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.

Building Basic Automations

Overview

Hi and welcome to the lesson where we will be building basic automations. This where you'll build your first bot using RPA software so if you're not already excited now would be a great time to turn the excitement levels up.

In this lesson we'll build on our knowledge of the process studio by creating new processes, using calculation stages and data items. At the end of this lesson we'll put together everything we've learnt and create a working process from scratch and you can tell your friends that you made a robot.

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

The Process Studio

First let's quickly review the process studio. A Blue Prism process is essentially code represented in a diagram that looks similar to a business flow diagram, and our stages within this process are connected using links to form a logical flow through our process.

Processes in Blue Prism have a special meaning and as we've mentioned, they're independent from objects and it's important to know the difference between a process and an object. A process is the holistic representation of the entire process that we're automating, whereas an object is called on to carry out specific functions within a process that involve the interaction with applications. For example, a process may open, add data to and close a microsoft excel workbook. Our process would include all these steps on a page but it would call on objects to interface with Microsoft Excel and carry out the individual opening, inputting data and closing.

For this reason we also have an object studio where we can build objects in a similar way to how we build processes. We'll reiterate this later in the course when we go through objects in more detail.

Creating a Process

Now let's cover some basic skills that you'll need to create a basic process using calculation stages and data items.

So let's jump into Blue Prism and create a new process called Basic Process.

After you've created your Basic Process you'll be back in the Process Studio. You'll recognise the Process Page from previous lessons with the stages on the left hand side and the buttons across the top.

Calculation Stage

Click and drag a calculation stage onto the main page. You can also drag a data item onto the page to the right of your calculation stage.

You will notice that one of the stages in the stages toolbar is highlighted with a blue rectangle. Whenever a stage is highlighted, that stage can be added to the main page whenever you click your mouse. This allows you to add multiple items to the main page without clicking and dragging each time. You can see on the screen I'm using this feature to add several data items.

You'll also notice that one of the stages on the main page will be highlighted with dark red squares around it. This means we can also use keyboard shortcuts to cut, copy, paste and delete the highlighted stage as required. For example, copy using control c, paste using control P and delete using the delete key.

You can also highlight areas of the main page and this will highlight every stage within your highlighted area. You can move all the highlighted stages together which can make life a lot easier and the same keyboard shortcuts apply with more than one stage being highlighted.

You can also link the stages of our process using the link stage. You link stages as shown by clicking and dragging the link stage from the center of one stage to the center of the next stage. You'll notice that if I highlight another stage and try to move it, it wont work because I still have the link icon highlighted. I need to go back to my pointer to get this functionality back. Also, before we head back to the slides note that as I move my stages around the links remain connected.

Ok, as we discussed briefly in the previous lesson, a calculation stage uses expressions in its properties window to calculate a value, and then retains or stores that value in a location of your choosing.

The "Store Result In" field is where we specify where we would like the result to be stored. These results can be stored in a data item stage.

Data Item

A data item is a parallelogram shaped stage that acts as placeholders for values such as numbers, text, dates, passwords or any of the other 10 data types. In most situations, your inputs and outputs from a calculation stage will come from and be sent to data items. Data items should be given meaningful names such as Account Number, Staff Number, or Password etc. A programming comparison would be to liken a data item to a variable which can be assigned a value and used throughout the remainder of the process.

Data items won't be connected to the process flow via links, rather, they are connected using the "Store Result In" field within the properties window.

All we want this process to do is add two numbers together and provide us with the answer. In addition to the start and end stages what other stages will we need?

Well, we'll need a calculation stage to do our calculation but how many data items will we need?

Well, we'll need at least two data items for the numbers we're going to add but we'll also need a third data item to put our answer in so let's drag them in now.

Now that's done we'll start by giving Blue Prism our numbers. We do this by putting some data in our data stages. So double click our first data stage, give it the name First Number, select the number data type and enter the initial value of 7. Now click ok.

Repeat this process for our second data item but let's call it Second Number, select the number data type again and enter the initial value of 4. Click ok.

And in our last data item we'll just give it the title Answer with number as the data type but we'll leave the initial value blank. We don't need an initial value here because we're going to get Blue Prism to provide us with a value in this data item (which will be our answer).

Now let's go into our calculation stage's properties by double clicking the calculation stage. First let's give it the name add numbers. In the data items section of the window, we drag our first number data item into the expression window, press the plus sign and then drag our second number data item across. You'll notice that these data items are under the number heading. Note that Blue Prism knows the data in these data items are numbers because we told it earlier by selecting the number data type in our data item.

You'll also recall that we don't necessarily need to drag these data items in. We could type the correct syntax in manually, but this practice is much more prone to errors through misspelling and capitalisation which is why dragging and dropping is considered best practice.

We then go back to the data items section of the window and drag the answer data item into the Store Result In box. We then click validate to make sure our expression is valid. Great. It's important to note that we always validate our expressions before we exit this window. This reduces the chance of errors which will make fault finding a lot easier when our processes become more complex. Let's quickly demonstrate this by intentionally getting the capitalisation of answer wrong in the store result in field. As you can see, when we do this and press validate we get an error. This highlights how easy it is to make errors when we type things in manually and why we always prefer to drag and drop where possible, which we'll do again now and click ok.

Ok, our process is looking pretty good but before we run a process by clicking go we always click the refresh button and review the Validation button for errors. As you can see, there are no errors with our process which is a good sign, however if we remove a link and refresh again, you can see that an error in

our process has been identified. If we click on the validation button a little more information about the error is provided. In this case we can see an error message has been produced and in the description it tells us there is a missing link.

The validation button produces these “error reports” if you like, by running a scan of our processes configuration and displays a list of basic errors and warnings. While this functionality can be very useful when checking for basic errors, it shouldn’t be relied upon too heavily because it can be somewhat ambiguous at times. The most useful information however, is in column 1 and 2 which gives us the page that the error is located, and the individual stage within that page that’s causing the error.

So let’s return the link, refresh the process and note that the errors return to zero. Now lets press the go button and put our process to work.

You can see our process has got the answer correct. You can see in our first two data items that the initial value is unchanged and because the process has cycled through the current value is equal to the initial value. In our Answer data type however, our initial value was blank but as you can see the current value is 11 which was inputted by our calculation stage.

Decisions

Overview

Hi and welcome to the lesson where we will be discussing decisions within our process.

In this lesson we will be building on our basic process and understanding a little more about decision stages.

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

Decision Stages

At the start of this course and in more detail in the RPA Analyst course, we learnt that RPA is very good at doing clearly defined and repetitive tasks with large datasets. In Blue Prism, the functionality that allows our robots to do these repetitive tasks are decision stages, circular paths, collections and loops. Noting their significance, let's look at decision stages in a little more detail now.

A process will rarely follow a straight path and we are likely to want to give our process the option of making Yes or No decisions and based on these decisions travel one way or another. Because of this, the decision stage has one inbound path and two outbound paths and if the result of a decision stage is true it takes the yes direction and if the result is false it takes the no path. In coding this is referred to as boolean logic.

The decision stage is the simplest method for creating multiple paths within a process, and the decision stage works by evaluating the result of an expression as either true or false. The expression area within the decision stage's properties is similar to what we saw in the calculation stage and functions in much the same way.

Let's now jump across to Blue Prism and use decision stages in our basic process.

Using Decisions in a Process

If you haven't already, open your basic process that we build in the previous lesson from the process studio.

Just to remind you, this is a basic automation that takes the values in these 2 data items and uses an addition function in a calculation stage to store the result in this 3rd data item

Now, let's add a decision stage to this process that will tell us whether the answer to our calculation is a positive or negative number.

So, let's delete the bottom link and give ourselves some room to add a decision stage. We'll add the decision stage after the calculation stage because we're going to be using the answer the calculation stage has produced and until we've passed through the calculation stage our answer is blank.

Now, let's get a decision stage from the left hand side of the screen and add it to the process. Let double click the decision stage to open its properties. We'll call this stage "Is answer positive?" It's always a good idea to have a question mark at the end of a decision stage title to make it really clear that the decision stage is asking a question and making a decision based on the data it's provided. Another way to think about a decision stage is that if you can't name a decision stage as a question, then you are probably not using it correctly.

So, we want to find out whether our answer is positive dont we? That means that we need to write an expression that will result in either a true or false answer, so how could we do this?

Well we could check if our Answer was greater or less than zero and this would tell us if the answer is a positive or negative number.

So to do this let's drag the answer data item across to the expression window, input the greater than sign and type zero.

A handy piece of functionality that blue prism has is the evaluate expression button. If we click on this button we can test our expression. As you can see though, because our answer data item doesn't have an initial value, Blue Prism can't give us a result just yet. It does however give us the opportunity to add a temporary value. So in another hypothetical scenario, if the value in my answer data item was -3 I could type that in and Blue Prism would evaluate that expression when I click test. As we know, because -3 is not greater than zero a False flag is generated.

So now we're confident that our expression can produce a reasonable answer we can close this window, validate our expression and click ok.

Now we need to connect our decision stage up but when we do that, notice that the first link is is the True or Yes path and the second is False or No. We need to make sure that we have two links coming out of a decision stage or we'll get an error message.

Anchor Stage

So let's link it up and also introduce an anchor stage. An anchor stage doesnt do anything as far as the flow of the process is concerned, but it allows us to make our processes look a bit more organised.

In this case, we also need to add another end stage to the No direction in the event our process goes down this path.

If we're not happy with the order of our yes and no links we can right click on either of these links and click switch. This will simply reverse their orientation as you can see. I'm happy with how it was before so we'll switch it back.

Ok, so before we run this process have a think about what we're expecting it to do.....

Did you get it? If you did well done. What happened was our process did our simple addition calculation, the answer was then sent to the answer data item and in the decision stage the answer data item was interrogated to assess whether the answer to our calculation was greater than zero. Because it was the process that went down the yes path.

Let's have another go, but before we do let's change the initial value or our First number data item to -17.

Double click, change our initial value and click ok. Now let's reset, still no errors, that's a good sign, and click go.

This time you'll notice that our process went down the no path.

Now this is pretty cool but there's not much use just having our process finish at one end stage or another. We generally want to store a result like this into a data item, so let's add a step in both the yes and no flow that will store this result.

First let's give ourselves some more space, delete the links and add a calculation stage to the yes and now paths of the process.

Using Calculation Stages to Write

Let's make them a bit bigger and set them up. We open the properties of the first calculation stage by double clicking and let's call this calculation stage "Write Yes". Now let's look at the expression window. So far we've put mathematical equations or expressions in this area but we can also put whatever we want the calculation stage to store in a data item of our choosing. So in this case lets just type "Yes" in inverted commas. You'll recall that we put text in inverted commas to denote that it's a text data type. We need to do this whenever we refer to text in an expression window or Blue Prism will generate an error message.

Now we also need to store our expression somewhere. We could have made a data item before we came into the calculation properties but let's say we forgot. That doesn't matter, in our "store results in" field that we said is where the result of our expression is sent, we simply type "Is the answer positive?" We then click the little square button next to our heading and Blue Prism will create a data item for us

with the heading that we just inputted. Note that it has also stored this data item under the text heading because it knows based on the inverted commas that we used, that Yes is a text data type.

Now all we need to do is validate our expression and click ok.

Now let's do the same for our second new calculation stage except we want it to write no.

So double click the stage to open the properties, give it a name and we'll fill in the expression area.

This time when we select where we want our result to be stored we don't need to create the data item because it's just been created. All we need to do is open the text data items section and drag "Is the answer positive" into the Store results in section.

Now validate, click ok and link it all up. I'm making sure when I link this part of the process up I pay particular attention to the yes and no icons that are generated by Blue Prism. It would be pretty embarrassing to go to all this effort only to have the yes flow pointing to no and vice versa. This reiterates the importance of labeling our stages really clearly to remove any ambiguity.

Ok, so we're all linked up, let's reset the process, make sure there's no errors, and click go.

Ok, so we can see the answer to our calculation stage was -13, our decision stage was able to determine that number was not greater than zero and it wrote no in the data item we just created.

Let's change -17 back to 7 and see what happens.

Remember to refresh our process and press go.

Now we can see the answer was greater than zero and we were able to write yes in our data item.

Circular Paths

Overview

Hi and welcome to the lesson where we will be discussing circular paths.

In this lesson we will be building another basic process to consolidate what we've learnt so far and we'll be discussing circular paths.

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

As we've discussed, the basic principle of Blue Prism is to automate repetitive work and because of the repetitive nature of many processes, we will often need to build processes that repeat steps over and over again. So far we've only looked at simplistic linear paths but when we want to incorporate repetition in our process we need to build in circular paths. Because repetition is so critical when building many processes, circular paths are one of the main building blocks of Blue Prism and can cover a range of complexity dependent on the complexity of the process we're building.

We've learned that decision stages can interpret an expression and make a True or False decision based on the result of our expression. These decision stages support the incorporation of circular paths within our process and we'll step through that now.

We'll start with a simple example of a circular path that loops around until it has met a predetermined number of loops before continuing on. In this process we'll also use some of the concepts that we've learnt about calculation stages and data items in previous lessons.

Circular Paths

Let's now jump across to Blue Prism and create a new process called Circular Paths Process.

We'll drag two data items onto the page, as well as a calculation stage and a decision stage.

Now let's make our data items a little bigger and set them up. We'll call our first data item Number of Loops, it will be a number data type and it will have an initial value of zero because at the moment we haven't completed any loops. We'll call our second data item max loops, it will also be a number data type and we'll give it an initial value of 5. This will be our predetermined number that our process will need to achieve before continuing on.

Now let's set up our calculation stage. We'll go into the properties window and call it Count. Now, how could I write an expression that increases the value of Loop Count by one each time you cycle through?

The best way to do this is to drag our Number of Loops data item into our expression window and enter +1. Then we want to store that result in the same data item, number of loops. This will increase the number within our data item by one each time we loop around our circular path. We then validate the expression and click ok.

Now we'll set up our decision stage. Call it Loop Again? and what do you think we need to do to get this decision stage to assess whether the loop count is less than or equal to max loops?

If you answered, drag our loop count into the expression window, enter less than or equal to and then drag in max loops then you'd be correct! Let's do that and validate to make sure that we got it right.

Ok, now let's connect it up. We'll use some anchor stages like we did in the previous lesson. We also need to pay attention to the Yes and No output making sure that these are orientated correctly.

Now let's reset the process and press go.

You can see our process start and our loop count go up by one each time it cycles through. You'll also see that because our Loop Count is less than or equal to 5, the process loops around again. and again, and again.

Now once our Loop Count is greater than 5 we can see that the decision stage responds to our expression with a false and the process proceeds to end.

No let's talk about increasing our number of loops and briefly touch on breakpoints which we use for troubleshooting.

The process we just created runs for 5 loops, but what if you wanted to set it to 20 loops, or 50 loops or 100 loops? How would we do that?

Easy, we just change the initial value of Max Loops to however many times we want our process to loop through. Let's change it to 20.

Now this will take a while to run through and we don't want to be here all day so we can increase the speed at which our process runs by clicking the dropdown icon next to the go button. Blue Prism will default to normal running speed but let's crank it up a bit, but let's be careful not to increase the speed of the process too much. Blue Prism can run incredibly fast and if we have this set too fast we won't be able to see anything at all.

Now let's click reset again and run our process. You can see our process is now running much faster than before.

While our process is running we can also pause it and have a look at the current values of our data items based on how far through the process we are.

We can also use the step function to step through our process step by step as you can see here. We'll cover this in a bit more detail in later lessons.

Collections and Loops

Overview

Hi and welcome to the lesson where we will be discussing Collections and Loops.

In this lesson we introduce a stage that we call a collection and discuss how we use collections to store several lines of data in much the same way as a spreadsheet would. We also introduce the concept of loops which allow us to go through data in a collection row by row.

As we've done so far, in this lesson we'll switch between the slides in this video and Blue Prism. To get the most out of the video, make sure you follow my actions in Blue Prism and pause the lesson if you need to.

Collections

So far we've used data items in our process to store data. Using data items has served us well in our basic processes so far but a limitation with these data items is that they can only store one piece of data at a time.

Fortunately, there is a stage that gives us the same functionality as a data item but allows us to store multiple lines of data. We call this a collection.

A collection is a type of data item that can hold multiple values and is arranged like a table or spreadsheet with columns and rows as you might see in Microsoft Excel. Another great thing about collections is that we can store different types of data in them as opposed to data items where we could only select one type of data.

Collection data can be accessed one row at a time by incorporating what we call a loop stage.

Loops

A loop stage is similar to a circular path and the incorporation of a loop stage allows us to step through each line of data within a collection.

For those with a programming or coding background, a comparison could be drawn between a collection and an array. We could also align our loop stages to the concepts of loops or 'for each' statement when coding.

Loops have two components that include a start and end stage. As we said, a loop stage works in conjunction with a collection and cycles through each line of data within the collection that it's linked to.

What this means is that our loop stage will keep cycling through each line of our collection until it's processed all the data within that collection. It's also worth noting that we can add additional stages between the start and end of our loop and those stages will also be cycled through each time our process carries out a loop.

Building a Process using Collections and Loops

Let's jump back into Blue Prism and put some context around Loops and Collections.

Let's create a new process and call it Collections and Loops.

Now, what we would like to do is create an employee IT register that contains the columns username, password, Number of Monitors and Date of Commencement with our company.

Now, we know a Data Item can't handle all this data but itself but a collection could store all this information in columns and rows just like in a spreadsheet. So let's grab a Collection and start inputting some data.

On the left side of the page let's grab a collection and drag it onto the main page. Once it's on our main page we can double click it to get to the properties window.

First, let's call our collection Employee Data. You'll notice that apart from the Name and Description, the properties window looks a bit different to what we saw in our data items. In our collection properties we have the fields tabs which you can think of as columns within a spreadsheet.

To add a field (or columns to our spreadsheet) we simply click the add button. As you can see, we can add as many fields as we like, and similarly we can remove and clear all our fields.

Let's set this collection up to have four fields. You can see that each field can be given a name, a data type and a description.

We'll use the example of an employee IT register so we'll call the first field Username, which will be a Text data type. We'll call our next field "Password" which will be a password data type, next we'll add the "number of monitor's" the user has which will be a number, and finally we'll call our last field "date of commencement" which will be a Date data type. This will make more sense shortly but remember that we can think of fields the same as we think of columns in a spreadsheet.

As we did with data items, we will need to set some initial values for Blue Prism to work with so let's click on our initial values tab. At the moment we don't have any data but if we click the add button you can see that we're effectively creating a row of data in our spreadsheet each time we click add and our fields (or columns) are pre populated.

Now we're getting a better understanding of what this spreadsheet might look like, let's input some data from an employee IT register.

Our first username will be john_smith, his password will be johnny1, he has 2 monitors and he commenced_smith on 20 October 2010. If you remember back to our data types lesson we annotate this using the US data format which is Months, Day, Year.

Our next user is sarah_jones, her password is SJ86, she has 3 monitors and she commenced on 25 November 2011 or 11/25/2011. We'll make our last user carol_baskin with a password crazycats. Carol has 1 monitor and her date of commencement was 30 December 2012 or 12/30/2012.

Even after we've inputted this data we can go back to our fields tab, add a field, like flag, and this will be shown in our initial value tab as you can see here.

We don't need a flag field through so let's go back and remove that and click ok.

Now let's add a loop stage to our diagram. We do this by dragging the loop icon across to our main page. You'll notice that when we drag the loop stage on our main page we can see the start loop icon and when we drop that stage, the end loop stage automatically appears.

Now that we've got a loop stage in place let's open the properties by double clicking the loop start stage. You'll notice this is a very simple properties area compared to what we've seen previously. This is because loops are only used with collections and all we need to do with a loop stage is assign it a collection that it will then loop through.

So if we open the Collection drop down we can see that Blue Prism has detected we only have one collection on this page. Let's select our Employee Data collection and click ok.

Now for ease of observation, let's add a note stage between the start and the end of our loop. Our note stage won't do anything as far as our processes or objects are concerned. They are used to annotate our solution to make it easier to follow if another developer was going to take over from our work.

All that said, adding a note will make it easier for us to see our loop in action, so let's add a note and link it up.

Now before we press play, let's quickly go through what we're expecting to happen. First we're about to see our process move from the start, into our loop start, through our note and to our end loop stage. When Blue Prism detects there are additional rows of data in our collection it goes back to our start loop stage and runs through again. This will continue to happen until the loop has gone through all the rows in the collection and once complete, it will leave the loop and go to end.

Also keep an eye on the collection and the number of the row that we're processing. As we loop through this process we'll see the row of the collection being worked on annotated on our collection. We can see

that we're on the first of three rows at the moment but we'll expect this to increase to 2 of 3 and 3 of 3 as we run this process, and when Blue Prism detects that we've processed 3 of 3 rows, we'll exit the loop and proceed to the end.

First we'll need some calculation stages and some data items that correspond to each of the fields within our collection.

Let's add four calculation stages. We'll call the first one User Name, we'll grab our username from within our collection and drag it into our expression field. Note that because we're essentially talking about a data item within a collection, our syntax for this starts with the name of the collection with a full stop or period, followed by the field we're looking for.

Because we're just using the calculation stage to write to a data item, we don't need anything in the expression window other than this. Now to speed things up let's create our data items while we're here. You'll recall that we do this by going to our "Store Result In" field, typing in what we want our data item to be called and clicking the blue and white button.

We'll do the same for our second calculation but call it Password and set up the expression and the store result in field up.

We'll do the same for our third calculation and call it Number of Monitors and set up the expression and store result in field up the same way.

Finally, we'll call our last calculation stage Date of Commencement and do the same for our expression and store result in field. Through all this you'll notice that the data type has been drawn from our collection and you can see this on the right of our calculation properties window.

Now let's link it all up, reset to check for errors and we're ready to go. This time as we run through our process we'll see the data from our collection being transferred to our individual data item line by line until we've processed all the rows in the collection, at which time our process proceeds to end.

Let's give it a go and see what happens.

Great! As our process looped through we could see that the data from each row of our collection was written to the data items.

I don't know about you, but I found it a little annoying that we had to use four calculation stages to write one row of data to our data items. It'd be great if we has a stage that could carry out multiple calculation in one...

Multi Calcs

Well great news! There is such a stage and it's called a multi calc stage. Let's get rid of our four calculation stages and replace them with one multi calc stage.

Let's grab our multi calc stage, drop it on the main page. If you open the properties window of our multi calc it looks like an abbreviated version of the properties calculation stage but only shows the key areas that we've been using i.e. the expression window and the Store Result In field.

If you're feeling nostalgic about the calculation stage properties you can click on the calculator icon and we can see everything that we had in our single calculation stage except the "Store Result In field". As we saw, this was on the main page so we don't need it in our Expression Chooser window.

I'm not feeling particularly nostalgic so let's go back to the properties window of our multi calc stage and set it up the same as we had when we had four calculation stages.

First, we want to store the username in our collection in our individual username data item. Add a row and next we want to store password within our collection in our password data item. We'll also do this for our number of monitors and date of commencement. Before we're done here we need to validate all our expressions and we do this by going into the properties window, so let's do that now.

Now we're set up click ok, link it up and click reset. So before we press play what are we expecting?

We'll see something very similar to last time but instead of stepping through three calculation stages, we'll get this done all at once and we'll see this reflected in our data items on the right. Once Blue Prism detects that we've run through each row of our collection it will leave the loop and proceed to the end stage as we saw before. So let's give it a go.

Great! That's much cleaner than using three calculation stages. But more importantly than looking good, Blue Prism can process a multi calc stage much faster than multiple calculation stages, and when we're talking about running process that will cycle through a calculation thousands of times, the speed at which a process can be completed and the computer processing power that's required becomes really important.