn_tosettings / Go to Settings / Home
btn_home	ID / Caption / Page	btn_home / Go Home / Settings

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_tosettings	Left_Click	GoSub	OnGoSettings
btn_home	Left_Click	GoSub	OnGoHome

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[NWID.PageName]	Name of the page that just became active
[NWID.PageIndex]	Zero-based index of the new page

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnGoSettings
NWID_Navigate "Rectangle1" "GoToPage" "Settings"
Return

:OnGoHome
NWID_Navigate "Rectangle1" "GoToPage" "Home"
Return

:NWID_OnPageChanged
```

```
. Fires automatically every time the page changes
AlertBox "Page Changed" "Now on: [NWID.PageName] (index [NWID.PageIndex])"
Return
```

Expected results:

- Click Go to Settings — page 2 appears, AlertBox confirms
- Click Go Home — page 1 appears

Note: Remove the AlertBox from :NWID_OnPageChanged once confirmed.

Note: NWID_Navigate also supports NextPage, PrevPage, and Reload.

Note: Page Enter/Exit: :Home_OnEnter and :Home_OnExit stubs are auto-generated.

■ **PASS** ■ **FAIL** **Notes:** _____

TEST 18 Navigation / NEW

Master Pages — Shared Navigation Button

1 Build in Designer

- 1 Create Master Page 1 — click the ■ tab. The canvas outline turns purple.
- 2 Add a Button on the master — ID = btn_nav | Caption = ≡ Menu
- 3 Wire event — ■ Add → Event: Left_Click → Action: GoSub → Value: OnNavClick
- 4 Add a Label on the master — ID = lbl_page | Caption = Page: Home
- 5 Switch to Page 1 — Right-click it → Assign Master → Master 1.
- 6 Add a Label on Page 1 — ID = lbl_content | Caption = Welcome to Home
- 7 Click + to add Page 2. Name it "Info". Right-click → Assign Master → Master 1.
- 8 Add a Label on Page 2 — ID = lbl_content2 | Caption = This is the Info page

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_nav	Left_Click	GoSub	OnNavClick

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

About Master Pages

A master page is a shared background layer. Controls placed on a master appear behind every standard page that uses it — ideal for navigation bars, logos, and recurring UI elements.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[btn_nav.name]	btn_nav — the master button that was clicked
[btn_nav.event]	Left_Click

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnNavClick
NWID_Navigate "Rectangle1" "NextPage" ""
Return

:NWID_OnPageChanged
NWID_Set "Rectangle1" "label:lbl_page" "caption" "Page: [NWID.PageName]"
Return
```

Alternative — using SetProp:

```
:NWID_OnPageChanged
SetVar "[lbl_page.caption]" "Page: [NWID.PageName]"
SetVar "[SetObjectProperty]" "lbl_page.caption"
GoSub "SetProp"
Return
```

Expected results:

- Both pages show the ≡ Menu button and "Page:" label from the master
- Click ≡ Menu — navigates and updates label

Note: Master controls cannot be selected from a standard page. Switch to the master tab to edit them.

■ **PASS** ■ **FAIL** Notes: _____

TEST 19 Navigation / NEW

Page Enter / Exit Events

1 Build in Designer

- 1 Name Page 1 "Home" and Page 2 "Settings".
- 2 Add a Label on Page 1 — ID = lbl_home_status | Caption = (empty)
- 3 Add a Label on Page 2 — ID = lbl_settings_status | Caption = (empty)
- 4 Add a Button on Page 1 — ID = btn_go | Caption = Go to Settings
- 5 Wire btn_go — Left_Click → OnGoSettings
- 6 Add a Button on Page 2 — ID = btn_back | Caption = Go Home
- 7 Wire btn_back — Left_Click → OnGoHome
- 8 Wire Page 1 events — click the Page 1 tab, open the Events tab:
- 9 ■ Add → Event: Page_Enter → GoSub → OnHomeEnter
- 10 ■ Add → Event: Page_Exit → GoSub → OnHomeExit
- 11 Wire Page 2 events:
- 12 ■ Add → Event: Page_Enter → GoSub → OnSettingsEnter
- 13 ■ Add → Event: Page_Exit → GoSub → OnSettingsExit

Properties Panel — what to set for each control

Control ID	Property	Value
lbl_home_status	ID / Caption / Page	lbl_home_status / (empty) / Home
lbl_settings_status	ID / Caption / Page	lbl_settings_status / (empty) / Settings
btn_go	ID / Caption / Page	btn_go / Go to Settings / Home
btn_back	ID / Caption / Page	btn_back / Go Home / Settings

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_go	Left_Click	GoSub	OnGoSettings
btn_back	Left_Click	GoSub	OnGoHome
Page 1 (Home)	Page_Enter	GoSub	OnHomeEnter
Page 1 (Home)	Page_Exit	GoSub	OnHomeExit
Page 2 (Settings)	Page_Enter	GoSub	OnSettingsEnter
Page 2 (Settings)	Page_Exit	GoSub	OnSettingsExit

How: In the Events tab, click ■ Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

4 Script + Test

NWID sets these per-control variables before calling your subroutine:

Variable	Contains
[NWID.PageName]	Name of the page being entered or exited
[NWID.PageIndex]	Zero-based index of that page

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnGoSettings
NWID_Navigate "Rectangle1" "GoToPage" "Settings"
Return

:OnGoHome
NWID_Navigate "Rectangle1" "GoToPage" "Home"
Return

:OnHomeEnter
NWID_Set "Rectangle1" "label:lbl_home_status" "caption" "Home entered: [NWID.PageName]"
Return

:OnHomeExit
NWID_Set "Rectangle1" "label:lbl_home_status" "caption" "Leaving Home..."
Return

:OnSettingsEnter
NWID_Set "Rectangle1" "label:lbl_settings_status" "caption" "Settings entered: [NWID.PageName]"
Return

:OnSettingsExit
NWID_Set "Rectangle1" "label:lbl_settings_status" "caption" "Leaving Settings..."
Return
```

Alternative — using SetProp:

```
:OnHomeEnter
SetVar "[lbl_home_status.caption]" "Home entered: [NWID.PageName]"
SetVar "[SetObjectProperty]" "lbl_home_status.caption"
GoSub "SetProp"
Return
```

Expected results:

- On load — OnHomeEnter fires, label shows "Home entered: Home"
- Click Go to Settings — OnHomeExit fires first, then OnSettingsEnter
- Click Go Home — OnSettingsExit fires, then OnHomeEnter fires again

Note: Page events are wired through the Events tab on the page itself — click the page tab, then open the Events tab.

Note: NWID also generates auto-named stubs (:Home_OnEnter, :Home_OnExit) in the subs file. Both the user-wired GoSub and the auto-generated stub fire independently. Use one approach or the other — if you wire a GoSub, leave the auto-generated stub empty.

■ PASS ■ FAIL Notes: _____

TEST 20 File I/O

filename — Image, RichText, TextArea

1 Build in Designer

- 1 Add Image — ID = img_photo | Fit = contain | ~200×150 px
- 2 Add RichText — ID = rtx_content | Height ~120 px
- 3 Add TextArea — ID = txa_readme | Height ~120 px
- 4 Add Button — ID = btn_loadimg | Caption = Load Image — wire Left_Click → OnLoadImg
- 5 Add Button — ID = btn_loadhtml | Caption = Load HTML — wire Left_Click → OnLoadHtml
- 6 Add Button — ID = btn_loadtxt | Caption = Load Text — wire Left_Click → OnLoadTxt

Properties Panel — what to set for each control

Control ID	Property	Value
img_photo	ID / Fit	img_photo / contain
rtx_content	ID / Height	rtx_content / ~120 px
txa_readme	ID / Height	txa_readme / ~120 px
btn_loadimg	ID / Caption	btn_loadimg / Load Image
btn_loadhtml	ID / Caption	btn_loadhtml / Load HTML
btn_loadtxt	ID / Caption	btn_loadtxt / Load Text

Events Tab — wire these subroutine names

Control ID	Event	Action	Subroutine
btn_loadimg	Left_Click	GoSub	OnLoadImg
btn_loadhtml	Left_Click	GoSub	OnLoadHtml
btn_loadtxt	Left_Click	GoSub	OnLoadTxt

How: In the Events tab, click  Add for each row. Set Event to the event name, Action to GoSub, and enter the subroutine name in the Value field.

2 Export

Click **Export** in NWID. This generates two files: **runtime.html** and **runtime_subs.txt**. Copy both to your VNW publication folder.

Note: After importing the subs file in VNW, locate the stub subroutine(s) listed in the Events table above — they will appear at the top of the subs file with a Return and no code. That is where you write your plugin commands.

Prepare test files — place these in your publication folder:

- **photo.jpg** — Any JPEG image
- **article.html** — Small HTML file
- **readme.txt** — A plain text file

4 Script + Test

Write these lines inside the stub subroutines in your VNW Subroutine Action:

Method 1 — Plugin commands:

```
:OnLoadImg
NWID_Set "Rectangle1" "image:img_photo" "filename" "[PubDir]photo.jpg"
```

```
Return
```

```
:OnLoadHtml
```

```
NWID_Set "Rectangle1" "richText:rtx_content" "filename" "[PubDir]article.html"
```

```
Return
```

```
:OnLoadTxt
```

```
NWID_Set "Rectangle1" "textArea:txa_readme" "filename" "[PubDir]readme.txt"
```

```
Return
```

Expected results:

- Click Load Image — photo.jpg displays in the Image control
- Click Load HTML — article.html renders with formatting
- Click Load Text — readme.txt appears as plain text

Note: filename automatically converts Windows paths to file:/// URLs. You can also pass a URL.

Note: Use NWID_Get with filename to read back the last path that was set.

■ **PASS** ■ **FAIL** Notes: _____

Designer Feature: Canvas Border

The Canvas Properties panel includes a Border group. You can add a decorative border to the entire page canvas.

Setting	Description
Where to find it	Canvas Properties → Border group — between Background and Grid & Snap.
Style	Dropdown: — none —, solid, dashed, dotted, double.
Color	Color picker + hex field. Default: #2d6a9f.
Width	Pixel width (0–20). Border is inset (box-sizing:border-box).
Radius	Corner rounding (0–200). Works independently of Style.
Master pages	Border belongs to individual standard pages. Not shown on master pages.

Note: Canvas border is a designer property — baked into exported HTML. Not changed at runtime via `NWID_Set`.

Designer Feature: Master Page Dimensions

Master pages now have their own Dimensions tab in Canvas Properties. Assigned standard pages inherit the master's width and height automatically.

Feature	Description
Master Dimensions tab	Click the master tab, then click the canvas background. Width and Height fields appear.
Assigned pages inherit	Assigning a master updates the page's width and height to match immediately.
Locked on assigned pages	Disabled Width/Height fields with a purple "■ Dimensions set by [Master Name]" label.
Unassigned pages	Keep their own independent dimensions, fully editable.
On unassign	Page keeps the master's size as its own — it does not revert.

Tip: Set the master dimensions first. Before placing controls, set Width and Height to match your largest intended page size.

Testing Checklist

Check each item off as you complete it.

Setup

- ezEdge rectangle named Rectangle1 exists Notes: _____
- NWIDCompanion plugin installed Notes: _____
- Subs file imported Notes: _____

Basic Controls

- TEST 01 Button — event chain + SetProperty Notes: _____
- TEST 02 Label caption Notes: _____
- TEST 03 NWID_Get reads TextInput Notes: _____
- TEST 04 TextArea count Notes: _____
- TEST 05 CheckBox branch Notes: _____
- TEST 06 RadioButton group Notes: _____
- TEST 07 Switch "1"/"0" Notes: _____

Selection Controls

- TEST 08 Slider — live value Notes: _____
- TEST 09 ComboBox — populate and read Notes: _____
- TEST 10 ListBox — Set Variable action Notes: _____

Display + Layout

- TEST 11 ProgressBar Notes: _____
- TEST 12 Image — filename Notes: _____
- TEST 13 Rectangle — color + gradient Notes: _____
- TEST 14 TabControl switches Notes: _____

Data

- TEST 15 Table rows + Row_Click Notes: _____

Navigation

- TEST 16 ButtonBar — per-item GoSub Notes: _____
- TEST 17 NWID_Navigate GoToPage Notes: _____
- TEST 18 Master page Notes: _____
- TEST 19 Page Enter/Exit Notes: _____

File I/O

- TEST 20 filename loads Image, RichText, TextArea Notes: _____

New Features

- Canvas Border Notes: _____
- Master page dimensions Notes: _____
- Per-control variables Notes: _____
- SetProperty pattern Notes: _____