Blue Prism: Community Challenge

A Digital DA

Blue Prism recommend a series of best practices (a consensus on the best way to do something) for automation builds to ensure reliability, robustness and maintainability; a Design Authority (DA for short) is in charge of ensuring that this Best Practice is adhered to within Blue Prism automations.

Best Practice has basic elements, such as ensuring the t's have been crossed and the i's dotted, which can often be considered as binary i.e. met or not; then more advanced levels where highly skilled insight and analysis is required.

A Design Authority's purpose is to ensure Best Practice has been adhered to, but an astute reader will be able to see the telltale signs of Blue Prism original problem statement – highly skilled resource completing manual, repetitive and time-consuming work.

The Challenge

Blue Prism's own Center of Excellence decided that a worthy challenge for the Blue Prism World (BPW) Community Challenge was to create a "Digital Design Authority". A digital worker capable of validating whether the basic or even intermediate best practices were met on a build.

Each team would have 3 hours to design, document, and build their Digital DA with two key objectives:

1. Check any automation against a defined set of criteria
2. Produce a validation report which includes enough detail for:
    a. A DA to approve the build
    b. A developer to fix the issues

The teams were marked on:

Category	Description
Best Practice	A DA would approve the build
Design	A robust and scalable design
Collaboration	Teamwork makes dreamwork
Performance	How much it can do and how well
Creativity	How unique the approach was

How is this even possible?

Blue Prism Automations all share a unique blueprint, written in a language called **XML (eXtensible Markup Language)** which tells Blue Prism: where to place things, what to place in them, and what conditions to apply to them.

Blue Prism's XML looks a little like this, where the instructions of where to place a start stage and what to do after would look like :

```
<stage name="Start" type="Start">
 <display x="-60" y="-135" />
<onsuccess>End"
</stage>
```

Developers can build automations that review the **XML blueprint** and compare it against the principles of **Best Practice.**

A Symphony of automation

The winning team of the 'BPW Hackathon Challenge' were from **Symphony**, one of Blue Prism's most experienced 'boutique' Partners. Symphony, unlike many teams, avoided complex custom code stages to validate the Blue Prism releases. Instead, they relied on best practice itself and the functionality within the product.

Symphony attacked the challenge by splitting their approach in two, they would create:

- A process: that would find automation files; loop through them; and produce an output/validation report for each
- Objects: custom pieces of Blue Prism functionality that would check the specified criteria

This approach gave them a competitive edge in design, best practice and performance.

- Design: by using objects, you would only ever have to build the process once, and you could drag and drop new objects in depending on what you wanted to check. Thus, a scalable and maintainable automation existed from the start.
- Best Practice; by using Blue Prism functionality, they could apply standard Best Practice and not have to evaluate the ramifications of custom code
- Performance; by using objects, they were capable of addressing one criteria, copy existing functionality into addressing a new one, address it, and copy even more functionality into the next – exponentially speeding them up.

In the end, their solution could take any automation and examine 13 out of the 18 criteria, before compiling the results ready for a report. Sadly 3 hours wasn't quite enough time to finish the report building section!

Noteworthy Contenders

While Symphony took out the prize, there were plenty of other teams who showed their own style; many of whom aimed at code stages to attack the XML and turn it into useful tables for Blue Prism to consume. Some, such as the team from DiRWA, made use of Microsoft's Linq libraries to query the XML in great detail. Others went for more basic XML libraries to ingest the data into collections which could then be looped through.

Blue Prism's Center of Excellence

Blue Prism's CoE team didn't shy away from this challenge either and they took on the challenge (under the same conditions) before BPW. Their approach mirrored Symphony's in many ways, specifically the decision to use utility objects, but their solution design was fundamentally different.

Where they differed was that Symphony were examining the complete XML (all the instructions on how to build the Blue Prism automation) every time they checked a condition. Whereas the CoE's solution broke down that XML into constituent elements before checking them for the criteria. This may not seem like a small difference, but in practice would yield vastly different results.

Symphony's solution would be quicker, it considered the automations as a whole and therefore could review them quicker. When taking a solution as a whole, Symphony didn't have to go looking for anything, if you wanted to check a condition you had everything you needed to check it. Whereas the CoE's solution would have to go find the element that it needed to check to approve another element.

However, the CoE's solution would be more robust as by breaking down the automations before reviewing them, any exceptions wouldn't affect the review – they wouldn't have to start again. As Symphony's solution only considered the automation as a whole, if any part of their solution failed when reviewing the automation – the whole thing failed. Whereas, the CoE's solution considered every element within the automation as a whole part. Therefore, if one failed the review could continue on the other parts.

Summary

The challenge provided a new paradigm to every team about automation within their business; it demonstrated that even your internal RPA team have opportunities for automation. One team even suggested if a digital worker can review automations, could it possibly also fix their issues…?

The answer? **Watch this space.**