

```
Sub CreateAndConfigureSkid()  
    ' This code creates a new copy of the skid assembly template in our new folder structure  
    ' It then updates the skid geometry based on values we pass to the assembly  
    Dim strNewSkidFilename As String  
    ' This string represents the new name of our unique, copied skid assembly file  
    strNewSkidFilename = PROJECT_PATH & PROJECT_ID & "\Skid Assy\Skid Assy - " & PROJECT_ID & ".iam"  
  
    ' We first check to make sure the skid assembly file has not been previously created  
    If System.IO.File.Exists(strNewSkidFilename) = False Then  
        ' This here is a sample of how to make a variable that represents an assembly document  
        ' It uses the Inventor API, which you can use freely (for the most part) throughout iLogic rules  
        Dim subAssyl As AssemblyDocument  
        ' This code uses the "CopyComponents" subroutine (see above) to copy the skid assembly, and all its children  
        ' This will not work if you don't want some of the parts in the assembly to have unique copies  
        ' The function also changes the references in the skid assembly to point to the newly created part files  
        CopyComponents(TEMPLATE_PATH & "Skid Assy\", "Skid Assy.iam", "Skid Assy")  
        ' This statement tells subAssyl to represent the template file, and opens it up in the Inventor interface  
        subAssyl = ThisApplication.Documents.Open(strNewSkidFilename, True)  
        ' This statement changes the occurrence names of the existing skid components in the model browser  
        ' This will allow the rule that passes parameters in our skid sub-assembly to still work  
        subAssyl.ComponentDefinition.Occurrences(1).Name = "Skid-1:1"  
        subAssyl.ComponentDefinition.Occurrences(2).Name = "Skid-1:2"  
        subAssyl.Save  
        subAssyl.Close  
        ' This can be taken from an iLogic snippet, and is used to insert components into assemblies  
        ' This code inserts our newly created skid assembly into our master tank assembly file  
        Dim componentB = Components.Add("Skid Assy:1", strNewSkidFilename, position := Nothing, grounded := True, visible := True, appearance := Nothing)  
        ' Change our flange radius if the TANK_OD is 30" or less  
        If TANK_OD <= 30 in Then SKID_FLG_RAD = .1 in  
        ' These statements pass parameters from our master assembly file into the skid assembly file  
        Parameter("Skid Assy:1", "PROJECT_ID") = PROJECT_ID  
        Parameter("Skid Assy:1", "PROJECT_PATH") = PROJECT_PATH  
        Parameter("Skid Assy:1", "TANK_OD") = TANK_OD  
        Parameter("Skid Assy:1", "TANK_L") = TANK_L  
        Parameter("Skid Assy:1", "SHELL_W_1") = SHELL_W_1  
        Parameter("Skid Assy:1", "SHELL_W_2") = SHELL_W_2  
        Parameter("Skid Assy:1", "SHELL_Q_1") = SHELL_Q_1  
        Parameter("Skid Assy:1", "SHELL_Q_2") = SHELL_Q_2  
        Parameter("Skid Assy:1", "SKID_FW") = SKID_FW  
        Parameter("Skid Assy:1", "SKID_FH") = SKID_FH  
        Parameter("Skid Assy:1", "SKID_FL_THK") = SKID_FL_THK  
        Parameter("Skid Assy:1", "SKID_FLG_RAD") = SKID_FLG_RAD  
        Parameter("Skid Assy:1", "SKID_WEB_THK") = SKID_WEB_THK  
        Parameter("Skid Assy:1", "SKID_BEND_L") = SKID_BEND_L  
        Parameter("Skid Assy:1", "SKID_ROD_D") = SKID_ROD_D  
        Parameter("Skid Assy:1", "DISH_DEPTH") = DISH_DEPTH  
        ' Once all the parameters are updated in the skid assembly file, we want to run its creation rule  
        ' This will allow the skid assembly to update all its own parts and components itself  
        iLogicVb.RunRule("Skid Assy:1", "Create Skid")  
    End If  
End Sub
```