Exercise 3

Geoprocessing in ArcGIS Pro

Exercise data

Exercise data for this book can be downloaded from links.esri.com/PythonPro3rdEditionData. This is a link to the ArcGIS Online group called *Python Scripting for ArcGIS Pro – 2024 (Esri Press)*. The data for Exercise 3 is posted as a zip file called PythonScripting_Ex03_Data.zip. Download this file and extract it to a folder of your choice. The instructions use a folder called C:\PythonPro, but you can use a different folder provided you update any paths.

Create a new project

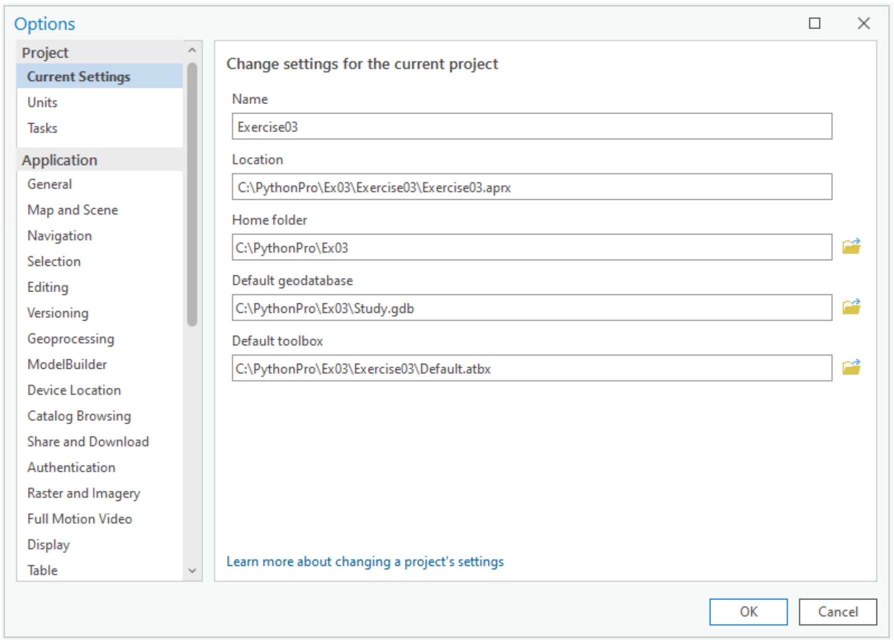
Prior to running geoprocessing tools, it is helpful to save your project and set project options for saving your work.

1. **Start ArcGIS Pro and click Start Without a Template to create a new blank project.**

2. **On the Project tab, click Save Project As to save your project as C:\PythonPro\Ex03\Exercise03.aprx.**

3. **Still on the Project tab, click Options and specify the following:**

- **Home folder: (C:\PythonPro\Ex03)**
- **Default geodatabase: (C:\PythonPro\Ex03\Study.gdb)**
- **Default toolbox: (C:\PythonPro\Ex03\Exercise03\Default.atbx)**



Change settings for the current project

Name: Exercise03

Location: C:\PythonPro\Ex03\Exercise03\Exercise03.aprx

Home folder: C:\PythonPro\Ex03

Default geodatabase: C:\PythonPro\Ex03\Study.gdb

Default toolbox: C:\PythonPro\Ex03\Exercise03\Default.atbx

Learn more about changing a project's settings

By setting the default geodatabase, the outputs of geoprocessing tools will be saved to the C:\PythonPro\Ex03\Study.gdb geodatabase.

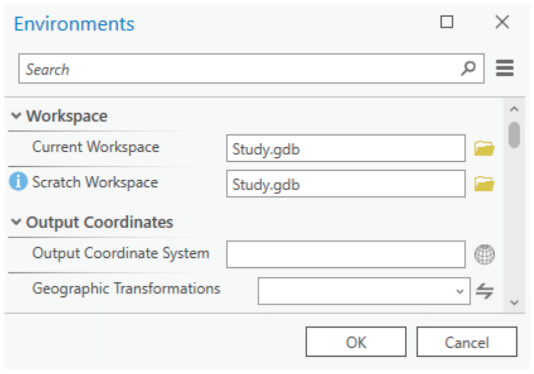
4. **Click OK and close the project options.**

Even though the home folder and default geodatabase have been set for the project, it is good practice to set workspaces for geoprocessing.

5. **On the Analysis tab, click Environments.**

6. **Set the current workspace and the scratch workspace to**

   C:\PythonPro\Ex03\Study.gdb.



Workspace

Current Workspace	Study.gdb
Scratch Workspace	Study.gdb

Output Coordinates

Output Coordinate System	
Geographic Transformations	

7. **Click OK.**

8. **Save your project.**

Note: The current workspace and scratch workspace default to the project geodatabase, so it is not required to set these workspaces. However, you can set these workspaces to a different folder or geodatabase as required during your workflow, and it is good to know how to do this.

Now you are ready to explore data and run tools.

Examine toolboxes and tools

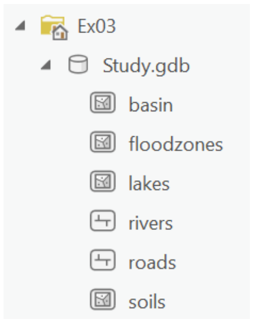
Before starting to work with the exercise data, you will preview the data in ArcGIS Pro.

1. **Make sure the Catalog pane is visible by clicking Catalog Pane on the View tab. Dock the Catalog pane to the right side of the ArcGIS Pro interface.**

2. **Create a new folder connection to the location of the exercise data by right-clicking Folders > Add Folder Connection and navigating to the C:\PythonPro\Ex03 folder.**

Note: With the home folder set for the project, this folder automatically shows up as a folder connection. However, if you are using a different home folder, you can connect to a different folder where the exercise data is located.

3. **Examine the contents of this folder.**



You will use all these feature classes in this exercise.

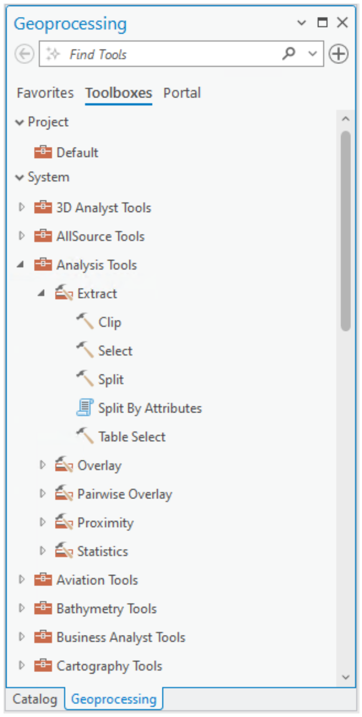
ArcGIS Pro contains hundreds of tools, organized in toolboxes and toolsets.

4. **On the Analysis tab, click Tools to bring up the Geoprocessing pane.**

5. **Click Toolboxes.**

The Toolboxes panel shows all the system toolboxes in ArcGIS Pro and allows you to browse through an organized list of all the tools.

6. **Expand the Analysis toolbox and then the Extract toolset to see the list of tools inside this toolset.**

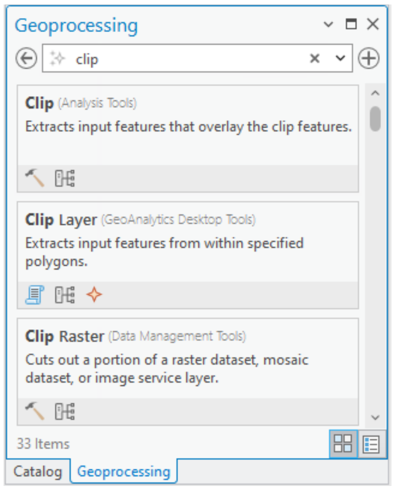


Finding the tool(s) you want can be a bit cumbersome if you are not sure where to look, but with some repetition you will start to remember where the tools you use most frequently are located.

You can also search for tools by name.

7. **In the search box at the top, type clip.**

This brings up tools with "clip" in their name or description, as well as tools with related functionality.


Geoprocessing
clip
Clip (Analysis Tools)
Extracts input features that overlay the clip features.
Clip Layer (GeoAnalytics Desktop Tools)
Extracts input features from within specified polygons.
Clip Raster (Data Management Tools)
Cuts out a portion of a raster dataset, mosaic dataset, or image service layer.
33 Items
Catalog
Geoprocessing

In this case, notice that there is a Clip tool for vector data such as polylines and polygons in the Analysis toolbox and a Clip Raster tool for raster data in the Data Management toolbox.

Double-clicking the tool name opens the tool dialog box, regardless of whether you found the tool by browsing through the list of tools or by searching.

Run a tool

Next, you will add some data so you can start using some of the geoprocessing tools.

1. **Navigate to the C:\PythonPro\Ex03 folder in the Catalog pane.**
2. **Expand the Study.gdb file geodatabase to view the feature classes.**
3. **While holding Ctrl, select the basin and soil feature classes.**
4. **Right-click one of them and click Add To New > Map. You can remove any basemap layers.**

Notice that the extent of the soils feature class is much larger than the basin feature class. You will use the Clip tool to reduce the extent of the soils feature class to match the basin feature class.

5. **Open the Geoprocessing pane if it is not already open (Analysis > Tools).**

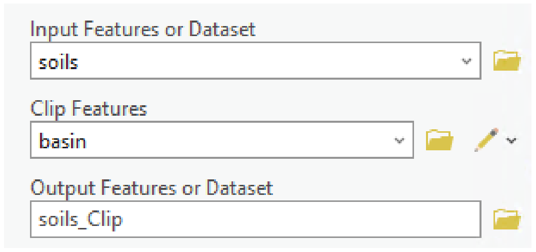
6. **Locate the Clip tool in the Geoprocessing pane, either by browsing under Toolboxes > Analysis Tools > Extract, or by searching for it by name.**

7. **Double-click the Clip tool to open the tool dialog box.**

   Next, you will specify the parameters of the Clip tool.

8. **For Input Features, select the soils feature layer from the drop-down list.**

9. **For Clip Features, select the basin feature class from the drop-down list.**



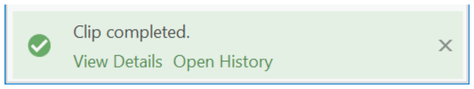
Input Features or Dataset: soils
Clip Features: basin
Output Features or Dataset: soils_Clip

Notice that the output feature class is automatically populated. The path is determined by the project settings (C:\PythonPro\Ex03\Study.gdb), and the file name is based on the inputs and the name of the tool (i.e., soils_Clip). You can use the Browse button to navigate to a different path and output file name. You can also change the path and output file name by typing it in the text box.

Once all required parameters have been specified, you can run the tool.

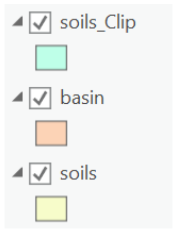
10. **Click Run to execute the tool. Do not close the tool dialog box.**

A progressor appears to show that the tool is running. Once tool execution is completed, a message appears at the bottom of the tool dialog box.



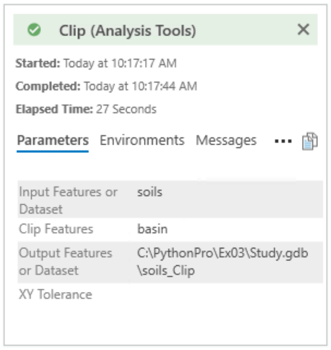
Clip completed.
View Details Open History

The result of running the tool is a new feature class called soils_Clip, which is added as a feature layer to the contents of the active map



You can now explore the soils_Clip feature class to confirm the result. You can also review the execution of the tool by opening the tool messages.

11. **Click on the View Details link and examine its contents.**


Clip (Analysis Tools)

Started: Today at 10:17:17 AM

Completed: Today at 10:17:44 AM

Elapsed Time: 27 Seconds

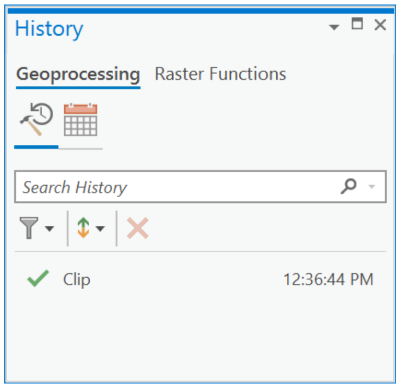
Input Features or Dataset	soils
Clip Features	basin
Output Features or Dataset	C:\PythonPro\Ex03\Study.gdb \soils_Clip
XY Tolerance	

The details include information on the time of tool execution, the tool parameters, any errors or warnings, environments used, and geoprocessing messages.

When a tool is run, it is also added to the history.

12. Close the tool message. On the Analysis tab, click History.

The History pane opens with a list of geoprocessing tools that were run.



Clip 12:36:44 PM

The History pane records the geoprocessing operations, including all the tool parameters, environment settings, and messages. You can double-click on an entry to open the tool dialog box and review how the tool was run.

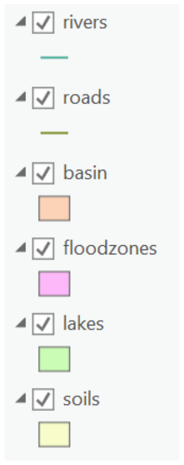
13. **Close the History pane.**

14. **In the contents of the active map, right-click the soils_Clip feature layer, and click Remove.**

Conduct batch processing

You will next use batch processing to clip the other layers.

1. **Add the other feature classes in the Study.gdb geodatabase to the current active map: floodzones, lakes, rivers, and roads.**



The exact order of the feature layers under Contents is not relevant.

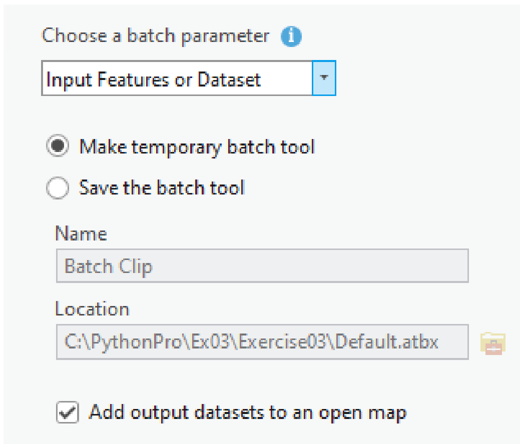
2. **Open the Geoprocessing pane if it is not already open (Analysis > Tools).**

3. **Locate the Clip tool.**

4. **Right-click on the Clip tool and click Batch.**

   The Batch Clip tool opens.

5. **Select Input Features as the batch parameter. You can leave the other parameters to their defaults.**



Choose a batch parameter

Input Features or Dataset

(●) Make temporary batch tool

( ) Save the batch tool

Name

Batch Clip

Location

C:\PythonPro\Ex03\Exercise03\Default.atbx

[✓] Add output datasets to an open map

6. **Click Next.**

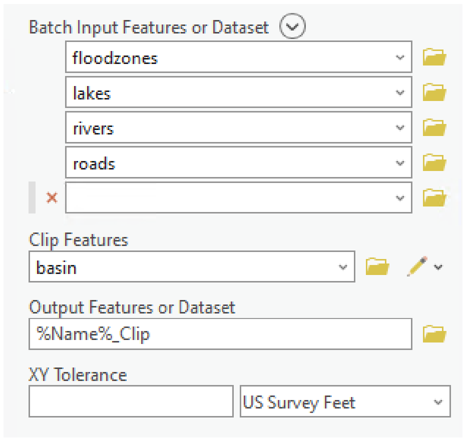
   The next panel allows you to select multiple data layers for the input features.

7. **Using the drop-down options, select the following feature classes as input features: floodzones, lakes, rivers, and roads.**

8. **For the clip features, select basin.**

9. **For the output feature class, enter %Name%_Clip.**

   The full path of the output feature class is C:\PythonPro\Ex03\Study.gdb\%Name%_Clip, but the geodatabase is already specified by default.


Batch Input Features or Dataset
floodzones
lakes
rivers
roads

Clip Features
basin

Output Features or Dataset
%Name%_Clip

XY Tolerance
US Survey Feet

10. Click Run to execute the Batch Clip tool.

Once the tool run is completed, the four output feature classes are added to the contents of the active map.

11. From the contents of the active map, remove the four newly added feature classes from the contents of the active map.

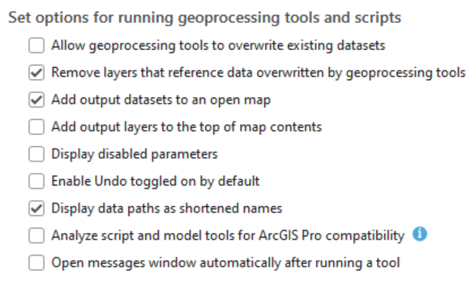
Modify geoprocessing options

Sometimes when you run geoprocessing operations, you encounter a situation in which you might want to overwrite the existing data. For example, you may realize you made a mistake and want to run a tool again using the same name you already used for an output feature class. This is a common scenario when running models and scripts. Overwriting existing datasets is a setting you can change.

First, you will examine the geoprocessing option to prevent overwriting files.

1. **On the Analysis tab, click on the Geoprocessing Options icon, which is in the lower-right corner of the Geoprocessing group. Alternatively, you can click Project > Options > Geoprocessing.**

2. **Under Geoprocessing Options, uncheck the first option, "Allow geoprocessing tools to overwrite existing datasets."**



Set options for running geoprocessing tools and scripts

- [ ] Allow geoprocessing tools to overwrite existing datasets
- [x] Remove layers that reference data overwritten by geoprocessing tools
- [x] Add output datasets to an open map
- [ ] Add output layers to the top of map contents
- [ ] Display disabled parameters
- [ ] Enable Undo toggled on by default
- [x] Display data paths as shortened names
- [ ] Analyze script and model tools for ArcGIS Pro compatibility
- [ ] Open messages window automatically after running a tool

3. **Click OK to close Geoprocessing Options.**

Next, consider the following example.

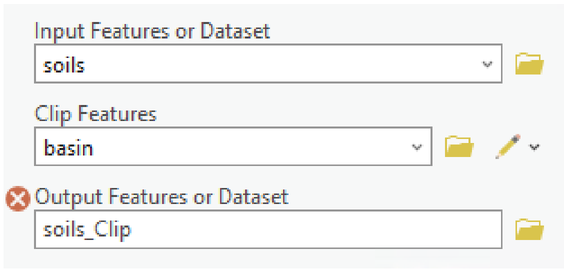
4. **Open the Clip tool dialog box.**

5. **For Input Features, select soils, and for Clip Features, select basin.**

Notice that, by default, in the naming of the output feature class, ArcGIS Pro recognizes that the soils_Clip feature class already exists and therefore the name soils_Clip1 is entered.

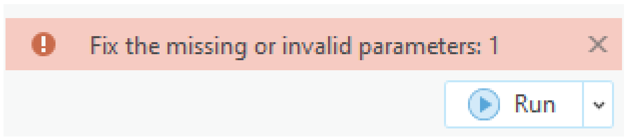
6. **Change the name of Output Feature Class to soils_Clip.**

Since this name already exists, an error icon appears indicating that the tool will not run.


Input Features or Dataset
soils
Clip Features
basin
Output Features or Dataset
soils_Clip

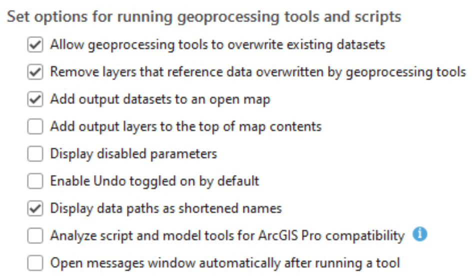
7. **Click the Run button.**

The tool does not execute, and a message comes up that the "Fix the missing or invalid parameters: 1".


Fix the missing or invalid parameters: 1

Next, modify the geoprocessing option to allow for overwriting files.

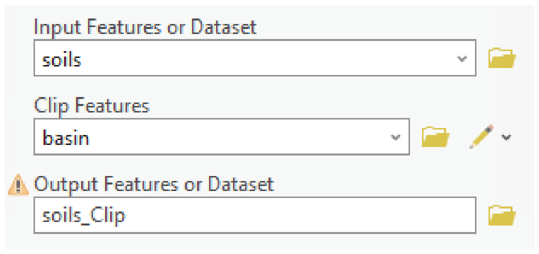
Return to Geoprocessing Options, and check the first option, "Allow geoprocessing tools to overwrite existing datasets."


Set options for running geoprocessing tools and scripts

- [✓] Allow geoprocessing tools to overwrite existing datasets
- [✓] Remove layers that reference data overwritten by geoprocessing tools
- [✓] Add output datasets to an open map
- [ ] Add output layers to the top of map contents
- [ ] Display disabled parameters
- [ ] Enable Undo toggled on by default
- [✓] Display data paths as shortened names
- [ ] Analyze script and model tools for ArcGIS Pro compatibility
- [ ] Open messages window automatically after running a tool

8. **Click OK to close Geoprocessing Options.**

On the Clip tool dialog box, the error icon has become a warning icon. The warning message says that the output feature class soils_Clip already exists, but this will not prevent the tool from running.


Input Features or Dataset
soils
Clip Features
basin
Output Features or Dataset
soils_Clip

Although the changes to the geoprocessing options take effect immediately, you may need to retype the name of the output feature class on the Clip tool dialog box for the error icon to disappear.

9. **Close the Clip tool.**

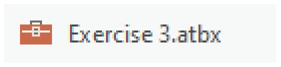
Explore models and ModelBuilder

Models are one way to create a sequence of geoprocessing operations in ArcGIS Pro. Like tools, models are organized in toolboxes within the geoprocessing environment. Before creating a model then, you must create a toolbox to store it.

1. **In the Catalog pane, right-click on the C:\PythonPro\Ex03 folder, and click New > Toolbox.**

   A new toolbox is created with a default name.

2. **Rename the toolbox Exercise 3.atbx.**

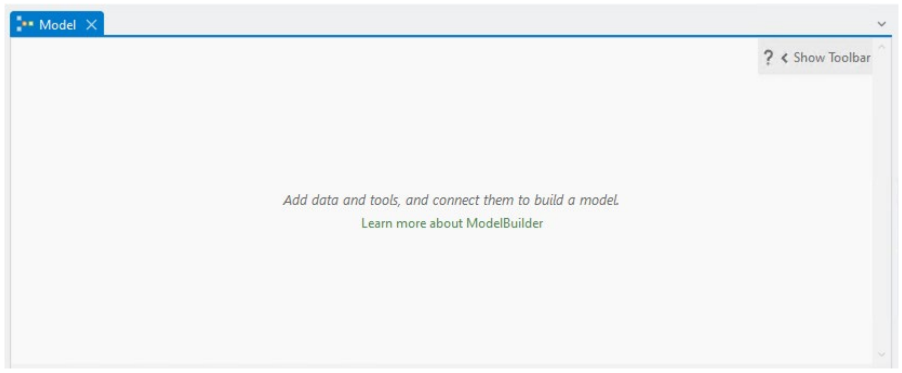


Note: You can create a custom toolbox in any folder or in a geodatabase. The key is that a model must be saved within a toolbox, so you must create a toolbox first before creating the model. You should already have another toolbox called Default.atbx, but you are using a new custom toolbox for this exercise.

Now you are ready to create a model.

3. **Right-click on the Exercise 3 toolbox and click New > Model.**

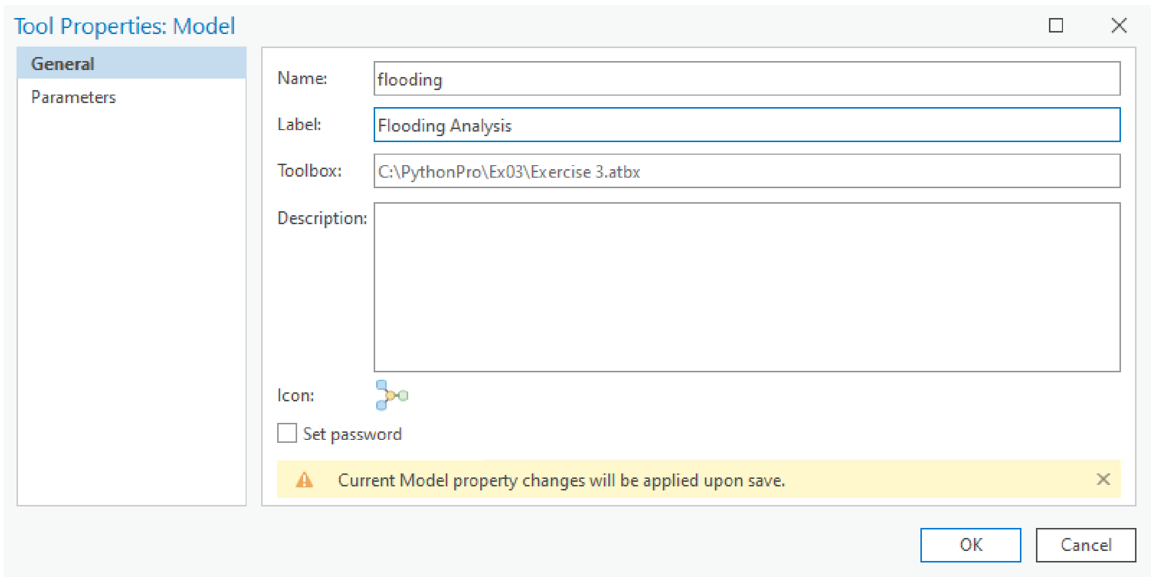
   A new blank model opens. You can resize the model.


Model
? Show Toolbar
Add data and tools, and connect them to build a model.
Learn more about ModelBuilder

Next, give the model a name and display label.

4. **On the ModelBuilder tab, in the Model group, click Properties.**
   - **For Name, type flooding.**
   - **For Label, type Flooding Analysis.**

The name indicates the actual name of the model. This name is used when calling the model from other tools in ArcGIS Pro. The name cannot contain any spaces. The label box indicates the label that will appear next to the model tool in the toolbox. The label can contain spaces.



Name: flooding

Label: Flooding Analysis

Toolbox: C:\PythonPro\Ex03\Exercise 3.atbx

Description:

Icon:

Set password

Current Model property changes will be applied upon save.

5. **Click OK to close the Properties dialog box.**

6. **On the ModelBuilder tab, click Save.**

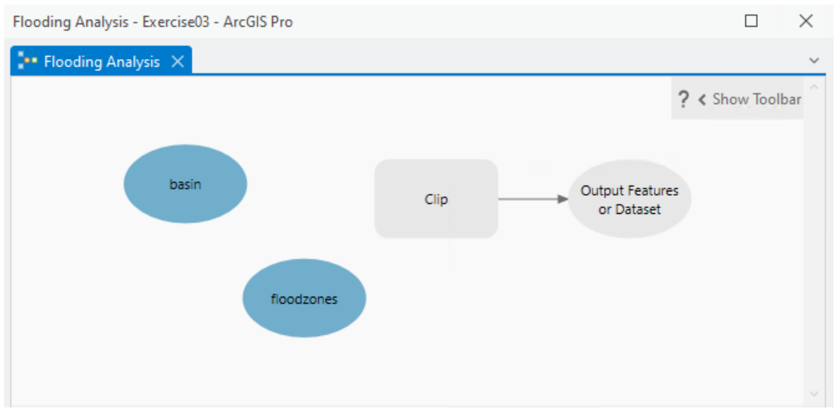
Both the label of the model in the Exercise 3 toolbox and the heading of the model itself are updated. You are now ready to start adding elements to the model.

7. **Drag the basin and floodzones feature layers from the contents of the active map into the model.**

8. **On the ModelBuilder tab, in the Insert group, click Tools to bring up the Geoprocessing pane.**

9. **Search for the Clip tool and drag this tool into the model.**

   The result so far looks like the figure.


Flooding Analysis - Exercise03 - ArcGIS Pro
Flooding Analysis
Show Toolbar
basin
floodzones
Clip
Output Features or Dataset

As the layers are added, their oval symbols are given a fill color (blue) because the file name for these data variables is specified. When the Clip tool is added, its rectangular symbol remains gray because the tool's parameters have not been specified yet. Gray symbols indicate that a model is not ready to run. In addition, by its nature, the Clip tool produces an output feature class, so this data variable is automatically added to the model, even though it is not pointing to an output feature class yet.

Connectors also need to be added to make the model ready to run. In this example, the layers basin and floodzones must be connected to the Clip tool.

10. **Double-click the Clip tool inside the model.**

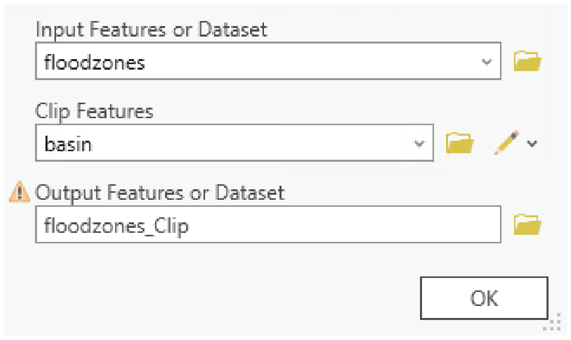
    You can now specify the tool's parameters as you normally would for a tool.

11. **For the Input Features, select floodzones, but make sure to select the existing Model Variable, not the data layer of the same name in the active map.**

12. **For Clip Features, select basin, but make sure to select from the existing Model Variable, not the data layer of the same name in the active map.**

Layer names will automatically have a colon and number appended and may vary from those shown in the figures.

13. **Specify the Output Feature class as floodzones_Clip.**



Input Features or Dataset
floodzones

Clip Features
basin

Output Features or Dataset
floodzones_Clip

Notice how the dialog look like the Clip tool dialog box but is not part of the Geoprocessing pane, and there is no Run button. This is because the tool is part of the model and does not run on its own.

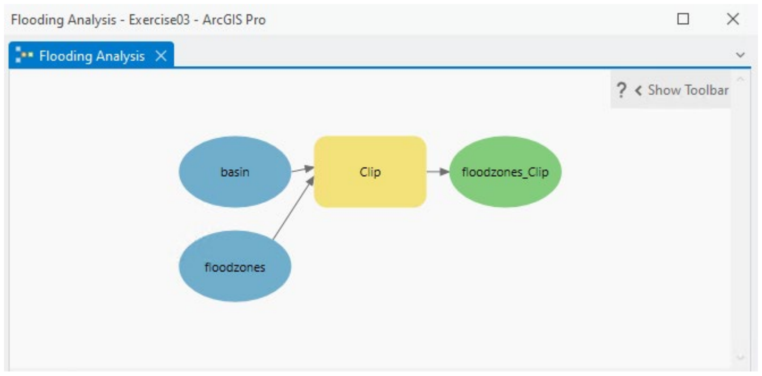
14. **Click OK to close the tool dialog box.**

When you close the tool dialog box, the tool does not run as it would if you were using the tool in stand-alone mode outside a model. Instead, with the tool parameters specified, the appropriate connectors are created in the model. As a result, the symbol for the Clip tool in the model is now given a fill color (yellow), and so is the symbol for the output data variable (green). When all parameters are specified and all elements in the model have a fill color, the model is ready to run.

Before proceeding, try cleaning up the look of the model a bit.

15. On the ModelBuilder tab, click Auto Layout and then click Fit To Window.

This organizes the model elements into a consistent pattern. Although the layout has no effect on running the model, it makes it easier to follow the workflow in your model. After every few modifications or additions to a model, it is a good idea to use Auto Layout and Fit To Window to reorganize the model elements.



Run your model

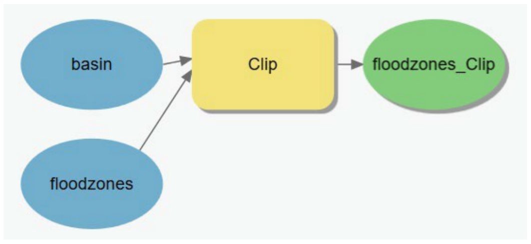
Now your model is ready to run. There are several ways to run a model. For example, you can run the entire model, or you can run individual tools in the model. Since there is only one tool in the current model, there is no difference between these options, but this becomes more relevant when your models become more complex.

1. On the ModelBuilder tab, click the Run button to run the entire model.

A model progress dialog box appears that shows the progress and time elapsed in running the tools in the model. The messages are like those that result from running individual geoprocessing tools.

2. **Close the model progress dialog box.**

When the model run is completed, the model elements (other than the input datasets) have a drop shadow to indicate that the tool has been run and the output datasets are created.



Note: Since you are running the model from within ModelBuilder, the execution of the model is not recorded in the History pane. If you saved your model and closed it, you could run it as a tool, and then the tool execution would be recorded in the History pane.

Although the output shapefile floodzones_Clip feature class was created, it has not been added to the Contents of the active map. By default, ModelBuilder assumes the model outputs represent intermediate data.

3. **Right-click the floodzones_Clip element in the model and click Add To Display.**

You can now confirm that the floodzones_Clip feature layer has been added to the active map and that it represents the clipped version of the floodzones layer.

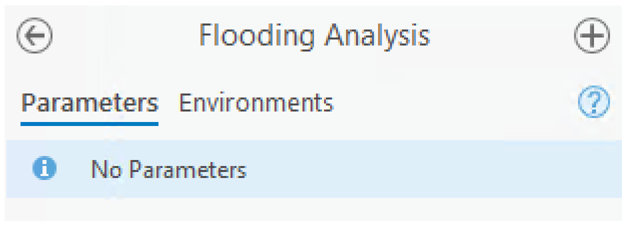
4. **On the ModelBuilder tab, click Save.**

5. **Close the model by clicking on the X on the window tab.**

Next, you will open the model as a tool.

6. **In the Exercise 3 toolbox, double-click the Flooding Analysis model.**

The Flooding Analysis tool dialog box opens with the discouraging message, "No parameters." What happened? Where is the model?



No Parameters

Models are tools, so by creating a model, you automatically create a tool. And tools have tool dialog boxes to specify parameters. However, in the ModelBuilder interface, you created only the model elements without indicating which elements should become parameters. In other words, the model is not yet fully ready to be used as a tool in which the user could specify the tool parameters.

Note: Creating tool parameters is not covered here.

So instead of running the model as a tool, you are going back into the model itself.

7. **Close the Geoprocessing pane.**

8. **In the Exercise 3 toolbox, right-click on the Flooding Analysis model, and click Edit.**

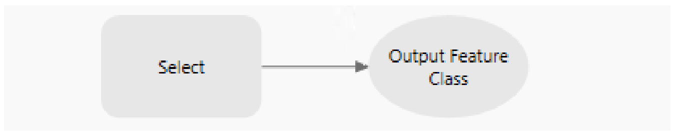
This brings you back into the ModelBuilder interface. Notice that the Clip tool and the output feature class still have drop shadows. This indicates the model has already been run, and it remembers its processing state.

Next, you can add another step to the model. You may have noticed that the polygons in the floodzones layer cover the entire study area. This is how traditional flood maps are organized: polygons cover the entire study area but are coded as being inside or outside particular flood zone categories. You will add a tool to select just the polygons of interest.

9. **On the ModelBuilder tab, click Tools to bring up the Geoprocessing pane.**

10. **Search for the Select tool.**

11. **Drag the Select tool into the model.**



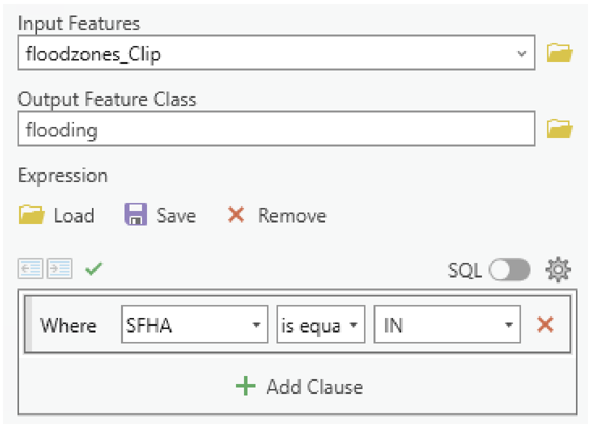
12. **Double-click the Select tool in the model.**

13. **For the input features, select the floodzones_Clip model variable.**

14. **Specify the output feature class as flooding.**

15. **Under Expression, create the following expression: Where SFHA is equal to IN. Alternatively, you can use the SQL expression SFHA = 'IN'.**

SFHA stands for Special Flood Hazard Area. The areas coded as IN are the flood hazard areas of interest. The Select tool dialog box should now look like the example in the figure.

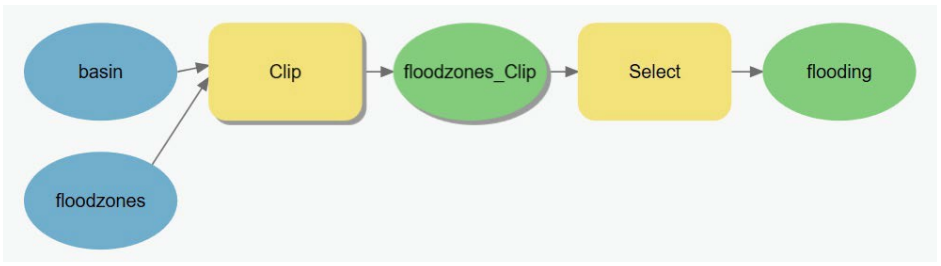

Input Features
floodzones_Clip
Output Feature Class
flooding
Expression

Where SFHA is equa IN

16. Click OK to close the tool dialog box.

17. On the ModelBuilder tab, click the Auto Layout and Fit To Window buttons.

Your model should look like the example in the figure and is now ready to run.



18. Right-click the floodzones_Clip element and turn off Add To Display.

19. Right-click the flooding element and turn on Add To Display.

20. On the ModelBuilder tab, click the Run button to run the model.

Because the Clip tool was run previously, only the Select tool needs to be run for the model run to be complete.

21. When the model run is completed, close the model progress dialog box.

22. Save and close the model.

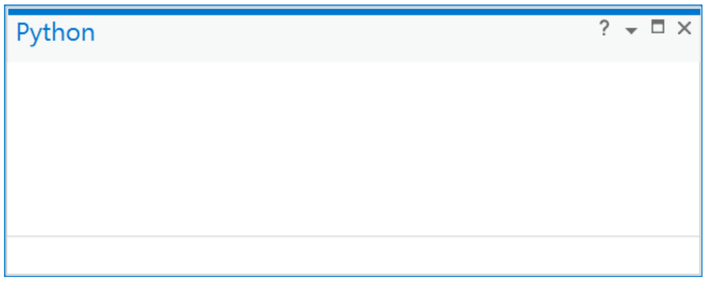
The result of the model is added to the active map. The figure shows the original basin feature layer (in beige) and the flooding feature layer (in blue).

Use scripting for geoprocessing

Scripting represents another way to carry out geoprocessing operations in ArcGIS Pro. A basic Python script is like a model, except that it uses code instead of the visual programming language of ModelBuilder. Python is the preferred scripting language for working with ArcGIS Pro, and Python code can be run directly in the Python window.

1. **On the Analysis tab, click the Python button > Python Window.**

This opens the Python window. The lower section of the Python window is called the "prompt," and this is where you type your code.



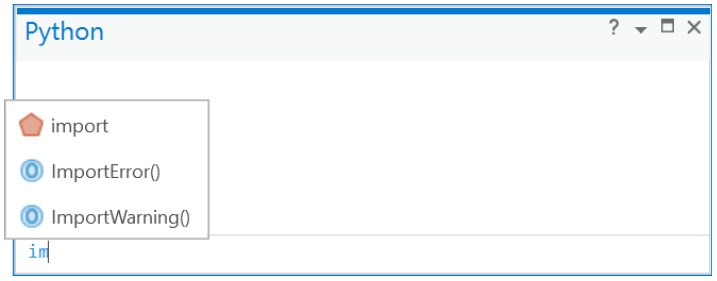
To run geoprocessing tools from Python, you first must import the ArcPy site package, which you'll do next. Importing the ArcPy site package makes all the tools in the geoprocessing framework in ArcGIS available for Python scripting.

2. **Enter the following code at the prompt:**

```
import arcpy
```

Within the Python window, ArcPy is automatically referenced, so the `import arcpy` statement is in fact not necessary to use the geoprocessing tools from within ArcGIS Pro. However, any Python code and stand-alone scripts outside ArcGIS Pro that use the functionality of ArcGIS Pro need the `import arcpy` statement.

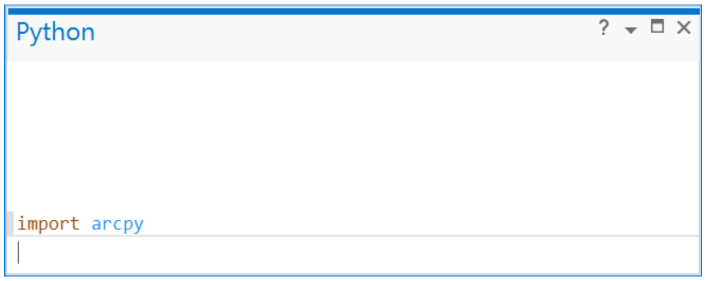
Notice that the Python window provides prompts to assist in writing proper syntax. For example, when you start typing the letters `im`, a list is provided of the code elements that start with these letters. You can select the option you want by clicking on it, or by using the arrow keys and pressing Enter.



Instead of using the prompts, you can also just keep typing. Even if you don't use the prompts, they can be useful as reminders of the proper syntax.

3. **At the end of your first line of code (`import arcpy`), press Enter.**

Pressing Enter runs the line of code. The code is copied to the top section of the Python window called the "transcript," any results are printed below (not the case here), and the prompt is empty again to write the next line of code.



```
import arcpy
```

Remember that Python is an interpreted language, which means that in the Python window, a single line of code is run as soon as you press Enter.

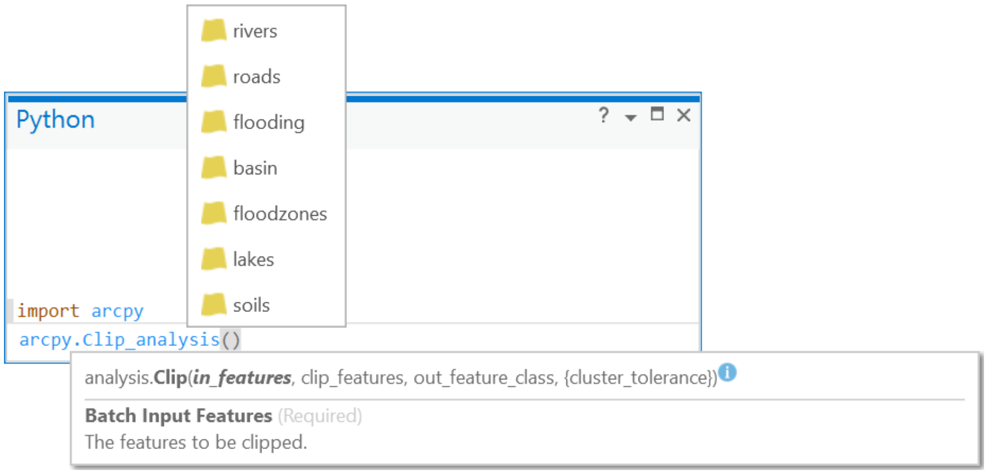
Now you are ready to run a geoprocessing tool.

4. **Enter the following code but do not press Enter yet:**

```
arcpy.Clip_analysis
```

This code calls the Clip tool. Python is case sensitive (for the most part), so make sure to type "Clip," not "clip." Calling the Clip tool is equivalent to opening the tool dialog box. The next step is to specify the tool's parameters, as if you were filling out the tool dialog box. As you start typing, the prompts will be helpful to ensure that you use the proper syntax.

When you type the opening parenthesis after Clip_analysis, a few things happen. First, a closing parenthesis is added automatically. Second, the syntax of the Clip tool shows as a pop-up window. And third, a list of feature layers in the active map appears for you to choose from.



```
import arcpy
arcpy.Clip_analysis()
```

analysis.**Clip**(***in_features***, clip_features, out_feature_class, {cluster_tolerance})

Batch Input Features (Required)
The features to be clipped.

5. Complete the following line of code:

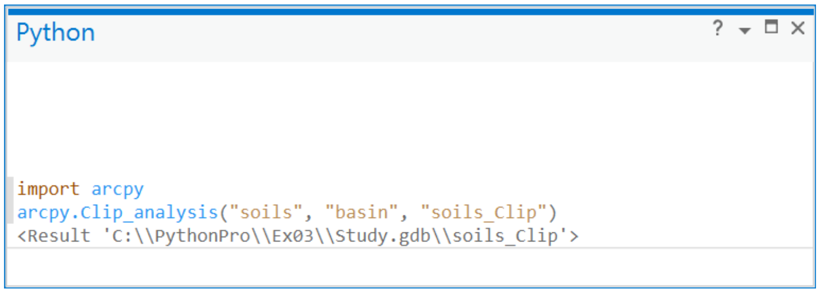
```
arcpy.Clip_analysis("soils", "basin", "soils_Clip")
```

As you are typing this code, you can select the feature layers from the list, or type the name yourself. Note that Python uses both single quotes (`' '`) and double quotes (`" "`), and these can be used interchangeably provided they are used consistently. So `'soils'` is the same as `"soils"`, but `'soils"` is incorrect. Also, as you are typing, the active parameter is shown in bold in the syntax.

The required tool parameters are listed inside the parentheses. The optional XY Tolerance is not included, which means that, just like a tool dialog box, the default value will be used.

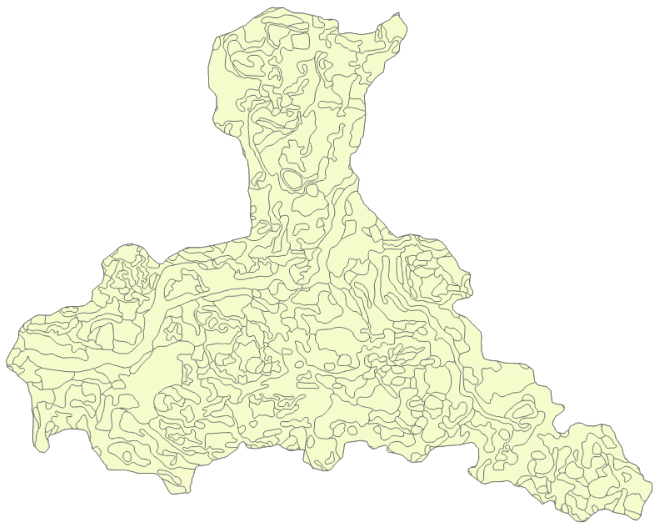
6. Press Enter to run the line of code.

When the line of code executes, it is copied to the transcript, and the result appears below. In this case, the result is a refence to the new feature class that was created: C:\PythonPro\Ex03\Study.gdb\soil_Clip.



```
import arcpy
arcpy.Clip_analysis("soils", "basin", "soils_Clip")
<Result 'C:\\PythonPro\\Ex03\\Study.gdb\\soils_Clip'>
```

The output feature class is also added to the active map.



Later exercises cover running tools using Python in more detail. For now, it is important to remember that you can run geoprocessing tools directly from the Python window. Lines of code are run immediately, and the Python window is highly integrated with the ArcGIS Pro interface.

In addition to working with Python code in the Python window, you can write and run code in a Python editor. You will use IDLE in the next set of steps to create a simple script.

7. **Start IDLE.**

8. **Click File > New to open a new script window.**

9. **Click File > Save As, and save the script as my_clip.py to the C:\PythonPro\Ex03 folder.**

Python script files are simply text files with a .py file extension. Python code in a script is not run until the script is run. You can therefore enter multiple lines of code before you run the script.

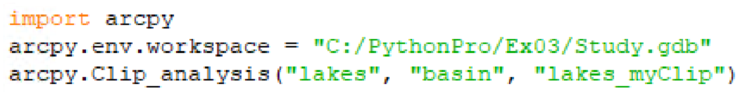
10. Enter the following code in the my_clip.py script window:

```
import arcpy

arcpy.env.workspace = "C:/PythonPro/Ex03/Study.gdb"

arcpy.Clip_analysis("lakes", "basin", "lakes_myClip")
```

The line of code that starts with `arcpy.env.workspace` is equivalent to setting the workspace in the Environment dialog box. This syntax is covered in more detail in the following exercises.

Note: The workspace must be set in the Python script, regardless of any settings in the ArcGIS Pro application. A stand-alone Python script does not inherit the environment settings in ArcGIS Pro.

Your script window should now look like the example in the figure.



```
import arcpy
arcpy.env.workspace = "C:/PythonPro/Ex03/Study.gdb"
arcpy.Clip_analysis("lakes", "basin", "lakes_myClip")
```

11. Click File > Save to save the script.

12. Click Run > Run Module to run the script.

Upon execution of the script, the Python Shell opens with a new prompt (>>>), but nothing else happens unless there was an error. Because you are running a stand-alone script, the output is not automatically added to a project in ArcGIS Pro. In fact, ArcGIS Pro doesn't need to be open for a script to run.

13. **In ArcGIS Pro, in the Catalog pane, navigate to the geodatabase C:\PythonPro\Ex0s\Study.gdb, and confirm that the lakes_myClip feature class has been created.**

Note: To view the results, you may need to refresh the Catalog view of the data. Right-click on the geodatabase in the Catalog pane, and click Refresh.

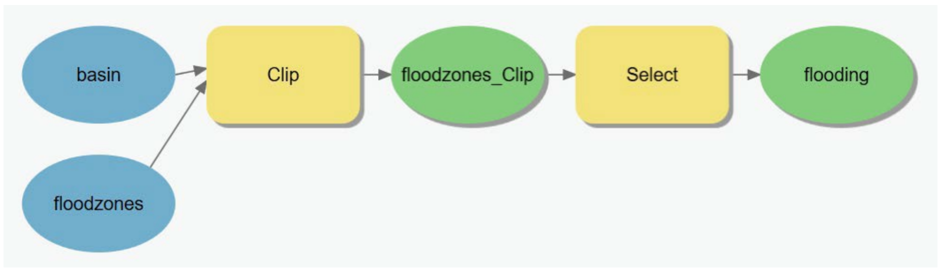
The Python script accomplishes the same task as the Python code in the Python window: in both places, the Clip tool runs and creates a new feature class. However, there are a few differences:

- The Python window inherits the environments of the geoprocessing framework in ArcGIS Pro, but in a stand-alone Python script, the environments must be set.
- A stand-alone Python script can run without ArcGIS Pro being open, whereas the Python window is an integral part of ArcGIS Pro.
- Code in the Python window is run line by line, as in any interactive interpreter, whereas the stand-alone Python script is run in its entirety.

These differences between running Python scripts and running code in the Python window or other interactive interpreters are revisited in later exercises.

Convert a model to a script

Another way to learn about Python scripting is to export a model to a script. This allows you to see what a logical sequence of geoprocessing operations looks like in Python. For starters, you can revisit the Flooding Analysis model created earlier, as shown in the example in the figure.



You will now convert this model to a script.

1. **In the Catalog pane, navigate to the C:\PythonPro\Ex03 folder, expand the Exercise 3 toolbox, right-click on the Flooding Analysis model, and click Edit.**

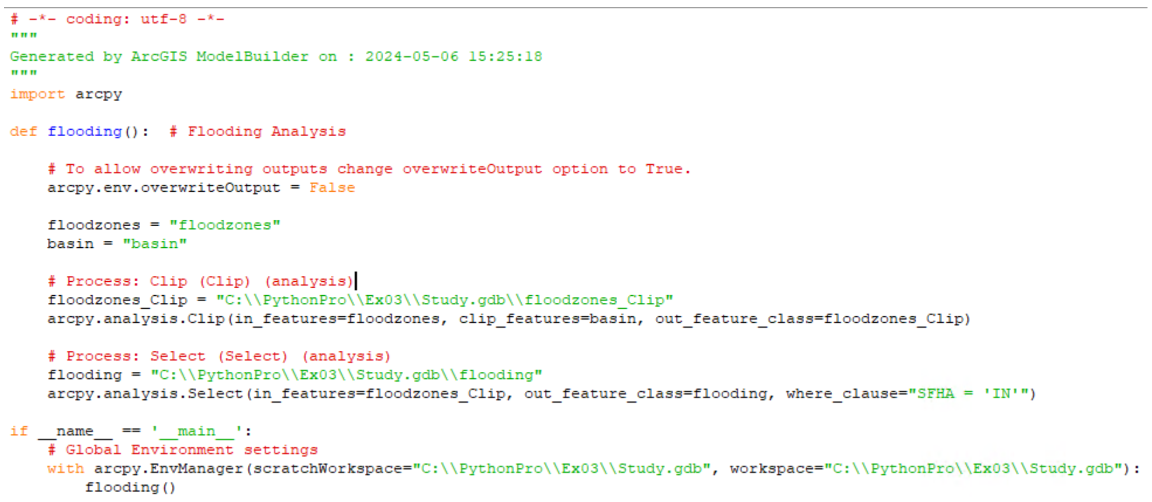
2. **On the ModelBuilder tab, in the Model Group, click Export > Export to Python File.**

3. **In the Export Script dialog box, navigate to the C:\PythonPro\Ex03 folder, and save the script as flooding.py.**

4. **Return to IDLE.**

5. **Click File > Open, navigate to the C:\PythonPro\Ex03 folder, click on the flooding.py file, and click OK.**

6. **Review the contents of the script.**



```python
# -*- coding: utf-8 -*-
"""
Generated by ArcGIS ModelBuilder on : 2024-05-06 15:25:18
"""
import arcpy

def flooding():  # Flooding Analysis

    # To allow overwriting outputs change overwriteOutput option to True.
    arcpy.env.overwriteOutput = False

    floodzones = "floodzones"
    basin = "basin"

    # Process: Clip (Clip) (analysis)
    floodzones_Clip = "C:\\PythonPro\\Ex03\\Study.gdb\\floodzones_Clip"
    arcpy.analysis.Clip(in_features=floodzones, clip_features=basin, out_feature_class=floodzones_Clip)

    # Process: Select (Select) (analysis)
    flooding = "C:\\PythonPro\\Ex03\\Study.gdb\\flooding"
    arcpy.analysis.Select(in_features=floodzones_Clip, out_feature_class=flooding, where_clause="SFHA = 'IN'")

if __name__ == '__main__':
    # Global Environment settings
    with arcpy.EnvManager(scratchWorkspace="C:\\PythonPro\\Ex03\\Study.gdb", workspace="C:\\PythonPro\\Ex03\\Study.gdb"):
        flooding()
```

In the figure, long lines of code in the original script have been broken up into multiple lines to fit the display, but this has no impact on the code itself.

Don't worry for now about being able to understand everything in the script. However, you should be able to recognize each of the model elements, including the data variables floodzones and basin, as well as the Clip and Select tools.

7. **Close IDLE.**

8. **Close ArcGIS Pro.**

In this exercise, you have learned how to run geoprocessing tools and control how they are run by using tool parameters and environment settings. You have created a model using the ModelBuilder interface. You have also run Python code as code snippets in the Python window and as a stand-alone Python script

End of Exercise 3.