Asheville Master Script Upgrade

Summary

Accela has upgraded the agency scripts in the Asheville agency from 1.x and 2.x formats to the newest scripting 3.0 format. The original scripts that were stored in standard choices have been migrated into JavaScript files, using the Accela best practice directory structure. All custom functions that were embedded in master script code have been analyzed and migrated into the custom function library. The standard choices have been converted to a structure that mirrors the record type structure for the agency. This document describes the details of the conversion.

Document Control

Date	Author	Version	Change Reference
10/31/16	John Schomp	0.1	Working Draft

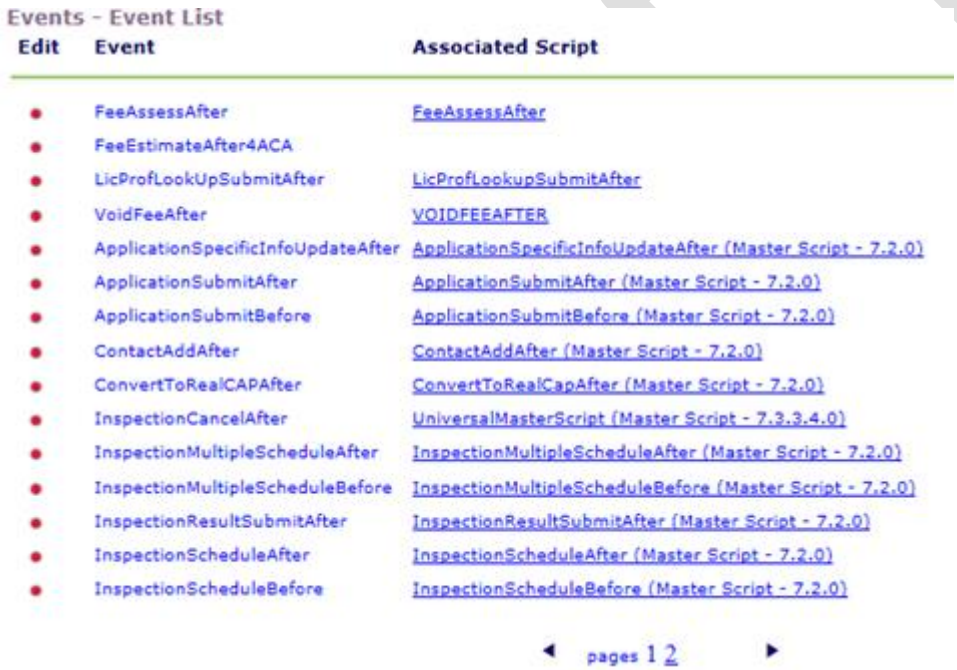
Analysis

A review of the scripting shows that customary standards for scripting based on version 1.X have been used. "Variable branching" did not exist in these versions, so was not used.

Several opportunities for code review and/or refactoring were identified. Please see the Recommendations section for details.

Master Script Usage

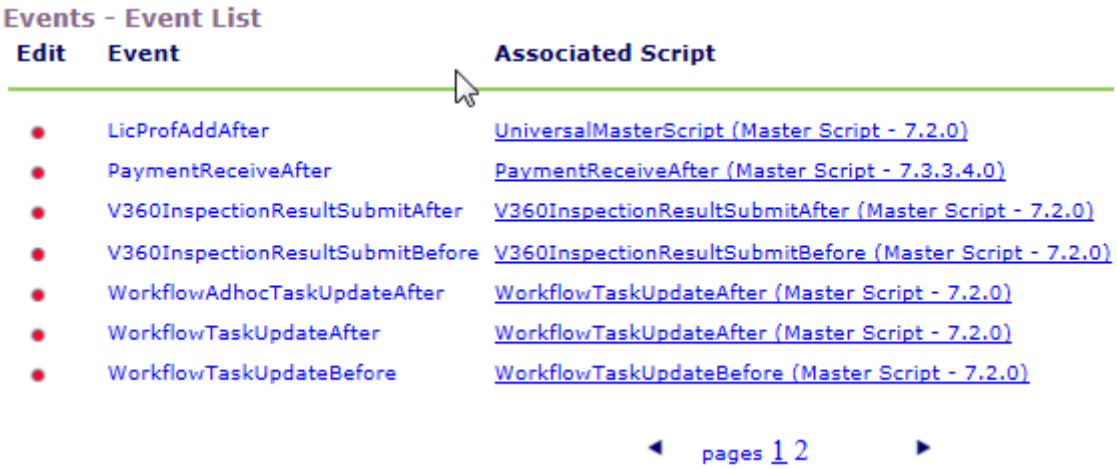
The following events are in use for the Ashville agency, shown with associated master scripts.



Events - Event List

Edit	Event	Associated Script
•	FeeAssessAfter	FeeAssessAfter
•	FeeEstimateAfter4ACA	
•	LicProfLookUpSubmitAfter	LicProfLookupSubmitAfter
•	VoidFeeAfter	VOIDFEEAFTER
•	ApplicationSpecificInfoUpdateAfter	ApplicationSpecificInfoUpdateAfter (Master Script - 7.2.0)
•	ApplicationSubmitAfter	ApplicationSubmitAfter (Master Script - 7.2.0)
•	ApplicationSubmitBefore	ApplicationSubmitBefore (Master Script - 7.2.0)
•	ContactAddAfter	ContactAddAfter (Master Script - 7.2.0)
•	ConvertToRealCAPAfter	ConvertToRealCapAfter (Master Script - 7.2.0)
•	InspectionCancelAfter	UniversalMasterScript (Master Script - 7.3.3.4.0)
•	InspectionMultipleScheduleAfter	InspectionMultipleScheduleAfter (Master Script - 7.2.0)
•	InspectionMultipleScheduleBefore	InspectionMultipleScheduleBefore (Master Script - 7.2.0)
•	InspectionResultSubmitAfter	InspectionResultSubmitAfter (Master Script - 7.2.0)
•	InspectionScheduleAfter	InspectionScheduleAfter (Master Script - 7.2.0)
•	InspectionScheduleBefore	InspectionScheduleBefore (Master Script - 7.2.0)

◄ pages 1 2 ►



Events - Event List

Edit	Event	Associated Script
	LicProfAddAfter	UniversalMasterScript (Master Script - 7.2.0)
	PaymentReceiveAfter	PaymentReceiveAfter (Master Script - 7.3.3.4.0)
	V360InspectionResultSubmitAfter	V360InspectionResultSubmitAfter (Master Script - 7.2.0)
	V360InspectionResultSubmitBefore	V360InspectionResultSubmitBefore (Master Script - 7.2.0)
	WorkflowAdhocTaskUpdateAfter	WorkflowTaskUpdateAfter (Master Script - 7.2.0)
	WorkflowTaskUpdateAfter	WorkflowTaskUpdateAfter (Master Script - 7.2.0)
	WorkflowTaskUpdateBefore	WorkflowTaskUpdateBefore (Master Script - 7.2.0)

pages 1 2

Edits to Standard Master Scripts

With minor exceptions, the master scripts in use are "Productized" which means that they are read-only and cannot be edited. Two master scripts (FeeAssessAfter, LicProfLookupSubmitAfter) are non-productized versions from the V1.6 master script distribution. These have not been edited. The VoidFeeAfter script is not a master script – it performs hardcoded functions for this event and won't be replaced.

Custom functions that conflict with Master Script distribution

Custom functions were found that matched standard functions that are part of the Master Script 3.0 distribution. The code was compared and the following recommendations were made.

Function	Description	Disposition of custom function
addAdHocTask	Matches distribution	Removed
updateFee	Matches distribution	Removed
updateFeeItemInvoiceFlag	Matches distribution	Removed

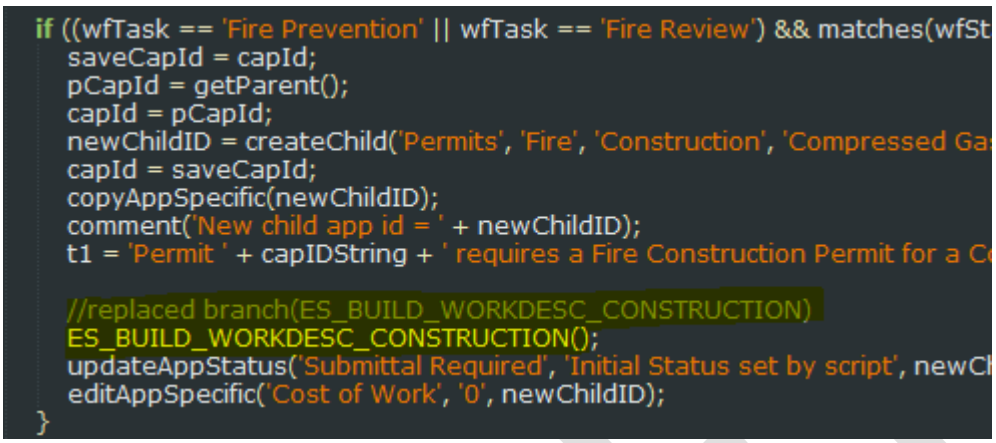
Functions with multiple versions

There were no functions found with multiple versions.

Standard Choice Conversion Details

- Approximately 1,252 Standard Choices containing scripts have been converted to 28 script files. 7 new functions were created.

- In general, any standard choice code that was branched to more than once has been created as a separate custom function. In most cases the function name is the same or similar to the name of the original standard choice. For example:

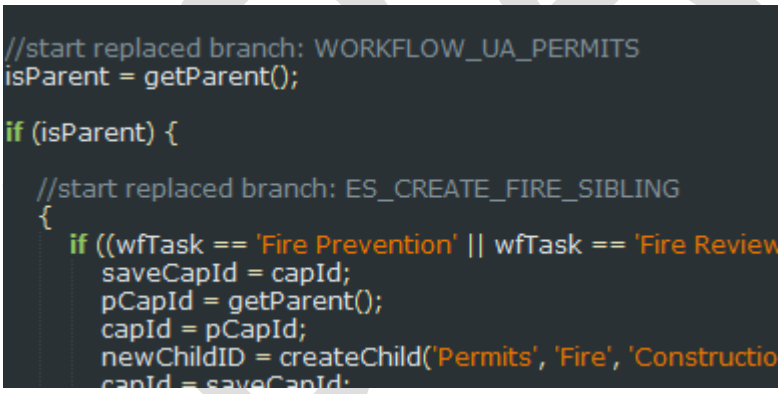


```
if ((wfTask == 'Fire Prevention' || wfTask == 'Fire Review') && matches(wfSt
    saveCapId = capId;
    pCapId = getParent();
    capId = pCapId;
    newChildID = createChild('Permits', 'Fire', 'Construction', 'Compressed Ga
    capId = saveCapId;
    copyAppSpecific(newChildID);
    comment('New child app id = ' + newChildID);
    t1 = 'Permit ' + capIDString + ' requires a Fire Construction Permit for a C

    //replaced branch(ES_BUILD_WORKDESC_CONSTRUCTION)
    ES_BUILD_WORKDESC_CONSTRUCTION();
    updateAppStatus('Submittal Required', 'Initial Status set by script', newCh
    editAppSpecific('Cost of Work', '0', newChildID);
}
```

- All converted scripts have been tested for proper JavaScript syntax.

- To assist with testing, the original "Branch" standard choice names have been added as comments to the converted scripts. Once testing is complete these comments won't be needed. For example:



```
//start replaced branch: WORKFLOW_UA_PERMITS
isParent = getParent();

if (isParent) {

  //start replaced branch: ES_CREATE_FIRE_SIBLING
  {
    if ((wfTask == 'Fire Prevention' || wfTask == 'Fire Review
      saveCapId = capId;
      pCapId = getParent();
      capId = pCapId;
      newChildID = createChild('Permits', 'Fire', 'Constructio
      capId = saveCapId;
```

- The conversion algorithm may have created some instances where double sets of braces {{ … }} denote a code block. These can be removed if desired, but they have no impact on the script execution.

- All references to showDebug within business scripts have been removed in order to simplify script debugging. Accela recommends setting this in one place, preferably using INCLUDES_CUSTOM_GLOBALS for this purpose. For example:

```
if (matches(currentUserID,'KHOBDAY','DMEEK')) {
        showDebug = true;
```

```
}
```

- Individual standard choices that were disabled are included but commented out inline. Any branch calls in this disabled code were migrated as "br_nch" as the branch function is no longer supported in 3.0. If this code needs to be enabled, these branch calls would have to be refactored. For example:

```
//start replaced branch: APP_SA_BRANCH_PERMITS

// DISABLED: APP_SA_BRANCH_PERMITS:1
//copyParcelGisObjects();
// DISABLED: APP_SA_BRANCH_PERMITS:2
//br_nch('ES_GET_PARCEL_ATTRIBUTES');
```

- Standard choice branches that were not referenced by any script were converted into the Scripts\Events\Unused Branches folder. This includes any code that was referenced by disabled standard choices. These should be reviewed.

- All script files have been formatted to JavaScript standards for indentation.

- All references to disableTokens global variable have been removed. This flag is deprecated in scripting 3.0. Curly braces ({}) can (and should) be used to designate bode blocks in 3.0 scripts.

- The caret (^) is no longer valid for creating conditional logic. Standard JavaScript should be used instead (i.e., if, else, else if, switch)

- All references to ASI/TSI/Parcel Attributes using curly braces (e.g., {ASI Field}) have been changed to use the AInfo global Array (e.g., AInfo['ASI Field']). Curly braces are now exclusively used to designate blocks of code.

- All references to the "branch" and "endbranch" functions have been removed. These are deprecated in Master Scripts 3.0.

Code Analysis and Recommendations

Some recommendations on the existing code are as follows:

- There are some script lines used to create outgoing emails. There is new email template functionality that can be used which would make management of these emails much easier. See the V360 Admin -> Notification Template feature for more information. The number of scripts used to send these emails could be dramatically reduced.

- Review environment specific code and add references to global variables in INCLUDES_CUSTOM_GLOBALS that represent the environment (DEV, STAGE, PROD, etc.) so that code can remain consistent from env to env.

- Review/Edit all TODO sections as shown below:

```
Asheville\Scripts\Event\WTUA;Planning!~!~!~.js
      Line 168: // TODO: review with Diane per email 10/26
      Line 200: // TODO: review with Diane per email 10/26
 Asheville\Scripts\Event\IRSA;~!~!~!~.js
      Line 1: // TODO:  repeated conditionals, consider refactoring
      Line 183: // TODO:  repeated conditionals, consider refactoring
      Line 333: // TODO:  repeated conditionals, consider refactoring
 Asheville\INCLUDES_CUSTOM\updateCommercialBuildingFees.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\updateMiscFees.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\updateSiteFees.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\attachSubCloseOutProcess.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\closeOutDocTaskActive.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\deactivateSubTasks.js
      Line 1: // TODO:  function not referenced in master script migration
 Asheville\INCLUDES_CUSTOM\doesTaskHaveSubProcess.js
      Line 1: // TODO:  function not referenced in master script migration
```

Script Repository

The master script 3.0 deployment is temporarily located in an Assembla repository

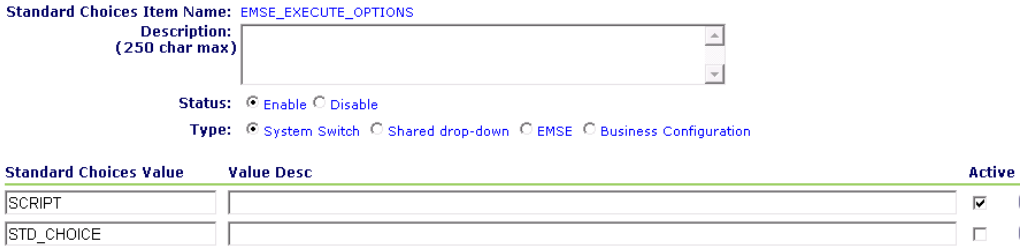
Repository Site	Assembla.com
Repository URL	https://subversion.assembla.com/svn/asheville/trunk/DEV

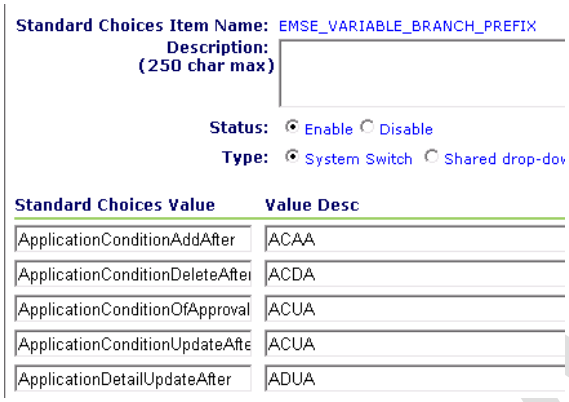
To connect to the repository, the “EMSEToolConfig” standard choice will need to be configured as per the Accela documentation. Once added, the “EMSETool” portlet can be added to an administrator console.

Standard Choices

The following Standard Choices need to be configured. These are documented in the Master Script distribution in the documentation folder.

- EMSE_EXECUTE_OPTIONS determines if Scripts or Standard Choices, or both should be executed by the master script.
- EMSE_VARIABLE_BRANCH_PREFIX determines the naming convention used to determine if a script is associated to a particular event.





Event to Script Mapping After Upgrade

TBD