```
# ----------------------------------------------------------------------------
# This software is in the public domain, furnished "as is", without technical
# support, and with no warranty, express or implied, as to its usefulness for
# any purpose.
#
# ForecastBuilderConfig - Version 10.7
#
# Author: ForecastBuilder Tech Working Group - nws.forecastbuilder@noaa.gov
#
# Please reference documentation within this file, as well as the ForecastBuilder
# documentation and configuration spreadsheet
#
# ----------------------------------------------------------------------------

import SmartScript


class ForecastBuilderConfig(SmartScript.SmartScript):
    def __init__(self, dbss):
        SmartScript.SmartScript.__init__(self, dbss)
        self._dbss = dbss

    def Local_Configuration(self, varDict, gridDef, var="varDict"):
        ###########################################################################
        # This is the local configuration section for ForecastBuilder. It has two
        # sections, varDict and gridDef.  Using this style of configuration allows for less
        # frequent config file updates and hence less need for you to merge this file with
        # updates in future builds. Also, it reduces code clutter in the main
        # ForecastBuilder procedure & utility.
        #
        # There are two sections: varDict and gridDef.
        #
        # Think of varDict as general FB settings such as on/off switches (True/False)
        # and lists you can add to.  Meanwhile, gridDef is for defining grid-specific
        # settings, such as changing the default grid length for an element.
        #
        # The configuration of ForecastBuilder is run in the following way:
        # 1) FB-wide default values of varDict are defined in the main
        #    ForecastBuilderNationalConfig file.
        # 2) Each region has an area where the defaults can be overridden called
        #    ForecastBuilderRegionalConfig
        # 3) This present file, ForecastBuilderConfig is called.  This is done first for
        #    varDict and then for gridDef
        #
        # There should be minimal configurations in this file, as the procedure and
        # regional configurations should handle most of the configuration. Consider
        # interoffice consistency implications with any additions made here.
        ###########################################################################
        #
        if var == "varDict":
            #
            # Put varDict local configuration in here. These options can be found in the
            # ForecastBuilderNationalConfig and RegionalConfig files.
            #

            ## Common examples of locally-configured items.
            varDict["Default PType Method"] = "NBM SnowLevel"
            # varDict["Other Possible PType Methods"] = [
            #     "SnowLevel & ProbIcePresent",
            #     "NBM",
            # ]
            # varDict["Hide Ice or Sleet in Step 4"] = False
```

```
63            # varDict["Interpolate After Populating"] = True
64            # varDict["Use Local Time Scale Periods"] = True
65            # varDict["Stratiform or Showery Precipitation?"] = "Shower"
66            # varDict["Include ESTF"] = True
67            # # Set to True if your office populates these elements in the day 4-7 period.
68            # varDict["Have QPF and Accum grids in Extended"] = True
69            # # If your office would like to populate Aviation grids as part of the Foundation
70            # # Grids Step in ForecastBuilder
71            # varDict["Include Aviation"] = True
72            # # If your office would like to create FireWx grids as part of ForecastBuilder
73            varDict["Include Fire"] = True
74            # varDict["Include GHWO"] = True
75            # varDict["EditRetention_automaticColor"] = "DarkGreen"
76            # varDict["EditRetention_editedColor"] = "Yellow"
77            # varDict["EditRetention_manualColor"] = "Red"
78            # varDict["CRON_thru_WX"] = True
79 ## Fire weather entries
   ##########################################################################################################
   ###################
80 #             varDict["Possible-Fire Wx: Calculate from GFE/D2D
   grids?"].extend(["CWR","TransWind","Wind20ft","MixHgt"])
81            varDict["Possible-Fire Wx: Calculate from GFE/D2D
   grids?"].extend(["CWR","Wind20ft","MixHgt","TransWind"])
82
83            varDict["Fire Button List"].append(["CWR",["self._FBUtility","callSmartToolFB",
   ["CWR","CWRfromQPFandPoP","var|gridDict","var|varDict"]]])
84            varDict["Fire Button List"].append(["TransWind",["self._FBUtility","callSmartToolFB",
   ["TransWind","copyFromNBM_PoP6","var|gridDict","var|varDict"]]])
85            varDict["Fire Button List"].append(["Wind20ft",["self._FBUtility","callSmartToolFB",
   ["Wind20ft","FB_Wind20ft","var|gridDict","var|varDict"]]])
86            varDict["Fire Button List"].append(["MixHgt",["self._FBUtility","callSmartToolFB",
   ["MixHgt","copyFromNBM_PoP6","var|gridDict","var|varDict"]]])
87            varDict["Additional Fire Weather Parms to Possibly Populate in Step 2"] =
   ["MixHgt","TransWind"]
88 #             varDict["Fire Wx: Update from NAM or GFS\n(overrides GFE choice)?"] = "TransWind"
89 #             varDict["Additional Grids to Load for Fire Weather"]=["FCST","MixHgt","SFC"]
90            varDict["Additional Grids to Load for Fire Weather"]=["MixHgt","TransWind"]
91
92
93 ### Freezing Level entries
   ##########################################################################################################
   ############
94            varDict["Parms to Load in Analyze/Adjust"].extend(["FzLevel"])
95            varDict["Grids to initialize"].extend(["FzLevel"])
96            varDict["Additional Buttons in Analyze/Adjust Step"].append(["Populate FzLevel",
   ["self._FBUtility","callSmartToolFB",["FzLevel","FzLevelCopy","var|gridDict","var|
   varDict"],]])
97 ##########################################################################################################
   ##########################################
98            # )
99            # Set to True to utilize NBM PPI06 grids for 6 hourly PoP
100           varDict["6 Hourly PoP from NBM PPI06"] = True
101           varDict["Include Marine"] = False
102
103
104 #             pass
105
106       elif var == "gridDef":
107           #
108           # Put gridDefs local configuration in here. Documentation for gridDef settings
109           # can be found in the ForecastBuilderNatioanlConfig file.
110           #
```

```
111
112             # Example of smoothing over an edit area representing a lake
113             # for grids in ["T", "Td", "MaxT", "MinT"]:
114             #     gridDef[grids, "smoothFactor", "EditArea"] = "NBMLakes"
115             #     gridDef[grids, "smoothFactor", "Factor"] = "10"
116
117             # Example of setting the snow preview feature in Step 2 to use the local
118             # SnowAmt grid:
119             # gridDef["SnowAmtPre", "gridName"] = "SnowAmt"
120
121             # Example of changing the grid lengths for several elements, and changing the
122             # sample method for Wind.
123             # for grids in [
124             #     "T",
125             #     "Td",
126             #     "RH",
127             #     "PotRain",
128             #     "PotSnow",
129             #     "PotSleet",
130             #     "PotFreezingRain",
131             #     "SnowRatio",
132             #     "SnowLevel",
133             # ]:
134             #     gridDef[grids, "gridDefinition"] = [
135             #         ["Extended", "Extended", 0, 3 * 3600, 3 * 3600]
136             #     ]
137             #     gridDef[grids, "mode"] = "Max"
138             #     if grids == "Wind":
139             #         gridDef[grids, "mode"] = "AverageWindSpeed"
140
141 ## Fire weather entries
142
143             gridDef["CWR","gridDefinition"] = [[0,12*3600,12*3600]]
144             gridDef["CWR","maxTime"]="Day 3"
145
146             gridDef["TransWind","gridDefinition"] = [[0,3*3600,3*3600]]
147             gridDef["TransWind","maxTime"]="Day 3"
148
149             gridDef["Wind20ft", "gridDefinition"] = [[0, 3600, 3600]]
150             gridDef["Wind20ft","maxTime"]="Day 3"
151
152
153             gridDef["MixHgt","gridDefinition"] = [[0,6*3600,6*3600]]
154             gridDef["MixHgt","maxTime"]="Day 3"
155
156             #Sky grids using NBM 4.2 experimental as primary, with operational NBM as backup.
    This should limit the 57% sky cover issues.
157             # gridDef["Sky", "primaryGuidance"] = "NBMEXP"
158             # gridDef["Sky", "secondaryGuidance"] = "NBM"
159             gridDef["FzLevel", "gridName"] = "FzLevel"
160             gridDef["FzLevel","gridDefinition"] = [[0,6*3600,6*3600]]
161             gridDef["FzLevel", "primaryGuidance"] = "CONSAll"
162             gridDef["FzLevel", "maxTime"] = 192
163             gridDef["FzLevel", "minTime"] = 0
164
165
166             gridDef["SnowRatio","primaryGuidance"] = "WPCGuide"
167             gridDef["SnowRatio","secondaryGuidance"] = "NBM"
168 #            pass
169         else:
170             print(f"Need a definition for var: {var}")
171
```