Introduction to Model Builder and Python v5
License

http://creativecommons.org/licenses/by-sa/4.0/
Topics Covered

• ArcToolbox
• Geoprocessing
• Models
• Iterators
• Model Only Tools
• Exporting to Python and ArcPy
• Modeling and Scripting with Python
Why Build Models

- Automate repeated processes
- Flowchart Analysis Process
- Provide documentation for process
- Provide tools others can use to perform analysis
- Standardize workflows
Introduction to Model Builder

Chapter 1: ArcToolbox Basics
Intro To ArcGIS

• **ArcINFO** – released in 1982
  ◦ Version 4.0 introduced AML
  ◦ Arc Macro Language

• **ArcView GIS** – released in 1995
  ◦ Arcview 3.0 introduced Geoprocessing Wizard
  ◦ Also introduced Model Builder w/Spatial Analyst
  ◦ Avenue

• **ArcGIS** – 1999
  ◦ Model Builder
  ◦ Python appears in 9.1
Python

• Open Source
• Used everywhere:
  ◦ Microsoft, Google, NASA, Open Source GIS software, etc.
• Developed in the late '80s by Guido Van Rossum
  ◦ Benevolent Dictator for life
  ◦ Worked for Google
  ◦ January 2013 began working for DropBox
• ArcGIS 10.1 ships with 2.7
  ◦ 3.x is the current release
ArcToolbox

• The fundamental purpose of geoprocessing is to provide tools and a framework for performing analysis and managing your geographic data.

• Geoprocessing tools are tools ranging from simple to complex that let you work with your data
  ◦ Add a field
  ◦ Buffer a feature
  ◦ Clip data
ArcToolbox

- ArcToolbox is made up of tools, scripts, and models.

- Can be executed one at a time or several at once.
Tools and Scripts

- Every Tool has a Python "Interface"
- Tools can live within Toolsets. Toolsets are a "directory" of tools
- Scripts and Tools can be combined in Models in ArcToolbox
Toolboxes Can Be Built Anywhere

- Stored in:
  - Directories
  - Geodatabases

- Can be shared

- Can be stored on ArcGIS Server
ArcToolbox

• Built into ArcCatalog and ArcMap

• ArcMap and ArcCatalog have an ArcToolbox window.

• Tools/Toolsets/Scripts can be deleted from an ArcMap MXD.

• Deleted tools can be restored from the System Toolbox located in ArcGIS Install folder.

• Can save customizations to and load from an XML
ArcToolbox in ArcMap
ArcToolbox in ArcCatalog
Exercise 1

Explore ArcToolbox

Get to know ArcToolbox organization

Examine different ways to access geoprocessing tools

15 to 20 minutes
Introduction to Model Builder and Python

Chapter 2: Geoprocessing Basics
2. Geoprocessing Tools

- ArcToolbox holds Geoprocessing tools
- Tools take an input (raster/vector/table) and a output (raster/vector/table) and several parameters to run
- These tools typically do three things usually to data
  - Modify Data
  - Create/Delete Data
  - Reports/Statistics on Data
Geoprocessing Tool

- Example of a tool: Buffer
Geoprocessing Tool

• Example of a Tool: Copy Data

Copy input data and pastes the output to the same or another location regardless of size. The data type of the input and output data element is identical.
Geoprocessing Options

• Found under the Geoprocessing Menu in ArcMap and ArcCatalog

• Controls how Geoprocessing runs.....
  ◦ Background Processing
  ◦ Foreground Processing
  ◦ Results (Temporary or Permanent)
  ◦ How long results are kept per MXD
Geoprocessing Options

- Overwrite the outputs of geoprocessing operations
- Log geoprocessing operations to a log file

Background Processing
- Enable Notification
  Appear for how long (seconds)
- Stay up if Error occurs

Script Tool Editor/Debugger
- Editor:
- Debugger:

ModelBuilder
- When connecting elements, display valid parameters when more than one is available.

Results Management
- Keep results younger than: 2 Weeks

Display / Temporary Data
- Add results of geoprocessing operations to the display
- Results are temporary by default

About geoprocessing options
Processing

- Foreground
  - Focus is on Processing
  - One Process is run at a time
  - Wait for the one process to finish

- Background
  - Process is "forked off" and runs by itself
  - Supports multiple processes being run at once
  - More Flexible
Results Window

- When you run a Geoprocessing tool your data is saved in the results window
  - Data location
  - Time run
  - Variables
  - Saved in MXD
  - Shared as a Geoprocessing Package (GPK) or Service
  - It is kept by default is two weeks
Results Window

- Found under Geoprocessing Menu
- Tools can be rerun from here without filling out all the information again
- Check messages if something happened
Results Window

- Notice your options when the result is right-clicked
Environmental Variables

These are parameters that are entered to allow the tool to run.

- File names
- Buffer widths

Four Levels
- Application
- Tool
- Model
- Model process
Environments

- Workspace
- Output Coordinates
- Processing extent
- XY Resolution & Tolerance
- M Values
- Z Values
- Geodatabase
- More...
Application Level

Set through Geoprocessing menu or ArcToolbox
Tool Level
Model Level
Model Process Level
Exercise - Chapter 2

- Open ArcToolbox
- Create a Toolbox
- Look at the Geoprocessing Options box
- Change if tools run foreground or background
- Look at the Results Window
- 15 to 20 minutes
Exercise

The data for this project comes from a Project with the Conasauga River Alliance.

This data was collected from Aerial photography and the National Hydro Dataset.

1. You should have a **C:\modelbuilder** directory. If not please create one. Under that you should be a data directory. Add all the shapefiles to your display and symbolize appropriately.

2. Create a **Toolbox** directory, a **MXD** directory, and an **output** directory.

3. Save this map document under the **c:\modelbuilder** directory as **Exercise2.mxd**.

4. Double check and make sure you have the following directories: **mxd**, **output**, and **toolbox**.

5. Open **ArcCatalog** (Either in Arcmap or as a standalone application).
Exercise

6. In ArcCatalog **Create a Toolbox** by right-clicking in the toolbox directory, selecting **New** and then **Toolbox**. Give it a name.

7. Close ArcCatalog

8. Open ArcToolbox. Explore the tools and scripts that are contained under the toolbox application.
Exercise

9. Open the **Geoprocessing Options** menu under the **Geoprocessing menu** at the top of ArcMap.

10. Check **Overwrite the outputs of geoprocessing operations**.

11. Turn off **Background processing**

12. Change Keep your **Results** to "never delete".

13. Click **OK**.
Exercise

14. Open the **Buffer** tool in ArcToolbox.

15. Buffer the streams **200 Feet**.

16. Save the output - **Buffer200Feet** in the output directory. Set the **Dissolve Type** to **All**.

17. Check the **Results** window by click on the Geoprocessing menu and going to Results.

18. Can you save to Python code from the results window? Right Click and see if you can

19. Save your MXD!
Introduction to Model Builder and Python

Chapter 3: Modelbuilder
3. Models

- Geoprocessing processes can be saved to use later
- Geoprocessing tools can be linked together to form a process
  - A Model
- Different tools can be linked in one model
Toolbox

- In order to save a model you must create a new toolbox

- Toolboxes can be created almost anywhere in ArcMap/ArcCatalog
  - Create directories to place toolboxes
  - Can be placed in geodatabases
    - File-based
    - Personal-based
  - SDE
Toolboxes and Models

• Once a Toolbox is made -> Create a Model
First Opened - Boring
Model Builder Interface
Creating a Model

Drag and drop tools from ArcToolbox onto the Model Builder Canvas

- Tool processes take shape and color according to purpose
- Shadows mean a tool has run
- Processes flash red when running
Symbology

Process State

Not Ready-To-Run
- Input Data
- Input Value
- Tool
- Iterator
- Derived Data
- Derived Value
  - No Color
  - Colored

Ready-To-Run
- Input Data
- Input Value
- Tool
- Iterator
- Derived Data
- Derived Value
  - Tools turn red
  - No change in variables

Running
- Tool
- Iterator
- Derived Data
- Derived Value
  - Tools and output get a shadow

Has Been Run
- Tool
- Iterator
- Derived Data
- Derived Value
  - (No change in variables)

(No change)
(No change)
Model with no Inputs
Fill in the Parameters
Hit the "Run" Button

Executing (Buffer): Buffer Streams C:\Users\rjhale\Documents\ArcGIS\Default.gdb\Streams_Buffer2_Buffer "500 Feet" FULL ROUND NONE #
Start Time: Mon Sep 03 20:29:04 2012
Success at Mon Sep 03 20:29:09 2012 (Elapsed Time: 5.00 seconds)
Results

- Once the Model runs it gains "shadows"

- Model must be validated to run again from beginning.
Exercise 3

- Build a simple model
- Save the model to be shared
- Work with Results Window
- 20 Minutes
Exercise

1. Hopefully you have your MXD open from the previous exercise. If not open it.

2. Open a New Model by clicking on the Model Builder Icon in ArcMap.

3. Save that Model to the Toolbox you created in the previous exercise. Name it **Buffer200Ft**

4. Drag the Buffer Tool into Model Builder. Double-click the Buffer tool and Fill out the values as you did in the previous exercise. You want a 200-Foot Buffer and the dissolve option set to ALL

5. Save the Model. Click "Run"

6. What color indicates the Model is Running? What has been added or changed with the model?

7. Once a Model has been run it needs to be validated. Click Model -> Validate Entire Model to prepare it to run again.
8. Check the **Results Tab** - Was your Model recorded?

9. Create a **New Empty Model**. Save it and Call it **ResultsBuffer**.

10. You should have a buffer from the previous exercise in the results window. Drag a buffer from the previous Exercise out of the results window into the Model. What happens?

11. Find the **Clip Command** in ArcToolbox. It is located under **Analysis Tools -> Extract**.

12. Right-click the tool and click **Copy**.

13. Go to your Toolbox, Right-click and select **Paste**. You have now saved a copy of the Clip Tool to your Toolbox. If you have a project where the user only needs certain tools they can be placed here.

14. Be sure all your models are saved!
Chapter 4: Variables & Parameters
4. Variables and Parameters

• Every tool has Parameters
  • Parameters allow the user to make changes to a tool.

• Parameters can be exposed as Variables
  • Variables can be changed by users
Parameters

• Every Geoprocessing tool has Parameters. Parameters control how the tool conducts itself.

• The Buffer tool has eight Parameters:
  - input
  - output
  - Distance
  - Units
  - Side Type
  - End type
  - Dissolve
  - Dissolve Fields
Parameters exposed
Parameter

- Double-clicking the Parameter will allow you to set it.
Parameters can be exposed as variables in Model Builder.
Why expose a Parameter?

Make it a Variable

- Variables allow for user interaction
- Models can be shared and allow some flexibility for the end user with Variables
- Models come with a User Interface - Variables show up on that GUI
The Buffer Tool

- From ArcToolbox
- As a Model with three parameters

![Buffer Tool Interface](image1)

![Model Interface](image2)
Setting Variables

- Right Click and make it a parameter
Rename Parameters

- Renaming Parameters renames them in the GUI
- Can make them easier to understand for users
Exercise 4

• Control and expose tool variables
• Set variables to parameters
• 20 Minutes
Exercise

1. Double-click the **Buffer200Ft** Tool in your Toolbox. We need to add Parameters to the Model.

2. Click Cancel and then Right-click the Buffer200Ft Model and click **Edit**. You should have the Model Builder interface back in front of you.

3. Right-click the Streams (or **Input**) and make it a **Parameter**.

4. Right-click the **Output** and make it a **Parameter** also.
Exercise

5. **Save** the Model Double-click it again and see what changed.

6. Click **Cancel** and begin editing the Model once more.

7. Right-click the **Streams** and **Rename** it to **Input Dataset**.

8. Right-click the **Buffer200Feet.shp** dataset and **Rename** it to **Output Dataset**.
Exercise

9. Right-click the square **Buffer Tool** and make a **Parameter from the Variable**. Right-click and make it a **Model Parameter**.

10. Save the Model.

11. Double-click the Model in the Toolbox.
Introduction to Model Builder and Python

Chapter 5: Connecting Models
5. Connecting Models

- Tools Can be linked together
- Outputs of one tool can be inputs for another tool
- Tools order is controlled by Preconditions
Connect

Connection Tool

Input Dataset → Buffer → Output Dataset

Distance [value or field]

Clip → Output Feature
Connect

Input Dataset → Buffer → Output Dataset

Distance [value or field]

Connection

Output Feature

- Input Features
- Clip Features
- Environments
- Precondition
Connect
Preconditions

Preconditions make sure one element runs before the other.
Setting a Precondition

- Clip will not run until buffer runs
Exercise 5

• Connect two tools together using different methods
• Control order in which tools are run using preconditions
• 20 Minutes
1. Open your MXD from the previous exercise if you closed it.

2. Open a **New Model** by clicking on the Model Builder Icon in ArcMap.

3. Add a Buffer tool to the model. Also add a **Select by Location** tool to the Model. Use the **Search** for tools menu to locate it. This menu is located under the Geoprocessing Menu in ArcMap.

4. Connect the **Output Feature Class** to the **Select Layer by Location** tool using the **Connect** icon. Be sure you choose selecting features.

6. Click **Autolayout**. Notice your Model will get "Re-arranged".
Exercise

7. Start filling out features in the Model. You want to Buffer the streams 100 Feet. The Input feature layer in the Select By Location tool will be the Structures.

8. Click Autoarrange.

9. Run the Model

10. How many Structures are within 100 Feet of the Streams?
Exercise

11. Delete the connection between the two tools.

12. Click Autoarrange.

13. Double-click the **Select by Location**. Set the selecting features to be the **Output of the Buffer** command

14. What happens?
15. Save your Model as **BufferSelection100Ft**.

16. Create a **New empty Model**. A Model doesn't have to contain connected tools. It can contain just a series of tools that need to be run at the end of a project. Copy the **Calculate Field** Tool and the **Delete Management** tool into your model (using the **Search** for tools).

17. You need the tools to run in a certain order. Right-click the **Delete** tool and select Properties. Go to the **Precondition Tab**. Select **Output Feature Class**. Click OK.

18. What happens?

19. The Tools will now run in a certain order.

20. You don't have to save this model. Just close it.
Introduction to Model Builder and Python

Chapter 6: Iterators & Model Only Tools
6. Iterators and Feature Only Tools

- **Iterators**
  - Looping
  - Loop through data (vector, raster).
  - Loop through tables

- **Model Only Tools**
  - Collect Values
  - Calculate Values
  - Parse Data
import arcpy
from arcpy import env
import os

# Set the workspace.
# List all of the feature classes that start with 'G'

env.workspace = "D:/St_Johns/data.gdb"
fcs = arcpy.ListFeatureClasses("G*")
After Iterators
Iterators

They don't export to Python

They can slow down a Model (IMO)

They are easy to use though
# Iterators

<table>
<thead>
<tr>
<th>Iterator</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>For</td>
<td>Iterates over a starting and ending value by a given value. It works exactly like for in any scripting/programming language, executing through a set number of items.</td>
</tr>
<tr>
<td>While</td>
<td>Works exactly like 'while' in any scripting/programming language, executing &quot;while&quot; a condition is true or false for the input or set of inputs.</td>
</tr>
<tr>
<td>Iterate Feature Selection</td>
<td>Iterates over features in a feature class.</td>
</tr>
<tr>
<td>Iterate Row Selection</td>
<td>Iterates over rows in a table.</td>
</tr>
<tr>
<td>Iterate Field Values</td>
<td>Iterates over each value in a field.</td>
</tr>
<tr>
<td>Iterate MultiValue</td>
<td>Iterates over a list of values.</td>
</tr>
<tr>
<td>Iterate Datasets</td>
<td>Iterates over datasets in a Workspace or Feature Dataset.</td>
</tr>
<tr>
<td>Iterate Feature Classes</td>
<td>Iterates over feature classes in a Workspace or Feature Dataset.</td>
</tr>
<tr>
<td>Iterate Files</td>
<td>Iterates over files in a folder.</td>
</tr>
<tr>
<td>Iterate Rasters</td>
<td>Iterates over rasters in a Workspace or a Raster Catalog.</td>
</tr>
<tr>
<td>Iterate Tables</td>
<td>Iterates over tables in a workspace.</td>
</tr>
<tr>
<td>Iterate Workspaces</td>
<td>Iterates over workspaces in a folder.</td>
</tr>
</tbody>
</table>

**Review the online help:**
Feature Selection Iterator loops through each feature in a feature class.
Iterate Feature Class

The Feature Class Iterator loops through all the feature classes in a directory or geodatabase.
## Model Only Tools

<table>
<thead>
<tr>
<th>Model Only Tool</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Calculate Value</strong></td>
<td>Calculate Value tool returns a value based on a specified Python expression.</td>
</tr>
<tr>
<td><strong>Collect Values</strong></td>
<td>The Collect Values tool is designed to collect output values of an iterator, or to convert a list of multivalues into a single input. The output of Collect Values can be used as input to tools like <a href="#">Merge</a>, <a href="#">Append</a>, <a href="#">Mosaic</a>, and <a href="#">Cell Statistics</a>.</td>
</tr>
<tr>
<td><strong>Get Field Value</strong></td>
<td>The Get Field Value tool gets the value of the first row of a table for the specified field.</td>
</tr>
<tr>
<td><strong>Merge Branch</strong></td>
<td>The Merge Branch tool merges two or more logical branches into a single output.</td>
</tr>
<tr>
<td><strong>Parse Path</strong></td>
<td>The Parse Path tool parses the input into its file, path, name, or extension. The output can be used as in-line variables in the output name of other tools.</td>
</tr>
<tr>
<td><strong>Select Data</strong></td>
<td>The Select Data tool selects data in a parent data element such as a folder, geodatabase, feature dataset, or coverage.</td>
</tr>
<tr>
<td><strong>Stop</strong></td>
<td>For the set of input values, iteration will stop if all the input values meet the specified condition of either True or False. It is functionally similar to <a href="#">While</a> iterator but is useful to stop a model if there is one While iterator in a model and no additional iterators can be added.</td>
</tr>
</tbody>
</table>
Example of Parse Path

Choose a parse type from File, Path, Name, or Extension. Given the input value of C:\ToolData\InputFC.shp:

- **FILE**—Output will be the file. Example: InputFC.shp
- **PATH**—Output will be the file path. Example: C:\ToolData
Parse Path

- Path was remove from filename
Exercise 6

- Work with Iterators in ModelBuilder
- Work with Model Only Tools
- 20 Minutes
Exercise

1. Open your MXD from the previous exercise if you closed it.

2. Open a **New Model** by clicking on the Model Builder Icon in ArcMap.

3. Add the **Iterate Field Values** Iterator to the Model Right-click and select it from the Iterator menu.

4. Open the Iterate Tool. For the input table select the Watershed.shp.

6. For the field select the **Hu_10_name** field.

7. Click OK.
Exercise

8. **Search** and add the **Select** (analysis) tool.

9. Double-click the Select tool and add **Watershed.shp** as the **Input Feature**.

10. Make the output `c:\modelbuilder\%Value%.shp`. The Iterate tool outputs a variable with the value of the **Hu_10_name** field.

11. Make the Expression be the following: "**HU_10_Name**" = '%Value%'

12. Run the Tool

13. Save it and give it a name.
Exercise

Bonus:
Take each feature in the Watershed shapefile and use it to clip out the streams.
Introduction to Model Builder and Python

Chapter 7: Python & ArcPy
7. Python and ArcPy

Python is a remarkably powerful dynamic programming language that is used in a wide variety of application domains. Python is often compared to Tcl, Perl, Ruby, Scheme or Java. Some of its key distinguishing features include:

• very clear, readable syntax
• strong introspection capabilities
• intuitive object orientation
• natural expression of procedural code
• full modularity, supporting hierarchical packages
• exception-based error handling
• very high level dynamic data types
• extensive standard libraries and third party modules for virtually every task
• embeddable within applications as a scripting interface
GIS

GIS software is currently having a love affair with Python

**Open Source:**
QGIS, PostGIS, OSGeo, GDAL, gispython.org

**Closed Source:**
ArcGIS is probably the biggest
Install of ArcGIS

Python 2.7
- IDLE
- PythonWin

For editing I prefer notepad or VIM
Features of Python

- Control Flow
  If, For, Else, While

- Declare Variables
- Work with Integers

- Capture Errors

- Build modules to import that do specific things
  ARCPY is the one you will be most concerned with
Resource Sites

http://www.python.org
http://www.codeacademy.com
http://www.swaroopch.com
http://www.oreilly.com
ArcGIS Help Documentation
ArcPy

ArcPy is a site package that builds on (and is a successor to) the successful arcgisscripting module. Its goal is to create the cornerstone for a useful and productive way to perform geographic data analysis, data conversion, data management, and map automation with Python.

ArcPy provides access to geoprocessing tools as well as additional functions, classes, and modules that allow you to create simple or complex workflows quickly and easily.

#import arcpy;
Interface(s)
Interface(s)
So for ArcUsers....... 

- Add Python to your path statement 
  Found under Environment Variables 
  Add: C:\Python27\ArcGIS10.1
Example of a for Statement

```python
import arcpy
from arcpy import env
import os

# Set the workspace for the ListFeatureClass function
env.workspace = "c:\Data"

# Use the ListFeatureClasses function to return a list of
# all shapefiles.
# fcList = arcpy.ListFeatureClasses()

# Prints list of shapefiles in workspace to screen
# for fc in fcList:
    print fc
```
Variable Example

Variables can be text or numeric

Scripts can accept inputs
• InputTable = arcpy.GetParameterAsText(0)
• Census = arcpy.GetParameterAsText(1)

If you build a script and want to pass data into it, you would use: arcpy.GetParameterAsText()
ArcPy

As with Model Builder - Know your Geoprocessing Tools

You do not have to be an expert programmer to get something done in a script

Make mistakes

I have made hundreds and will continue making hundreds more
Exercise 7

• No exercise
• Break
Introduction to Model Builder and Python

Chapter 8: Exporting a Model to Python
8. Exporting Model to Python

- Model Builder can export to Python
- Iterators can't be exported
- Model Only Tools can't be exported
Once Exported

• Script can be customized to fit your need
• Can be re-imported into model Builder
• Can be run from Python window in ArcMap
Add script back to ArcToolbox

If you write a script look at the variable

You can make an input and output

Will appear that way in Model Builder
Tips

- Simpler the better
- Don't get too fancy
- Make lots of notes #because notes are important
- Nothing works right the first time
- It might not work right the second time
Example of True/False

```python
# Import modules
import arcpy
import sys
import traceback

# Set local variables
data = 0
indata = arcpy.GetParameterAsText(0)
arcpy.AddMessage(indata)

try:
    data = int(str(arcpy.GetCount_management(indata)))
arcpy.AddMessage('***************')
arcpy.AddMessage(data)
arcpy.AddMessage('***************')

    if data <= 2:
        # Param1 is LT; Param2 is GT
        # set appropriately to control the flow
        arcpy.SetParameterAsText(1, 'true') # LT
        arcpy.SetParameterAsText(2, 'false') # GT
        arcpy.AddMessage('***************')
arcpy.AddMessage('Less than 2....')
arcpy.AddMessage('***************')
def data

    else:
        arcpy.SetParameterAsText(1, 'false') # LT
        arcpy.SetParameterAsText(2, 'true') # GT
        arcpy.AddMessage('***************')
arcpy.AddMessage('Greater than 2....')
arcpy.AddMessage('***************')
def data

except:
    tb = sys.exc_info()[2]
tbinfo = traceback.format_tb(tb)[0]
pymsg = tbinfo + 'n' + str(sys.exc_type) + ' : ' + str(sys.exc_value)
```
Added to ArcToolbox
Added To Model Builder
Exercise 8

- Export a model to python
- Load a Python Script into Python Window
- 20 Minutes
Exercise

1. Open your MXD from the previous exercise if you closed it.

2. Open the Buffer200Ft Model.

3. Go to Model -> Export -> To Python Script.

4. Right-click the script in Windows Explorer and Edit it with IDLE.

6. Take a look at the script that is created from Model Builder.

7. If you want create a very simple Model and export it to Python and see what it looks like.
Exercise

8. In ArcMap, open your Python console.


10. What happens?

11. Finish out the command. It should look something like:

   ```python
   arcpy.Buffer_analysis("Streams", "C:\modelbuilder\data\streamstest.shp", 200, "FULL", "ROUND", "ALL")
   ```

12. Right-click in the Python console and select Save as and save that to a Python script.

13. Take a look at the script.
Exercise

14. Right-click in your Python console and click **Load**

15. Load the **Buffer** script that you exported from Model Builder earlier.

16. Press **Return** in the Python window. What happens?

17. Open Your Python Console.

18. Type: import arcpy

19: Copy your **Buffer** command from your saved script (see below) into the window.

```python
arcpy.Buffer_analysis("Streams", "C:\modelbuilder\data\streamstest.shp", 200, "FULL", "ROUND", "ALL")
```

20. Press **Return**.

21. What happens?
Introduction to Model Builder and Python

Chapter 9: Documenting your work
9. Writing Help

• You've done all this work

DOCUMENT IT
Item description

- Right-click the Model in ArcToolbox or ArcCatalog
- Go To Item Description
- Think about why you created this Model/Script and who can use it.
- Edit the description
Item Description

**Toolbox**

**Title**  Toolbox

**Description**
There is no description for this item.

**Summary**
There is no summary for this item.

**Tags**
There are no tags for this item.

**Credits**
There are no credits for this item.

**Use limitations**
There are no access and use limitations for this item.
Item Description Edited
Exercise

1. Open your MXD from the previous exercise if you closed it.

2. Open the **Buffer200Ft Model**

3. Go to Model -> Export -> To Graphic. Make the file type jpg.

4. Right-click the Model in ArcCatalog and go to **Item Description**.

6. Click **Edit**.

7. Add the exported Graphic.

8. What are tags and why are they important?
Exercise 9

- Create metadata for your model
- Show how it helps when running it
- 20 Minutes
Authors:
Carol Kraemer – Friend, Wizard, Cat Whisperer
Randal Hale – Blues Aficionado, Uphill Battles, Canoeist

North River Geographic Information Systems, Inc
http://www.northrivergeographic.com

Miso. 3.5 lbs of happy.

If you got this far in the class you should go and donate some Resources to your local human/animal shelter.