Differences in scripting between Master Scripts 3.0 and earlier versions

Description When moving to scripting 3.0, it's important to recognize a few differences from the earlier scripting frameworks. Earlier Master Scripts relied on several shortcuts, primarily in an attempt to make the code easier to read in the compressed space in Standard Choices. The 3.0 scripting framework does away with these shortcuts and uses pure JavaScript instead. A summary of the changes are listed here:

- No more line numbers. Since the 3.0 scripts are stored in the scripts area instead of standard choices, there is no longer a need to indicate the order of execution with a line number. Instead the JavaScript code is evaluated top-to-bottom as always.
- "Carets" as if/then/else: Probably the most prominent difference is the elimination of the caret ^ to separate if/then/else clauses. JavaScript has a different purpose for the caret, so this notation is no longer allowed. Instead:

```
wfStatus == "Issued" ^ editAppStatus("Issued",""); ^ editAppStatus("Not Issued","");
```

becomes

```
if (wfStatus == "Issued") {
        editAppStatus("Issued", "");
} else {
        editAppStatus("Not Issued", "");
}
```

- "branch" function is removed. The branch function was used to shift code execution to a different Standard Choice, much like a subroutine in other programming languages. Since the 3.0 framework no longer stores code in Standard Choices, this function is deprecated. If you are converting to 3.0, consider the following options for your branch code:
  - If the subroutine (code that is "branched" to) is only called from one Standard Choice, consider consolidating it into a single script.
  - If the subroutine is called from several Standard Choices, or may be re-used in the future -- consider creating a function with this code, then calling the function from your new 3.0 script. For example: `branch("EMSE:AddBuildingFees");` would become `addBuildingFees();`
- Curly braces as a shortcut to ASI fields: Earlier master scripts could use curly braces { } surrounding an ASI / TSI / Parcel Attribute label to substitute the value of that field. Since pure JavaScript has a much different purpose for these braces, this feature is deprecated. Instead, the 3.0 Master Script populates the AInfo array with the same information. For example:
  - `{Square Footage} > 10`
    becomes
    `AInfo["Square Footage"] > 10`
- disableTokens variable is removed: As a workaround to the problems caused by the previous ASI notation, the disableTokens boolean global variable could be used to temporarily disable the curly-brace-ASI-substitution. With 3.0 this is no longer necessary.
- "Disabling" code. When code was stored in Standard Choices, you could individual lines from executing by un-checking the "Active" checkbox next to the line. There is no such capability with scripts, although the standard JavaScript notation for commenting code works fine.
- `// here is a single line comment`

```
/*
Here is an
extended multi-line
comment
*/
```

Version Number All Versions

Products Accela Automation

File