

Application note

How to display a page to indicate communication error

AN00201

Rev B (EN)

Java script allows CP600 HMIs to perform a vast array of user-defined functions with ease



Introduction

The CP600 range of intelligent HMI panels is able to communicate with other peripherals (e.g. AC500 PLCs, ABB motion products) via a selection of communication protocols. This application note details how these HMIs can be programmed to display a particular page from the HMI project whenever a communication error is present. For general guidance on the use of Panel Builder 600 please refer to ABB manual 2CDC159007M0201.

To configure a CP600 HMI to communicate with an ABB motion control product via Modbus RTU/TCP requires Panel Builder 600 version 1.80.00.34 (or later). Please contact your local Sales office if you need to update your existing version of this software.

The following application notes may also be useful references:

- AN00198 : Integrated Modbus Support
- AN00199 : Connecting ABB CP600 HMIs to ABB Motion Products via Modbus TCP
- AN00200 : Connecting ABB CP600 HMIs to ABB Motion Products via Modbus RTU

For the purposes of this application note we will assume a NextMove ESB-2 motion controller has been connected to a CP620 HMI and is using the ABB Modbus RTU protocol for communication

Overview of functionality

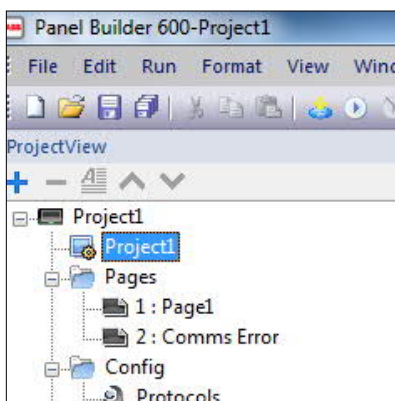
To enable the HMI to detect a loss of communication with the connected PLC / Motion Control product we need to configure the following elements:

- A page to display in the event of the error being detected
- An internal variable used to store the last known communication status
- A scheduled task that calls a Java script function to check the current communication status
- The Java script function code

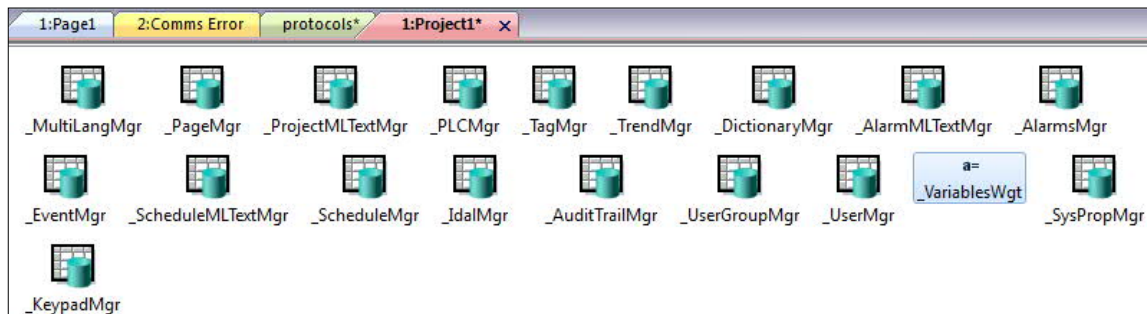
It is assumed the reader is already familiar with creating HMI pages using the Panel Builder 600 software. Refer to the Panel Builder 600 software manual or the help file integrated within the software if more details are required on how to create and edit HMI pages.

Creating an internal variable

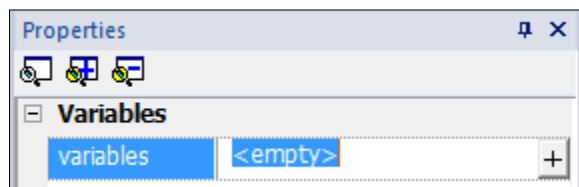
Double click the project configuration icon shown in the ProjectView pane (highlighted in our screenshot below).



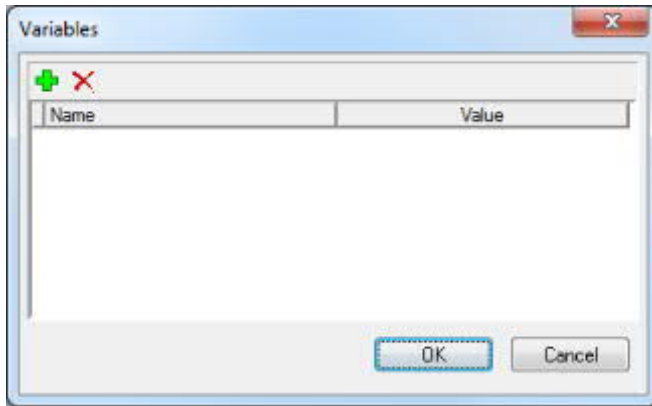
Our project is called Project1 and we have already added a second page to this project and named this page “Comms Error”. Having double-clicked the project configuration icon the software now displays the projects configuration screen in the right pane. We need to click on the Variables Widget icon as shown below so it becomes highlighted:



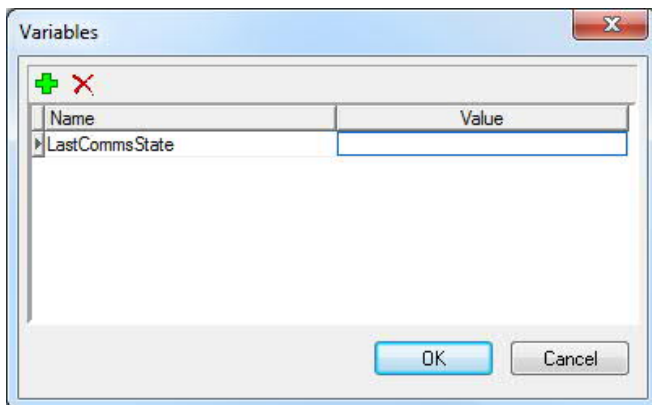
In the Properties window click on the “<empty>” text....a “+” button will appear on the text box as shown below:



Now click on the “+” button and then click “Select Variable”. A dialog appears listing the variables defined in the project (there are none at present of course)...



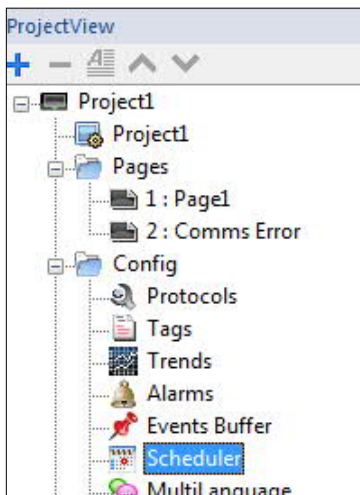
Click on the green plus symbol and create a variable called LastCommsState



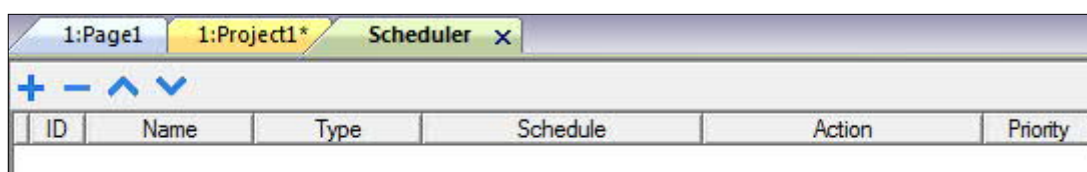
Click OK and the Properties dialog for the project variables will now show our variable called LastCommsState exists.

Creating a scheduled task

Double click the Scheduler icon shown in the ProjectView pane (highlighted in our screenshot below).



The software now displays a list of all of the programmed scheduled tasks (this list is empty at present as we're yet to create one)...



Click on the blue plus symbol to add a scheduled task.

The software will add some default information about the scheduled task automatically...

ID	Name	Type	Schedule	Action	Priority
1	Schedule1	Recurring	Daily, Time, 13:57		Medium

Click on the text in the Schedule column (Daily, Time, 13:57).....we need to change this to a schedule type that is performed every 2 seconds. The screenshot below shows how to configure the settings to achieve this...

Schedule1 Properties

Type: Every Date: N/A

Mode: Time Time: 00:00:02

Condition: | Location:

Actions: ...

On startup

Enable schedule

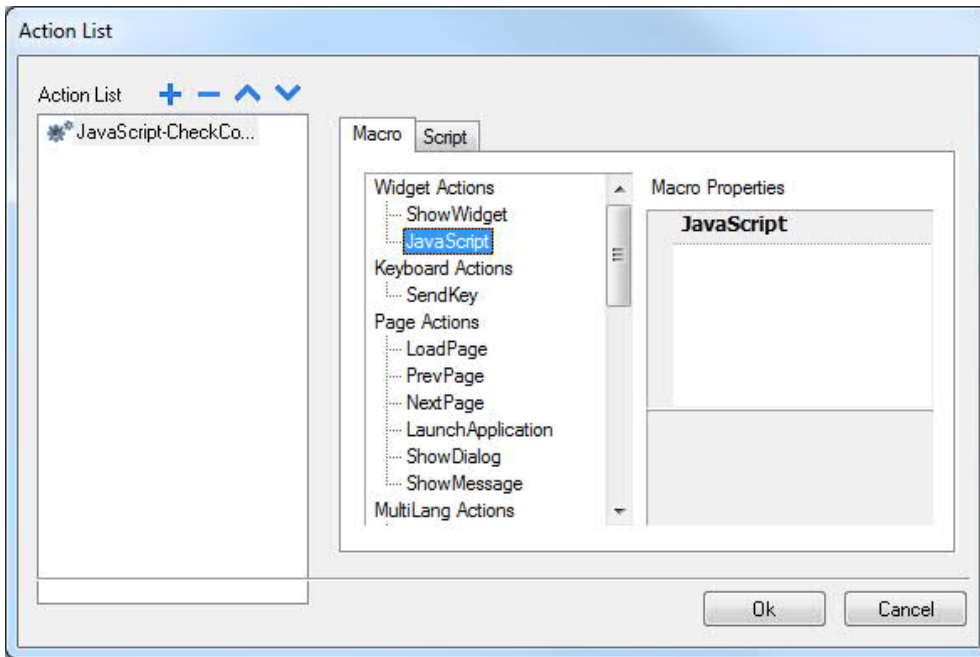
Ok Cancel

Click OK and our Scheduler list now updates to reflect these changes.

It is now worth giving this scheduled task a name that indicates what the task does....click on the text in the Name column (Schedule1) and rename this CheckComms (as shown below)...

ID	Name	Type	Schedule	Action	Priority
1	CheckComms	Recurring	Every, Time, 00:00:02		Medium

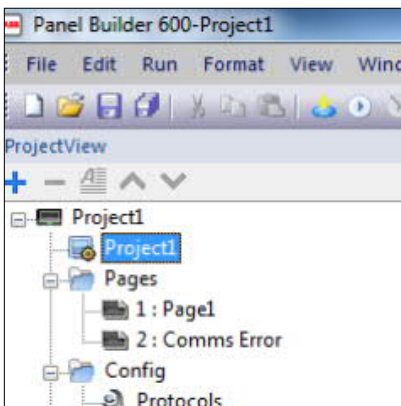
Now click in the Action text box, a dialog will appear allowing us to program what action will be taken every 2 seconds...We need to program the scheduled task to call a Java script function so on the Macro tab select 'Javascript' as shown below...



The software automatically lists an action called “Javascript-CheckComms”. The name for the Java script function is inherited automatically from the name we gave the scheduled task. Click OK to accept this.

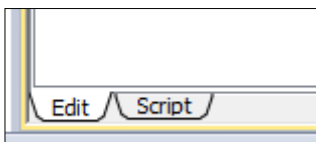
Creating the Java script code to display a page when a communication error occurs

Double-click the project configuration icon again so that the Variables widget icon and the rest of the Mgr icons are displayed

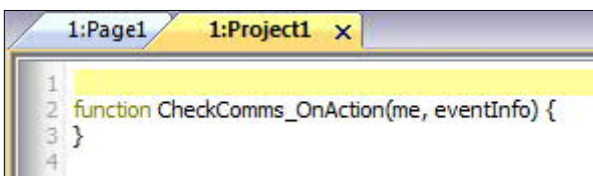


again.

In the project window, at the bottom left corner, you will see we are currently viewing the ‘Edit’ tab. Click on the ‘Script’ tab instead...



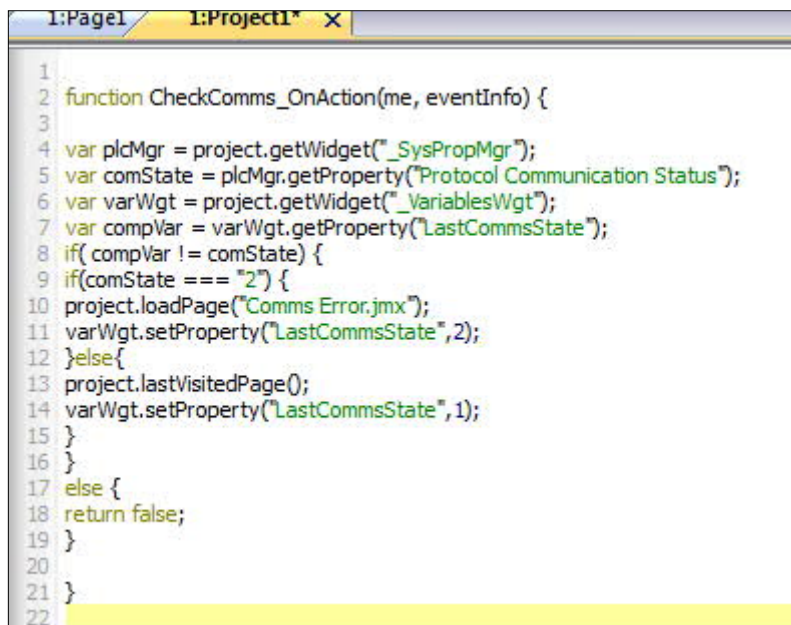
The software now displays the Java scripts that are present in our project. As we saw in the previous step, the software automatically created a script called CheckComms (named after our schedule) and the code window now displays the outline for this script.



Click in the code editor, insert some blank lines before the final “}” character and then either paste or write the following lines of code so they become part of the function...

```
var plcMgr = project.getWidget("_SysPropMgr");
var comState = plcMgr.getProperty("Protocol Communication Status");
var varWgt = project.getWidget("_VariablesWgt");
var compVar = varWgt.getProperty("LastCommsState");
if( compVar != comState) {
if(comState === "2") {
project.loadPage("Comms Error.jmx");
varWgt.setProperty("LastCommsState",2);
}else{
project.lastVisitedPage();
varWgt.setProperty("LastCommsState",1);
}
}
else {
return false;
}
```

Your Java script should now look something like the following...

A screenshot of a code editor window titled "I:Project1" with a tab "I:Page1". The editor contains the following JavaScript code:

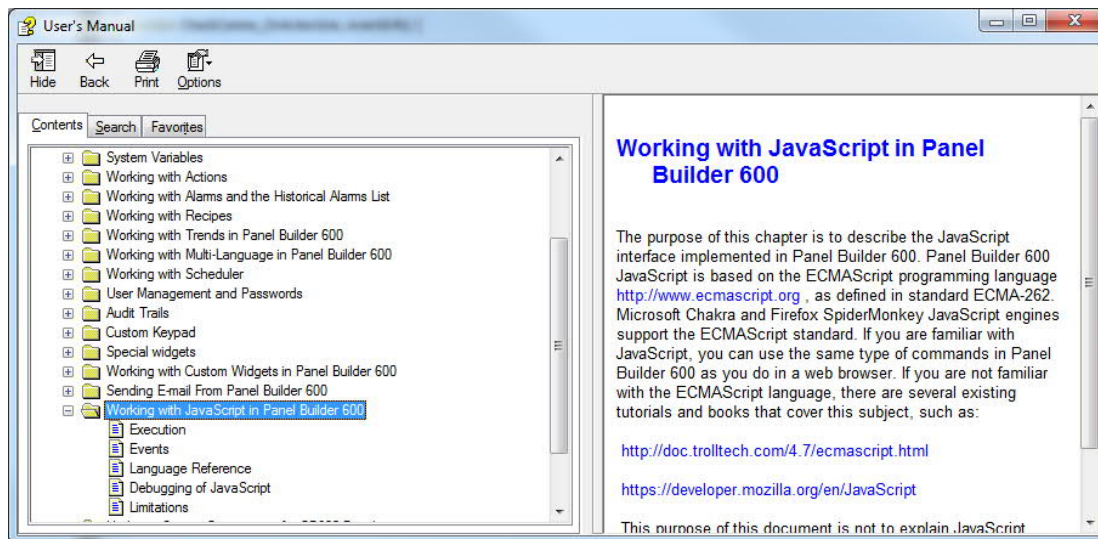
```
1
2 function CheckComms_OnAction(me, eventInfo) {
3
4 var plcMgr = project.getWidget("_SysPropMgr");
5 var comState = plcMgr.getProperty("Protocol Communication Status");
6 var varWgt = project.getWidget("_VariablesWgt");
7 var compVar = varWgt.getProperty("LastCommsState");
8 if( compVar != comState) {
9 if(comState === "2") {
10 project.loadPage("Comms Error.jmx");
11 varWgt.setProperty("LastCommsState",2);
12 }else{
13 project.lastVisitedPage();
14 varWgt.setProperty("LastCommsState",1);
15 }
16 }
17 else {
18 return false;
19 }
20 }
21 }
22
```

This script compares the current communication status (read via the inbuilt system tag “Protocol Communication Status”) against our defined internal variable “LastCommsState”. If they are different it decides whether to display the Comms Error screen (via the project.loadpage function) or to return to the last page that was displayed (via the project.lastVisitedPage function).

These actions are performed every 2 seconds (as setup by our scheduled task) and if a communication error is detected our Comms Error page is displayed automatically...



For more information about Java script and the available Java script functions within Panel Builder 600 please refer to the “Working with Javascript in Panel Builder 600” topic in the help file included with the software.



Contact us

For more information please contact your local ABB representative or one of the following:

new.abb.com/motion
new.abb.com/drives
new.abb.com/drivespartners
new.abb.com/PLC

© Copyright 2012 ABB. All rights reserved.
 Specifications subject to change without notice.