

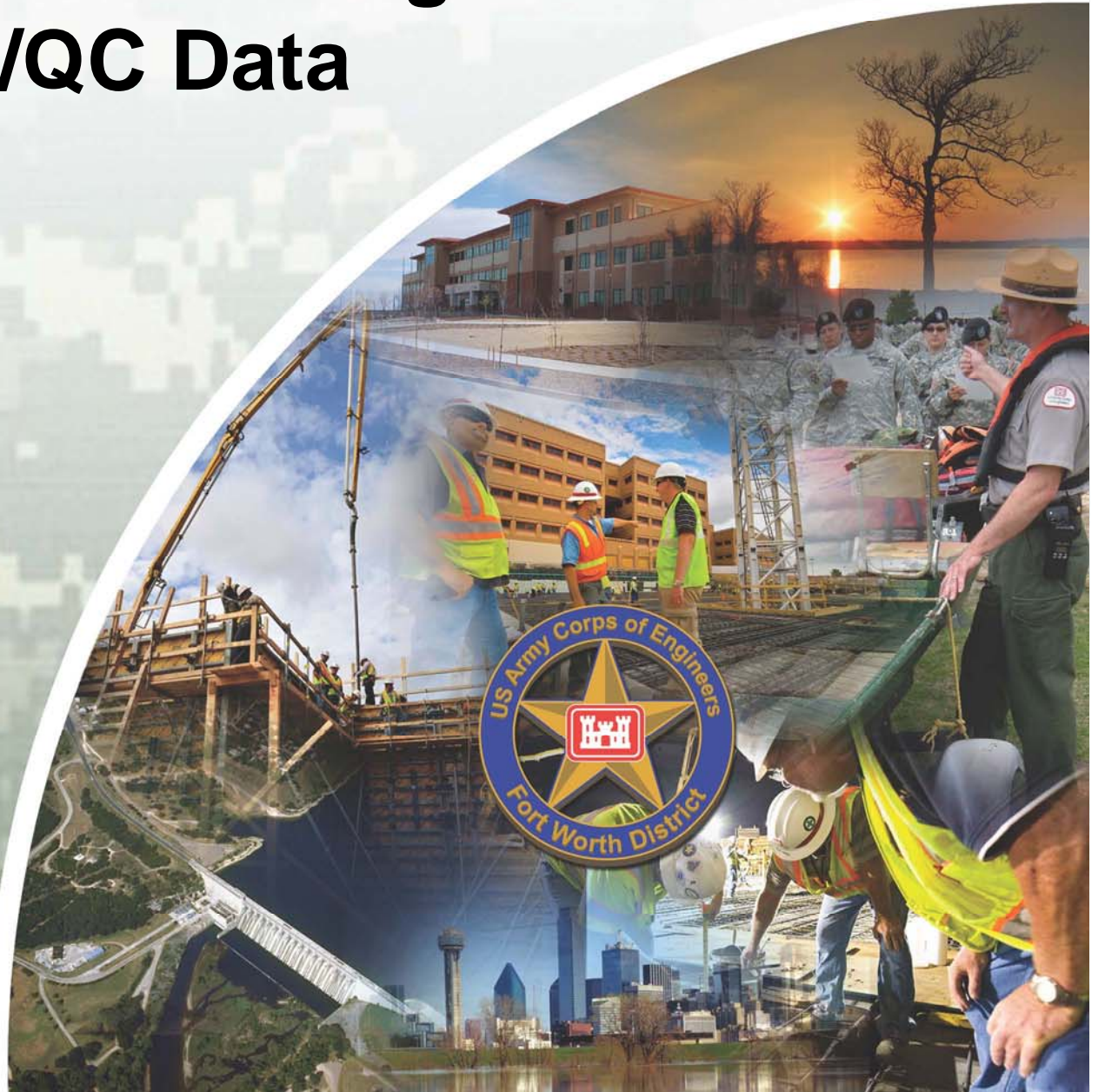
Python – It's Like Having an Extra Person to QA/QC Data

Jennifer M. Holland, GISP

28 June 2016



US Army Corps of Engineers
BUILDING STRONG



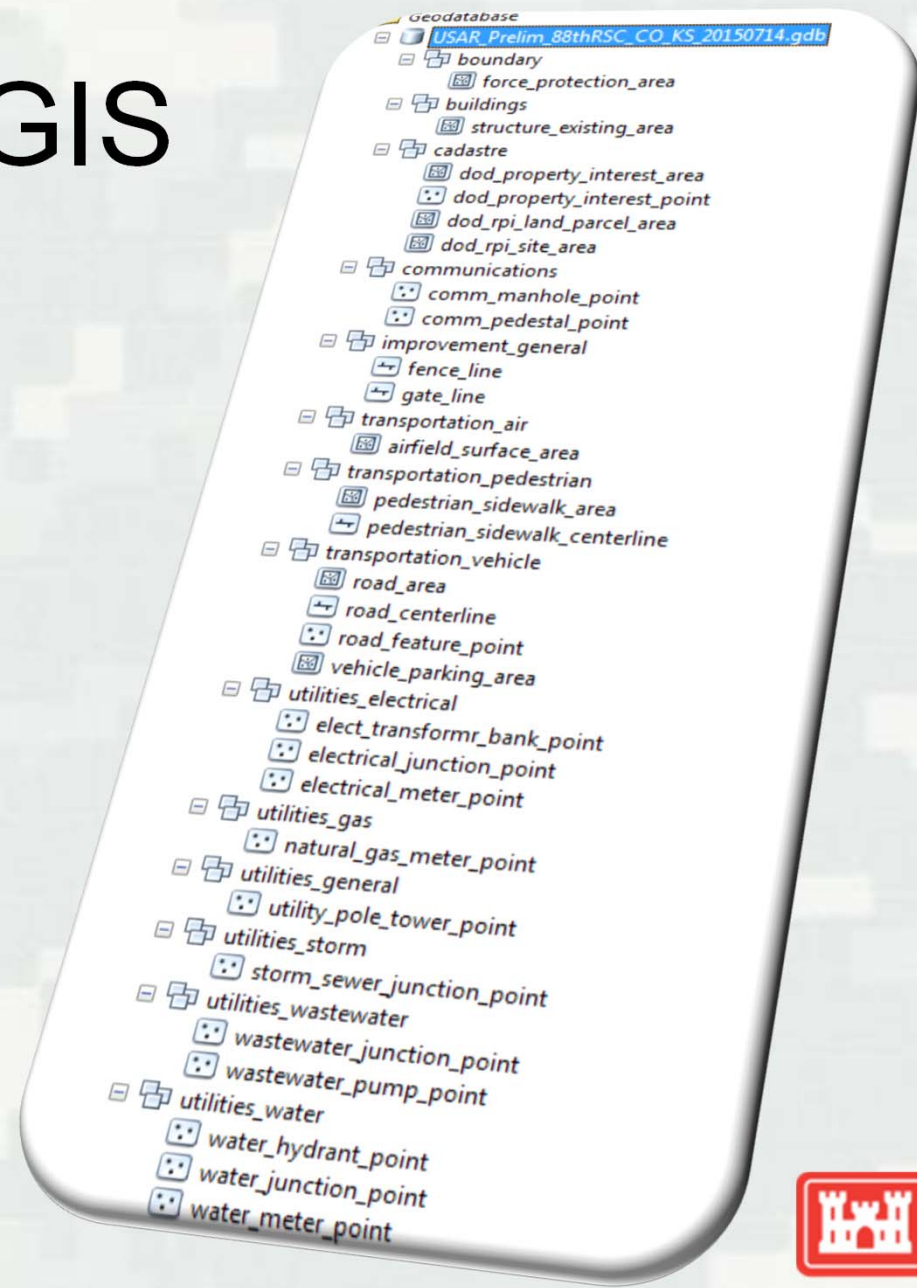
Project Details

- U.S. Army Reserve (USAR)
 - ▶ Army requirement to have complete data
 - Not met
 - ▶ Need completed data to manage their facilities
- Data collection project
 - ▶ 473 sites
 - On site data collection
 - ▷ GIS (map grade GPS)
 - ▷ Space and Facility Utilization



GIS

- SDSFIE version 2.6 data schema
 - ▶ 14 feature datasets
 - 28 feature classes



Fort Bliss











QA/QC Process

- Requirements
 - ▶ Must QA/QC at least 20% of the GIS data
 - Spatial
 - Attributes
- Interested in automating the process as much as possible
 - ▶ Python

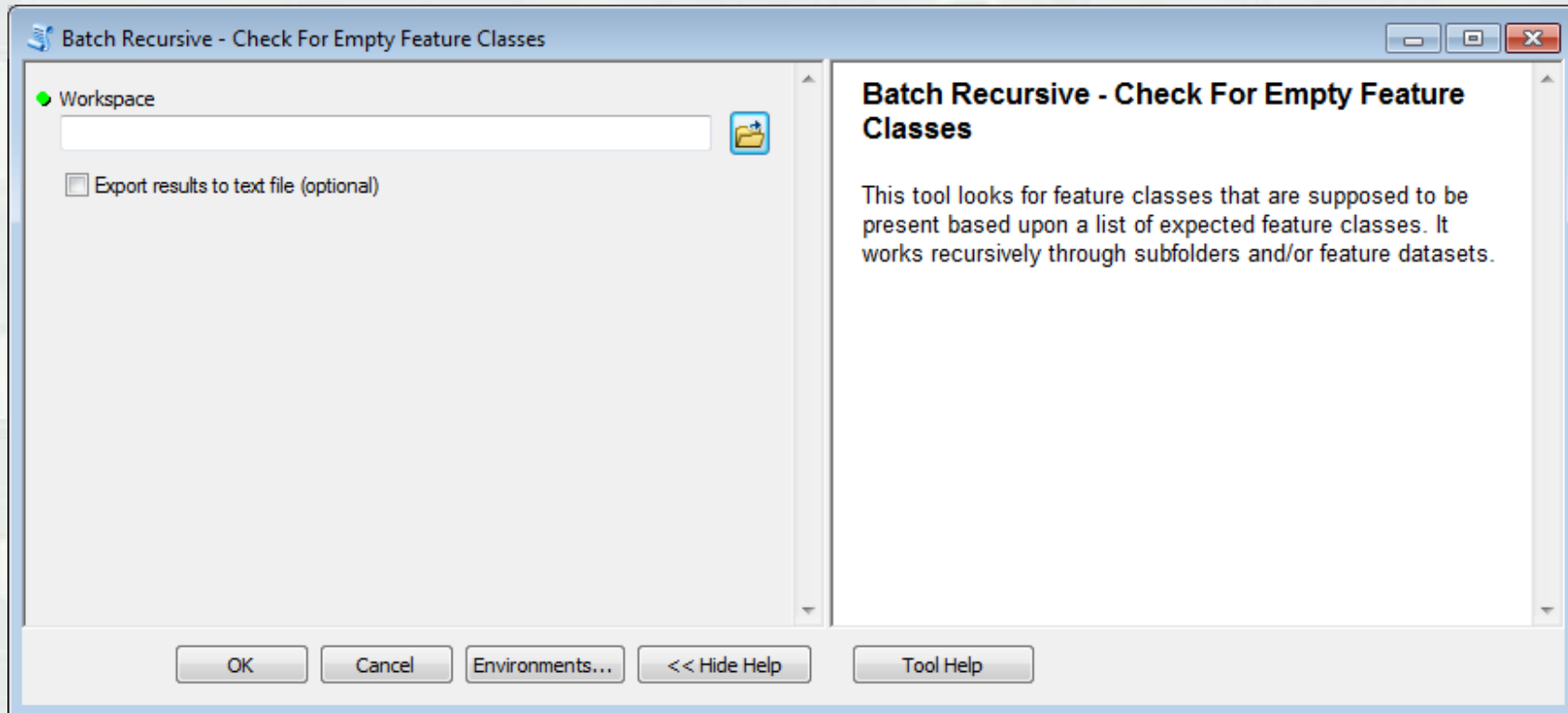


[-]  USAR

-  Batch Recursive - Check For Empty Feature Classes
-  Batch Recursive - Check For Empty Fields
-  Batch Recursive - Check For Geometry Errors
-  Batch Recursive - Check For Missing Layers
-  Batch Recursive - Check Spatial Data
-  Batch Recursive - Check Spatial Location
-  Batch Recursive - Move Feature Classes to One Feature Dataset
-  USAR_Tool



Check for Empty Feature Classes



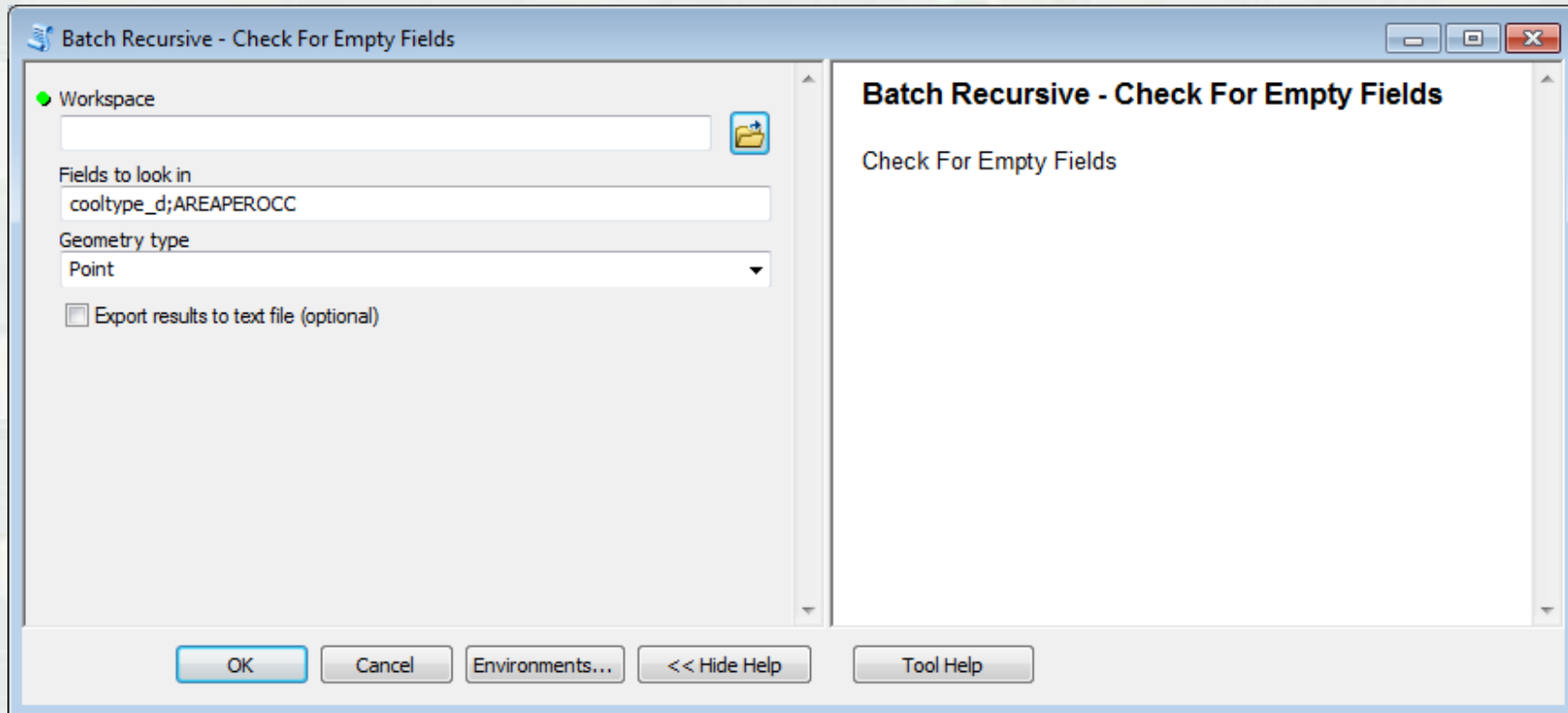
```

Python 2.7.8: BATCHR-1.PY - \\swf-netapp1\Military\US_ARM-1\Spatial\PYTHON-1\USAR_P-1\BATCHR-1.PY
File Edit Format Run Options Windows Help
import arcpy, sys, os, string, time
# Batch Recursive - Check for Empty Feature Classes
# - Jennifer Holland
# - U.S. Army Corps of Engineers
# - jennifer.m.holland@usace.army.mil
# - 31 May 2016
#
# - Version 1.0 - 5 Feb 2014 - Jennifer Holland - Original code to loop through feature classes to check to see if one is mis
# - Version 2.0 - 31 May 2016 - Jennifer Holland - Updated code to use arcpy.da.walk
#
# - Acknowledgments - Thanks to numerous authors on ESRI forums and other internet forums for coding help and examples
# - Created and tested on:
# - Windows 7 (64 bit) operating system
# - ESRI ArcGIS 10.3.1.4959
# - Python 2.7.8
# Usage string
usage = "<input_workspace> {export_file}"
# Set wildcard for data filter. Default is * (all)
wildcard = "*"
# Set type for data filter. Default is ALL
type = "ALL"
def SearchForFeatureClasses(inputWorkspace):
    walk = arcpy.da.Walk(inputWorkspace, datatype="FeatureClass")
    for dirpath, dirnames, filenames in walk:
        for filename in filenames:
            CheckFeatureClass(os.path.join(dirpath, filename))
def CheckFeatureClass(myFeatureClass):
    matchCount = int(arcpy.GetCount_management(myFeatureClass).getOutput(0))
    if matchCount == 0:
        arcpy.AddWarning(myFeatureClass + " - EMPTY")
        if outputTextFile == "true":
            OutFileText.write(myFeatureClass + " - EMPTY" + "\n")
    else:
        arcpy.AddMessage(myFeatureClass + " - OK")
        if outputTextFile == "true":
            OutFileText.write(myFeatureClass + " - OK" + "\n")
try:
    # Check if all required parameters have been provided.
    if len(sys.argv) < 2:
        raise Exception, "Invalid number of parameters provided. \n" + "Usage: " + usage
    # Set the input workspace environment
    inputWorkspace = sys.argv[1]
    # Check if <input_workspace> exists
    if not arcpy.Exists(inputWorkspace):
        arcpy.AddError("Workspace " + inputWorkspace + " does not exist")
        raise Exception, "<input_workspace> does not exist."
    else:
        arcpy.AddMessage("Input workspace: " + inputWorkspace)
        desc = arcpy.Describe(inputWorkspace)
        workspace_type = desc.workspaceType
        if workspace_type == "LocalDatabase":
            path = os.path.abspath(os.path.join(sys.argv[1], os.pardir))
        else:
            path = sys.argv[1]
    # Output results to text file
    outputTextFile = sys.argv[2]
    if outputTextFile == "true":
        workspaceName = os.path.splitext(os.path.basename(inputWorkspace))[0]
        # Create timestamp for output text file name
        localtime = time.localtime()
        timeString = time.strftime("%Y%m%d%H%M%S", localtime)
        pathText = path + "\\\" + "CheckEmptyFeatures_" + workspaceName + "_" + timeString + ".txt"
        OutFileText = open(pathText, "w")
        SearchForFeatureClasses(inputWorkspace)
except Exception, ErrDesc:
    arcpy.AddError(ErrDesc)
    print ErrDesc

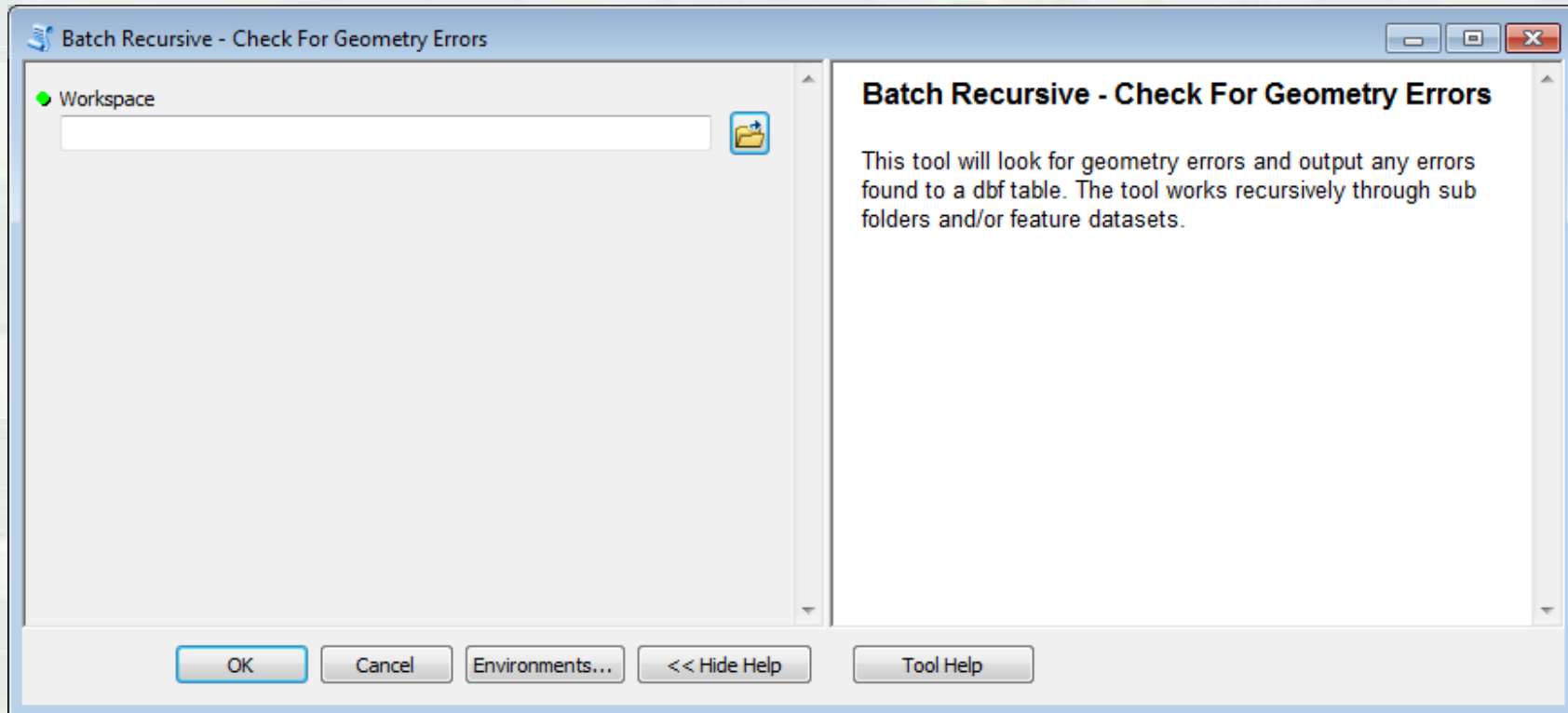
```



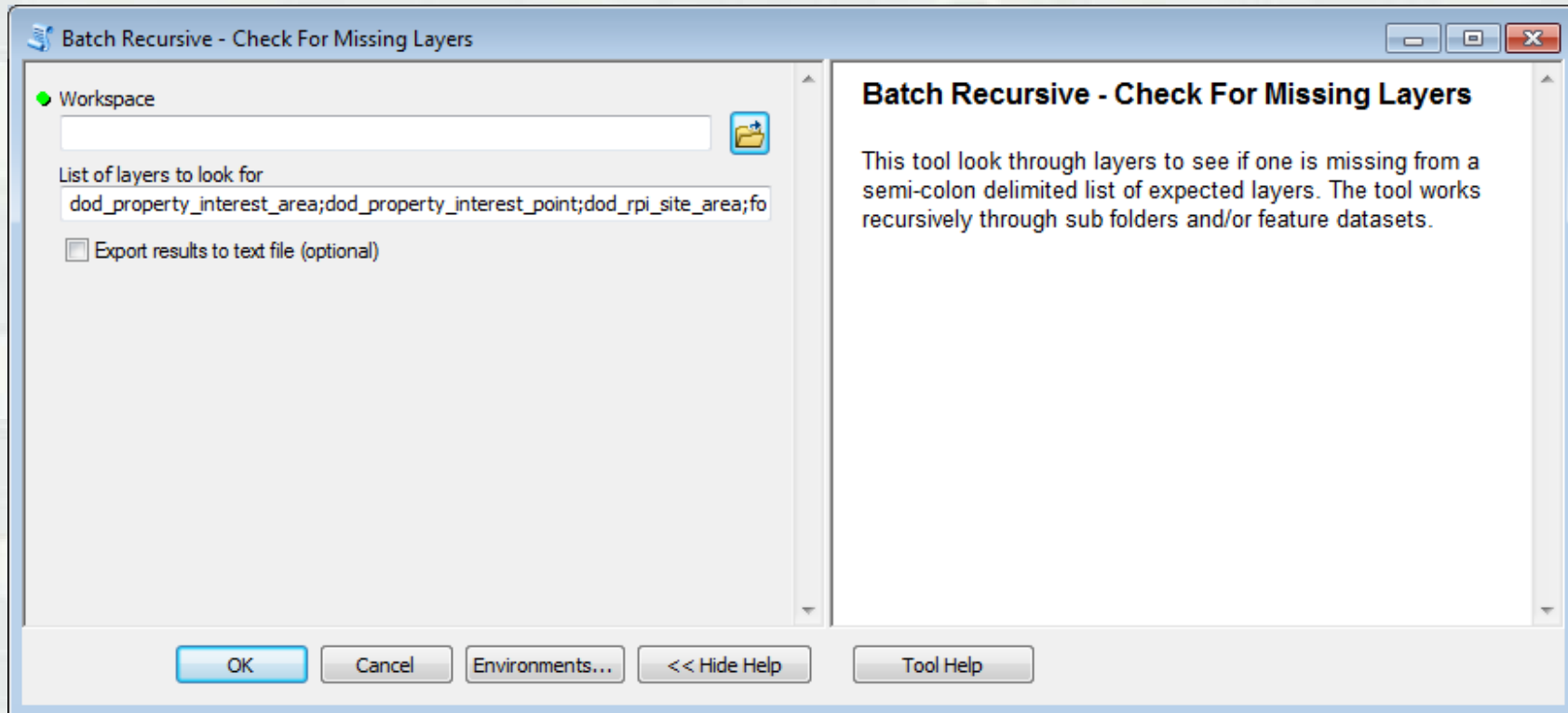
Check for Empty Fields



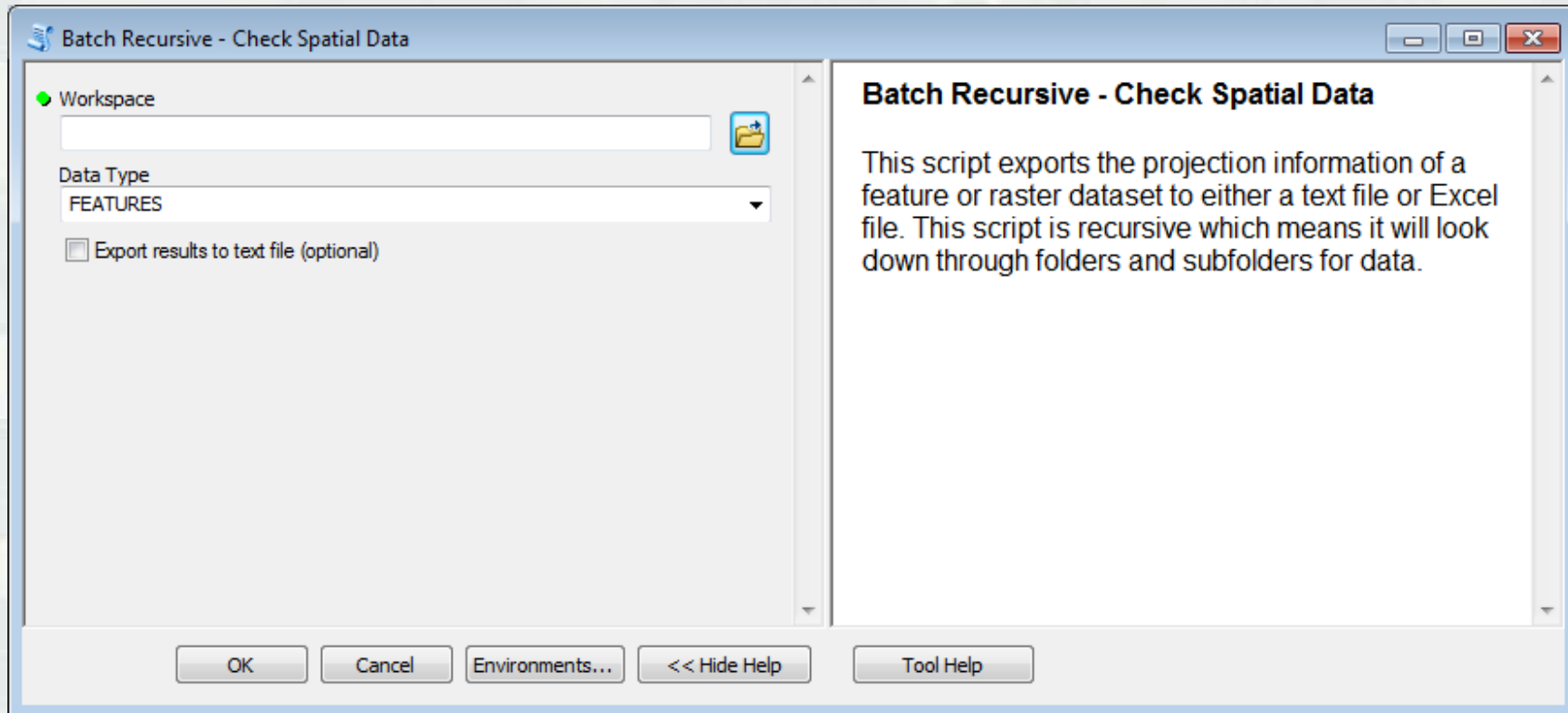
Check for Geometry Errors



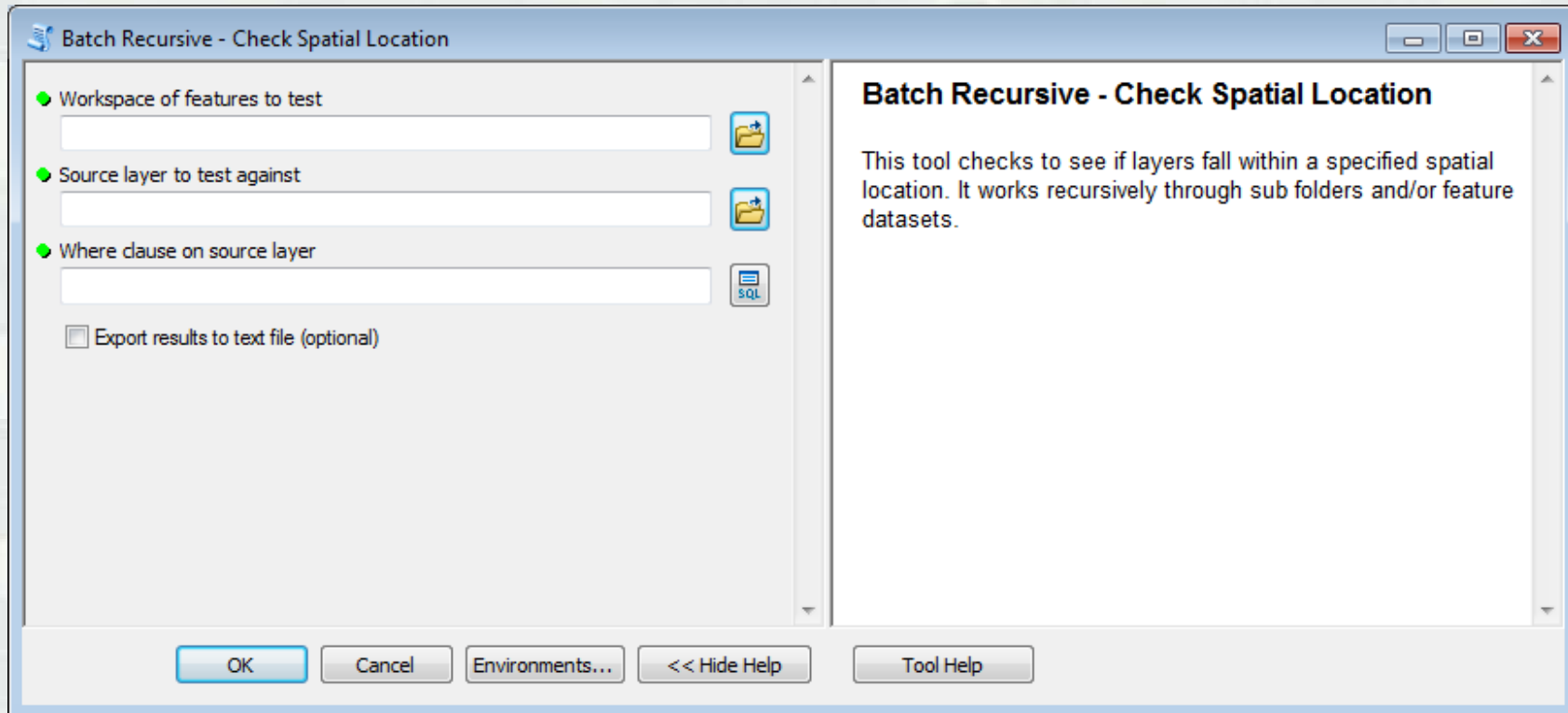
Check for Missing Layers



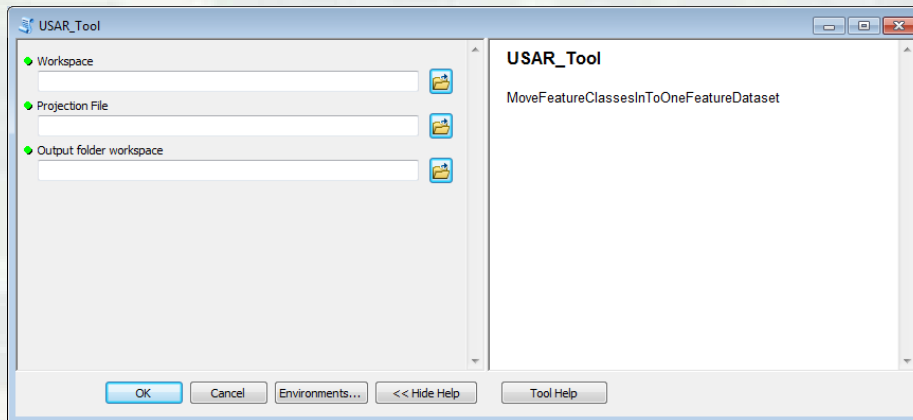
Check Spatial Data



Check Spatial Location



USAR Tool



- Create FGDB
- Create feature dataset
- Create topology
- Add feature classes
- Load topology rules
- Validate topology



Totals

- ~60,000 features
 - ▶ Manual review
 - 20% review
 - 333 hours
 - 8.5 weeks
 - ▶ Automated review
 - 90% review
 - 400 hours
 - 10 weeks



Lessons Learned

- Data Reviewer

- ▶ Has *most* of the functionality needed out of the box

- ▶ Has some additional functionality that was not needed for this project

- USAR Python Tools

- ▶ Topology

- ▶ Simple

- ▶ Can be targeted to *exactly* what is needed with customization



Conclusion

- Automating the QA/QC process saved
 - ▶ Time
 - ▶ Money
 - ▶ Data Quality (most important)
 - More data reviewed (+/- 90%)
 - More time spent visually inspecting data
- Great alternative to Data Reviewer
- Tools available for other projects



▶ “Toolified”



Questions

Python!

It's like having an extra person to QA/QC data

Jennifer M. Holland, GISP
Geographer
U.S. Army Corps of Engineers
Fort Worth District
jennifer.m.Holland@usace.army.mil

