**Coding Standards**

1. Functions and variables should be written in camelcase (first letter of function/variable name should be lowercase, and each word of the name should be capitalized after that)
   1. **Good Examples**
      * function update**P**lan**R**eview**F**ee()
      * var current**P**lan**R**eview**F**ee = 0;
      * function get**F**ee**Q**ty()
      * var fee**Q**ty = 1;
   2. **Bad Examples**
      * function UpdatePlanReviewFee()
      * var CURRENTPLANREVIEWFEE = 0;
      * function getfeeqty()
      * var FEEQty = 1;
2. Functions and variables should be named descriptively, providing a clear understanding of what the function is supposed to accomplish, or what data the variable is supposed to contain.
   1. **Good Examples**
      * function getAllFeesForRecord()
      * var allFeesOnRecord = getAllFeesForRecord();
   2. **Bad Examples**
      * function fees()
      * var fees = fees();
3. Variables should ideally be initialized and set in the same line.
   1. **Good Example**
      * var allFeesOnRecord = getAllFeesForRecord();
   2. **Bad Example**
      * var allFeesOnRecord;
      * allFeesOnRecord = getAllFeesForRecord();
4. Comments should be added before every function with the following information
   1. A short explanation of the function’s purpose
   2. Which events call that particular function (if applicable)
   3. Which record types it affects
   4. A short explanation of each parameter (if any) being passed to the function
   5. What data should be returned (if any)
   6. The issue # if the code change is associated with the master issues list
   7. The function’s author
   8. The last time the function was modified

**Example**/\* Purpose:  
 \* Assess revised fees from ASIT revisions table. If valuation   
 \* is below 100k, remove all fees.

\*  
 \* Event(s):   
 \* ASIUA  
 \*  
 \* Record Type(s):   
 \* Building/Commercial/Structural Permits/NA,  
 \* Building/Residential/Structural Permits/NA  
 \*  
 \* Parameter(s):   
 \* subGroup - The fee sub group.   
 \*  
 \* Returned data:  
 \* NONE  
 \*  
 \* Developed by:   
 \* Andrew Attebery  
 \*  
 \* Modified on:   
 \* 02/11/2017, 03/18/2017, 05/01/2017, 08/22/2017  
 \*/

1. Code that’s been “commented out” should have a date so that it can be deleted thirty days after being deactivated.
   1. **Example**  
      // Commented out on 9/18/17
2. All code should be TAB indented after each IF/ELSE/FOR/LOOP/TRY/CATCH/FUNCTION statement.
3. All IF statements should have an opening and closing bracket.
   1. **Example (For 6 and 7)**  
      if (blah != “blah!”){  
       blah = blah2;  
      } else {  
       blah = blah3;  
      }
4. Comments should be added to any line in the code that would require explanation if you had to come back and rewrite it six months later. Anything that wouldn’t be immediately obvious and understandable at first read should be spelled out. The comment should be added before the line in question. Most **good, well-documented** code has about one line of comments for every five lines of code.
   1. **Example**  
      // If b is not equal to c, then i is equal to t; otherwise i is equal to a.   
      var i = (b != c) ? t : a;
5. Hard coding numbers into a function should be avoided at all cost. Try to use a fee item instead. But, since we live in the real world, if a number has to be hard coded into a function, be very sure to leave a comment explaining what the number represents and the business use for that particular number. Also, place that number as its own variable so it can be changed in just one place in the code later, instead of several.
   1. **Good Example**var baseElectricalFee = 151.04;  
      var additionalElectricalFee = 25.60;var feeTotal = (baseElectricalFee + (additionalElectricalFee \* numberOfUnits));
   2. **Bad Example**var feeTotal = (151.04 + (25.60 \* numberOfUnits));
6. Functions should be wrapped in try/catch brackets to prevent an entire event from failing to execute because of a single bad function.
   1. **Example**function thisIsMyExampleFunction(){  
       try{  
       var b = doSomeStuff();  
       var c = doMoreStuff();  
       } catch (err) {  
       logDebug(“Error in function: thisIsMyExampleFunction()”);  
       logDebug(“Error was: “ + err.message);  
       logDebug(“Error line: “ + err.lineNumber);  
       }  
      }