

GIS400/500 GIS Programming (SFASU)**Lab 1 ModelBuilder and Python**

Reference: ESRI, ModelBuilder – creating tools tutorial

Due: 11:00 am, Thursday, 1.31.2019

Goal:

Through this lab, students are expected to learn how to use ModelBuilder to develop tool dialog box (GUI) and how to export model to Python code.

Data

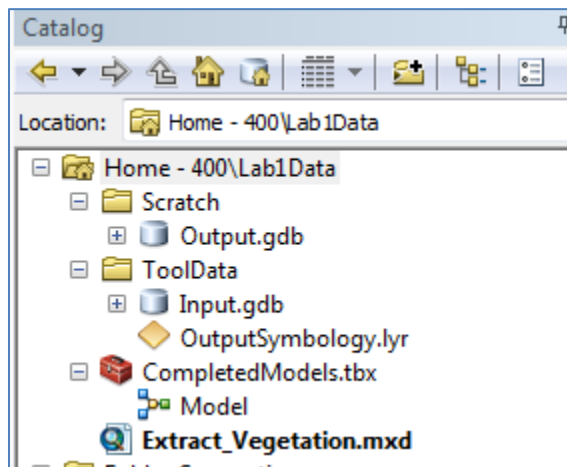
Y:\UnderGraduate\GIS400_500 \lab1, copy the folder to your drive (Z or flash drive), do not work directly on Y drive.

Instruction**1. Model introduction**

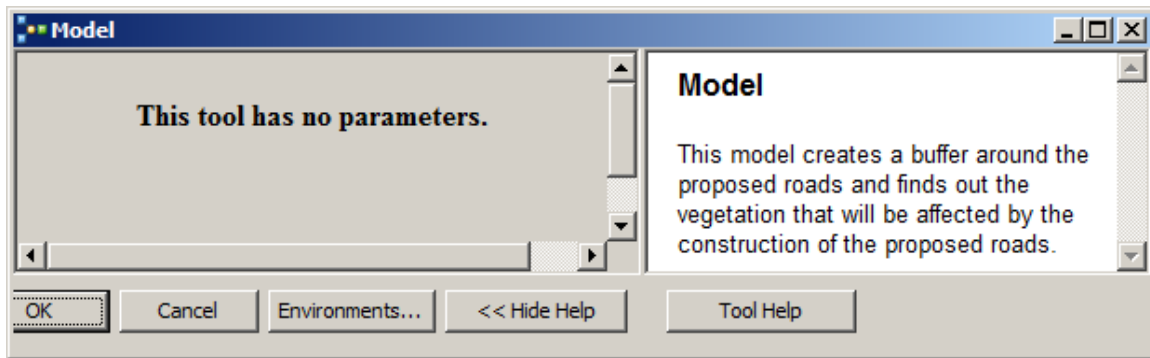
- a. Open the Extract_Vegetation.mxd file in the lab data folder.

In this lab, we have a simple model to identify vegetation types near proposed roads (buffer and clip).

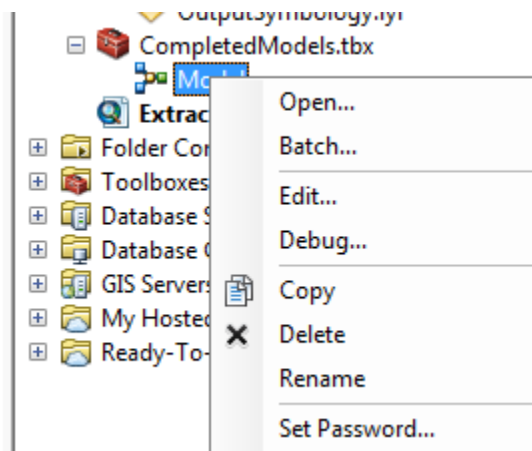
- b. Open the Catalog window within ArcGIS and navigate to “Model”.



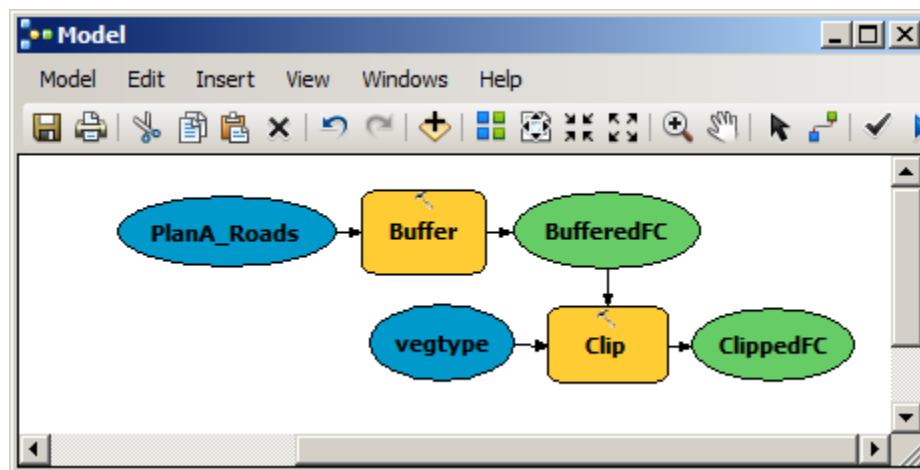
- c. Double click on Model and the tool dialog box opens but show no parameters.



- d. Right click on the Model in ArcCatalog and choose “Edit”.



- e. ModelBuilder window will open and show the model structure.

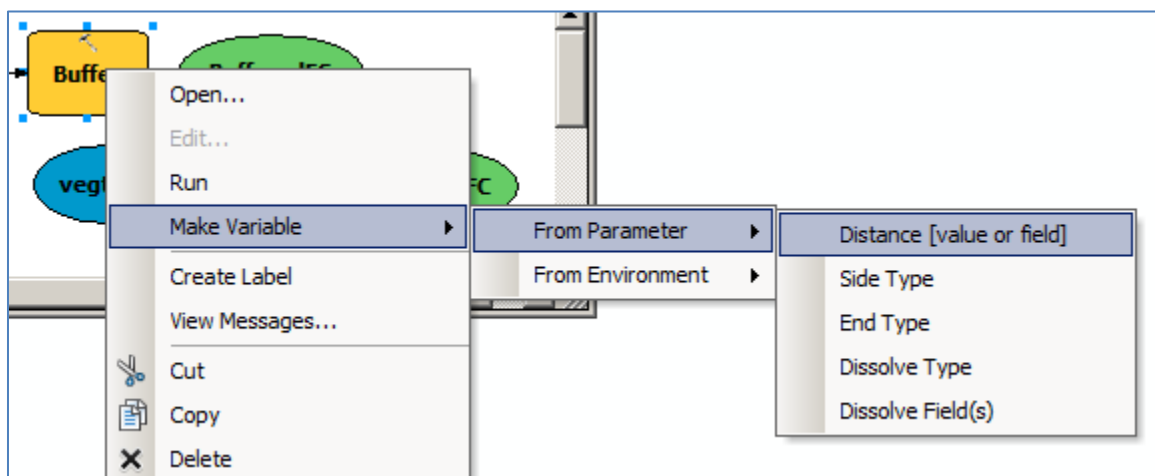


- f. Model in ModeBuilder is not convenient because the parameters (inputs and outputs) are “set”. If we need to change the input, we need to open the model elements to make changes. One of our purposes here is to create tool dialog box or GUI (Graphical User Interface) for the model.

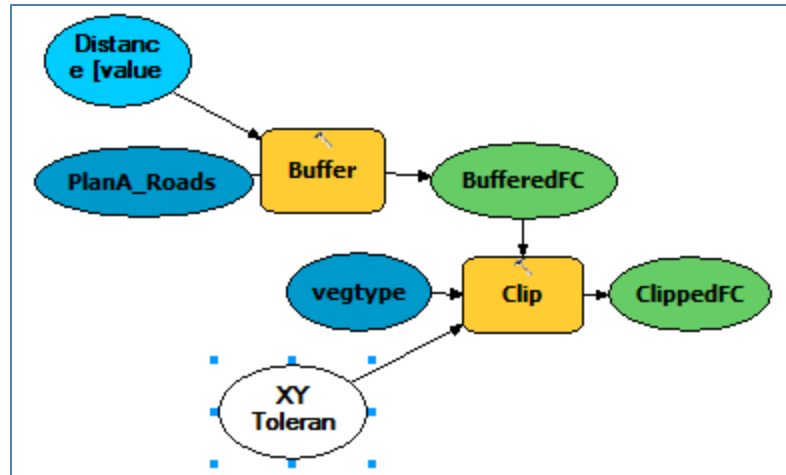
2. Exposing tool parameters

When you add a tool to a model, model variables are automatically created for input and output datasets, but not for any other tool parameters. The reason is aesthetics—if variables were automatically created for every tool parameter, the model diagram would quickly become unreadable. For example, when you add the Buffer tool to a model, a variable is automatically made for the Output Feature Class parameter. After you right-click Buffer and fill in the Input Features parameter, a model variable is created for the input features. All other parameters, such as Distance, Side Type, and End Type, are not automatically added as variables in the model. The steps below create model variables for Distance [value or field] and XY Tolerance.

- a. Right click Buffer in the ModelBuilder Window.
- b. Click Make Variable – From Parameter – Distance [Value or field]. This adds the Distance parameter as a variable in the model.



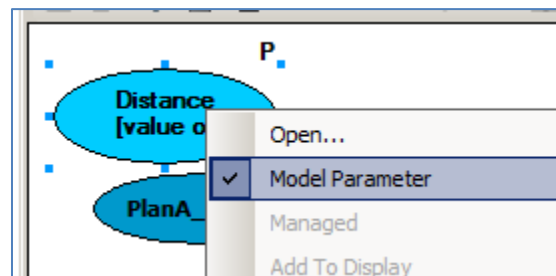
- c. Right click Clip in the ModelBuilder Window.
- d. Click Make Variable – From Parameter –XY Tolerance. Then the model will look like



3. Exposing tool parameters

Now we have created variables for Distance and XY Tolerance, we are ready to make model parameters.

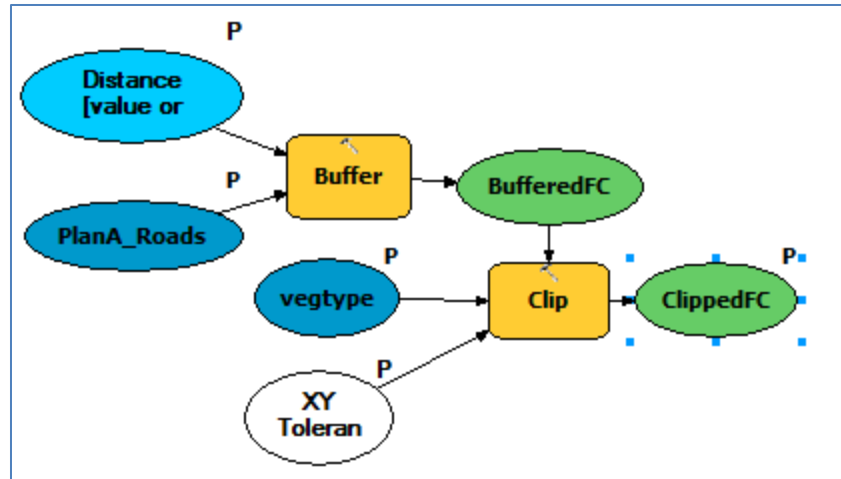
- Right click Distance [value or field] and check the Model Parameter option, as showing below. The letter P appears beside the variable, indicating it is a model parameter. This model parameter then also appears on the model tool dialog box.



- Create model parameters for the following variables (do not make a model parameter for BufferedFC):

PlanA_Roads
 Vegetype
 XY Tolerance
 ClippedFC

- Now the model should like

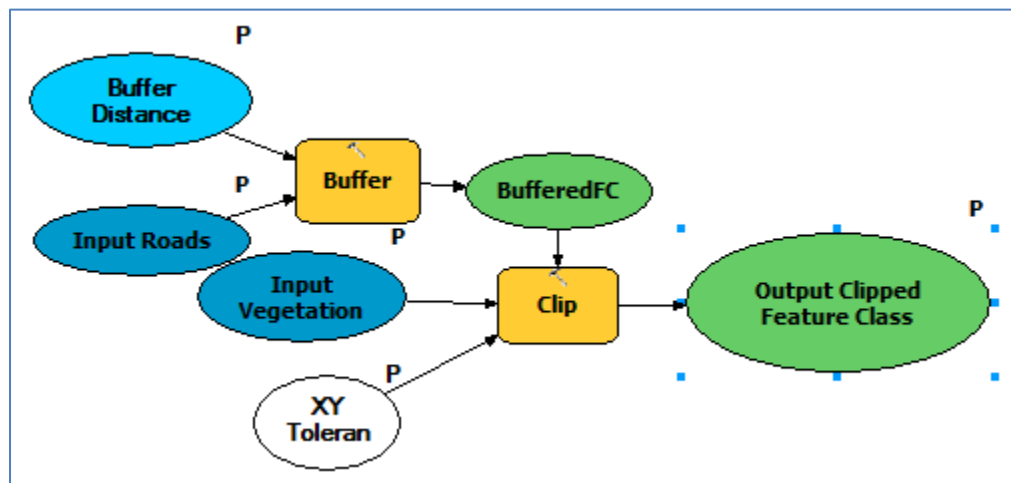


4. Renaming model elements

ModelBuilder assigns default names to variables. Variable names are used for parameter names on the model tool dialog box. It is good practice to rename variables, especially if they are model parameters.

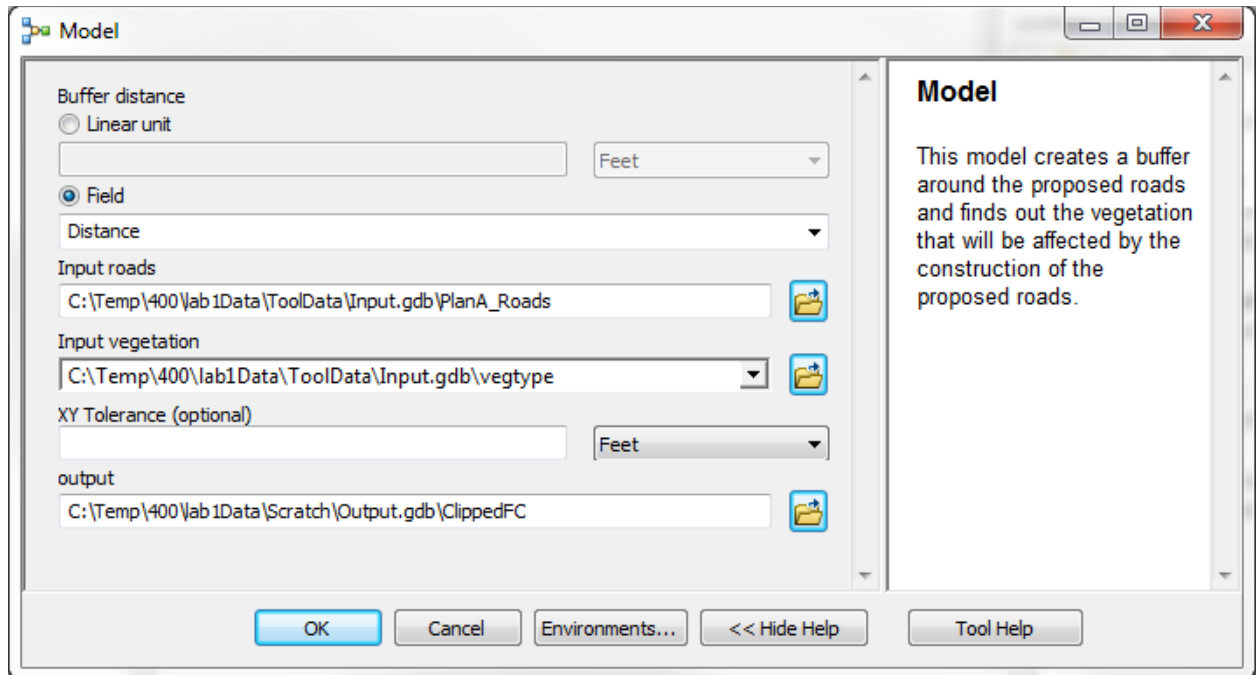
- Right click PlanA_Roads and click Rename.
- Type Input Roads and click OK.
- Rename the remaining variables as follows:

Distance [value or field]: Buffer Distance
 Vegetype: Input vegetation
 ClippedF: output Clipped Feature Class.



- Save the model. Do not exit ModelBuilder.
- Double click the model from the Catalog window to open the model tool dialog box. You may have to move or minimize the ModelBuilder window to access the Catalog window. The model tool dialog box should look similar to the illustration below. The

order of the parameters may be different on your dialog box, but this is not an issue since you will change the order below.



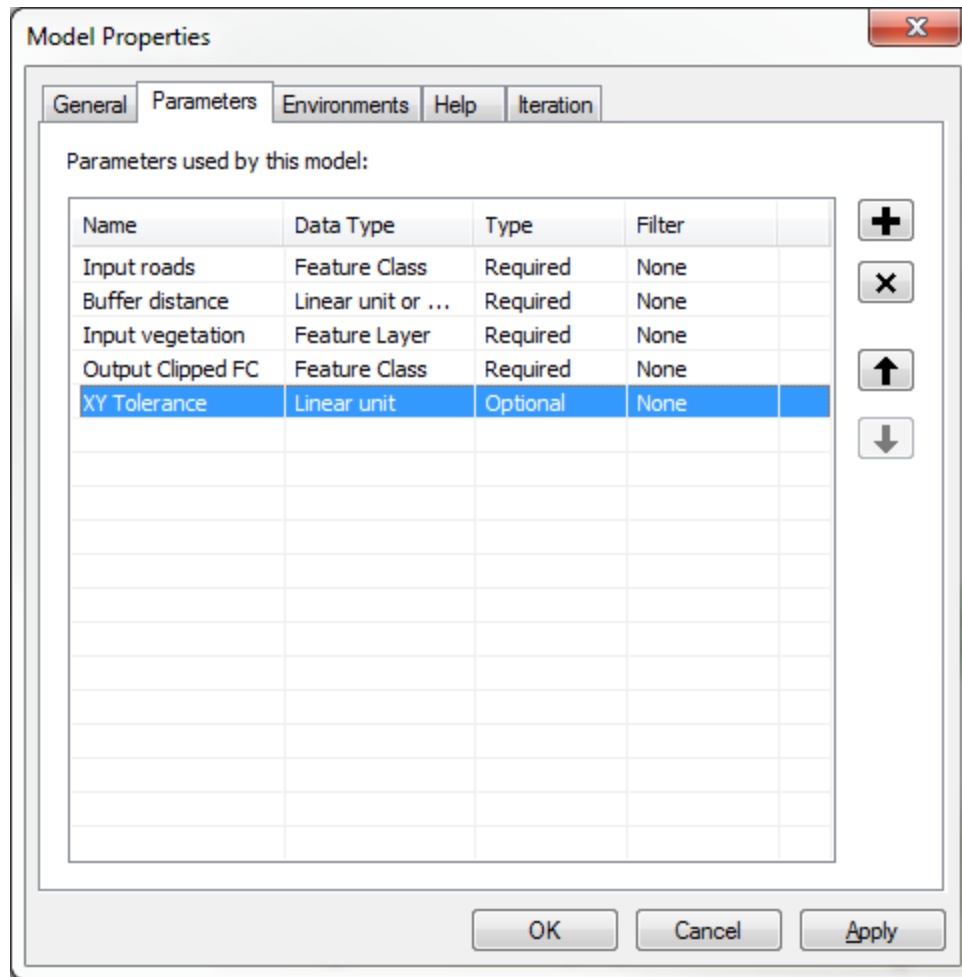
- f. You can execute the tool by clicking OK. But it is suggested that you choose a different output feature class before executing. The tool executes, and the output feature class is added to the ArcMap table of contents. Unlike running a model within ModelBuilder, running the model from its dialog box does not change the model diagram.

5. Setting model parameter order

As illustrated above, the order of the parameter is not ideal. The standard practice is to order parameters as follows:

- Required input datasets
- Other required parameters that affect tool execution
- Required output datasets
- Optional parameters

- a. In ModelBuilder, click Model – Model Properties.
- b. Click the Parameters tab.
- c. Choose the input Roads parameter and move it to the top using the Up and Down arrow buttons on the right side.
- d. Change the position of other parameters as showing below



6. Setting model parameter type

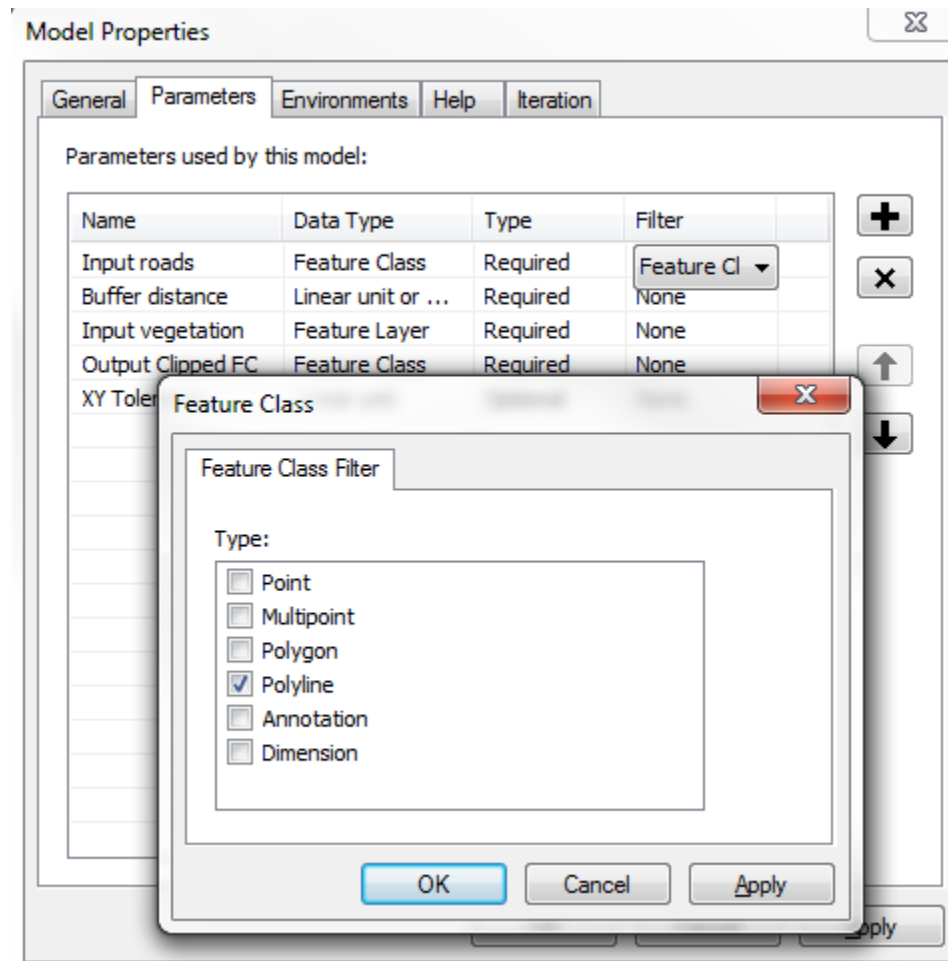
Once the model parameters are set in the correct order, change the type of the parameter. If a parameter is required parameter of a tool in the model, you will not be able to change the type to optional from these settings.

- a. With the Model Properties window still open, click the cell under the Type category for XY Tolerance as an optional parameter and the rest as required parameter type.

7. Setting filters on model parameters

We can restrict the type of input to any parameter by applying filters to parameters. The model in this example expects the Input Roads parameter to be line features. In the following steps, the parameter is modified by applying a filter so that it only accepts line features.

- With the Model Properties window still open, choose Input Roads and click the cell under the Filter category.
- Choose the Feature Class filter. The Feature Class dialog box opens.
- Uncheck all the types except Polyline and click Ok.
- Click OK on the Model Properties dialog box to apply the filter.

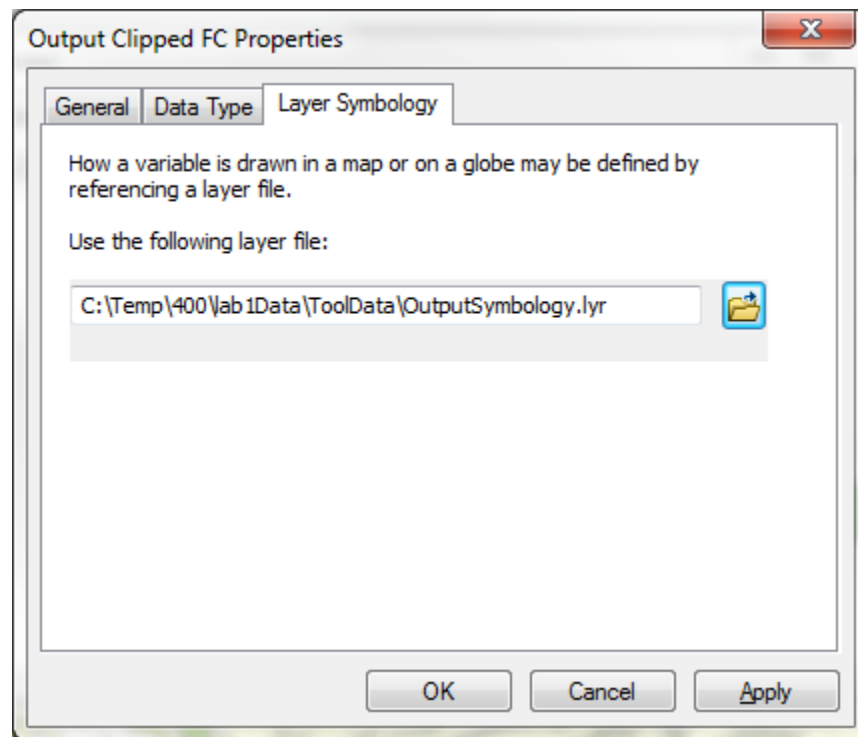


8. Setting symbology for output data

The output of a model can be set to have a particular symbology that is used to display the output. For this example, the symbology is based on the type of vegetation within the buffer zone. To set the symbology for the output data, the first step is to create a layer file, and the second step is to define the layer file in the output data properties. For this lab, a layer symbology file has been created for you.

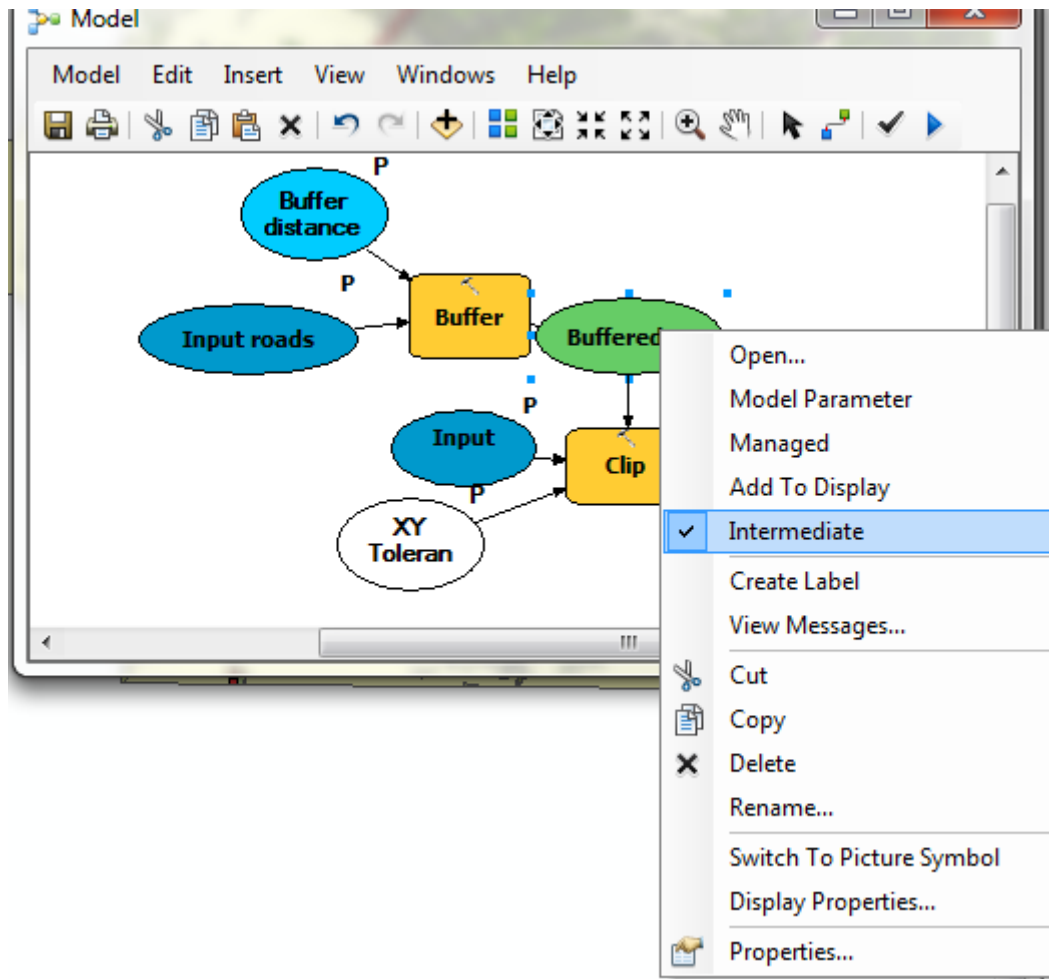
- In the ModelBuilder window, right click Output Clipped Feature Class and click Properties.
- Click the Layer symbology tab.

- c. Browse the layer file from the ToolData folder within Lab1 folder.
- d. Choose OutputSymbology.lyr and click Add.
- e. Click OK.



9. Managing intermediate data

When you run a model, output data is created for each process in the model. Some of the data created is of no use after the model is run since it was only created to connect to another process that creates new output. Such data is called intermediate data. All outputs except the final output, or those that have been made model parameters, are automatically made intermediate data in the model. In this example, the output of the Buffer tool is only useful as an input to the Clip tool and is not used after that, so the Intermediate option is checked. You can choose to save the intermediate data by unchecking the Intermediate option.



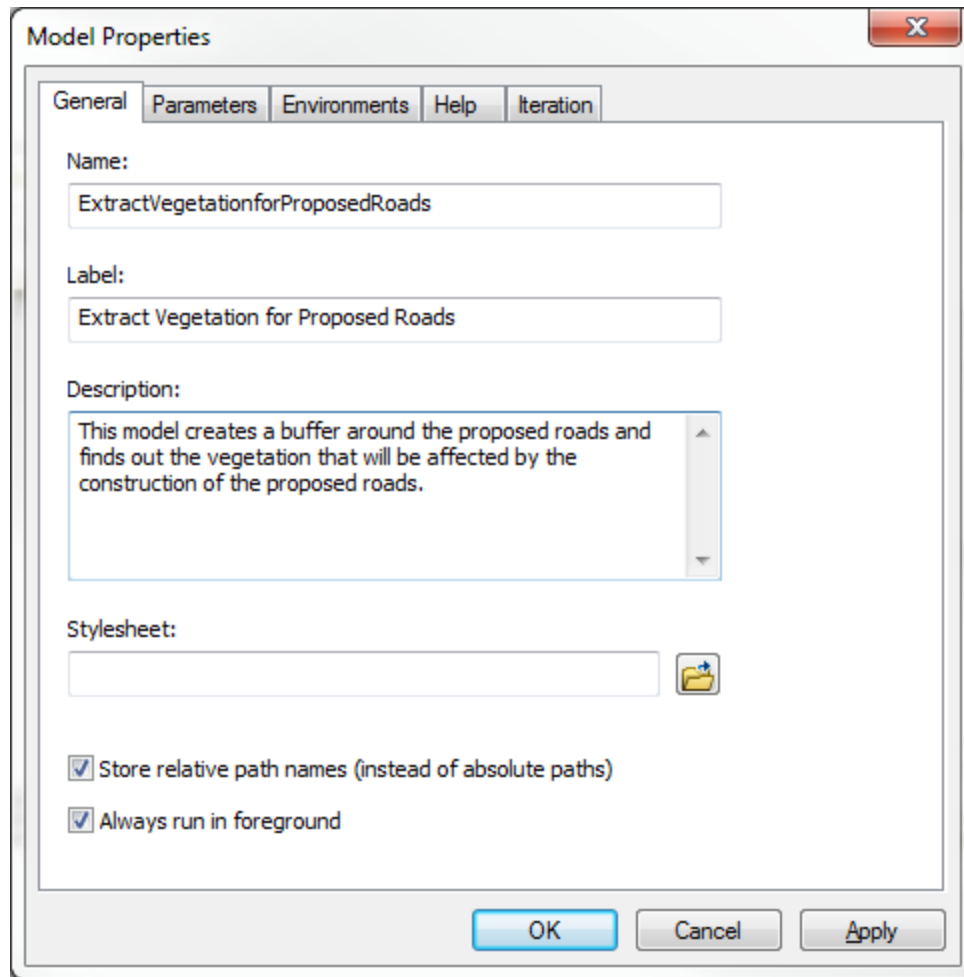
10. Changing general model properties

We can set the model name, label, and description.

- In the ModelBuilder window, click Model – Model Properties.
- Click the General tab, then type *ExtractVegetationforProposedRoads* for the name of the model. No spaces are allowed in the model name.
- Type *Extract Vegetation for Proposed Roads* in the Label text box. Spaces are allowed in the model label. This label is used to display the model name in the Catalog window.
- Type the desired text in the Description text box.

Check the Store relative path names (instead of absolute paths) option so that you can share your model tool or move your model data and model to a different location. This option is not useful in this tutorial but has been introduced here as a good practice to follow for your future models and model tools that you will share.

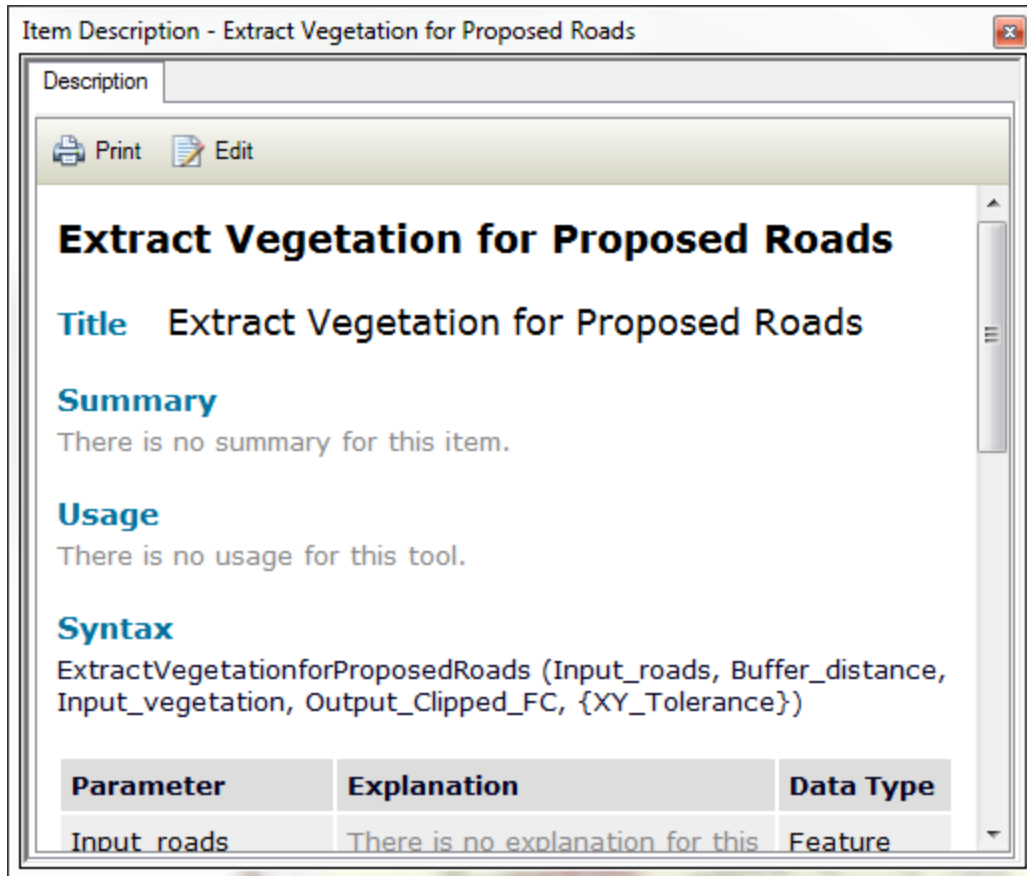
- Click OK.
- Click Model – Save. Then close ModelBuilder window and the tool dialog box.



11. Documenting the model

It is always a good practice to document the model before you share it.

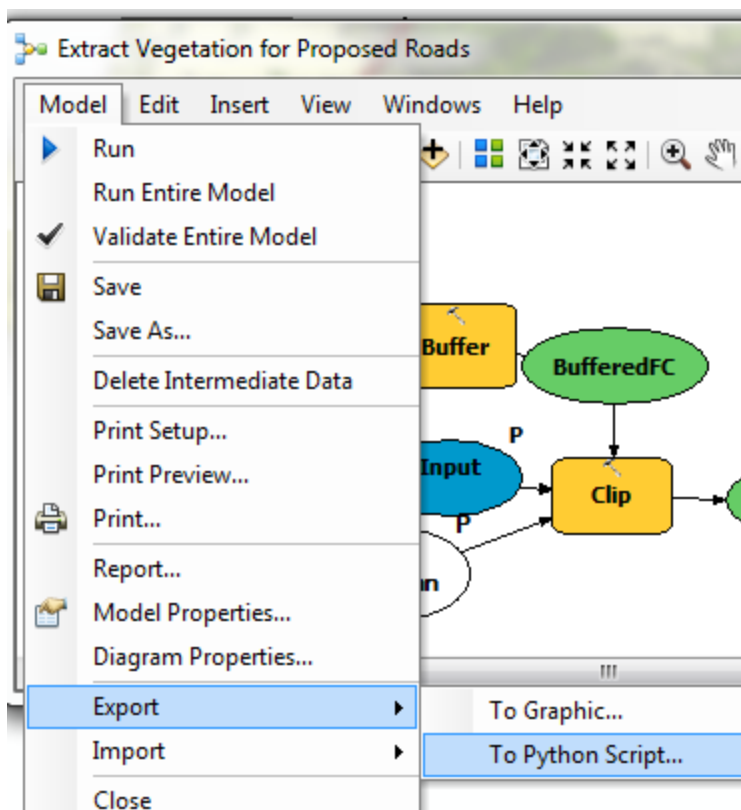
- Right click the model in in Catalog Tree and select Item Description to open the Item Description window.
- Click the Edit button on the Item Description window. This opens the documentation editor to enter the item description.
- Type in appropriate description for each item in this model and click the Save button.



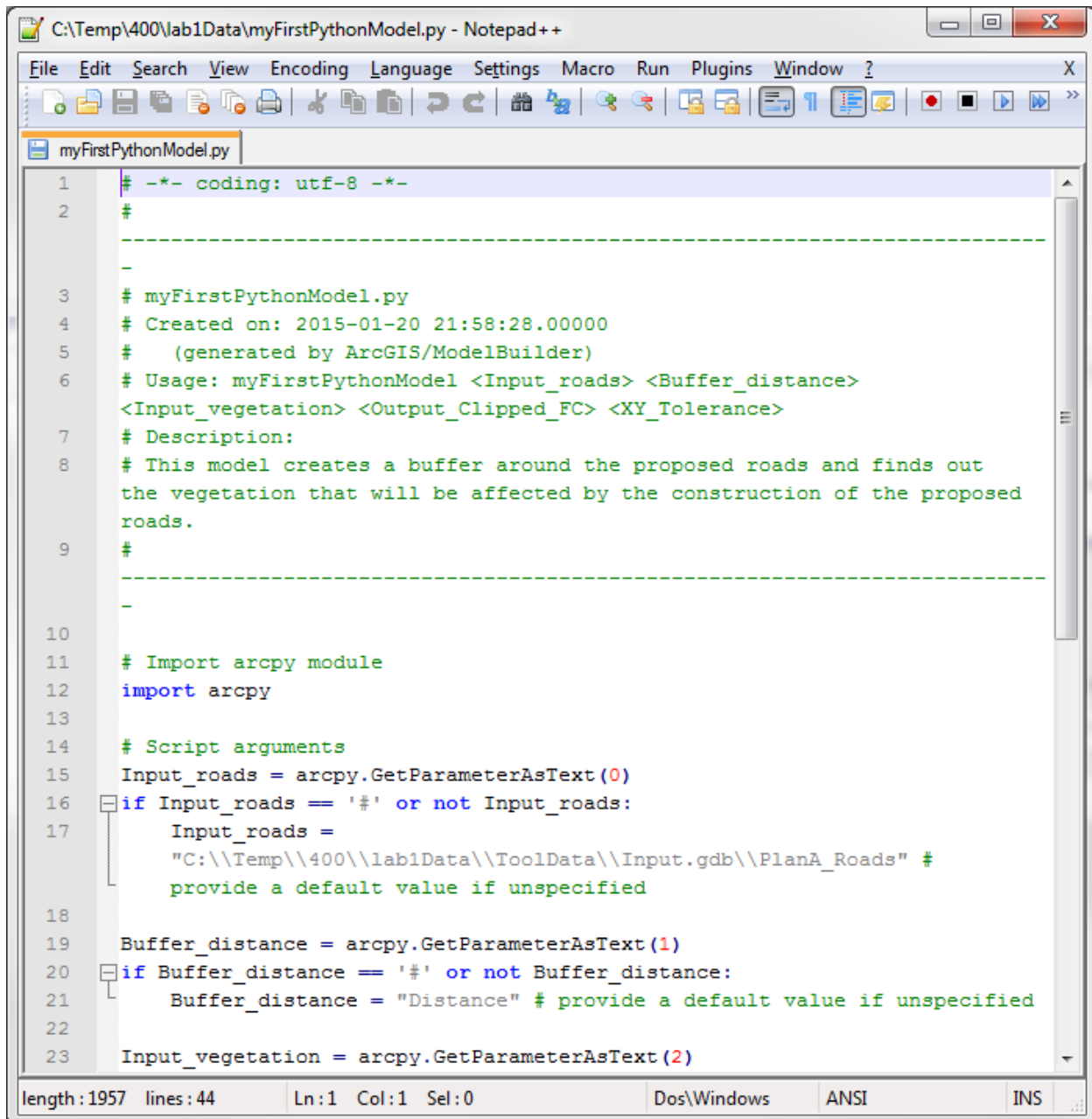
12. Export to python script

So far, we learned how to develop tool dialog box or GUI for a model. Next step, we will export the model to Python script.

- d. Right click the model in Catalog window and click Edit.
- e. Click Model – Export – To Python Script. The Save As window opens and name it as MyFirstPythonModel.py in your lab1 folder.



- f. Start Notepad++.
- g. Open MyFirstPythonModel.py with Notepad ++.
- h. Check the code by yourself.
- i. Print (in color) and submit the code.



```

1  # -*- coding: utf-8 -*-
2  #
3  # myFirstPythonModel.py
4  # Created on: 2015-01-20 21:58:28.00000
5  # (generated by ArcGIS/ModelBuilder)
6  # Usage: myFirstPythonModel <Input_roads> <Buffer_distance>
7  #         <Input_vegetation> <Output_Clippped_FC> <XY_Tolerance>
8  # Description:
9  # This model creates a buffer around the proposed roads and finds out
10 # the vegetation that will be affected by the construction of the proposed
11 # roads.
12 #
13 # Import arcpy module
14 import arcpy
15
16 # Script arguments
17 Input_roads = arcpy.GetParameterAsText(0)
18 if Input_roads == '#' or not Input_roads:
19     Input_roads =
20         "C:\\Temp\\400\\lab1Data\\ToolData\\Input.gdb\\PlanA_Roads" #
21         provide a default value if unspecified
22
23 Buffer_distance = arcpy.GetParameterAsText(1)
24 if Buffer_distance == '#' or not Buffer_distance:
25     Buffer_distance = "Distance" # provide a default value if unspecified
26
27 Input_vegetation = arcpy.GetParameterAsText(2)

```

length: 1957 lines: 44 Ln: 1 Col: 1 Sel: 0 Dos\Windows ANSI INS