WITHYOU
WITHME

RPA Blue Prism Developer Course

Written Guide

Contents

Intent

This document is designed to augment your online learning experience and provide you with the opportunity to review and revise the content that has been discussed during the individual lessons of the course

Feedback

Please contact your WithYouWithMe RPA instructor if you have feedback on this document or any of your WYWM courseware.

Complex Objects

Overview

Hi and welcome to the lesson where we will be building more complex objects using another application called Training Order System.

In this lesson we'll build on our discussions about the Notepad object and apply it to a training order system application so that we can set ourselves up to produce some useful work in the first assignment. We won't be stepping through the whole assignment in this lesson, but we'll be setting up the application modeller in preparation for the objects we're going to build during that assignment.

As usual we'll switch between the slides and Blue Prism. To get the most out of the video, make sure you follow along in Blue Prism and pause the lesson if you need to.

Training Order System

First, we'll first familiarise ourselves with the training order system application. If you haven't already you can download this to your desktop from the materials tab of this lesson.

The Training order system application is a simple application that will simulate an order system where you can login, place and order and receive an order confirmation code once the order has been placed.

Now, let's jump back into Blue Prism and we'll create an object called Order System. In this object we're going to use spy mode again to identify the fields in our Training Order System application.

So as we did before, we need to set up the elements of our Training Order System application and we do this in the Application Modeller.

So let's go in there and step through the wizard. First we'll leave it called Order System, it's a windows application, it's an executable file and we need to tell Blue Prism where our Training Order System Application is located. We can now accept the defaults in the remaining pages of the wizard.

Great, now we're in the main page of our application modeller, let's launch our application, select element 1 and call it Log In Window. Now, do you remember what we need to do next?

We need to tell Blue Prism what we're looking for when we say Log In Window. Do you remember how we did this?

We click identify, enter spy mode and now that we're in spy mode we can see the different fields within the application are highlighted. At the moment we're looking for the whole login window, so we hold down control or command and left click.

Now we can see that Blue Prism has identified a bunch of default attributes that make up the Login window's fingerprint. Now we click have a quick look at the matched elements and highlight to double check Blue Prism can see our element. We're happy it can see it so we can move on to the next step.

Now, in our element tree on the left of the application modeller window, we can add elements, and we can add subordinate child elements. While all the elements in this tree are considered elements, some are child elements to a high level element. For example, if we were spying a Microsoft Excel Window, the entire window would be the Element, and all the fields on that element would be considered child elements.

So we're going to spy the staff number and password text fields, as well as the sign in button. So, to add an element or child to our element tree we go down to the button left of the application modeller window and we can select either an element or a child. In relation to the Login In Window element, would our staff number text field be an element or child?

Have a think back to our hierarchy of processes and see if you can get the answer.

It'll be a child because the Log In Window is the Element in this case and everything on this Log In Window will be considered a child.

So with that said, let's add a child to the Log In Window Element, we'll call it Staff Number Text Field and we'll identify it in spy mode.

Great, we can see that Blue Prism has identified some attributes. We now have a quick look at the matched elements and highlight to double check Blue Prism can see our element. So what happened this time?

We'll Blue Prism found some attributes that were relevant to our staff number text field, those elements were also relevant to the password text field directly below so this field was highlighted as well. Blue Prism them developed an error message to warn us that there was more than one highlighted field and that it couldn't tell the difference between the two.

I'll click highlight and run this again in case you missed it.

So what are we going to do to fix this issue?

Matched Attributes

You guessed it, we're going to need to select and deselect some attributes so that the staff number text field and the password text field are different.

There’s a number of ways to do this. We could just blindly try some options and see how we go but it’s best practice to review the names of the attributes and the value they’ve been given and whether it’s been used in the matching process.

But before we do this, it will be helpful to also spy the password text field so we can compare the two. Lets add another child to the Log In window by highlighting it, clicking child, giving it a name and identifying the desired field.

Now we can compare the attributes associated with both text fields so let’s have a look. There’s a few options but we want to make sure that we pick something that’s not going to change, like the height attribute in our notepad element. Ordinal stands out as a good option given the term ordinal refers to an order or series of numbers. Blue Prism assigns an order number to every element that it detects, in the order in which it detects it. The Ordinal attribute displays this order reference.

Here we can see that the staff number text field has an ordinal value of 3 and our password text field has an ordinal value of 2. So let’s tick the ordinal box on both password and staff number and see how we went by clicking highlight.

Perfect, that worked really well. Now let’s add one final child to the Log In Window that will be our Sign In button. So we highlight, add a child, give it a name and identify it. Great, we’re happy with the default matches so we highlight again. We don't have any duplicates this time which is good, so we’re happy. We can now click apply and ok and this will be ready and waiting for us when it comes time to do our assignment.

Wait Stages

Overview

Hi and welcome to the lesson where we will be discussing wait stages.

We've very briefly discussed wait stages when we built an object on one of our earlier lessons but we'll go into wait stages in more detail during this lesson and we'll discuss their use in a business object.

As usual we'll switch between the slides and Blue Prism. To get the most out of the video, make sure you follow along in Blue Prism and pause the lesson if you need to.

Blue Prism's Speed and Wait Stages

It may seem like our applications take the same amount of time to start up every time, however, in real life situations where we have no control over the response time of an application it is not beyond the realms of possibility that an application like microsoft excel may take 3-4 seconds to open.

This isn't an issue for a human because we would simply wait for the application to open, however as we've mentioned before, Blue Prism can run incredibly fast and it is therefore possible for Blue Prism to to try and write something to microsoft excel before it's even been able to launch. This will of course cause an error and stop our process in its tracks.

For this reason we incorporate wait stages into our process. The wait stage is used to pause a business object and wait for the application element that it's looking for to open or what we call exist. This also allows a business object to deal with applications that may have erratic performance.

The wait stage has several element conditions, however in the majority of cases we'll be waiting for a specific element to exist and to achieve this we select the check exists condition from the drop down menu.

Now, let's jump back in to Blue Prism and revisit the wait stage in the launch tab of our notepad process.

First, let's familiarise ourselves with this process. We have a launch page within our object that launches our application, notepad in this case. We have a Write page which writes the text from a data item in the text window of notepad, and we have a close page that closes our program.

You'll recall that we added a wait stage to our write page when we built this object, and I said that we'd revisit this later. Well, I didn't lie to you, we're revisiting the wait stage now.

The reason we added a wait stage on this page was to slow Blue prism down before it started writing. Without this wait stage there is a pretty good chance that Blue Prism would have tried to write something in the text field of notepad before it had even been able to launch, and as we've said, this would have produced an error.

Let's demonstrate this now by deleting the wait stage, connecting our object back up, saving and heading over to the process studio.

Let's open our notepad process and press go.

You can see that an error is generated when we get to the write stage. If we look at this error we can see that the write stage on notepad was unable to match any windows to the query item. In english, this means that Blue Prism couldn't find the element it was looking for.

Now, to the naked eye we saw that notepad opened, but blue prism was moving so fast that at the time it tried to carry out it's write action, notepad hadn't yet launched. This is exactly why we need wait stages.

So now that you know I'm not making this stuff up, let's go back to the write page of our notepad business object and add the wait stage again.

Let's delete the link and drag over a wait stage. As we saw with the loop stage, we can see that when we drag the wait stage across we can see the wait icon, and when we drop it the timeout icon appears. This will default to a vertical orientation but the wait and timeout stages are typically orientated horizontally in industry so we'll set that up now.

Now let's click on the wait stage to open its properties. We'll call it Wait For Text Field.

The only thing we need to do for this wait stage to be of use is to select the value we want our wait stage to wait, let's say three seconds, we can click ok and link it up. Noting that in this case we're linking our timeout stage to our write stage.

So if we launch our notepad and run our write page, you can see the wait stage will simply wait 4 seconds, timeout and progress to our write stage where Blue Prims wrote on our notepad.

Intelligent Wait Stages

That worked well and we didn't get an error this time. But imagine if we needed to call on this action 1000 times. That would mean we'd be held up at this wait stage for 3000 seconds or 50 minutes. Now considering Blue Prism could probably run this object 1000 times in less than a minute, 50 minutes is a lot of unnecessary time to be waiting around.

If only there was a way to tell Blue Prism to look for a specific thing, and once that thing appeared, it could move on without having to wait the remaining 3 seconds.

We'll once again we're in luck. Let's go back into the wait stage property and check out the actions area in the centre of the window. Here we tell the wait stage what to look for, using the element filed, and what condition needs to be met before the wait stage can move on.

So, let’s tell the wait stage what element it should be looking for by selecting text field from the element tree and dragging it into the element field. And as we said the condition for this action is to check if the text field exists, so let’s select that as well.

Notepad is pretty fast when opening but 5 seconds is a good amount of time to wait before an error is generated, noting that if the element appears earlier, blue prism won’t need to wait the full 5 seconds before progressing. Ok great, now click ok.

We can see that now we’ve told Blue Prism what to wait for, we have a node on our wait stage that we can link up instead of our timeout stage. You can also see that our link is given a name so we can see what action our wait stage is carrying out. This means that when the condition ‘check exists’ is met, the process will follow the path of the node. If the condition is not met within 5 seconds the process will proceed to the timeout stage.

Now, if we go back into our wait stage and add another wait parameter, drag in text field again but this time check window and click ok what happens?

Did you notice that we got a second node in our wait stage. We can link this up as well and it’s also given a representative name based on what action it’s carrying out.

Now before we move on, it’s important to note that the wait stage will wait for each element in order depending on how we’ve set it up. In this case if notepad doesn’t launch our wait stage will wait for our text field of notepad to exist for 5 seconds, then it will wait another three seconds to check whether an attribute within or text field exists, and if none of these things occur it will the progress to the time out stage.

In the event this happens we would need to incorporate some exception handling but there’s a whole lesson dedicated to this so we’ll leave exception handling there for now.

So let’s remove the second node of our wait stage, square it all up and press reset.

Finally, note that for now, because we haven’t connected anything to our timeout stage an error will be produced. This is ok as long as notepad launches within 5 seconds, which based on our present situation, we know it will. So for now we can ignore this error, save our object and give it a test using our notepad process. So let’s do that now.

Great, it’s working again and now we have a much better understanding of wait stages.

Navigate and Write Stages

Overview

Hi and welcome to the lesson where we will be discussing navigate and write stages.

We've very briefly discussed Navigate and Write stages when we built an object on one of our earlier lessons but we'll go into Navigate and Write stages in more detail during this lesson and we'll discuss their use within a business object. We'll then apply this knowledge to Blue Prism by using a bot to log on to our training order system application.

As usual we'll switch between the slides and Blue Prism. To get the most out of the video, make sure you follow along in Blue Prism and pause the lesson if you need to.

Navigate Stages

First let's discuss Navigate stages. As we've touched on briefly, Navigate stages are used within an object to help Blue Prism 'navigate' a particular application. These navigate stages send Blue Prism to a particular place within an application and then perform a particular action, the actions that can be carried out by the navigate stage depends on the element you connect it to. For example, we saw this when we used a navigate stage to launch and terminate our notepad application.

We can also use navigate stages to "click" on elements within an application window by identifying the element from the element tree and selecting the action that we want Blue Prism to execute, this this case click or "press".

We'll discuss this in more detail when we run through the Blue Prism Exercise shortly.

Next we have Write stages. Write stages as you've probably guessed allow us to write text to an application as we saw in our notepad process.

Similar to a calculation stage, in the write stage we select a value that we want to 'write' and we select the location we want to write this value to by selecting an element that's been spied in the application modeller.

Let's jump back into Blue Prism Object Studio and walk through navigate and write stages in the context of our Order System object.

In the complex objects lesson of this course we started creating an application model for our order system application in the application modeller. We spied the Sign In window, the Staff Number Text Field, the Password text field and the Sign In Button.

Now we're going to add the staff number and passwords using a write stage and once complete we're going to click the sign in button using a navigate stage.

Firstly though, we'll need to launch our application, so let's change the name of our first action tab to Launch, we've grab a navigate stage, make it a little bigger, call it launch. Do you remember how to tell Blue Prism to Launch our Order system?

Well, first we need to identify the element we want our object to do something with, in this case Order system, and we need to select our action. In this case launch. And that's it, now if we click ok, do you remember what else we need to do to our launch action page before we can move on?

We need to add a wait stage to check that this action has been able to do what we've asked for. So we grab a wait stage, we'll reorientate it and call it Sign In Window. We want to check that the Sign in window exists to let's drag over our Sign In window element and select the action check exists.

Now we link it up, press reset and click go,,, we can see that our order system is launched.

Great, next we want to write our staff number and password to the associated text fields within our application. So let's create a new action tab and call it Enter details.

Now what do we need to do next? We could drag across a write stage but what are we going to write?

First we're going to need some data items that will allow us to input some values that we can call on our write stage.

Let's get two data items, we'll make the first data item a bit bigger and call it Staff Number, it will be a text data type and our Staff Number for our order system application is bp so let's put that in our initial values and click ok.

Now we'll make our second data item a little bigger, we'll call it password and it will be a password data item. So we select that and our password is just password so we enter that in the initial values section and click ok.

Write Stages

Now that we've got some values that we can write, we get our write stage, drag it across and make it a bit bigger. We'll call it enter details and we'll set it up.

We want to enter our Staff Number first so how do we do this? If you guessed that we need to do some dragging of values and elements into the actions window you're spot on.

Let's first drag our Staff Number text data item into the value field of the action window, which is the value we're writing, and let's drag Staff Number Text Field element in to the element field. Is the section of the application that we're writing our value to.

Ok, now let's add a row and we'll do the same again, but this time we'll use the password value and element. Once we've done that we can click ok and connect it up. Now if we reset again, we can see that our Staff Number and Password are written on our application.

Finally, now we've entered our details we want to click the Sign In button. Do you remember which stage we use to click a button?

Yep, we use the navigate stage. So let's add another action tab, call it Click Sign In and drag across a navigate stage.

We'll make it a bit bigger and call it Click Sign In. Any guesses on how we set this up?

Well first we select the element we want to click, and drag it into the element field. Now, we need to select the task or action that we'd like our object to do within our application, in this case press because we want the object to press the button.

Great, we click ok, link it up, reset and press go. Now we can see that we've signed in and we're at the next page of our Order System Application and we've done this with Navigate and write stages.

Is there anything else we need to do before we can move on? Yes, we need to check that we've moved to the next page of the order system. Do you remember how we do this?

Again, we use a wait stage, but before we do that, we'll need to identify this new Options window, which you'll recall, we do in the application modeller.

So let's go into the application modeller, we'll add a child to the Order system by highlighting it and clicking child and we'll call it options window.

Now we identify, control left click the whole Options windows and review the attributes that Blue Prism has identified. These look pretty good so we click highlight to make sure we're happy and we click ok.

The last step is to set up our wait stage so that we can confirm that our object action has been completed as planned and we are at the page that we expect to be on. So let's delete the last link, add a wait stage and reorient. We'll call it Options window and we'll check that the options window exists the same as we did last time.

Now we click ok and and link it up again.

To test this object is working as we want it to let's first close the order system application and go back to the launch tab. We click reset, while noting the error associated with the timeout within our wait stage, and press go. Here we can see the application has been launched and we've checked it exists. Next we go to the Enter Detail tab, click reset and run and we can see that our details are entered.

Finally we go to the click to sign-in tab, reset again and click ok. Here we can see that we've clicked the sign in button and our options window appears and we've also checked that this exists through our wait stage.

Great, that all worked really well.

Attaching and Detaching an Object

Overview

Hi and welcome to the lesson where we will be go through attaching and detaching an object to an application.

In this lesson we'll introduce the concept of attaching and detaching from an application and build on our knowledge of business objects and the application modeller. We'll also incorporate attach and detach actions within our Notepad process.

As usual we'll switch between the slides and Blue Prism. To get the most out of the video, make sure you follow along in Blue Prism and pause the lesson if you need to.

Attaching Blue Prism to an Application

In some scenarios an application may already be running and we therefore won't need to launch it before we can start interacting with it. In the event an application is already running, we want to make use of this. Also, In the real world, you may not have the luxury of launching an application each time you run a process. In many situations, the process may require the BP bot to connect (or attach) to an application that is already open.

We also want to avoid launching an application and having multiple instances of an application running unnecessarily. The use of attach functionality allows our object to "attach" to a running application and interact with it as we've done following the launch action of our previous processes.

This means that when we are launching applications we'll now look to incorporate some logic that will assess whether our application is running, and dependant on the result of this assessment, attach or launch as required.

So, let's jump across to Blue Prism, open our notepad object and add an action tab called Attach.

Up until now we've used the launch tab to launch our notepad application under the assumption that Notepad is not already running. What we want to do during this exercise is create some logic that will assess whether our Notepad application is running and if it is, we want to attach our object to it.

Now let's have a think about how we could do this?

I'll give you a hint, it involves a read, decision and navigate stage...

Well, first we need to assess whether our application is running and we do this through the use of an read stage. So let's drag a read stage across, make it a bit bigger and call it Notepad Connected?

We can now drag the notepad element across to the element field as we did in our action stage. We now check the data field and instead of check exists that we saw in the wait stage we need to select “is connected” which is Blue Prism language for is Notepad attached?

Now, have a look at the next two fields. We have Data Type and flad selected and a store in field. This means that the result of our question, “is notepad attached” will be either a true or false answer, which makes sense, but we need somewhere to store this answer, so let’s click ok and get a data item.

We’ll make it a bit bigger and call it Is Notepad Attached, as we just saw in the read properties, this will be a flag data type and because we’re storing a value that’s being calculated for us we dont need to worry about and initial value so let’s click ok.

We now go back into our read stage and you can see that we’re now able to drag our newly created data item into the store result in section of our action window. Now we’re happy with this let’s click ok again.

Now the next thing to do is set our logic up so that our object can make a decision as to whether our notepad application is attached or not based on the result of our read stage. Do you remember which stage we use to make a True or False decision based on an expression or data item?

Yep, we use a decision stage, so let’s drag that across, make it a little bigger and set it up. We’ll call it Is Notepad attached. Now in our expression window we can simply drag in our data item. We can do this because the only thing we’ll ever store in our data item is a true or false because it is a flag data type, and all our decision stage needs is true or false data on which to base a yes or no decision.

So we click ok and link up our logic so far. Now, if our notepad application is attached we’re happy and we can continue on and interact with our application, so this path can proceed to end. However if it’s not attached we need to attach to the notepad application. Do you remember how we do this?

Well the same as in the launch page, we use a navigate stage. Let’s drag one of these across, place it underneath our decision stage and make it a bit bigger. Now let’s open the navigate stage’s properties and call it attach.

Now, we’re looking to attach the notepad application so we drag the notepad element into the element field of the action window and we select an action. Here, instead of selecting launch as we did before we select attach.

As you just saw, the inputs area was activated as soon as we selected attach and in this inputs window we now need to give our object a bit more information to make sure it attaches to the correct application.

A great option for more information that we can give Blue Prism in this case is specifying the window title of our application. Now I recall that the name of our notepad was “Untitled - Notepad” so let’s type that in making sure that we use inverted commas denoting that it’s a text data type.

Now let’s click ok and link it up noting the orientation of our yes and no links are still correct. Now we reset for good measure.

Now let’s put our attach action to use but first we need to launch notepad.

So far we’ve used the launch action to launch our application but I’m getting sick of selecting this tab, clicking reset and clicking go every time. It would be great if there was a faster way to do this.

Well guess what, there is a better way. Have you noticed this launch button since you’ve been spending time in the object studio?

Now instead of using our launch tab or going into the application modeller we can simply click this launch button and our target application for this object will launch.

You’ll also notice that as soon as we click launch the launch button will switch and give us a detach option. This is because when we launch our application, Blue Prism will automatically attach and the option of launching or attaching is no longer available.

Ok, so looking at our attach action, what do you think will happen if we run this action now?

I think our read stage will detect that our notepad application is connected or attached, it will send a true flag to our data item, this value will then be used by our decision stage to navigate toward the true path and our object will proceed to end.

Let’s see how close we got.

Great, got it in one!

We didn’t really use the attach functionally during that exercise though, so let’s reset and click our detach button. This is removing the link between this object and our application and as you can see the button has changed again from detach to launch, indicating to us that our application is no longer attached.

To demonstrate that our object is no longer attached to our application, we can navigate to our write tab, reset and press go, you can see we get an error. This error is generated because our object can no longer see our notepad text field, because it’s no longer attached.

We’ll accept this error and head back to our attach action to reattach our application to our object using the object we just created.

So we click reset and press go. We notice this time that the output of our read stage is false so our decision stage goes down the no path and our navigate stage attached to our notepad. You may have also seen that when the notepad application was attached by our navigate stage, the launch button also switched back to the detach, indicating that our application is attached.

If you're ever in doubt as to whether an application is attached to an object or not, this button will often provide you with some preliminary guidance.

Now let's give our attach action tab a bit more of a test by having a few notepad instances open and having it select one of these instances to attach to.

So let's start by detaching our object from notepad. Now let's call this notepad Instance Note 1 but saving as and typing Note 1. We'll do the same again but we'll call our second instance Note 2.

Now as you can see, we've got two notepads open but we would like our attach action to attach to our second notepad Note 2.

How do we do this?

Well, it'll be in the properties window of our attach navigate stage but what do we need to change to get our object to attach to our Note 2?

We need to update the value field of our window title. We initially had this as untitled but if we update this to Note 2 and click ok we'll see what happens. Well that looked like it worked but to be sure let's go over to our write action, and see which notepad Blue Prism chooses to write on . This will indicate which notepad it was attached to.

So we reset and press go. And look at that, our note was written to our Note 2 notepad.