

iSee Ui

URCap

User manual

For UR robots

v1.4

Original Instructions
Date: 2023, August

The information contained herein is the property of Industrial Cobotics BV and shall not be reproduced in whole or in part without prior written approval of Industrial Cobotics BV. The information herein is subject to change without notice and should not be construed as a commitment by Industrial Cobotics BV. This document is periodically reviewed and revised. Industrial Cobotics BV assumes no responsibility for any errors or omissions in this document. Copyright © 2022 by Industrial Cobotics BV.

Contents

1	General information	1
1.1	Purpose	1
1.2	Company details	1
1.3	Disclaimer	1
2	Compatibility	2
3	Installation	2
4	iSee Ui installation node	4
4.1	Configuration of the iSee Ui interface	4
4.1.1	Load a new iSee Ui interface	4
4.1.2	Activate an iSee Ui interface	5
4.1.3	Delete an iSee Ui interface	5
4.1.4	Daemon	6
4.2	Settings	6
4.2.1	Update rate	6
4.2.2	Priority on variable value change	7
4.3	License	7
4.3.1	Free version	7
4.3.2	Trial and full version	8
4.4	About	9
5	iSee Ui operation	10
5.1	Opening the iSee Ui interface	10
5.2	Closing the iSee Ui interface	11
6	Variables and IO's	13
6.1	Variable initialization	13
6.2	Variable use in UR program	13
7	iSee Ui program node	14
7.1	Parameterization of iSee Ui program node	15
7.2	Program node iSee Ui actions	16
7.2.1	Reset	16
7.2.2	Open	16
7.2.3	Close	16
7.3	Parameterization of general properties	17
7.3.1	Visible	17
7.3.2	Location	18
7.3.3	Size	19
7.3.4	Enabled	20

7.3.5	Text	21
7.3.6	Background color	21
7.3.7	Foreground color	22
7.4	Parameterization of item specific properties	23
7.4.1	PasswordField specific properties/actions	23
7.4.2	List specific properties/actions	25
7.4.3	Combobox specific properties/actions	27
7.5	General behaviour of the program node	28
8	Error handling	30
9	Caution with service	31
10	iSee Ui examples	32
10.1	Simple iSee Ui programming	32
10.2	Advanced iSee Ui programming	34

1. General information

1.1 Purpose

The purpose of the iSee Ui URCap is to provide an easy and user-friendly means for using and interacting with a custom iSee Ui user interface that can be created by means of the iSee Ui Builder.

1.2 Company details

Industrial Robotics BV
Atomveldstraat 10 - bus 2
9450 Haaltert
Belgium
Tel: +32 27 93 10 41
mail: info@industrialrobotics.be

1.3 Disclaimer

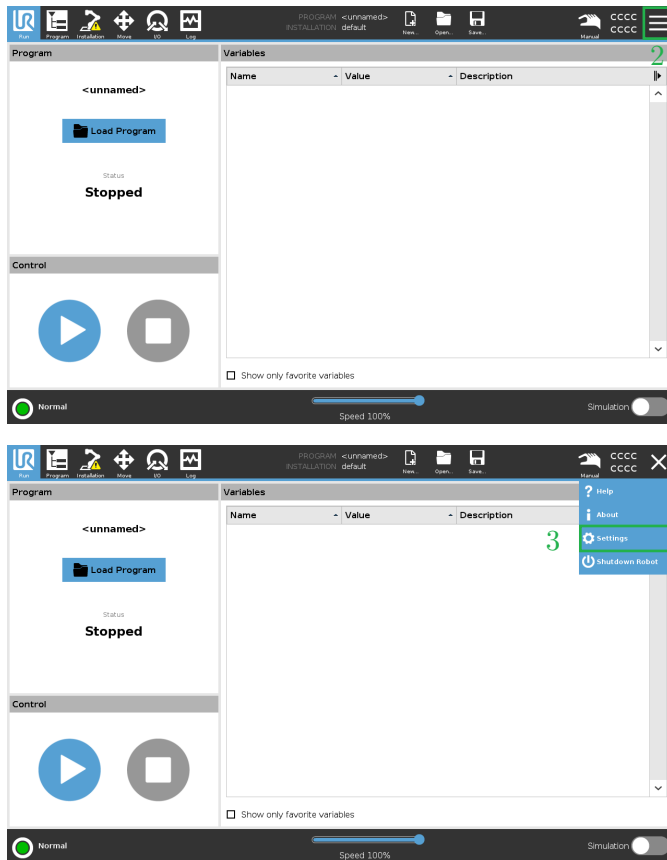
Industrial Robotics continues to improve reliability and performance of its products, and therefore reserves the right to upgrade the product without warning. Industrial Robotics takes every care that the contents of this manual are precise and correct, but takes no responsibility for any errors or missing information.

2. Compatibility

The iSee Ui URCap is compatible with the UR e-series, PolyScope version 5.6+. This version of the URCap is also compatible with the iSee Ui Builder v1.5 software.

3. Installation

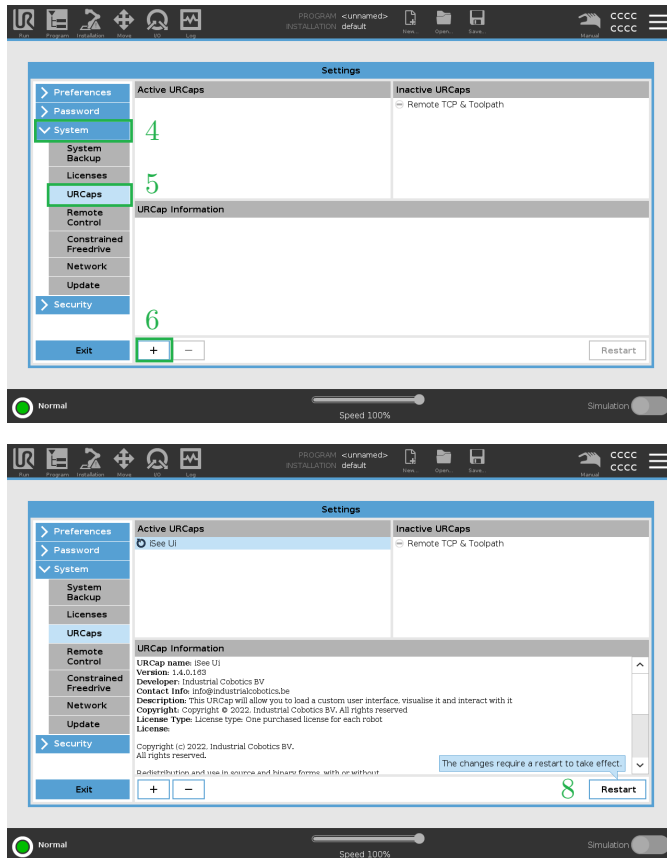
To install the iSee Ui URCap, follow these steps:



1. Put the iSee Ui URCap on a USB and plug this into one of the USB ports

2. Go to the settings by pressing the menu button in the top right corner

3. Press the settings button



4. Navigate to the “System” section
5. Select “URCaps”
6. Press the “+” button
7. Navigate to the .urcap file located on your USB, and open this file
8. Once opened, press the restart button. The controller will restart and install the URCap

4. iSee Ui installation node

The iSee Ui installation node can be loaded by selecting the “iSee Ui” node in the “Installation” tab of the robot, under the “URCaps” section, as depicted in figure 4.1.

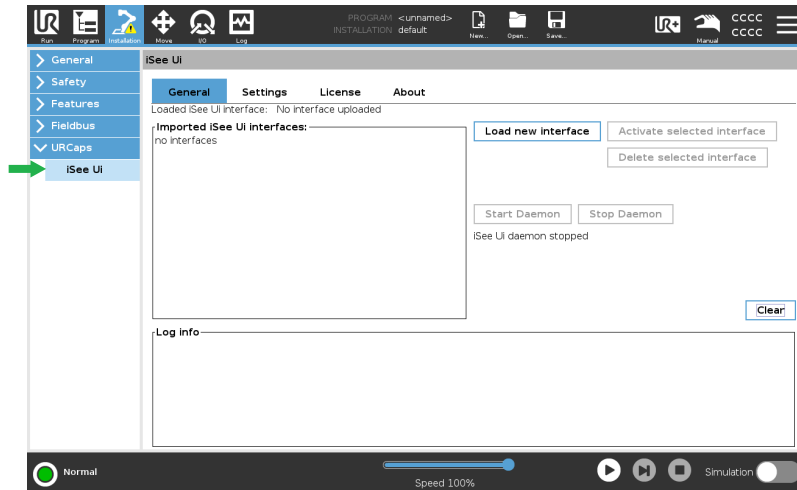


Figure 4.1: iSee Ui installation node, general tab

4.1 Configuration of the iSee Ui interface

In the “General” tab (see figure 4.1) of the “iSee Ui” node, the configuration of the iSee Ui URCap can be done. Here, the name of the current active interface will always be displayed at the top of the page.

4.1.1 Load a new iSee Ui interface

To load a new iSee Ui interface, or upload a new version of a previously loaded interface, the .isui file can be stored on a USB stick, the USB stick can be connected to one of the USB ports and then the “Load new interface” button can be pressed. A file explorer window will open, allowing the user to navigate to the location where the .isui file is stored.

Once opened, the selected .isui file will be added to the “Imported iSee Ui interfaces” list or will replace the older version. The file will also be stored on the UR controller storage, thus the USB will no longer be needed.

In the “Imported iSee Ui interfaces” list, an overview is given of all the .isui files that have been uploaded and stored onto the UR controller.

Remark: An iSee Ui interface name needs to be unique. If a different iSee Ui interface, with the same name, is loaded, the new interface will replace the existing one.

4.1.2 Activate an iSee Ui interface

To activate an interface, select the desired interface from the “Imported iSee Ui interfaces” list. If the interface is not listed, see section 4.1.1 for loading a new interface. Once the interface is selected, the button “Activate selected interface” will be enabled to activate the interface.

The name of the activated interface will be visible at the top of the page. When activating an interface that uses variables, the URcap will determine if any of these variables have not yet been created. Variables that are missing will be reported in the “Log info” list, as can be seen in figure 4.2. In chapter 6, the use of variables by an interface will be discussed in detail.

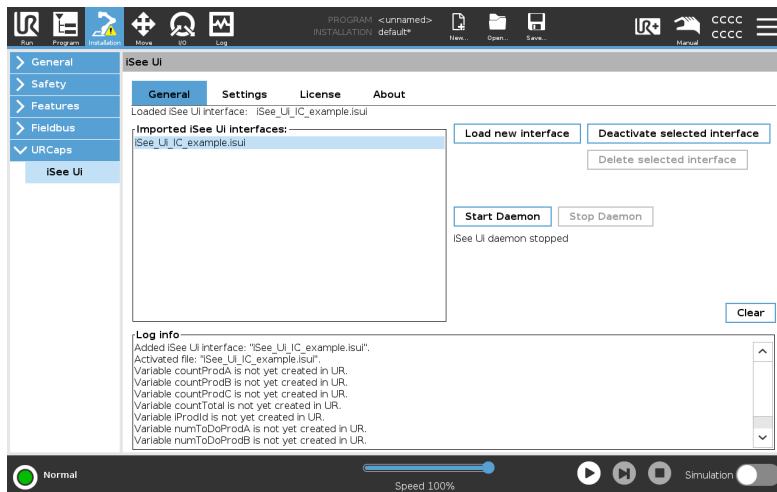


Figure 4.2: Activation of an iSee Ui interface

After activating a new interface, save the installation file such that changes are stored. The next time the installation file is loaded, the new interface will be activated automatically.

4.1.3 Delete an iSee Ui interface

In order to remove an interface from the “Imported iSee Ui interfaces” list, select the interface in the list and press the “Delete selected interface” button. Deleting an interface will also delete it from the UR controller storage. Thus, the next time the same interface needs to be activated, it will first have to be loaded again as described in section 4.1.1.

If the selected interface is also the one that is currently active, the interface will first have to be deactivated before it can be removed. This can be done by first pressing the

“Deactivate selected interface” button. Once pressed, the status at the top will denote that no interface is loaded, and the button “Delete selected interface” will be enabled.

4.1.4 Daemon

In order for the iSee Ui URCap to be able to communicate with the running program, the daemon service needs to be running. The daemon can only be started if an iSee Ui interface has been activated. If the iSee Ui interface is deactivated, the daemon will automatically stop running.

To start the daemon interface, press the “Start Daemon” button. Below the button, the current status of the Daemon can be viewed. The status can be:

- Starting
- Running
- Stopping
- Stopped
- Failed
- Error

If the status of the daemon is “Failed” or “Error”, waiting a few seconds and then restarting the daemon should solve the problem. If the status remains “Failed” or “Error”, contact support@industrialrobotics.be.

4.2 Settings

In the “Settings” tab of the iSee Ui installation node (see figure 4.3), settings for the behaviour of the iSee Ui can be configured. The settings will be discussed separately in the following sections.

4.2.1 Update rate

With the “Update rate (ms)” setting, the frequency for the update of the iSee Ui can be configured. The update rate determines the time between two successive update cycles of the iSee Ui.

In an update cycle, both the iSee Ui interface and the program communicate to each other all the changes that have occurred since the previous update cycle. These changes can be:

- Variable value changes
- IO state/value changes
- Property changes for items on the iSee Ui interface

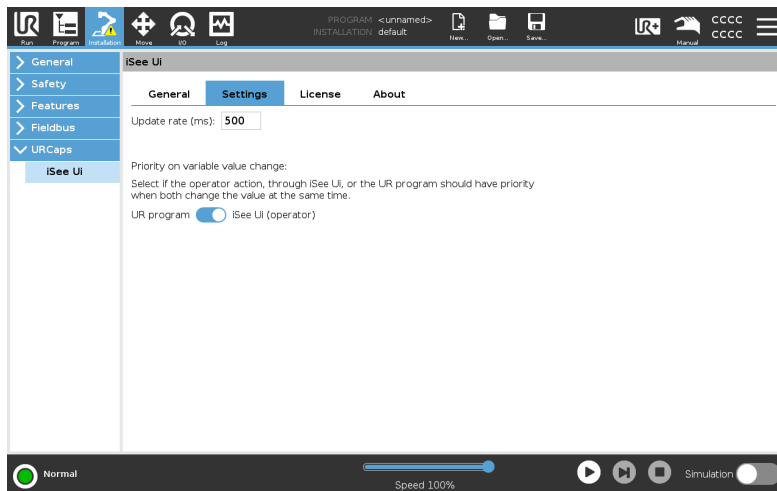


Figure 4.3: iSee Ui installation node, settings tab

By lowering the time between update cycles, the iSee Ui interface will react faster to operator interactions and value changes. The interface will thus seem to be running smoother. Due to the faster response, the CPU load of the iSee Ui on the UR controller could increase, which could prove problematic if other high-load tasks or URcaps are being used. A value lower than 100ms cannot be used for the update rate.

To lower the CPU load of the iSee Ui, a higher value can be chosen for the update rate, increasing the time between update cycles and thus making the iSee Ui react slower to operator interactions and value changes.

4.2.2 Priority on variable value change

With the “Priority on variable value change” a write conflict between the running program and the iSee Ui can be avoided. When both parties have a value change for the same variable, they both would communicate their own new value to each other. To avoid this conflict, priority can be given to either the iSee Ui or the running UR program. In the event that both have a value change for the same variable, only the value determined by the side that has priority will be considered as the new value for that variable.

4.3 License

The license status can be viewed under the “License” tab, as can be seen in figure 4.4.

4.3.1 Free version

If no valid license is available, the iSee Ui URcap can be used for **free** with interface files that contain maximally 8 items.

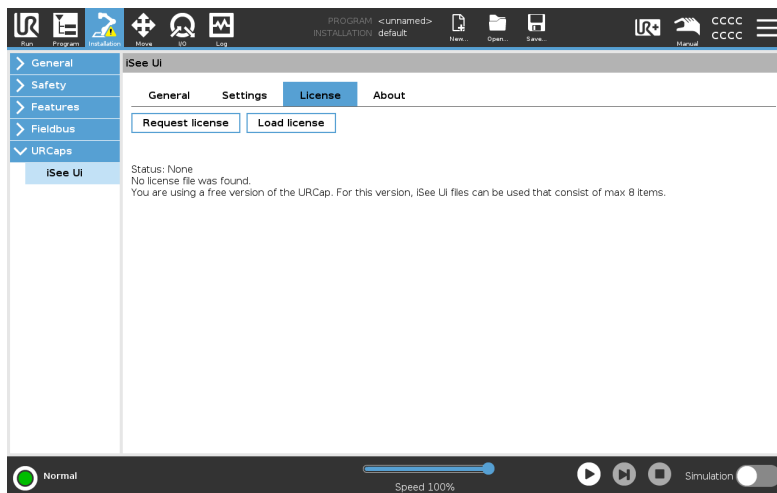
4.3.2 Trial and full version

In order to use the iSee Ui with interface files of more than 8 items, a license needs to be activated. A trial license of 2 weeks can be requested, or a full license can be purchased. First, a license request file will need to be generated. This can be done by connecting a USB to one of the USB ports. Then, press the “Request license” button. The license request file will automatically be generated onto the USB drive.

The license request file can then be mailed to info@industrialrobotics.be.

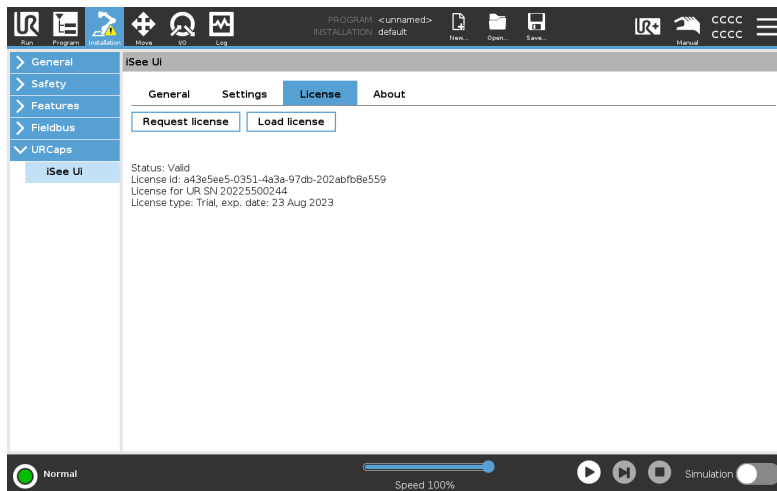
A license file will then be sent back that is valid for that particular robot. This license file can then be put onto a USB drive, the USB drive can be connected with one of the USB ports, and the button “Load license” can be pressed.

When a correct license file is found, the license information will be displayed, as can be seen in figure 4.4b.



(a) No license

Figure 4.4: iSee Ui installation node, license tab



(b) Valid license

Figure 4.4: iSee Ui installation node, license tab

When using a full URCap license, interface files created using a trial license for the iSee Ui Builder cannot be activated. These can only be used when the URCap has a trial license as well. In order to use these files, load them in the iSee Ui Builder using a full license.

4.4 About

In the “About” tab of the iSee Ui installation node (see figure 4.5), the version of the iSee Ui URCap can be consulted as well as a short description about the URCap.



Figure 4.5: iSee Ui installation node, about tab

5. iSee Ui operation

The custom-made iSee Ui interface acts as an interface between the operator and the running UR program. When the UR program is not running, the iSee Ui interface cannot communicate with the program and will thus not respond to any operator interaction. How this interface looks, what is shown, or what the operator can do, depends entirely on the activated iSee Ui interface which is built using the iSee Ui Builder software. Consult the iSee Ui Builder manual for a detailed description of how to create such an interface, and what items are available.

The iSee Ui interface is only capable to change values of predefined variables or IO's. Which object is set to which value or state, is preconfigured using the iSee Ui Builder software. The iSee Ui interface will thus not define robot actions. For this, the programmer can use UR programming to easily define the desired result of an interaction.

In chapter 6, the use of variables and IO's between the iSee Ui interface and the robot program is discussed in detail. In section 10.1, a simple example of an iSee Ui program is discussed.

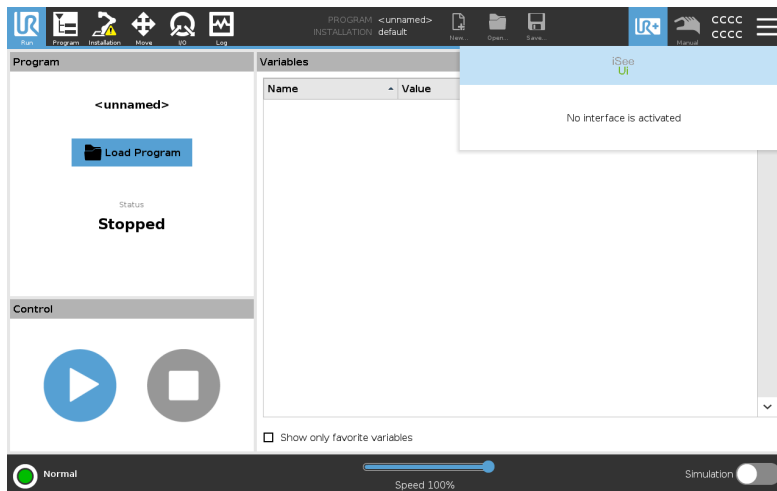
For more advanced use, chapter 7 explains how the iSee Ui program node can be used, as well as the more advanced possibilities it provides. In section 10.2, a more advanced example can be found.

5.1 Opening the iSee Ui interface

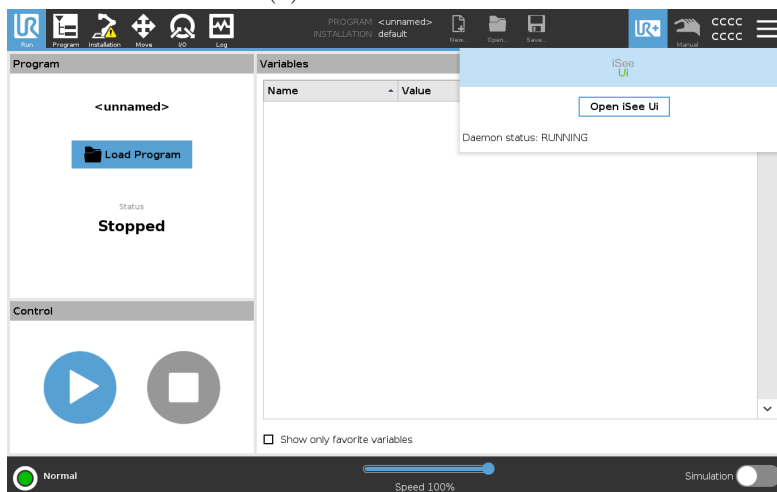
The iSee Ui interface can manually be opened by pressing the “UR+” toolbar icon as shown in figure 5.1. When multiple URCaps are installed, select the iSee Ui tab.

If the toolbar mentions that no interface is activated, as shown in figure 5.1a, check the license status (section 4.3) and activate an interface (section 4.1.2).

When an interface is activated, the iSee Ui interface can be opened by pressing the “Open iSee Ui” button, and the daemon status can be consulted.



(a) No active interface



(b) Active interface and daemon running


Figure 5.1: iSee Ui toolbar

The iSee Ui interface can also be opened automatically by using the iSee Ui program node. This will be discussed in more detail in chapter 7.

5.2 Closing the iSee Ui interface

Once the iSee Ui interface is opened, the operator can manually close the interface by pressing the “UR+” toolbar icon. This method will only work when the “iSee Ui” tab in the toolbar was selected. When multiple URCaps are installed, and the iSee Ui interface was automatically opened using the program node functionality, the toolbar might be in use by one of these other URCaps and thus pressing the “UR+” button will not close the interface.

The interface can also be closed by pressing on the interface and dragging upwards toward the black toolbar. This functionality only works if the interface itself is pressed, and not any of the items placed on the interface.

The easiest method for closing the interface is by pressing the  button in the top right corner, as can be seen in figure 5.2.

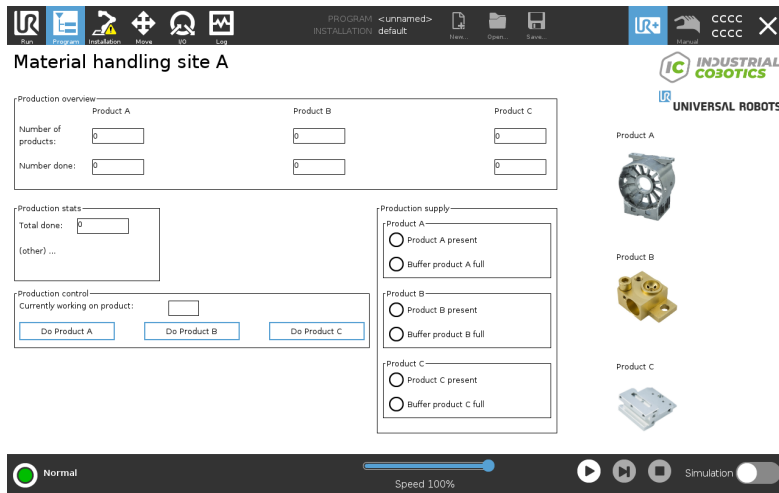


Figure 5.2: Example of an open interface

6. Variables and IO's

As mentioned in chapter 5, the operator interactions set predefined values or states to predefined variables or IO's. For the IO's, the iSee Ui uses the default name of the IO, renaming the IO will thus not have an effect on the operation.

For variables, the variable name needs to be the same in order for the communication between the iSee Ui and the program to work.

The iSee Ui will not create missing variables, but will, upon activating the .isui file, mention which variables have not yet been created.

6.1 Variable initialization

The variables used by the interface have to be created and initialized in the robot program. The robot program has total control over the initial value of all the variables, even the ones that are used in iSee Ui interface items where a value is entered by the operator.

The programmer can thus define the initial value for each variable and choose to either create the variable as an installation or program variable. In case the value should not be reset when the program is restarted, the variable is best initialized in the “Set initial variable values” or “Variables Setup” section of the robot program. However, the value will still be reset when the controller is restarted. To avoid this, consider using installation variables.

6.2 Variable use in UR program

Variables can be used to realize all kinds of interactions, some examples are:

- Parameterization. The operator can enter a value for a parameter, such as number of objects to pick, number of objects available, number of seconds to wait, ...
- Switch-case selection. The operator can enter the value of the case the robot needs to perform
- Event triggering. Variables can be used to trigger an event when the operator presses a button. For instance, to start a sequence, pause, stop, ...

Depending on the interaction that needs to be programmed, keep in mind the source for the new value of the variable. As discussed in section 4.2.2, if both the iSee Ui interface and the UR program write a value to the same variable at the same time, the actual value stored will be the value for the side that has priority.

In case the value should only be altered by the operator, using the iSee Ui interface, and not by the UR program itself, the UR program should only read the value that is stored in that variable.

7. iSee Ui program node

In this chapter, the iSee Ui program node and all its capabilities will be discussed. In general, the iSee Ui program node can be used to alter a property of an item on the iSee Ui interface.

Remark: When setting property changes, be aware of the initial situation when starting the program and opening the iSee Ui interface.

The first time the iSee Ui interface is opened, after activating the corresponding .isui file, the items will have their default properties as set in the iSee Ui Builder. However, once an iSee Ui program node is executed, and a property has been changed, the iSee Ui will not automatically reset the changes when the program is restarted. Thus, the programmer will also have to reset the properties or set them to the desired value in the “BeforeStart” sequence of the UR program.

First, the parameterization of the program node is discussed in section 7.1.

Section 7.2 discusses the actions that can be programmed for the iSee Ui interface itself. In section 7.3, the general properties of iSee Ui items and their parameterization is discussed.

More item specific properties and the parameterization thereof is discussed in section 7.4.

This chapter will then end with a description of the general behaviour of the parameterized iSee Ui program node, discussed in section 7.5.

The iSee Ui program node can be inserted by pressing the node under the “URCaps” tab as seen in figure 7.1.

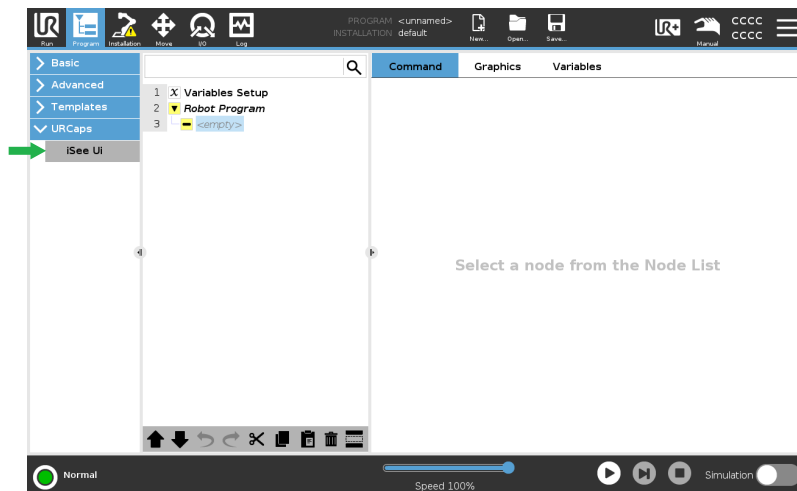


Figure 7.1: Insert iSee Ui program node

7.1 Parameterization of iSee Ui program node

Once the node has been added to the program, the node needs to be fully parameterized before it can be used. As mentioned, this node will provide the possibility to adjust a property of or do an action for an item on the iSee Ui interface. Every object is specified by a type and an ID or name. Figure 7.2 shows that after inserting the node, first the item type needs to be selected for the item for which an action or a change of a property is desired.

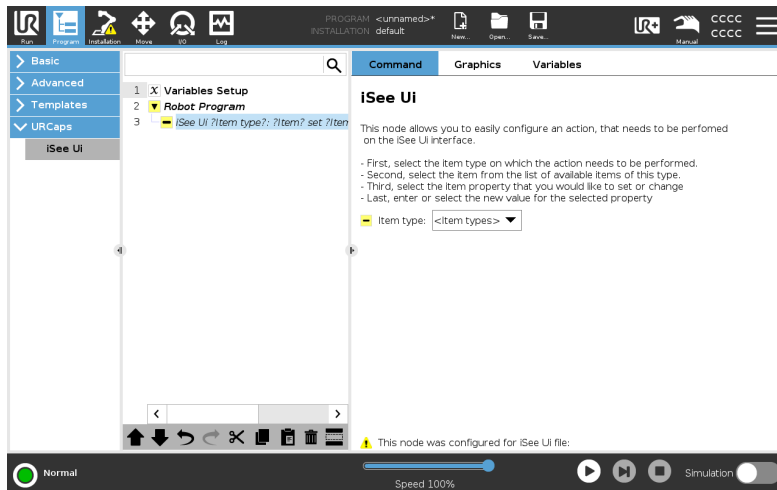


Figure 7.2: Empty iSee Ui program node

In the item type list, only the item types for which there are items on the iSee Ui interface will be listed. E.g. if the activated iSee Ui interface doesn't include an item of the type "PushButton", then "PushButton" will not be listed.

After the item type has been selected, an additional list selection will appear (see figure 7.4). Here are all the items listed, of the selected item type, that exist on the active iSee Ui interface. These items are denoted by their ID or name that was given in the iSee Ui Builder. From this list, the desired item can be selected.

After selecting the item, an additional list selection will appear where the desired property or action can be selected. Every item has some general properties, which are discussed in section 7.3. Some items have additional properties and/or actions unique to that item type, these properties and/or actions with the corresponding item types are discussed in section 7.4.

7.2 Program node iSee Ui actions

The program node also provides some actions for the iSee Ui interface itself. These can be programmed by selecting “iSee Ui” in the item type list. When selecting the item type “iSee Ui” from the list, the list to select an item will not appear. The iSee Ui interface has the following actions, as can be seen in figure 7.3:

- Reset
- Open
- Close

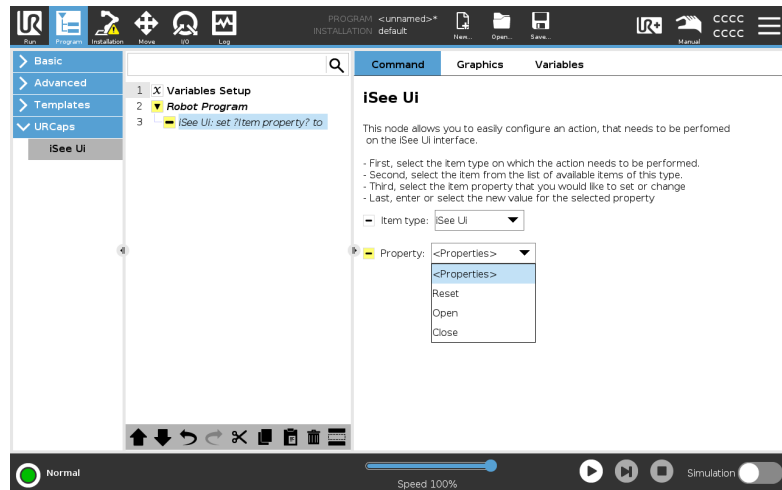


Figure 7.3: iSee Ui properties/actions

7.2.1 Reset

The “Reset” property allows the programmer to reset the iSee Ui. Thus, reset all the properties of all the items to their default value, which is the value configured in the iSee Ui Builder.

7.2.2 Open

The “Open” property allows the programmer to program the iSee Ui interface to open whenever this action is performed by the UR program. In case an automatic opening of the iSee Ui interface is desired, the programmer can add this action in the “BeforeStart” sequence. In this way, the operator no longer needs to open the iSee Ui interface through the “UR+” icon, since it will automatically be opened once the program has been started.

7.2.3 Close

The “Close” property is similar to the “Open” property, only now the iSee Ui interface will be closed when such an action is performed. This action would be of interest if, in a certain situation, the program is to be halted. Then the programmer can use this action such that the iSee Ui interface is automatically closed before stopping the UR program itself.

7.3 Parameterization of general properties

General properties that all item types have:

- Visible
- Location
- Size

General properties that not all item types have:

- Enabled
- Text
- Background color
- Foreground color

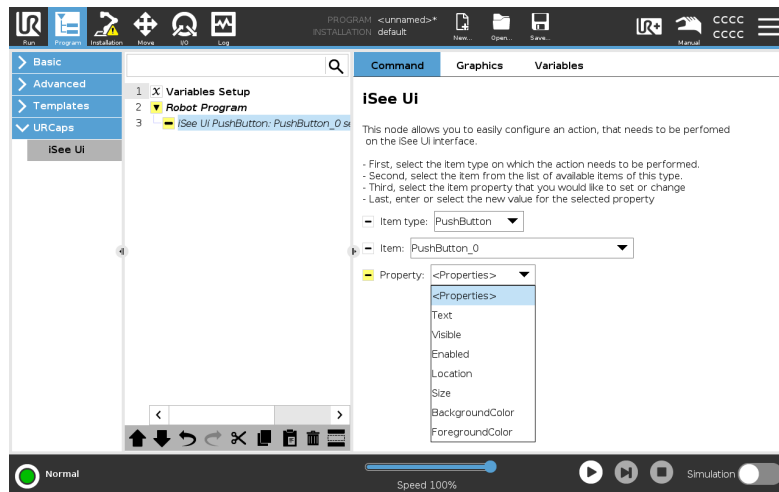


Figure 7.4: General properties

Each of these general properties will be explained in the next sections. For every property, an initial value is shown which is the value for the property, for that particular item, that has been given in the iSee Ui Builder. This is also the value of the property that the item will have when the iSee Ui interface is activated or reset using the “Reset” program node action discussed in section 7.2.1.

7.3.1 Visible

By changing the value of the “Visible” property, an item can be made visible (viewable) or invisible. Moreover, it allows the programmer to change this property at some point during the robot execution. This property is especially interesting if the programmer wants to have items appear or disappear when a certain situation occurs. It also allows

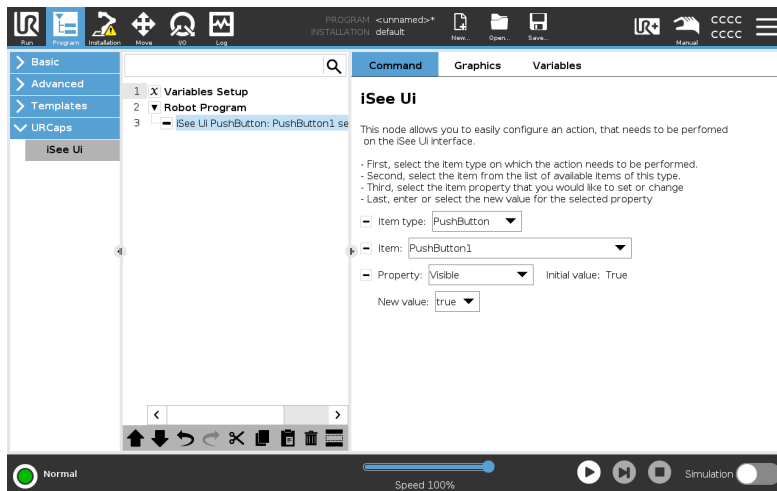


Figure 7.5: Visible property

keeping the iSee Ui interface as simple as possible by setting items, that do not always need to be visible, invisible. Then when the situation arises that these items need to be viewable, the programmer can program them to become visible, and then at a later moment back to invisible.

7.3.2 Location

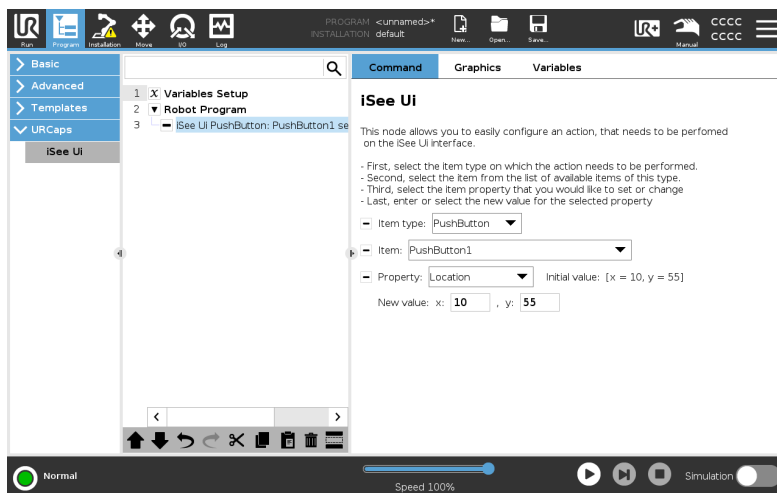


Figure 7.6: Location property

This property allows the programmer to move items to a new location on the iSee Ui interface. One such example could be to create a 'menu opening' behaviour. Once pressed, the menu button can be displaced and a panel of several buttons can be made

visible. This example is also discussed in section 10.2.

The position of the item is defined by an 'x' and 'y' coordinate. The 'x' coordinate denotes the horizontal pixel distance from the left side of the item to the left side of the items's parent. The 'y' coordinate in its turn denotes the vertical pixel distance from the top side of the item to the top side of the item's parent. The item's parent is either the main iSee Ui interface or a panel item. This relation can also be seen in figure 7.7.

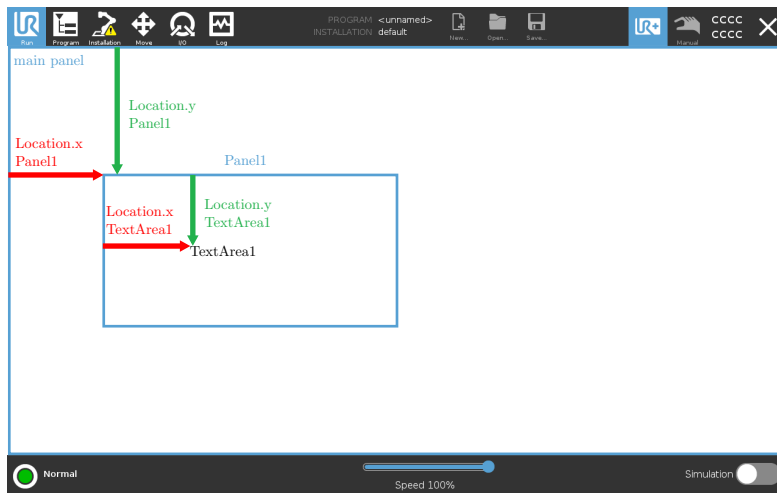


Figure 7.7: Item location definition

7.3.3 Size

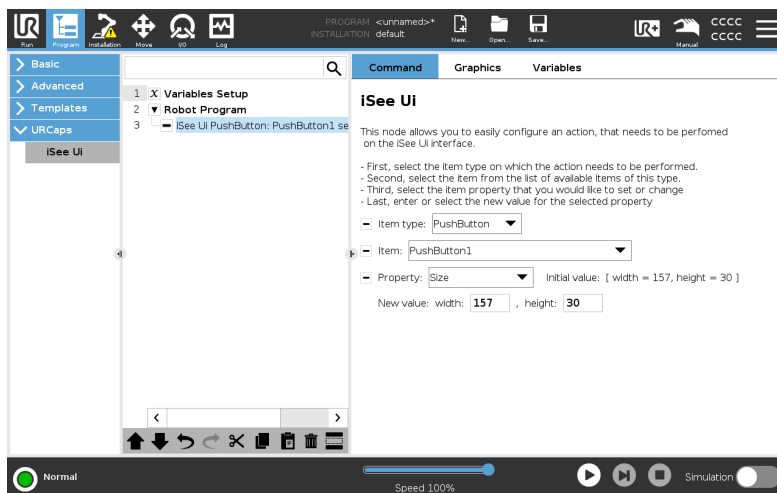


Figure 7.8: Size property

By changing the value of the “Size” property, an item can be given a different size than the initial one. The property can be used for all kinds of things, such as enlarging an item for which the “Text” property changed to a larger text.

7.3.4 Enabled

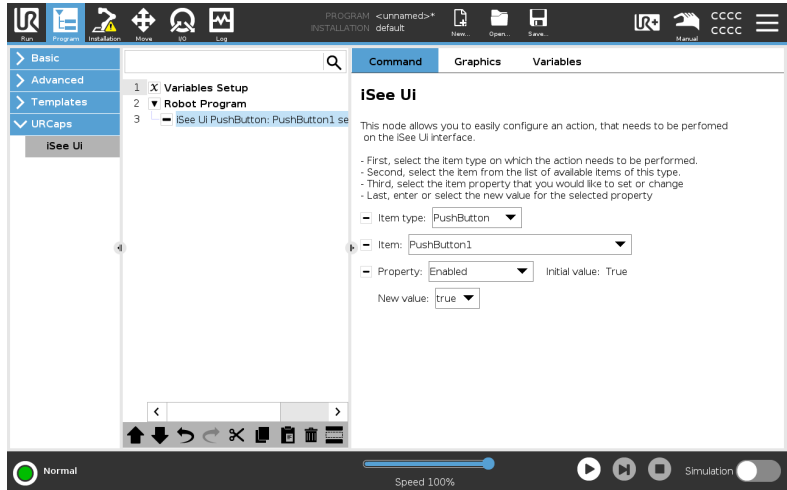


Figure 7.9: Enabled property

The “Enabled” property allows the programmer to change the enabled state of an item. By changing this state, it can be made clear to the operator if a certain item can be interacted with or not in a current situation. For instance, a “Stop” button can be made disabled by default, and as soon as the program is running the button can be made enabled.

The item types for which the “Enabled” property can be used are:

- TextField
- PasswordField
- PushButton
- RadioButton
- CheckBox
- Label
- List
- Combobox
- Slider

7.3.5 Text

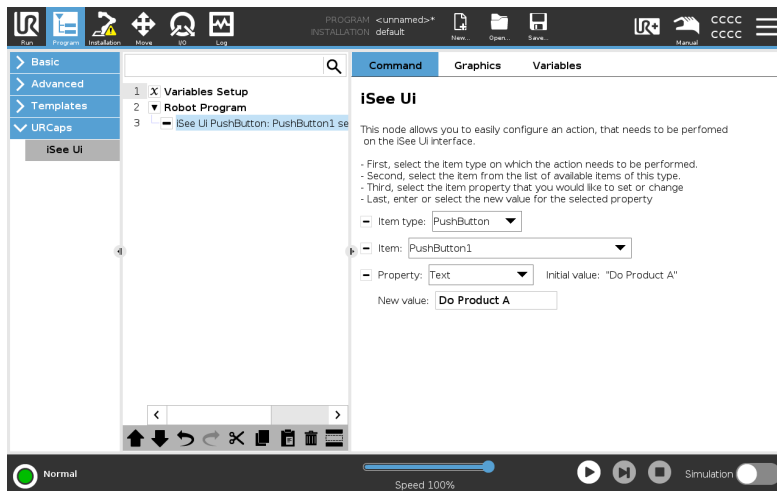


Figure 7.10: Text property

By changing the value of the “Text” property, the text for the selected item can be changed at some point during the program execution. The property can only be changed if the item uses a fixed “Text” property. If the item is using a variable “Text” property by use of an iSee Ui expression, then the property will not be able to be adjusted by the program node. In this case, the text can be adjusted by changing the value of the object used in the iSee Ui expression.

The item types for which the “Text” property can be used are:

- TextArea
- PushButton
- RadioButton
- CheckBox

7.3.6 Background color

The background color of an item can be changed during runtime by changing the “BackgroundColor” property of an item.

The item types for which the “BackgroundColor” property can be used are:

- Panel
- LayeredPane
- TextArea

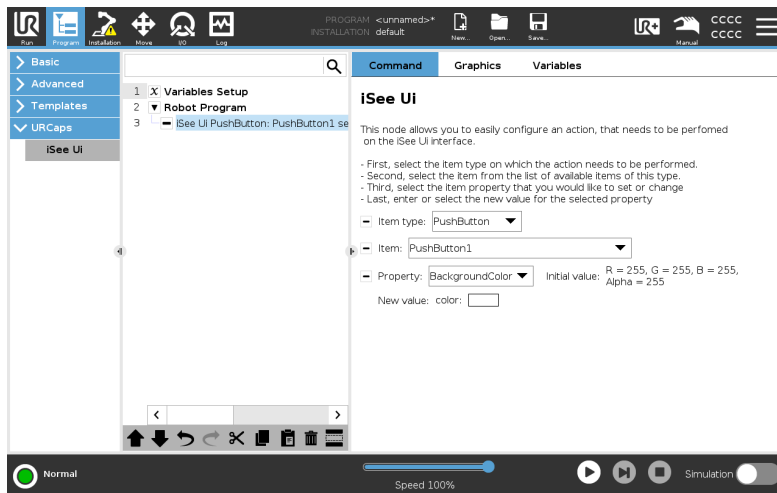


Figure 7.11: BackgroundColor property

- TextField
- PasswordField
- PushButton
- RadioButton
- CheckBox
- List
- Combobox
- Slider

7.3.7 Foreground color

The foreground color, or text color, of an item can be changed during runtime by changing the “ForegroundColor” property of an item.

The item types for which the “ForegroundColor” property can be used are:

- TextArea
- TextField
- PasswordField
- PushButton
- RadioButton
- CheckBox
- List

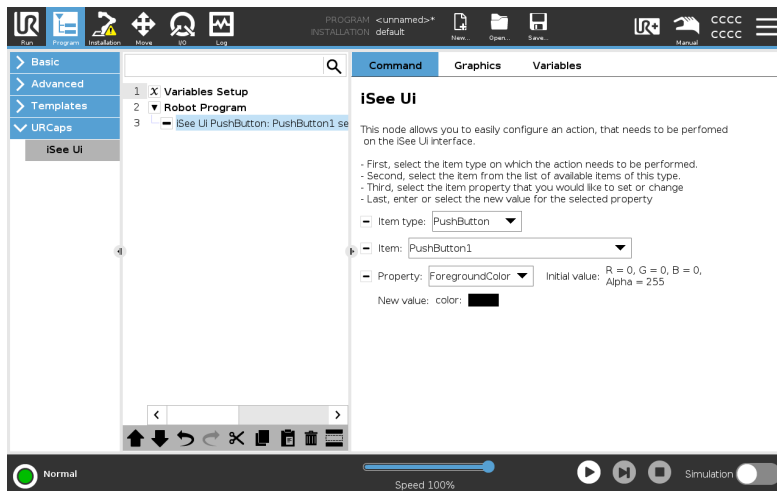


Figure 7.12: ForegroundColor property

- Combobox
- Slider

7.4 Parameterization of item specific properties

Some item types have additional properties. In this section, we will discuss these additional properties for each item type. The item types which have additional properties are:

- PasswordField
- List
- Combobox

7.4.1 PasswordField specific properties/actions

The properties and actions, additional to the ones discussed in section 7.3, for a “PasswordField” item that can be changed using the program node are:

- Clear field
- Show password

Clear field

With the “Clear field” action, the “PasswordField” content can be cleared. The content shows the value for a variable that is configured in an iSee Ui expression. The “Clear field” action thus allows clearing the content without also clearing the value of the variable it is representing. For this action, no value needs to be given.

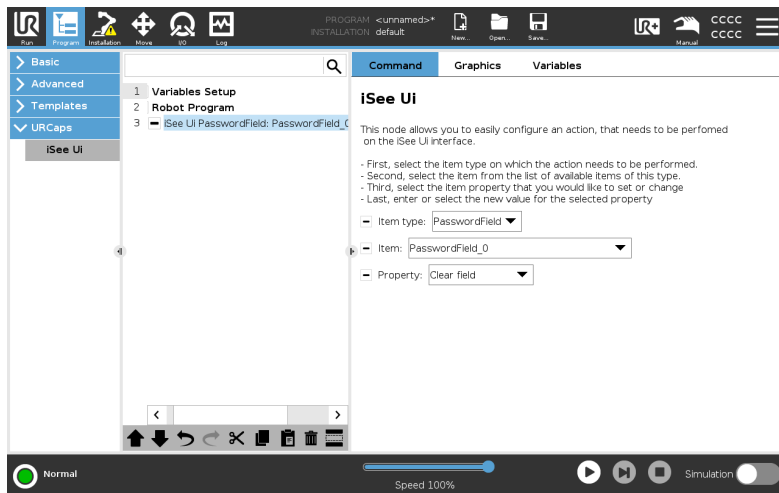


Figure 7.13: Clear field action

Show password

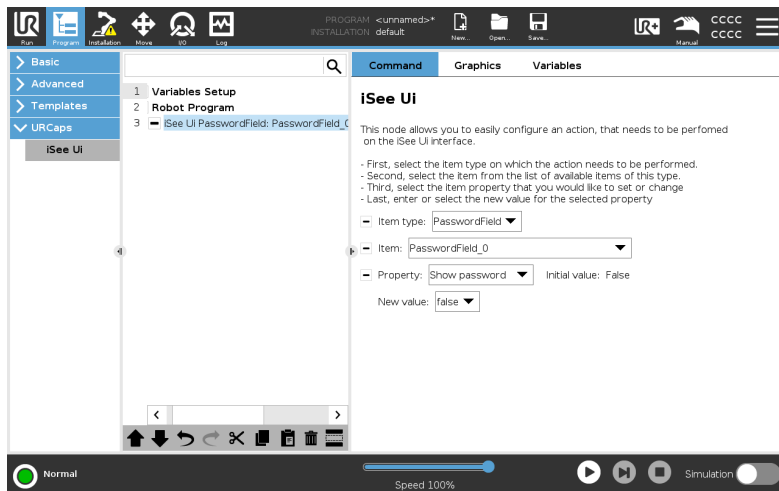


Figure 7.14: Show password action

The content of the “PasswordField” is by default unreadable. The content can be made readable by using the “Show password” action. When using this action, the content will remain readable until this action is set back to “false” by using another program node or by resetting the iSee Ui interface.

7.4.2 List specific properties/actions

The properties and actions, additional to the ones discussed in section 7.3, for a “List” item that can be changed using the program node are:

- Add¹
- Remove¹
- RemoveAll¹
- Clear selection

Each of these actions will respectively be discussed in more detail in the next paragraphs. Furthermore, the “List” also has an additional property whether to store its content or not, such that the content is not lost upon restart of the program or controller. This property, however, cannot be specified in either the installation or an iSee Ui program node. This property is specified in the iSee Ui Builder, in the “List” item properties panel, and can only be used if the “List” item has a content type of type “Fixed” (see also footnote ¹).

Add

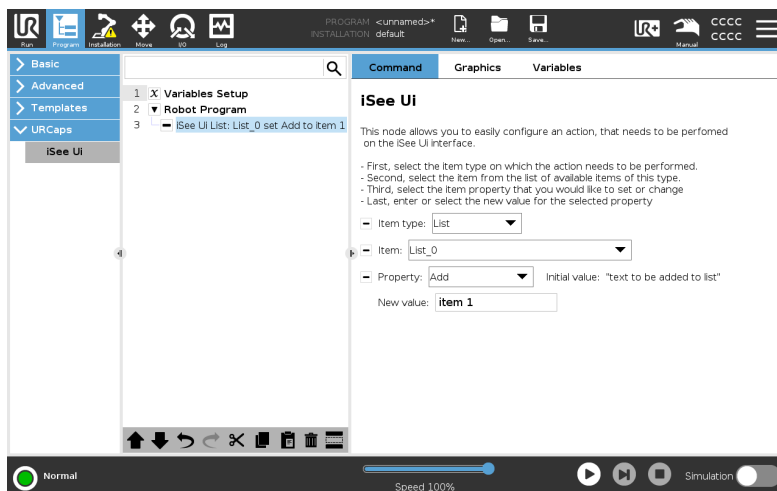


Figure 7.15: Add action

The “Add” action allows the programmer to specify what string value needs to be added to the content of the “List”. The programmer can add a text, a value of a variable, or

¹The “Add”, “Remove” and “RemoveAll” actions can only be used if the “List” is configured in the iSee Ui Builder to have a content of type “Fixed”. “List” items with a “Variable” content type, will not be able to use these actions.

a combination of both.

To convert the value of a variable to a string, the following function can be used:

`to_str(< variable_name >)`

When using this function, the variable cannot be selected from a list of variables, thus the variable name needs to be typed in correctly.

To concatenate a text and a value of a variable, no additional string function is required. The result can simply be typed as a normal text with the use of the “to_str” function for the value of the variable. For example, the value to be added could be:

“Row to_str(iCurrRow) and column to_str(iCurrColumn) failed!”

Remove

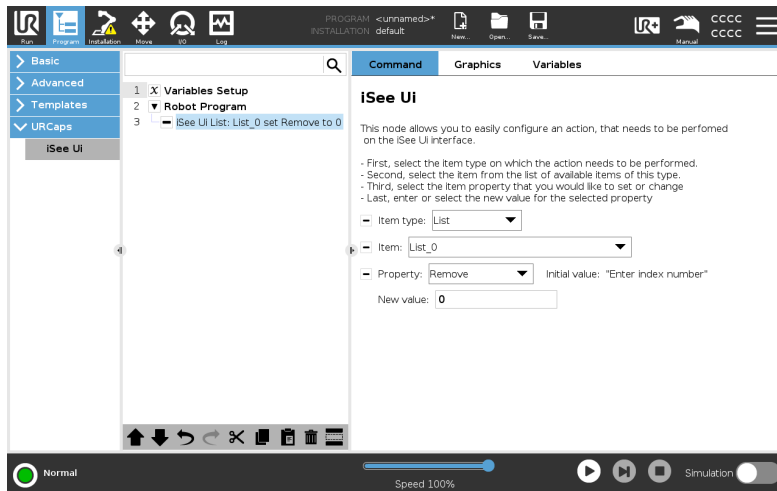


Figure 7.16: Remove action

The “Remove” action allows the programmer to specify which element in the list needs to be removed. The element to be removed is specified by its index in the list. The list is 0-indexed, meaning that the first element is at index 0, the last value is at index $(\text{length_of_list} - 1)$. If the entered value is greater than the last possible index, then the request to remove this element from the list will be ignored.

RemoveAll

The “RemoveAll” action allows the programmer to specify to remove all elements from the “List” when the function is performed. For this action, no value needs to be given.

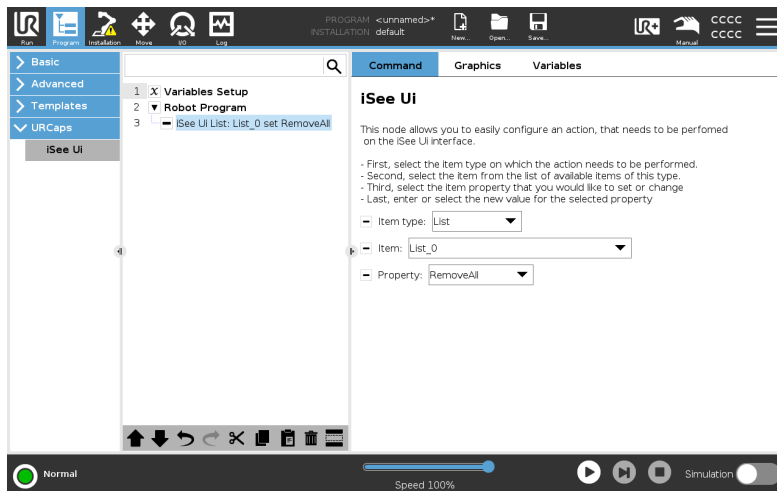


Figure 7.17: RemoveAll action

Clear selection

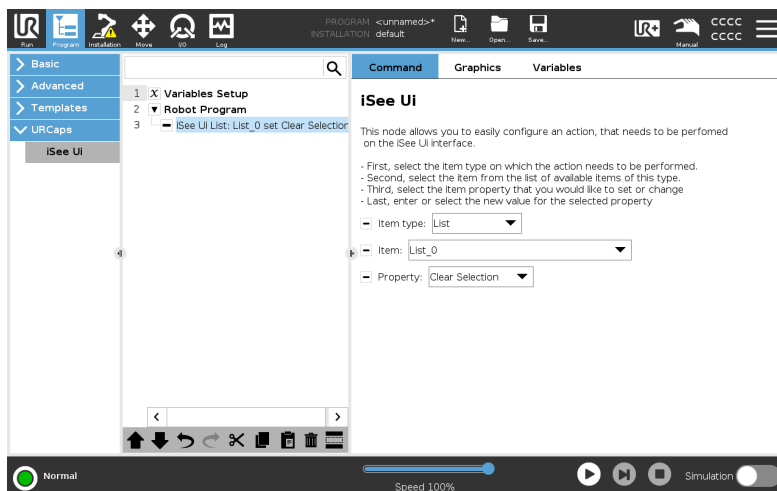


Figure 7.18: Clear Selection action

The “Clear Selection” action allows the programmer to clear the selection in the “List”. This action can be used to visualize to the operator that no item is currently selected. For this action, no value needs to be given.

7.4.3 Combobox specific properties/actions

The properties and actions, additional to the ones discussed in section 7.3, for a “Combobox” (or “Dropdown menu”) item that can be changed using the program node are:

- Add
- Remove
- RemoveAll

These actions behave and are used the same way as they do for a “List” item. For more detail about these actions, the paragraphs in section 7.4.2 can be consulted.

As with the “List”, the “Combobox” also has an additional property whether to store its content or not, such that the content is not lost upon restart of the program or controller. This property, however, cannot be specified in either the installation or an iSee Ui program node. This property is specified in the iSee Ui Builder, in the “Combobox” item properties panel, and can only be used if the “Combobox” item has a content type of type “Fixed” (see also footnote ¹ on page 25).

7.5 General behaviour of the program node

In this section, the general behaviour of the program node is discussed. Moreover, the behaviour of the node when:

- no iSee Ui interface is active
- another iSee Ui interface is activated
- the program and installation are loaded on another robot

In general, the iSee Ui program node is linked to the activated iSee Ui interface.

Behaviour when no iSee Ui interface is active

When no iSee Ui interface is activated, the program node will mention this and the node cannot be configured until an iSee Ui interface is activated. The node will also mention at the bottom for which interface it was configured. Figure 7.19 shows this behaviour.

Behaviour when loading a different iSee Ui interface

When an iSee Ui program node is configured for a certain iSee Ui interface, the node will reevaluate if the same configuration can also be applied to this newly activated interface. The configuration can be applied if, for the newly activated interface, there exists an item of the same type and name, and for which the same property can be changed. If the configuration cannot be applied, the configuration rows, for which there is a conflict, will be indicated by a yellow icon, as can be seen in figure 7.20.

The node will also mention at the bottom the interface file for which it was originally configured.

Behaviour when loading on another robot

Since the program node is linked to the iSee Ui interface that was active at the time the node was parameterized, the program with iSee Ui program nodes can be loaded on

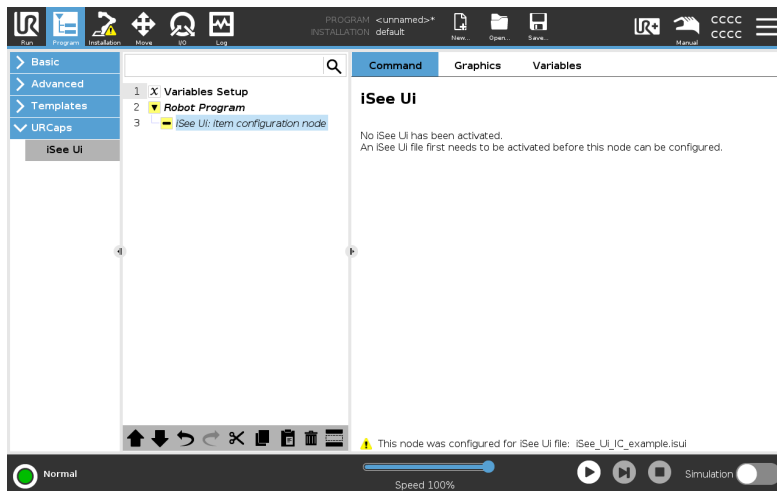


Figure 7.19: No active iSee Ui interface

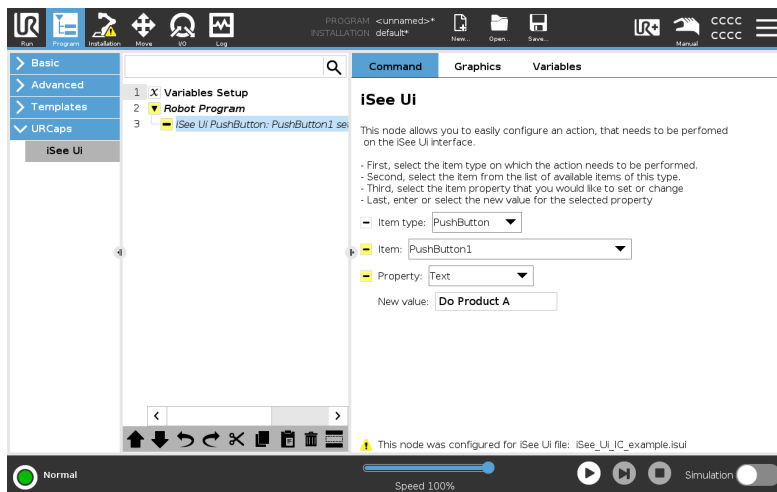


Figure 7.20: Reconfigure program node

a different robot of the same series. The requirement, however, is that the same iSee Ui interface is activated on this second robot. When another iSee Ui interface is active, the iSee Ui program node could be incomplete and additional action might be required before the program can run, as discussed in the previous paragraph.

8. Error handling

In the event an error occurs, a message as can be seen in figure 8.1 will appear. The user has the choice to either generate an error file, or do nothing. By sending the error file, the program and the “.isui” file to Industrial Cobotics, the error can be analysed to be solved in the next software update. These files can be sent to support@industrialcobotics.be.

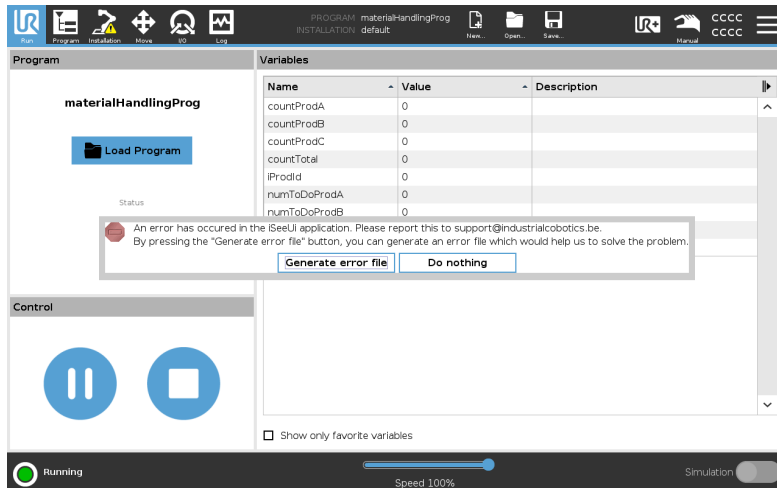


Figure 8.1: Error message

In some situations where an error occurred in the communication between the iSee Ui interface and the running program, the daemon server will automatically stop. In this case, the server will have to be started again in the installation tab or in the UR+ toolbar.

9. Caution with service

CAUTION:

When a service will be performed, and a new image file is placed on the UR controller, the service engineer needs to back up the programs folder before formatting the storage device!

The iSee Ui URCap uses a hidden folder stored in the programs folder. Inside the hidden folder, all imported interfaces are stored, as well as a backup of the license file. When the programs folder is restored after the service, the iSee Ui URCap will automatically recognize the license backup file and restore the license file. Furthermore, the imported interface files (.isui files) will be restored as well, avoiding the need to manually import these again.

The operator will thus be able to continue the work, after the service, without the need for additional actions.

10. iSee Ui examples

In this chapter, we will give an example of both a simple and a more advanced iSee Ui interface and their program.

10.1 Simple iSee Ui programming

In a simple iSee Ui program, only variable and/or IO communication between program and iSee Ui interface takes place. Thus, the iSee Ui interface is designed “as it is”, and the program doesn’t make changes to one of the item properties.

For this example, we will be using the iSee Ui interface shown in figure 10.1, and explain how an operator can set values and determine what the robot has to manufacture.

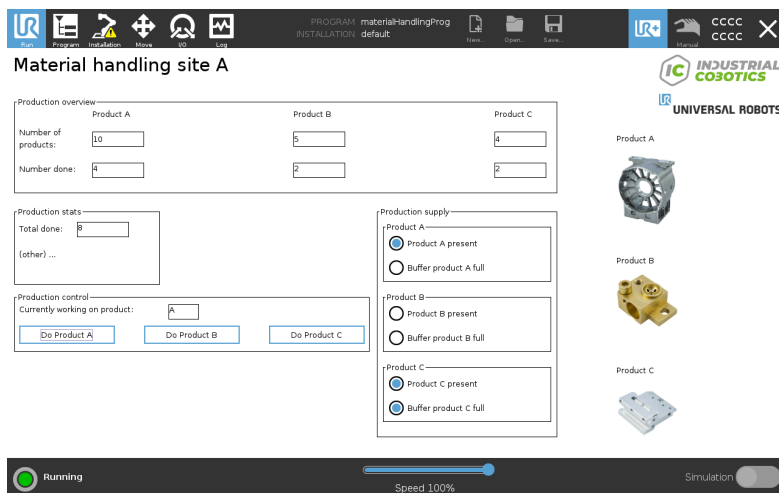


Figure 10.1: Simple iSee Ui example

In this example, the operator can enter for each of the 3 items, “Product A”, “Product B” and “Product C”, the number of items that need to be processed. This number can be entered in the “Number of products” fields. The “Number done” then informs the operator how many items have already been manufactured. In the panel “Production control”, the operator then also has 3 push buttons with which the current product to be manufactured can be set.

For this example, we have:

- 3 variables, 1 for each product, which will contain the amount of that product that has to be manufactured.
- 3 variables, 1 for each product, which contains the information of the amount of that product that already has been manufactured.

- 1 variable indicating which product type that currently needs to be manufactured. The 3 push buttons then each will write a different value to this variable.
- 1 variable to keep count of the total number of items that have been produced.

The programmer then has to program the UR program such that these variables, and the value of them, are handled correctly. Figure 10.2 shows the UR program for this example.

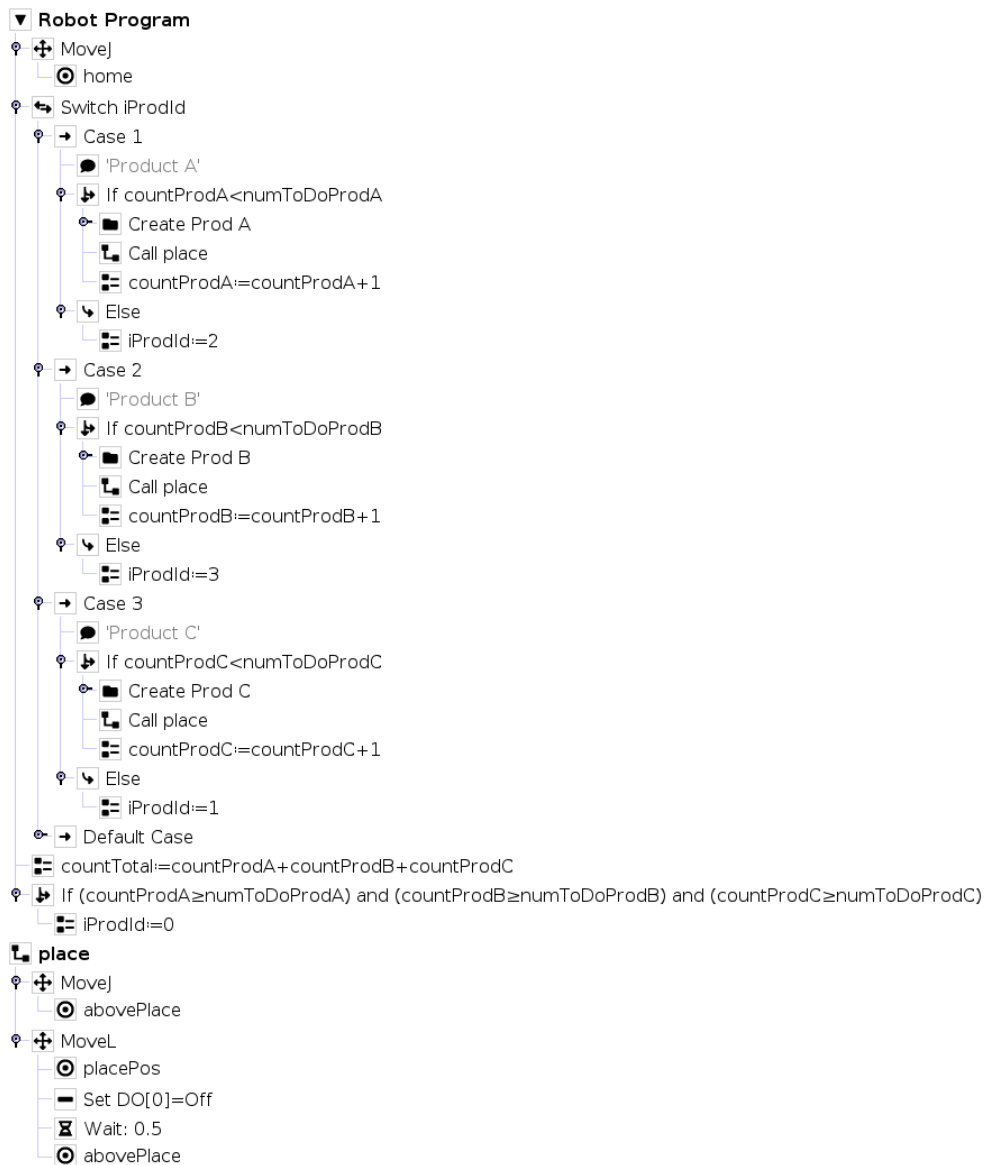


Figure 10.2: Simple iSee Ui example, UR program

By pressing the “Do product x” button, the variable “iProdId” is set to either 1, 2, or

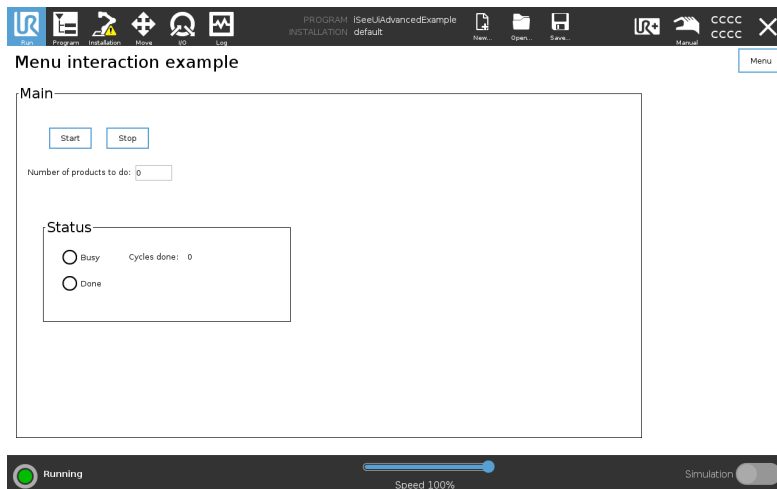
3 depending on whether the button for product A, B, or C was pressed. The variable “countProdA” holds the amount of product A that has been manufactured. The variable “numToDoProdA” holds the amount of product A that has to be manufactured. This variable thus receives its value from the iSee Ui interface, being the value that the operator entered. The same also holds for products B and C.

The iSee Ui configuration of what value is written to which variable, when a push button is pressed, to which variable is written when a value is entered in the fields, or reading out the value of which variable, is entirely performed in the building of the iSee Ui file in the iSee Ui Builder software.

The programmer thus needs to know exactly the name of the variables used in the UR program for the configuration of the iSee Ui file, or vice versa needs to know the name of the variables used in the configuration of the iSee Ui file when creating the UR program!

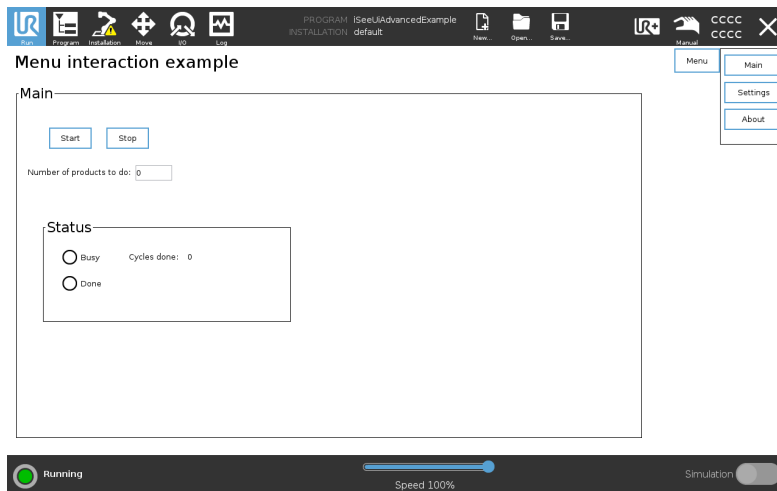
10.2 Advanced iSee Ui programming

In a more advanced iSee Ui program, the programmer programs property changes of items on the iSee Ui interface. Here, an example is discussed of how to create a menu in the iSee Ui interface. For this example, the iSee Ui interface shown in figure 10.3 will be used.



(a) Initial layout

Figure 10.3: Advanced iSee Ui example



(b) Open menu action

Figure 10.3: Advanced iSee Ui example

Here, the UR is programmed such that when pressing the “Menu” button, an additional panel will appear as shown in figure 10.3b, as well as the location for the menu button will be relocated. After pressing one of the menu panel items, the selected page needs to be loaded and the menu will also be closed automatically. For this, some variables will be needed as well:

- 2 boolean variables. 1 boolean variable is used for the “Menu” push button. When pressing the push button, the iSee Ui interface sets the variable to “True” such that it can be handled in an event. Once the event is triggered, the program sets the variable back to “False”. Thus, pressing the button will have a “Trigger” effect. The second boolean variable will be used to keep track if the menu is open or not. This variable is initialized to be “False” since the menu will be closed when activating and opening the iSee Ui interface.
- 2 integer variables. 1 integer variable is used to indicate which of the pages has to be loaded. When pressing the buttons “Main”, “Settings”, and “About”, the integer variable will get respectively the values 0, 1, or 2. This value will then be evaluated in a switch-case structure, where the appropriate program nodes are called to make the page panels visible or invisible. The second integer variable is used to evaluate if the first variable value has been changed by pressing one of the 3 push buttons. When the value between the 2 variables is different, an event is triggered to perform the switch-case and where the second variable will immediately receive the value of the first variable, hence we get a “Trigger” effect.

Remark: A “iSee Ui BeforeStart” or “iSee Ui reset” sequence will have to be added in which the initial property values are set or where the iSee Ui “Reset” action is called.

Furthermore, the above-mentioned variables will be given an appropriate value that represents the iSee Ui after resetting. Thus, settings the boolean variables false, and the current page to a value of 0 that represents the “Main” page.

An example of a UR program for this interface can be seen in figure 10.4.

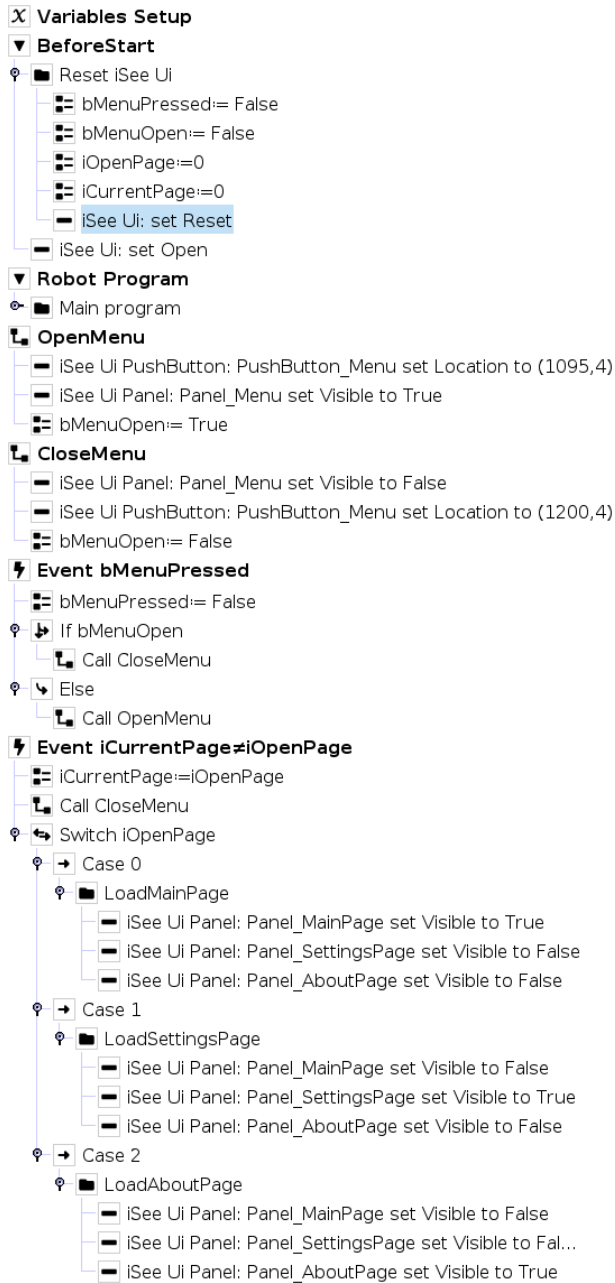


Figure 10.4: Advanced iSee Ui example, UR program