CONFERENCE 2009 EXERCISE HANDOUTS

Beginning Scripting – EMSE Exercises

Accela Services

Getting Started

- Login into Accela Automation
  - Agency: scriptb
  - User: userXX (where XX is your user number, ie. user03 or user18)
  - Password: Same as user, all capital letters

Installing Scripts in Accela Automation

- The master scripts have already been installed for the training class so please follow along.

- Copy the master script text
  - In Windows Explorer, browse to the script folder
  - Locate Master Script to Install
  - Right click on the file, open with Notepad
  - Select the text (CTRL-A)
  - Copy the text to the clipboard (CTRL-C)

- Install the master script
  - Go to Admin Tools -> Events -> Scripts
  - Click “Submit” to get a list of Scripts
  - Click “Add” to add a new script
  - Script Code: AppSpecInfoV1.5
  - Script Title: AppSpecInfoV1.5
  - Script Initializer: Blank
  - Script Content : Paste from clipboard (CTRL-V)
  - Click “Add”

- Enable the event
  - Go to Admin Tools -> Events -> Events
  - Click “Submit” to get a list of Events
  - Click “Add” to add a new event
  - Select “ApplicationSpecifcInfoUpdateAfter”
  - Click “Add”

© 2008 Accela, Inc.

- Attach the script to the event
    - Click the red circle to edit the event
    - Select “AppSpecInfoV1.5” in the Script Name drop-down
    - Click “Save” to save your settings

- Test the event and master script
    - Go to Daily -> Application -> Application Specific Info
    - Change / Add some data
    - Click “Save”

Standard Choices Setup

- Scripts begin running from the standard choice identified by the variable controlString identified in Configurable Parameter section of master scripts. Examples are:
    - ApplicationSubmitAfter
    - WorkflowTaskUpdateBefore
    - WorkflowTaskUpdateAfter
    - InspectionResultSubmitAfter
- Scripts start and end from this standard choice.
- For this class Variable Branching has already been setup:
    - Cap Type: Building / User 15 / NA / NA
    - Master Script Branches:
        - ApplicationSubmitAfter -> ASA:Building/User 15/NA/NA
        - WorkflowTaskUpdateBefore -> WTUB:Building/User 15/NA/NA
        - WorkflowTaskUpdateAfter -> WTUA:Building/User 15/NA/NA
        - InspectionResultSubmitAfter -> IRSA:Building/User 15/NA/NA
- All exercises in this class will be coded from the above standard choices. Be sure when creating test caps that you create the correct one based on your User number.

Exercise 1 – ApplicationSubmitBefore – Validating Application Data

Objective:

- When application is submitted make sure that the project type is not a “Shed”.

Exercise:

- For the event Application Submit Before, perform the following:
    - Check the application specific information field “Project Type”.
        - If the “Project Type” equals “Shed” cancel the transaction and display a message to the user, “No more sheds are allowed to be built until further notice”.

Functions:

- The following function(s) are required to complete this exercise:
    - comment(cstr)
        - cstr (string) – Message to display to user

Exercise Notes:

- Standard Choice: ASB:Building/User XX/NA/NA
- Remember the ASI field values are referenced in standard choices by { } brackets. For example, the field Project Type is {Project Type}. This is used only when retrieving the value from that field.
- Equals in JavaScript is ==
- To stop the execution of an event from a before event, set the variable cancel = true
- showMessage must be set to true to display a message to a user.

Exercise 2 – ApplicationSubmitAfter – Updating Application Specific Info

Objective:

- Upon application submittal edit application specific info fields with a calculated value and a global variable value.

Exercise:

- For the event Application Submit After, perform the following:
  - Update the following Application Specific Info fields:
    - Total Area – Update this field with the result of Length * Width
    - City Contact – Update this field with the user id submitting the application

Functions:

- The following function(s) are required to complete this exercise:
  - editAppSpecific(itemName,itemValue)
    - itemName (string) – App Spec Info field to edit
    - itemValue (string) - Value that the app spec info field itemName should be changed to.

Exercise Notes:

- Standard Choice: ASA:Building/User XX/NA/NA
- Remember the ASI field values are referenced in standard choices by { } brackets. For example, the field length is {Length}. This is used only when retrieving the value from that field.
- When naming the field to edit for the itemName parameter, surround the field name with double quotes. For example, “Total Area”.
- The current user id is stored in the global variable: currentUserID

Exercise 3 – WorkflowTaskUpdateAfter – Assess/Invoice Application Fee

Objective:

- Assess/Invoice a fee after a workflow task update.

Exercise:

- For the event Workflow Task Update After, perform the following:
  - Upon the task “Application Acceptance” being updated to the status of “Accepted”:
    - Assess and invoice the Application Fee.

Functions:

- The following function(s) are required to complete this exercise:
  - addFee(fcode,fsched,fperiod,fqty,finvoice)
    - fcode (string) – Fee code to be added
    - fsched (string) – Fee schedule of the fee to be added
    - fperiod (string) – Fee period to be used
    - fqty (integer) – Quantity to be added
    - finvoice (string) – Flag for invoicing (“Y” or “N”)

Exercise Notes:

- Standard Choice: WTUA:Building/User XX/NA/NA
- Application Fee Details
  - fcode = BLD_APP
  - fsched = BLDRES
  - fperiod = FINAL
  - fqty = 1
  - finvoice = Y
- When using calling a function remember that when hard coding a string in the function call to surround the string with double quotes. For example, “BLD_APP”. Double quotes are not needed around integers.
- The following are workflow specific variables available when defining your criteria:
  - Workflow Task – wfTask
  - Workflow Status - wfStatus

Exercise 4 – WorkflowTaskUpdateBefore – Validate Payment before Issuance

Objective:

- Validate that balance due is zero prior to allowing a workflow event to continue.

Exercise:

- For the event Workflow Task Update Before, perform the following:
    - Prevent the workflow task “Permit Issuance” from being updated to the status of “Issued” if the balance for the permit is greater than zero.

Functions:

- The following function(s) are required to complete this exercise:
    - comment(cstr)
        - cstr (string) – comment to display

Exercise Notes:

- Standard Choice: WTUB:Building/User XX/NA/NA
- The following are workflow specific variables available when defining your criteria:
    - Workflow Task – wfTask
    - Workflow Status - wfStatus
- The balance for the permit is available via a global variable:
    - balanceDue
- The comment function is used to capture text to display to the user.
- To prevent an event from occurring and displaying a message to the user the following code can be used:
    - showMessage = true; comment(“Text to display”); cancel = true;

Exercise 5 – WorkflowTaskUpdateAfter – Issuance Automation

Objective:

- Update Application Specific Info Date field and Schedule an Inspection for X amount of days in the future.

Exercise:

- For the event Workflow Task Update After, perform the following:
  - When the task “Permit Issuance” is set to the status of “Issued”:
    - Set the Application Specific Info field “Expiration Date” to 180 days in the future.
    - Schedule a “Final Inspection” for 5 days in the future.

Functions:

- The following function(s) are required to complete this exercise:
  - editAppSpecific(itemName,itemValue)
    - itemName (string) – App Spec Info field to edit
    - itemValue (string) - Value that the app spec info field itemName should be changed to.
  - dateAdd(date,numDays,workDays)
    - date (string) - Starting date, in format “MM/DD/YYYY” (or any string that will convert to JS date). If null is used, date will be the current date.
    - numDays (integer) - Number of days to add to date. Use negative number (e.g. –20) to subtract days from date.
    - workdays (string) (optional) – “Y” if numDays workdays should be added to date. Omit if numDays calendar days should be added to date.
  - scheduleInspection(inspDesc,daysAhead)
    - inspDesc (string) – inspection description
    - daysAhead (integer) - Number of days in the future to schedule the inspection for.

Exercise Notes:

- Standard Choice: WTUA:Building/User XX/NA/NA
- The following are workflow specific variables available when defining your criteria:
  - Workflow Task – wfTask
  - Workflow Status – wfStatus
- The dateAdd() function can be used as the itemValue parameter in the editAppSpecific() function.
- Script controls can be continued onto a second line by starting the next line with: ^
- Final Inspection details:
  - inspDesc = “Final Inspection”

Exercise 6 – InspectionResultSubmitAfter – Synchronizing Results with the Workflow

Objective:

- Upon submittal of inspection results synchronize with the workflow.

Exercise:

- For the event Inspection Result Submit After, perform the following:
  - When the inspection “Final Inspection” result is “Approved”:
    - Close the workflow task “Inspection” with a status of “Approved”, with a workflow comment that matches the result comment.
    - Close the workflow task “Closure” with a status of “Closed”
  - When the inspection “Final Inspection” result is “Denied”:
    - Update the workflow task “Inspection” with a status of “Denied”, with a workflow comment that matches the result comment.
    - Schedule another “Final Inspection” for 5 days in the future.

Functions:

- The following function(s) are required to complete this exercise:
  - closeTask(wfTask, wfStatus, wfComment, wfNote)
    - wfTask (string) – workflow task name
    - wfStatus (string) – status to update
    - wfComment (string) – comment to add
    - wfNote (string) – note to add to workflow task
  - updateTask(wfTask, wfStatus, wfComment, wfNote)
    - wfTask (string) – workflow task name
    - wfStatus (string) – status to update
    - wfComment (string) – comment to add
    - wfNote (string) – note to add to workflow task
  - scheduleInspection(inspDesc,daysAhead)
    - inspDesc (string) – inspection description
    - daysAhead (integer) - Number of days in the future to schedule the inspection for.

Exercise Notes:

- Standard Choice: IRSA:Building/User XX/NA/NA
- The following are inspection specific variables available when defining your criteria or actions:
  - Inspection Type – inspType
  - Inspection Result – inspResult
  - Inspection Result Comment - inspComment

- When updating the workflow via EMSE must use the correct workflow function based on the desired action: updateTask (No Change), closeTask (Go to Next Task), branchTask (Go to Branch Task), loopTask(Go to Loop Task)
- Script controls can be continued onto a second line by starting the next line with: ^

Solutions

Exercise 1:

Standard Choice: ASB:Building/User XX/NA/NA

```
01    true ^ showMessage = false; showDebug = false;
02    {Project Type} == “Shed” ^ showMessage = true; comment(“No more sheds are allowed to be
      built until further notice”); cancel = true;
```

Exercise 2:

Standard Choice: ASA:Building/User XX/NA/NA

```
01    true ^ showMessage = false; showDebug = false;
02    true ^ editAppSpecific("Total Area",{Width}*{Length});
03    true ^ editAppSpecific("City Contact",currentUserID);
```

Exercise 3 & Exercise 5:

Standard Choice: WTUA:Building/User XX/NA/NA

```
01    true ^ showMessage = false; showDebug = false;
02    wfTask == "Application Acceptance" && wfStatus == "Accepted" ^
      addFee("BLD_APP","BLDRES","FINAL",1,"Y");
03    wfTask == "Permit Issuance" && wfStatus == "Issued" ^ editAppSpecific("Expiration
      Date",dateAdd(null,180));
04    ^ scheduleInspection("Final Inspection",5);
```

Exercise 4:

Standard Choice: WTUB:Building/User XX/NA/NA

```
01    true ^ showMessage = false; showDebug = false;
02    (wfTask == "Permit Issuance" && wfStatus == "Issued") && balanceDue > 0 ^ showMessage
      = true; comment("Cannot issue permit until fees are paid, balance due is $" + balanceDue);
      cancel = true;
```

Exercise 6:

Standard Choice: IRSA:Building/User XX/NA/NA

```
01    true ^ showMessage = false; showDebug = false;
02    inspType == "Final Inspection" && inspResult == "Denied" ^
      updateTask("Inspection","Denied",inspComment,"updated via script")
```

```
03      ^ scheduleInspection("Final Inspection",5);
04      inspType == "Final Inspection" && inspResult == "Approved" ^
        closeTask("Inspection","Approved",inspComment,"updated via script");
05      ^ closeTask("Closure","Closed","closed via script","closed via script");
```

- Solutions can also be found in Accela Automation under the following standard choices:
  - ASB:Building/User XX/NA/NA
  - ASA:Building/User XX/NA/NA
  - WTUA:Building/User XX/NA/NA
  - WTUB:Building/User XX/NA/NA
  - IRSA:Building/User XX/NA/NA