Using <<CALLPOINT>> as the Enable Column

You can use the Enable Column/Enable value settings in the Barista Form Designer to enable or disable one control on a form based on a value or set of values in a different control. For example, if a form contains a *Start Date* and *End Date*, you might want to disable the *End Date* unless/until the user specifies a *Start Date.* You do that by selecting the *End Date* control in the Form Designer, then setting the *Enable Column* to reference the *Start Date,* and the *Enable Value* to an exclamation point with no value after it. The exclamation point means "not," so an exclamation point alone would mean to enable the *End Date* column if the *Start Date* column isn't null/empty.

In some cases, the scenario for enabling/disabling columns might be more complex. You can use callpoint!.setColumnEnabled("<column_name>",status) in callpoint code any time you need to toggle the enabled status of a control. In addition, you can use the <<CALLPOINT>> placeholder in place of the Enable Column property to enable/disable several columns at once. The easiest way to understand using <<CALLPOINT>> is with an example from AddonSoftware.

In Addon's Purchase Order form, you begin each detail line by selecting a *Line Code*. Each *Line Code* is associated with a *Line **Type*** (S=standard item, M=message, N=Non-stock, etc.) that controls the enabled status of other fields on the row. For example, if entering a code that's associated with a Message type line, the message field should be enabled, but not the item, quantity, or price fields.

Since this structure makes a simple relationship between the *Line Code* and the enabled/disabled status of other fields impossible, Addon uses the <<CALLPOINT>> placeholder instead. The <<CALLPOINT>> placeholder signals Barista that callpoint code will determine the enable values, rather than simply looking at the value in some other control on the form.

For any field that should be enabled/disabled based on the *Line **Type***, the *Enable Column* property is set to <<CALLPOINT>>, and the *Enable Value* to the desired value(s). The After Column Validation (AVAL) callpoint for the *Line Code* reads the code file to retrieve the corresponding *Line **Type***, and sets callpoint!.setStatus("ENABLE:"+<line_type>). Barista processes the setStatus() method by searching through all of the columns on the form looking for the <<CALLPOINT>> placeholder, and enabling or disabling those columns if their *Enable Value* matches the <line_type>.

Returning to the previous example, if the *Line Code* entered corresponds to a Message *Line **Type*** of "M," then the callpoint!.setStatus("ENABLE:"+<line_type>) will see to it that any column with *Enable Column* = <<CALLPOINT>> and an *Enable Value* of "M" will be enabled as soon as the *Line Code* is entered/validated. Any column with *Enable Column* = <<CALLPOINT>> and an *Enable Value* not equal to "M" will be disabled.