

Speedy Incentives from Blaise 5 Instruments

Emily Caron, Jerry Copperthwaite, and Rhymney Weidner, RTI International

1. Abstract

One of the popular ways to encourage respondents to complete a survey is to provide them with a monetary incentive. Blaise 5 does not have a direct way to instantly send incentives to respondents, but by leveraging existing Blaise functionality and PowerShell scripts—along with a separate, specially developed system—we were able to implement “instant” incentives for a recent project. The respondent was able to receive a digital incentive within minutes of completing the survey. This paper will examine features utilized in Blaise 5 and provide a brief explanation of the accompanying scripts and other pieces involved that made this feature possible.

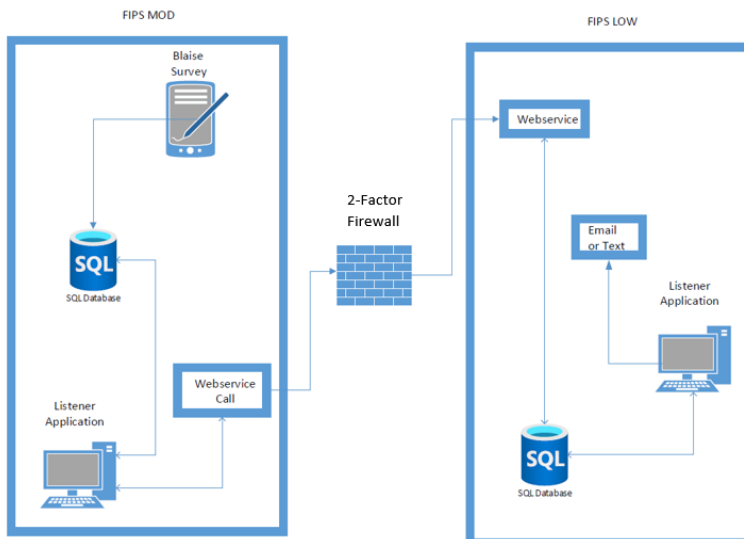
2. Introduction

One of the best ways to encourage respondent participation in a survey is to provide a monetary incentive for completing it. In the past, these incentives were provided by RTI either via a mailed check or an e-gift card. E-incentives were processed by overnight batch jobs, which resulted in a delay of roughly 24 hours. Mailed incentives were slower and took up to a few weeks to reach respondents.

Recently, RTI endeavored to implement a “speedy” incentives process, which enabled respondents to receive an incentive via email or text within minutes of completing the survey if one of those digital delivery methods was selected. These incentives came in the form of an e-gift card that could be retrieved via a link included in the email or text (as selected by the respondent). This paper will cover the specialized process developed to trigger these incentives, focusing on the Blaise 5 features utilized to hook into the backend processes that take care of delivering the incentives. We will also outline some of the challenges encountered and lessons learned while implementing this process.

3. High-Level Overview of the Speedy Incentives Process

Figure 3a. An Overview of the Speedy Incentives Process Flow

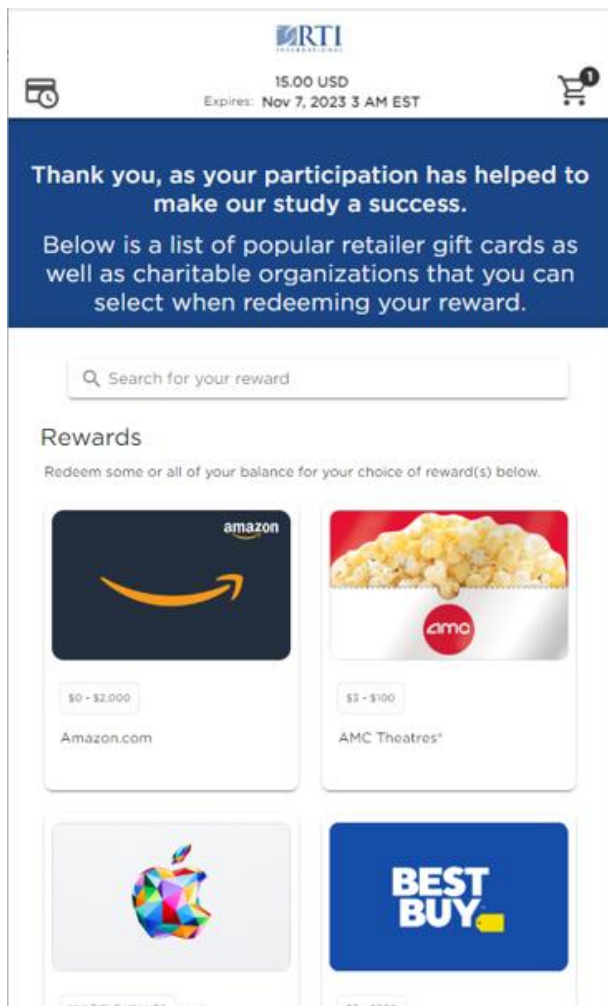


The processes in the Figure3a diagram flow from the FIPS Moderate (FIPS Mod) side of the firewall to the FIPS Low side. Our FIPS Mod network is a standalone, dedicated network that employs a highly restrictive set of security controls and requires multifactor authentication. This is where the Blaise-collected survey data are stored. The starting point of the speedy incentives process is the Blaise survey, which contains logic to write incentive metadata to a SQL database table. The next process is a Listener Application that runs in the background in the FIPS Mod network and contains a timer to query the SQL database table for new incentive records inserted via the Blaise survey.

If a new record is found, an insert function of a web service located in the FIPS Low network is called to load the incentive metadata to a corresponding SQL database table. Next is another Listener Application that runs in the background in FIPS Low. This process contains a timer to query the SQL database table for new incentive records inserted via the web service. If a new record is found there, the process calls an incentive API to purchase a new incentive, which is then sent to the respondent via email or text.

A Simple Mail Transfer Protocol application is used for building and sending the email messages, and an in-house ARTEMIS application builds and sends the text messages. The email and text messages contain a link for respondents to claim their incentive via Tango®, where they have a number of choices for “cashing in” their incentive amount.

Figure 3b. Example Screen for Redeeming Incentives

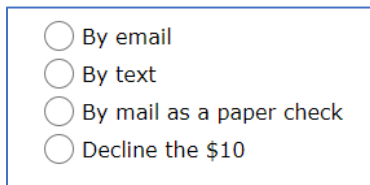


4. Blaise Features Utilized to Trigger Incentives

The first implementation of the speedy incentives process triggered by a Blaise survey was in a survey designed in version 5.12.8. Triggering the incentive required a specialized template, a call to a PowerShell script from an actions setup function, and a call to a stored procedure from the PowerShell script.

While developing the templates for the incentive process, we had to determine at what point we wanted to send the incentive and how we could prevent duplicate incentive attempts. Even though the backend processes contained checks to prevent duplicates, we wanted to avoid unnecessarily triggering any of these from Blaise in the first place. First, it was decided that the incentive should be sent as soon as the respondent completed the screens containing incentive questions. This location was towards the end of the survey, although it was not required that respondents fully *complete* the survey to receive the incentive. The respondent could elect to receive their incentive either via email or text (digital delivery) to be eligible for a “speedy” type of incentive. Other options not involving this new process included receiving the incentive via mail or choosing not to receive any incentive.

Figure 4a. Incentive Delivery Options



By email
 By text
 By mail as a paper check
 Decline the \$10

After selecting one of the digital incentive methods, the respondent is routed to a screen where their email address or cell phone number is collected, depending on which delivery mode was selected. This is followed by the appropriate screen routing to confirm the email or cell phone number, which also serves as verification of consent for this type of contact. On these confirmation screens, a special “SpeedyIncentive” template is used to trigger the actions setup process from the OnTryLeavePageForward event. On the screen immediately following the confirmation screens, the “Back” button is removed to prevent respondents from backing up and retriggering the actions setup.

Figure 4b. The SpeedyIncentive Process is Triggered in the “OnTryLeavePageForward” Event of a Special Master Page Template

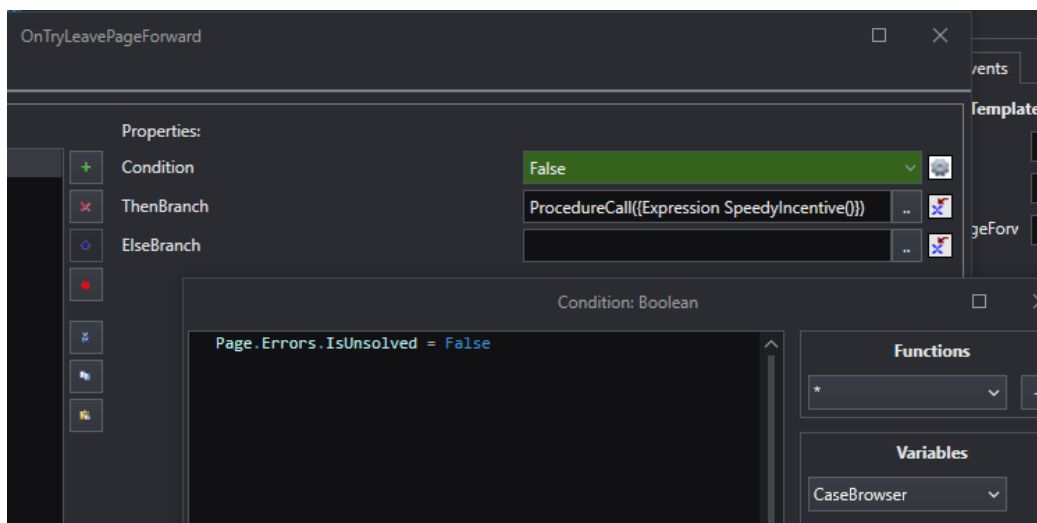


Figure 4c. Confirmation of Email Address After Email Incentive Is Selected; When the Respondent Selects “Yes” and Clicks “Next,” the SpeedyIncentive Process Is Triggered

Just to confirm, would you like us to send your gift card by email to the following address?

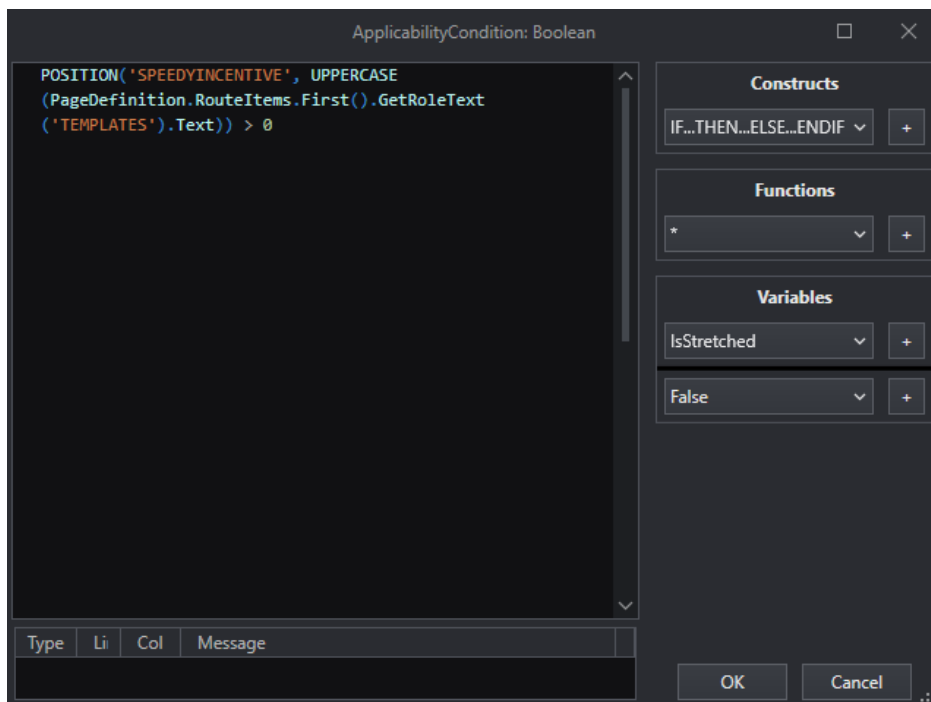
fakeaddress@test.com

Yes

No

The speedy incentive template is automatically assigned to appropriate fields using an applicability condition on the template. Each field that should trigger a speedy incentive has “SpeedyIncentive” in the Templates role.

Figure 4d. This Applicability Condition Ensures That All Fields with “SpeedyIncentive” Included in the Templates Role Are Assigned the Speedy Incentive Template



The “SpeedyIncentive” process is in the “Actions Setup” file defined in the project settings and mapped using Blaise 5’s “Mappings Management” screen. The process pulls the relevant information from the active survey using the SURVEYRECORD file definition. This information is then passed to the PowerShell scripts, as shown in Figure 4e.

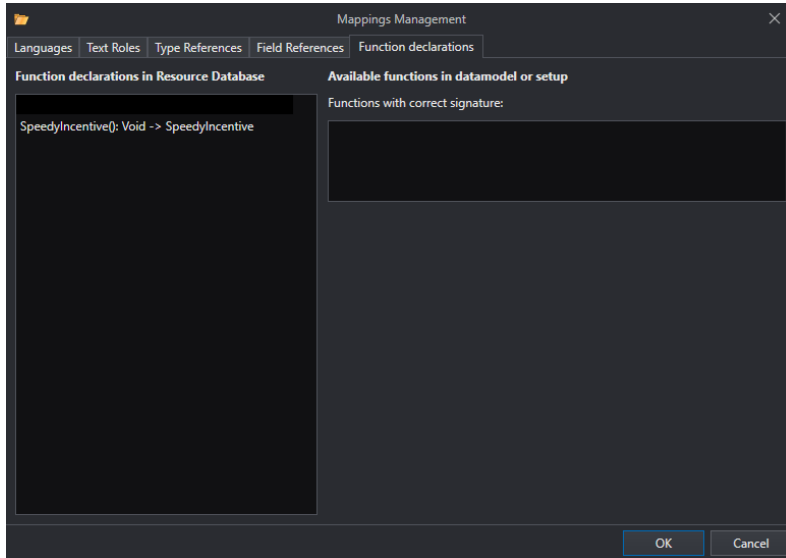
Figure 4e. Code in the Actions Setup Process That Calls the PowerShell Scripts

```
IF UPPERCASE(SelectedMethod) = 'EMAIL' THEN
  //IF Email: Incentive
  strRun := 'PowerShell.exe -ExecutionPolicy Bypass -File "SpeedyIncentiveEmail.ps1" -caseid '"+CaseID+"' -email '"+Email+"' -fullname '"+ FullName+"' -blaiseid '"+ GUID + "' -amount '+ Amount
  aResult := RUN(strRun,WAIT,HIDE)

ELSEIF UPPERCASE(SelectedMethod) = 'TEXT' THEN
  //IF Text: Incentive
  strRun := 'PowerShell.exe -ExecutionPolicy Bypass -File "SpeedyIncentiveText.ps1" -caseid '"+CaseID+"' -phone '"+ Cell + "' -fullname '"+ FullName+"' -blaiseid '"+ GUID + "' -amount '+ Amount
  aResult := RUN(strRun,WAIT,HIDE)

ENDIF
```

Figure 4f. The Process Must Be Mapped to a Function Declaration in the Resource Database to Be Called from the Appropriate Template



The PowerShell script then makes the final call to the stored procedure that will insert an entry to a specific SQL database in the FIPS Mod network. The information added to the table includes case ID with attached language indicator (surveys conducted in multiple languages aren't an issue), confirmed email or cell phone number, a placeholder value for the full name (since the resulting text message or email does not include the respondent's name in this case), the Blaise GUID (for identification purposes since the table is shared with other surveys), and the amount of the incentive to be sent. The Listener Application will pick up new incentive information from this SQL table and kick off the rest of the process discussed in Section 3, "High-Level Overview of the Speedy Incentives Process."

5. Challenges and Lessons Learned

5.1 Initial Setup and Testing Challenges

Initial challenges in the FIPS Low environment where we conducted testing included getting an appropriate PowerShell script programmed to trigger the insert into the stored procedure. We iterated through a number of test attempts while tweaking the call, both to adjust the syntax in general and to account for passing new parameters when new information was found to be needed in the incentives database—namely, for language indicator and GUID.

We also had trouble with the mapping of the actions setup function. Blaise 5 appeared to drop the mappings frequently, so we consistently checked to make sure it was still mapped correctly and remapped when needed.

After moving programs into the FIPS Mod network, the first set of tests for our new process was unsuccessful. Once we had confirmed everything in Blaise was running as expected as part of troubleshooting, we tested the PowerShell script by running it manually. Attempts to run the PowerShell script manually resulted in the error shown in Figure 5a. The firewall required modification to allow communication between the server hosting the Blaise instrument and the server hosting the SQL Server. After proper ports were opened, the process was successful.

Figure 5a. The Error Message Displayed When Initially Running the PowerShell Script from the Webserver to Insert into the SQL Table for Speedy Incentives

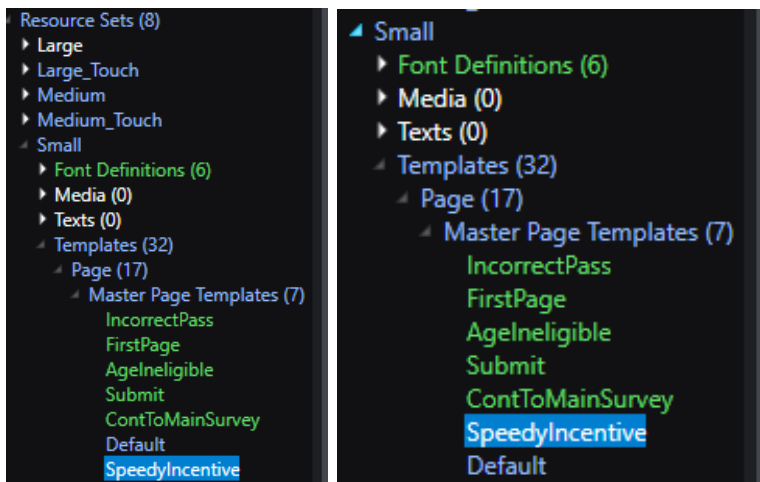
```
PS C:\InstallBlaise\ > .\test1.ps1
WARNING: A network-related or instance-specific error occurred while establishing a connection to SQL Server. The
server was not found or was not accessible. Verify that the instance name is correct and that SQL Server is configured
to allow remote connections. (provider: Named Pipes Provider, error: 40 - Could not open a connection to SQL Server)
PS C:\InstallBlaise\ > -
```

5.2 The Importance of Thoroughly Testing All Layout Sets

One of the most important lessons learned from implementing the speedy incentive process is that *all* layout sets need to be thoroughly tested for speedy incentive functionality. Our process relied on the PowerShell script being triggered by a function associated with a specific template. Without that template, the entire house of cards would collapse, and the speedy incentives wouldn't go out.

Initial instrument testing was heavily focused on the large layout set due to 508-compliance needs and testing out special text and functionality related to CATI mode. The small layout set was tested, but fewer test cases went through the small layout set, and unfortunately, it appeared most testers did not select digital incentive modes. Shortly after launch, it was discovered that surveys completed in the small layout set were not receiving digital incentives. While implementing the speedy incentive template in the small layout set, there was an issue with hierarchy that originally went unnoticed. The speedy incentive template was below the default Master Page template, which led to screens that should have had the speedy incentive template receiving the default Master Page template instead. Incentives are not triggered on the default Master Page template, which meant these small layout set cases were missing a key step in the speedy incentive process.

Figure 5b. On the Left Is an Image of the Incorrect Hierarchy for Master Page Templates; This Hierarchy Allowed the Assignment of the "Default" Master Page Template Prior to the Applicability Condition on the SpeedyIncentive Template Being Considered; The Image on the Right Shows the Correct Hierarchy for the Master Page Templates



As soon as this issue was discovered, an update was released with this slight, but very important, error correction. Yet another lesson for future development: make sure to double check the assigned template for key fields when functionality is dependent on the correct template.

Once the error was corrected, queries were run against the survey data and speedy incentives dataset to pinpoint cases that had elected to receive a speedy incentive but had no entry in the speedy incentives table. We manually triggered incentives for these cases by constructing appropriate calls to the PowerShell script to insert the data into the speedy incentives table. The queries used to find cases missing from the incentives table became part of future, regular quality check routines to ensure that the speedy incentives process continued to work as expected once the small layout set was fixed.

Another lesson learned from this mistake is that the number of respondents using mobile devices to complete the survey was far higher than expected. Based on the number of missing incentives, over 50% of cases were being completed on a mobile/small screen device. Future surveys should include an equal amount of testing between the large and small layout sets.

5.3 Consider Carefully When Events Are Executed

After the correction was made for the small layout sets not receiving incentives, another issue was discovered. When respondents attempted to move forward without selecting a response on the email address or cell phone number confirmation screens, a double entry was made in the speedy incentives table. This issue stemmed from a misunderstanding of where the `OnTryLeavePageForward` event was called. It was assumed that this event was not called until just before the page was left, but it was also triggering as soon as the “Next” button was clicked. This resulted in the speedy incentives process being called twice in situations when an error occurred on the page. An entry for the case was inserted when the error was triggered, and then inserted a second time when the respondent corrected the issue and was able to move forward.

Fortunately, the backend processes ensured that only one incentive was sent per case in these situations. To correct this issue, another build was released with a conditional statement checking to make sure no errors existed on the page prior to calling the speedy incentives process. The corrected `OnTryLeavePageForward` event is shown in Figure 4b. The conditional statement was added to make sure the `SpeedyIncentive` process was not triggered until any errors on the page got resolved.

6. Future Uses and Potential Enhancements

The backend processes involved (refer to Section 3) were set up to handle receiving entries into the speedy incentives database not only from Blaise software, but also from other survey data collection systems utilized by RTI International, which makes the process flexible enough to serve a wide array of projects.

We foresee future projects making use of this speedy incentive process, as it adds a new layer to the methodologies behind incentives by rewarding the respondent with a form of immediate gratification. Our survey included a household screener component followed by a selected respondent survey, so we felt getting the incentive into the hands of the screener participant ASAP would be highly encouraging for participation in the survey component.

Situations where sending other timely digital communications to respondents during or immediately after their survey would be useful could also be considered for this process. A couple of examples: (1) specific

reminders to the respondent or someone in their household for completing additional surveys; (2) handing another portion of the survey off to a coworker as part of a business survey where different sections are completed by various individuals. As clients approach us with challenging criteria for complex surveys, we will have this process available in our toolbox.

7. Conclusion

We are continuing to monitor this new speedy incentives feature and will be analyzing results of its use. For example, it will be interesting to review survey component participation rates for cases where the screener respondent opted for a digital incentive vs. a mailed incentive or no incentive. Lessons we learned will surely be applied to future projects that implement this feature.