



# Creating Geoprocessing Services Tutorial



Copyright © 1995-2010 Esri All rights reserved.

# Table of Contents

Guide to the geoprocessing service examples . . . . .	3
GP service step by step: Buffer points . . . . .	5
GP service example: Buffer features . . . . .	16
GP service step by step: Watershed . . . . .	18
GP service example: Stream network . . . . .	30
GP Service example: More Stream network . . . . .	40
GP Service example: Clip and ship . . . . .	46
GP Service example: Data on demand . . . . .	53
GP service example: Selecting data . . . . .	57
GP service example: Drive-time polygons . . . . .	73
GP service example: Shortest route on a street network . . . . .	85
GP service example: Finding nearby features over a street network . . . . .	99

# Guide to the geoprocessing service examples

Each topic in this book describes an example geoprocessing service. These example services were carefully chosen to demonstrate the following:

**Complexity:**  
Beginner  
**Data Requirement:**  
ArcGIS Tutorial Data Setup

- Useful GIS functionality such as buffering features, spatial selection of features, selecting features, finding nearby features using network distance, and packaging databases for sending back to the client
- Common geoprocessing service design patterns, such as the use of feature and record sets and tool layers, using layers from a source map document, and drawing results with a result map service
- Preprocessing data to make fast and efficient services
- Useful tips and tricks

## Each topic has a corresponding folder

Each topic in this book has a corresponding folder that contains data, toolboxes, and map documents of the completed example. These folders are in the GP Service Examples folder found in the ArcTutor directory that is installed with the ArcGIS tutorial data. The tutorial data is available on the ArcGIS installation media. If the tutorial data has been installed on your system, look for it in `C:\arcgis\ArcTutor` (the default installation location).


At the top of the page of each topic, you will find the name of the corresponding folder.

Typically, you should copy the corresponding folder from the ArcTutor directory to another folder before making changes or publishing the services.

## Types of examples

There are two types of example topics in this book, distinguished by their titles:

- **GP service step by step:** These topics are step-by-step guides to creating a service. They assume you have cursory knowledge of geoprocessing. The idea behind these step-by-step examples is to show you how everything in the corresponding folder was created. When you complete the steps, you will have created a duplicate of the corresponding folder in the tutorial directory. You can use the corresponding folder to check your work.
- **GP service example:** These topics assume you are familiar with geoprocessing and ArcGIS Server and you do not need step-by-step instructions on how to create models and tool layers or publish services. Rather, they focus on particular aspects of the service, providing step-by-step instructions only when more advanced concepts are introduced.

 **Note:** While authoring Geoprocessing Services in ArcMap it is advised to disable background processing.  
Learn more about [background processing](#)

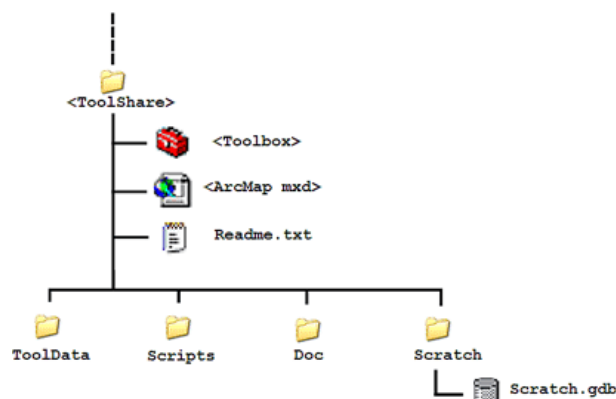
Example	Description
<a href="#">Buffer points (step by step)</a>	A simple model to buffer point features
<a href="#">Buffer features</a>	Expands the above service to buffer line and polygon features

<a href="#">Watershed (step by step)</a>	Creates a watershed polygon from input points
<a href="#">Stream network</a>	Produces a stream network for cartographic display
<a href="#">More stream network</a>	Expands the above service by allowing the user to download existing stream networks
<a href="#">Clip and ship</a>	Extracts data based on area of interest polygons, creates a file geodatabase of the extracted features, compresses the geodatabase into a .zip file, and optionally e-mails the .zip file to the user
<a href="#">Data on demand</a>	Much like the above example, but makes extensive use of scripting
<a href="#">Selecting data</a>	Shows a variety of ways to select data by attribute and location
<a href="#">Drivetime polygons</a>	Creates polygons based on drive time around points
<a href="#">Shortest route on a street network</a>	Finds the shortest route on a street network
<a href="#">Finding nearby features over a street network</a>	Finds features closest to a given location based on shortest route on a street network

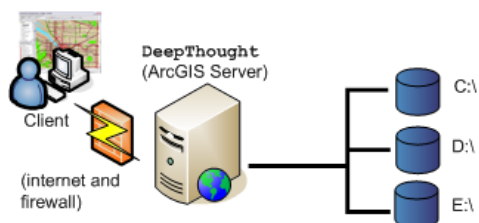
Quick guide to the examples

## Examples use the toolshare folder structure

All corresponding folders use the [toolshare folder structure](#), illustrated below and described in the [Sharing tools and toolboxes](#) book. You are not required to use this folder structure for your services; it is provided only as a guideline.



## Server configuration for all examples



Stand-alone configuration: all resources (documents, toolboxes, and data) needed to execute tasks are found on the server's local disk drives (C: , D:, and E:).

# GP service step by step: Buffer points

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	BufferPoints
Purpose	Creates polygons around points.
Services	BufferService (geoprocessing service).
Geoprocessing tasks	Buffer Points
Inputs	User digitizes a point.
Outputs	Polygon
Data	None
Extensions	None
Of note	Of all the examples, this is the most basic service.

About this example

## Corresponding folder

Data can be found at `C:\arcgis\ArcTutor\GP Service Examples\BufferPoints`. After completing all the steps described below, you will have duplicated the contents of this folder.

## Data preparation

### Create a toolshare folder

Steps:

1. Start ArcCatalog.
2. Create a new folder, `BufferPts`, in a location of your choice, as follows:
  - a. Navigate to an existing folder of your choice.
  - b. In the Catalog tree, right-click the folder and click **New > Folder**.
  - c. Name this folder `BufferPts`.
3. In the Catalog tree, right-click `BufferPts` and click **New > Folder**.
4. Name the folder `ToolData`.
5. In the Catalog tree, right-click `ToolData` and click **New > File Geodatabase**.
6. Name the file geodatabase `Schema`.
7. Using the same steps, create a folder within `BufferPts` named `Scratch`. Within the `Scratch` folder, create a new file geodatabase named `Scratch`.

## Create the toolbox

In the Catalog tree, follow these steps:

Steps:

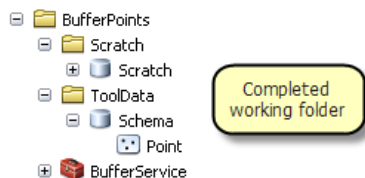
1. Right-click the `BufferPts` folder.
2. Point to **New > Toolbox**.
3. Name the toolbox `BufferService`.

## Create schema feature class

You will need a point feature class to use as a schema for the feature set variable you will use in the model described below.

Steps:

1. In `BufferPts/ToolData`, right-click the Schema geodatabase and click **New > Feature Class**.
2. Name the feature class `Point`.  
The feature type is Points.
3. Click **Next**.  
For a coordinate system, choose **Geographic Coordinate Systems > World > WGS 1984**.  
(Although you can choose Unknown as the coordinate system, it is not good practice to do so.)
4. Click **Next**.
5. Accept the default value for **XY Tolerance** and click **Next**.
6. Accept the default value for **Configuration keyword** and click **Next**.
7. Click **Finish** (the feature class does not contain any user-defined attributes).



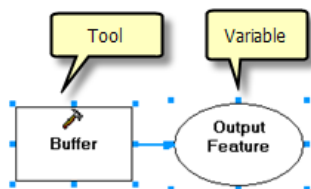
## Create the model

With the steps below, you will create a new geoprocessing model to buffer point features.

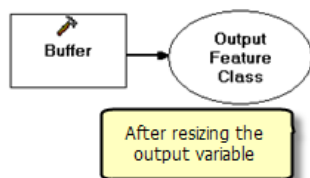
Steps:

1. Right-click the `BufferService` toolbox, point to **New**, then click **Model**. This opens the **ModelBuilder** window.
2. To add the **Buffer** tool
  - a. Click **Add** on the ModelBuilder toolbar.

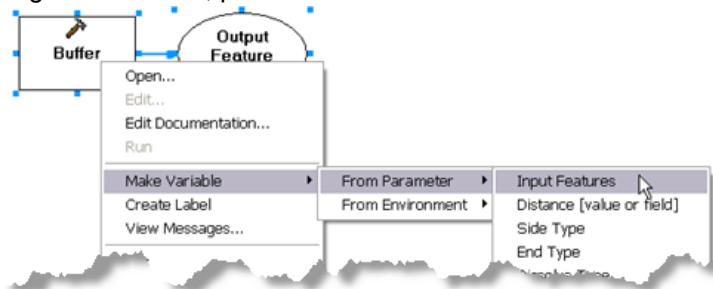
- b. On the **Add Data or Tool** dialog box, navigate to **Toolboxes > System Toolboxes > Analysis Tools > Proximity**.
  - c. Click Buffer and click **Add**. (An alternative to this method is to drag the Buffer tool from the **ArcToolbox**, **Search**, or **Catalog** window onto the ModelBuilder canvas.)
- The ModelBuilder canvas now appears as follows:



3. When using ModelBuilder, you often need to move and resize elements (tools and variables). In the illustration above, the output variable is actually named Output Feature Class but is truncated because of the size of the variable. To resize, click the output variable and click and drag one of the small blue selection squares. Now your model looks as follows:



4. Right-click Buffer, point to **Make Variable > From Parameter**, then click **Input Features**.



The model now looks as

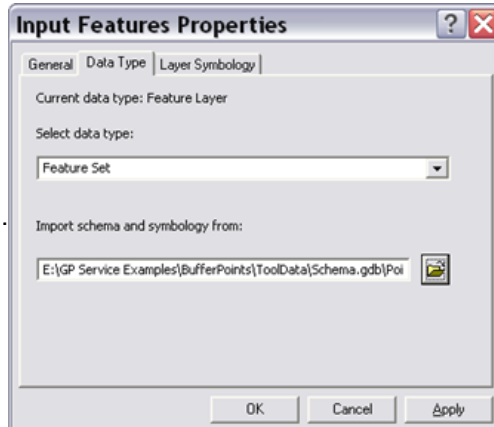
illustrated below:



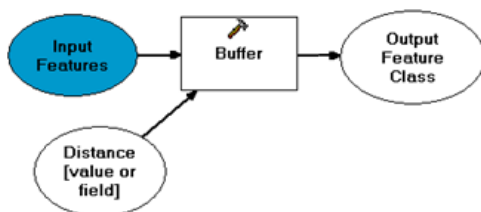
5. Right-click Input Features and click **Properties**. On the **Input Features Properties** dialog box, click the **Data Type** tab.

6. In the **Select data type** drop-down list, choose **Feature Set**. On the **Import schema and symbology from** box, enter the path or browse to the point feature class `Points` you created

in the above steps.



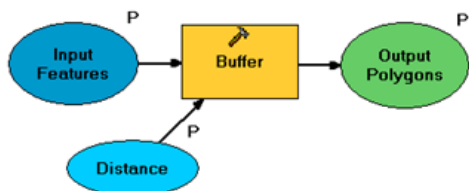
7. Click **OK**. The Input Features variable is now a blue color.
8. Right-click Buffer, point to **Make Variable > From Parameter**, then click **Distance [value or field]**. A new model variable is created.
9. You may need to click and drag the variable so that it is not on top of the Input Features variable and resize it to show the complete variable name.



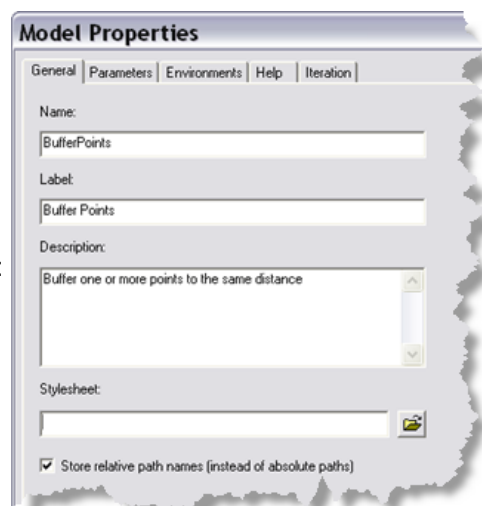
10. Right-click the Distance [value or field] variable and click **Rename**. Rename the variable to `Distance`.
11. Right-click Distance and click **Properties**. On the **Distance Properties** dialog box, click the **Data Type** tab.
12. In the **Select data type** drop-down list, choose **Linear Unit**.
13. Click **OK**.
14. Double-click Distance (or right-click and click **Open**). Set the distance to `1000 meters`. The Distance variable is now a blue color, and the Output Feature Class variable is a green color. This signifies that all required inputs to Buffer have been provided.
15. Right-click the Output Feature Class variable (the output of Buffer) and click **Rename**. Rename the variable to `Output Polygons`.
16. Double-click Output Polygons and enter `%scratchworkspace%\BufferedPoints.shp`



17. Click **OK**
  18. Right-click Input Features and click **Model Parameter**. **P** (for parameter) appears next to the variable.
  19. Right-click Distance and click **Model Parameter**. **P** appears next to the variable.
  20. Right-click Output Polygons and click **Model Parameter**. **P** appears next to the variable.
- The model should now appear similar to the illustration below:



21. In the main ModelBuilder menu, click **Model** and click **Model Properties**
  - a. Set **Name** to BufferPoints
  - b. Set **Label** to Buffer Points
  - c. Check the **Store Relative pathnames** option.



The illustration below shows these settings:

22. Click **OK**.
23. In the main ModelBuilder menu, click **Model** and click **Save**. Then click **Model** again and click **Close**

## Test the model

In the next series of steps, you will test your model using ArcMap—always a good idea before publishing a service.

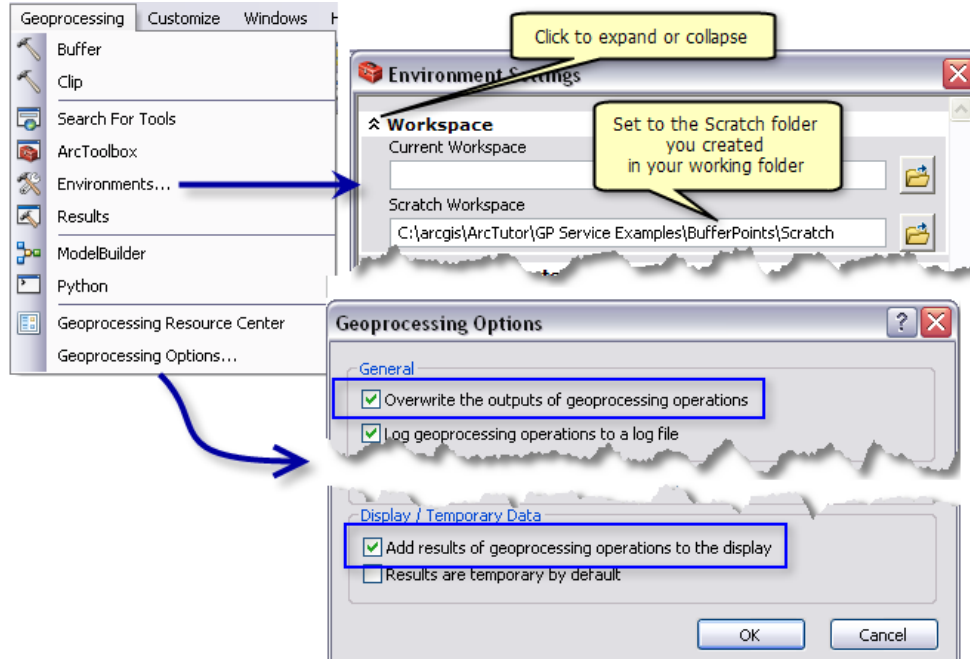
Steps:

1. Start ArcMap and add data to create a map, or open an existing map document.
2. In ArcMap, use the **Geoprocessing** menu to open both the **Geoprocessing Options** and **Environments Settings** dialog boxes.

3. Configure the following:

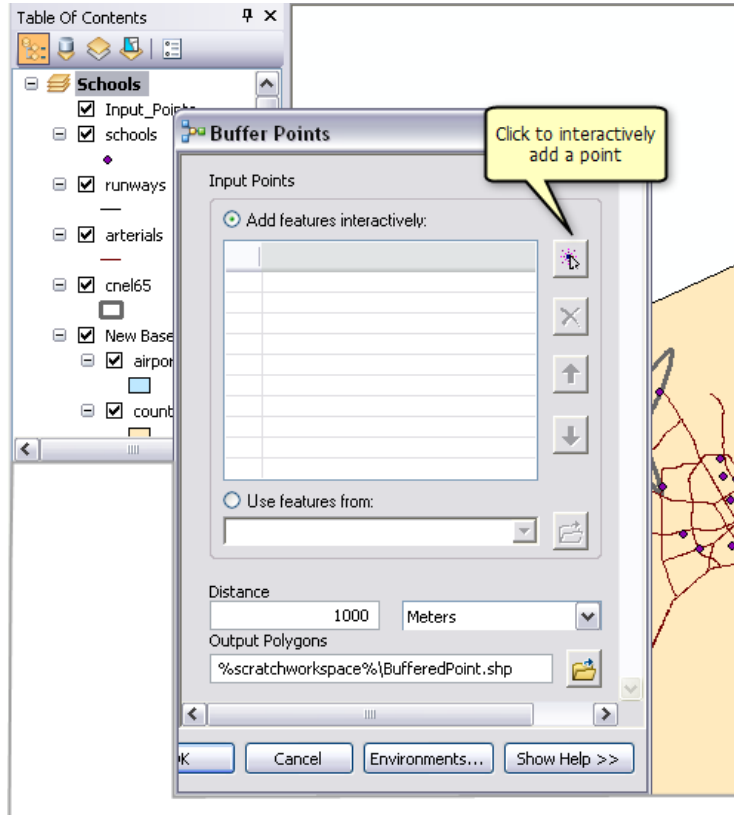
- **Geoprocessing Options:** Check the **Overwrite the outputs of geoprocessing operations** check box.
- **Geoprocessing Options:** Check the **Add results of geoprocessing operations to the display** check box.
- **Environments Settings:** Expand the Workspace category and set the scratch workspace to BufferPoints\Scratch, the folder you created above.


These settings are illustrated below:



4. In the **Catalog** window of ArcMap, navigate to your toolshare folder and choose the **BufferService** toolbox you created above.

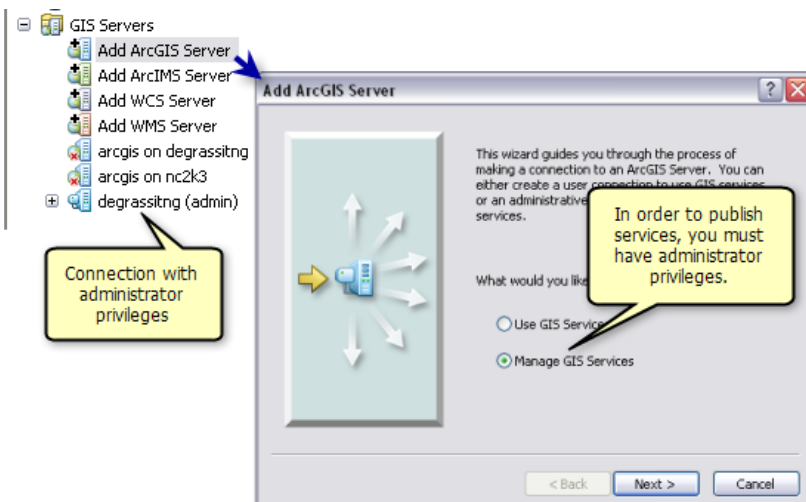
- Double-click **Buffer Points**. (Optionally, you can right-click **Buffer Points** and click **Open**). This opens the **Buffer Points** tool dialog box, as illustrated below.



- Click **Add Feature**  to add a point feature to buffer. You can add more than one feature. [Learn more about using the feature set control to add features](#)
- Optionally, change the **Distance** parameter.
- Click **OK**.  
The tool begins to execute. When the tool finishes execution, a new layer named **BufferedPoints** is added to the ArcMap table of contents.
- Exit ArcMap. You do not need to save your changes.

## Publish the service

To publish a toolbox to ArcGIS Server, you must have administrator access to ArcGIS Server. To connect to a server, expand the GIS Servers entry in the Catalog tree and click **Add ArcGIS Server**. Your server administrator—the person in charge of setting up and maintaining accounts for your ArcGIS Server installation—is responsible for setting up an account for you and granting you administrator privileges.



Once you have established an administrator connection to ArcGIS Server, you are ready to publish your toolbox.

Steps:

1. In ArcCatalog, right-click the `BufferService` toolbox and click **Publish to ArcGIS Server**.
2. On the **Publish to ArcGIS Server** dialog box, choose the server you want to publish to. Name the service `BufferService` (this is the default name—the same as the name of the toolbox).
3. Click **Next**.
4. Click **Finish**.

[Learn more about publishing geoprocessing services](#)

## Use the service

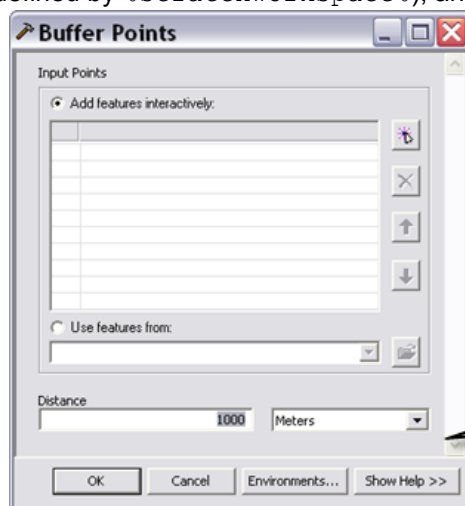
In the next series of steps, you will use your geoprocessing service in ArcMap.

Steps:


1. Open ArcMap and add data of your choosing, or open an existing map document.
2. In the **Catalog** window of ArcMap, navigate to your server connection under GIS Servers.
3. Find the geoprocessing service **BufferService** under the server and expand **BufferService** to display its contents, the **Buffer Points** task.
4. Double-click **Buffer Points** task. (Optionally, you can right-click Buffer Points and click **Open**). This opens the **Buffer Points** task dialog box. Note that unlike the **Buffer Points** tool dialog box (shown above) when you tested your model, this dialog box does not show the **Output Polygons** parameter. This is because ArcGIS Server writes the output polygons to a location

on the server (defined by %scratchworkspace%), and there is no need for you to specify an

output location.



The tool dialog of a geoprocessing task does not display output parameters

5. Click **Add Feature** (  ) to add a point feature to buffer. You can add more than one feature. [Learn more about using the feature set control to add features](#)
6. Optionally, change the **Distance** parameter.
7. Be sure to turn background process off to allow process dialog box appears by running following steps:
  - a. Click **Geoprocessing** from ArcMap.
  - b. Select **Geoprocessing Options**
  - c. If the **Enable** check box of **Background Processing** is checked, uncheck it. Otherwise leave the way it is.

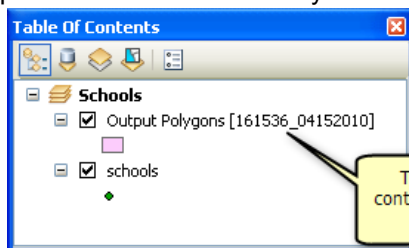
[Learn more about foreground and background processing](#)

8. Click **OK**.
9. Click **Geoprocessing menu > Results**, expand **Current Session**, then expand the **BufferPoints** entry. Note how you can view outputs, inputs, environments, and messages for the execution of the task. Anytime you execute a geoprocessing tool, a result is written to the **Results** window.

[Learn more about using results](#)

It should only take a few moments for the task to complete. A new layer is added to the ArcMap table of contents. This layer is named Output Polygons [<time>\_<date>], as illustrated

below.



The output layer name contains time (16:15:36) and date (04/15/2010)

In the next series of steps, you will be modifying the `BufferService` properties and running the `Buffer Points` task again to see the effect of changing these properties. Therefore, you do not want to exit ArcMap.

## Modifying service properties

By changing service properties, you can affect how tasks execute.

Steps:

1. In the **Catalog** window, navigate to the server containing the `BufferService` geoprocessing service.
2. Right-click `BufferService` and click **Stop**. Geoprocessing services must be stopped before their properties can be changed.
3. Right-click `BufferService` and click **Service Properties**.
4. Click the **Parameters** tab.

There are three basic parameters that you can modify that affect how the task executes. Change these parameters as discussed below, click **OK**, then start the service (right-click `BufferService` and click **Start**). Then run the task again in ArcMap to see the effect.

## Execution type

**Synchronous** means that the client waits until the server has finished executing the task. **Asynchronous** means that the client is free to do other work while the server executes the task. You should choose synchronous only for tasks that execute quickly.

`BufferService` has one task, `Buffer Points`, and this task executes quickly. Change the execution type to synchronous, restart the service, then execute `Buffer Points` again. With synchronous execution, the progress dialog box remains open until the task is finished executing.

## Maximum Number of Records Returned by Server

The number you enter is the maximum number of records or features that can be transferred from the server to the client. A value of 0 means no records can be transferred. The default is 1000. After stopping the service, change the value to 0, restart the service, then execute `Buffer Points` again. The task executes and a layer is created, but there are no features since the maximum is set to 0. If you view the result on the **Results** window, `Output Polygons` has `<data exceeds transfer limit>`.

## Show Messages

Geoprocessing models write messages during the execution of model processes. These messages include warnings, errors, and other information. The messages can contain paths to data residing on your server or local area network, and you may not want the paths to this data to be viewed by users. By default, messages are not shown.

Check the checkbox next to **Show Messages**, restart the service, then execute Buffer Points again. You should see more messages in both the progress dialog box (if the service is running synchronously) and in the result.

When you are developing and testing services, you almost always want to show messages.

# GP service example: Buffer features

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	BufferFeatures
Purpose	Create polygons around point, line, or polygon features.
Services	BufferFeatures (geoprocessing service).
Geoprocessing tasks	Buffer Points, Buffer Lines, Buffer Polygons.
Inputs	User digitizes a point, line, or polygon.
Outputs	Polygon
Data	None
Extensions	None
Of note	<a href="#">GP service step by step: Buffer points</a> showed how to build a service that creates buffer polygons around points. This service allows you to create buffers around the three basic feature types: points, lines, and polygons.

About this example

## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\BufferFeatures contains the completed models and data.

## Data

Similar to the [BufferPoints service](#), you need to create schema feature classes for use in the three models. For this service, you need three schema feature classes: one containing point features, one containing line features, and another containing polygon features. Create these feature classes in the Schema geodatabase inside the ToolData folder.

## Models

There is one model for each of the feature types. These models are constructed using the same basic steps as described in [GP service step by step: Buffer points](#). The main difference between the models is that they use different schemas for the input variable to Buffer.

- Buffer Lines model—The Input Lines variable uses line schema.
- Buffer Points model—The Input Points variable uses point schema.
- Buffer Polygons model—The Input Polygons variable uses polygon schema.

The Buffer Lines model has one additional parameter, the end type of the line, which is either ROUND or FLAT, as described in the [Buffer tool reference](#). This variable was created as follows:

- In the Buffer Lines model, right-click Buffer and click **Make Variable > From Parameter > End Type**.

- Right-click the End Type variable and check **Model Parameter**.

It is not required that you make a variable for the **End Type** parameter. End Type will default to ROUND if you do not create a variable.

## Publishing

The BufferFeatures toolbox is published using similar steps as described in [GP service step by step: Buffer points](#).

# GP service step by step: Watershed

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	Watershed
Purpose	Using the Spatial Analyst extension, this basic service computes a watershed polygon.
Services	StoweBasemap (map service), StoweHydro (geoprocessing service).
Geoprocessing tasks	Create Watershed Polygon
Inputs	User digitizes a point in the study area.
Outputs	Polygon of computed watershed and a snapped pour point.
Data	This example uses digital elevation data (raster) and other data found in the Spatial Analyst tutorial.
Extensions	Spatial Analyst.

About this example

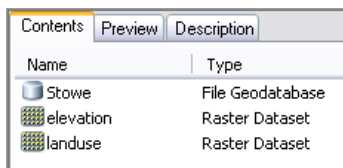
## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\Watershed contains the completed model and data. After completing all the steps described below, you will have duplicated the contents of this folder.

## Data preparation

### Data

You can find the data for this example in C:\arcgis\ArcTutor\Spatial Analyst. The data is for the town of Stowe, Vermont. The contents of this location are shown below:



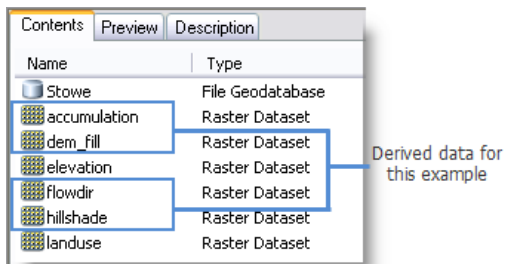
Contents	
Preview	
Description	
Name	Type
Stowe	File Geodatabase
elevation	Raster Dataset
landuse	Raster Dataset

## Create a toolshare folder

1. Start ArcCatalog.
2. In a location of your choice, create a new folder named Watershed. Within Watershed, create a ToolData and Scratch folder. Within Scratch, create a new file geodatabase named Scratch.
3. Copy the contents of the C:\arcgis\ArcTutor\Spatial Analyst folder into ToolData.

## Data processing

For this service, you'll need to create four new raster datasets, as shown below.



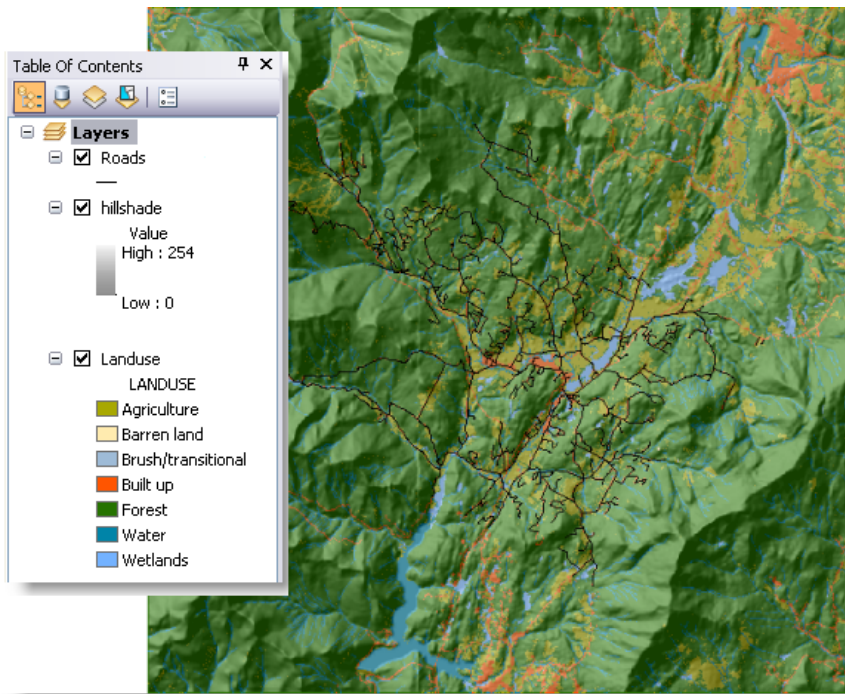
- `Dem_fill` is the result of executing the [Fill](#) tool using elevation as input.
- `Hillshade` is the result of executing the [Hillshade](#) tool using `dem_fill` as input.
- `Flowdir` is the result of executing the [Flow Direction](#) tool using `dem_fill` as input.
- `Accumulation` is the result of executing the [Flow Accumulation](#) tool using `flowdir` as input.

The model in this example uses a feature set, and you will need a point feature class to use as the feature set schema. In `Stowe.gdb`, create a new point feature class named `PourPoint`.

- For a coordinate system, import any of the existing datasets within the `ToolData` folder.
- Use the default values for `xy` tolerance and configuration keyword.

## Basemap

The basemap contains three layers: Landuse (landuse raster), Hillshade (hillshade raster), and Roads (Stowe.gdb/roads feature class). You will need to create a map document containing these layers. Name the ArcMap document StoweBasemap.mxd.




The Hillshade layer is drawn with a transparency of 55 percent. To change transparency, right-click the Hillshade layer, choose Properties, click the **Display** tab, then change the transparency.

After creating and saving StoweBasemap.mxd, publish it as a map service to your server.

### Steps:

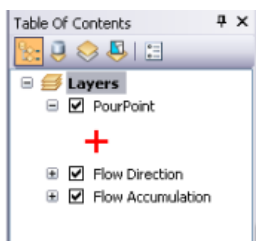
1. In the **Catalog** window, right-click StoweBasemap.mxd and click **Publish to ArcGIS Server**.
2. In the first panel, name the service StoweBasemap (the default).
3. Click **Next**.
4. The only capability that you need is mapping—all other capabilities are optional for this service.
5. Click **Next**.
6. Click **Finish**.

You can test the StoweBasemap service by starting ArcMap with a blank document, then add the service by clicking Add Data , navigating to the server, and choosing StoweBasemap.

## Toolbox and map document

1. In the Stowe folder, create a new toolbox with the name `StoweHydro`.
2. Start ArcMap with a new document and add the accumulation and flowdir rasters to the table of contents, renaming the layers to `Flow Accumulation` and `Flow Direction`, respectively. Add the `PourPoint` feature class you created above.
3. Optionally, change the symbology of `PourPoint` to a red plus sign, as illustrated.

The Flow Direction and Flow Accumulation layers will be used in the published task but never displayed to the user. Therefore, the symbology of these layers does not matter.



Set the geoprocessing scratch workspace environment to the Scratch folder, as follows:

Steps:

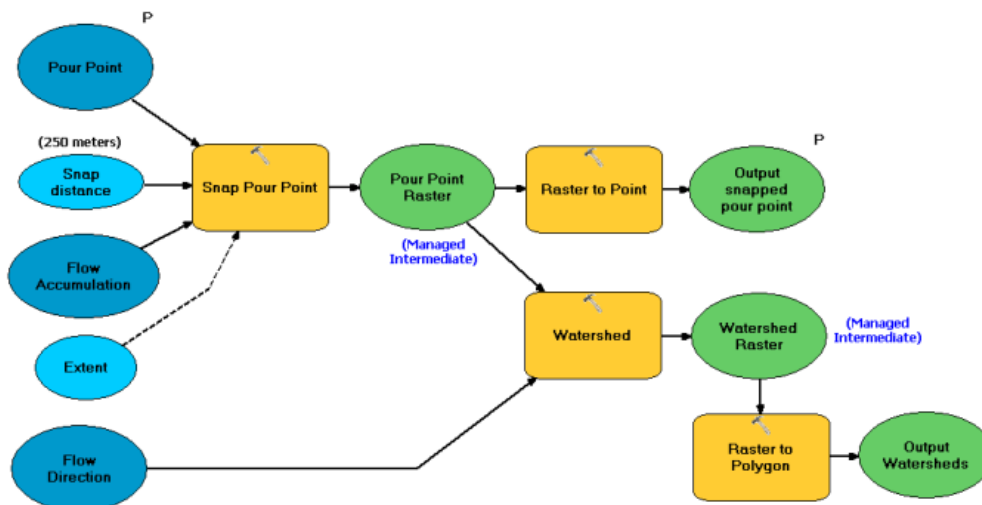
1. In ArcMap, click **Geoprocessing > Environments**.
2. Expand the **Workspace Settings** category.
3. Enter the path to the scratch workspace you created above (for example, `E:\Watershed\Scratch`).
4. Accept the change by clicking **OK**.
5. Save the map as `StoweHydro.mxd`.

## Create the model

The input to the Create Watershed Polygon model is a user-supplied point. A watershed polygon is output for each input point. In addition, the user-supplied points will be snapped to the cells of highest flow accumulation using the [Snap Pour Point](#) tool. The snapped points will also be output.

Since these models make use of layers in the map document, you create these models in ArcMap using the `StoweHydro` map document.

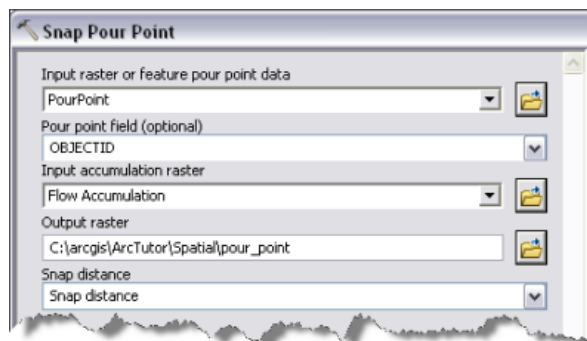
The Create Watershed Polygon model is illustrated below:



## Steps to building the model

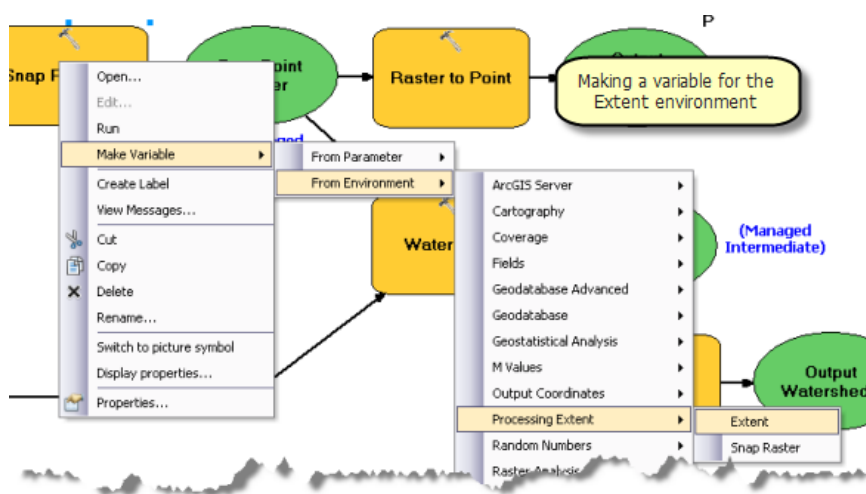
### Steps:

1. In the **Catalog** window, right-click the StoweHydro toolbox and choose **New > Model**. This opens ModelBuilder with a new empty model.
2. Create a variable with the Feature Set data type, as follows:
  - a. Right-click in the ModelBuilder canvas and click **Create Variable**.
  - b. Choose **Feature Set** as the data type.
3. Click **OK**.
4. Rename the variable as `Pour Point`.
5. Right-click `Pour Point` and click **Properties**. Click the **Data Type** tab. Set the schema to the `Pour Point` layer by choosing it in the drop-down list.
6. Right-click `Pour Point` and click **Model Parameter**. A check mark appears next to **Model Parameter**, and a **P** appears next to the `Pour Point` variable.
7. Add the **Snap Pour Point** tool to the model by dragging the tool from the **ArcToolbox**, **Catalog**, or **Search** window.
8. Right-click `Snap Pour Point` and make a variable from the **Snap Distance** parameter.
9. Double-click `Snap Distance` and set the snap distance to 250 meters. It is not a requirement that you make `Snap Distance` a model variable, but doing so gives anyone that views the model a visual clue that snap distance is an important variable.
10. Double-click `Snap Pour Point` and set the input raster parameter to `Pour Point`. Set the **Pour point field** parameter to `ObjectID` if not already set. Set the **Input accumulation raster** parameter to the `Flow Accumulation` layer.

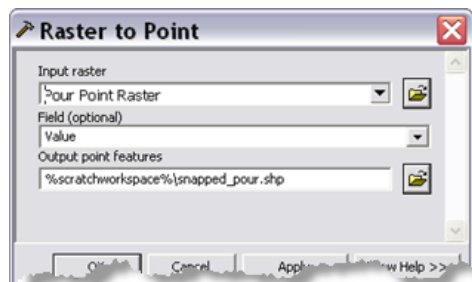


Snap Pour Point parameter settings

11. Right-click Snap Pour Point and click **Make Variable** > **From Environment** > **Processing Extent** > **Extent**, as illustrated below.



12. Double-click the Extent variable and set it to Union of Inputs.
13. Rename the output of Snap Pour Point as `Pour Point Raster`. This is intermediate output and should be deleted after the model executes. Right-click Pour Point Raster and check **Managed**.
14. Add the [Raster To Point](#) tool to the model. Double-click Raster To Point and set the **Input raster** parameter to Pour Point Raster. Set the **Output point features** parameter to `%scratchworkspace%\snapped_pour.shp`.



Raster To Point parameter settings

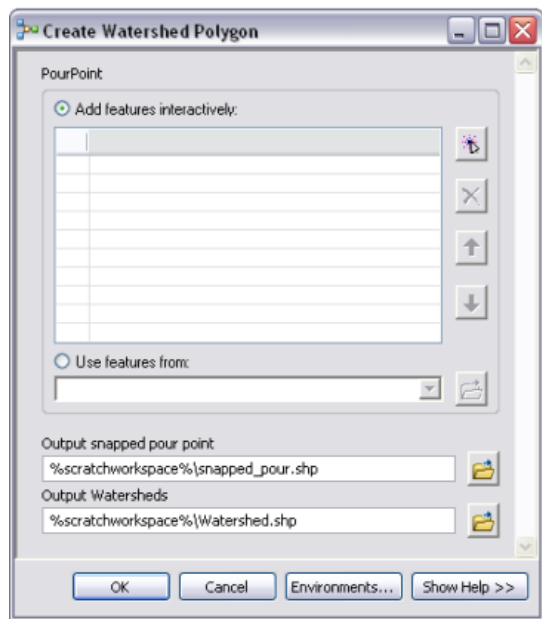
15. Rename the Raster To Point output variable `atos` `Output snapped pour point`.
16. Right-click `Output snapped pour point` and make it a **Model Parameter**.
17. Add the [Watershed](#) tool to the model and use the Flow Direction layer for the **Input flow direction raster** parameter. Rename the output of the Watershed tool as `Watershed raster`. `Watershed raster` is intermediate output.
18. Right-click `Watershed raster` and click **Managed**.
19. Add the [Raster To Polygon](#) tool to the model. Set the **Input raster** parameter to `Watershed raster`. Set the **Output Polygon features** parameter to `%scratchworkspace%\Watershed.shp`.
20. Rename the output variable of the Raster To Polygon tool as `Output Watershed`.
21. Right-click `Output Watershed` and make it a **Model Parameter**.
22. In the main ModelBuilder menu, click **Model > Model Properties**.
  - a. Set Name to `CreateWatershedPolygon`.
  - b. Set Label to `Create Watershed Polygon`.
  - c. Check **Store Relative pathnames**.
23. Save and close the model.

## Creating symbology layer files

In this next series of steps, you will be creating layer files to use as output symbology for your task.


Steps:

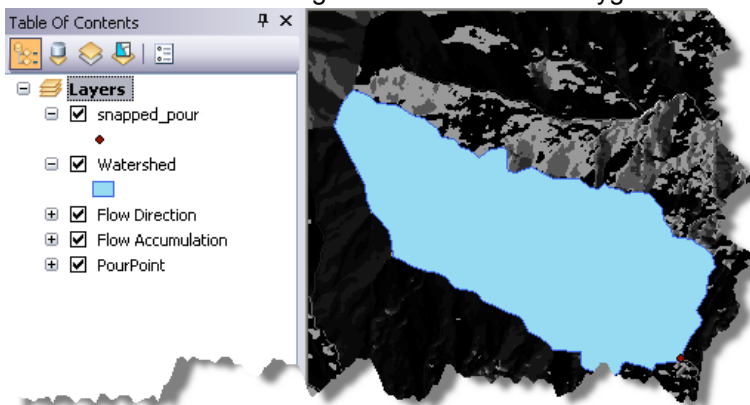
1. In the **Catalog** window, double-click the **Create Watershed Polygon** tool to open its dialog box. The dialog box appears as shown below:



Create Watershed Polygon tool dialog box

(The order of the parameters in your tool dialog box may be different, depending on the order in which you made variables model parameters. You can change the order of parameters in the model tool's properties.)

2. Click Add Feature (  ) to add a pour point. The location you use for the pour point should be in a valley, not on a mountaintop where there is no watershed. The following illustration shows a result of executing Create Watershed Polygon:

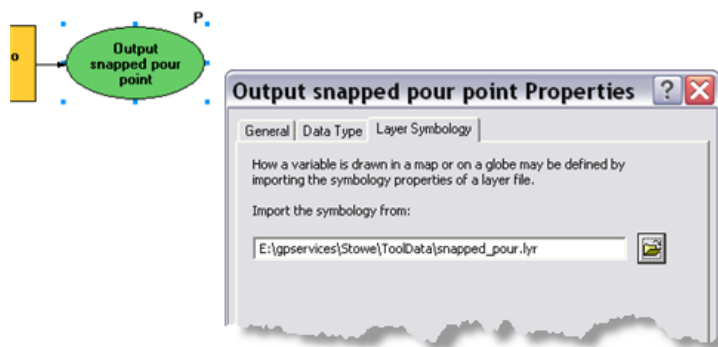


In the above illustration, note that the output watershed polygon is blue and the snapped pour point is a small dot. (Your symbology may be different.) In the following steps, you will create better symbology,

save this symbology as a layer file (.lyr), then use these layer files to define the symbology of your output variables.

#### Steps:

1. Open the properties for the snapped\_pour layer and set the symbol to a red cross.
2. Open the properties for the Watershed layer and set the fill symbol to a light blue.
3. Right-click snapped\_pour, click **Save As Layer File**, then save as Stowe\ToolData\snapped\_pour.lyr.
4. Right-click Watershed, click **Save As Layer File**, then save as Stowe\ToolData\Watershed.lyr.
5. Edit the Create Watershed Polygon model.
6. Right-click the Output snapped pour point variable and click **Properties**. On the Properties dialog box, click the **Layer Symbology** tab and set the symbology layer to snapped\_pour.lyr.



Setting layer symbology

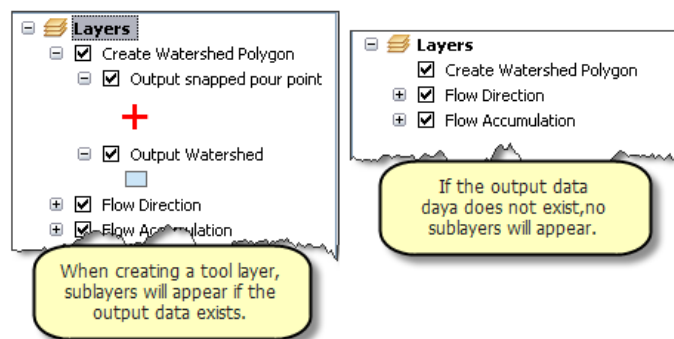
7. Do the same for the Output Watershed variable, setting the symbology layer to watershed.lyr.
8. Save and close the model.
9. Remove the existing snapped\_pour and Watershed layers from the table of contents.

[Learn more about setting symbology using layer files](#)

## Create tool layer

#### Steps:

1. If you have not done so, remove the snapped\_pour and Watershed layers from the table of contents.
2. To create the tool layer, drag the Create Watershed Polygon tool into the ArcMap table of contents. Your tool layer may appear with or without sublayers, as illustrated below. Sublayers appear if the tool outputs exist at the time you create the tool layer.



Tool layer for the Create Polygon Watershed tool

3. Right-click the tool layer and click **Open**. The tool dialog box opens. Add a pour point using the feature set control and run the tool. After the tool runs, the two sublayers in the ArcMap table of contents are refreshed and have the symbology as defined in the symbology layers you created above.
4. Finally, remove the PourPoint layer since it is no longer needed. You should first save the layer to a layer file for subsequent reuse.
5. Save the map as `StoweHydro.mxd` and exit ArcMap.

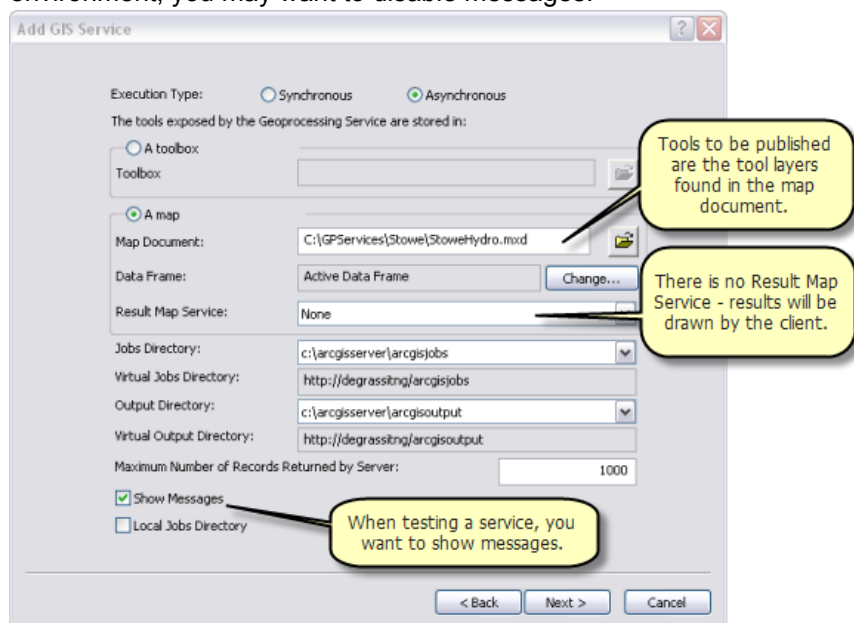
## Publish the service

In [previous steps](#), you published the StoweBasemap map service. Now you will publish the StoweHydro geoprocessing service. The StoweHydro service contains the Create Watershed Polygon task.

### Steps:

1. In the **Catalog** window, navigate to your server, right-click, then choose **Add New Service**. Name the service `StoweHydro` and choose **Geoprocessing Service** as the type.
2. Click Next.
3. Choose `StoweHydro.mxd` as the source for your geoprocessing service, as illustrated below. Since you will test your service, check the **Show Messages** checkbox. In a production

environment, you may want to disable messages.

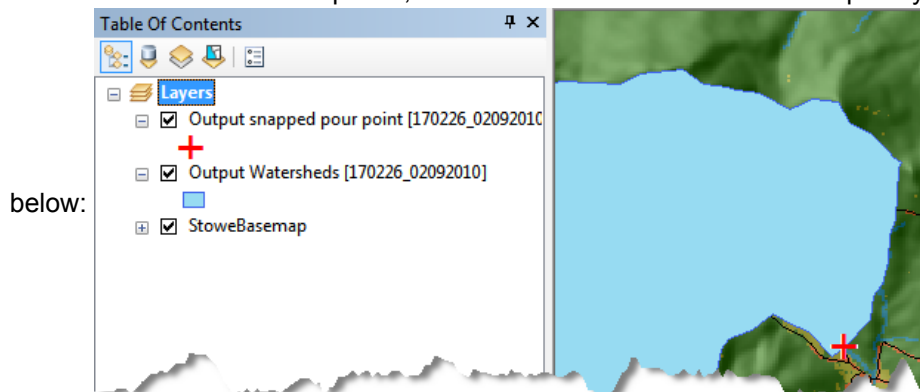


4. Click **Next**. From this point on, you can accept the default values provided by the wizard.

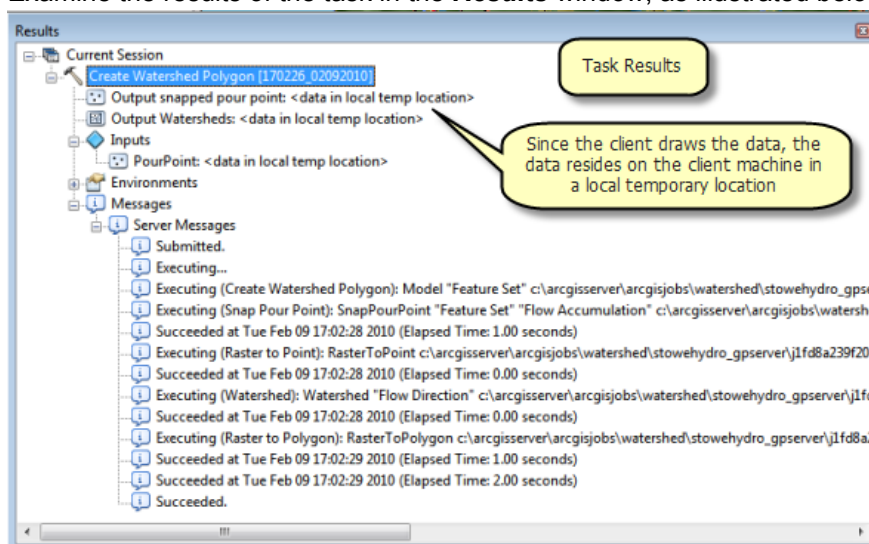
## Use the service

Steps:

1. Start ArcMap with a blank document.
2. Add the StoweBasemap map service to the ArcMap table of contents from your ArcGIS Server connection.
3. Expand the StoweHydro geoprocessing service from the ArcGIS Server connection and open the Create Watershed Polygon tool.
4. Add a point to the Create Watershed Polygon using the feature set control and click **OK** to run the task. After the task completes, the table of contents has the two output layers, as illustrated



- Examine the results of the task in the **Results** window, as illustrated below:



# GP service example: Stream network

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	StreamNet
Purpose	Creates a stream network from flow and accumulation data.
Services	<ul style="list-style-type: none"> <li>• StoweHillshade (map service)</li> <li>• StoweStreamNet (geoprocessing service and result map service)</li> </ul>
Geoprocessing task	Create Stream Network
Inputs	Minimum upstream drainage area in hectares.
Output	Stream network
Data	Uses digital elevation data (raster) and other data found in the Spatial Analyst tutorial.
Extension	Spatial Analyst
Of note	Uses a result map service to draw the network.

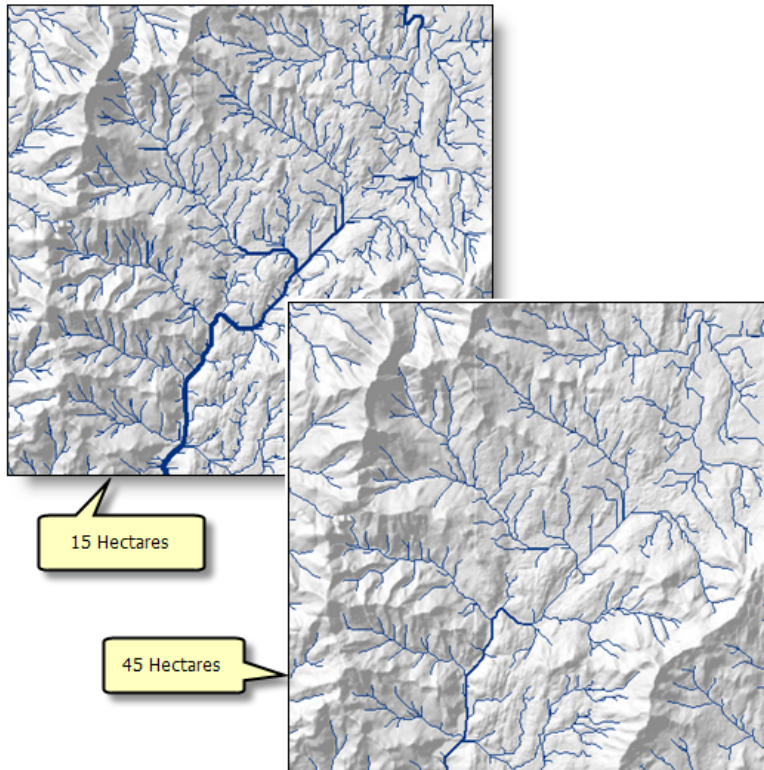
About this example

## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\StreamNet contains the completed model and data.

## About the Create Stream Network task

The primary purpose of the Create Stream Network task is to produce a stream network for cartographic display. Two results are illustrated below, using a hillshade of the study area (the town of Stowe, Vermont) as a basemap. The idea behind this service is to let the user produce several different networks and choose one that suits his or her cartographic needs.



The task creates a stream network defined by a minimum area—the smaller the area, the more stream segments are produced. Each stream segment is assigned a stream order value based on the number of upstream stream segments, and this value is used to determine the line thickness to display each stream segment.

The number of stream segments produced depends on the minimum area used. Using an input of 1 hectare, approximately 32,000 stream segments are produced. For 45 hectares, approximately 600 stream segments are produced.

One of the decisions that you make when creating a geoprocessing service is determining the maximum number of records and features that can be returned from the server to the client. The default is 1,000, and you can increase this to billions. However, transporting large numbers of features across the Web is expensive and slow. Whenever you have a service that may output a large number of features, you should consider using a result map service to create a map of the result and let ArcGIS Server transport the map across the Web instead of the features. Because the number of stream segments produced by the Create Stream Network task is dependent on the minimum drainage area and, if you substitute your own data, the size and topography of the study area, this example uses a result map service to draw the results instead of transporting the features across the Web to the client.

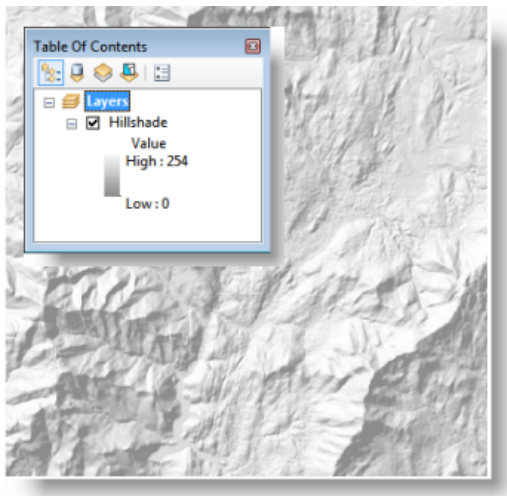
## Data

### Data

The data for this example comes from `C:\arcgis\ArcTutor\GP Service Examples\Watershed`. This data, in turn, was derived from `C:\arcgis\ArcTutor\Spatial Analyst`. To learn how the data in the Watershed folder was created, see the [Watershed example](#).

### Basemap

The basemap for this example, `StoweHillshade.mxd`, has only one layer, Hillshade, as illustrated below. The Hillshade layer is drawn with a transparency of 55 percent.



Stowe hillshade basemap

`StoweHillshade.mxd` is published as a map service.

### Toolbox and map document

The toolbox for the geoprocessing service is `StoweStreamNet`, and the source map document for the service is `StoweStreamNet.mxd`. `StoweStreamNet.mxd` contains two source data layers, Flow Direction (the flowdir raster) and Flow Accumulation (the accumulation raster).

## Model

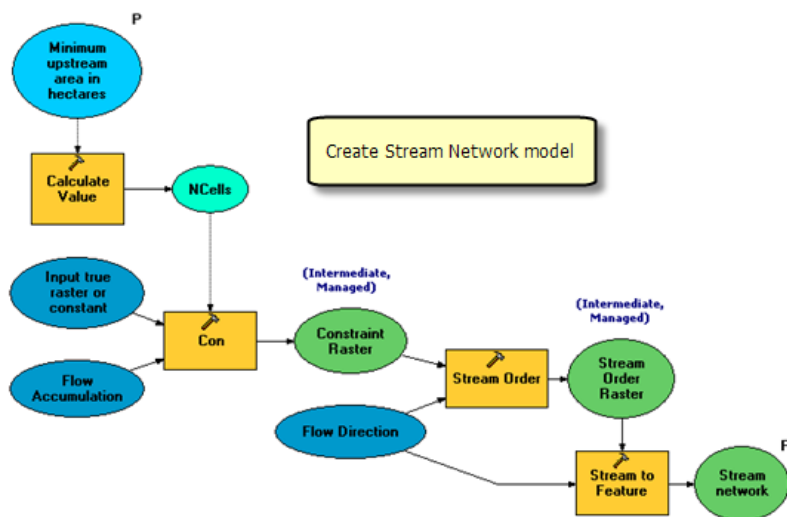
### Model overview

The Create Stream Network model is illustrated below. There is one input variable, Minimum upstream area in hectares. This variable is a double and is the minimum drainage area required to create a stream segment. Smaller areas create more stream segments.

The model calculates the number of raster cells for the input area, then uses the [Con](#) tool to perform a conditional if-else evaluation on each of the input cells in the Flow Accumulation raster. Since a cell value in the Flow Accumulation raster is the number of upstream cells flowing into the cell, any cell with an upstream area greater than the cutoff is selected. Selected cells form stream segments and are assigned a 1 in the output raster.

The [Stream Order](#) tool assigns a numeric order value to a raster representing branches of a linear network, such as the output of the [Con](#) tool. In general, streams with a high order have higher water flows, so stream order can be used as a surrogate for stream width. The output of the model is symbolized so that segments with high order values are drawn with thicker lines.

The [Stream To Feature](#) tool converts the Stream Order Raster into the output line feature class. This feature class will be drawn by the result map service using symbology defined in the result map service.



Details about the model follow.

### Project data

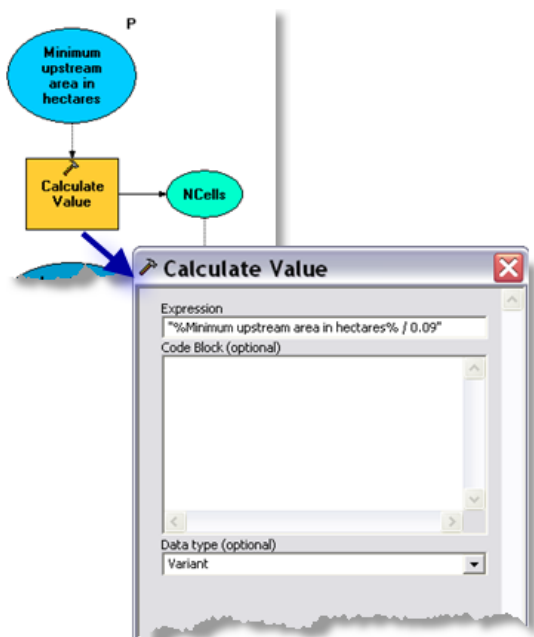
The Flow Accumulation and Flow Direction variables are layers from the source map document, `StoweStreamNet.mxd`. These variables are termed **project data** because they are nonparameter input data.

## Calculating number of cells for the minimum area

The Flow Accumulation and Flow Direction rasters have projected Vermont State Plane coordinates, with a linear unit of meters. The cell size is 30 by 30 meters. To determine the spatial reference and cell size of a raster, do either one of the following:

- In the **Catalog** window, right-click the raster and click **Properties**.
- In the ArcMap table of contents, right-click a raster layer, click **Properties**, then click the **Source** tab.

The **Calculate Value** tool divides the input hectares by the size of a cell in hectares (30 x 30 meters = 900 square meters = 0.09 hectares) to yield the number of cells for the minimum upstream area. The parameters for the Calculate Value tool, illustrated below, make use of variable substitution. By placing percent signs (%) around a variable name, the contents of the variable are substituted for the value.



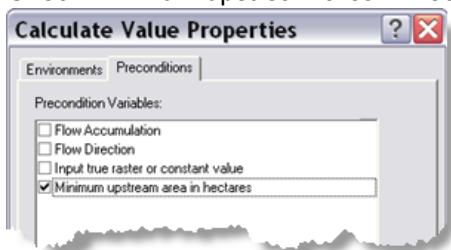
The Minimum upstream area in hectares variable is a double data type, created as follows:

1. Right-click on the ModelBuilder canvas and click **Create Variable**.
2. On the **Create Variable** dialog box, choose Double as the data type.
3. Click **OK**.
4. Rename the newly created variable to Minimum upstream area in hectares.
5. Double-click the variable and enter a default value. For the Stowe area, 45 is a reasonable default.

The Minimum upstream area in hectares variable is a precondition to the execution of Calculate Value. A precondition means that a variable must contain a value before Calculate Value executes. You can set preconditions as follows:

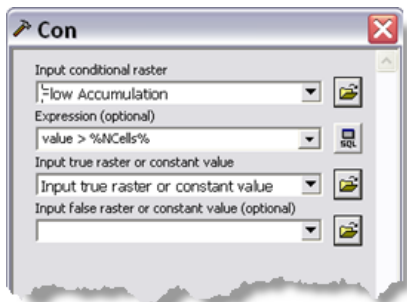
1. Right-click Calculate Value.
2. Click **Properties**.

3. Click the **Preconditions** tab.
4. Check Minimum upstream area in hectares.



## Con process

The **Con** tool performs a conditional if-else evaluation on each of the input cells in the Flow Accumulation raster.



Con tool parameters

Note that the expression makes use of variable substitution. In the expression, all cells with a value greater than the minimum number of cells are selected. (The value of a cell in the Flow Accumulation raster is the number of cells flowing into an individual cell.)

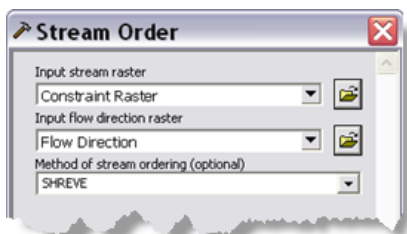
The **Input true raster or constant value** parameter is the value to assign each selected cell and is set to 1. If you are creating this model from scratch, do the following:

1. Double-click the Con tool.
2. For the **Input true raster or constant value** parameter, enter 1.
3. Click **OK**.

The Input true raster or constant value model variable is automatically created and connected to the Con tool.

## Stream order process

The [Stream Order](#) tool calculates a numeric value for cells that represent branches of a stream network. The parameter values for Stream Order are shown below.

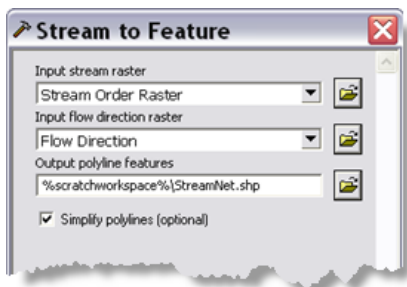


Stream Order parameters

The output raster contains the Shreve stream order value for every cell that represents a stream.

## Stream to feature process

The [Stream To Feature](#) tool creates a shapefile of the stream segments contained in Stream Order Raster.



Stream To Feature parameters

Note that the output is written to %scratchworkspace%. This location is the [scratch workspace environment](#) that is set and used by ArcGIS Server.

## Tool layer

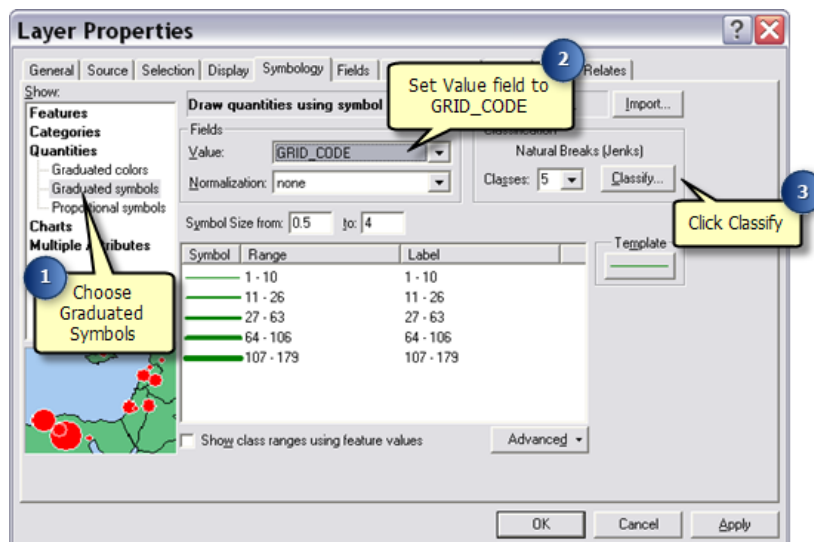
The tool layer in `StoweStreamNet.mxd` defines the result map symbology. The steps below show how to create the tool layer and appropriate symbology:

1. In ArcMap, drag the Create Stream Network tool from the `StoweStreamNet` toolbox into the ArcMap table of contents to create the Create Stream Network tool layer.
2. Right-click the Create Stream Network tool layer and click **Open** to open the tool dialog box.
3. Use 45 (the default) as the minimum area.
4. Click **OK** to run the tool. Note that the default symbology for the stream network is a single line.

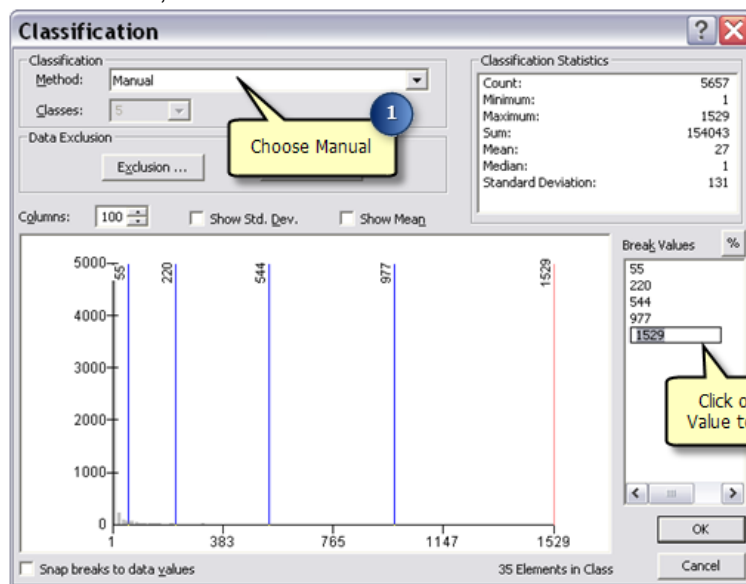
In the steps below, you will change the symbology to graduated line symbols. Right-click the Stream Network sublayer and click **Properties**, then click the **Symbology** tab.

1. Click **Graduated symbols** found beneath the **Quantities** category.

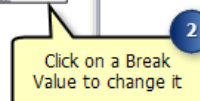
2. Choose GRID\_CODE as the value field.
3. Click **Classify**.



1. On the **Classification** dialog box, choose **Manual** as the classification method.
2. You will set up a manual classification with breaks of 100, 250, 500, 750, and 10000. In the **Break Values** panel, click the last value in the list. This allows you to edit the value. Enter one of the break values, such as 10000. The list refreshes so that the values are sorted.



Continue editing and

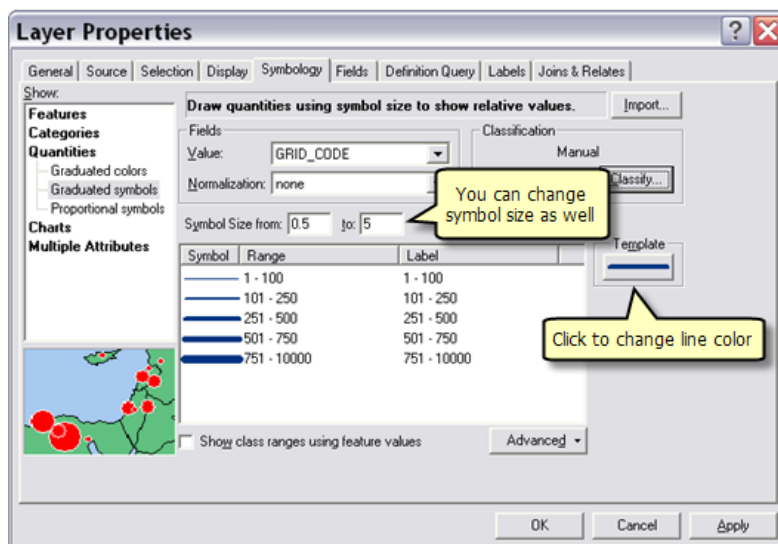


setting the break values until the list appears as illustrated below.



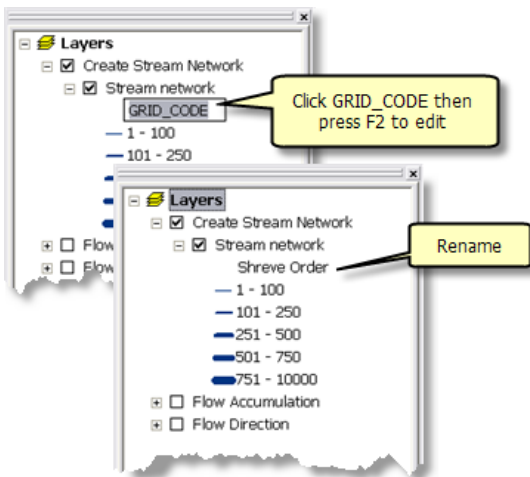
3. Click **OK**.

The **Symbology** tab now looks as illustrated below. You may need to reset the color to blue by clicking **Template** and choosing a suitable blue color.



4. Click **OK**.

The legend for the Stream Network sublayer displays **GRID\_CODE** as the classification field, which will not make much sense to the user of your service. To change it, click **GRID\_CODE** in the legend, press the **F2** key, and change the text to something more descriptive, such as **Shreve Order** or **Stream Order**.



## Publishing

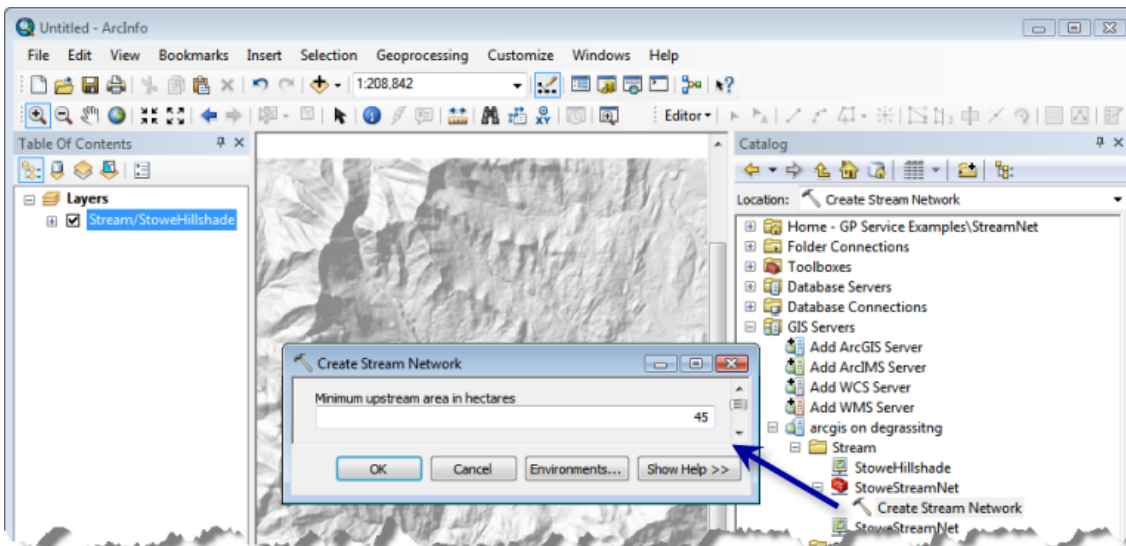
StoweHillshade.mxd is published as a map service.

To publish StoweStreamNet.mxd as a geoprocessing service with a result map service, do the following:

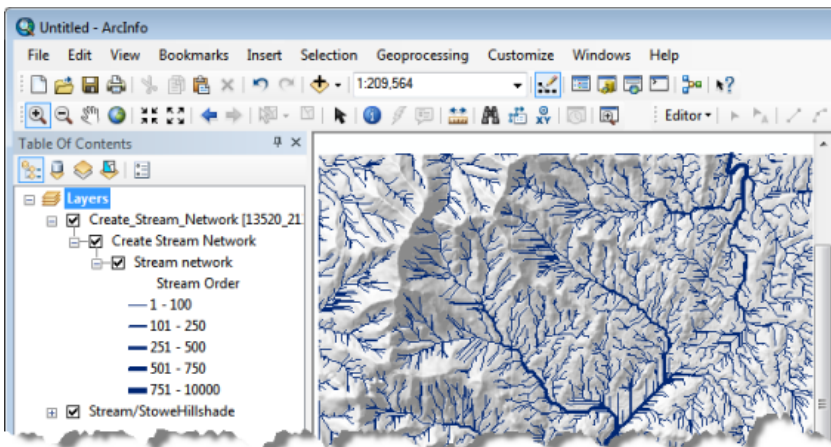
1. In the **Catalog** window, right-click **StoweStreamNet.mxd** and click **Publish to ArcGIS Server**.
2. Accept all defaults.

## Using

To use the services, start ArcMap and add the StoweHillshade map service and the StoweStreamNet geoprocessing service. Do not add the StoweStreamNet map service since this is a result map service and contains the Flow Accumulation and Flow Direction layers, which have nothing to do with results.



ArcMap before executing the service



ArcMap after executing the Create Stream Network task

# GP Service example: More Stream network

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	StreamNet2
Purpose	User can download a precomputed network or create their own.
Services	StoweStreamNetworksBasemap (map service) StoweStreamNetworkTasks (geoprocessing service).
Geoprocessing tasks	Get Precomputed Stream Network, Create Stream Network.
Inputs	For the Get Precomputed Stream Network task, the user inputs a layer name. For Create Stream Network, the user enters the minimum upstream area in hectares.
Outputs	Stream network
Data	This example uses digital elevation data (raster) and other data found in the Spatial Analyst tutorial.
Extensions	Spatial Analyst.
Of note	Uses layer symbology files (.lyr) to instruct the client application on how to draw the output stream networks, as described in the <a href="#">GP Watershed service example</a> .

About this example

## Corresponding Folder

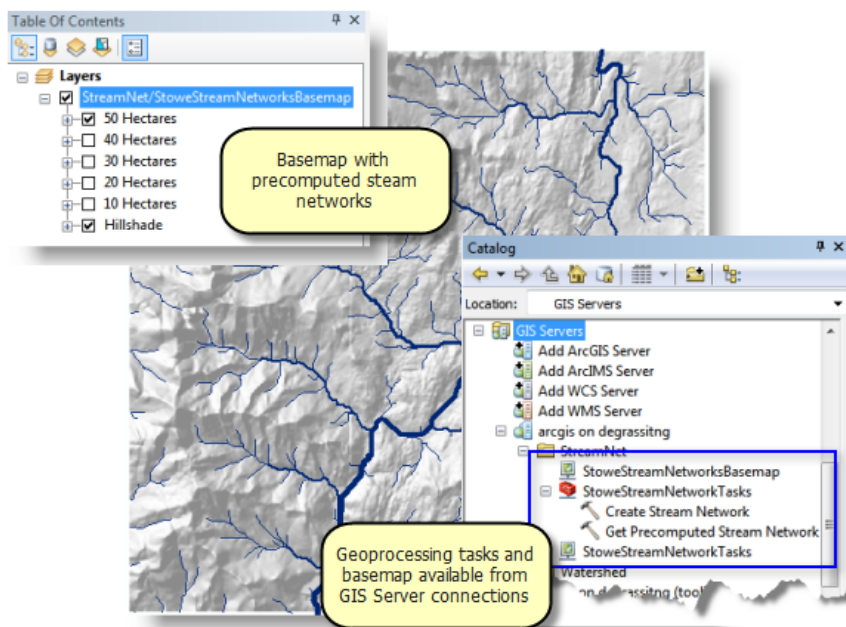
C:\arcgis\ArcTutor\GP Service Examples\StreamNet2 contains the completed model and data.

## About this example

This example builds on the Create Stream Network task created in [GP Service example: Stream network](#). As discussed in that example, the primary purpose of the task was to produce a stream network for cartographic display, allowing the user to produce several different networks, ultimately choosing one that suits their cartographic needs. The Create Stream Network task used a result map service to display its results.

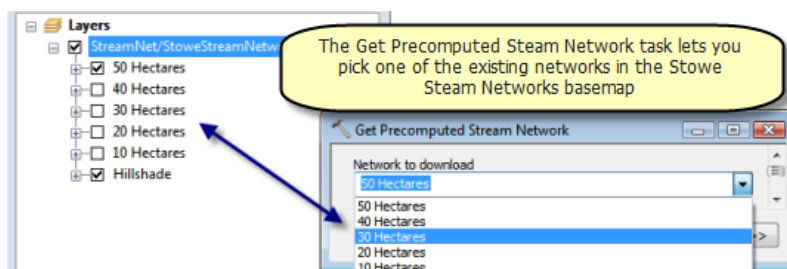
In this example, stream network features are transported to the client for display. Since the features are transported to the client application, a result map service is not needed. The service contains two tasks, one to fetch precomputed networks and another to create a new network.

The illustration below shows the final resulting StoweStreamNetworksBasemap map service table of contents and the geoprocessing tasks in the StoweStreamNetworkTasks geoprocessing service.



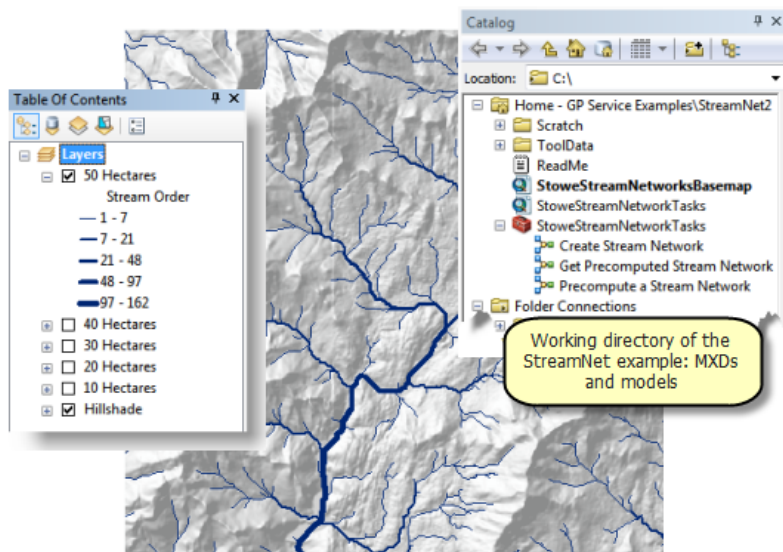
The basemap service allows you to view five precomputed networks, ranging from 10 to 50 hectares. There are two geoprocessing services:

- **Get Precomputed Stream Network**—The user chooses a precomputed stream network, and the features are transported back.
- **Create Stream Network**—The user creates a new network by entering an upstream drainage area. This is a slightly modified version of the Create Stream Network task discussed in the [GP Service example: Stream network](#).



## Basemap

The table of contents of `StoweStreamNetworksBasemap.mxd` is shown below, along with the `StoweStreamNetworkTasks` toolbox as shown in the **Catalog** view.

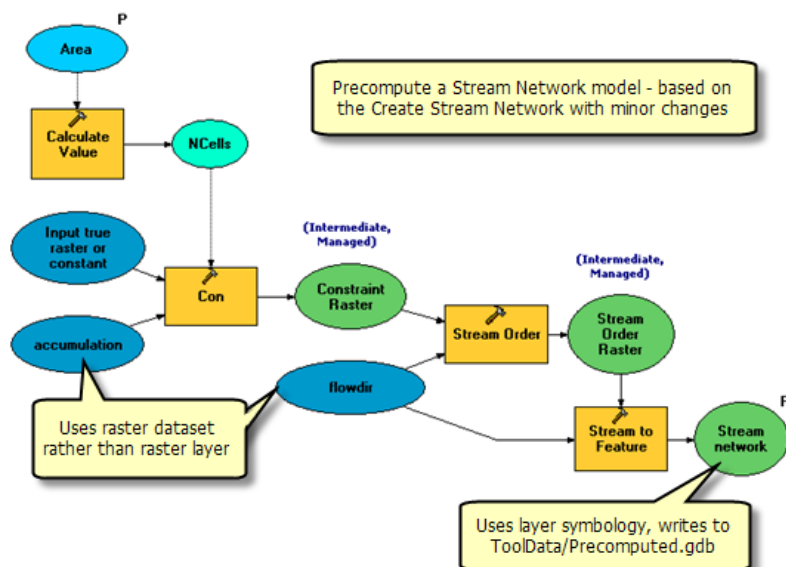


The five stream network layers were created using the Precompute a Stream Network model. This is a utility tool and is not part of the geoprocessing service. Precompute a Stream Network is similar to the Create Stream Network with the following differences:

- The input variable is named Area.
- The accumulation and flowdir raster dataset variables refer to raster datasets in the ToolData folder rather than layers in the ArcMap table of contents. This allows you to execute the model without having the raster layers in the ArcMap table of contents.
- The Stream Network variable has a [Layer Symbology file](#).

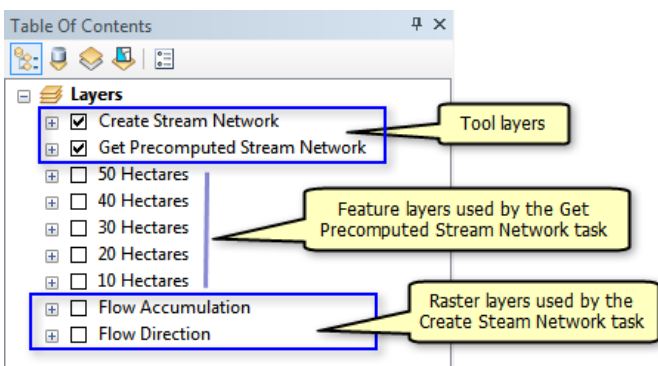
The basic steps for creating and using a layer symbology file are

- Run the Precompute a Stream Network model from the **Catalog** window. A new layer is added to ArcMap.
- Right-click the new layer, choose **Properties**, then click the **Symbology** tab.
- Change the symbology to **Graduated Symbols** using GRID\_CODE as the value. Use a **Natural Breaks (Jenks)** classification.
- Right-click the layer in the table of contents and click **Save As Layer File**. Save to ToolData/Stream Network.lyr
- Edit the Precompute a Stream Network model.
- Right-click the Stream Network output variable and click **Properties**.
- In the Properties dialog box, click the **Layer Symbology** tab and enter the layer file created above.




## Models

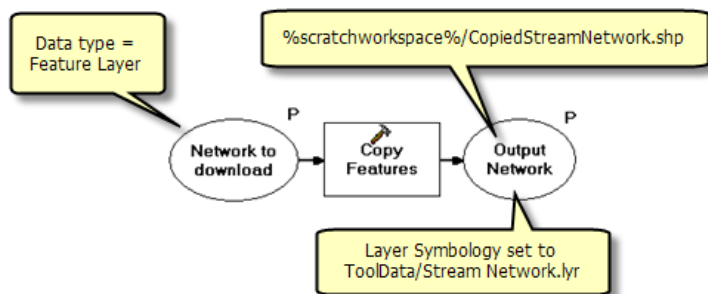
The contents of `StoweStreamNetworkTasks.mxd` are illustrated below.



The table of contents has the same precomputed stream network layers as found in `StoweStreamNetworksBasemap.mxd`. These layers were created as follows:

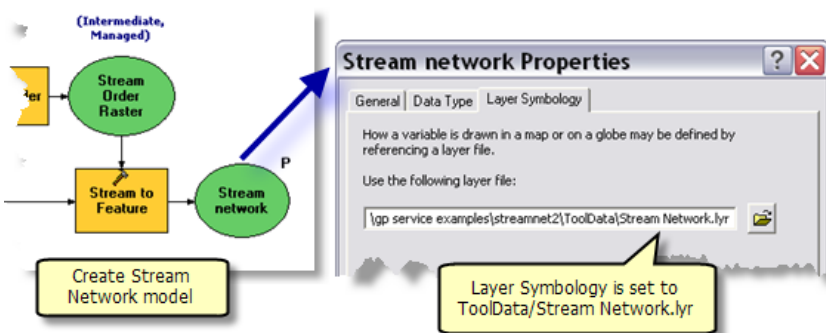
1. Each stream network layer in `StoweStreamNetworksBasemap.mxd` was saved as a layer file to the `ToolData` directory.
2. Using **Add Data**  in `StoweStreamNetworkTasks.mxd`, each layer file created above was added to the table of contents.

The Get Precomputed Stream Network model is a simple model that copies the contents of a feature layer to the scratch workspace.



The input variable, Network to download, is a Feature Layer data type. When the task executes, all feature-based layers are listed, and the user picks one. (Note that the raster layers are not listed since they are not features. Nor are any of the sublayers in the tool layers listed, because ArcGIS Server automatically filters out any tool layers and their sublayers from the list of feature layers.) The underlying features are then copied to the scratch workspace and, when transported back to the client, drawn using the symbology defined in `ToolData/Stream Network.lyr`.

The Create Stream Network task is the same as found in [GP Service example: Stream network](#) with one notable exception: the output variable, Stream Network, has its Layer Symbology property set to `Stream Network.lyr`, as shown below.



## Publishing

`StoweStreamNetworksBasemap.mxd` is published as a map service.

`StoweStreamNetworkTasks.mxd` is published as a geoprocessing service:

1. In the **Catalog** window, navigate to your GIS Server, right-click, then choose **Add New Service**. Name the service `StoweStreamNetworkTasks` and choose **Geoprocessing Service** as the type.
2. Click **Next**.
3. Choose `StoweStreamNetworkTasks.mxd` as the source for your service.
  - Change the **Maximum Number of Record Returned by Server** to 10000. You need to change this because there is no result map service to draw the results, and the features are transported back to the client. If you leave the number of records to the default 1000, none of the tasks return the full dataset, since they all generate more than 1,000 records.
  - For testing purposes, check **Show Messages**.
4. Click **Next**. From this point on, you can accept the default values provided by the wizard.

## Using

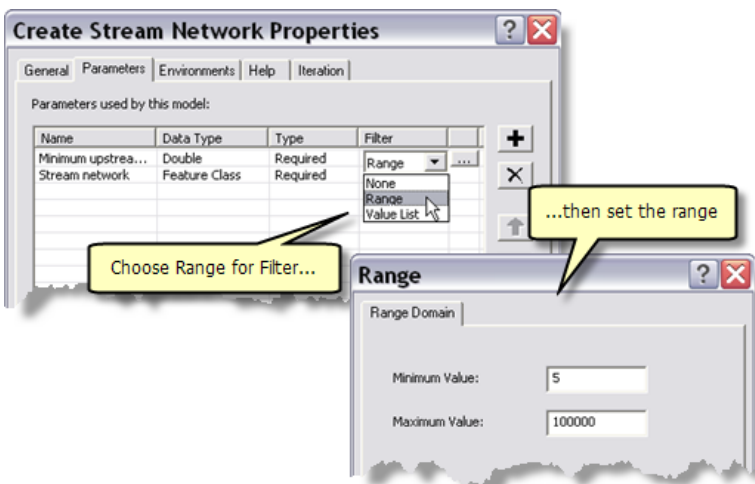
To use the service, start ArcMap with a blank document and add the `StoweStreamNetworksBasemap` map service and the `StoweStreamNetworkTasks` geoprocessing service. Open and execute both tasks. If you do not get results from either of the tasks, it is probably because the **Maximum Number of Record Returned by Server** parameter was not set to a large number as described above.

## Using a range filter

Model tool parameters can have filters, which are used to validate parameter values. For Create Stream Network, you can use a range filter to validate the Minimum upstream area in hectares parameter. Minimum areas less than 5 hectares produce thousands of stream segments and generate unusable output (at least for cartographic display purposes). You can use a filter to prevent processing for any values less than 5 hectares.

To set a range filter

1. In the Catalog window, right-click the Create Stream Network tool and click **Properties**. Alternatively, with Create Stream Network open in ModelBuilder, click **Model** in the main ModelBuilder menu, then click **Model Properties**.
2. On the **Properties** dialog box, click the **Parameters** tab.
3. Click the **Filter** cell in the `Minimum upstream area in hectares` parameters and choose **Range**. The **Range** dialog box will automatically open.
4. On the **Range** dialog box, enter the minimum and maximum values.



After making this change to the Create Stream Network model, the service must be restarted for the change to take effect.

With these changes, anytime a value less than 5 is entered, the task will display an error in its messages. The service must have **Show Messages** checked for the user to see the message.

# GP Service example: Clip and ship

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	ClipAndShip
Purpose	Using a polygon digitized by the user, this service clips layers from the study area into a file geodatabase, then creates a .zip file that can be downloaded by the user.
Services	Portland (Map service), ClipAndShip (Geoprocessing service).
Geoprocessing tasks	Extract Data Task.
Inputs	Layers to Clip, Area of Interest, Feature Format, Raster Format.
Outputs	A ZIP file containing the data.
Data	The example uses a number of datasets from the study area of Portland, Oregon.
Extensions	None.
Of note	Creates output.zip, a compressed file containing a file with the format specified. This file mimics the arrangement of the layers in the ArcMap document that is published as a geoprocessing service.

About this example

## Corresponding Folder

C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip contains the completed models and data.

## About this example

This example shows the detail steps of publishing and using a map and a geoprocessing service. The geoprocessing task lets you select the layers you want, digitize a polygon (or multiple polygons) of the area of interest, and select the output feature and raster data format. Data in the map is clipped to the area of interest, then bundled and shipped to the user—hence the name "clip and ship".

## Data

The study area for this example is a small area in the city of Portland, Oregon. Data includes places, transportation networks, hydrologic features, land records, and a hillshade raster of the study area. The data can be found in C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip\ToolData\Portland.gdb.

## Publishing

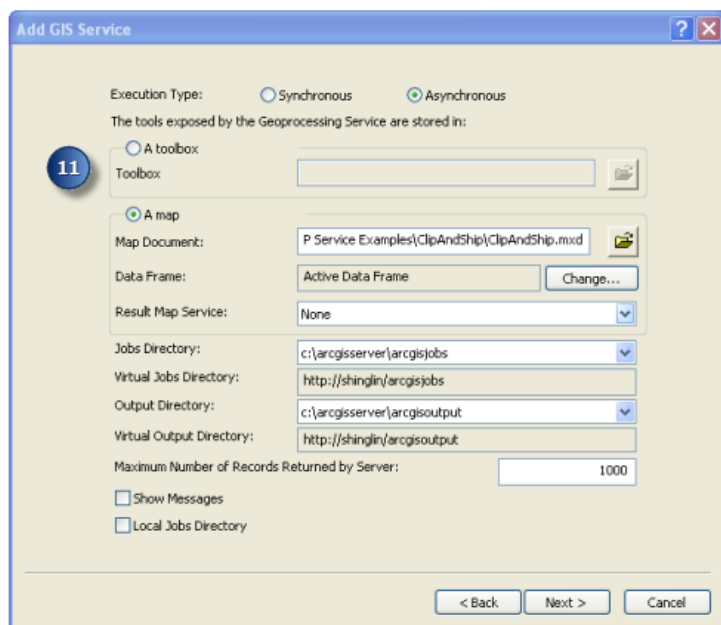
Steps:

You need to have an administrative connection to an ArcGIS server to publish services. To create an administrative connection, in the **Catalog** window, go to **GIS Servers > Add ArcGIS Server** and check **Manage GIS Services**. On the General panel, enter the **Server URL** and

**Host Name**, then click **Finish**. A server administrative connection with the host name will appear under GIS Servers.

1. From the **Catalog** window, navigate to C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip.
2. Right-click Portland.mxd and select **Publish to ArcGIS Server**.
3. In the first window, keep all defaults and click **Next**.
4. In the next panel, deselect all check boxes except **Mapping (always enabled)**. Click **Next**.
5. Click **Finish**. The map service Portland will be published to the ArcGIS Server. You will use it as a basemap later.
6. Open C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip\Portland.mxd in ArcMap.
7. In the **Catalog** window,
  - a. Expand **Toolboxes > System Toolboxes > Server Tools.tbx > Data Extraction > Extract Data Task**.
  - b. Drag the **Extract Data Task** tool to the ArcMap table of contents.

The tool layer Extract Data Task appears in the table of contents.
8. Save the map document containing the tool layer as ClipAndShip.mxd.
9. In the **Catalog** window, right-click the connection to your ArcGIS Server and select **Add New Service**.
10. In the **Add GIS Service** panel, type **ClipAndShip** as the service name and select **Geoprocessing Service** as the Type. Click **Next**.
11. In the next panel, check **A Map** as your source file. Enter the path to the ClipAndShip.mxd map document that you created above and click **Next**.



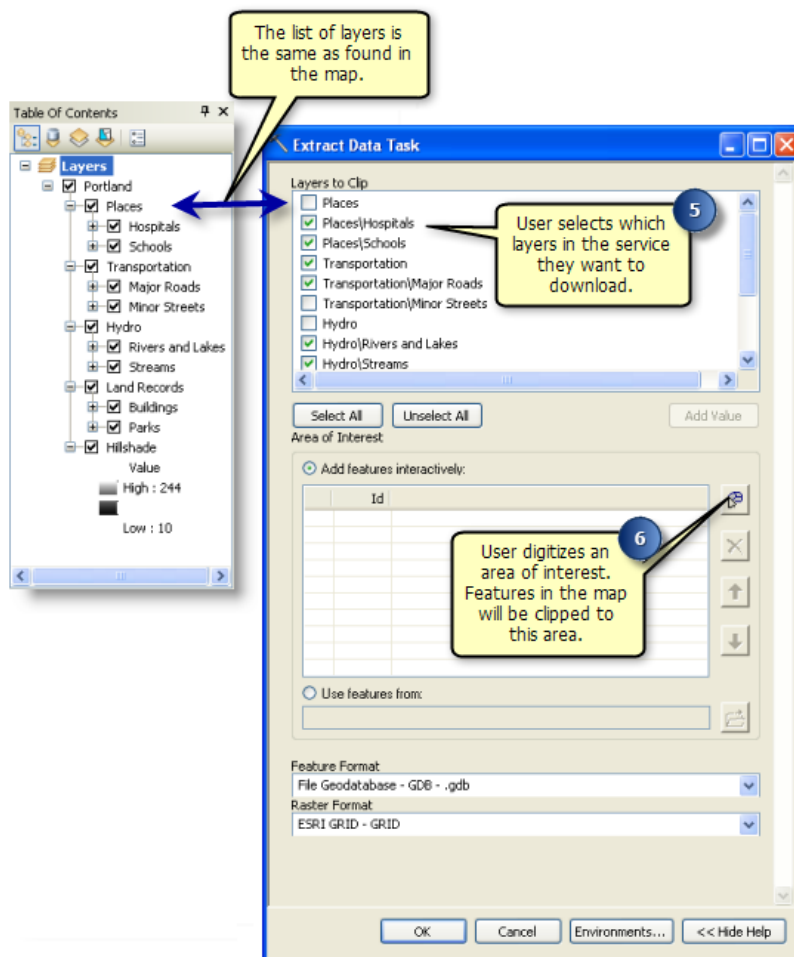
12. For any remaining panels, leave the default value and click **Next** until the last panel, where you click **Finish**. You should see the service ClipAndShip under your ArcGIS Server connection.

## Using

The following steps show how to use the map and geoprocessing services that you previously published.

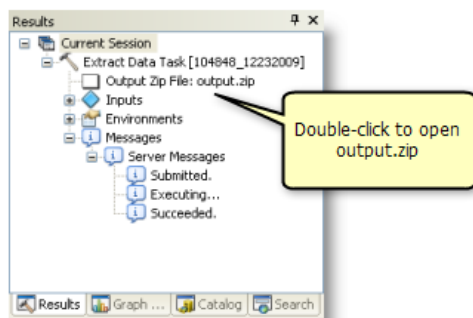
Steps:

1. Open a new ArcMap document.
2. In the **Catalog** window,
  - a. Navigate to your server connection under GIS Servers, select the map service Portland, and drag it into the ArcMap table of contents.
  - b. Expand the Portland map service in the table of contents, and you will see all the layers belonging to the Portland map service.
3. In the **Catalog** window, navigate to your server connection under GIS Servers, find the geoprocessing service ClipAndShip, and expand it. The task Extract Data Task appears.
4. Double-click the task **Extract Data Task** to open the task dialog box.
5. At the top of the task panel is a list of **Layers to Clip**. Note that the layer names in the check box are the same as the layers of the Portland map service in the table of contents. Check the box next to each layer that you want to be included in your download.



6. The next parameter of the Extract Data Task is the **Area of Interest**. This parameter is used to digitize an area of interest that will be used to clip each of the Layers to Clip. Click **Add Feature**, then digitize a polygon as an area of interest.
7. Other input parameters, **Feature Format** and **Raster Format**, can be modified or left as default.
8. Click **OK** to run the geoprocessing task.  
The output is returned from the server and written to the location set in your geoprocessing scratch workspace environment.

When the task completes, open the **Results** window to view the results of the task. Double-click output.zip to open the compressed file, then extract the data to a location of your choice.




## Customizing the GP Service source model

The previous example uses the model tool [Extract Data Task](#) from the Server toolbox. If you copy the model tool to a new custom toolbox, it can be opened in ModelBuilder and edited. The Extract Data Task model contains input variables Spatial Reference and Customized Spatial Reference Folder, which are not exposed as model parameters. Spatial Reference has a default value, Same as Input, which means the spatial reference of the output files is the same as the input layers. If you want to get a spatial reference for the output files that is different from that of the input files, you can set Spatial Reference as an input parameter so that when the model tool dialog box is opened, a different spatial reference can be specified. The variable Customized Spatial Reference Folder is an optional variable allowing you to specify a path that saves the custom and/or standard spatial reference.

In addition to the two Spatial Reference variables, which can be modified to customize the Extract Data Task, the existing model parameter **Feature Format** can also be modified to contain additional formats you need.

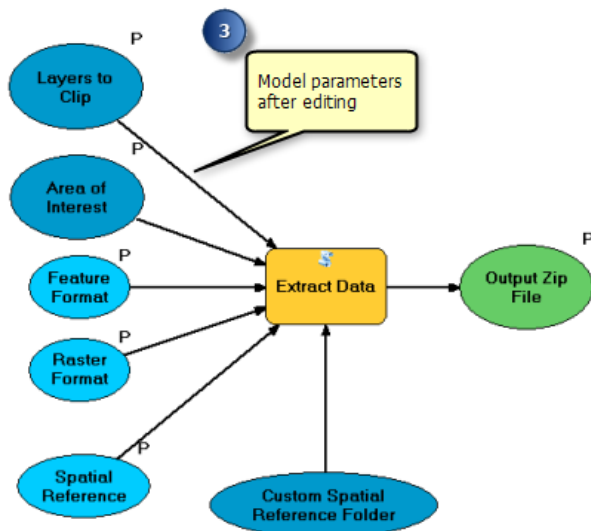
The following steps show how to customize the Extract Data Task by exposing the variable Spatial Reference as a parameter, adding a path for the custom spatial reference folder, and adding new output formats for the **Feature Format** parameter.

 **Note:** A toolbox named ExtractPortlandTbx that contains a customized model ExtractPortlandData is saved under `C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip`. The model is the finished product after performing the following steps.

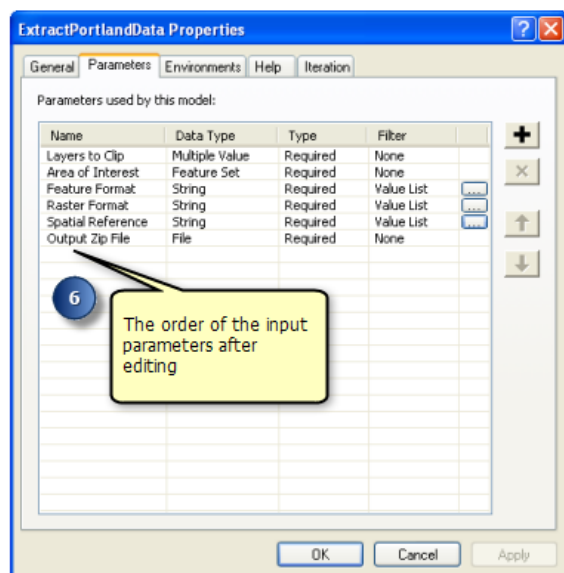
Steps:

1. The first step is to create a new toolbox and model. In the **Catalog** window, go to `C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip`.
  - a. Right-click the folder and select **New > Toolbox**.
  - b. Rename the toolbox ExtractPortland.
2. In the **Catalog** window,
  - a. Expand **Toolboxes > System Toolboxes > Server Tools.tbx > Data Extraction > Extract Data Task**.

- b. Select the Extract Data Task, right-click the model, then select **Copy**.
  - c. Go to ExtractPortland.tbx that is created in step 1b, right-click and select **Paste**. The model Extract Data Task is copied to the toolbox.
  - d. Right-click the model, select **Rename**, then rename the model ExtractPortlandData.
3. Right-click the model ExtractPortlandData and select **Edit** to open the model in ModelBuilder. In the model, right-click the variable Spatial Reference and select **Model Parameter**. The letter P appears on the upper right-hand corner of the variable so it is now a model parameter.



- a. In Windows Explorer, go to C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip. Right-click the folder and select **New > Folder**.
  - b. Rename the new folder SpatialReference.
  - c. Go to <ArcGISInstallDirectory>\Desktop10.0\Coordinate Systems\Projected Coordinate Systems, copy the State Plane folder.
  - d. Paste it into the new SpatialReference folder.
5. Right-click the model ExtractPortlandData and select **Edit**. Double-click the variable Custom Spatial Reference Folder and enter the path to the new SpatialReference folder (C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip\SpatialReference), then click **OK**.
6. With the ExtractPortlandData model still open in ModelBuilder,
  - a. Go to the **Model menu > Properties > Parameters** tab. The parameter **Spatial Reference** appears at the bottom of the parameter list.
  - b. Reorder the parameters so the output parameter is the last parameter. Highlight the parameter **Output Zip File** and click the button to move it to the bottom of the list.



7. Still in the **Parameters** tab,
  - a. Highlight the variable **Spatial Reference**, then change the **Filter** value to Value List.
  - b. Enter the values Same As Input, NAD 1983 StatePlane Oregon South FIPS 3602 (US Feet), and WGS1984 to the value list.
  - c. Click **OK** to close the Value List dialog box.
  - d. Click **OK** to close the model Properties dialog box.
  - e. Save the model and exit ModelBuilder.

The **Feature Format** parameter can also be customized by adding more data types to the list of formats. [Learn more about the different data formats supported by the Data Interoperability extension.](#)

8. Save the customized model and add it as a tool layer in a new ArcMap document that contains all the layers from Portland.mxd.
9. Refer to the instruction in the [Publishing](#) and [Using](#) sections above to publish and use the customized model as a geoprocessing service.

# GP Service example: Data on demand

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

Folder	DataOnDemand
Purpose	Using a polygon digitized by the user, clips datasets in a file geodatabase, outputting shapefiles, then creates a .zip file that is e-mailed to the user.
Services	PortlandDataMapService (map service), DataOnDemand (geoprocessing service).
Geoprocessing tasks	ClipZipAndEmail
Inputs	Area of interest (polygon Feature Set) and an e-mail address to send the data.
Outputs	aoizip.zip, a compressed file containing the data.
Data	The example uses a small dataset of the city of Portland, Oregon.
Extensions	None.
Of note	This service is hosted on ESRI's sampleserver—see note below. This is a clip and ship service, as described in the <a href="#">clip and ship example</a> .

About this example

## Corresponding Folder

C:\arcgis\ArcTutor\GP Service Examples\DataOnDemand contains the tools and data.

## About this example

This DataOnDemand service is another example of a clip and ship service. Before exploring this service, you should first read the [clip and ship example](#), since the features and capabilities of this service are compared with that service, as described briefly in the table below. If you are building your own clip and ship service, you may want to combine features and capabilities of both services into your service.

Clip and Ship example	This example
User gets to choose which layers to download.	A fixed set of data is downloaded.
The spatial reference of the output data can be specified.	Spatial reference cannot be specified—it is set to the spatial reference of the datasets being clipped.
Output format can be specified.	Only shapefiles are output.
Uses models, scripts, and tool layers. The service is published using a map document.	No models are used, only scripts. The toolbox is published rather than a map document.
Layers from the map document are clipped.	Datasets are clipped. (Because there is no source map document containing layers, layers cannot be used, only datasets.)
Area of interest (the clip polygon the user digitizes) is not downloaded.	Area of interest is downloaded.
Output cannot be e-mailed.	Output can be e-mailed by specifying an e-mail server name in the ClipZipAndEmail script (source file is DataOnDemand/Scripts/zipandemail.py).

Comparison between the Clip and Ship example and this DataOnDemand example

Other features of this service include


- An ArcMap document that views the downloaded data is included in the .zip file.
- The Python scripts demonstrate many useful techniques such as the following:
  - Finding data relative to the script location
  - Adding a toolbox and using its tools
  - Importing a script and calling routines in the imported script
  - Using system functions to copy the map document

## This service is hosted by ESRI

This service is hosted on ESRI's ArcGIS Online servers. You can try out this service as follows:

1. Add `http://sampleserver1.arcgisonline.com/arcgis/services` as an ArcGIS server.
2. Add `http://sampleserver2.arcgisonline.com/arcgis/services` as an ArcGIS server.
3. In ArcMap, add the `Portland/Portland_ESRI_LandBase_AGO` map service from `sampleserver1`.
4. Add the `Portland/ESRI_CadastralData_Portland` geoprocessing service from `sampleserver2` to ArcToolbox.
5. Expand the `ESRI_CadastralData_Portland` toolbox and execute the `ClipAndShip` task.

The scripts and tools in the `DataOnDemand` folder are the same as those used by the `ClipAndShip` task in the `ESRI_CadastralData_Portland` geoprocessing service. The data used for this example, found in `DataOnDemand/ToolData/Portland.gdb`, is a small subset of the data used in the `Portland_Portland_ESRI_LandBase_AGO` map service.

 **Note:** The map and geoprocessing services found in `sampleserver1` and `sampleserver2` may change in the future. There is no guarantee that the services described above are always available.

## Data

The data is of a small area in the city of Portland, Oregon, and is found in `C:\arcgis\ArcTutor\GP Service Examples\DataOnDemand\ToolData\Portland.gdb`.

The `ClipZipAndEmail` tool uses a `Feature Set` variable, which in turn needs a schema to define the feature types and fields. The schema can be found in `C:\arcgis\ArcTutor\GP Service Examples\DataOnDemand\ToolData\Templates.gdb`.

The `ToolData` folder also contains `Mapofzip.mxd`, which is included in the ZIP file and displays the clipped and shipped data.

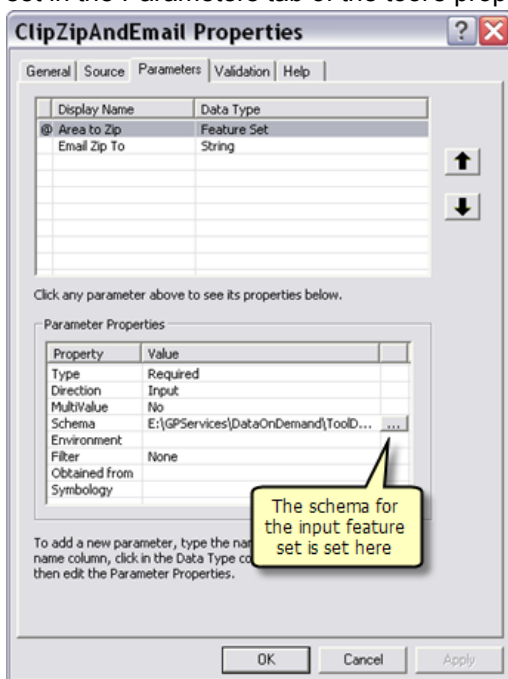
## Scripts

The `DataOnDemandTools` toolbox contains one script tool, `ClipZipAndEmail`. The source for this script tool is `DataOnDemand/Scripts/zipandemail.py`.

Before using the ClipZipAndEmail tool, you must edit the code and provide the name of your e-mail server. (You can either edit the Python source directly in an application like PythonWin or right-click the script tool and click Edit.) Your system administrator should be able to supply you with the name of your e-mail server.

Some important properties and features of this script are described below:

- The **Area to Zip** parameter is a feature set and, therefore, needs to have a schema. The schema is set in the Parameters tab of the tool's properties.



- In the ClipZipAndEmail script, the UtilityTools toolbox is added, and the Zip script tool from that toolbox is used. See the zipData() routine in the script.
- In the ClipZipAndEmail script, the emailZip() routine imports the send\_mail() routine found in the sendemail.py script (found in DataOnDemand/Scripts), as follows:

```
from sendemail import send_mail
```

## Publishing

PortlandDataMapService is published as a map service.

The DataOnDemandTools toolbox is published as a geoprocessing service.

## Configuring the service

To configure this service for your data, you will need to edit the ClipZipAndEmail script tool. The source for this script tool is DataOnDemand/Scripts/zipandemail.py. You can either edit the Python source directly in an application like PythonWin or right-click the script tool and click **Edit**.

You will need to change the location of data and the list of datasets. In the main routine,

```
if __name__ == '__main__':
```

Locate the definition of the dataloc variable and change it:

```
global dataloc; dataloc = os.path.dirname(sys.path[0]) + g + "tooldata" + g + "portland.gdb" + g
```

The location of the data is relative to the location of the script.

[Learn more about using the script location to build paths](#)

Next, change the list of datasets to clip, found in this code snippet:

```
ds = ["Streets" + g + "streets", \
      "Water" + g + "StreamRoute", "Water" + g + "floodplain", "Water" + g + "riv_fill", \
      "Transit" + g + "railroad", \
      "Census" + g + "blockgrp", \
      "Develop" + g + "Buildings", \
      "Land" + g + "zoning", "Land" + g + "Parks", \
      "Places" + g + "schools", "Places" + g + "hospital"]
```

Finally, you will need to provide the name of your e-mail server in the sendemail.py script. The code you need to modify is near the top of the script:

```
def send_mail(send_from, send_to, subject, text, f=""):
    assert type(send_to)==list

    # Provide the name of your email server below
    #
    server = "ouremailserver.somewhere.com"
```

# GP service example: Selecting data

**Complexity:**  
Beginner

**Data Requirement:**  
ArcGIS Tutorial Data Setup

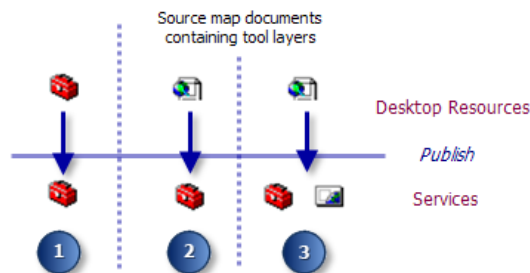
Folder	SelectingData
Purpose	Demonstrates various ways to select data by attribute query or location query.
Services	<ul style="list-style-type: none"> <li>• <code>SelectingData</code> (geoprocessing service with a source map document)</li> <li>• <code>SelectingDataRMS</code> (geoprocessing service with result map service)</li> </ul>
Geoprocessing tasks	Several (See the <a href="#">models</a> below.)
Inputs	Varies by model; usually a query string or a feature set.
Output	Selected data
Data	Uses a dataset of a small area in the city of Portland, Oregon.
Extensions	None

About this example

## About these services

A common task for geoprocessing services is to select a subset of data based on an attribute query or a spatial query. The models in this example demonstrate several useful ways of selecting data. The primary geoprocessing tools used in these models are [Select Layer By Attribute](#) and [Select Layer By Location](#). Both tools take layers as input. Their output is the updated input layer containing the selected features.

Because [Select Layer By Attribute](#) and [Select Layer By Location](#) output layers (as opposed to datasets), you need to be aware of how ArcGIS Server handles tasks that output layers. Recall that there are three geoprocessing service configurations, shown below:



- 1 **Geoprocessing service**  
Publishing a toolbox to create a geoprocessing service. Each tool in the toolbox becomes a task. Tasks can use datasets on disk. *Outputs of tasks are drawn by the client.*
- 2 **Geoprocessing service with a source map document**  
Publishing a map document containing tool layers to create a geoprocessing service. Each tool layer becomes a task. Tasks can access layers in the source map document as well as datasets on disk. *Outputs of tasks are drawn by the client.*
- 3 **Geoprocessing service with result map service**  
Publishing a map document containing tool layers to create a geoprocessing service. Each tool layer becomes a task. Tasks can access layers in the source map document as well as datasets on disk. *Outputs of tasks are drawn by the result map service.*

In the first two configurations, when ArcGIS Server executes a task that outputs a layer, it reads the selected features from the layer and transports the selected features back to the client.

In the third configuration, there are two services—the geoprocessing service and the result map service. The two services execute independently of each other. When the task executes, ArcGIS Server executes the geoprocessing task first, then executes the result map service, which draws the output of the geoprocessing service, sending the map image of the output data back to the client. Because of this execution order, the result map service needs datasets on disk produced by the geoprocessing service. This means that the output of the tasks in the geoprocessing service must be datasets, not layers. Layers, allowed in the first two configurations, do not work with the result map service configuration.

In this example, you will find two toolbox/map document pairs; SelectingData (for publishing as a geoprocessing service with a source map document) and SelectingDataRMS (for publishing as a geoprocessing with a result map service). The tools in the SelectingDataRMS toolbox result in datasets on disk, while the tools in the SelectingData toolbox result in layers. In general, the tools in the SelectingDataRMS toolbox take a bit longer to execute since they must copy their data to disk.

Which service you choose to publish—SelectingData or SelectingDataRMS—is up to you. Typically, you publish a result map service when any of the following are true:

- The selected set of features can be large and you do not want to transport a large number of features back to the client.
- You want to protect your data and only let the client see a map image of the data.
- You have advanced cartography that can only be drawn by ArcMap, not other client applications.

## Data

The data is of a small area in the city of Portland, Oregon, and is found in `C:\arcgis\ArcTutor\GP Service Examples\SelectingData\ToolData\Portland.gdb`. This file geodatabase has been

compressed to save space using the [Compress File Geodatabase Data](#) tool. You cannot make edits to a compressed file geodatabase. If you need to make edits, use the [Uncompress File Geodatabase](#) tool.

Several of the models use feature set and record set variables, which need schemas to define their feature types and fields. These schemas can be found in `C:\arcgis\ArcTutor\GP Service Examples\SelectingData\ToolData\Templates.gdb`.

## Overview of the models

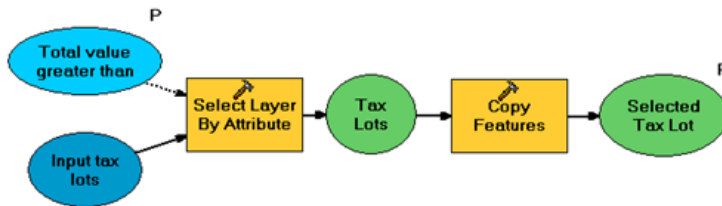
### SelectingData toolbox and map document

Toolset	Tool	Description
<b>Select By Attribute</b>		
	Select Tax Lots With Value Greater Than	Selects all tax lots whose total value is greater than the value the user enters.
	Select Tax Lots With Value Greater Than (with summary)	Same as above, but additionally summarizes the total value of all selected lots.
	Select Tax Lot By Address (attribute method)	Given an address, selects the tax lot.
	Select Neighborhood	Selects a neighborhood from a drop-down list.
	Select Layer By Area	Allows the user to first select a layer, then selects features based on their area.
	Mailing List	Selects a tax lot by address, selects all tax lots within a given distance, then produces a mailing list for all selected tax lots.
<b>Select By Location</b>		
	Select Tax Lot By Address (location method)	Given addresses, creates a point feature for each address and uses these point features to select tax lots.
	Select Tax Lots By Neighborhood	Selects tax lots based on a selected neighborhood polygon.
	Select Tax Lots By Neighborhood (optimized)	Same as above, but uses a preprocessed dataset to perform the selection.

SelectingData toolbox

## SelectingDataRMS toolbox and map document

The tools in the SelectingDataRMS toolbox perform the same work as the tools in the SelectingData toolbox. The main difference is that the SelectingDataRMS tools output feature classes that can be drawn by the result map service. The [Copy Features](#) tool is used to copy the features from the layer to the feature class. Illustrated below is the **Select Tax Lots With Value Greater Than** model using the Copy Features tool. All the tools in the toolbox follow a similar pattern.



Select Tax Lots With Value Greater Than (for result map service)

## Attribute and spatial indexes

Whenever you query a particular field often, as these services do, the speed of the queries can be increased by creating indexes on the query fields. For example, the Select Tax Lots With Value Greater Than task queries the TOTALVAL field, so to optimize the queries, TOTALVAL has an attribute index. You can use the [Add Attribute Index](#) tool to create attribute indexes.

Similarly, if you are performing spatial queries, spatial indexes will increase how quickly features can be located. Spatial indexes are automatically created and maintained for feature classes in a geodatabase, but not so for shapefiles.

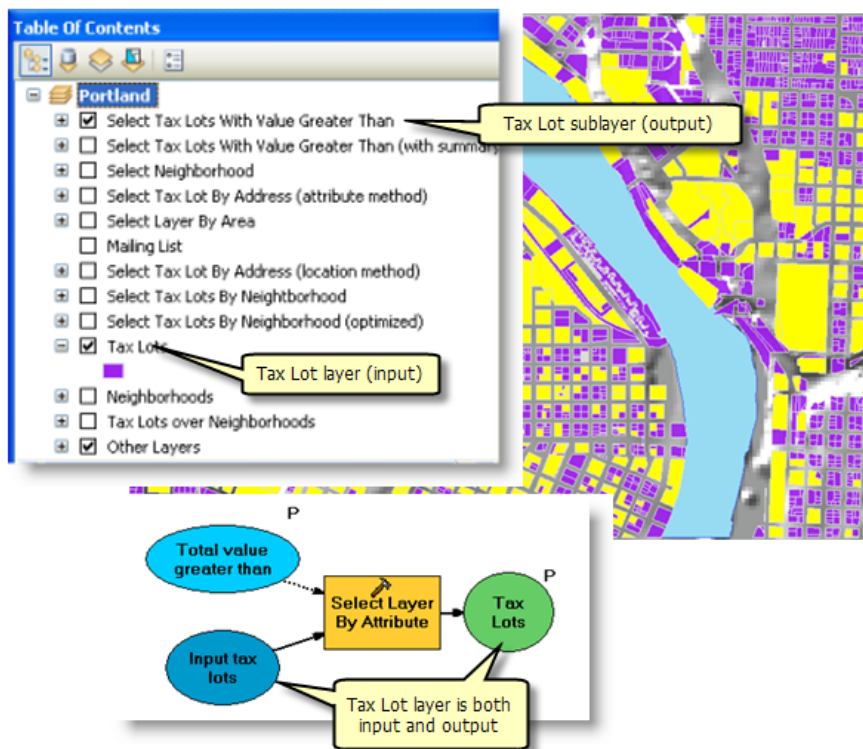
[Learn more about geodatabase spatial indexes](#)

[Learn more about shapefile spatial indexes](#)

[Learn more about the Add Spatial Index tool](#)

## Working with layer selections and tool layers

The SelectingData map document contains tool layers for each of the tools listed above. The sublayer found in the tool layer (the output of the tool) is the same as the input layer—it is not an independent layer. Because it is not independent, if you change the properties of either the original input layer or the output sublayer, the other layer's properties change as well. As illustrated below with the Select Tax Lots With Value Greater Than tool, both the input and output layer are Tax Lots. If you right-click the Tax Lot layer (the input layer) in the ArcMap table of contents and change the symbology, the symbology on both layers will change.



When you work with layer selections (as these tools do), there are a few guidelines to remember:

- When you create the tool layer in ArcMap (before publishing), the sublayer (the output of the model) will always be a layer named the same as the input layer, regardless of the output variable name in the model. The symbology will be the same as the input layer.
- When you execute the server task (after publishing), the output layer will be the name of the model variable.

## Building expressions using variable substitution

Selection expressions usually require an input from the user. As shown in the Select Tax Lots With Value Greater Than model below, the user's input is the Total value greater than variable. This is a double variable created as follows:

Steps:

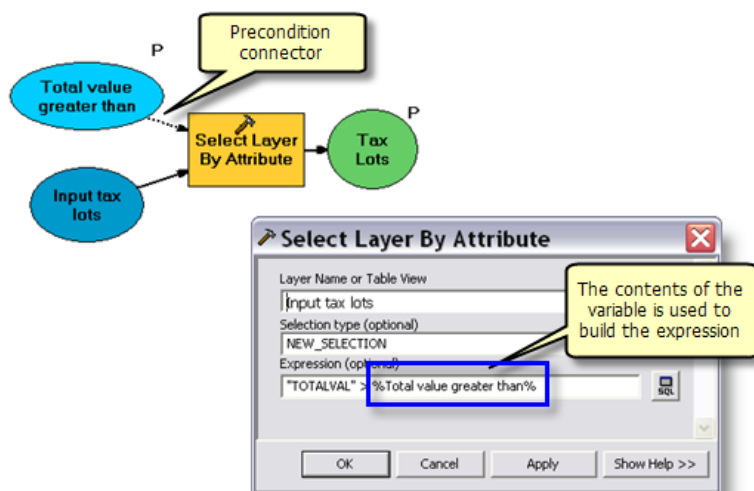
1. In ModelBuilder, right-click the canvas and click **Create Variable**
2. On the **Create Variable** dialog box, choose **Double** as the variable type.
3. Click **OK**.
4. Rename the variable.
5. Open the variable and provide a default value, if desired.

When creating the expression on the Select Layer By Attribute tool dialog box, you use percent signs (%) around the variable name.

Although it is not required, it is good practice to make the variable a precondition to the process that substitutes the variable. To make a precondition, follow these steps:

Steps:

1. In ModelBuilder, right-click a tool and click **Properties**.
2. Click the **Preconditions** tab.
3. Choose the variable or variables that are a precondition to tool execution.

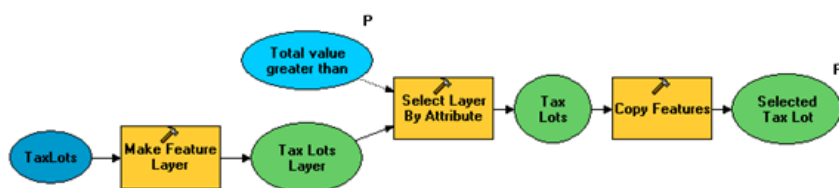


## Selection method

Both the Select Layer By Attribute tool and the Select Layer By Location tool have several options for selection. For example, instead of creating a new selection, you can add or remove features from the current selection. The Select Layer By Location tool lets you establish a spatial relationship, such as CONTAINS, WITHIN, and INTERSECT. The example models make use of just a few of the options. Consult the [Select Layer By Location tool documentation](#) for more information.

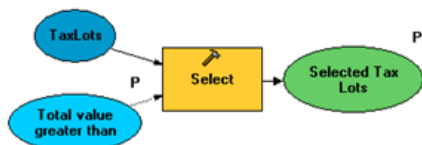
## Selecting data without map layers

You do not need to use layers from a map document to make use of the Select Layer By Attribute and Select Layer By Location tools. You can create a feature layer from a dataset by using the [Make Feature Layer](#) tool, as shown below.



Using Make Feature Layer in a model

You can also use the [Select](#) tool to create a dataset of selected features from a dataset—no layer is required.



Using the Select tool

The Select Tool performs attribute selection. If you need to perform spatial selection, consider using one or more of the following tools:

- The Make Feature Layer tool
- Tools from the [Analysis toolbox](#), such as [Clip](#) or [Spatial Join](#)

## SelectionUtilities

This toolbox contains several utility tools to support the tasks.

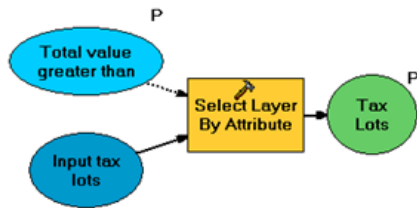
Tool	Description
Check Match Results	Checks the output of the Geocode Addresses tool and outputs an error if addresses do not match.
Get 1 Field Value	Reads a field value from the first record of a table and outputs it to a geoprocessing model variable.
Get 2 Field Values	Same as above, but reads two fields and outputs two variables.
Get 3 Field Values	Same as above, but reads three fields and outputs three variables.
Overlay Tax Lots And Neighborhoods	Creates the dataset used by the Select Tax Lots By Neighborhood (optimized) model above.
Print Tax Lot Mailing Labels	Given a selected set of tax lots, prints a file of mailing labels.

Utility tools

## Notes on the models

### Select Tax Lots With Value Greater Than

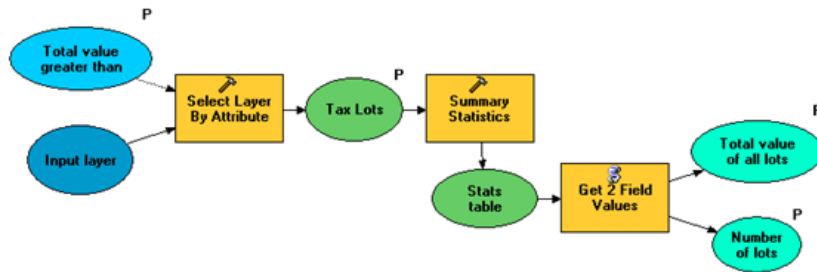
This is the simplest of the models. It selects all tax lots whose total value is greater than the input value. As [noted above](#), it uses variable substitution to build the select expression.



Select Tax Lots With Value Greater Than

## Select Tax Lots With Value Greater Than (with summary)

This performs the same work as the model above, then adds the [Summary Statistics](#) tool to sum the total value of all selected lots. Summary Statistics outputs a table that is read by the Get 2 Field Values tool, a utility tool found in the SelectionUtilities toolbox included with this example. This is a script tool that outputs the values of two fields, Total value of all lots and Number of lots. These values, along with the selected tax lots, are returned to the client.

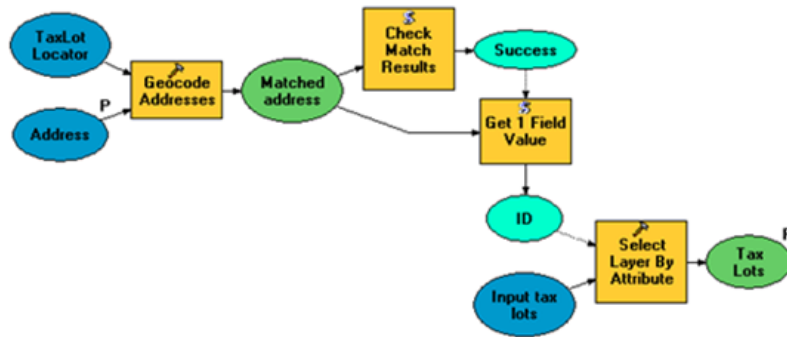


Select Tax Lots With Value Greater Than (with summary)

The output of Summary Statistics is written to an in-memory table rather than a table on disk. Writing tables and feature classes to memory is fast compared to writing to disk.

## Select Tax Lot By Address (attribute method)

Selecting features by address is a common task. This model shows one method of selecting features by address (another method is shown [below](#)).



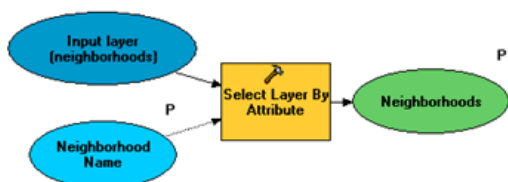
Select Tax Lots By Address (attribute method)

Model element	Description
TaxLot Locator	An address locator for the TaxLots feature class. This locator uses the U.S. Single Address style. <a href="#">Learn more about address locators</a>
Address	This is a record set variable. The template is <code>ToolData/templates.gdb/Address</code> . For some sample addresses in the study area, see <code>ToolData/sample addresses.txt</code> .
<a href="#">Geocode Addresses</a>	Takes a table containing addresses and outputs a point feature class of the address location. A status field is also output. This field will contain an <b>M</b> if the address was found.
Check Match Results	A custom script tool found in the SelectionUtilities toolbox included with this example. It checks that the address was successfully match.
Success	Output of Check Match Results. True if there was one address input and the address matched.
Get 1 Field Value	A custom script tool found in the SelectionUtilities toolbox included with this example. It retrieves the unique object ID of the tax lot matching the address.
Select Layer By Attribute	Selects the tax lot with the object ID output by Get 1 Field Value.

Model elements

## Select Neighborhood

This model selects a polygon from the Neighborhood layer.



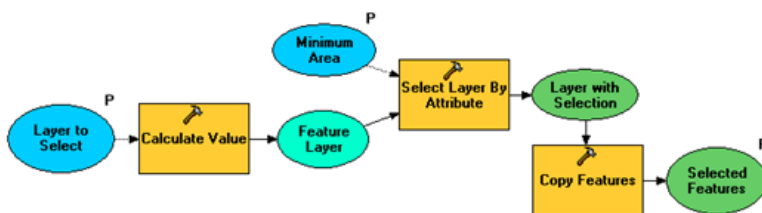
Select Neighborhood

This model lets you pick a neighborhood from a list of neighborhoods. The **Neighborhood Name** variable is a string variable that has a **Value List** filter containing all neighborhood names. To create or change a filter, open the model properties dialog box and click the **Parameters** tab, shown below.

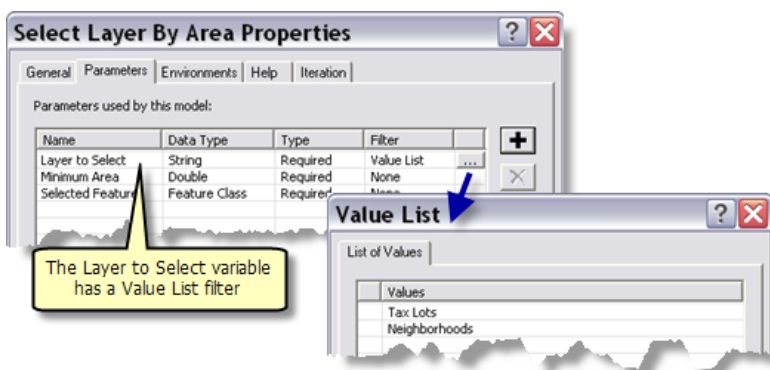


## Select Layer By Area

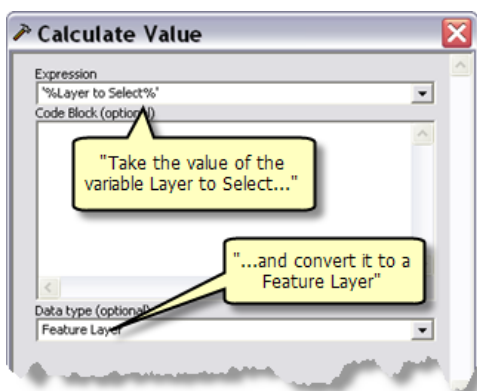
This model allows the user to first select a layer, then specify a minimum area for selection.



The main feature of this model is how it allows you to first choose a layer. The **Layer to Select** variable is a string variable that has a **Value List** filter containing Tax Lots and Neighborhoods. To view the filter, open the **Model Properties** dialog box and click the **Parameters** tab, as illustrated below.



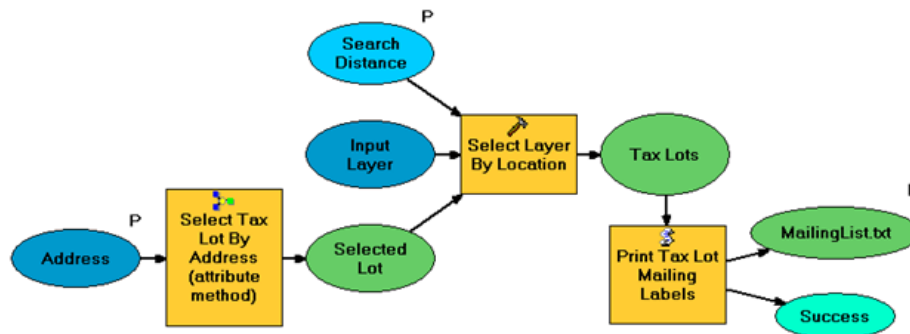
The **Calculate Value** tool takes the string and turns it into a feature layer variable, as illustrated below. This technique of turning a string into another data type is discussed in [Input and output data types](#). The output of Calculate Value is input to Select Layer By Attribute.



Finally, the selected features are copied to a feature class using the Copy Features tool. This final step is not required. (Copying the features helps to avoid confusion when creating and testing the tool layer.) The features are written to memory rather than to disk (writing features to memory is faster than writing to disk). In the SelectingDataRMS toolbox, the selected features have to be written to disk for the result map service to draw them.

## Mailing List

This model shows how to select nearby features based on a selected feature, as well as generating a simple text containing the addresses of the nearby features.



Mailing List

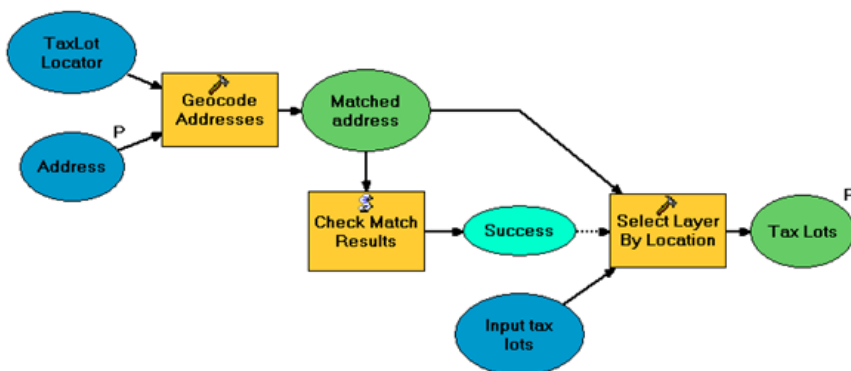
Model element	Description
Address	This is a record set variable. The template is ToolData/templates.gdb/Address. For some sample addresses in the study area, see ToolData/sample addresses.txt.
Select Tax Lot By Address (attribute method)	This is the model described <a href="#">above</a> .
Selected Lot	The selected tax lot. This is the Tax Lot layer.
Search Distance	Search distance to use.
Input Layer	The Tax Lot layer.
Select Layer By Location	Selects all tax lots within the distance specified, using Selected Lot as the center of the search.
Tax Lots	Tax lots within the search distance.
Print Tax Lot Mailing Labels	A custom script tool found in the SelectionUtilities toolbox.
MailingList.txt	A text file containing mailing addresses.
Success	True if every record has a mailing address. False if one or more of the records has an empty or corrupted mailing address.

Model elements

## Select Tax Lot By Address (location method)

This is the first of the tools found in the Select By Location toolset. It does the same work as the Select Tax Lot By Address (attribute method) tool, but instead of selecting the tax lot using the feature ID, it uses the output point feature of [Geocode Addresses](#) as input to Select Layer By Location .

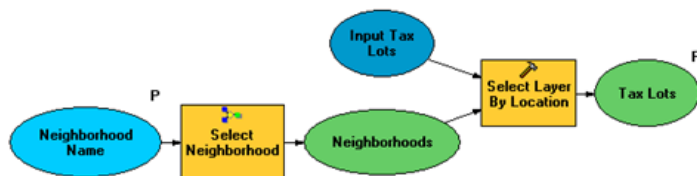
This is the method to use when the geocoded point comes from a different locator. For example, the locator may be based on a street dataset, and you need to use the geocoded point to select a nonstreet feature, such as a building, tax lot, or point of interest.



Select Tax Lot By Address (location method)

## Select Tax Lots By Neighborhood

Another common selection task is to use a feature from one dataset to select a feature from another dataset. This model uses the Select Neighborhood model described above to select a single neighborhood, then uses the Select Layer By Location tool to select all tax lots that fall within the neighborhood.



Select Tax Lots By Neighborhood

## Select Tax Lots By Neighborhood (optimized)

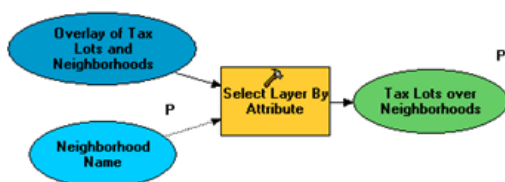
The methodology used by the Select Tax Lots By Neighborhood model above is acceptable for infrequent queries. That is, an analyst might use this method to find the tax lots for a quick onetime analysis. But geoprocessing services are usually tasks that are run repeatedly and often. Since selecting tax lots by a neighborhood is a task that will be used frequently, you want to optimize it.

For this optimization, the [Intersect tool](#) is used to assign the neighborhood name to each tax lot. The Overlay Tax Lots And Neighborhoods tool found in the SelectionUtilities toolbox shows how to use the Intersect tool to create a new dataset that contains all tax lots with all the neighborhood attributes. This model only needs to be executed once to produce the TaxLotsOverNeighborhoods dataset.



Overlay Tax Lots And Neighborhoods

Once TaxLotsOverNeighborhoods is created, all that is necessary is a simple attribute query to select all tax lots within a neighborhood.



Select Tax Lots By Neighborhood (optimized)

## Publishing

Publish Portland.mxd found in the [publishing section of GP service example: Clip and ship](#) as a map service. You will use this map service as a basemap for the services in this example. (The corresponding folder for this service is C:\arcgis\ArcTutor\GP Service Examples\ClipAndShip.)

Publish SelectingData.mxd as a geoprocessing service based on a source map document. Because features are sent back to the client, be sure to increase the maximum number of records that can be returned by the service to 11,000 (slightly more than the number of features in the Tax Lot layer).

[Learn more about publishing geoprocessing services with a source map document](#)

Publish SelectingDataRMS as a geoprocessing service with a result map service.

[Learn more about publishing geoprocessing services with result map services](#)

## Using

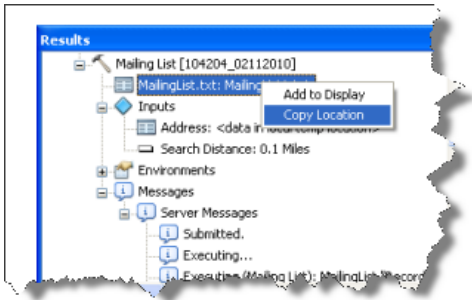
To use the services, open a new ArcMap session and add the Portland map service as a basemap.

Open the **Catalog** window and go to the ArcGIS Server user connection where you can see and execute all the tasks you published. Expand SelectingDataRMS service and execute the Mailing List task.

Open the **Results** window and perform the following steps:

Steps:

1. Expand the MailingList result.
2. Right-click the Mailing List task result and choose **Get Data**. The task result changes from <Data on Server> to MailingList.txt.
3. Right-click MailingList.txt and click **Copy Location**. (Since the result is a text file, Add to Display has no effect.)
4. Open your Internet browser and paste the location into the address bar. The contents of MailingList.txt appear.
5. Optionally, you can click the **Copy** tool and drag MailingList.txt from the **Results** window onto the input parameter of the Copy tool dialog box. The file is copied to a location of your choice.



# GP service example: Drive-time polygons

**Complexity:**  
Intermediate

**Data Requirement:**  
ArcGIS Tutorial Data Setup

**Data Path:**  
C:\ArcGIS\ArcTutor\GP Service Examples\DriveTimePolygons

**Goal:**  
Author, publish, and use a geoprocessing service that creates polygons based on drive time around points.

Folder	DriveTimePolygons
Purpose	Creates drive-time polygons around input points for the given drive-time values
Services	<ul style="list-style-type: none"> <li>• SanFranciscoBaseMap (map service)</li> <li>• DriveTimePolygonsService (geoprocessing service)</li> </ul>
Geoprocessing task	Calculate Drive Time Polygons
Inputs	A digitized point and a space-separated list of drive-time values in minutes.
Outputs	One drive-time polygon corresponding to each input drive-time value for all points.
Data	Uses a street network dataset for the San Francisco area provided in the ToolData folder.
Extensions	<a href="#">Network Analyst</a>
Of note	Demonstrates basic steps required to perform any network analysis workflow.

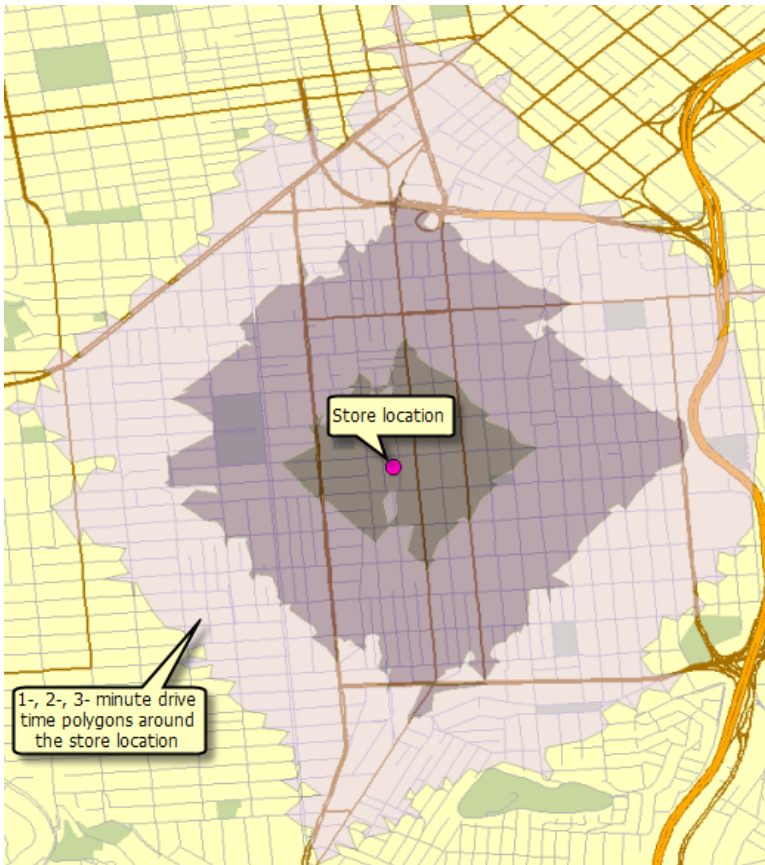
About this example

## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\DriveTimePolygons contains the completed model and data.

## About the Calculate Drive Time Polygons task

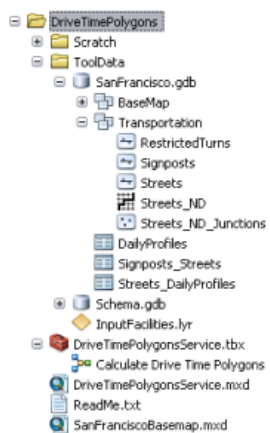
The primary purpose of the Calculate Drive Time Polygons task is to create drive-time polygons around user-specified points. A drive-time polygon is a region that encompasses all accessible streets that lie within a specified drive time from that point. Drive-time polygons can be used to evaluate accessibility of a point with respect to some other features. For example, one-, two-, and three-minute drive-time polygons around a grocery store location can be used to determine which people are most likely to shop at the store.



Example output from Calculate Drive Time Polygons task

## Data

The data for this example comes from C:\arcgis\ArcTutor\GP Service Examples\DriveTimePolygons.



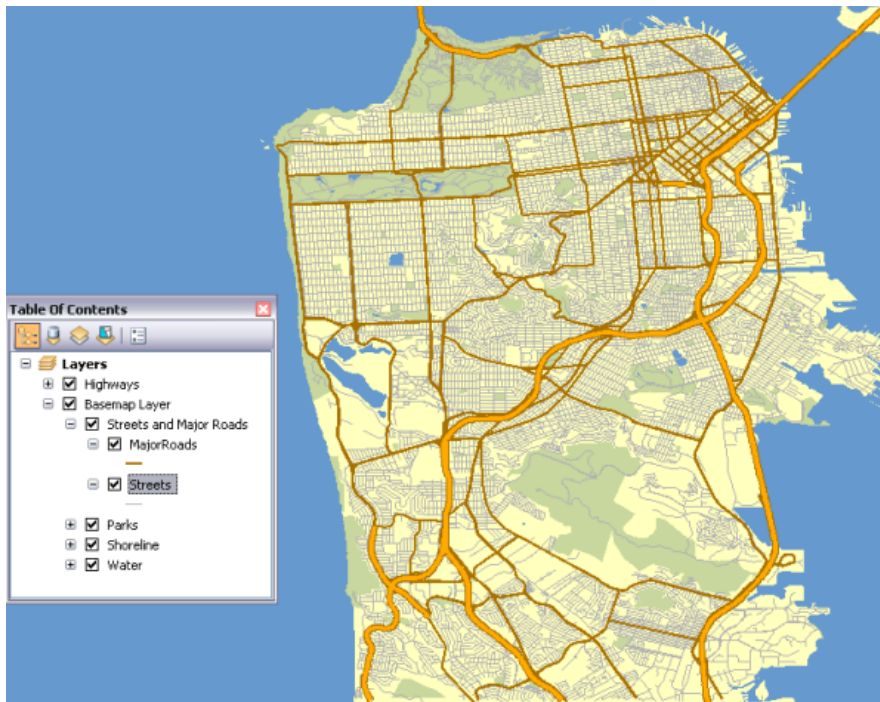
DriveTimePolygons folder contents

## Network dataset

The ToolData folder contains a file geodatabase, `SanFrancisco.gdb`. This geodatabase contains a network dataset, `Streets_ND`, within the Transportation feature dataset. This [network dataset](#) models the street network for the San Francisco area. It provides a [network attribute](#), `TravelTime` (among others), which indicates the time taken to travel each street segment in minutes.

## Basemap

The basemap layer within `SanFranciscoBasemap.mxd` has a layer, `Streets`, as illustrated below. This layer shows the extent of the network dataset. This means that this task can be used to determine drive-time polygons only in this extent.



San Francisco basemap showing the network dataset extent

`SanFranciscoBaseMap.mxd` is published as a map service.

## Toolbox and map document

The toolbox for the geoprocessing service is `DriveTimePolygonsService`, and the source map document for the service is `DriveTimePolygonsService.mxd`. `DriveTimePolygonsService.mxd` contains one source data layer, `Streets_ND` (the network dataset).

## Model

## Model overview

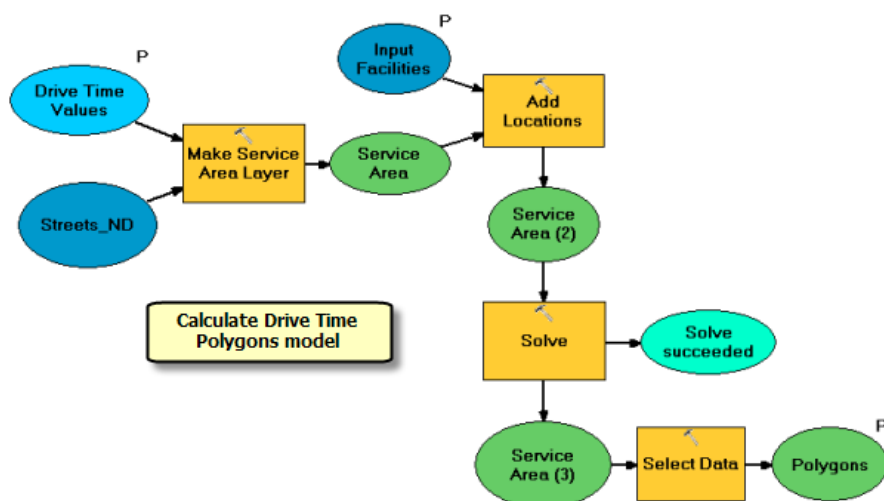
The Calculate Drive Time Polygons model is illustrated below. There are two input variables:

- Input Facilities are the centers of the drive-time polygons to be generated. (In network analysis, a facility is any fixed location on the network, such as a building or your current location.)
- Drive Time Values is a space-separated list of drive-time values in minutes.

The model creates a service area network analysis layer, adds the user-digitized points as facilities, and performs a solve to determine the drive-time polygons.

Element	Type	Description
Streets_ND	Network dataset layer	The network dataset layer.
Drive Time Values	String, input parameter	Space-separated list of drive-time values in minutes.
<a href="#">Make Service Area Layer</a>	Tool	Creates a service area network analysis layer. This layer contains both data and properties that determine how service areas will be calculated, along with the results of the calculation.
Service Area	Network analysis layer	Service area layer.
Input Facilities	Feature set (points), input parameter	Point features around which the drive-time polygons are determined.
<a href="#">Add Locations</a>	Tool	Adds the input points as facilities to the service area layer.
Service Area (2)	Network analysis layer	Service area layer with facilities.
<a href="#">Solve</a>	Tool	Calculates the drive-time polygons.
Service Area (3)	Network analysis layer	Service area layer containing the calculated drive-time polygons.
SolveSucceeded	Boolean	The derived output from the Solve tool that indicates if the solve was successful.
<a href="#">Select Data</a>	Tool	Selects the polygons sublayer from the service area layer.
Polygons	Feature layer, output parameter	The polygons layer from the Service Area (3) network analysis layer.

Model elements



### Network analysis workflow

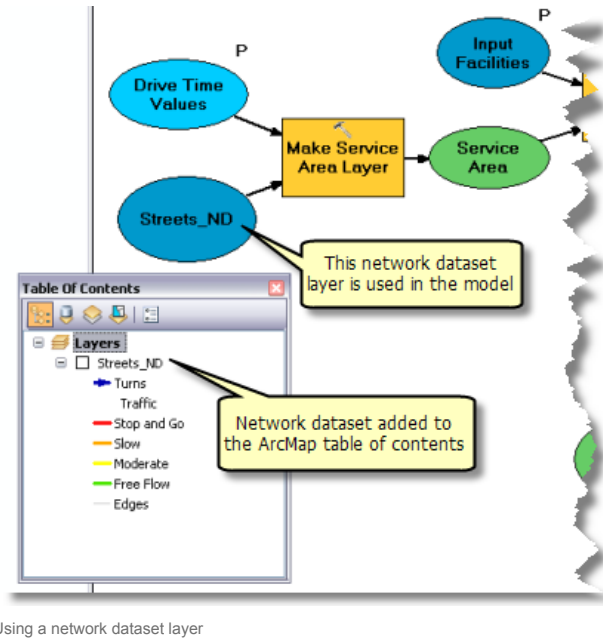
This model illustrates the four-step workflow that is common while performing any kind of network analyses.

1. Make a network analysis layer.
2. Add locations to one or more network analysis classes.
3. Solve the network analysis layer.
4. Use the results after the solve.

## Using the network layer

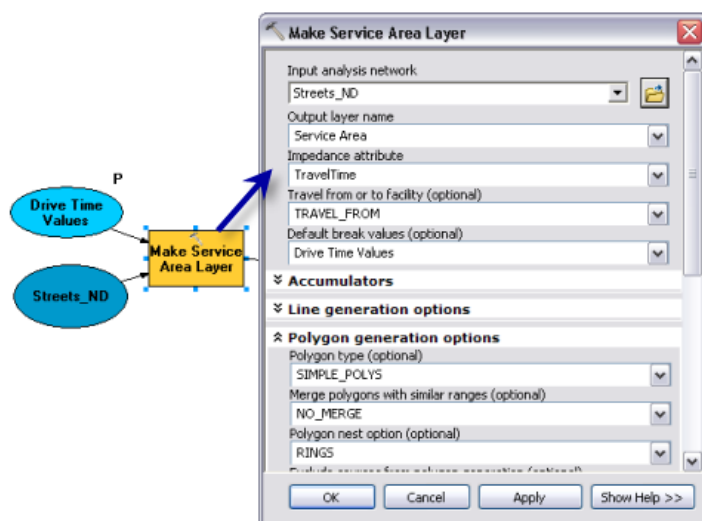
The network dataset for the San Francisco area is added to the map document

`DriveTimePolygonsService.mxd` as a network layer (`Streets_ND`). This layer is used in the model as an input variable for the [Make Service Area Layer](#) tool. Using a network layer greatly improves the overall model execution time since a connection to the network dataset is kept open by the network layer. Otherwise, if the network dataset is referenced from its disk location, a connection to the network dataset is made each time the model executes, which reduces the performance of the geoprocessing service created using the model.



## Model processes

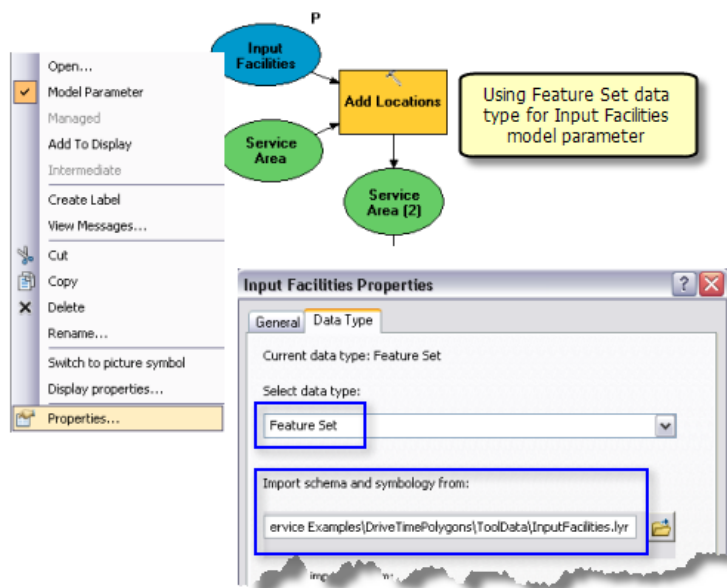
The [Make Service Area Layer](#) tool creates a new Network Analyst (NA) layer, *ServiceArea*, that stores the analysis properties, references the *Streets\_ND* network dataset layer used for the analysis, and stores the input facilities and the output polygons. The network dataset has a network cost attribute called *TravelTime* that specifies the travel time required to traverse each street segment. This attribute is used as an impedance attribute. The default break values are read from the Drive Time Values variable as a space-separated list of values.



Make Service Area Layer tool parameters

For this service, the `NO_MERGE` option was used to create overlapping polygons that do not merge for each facility. The `RINGS` option is used so that for each drive-time value, the polygons are drawn as rings. This results in polygons that encompass the area from the previous break up to the cutoff value for the break and do not include the area of the smaller breaks.

The [Add Locations](#) tool adds the user-digitized points as facilities to the service area layer. The Input Facilities parameter is a feature set data type so that the model can interactively accept the user-digitized points as facilities. The schema and symbology for the feature set are derived from the `InputFacilities.lyr` file found within the `ToolData` folder.



Using a feature set for input facilities

The **Solve** tool calculates the service area based on the options specified in the input service area layer and generates the drive-time polygons. The calculated polygons are written to the Polygons sublayer in the output service area layer.

Network Analyst layers are not [supported output parameter data types](#) for ArcGIS Server clients. So the **Select Data** tool is used to retrieve the Polygons sublayer from the service area layer. The Polygons sublayer is a feature layer data type.

## Tool layer

The **Calculate Drive Time Polygons** tool layer is created by dragging the model into the ArcMap table of contents. You should test the model before publishing as follows:

1. Create the tool layer.
2. Right-click the tool layer and click **Open**. The tool dialog box opens.
3. Enter a point and drive-time distances, such as 1 2 3, and click **OK**.  
The drive-time polygons are added as a sublayer to the tool layer.

The output of the **Calculate Drive Time Polygons** model is a feature layer. When a feature or raster layer is output by a model, the output layer is added to the tool layer as is, meaning symbology you define in the tool sublayer is ignored. You can try the following experiment to confirm this:

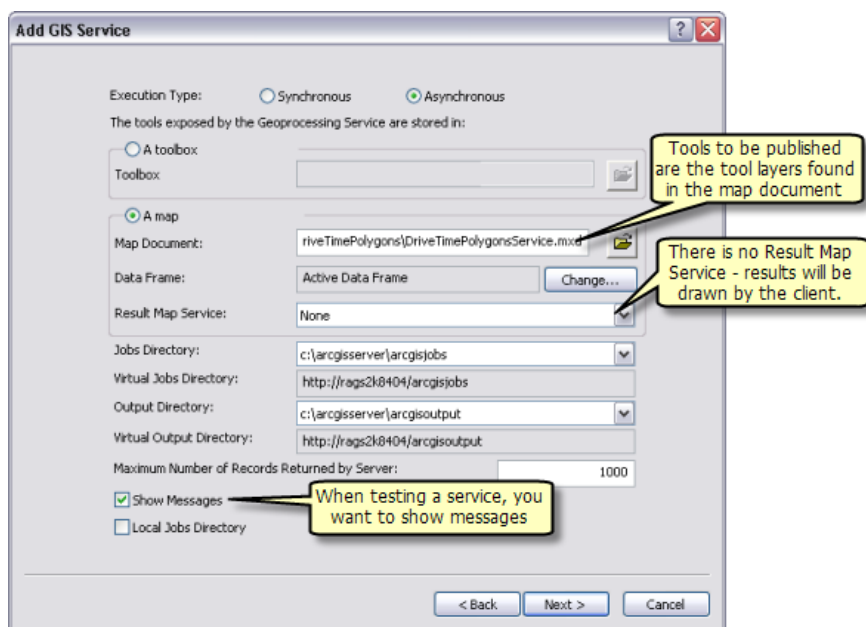
1. After opening and executing the tool layer, right-click the Polygons sublayer and click **Properties**.
2. Click the **Symbology** tab.
3. Change the symbology to a single symbol (a blue polygon fill, for example).  
The Polygons sublayer is now drawn with a single color instead of graduated colors.
4. Open the tool layer and execute.  
The Polygons sublayer is again drawn with graduated colors.

As explained in the topic [Defining output symbology for geoprocessing tasks](#), when a model outputs a layer, the symbology found in the layer takes precedence over tool layer symbology. The reason for this rule is that some tools, like Make Service Area Layer, output layers containing custom symbology. To preserve this custom symbology, the symbology in the tool sublayer is ignored. If you want to change symbology of the drive-time polygons, the Calculate Drive Time Polygons model will need to output a feature class instead of a feature layer. This is easily accomplished by adding the [Copy Features](#) tool to the model, using the Polygons variable as input to Copy Features.

## Publishing

SanFranciscoBaseMap.mxd is published as a map service. DriveTimePolygonsService.mxd is published as a geoprocessing service with no result map service, as follows:

1. In the **Catalog** window, right-click SanFranciscoBaseMap.mxd and click **Publish to ArcGIS Server**.
2. Accept all defaults.
3. In the **Catalog** window, navigate to your server administrative connection under the **GIS Servers** node, right-click, then choose **Add New Service**. Name the service DriveTimePolygonsService and choose **Geoprocessing Service** as the type.
4. Click **Next**.
5. In the next panel, choose **Asynchronous** for **Execution Type**. For **The tools exposed by the Geoprocessing Service are stored in** option, choose **A map** and specify DriveTimePolygonsService.mxd for **Map Document**. Since you will test your service, check **Show Messages**.

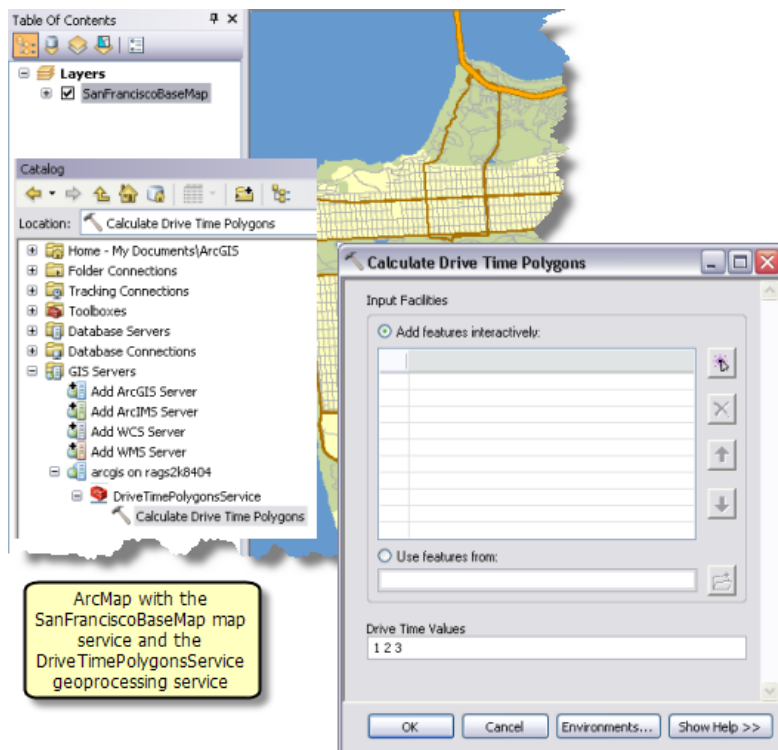


Publishing DriveTimePolygonsService

6. Click **Next**. From this point on, you can accept the default values provided by the wizard and create the service.

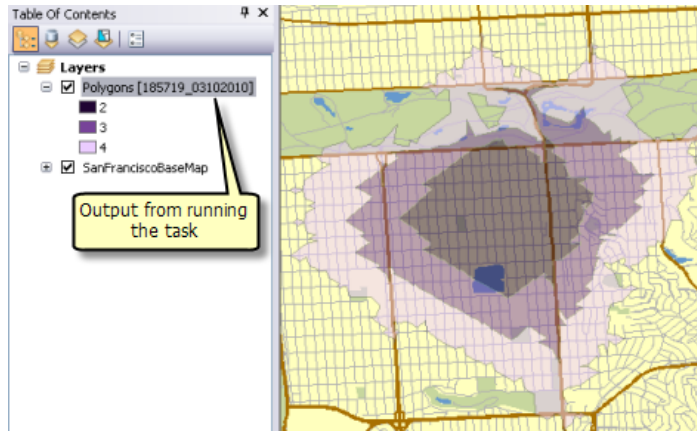
## Using

1. Start ArcMap with a blank document.
2. Create a user connection to ArcGIS Server from the **Catalog** window if one does not exist.
3. Add the SanFranciscoBaseMap map service to the ArcMap table of contents.
4. In the **Catalog** window, under your **GIS Servers** user connection node, expand the DriveTimePolygonsService toolbox and open the Calculate Drive Time Polygons tool by double-clicking it. The illustration below shows the result of these steps:



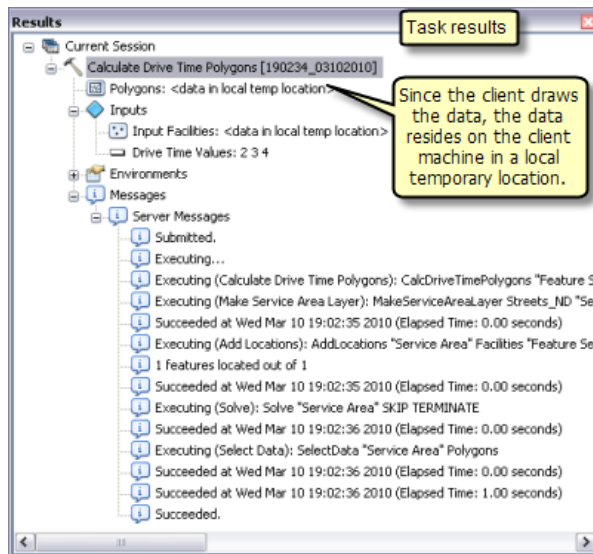
5. Add a point to create a facility location. Specify 2 3 4 for drive-time values and click **OK** to run the task.

After the task completes, the table of contents contains the Polygons output layer, as illustrated below.



Completed task

6. Take the opportunity to examine the result of the task in the **Results** window as illustrated below.



Task results

# GP service example: Shortest route on a street network

**Complexity:**  
Intermediate

**Data Requirement:**  
ArcGIS Tutorial Data Setup

**Data Path:**  
C:\ArcGIS\ArcTutor\GP Service Examples\ShortestRoute

**Goal:**  
Author, publish, and use a geoprocessing service that finds the shortest route on a street network and generates driving directions.

Folder	ShortestRoute
Purpose	Creates a shortest route between given points on a street network and generates driving directions in a text or HTML file.
Services	<ul style="list-style-type: none"> <li>SanFranciscoBaseMap (map service)</li> <li>ShortestRouteService (geoprocessing service)</li> </ul>
Geoprocessing tasks	<ul style="list-style-type: none"> <li>Calculate Shortest Route and Text Directions</li> <li>Calculate Shortest Route and HTML Directions</li> </ul>
Inputs	Two or more user-digitized points.
Outputs	<ul style="list-style-type: none"> <li>A shortest route between the user-specified points based on the travel time.</li> <li>An HTML or text file (depending on the task used) containing driving directions.</li> </ul>
Data	Uses a street network dataset for the San Francisco area provided in the ToolData folder.
Extensions	<a href="#">Network Analyst</a>
Of note	<ul style="list-style-type: none"> <li>Demonstrates how to reuse an existing Route network analysis layer each time the task is executed.</li> <li>Calculate Shortest Route and HTML Directions task—How an external Python library can be used from a script tool to convert driving directions from XML to HTML format.</li> </ul>

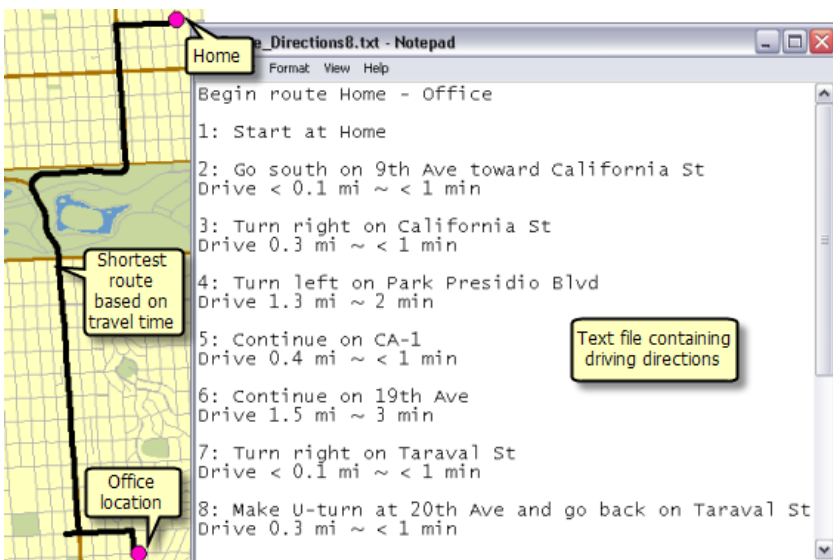
About this example

## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\ShortestRoute contains the completed models, script tool, and data.

## About this example

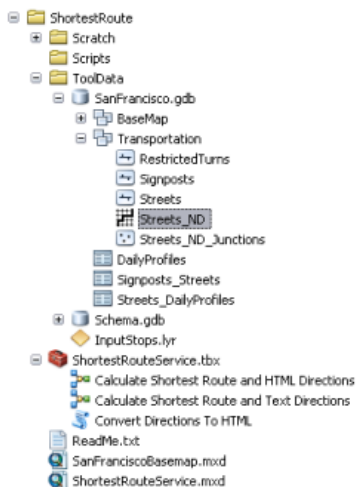
ShortestRouteService, created in this example, shows you how to publish geoprocessing tasks that calculate the shortest route on a street network between user-specified points and generate a file containing the driving directions. The Calculate Shortest Route and Text Directions task generates driving directions in a text file whereas the Calculate Shortest Route and HTML Directions task generates driving directions in an HTML file. Both the tasks also output the shortest route as a feature set.



Example output from Calculate Shortest Route and Text Directions task

## Data

The data for this example comes from C:\arcgis\ArcTutor\GP Service Examples\ShortestRoute.



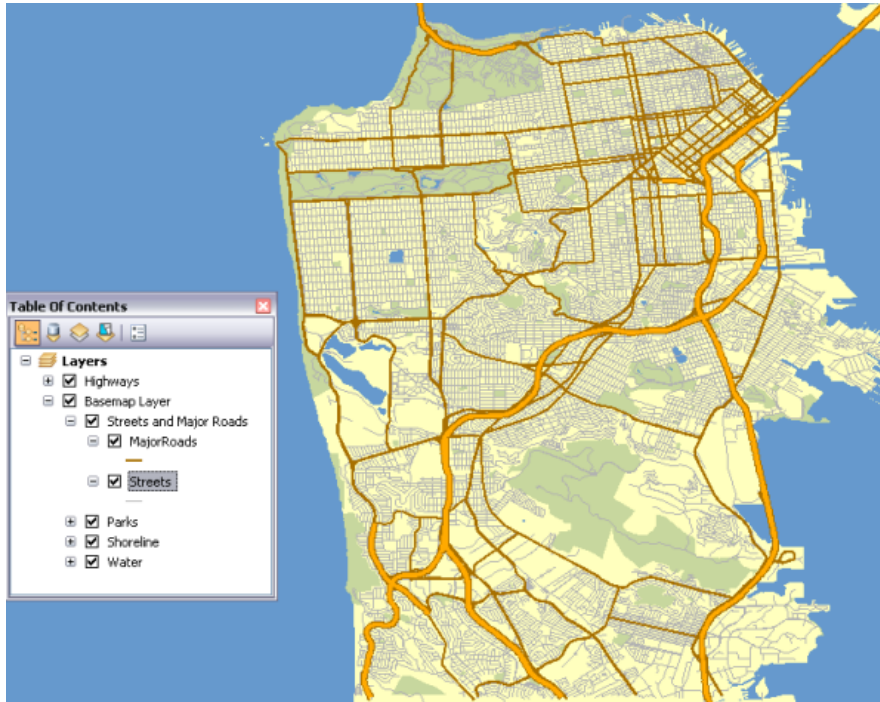
ShortestRoute folder contents

## Network dataset

The ToolData folder contains a file geodatabase, SanFrancisco.gdb. This geodatabase contains a network dataset, Streets\_ND, within the Transportation feature dataset. This [network dataset](#) models the street network for the San Francisco area. It provides a [network attribute](#), TravelTime, which indicates the time taken to travel each street segment in minutes.

## Basemap

The basemap layer within `SanFranciscoBaseMap.mxd` has a layer, `Streets`, as illustrated below. This layer shows the extent of the network dataset. This means that this task can be used to determine the shortest route only in this extent.



San Francisco basemap showing the network dataset extent

`SanFranciscoBaseMap.mxd` is published as a map service.

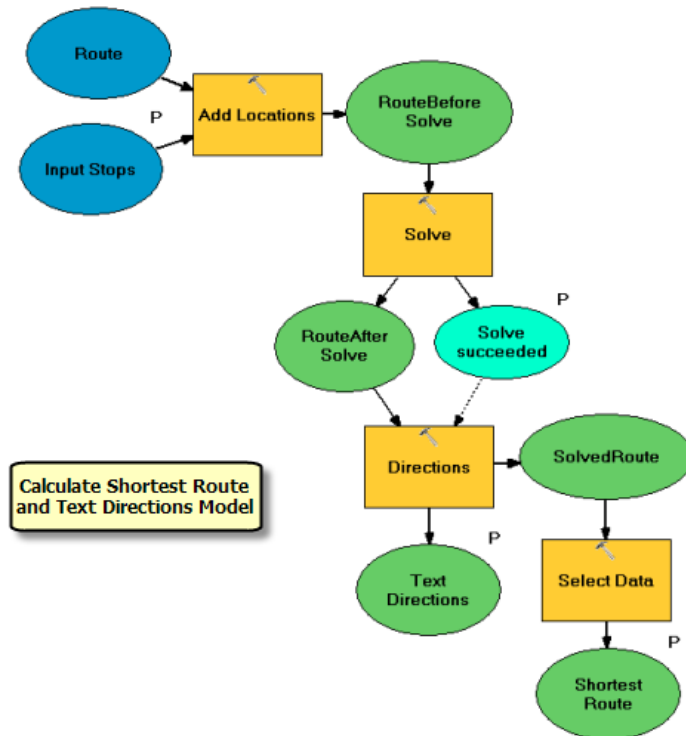
## Toolbox and map document

The toolbox for the geoprocessing service is `ShortestRouteService`, and the source map document for the service is `ShortestRouteService.mxd`. `ShortestRouteService` contains two models and a script tool, `Convert directions to HTML`. This script tool is used within the `Calculate Shortest Route` and `HTML Directions` model. `ShortestRouteService.mxd` contains two source data layers, `Streets_ND` (the network dataset) and `Route` (the network analysis layer).

## Model

### Model overview

The Calculate Shortest Route and Text Directions model is illustrated below. There is one input parameter, Input Stops, which is points. The shortest route visits the stops based on the digitized sequence. The model adds the user-digitized points as stops to an existing Route network analysis layer, performs a solve to determine the shortest route, generates driving directions, and writes them to a text file.



## Using an existing network analysis layer

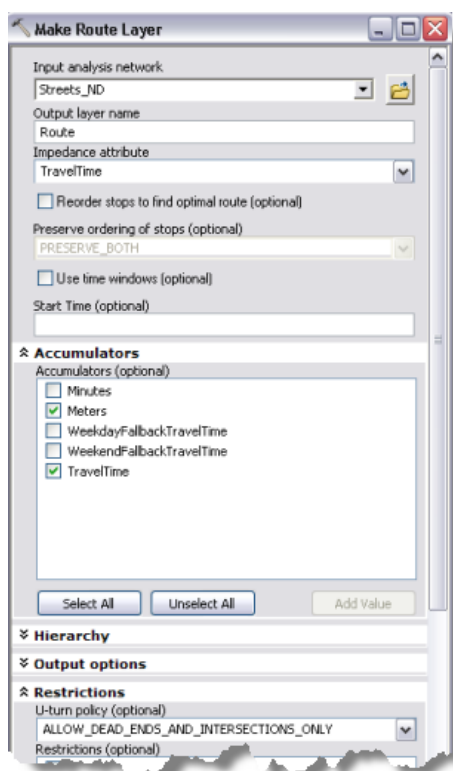
In this model, a network analysis layer is not created in the model as it is in the [DriveTimePolygonsService example](#). Instead, the existing route layer, Route, is used. (This layer was created using the [Make Route Layer](#) tool.) The existing layer can be used because none of the analysis properties for the route layer, such as impedance attribute, are exposed as model parameters. In the DriveTimePolygonsService example, the default break values (an analysis property for service area layers) was exposed as a model parameter, so the Make Service Area Layer tool was required as a model process.

Note that if any of the route analysis properties, such as impedance attribute, are to be exposed as model parameters, the Make Route Layer tool has to be used as a first process in the model.

The Route layer used as an input variable in the model was created by first adding the Streets\_ND network dataset to the ShortestRouteService.mxd, then using the Make Route Layer tool. For this example, the following parameters were used for the Make Route Layer tool. Default values were used for the parameters not mentioned in this table.

Parameter	Value
Input analysis network	Streets_ND
Output layer name	Route
Impedance attribute	TravelTime
Accumulators	TravelTime;Meters
U-turn policy	ALLOW_DEAD_ENDS_AND_INTERSECTIONS_ONLY

Parameter values used with Make Route Layer tool



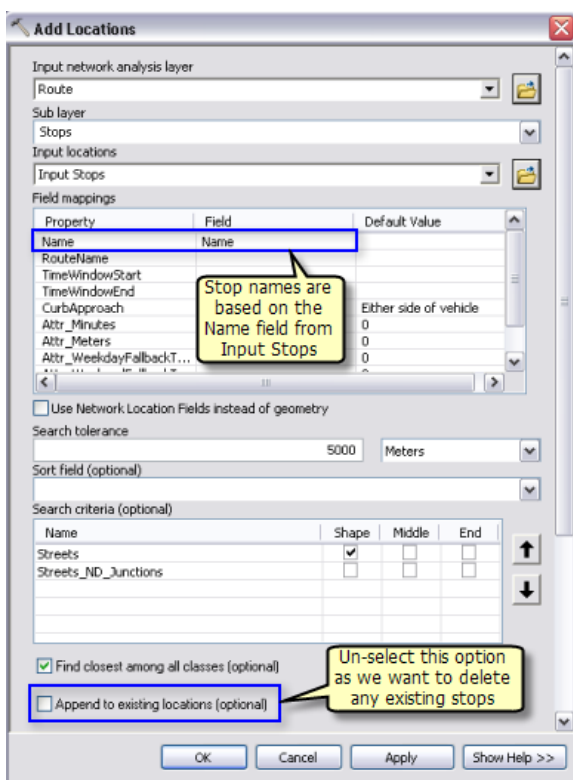
Creating a route layer

## Model processes

The [Add Locations](#) tool adds user-digitized points as stops to the route layer. The Input Locations parameter for the tool is specified through a model parameter, Input Stops. This parameter is a feature set data type. The schema and symbology for the feature set are derived from the InputStops.lyr file found within the ToolData folder.

The schema for the Input Stops feature set contains a text field, `Name`. This field can be used to provide stop names while generating driving directions by mapping the `Name` property to the name field in **Field mappings**, as illustrated below. If the value for the name field is not specified by the user, default values of Location 1, Location 2, and so on, are used for stop names.

Since the same route layer is used for each model run, any existing stops need to be removed before adding the new stops. To delete existing stops, the **Append to existing locations** option, illustrated below, is unchecked.



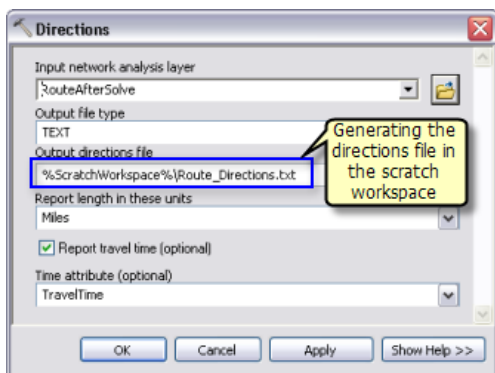
Adding stops

The [Solve](#) tool calculates the shortest route based on the TravelTime network attribute and other options as specified in the Route layer. The calculated route is written to the Routes sublayer in the output route layer.

Network analysis layers (such as Route) are not [supported output parameter data types](#) for ArcGIS Server clients. So the [Select Data](#) tool is used to get the Routes sublayer from the Route layer.

The [Directions](#) tool is used to generate the driving directions and output them to a text file. The output text file containing driving directions is created in the jobs directory on the server using the `%scratchworkspace%` [inline variable](#). The SolveSucceeded variable derived from the Solve tool is used

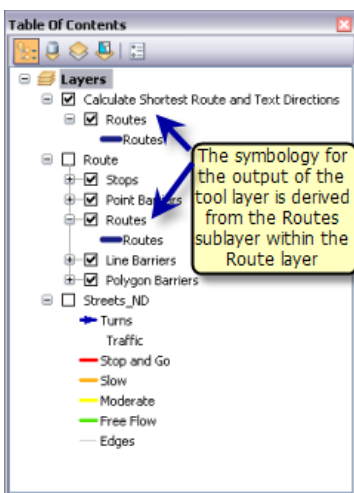
as a **precondition** for the Directions tool. This means that the directions file is generated only if the Solve tool can find a route between input points.



Generating driving directions

## Tool layer

The Calculate Shortest Route and Text Directions tool layer is created by dragging the Calculate Shortest Route and Text Directions model into the ArcMap table of contents. Since the model updates the existing Route layer, the tool layer output derives its symbology from the Routes sublayer within the Route layer in the table of contents.



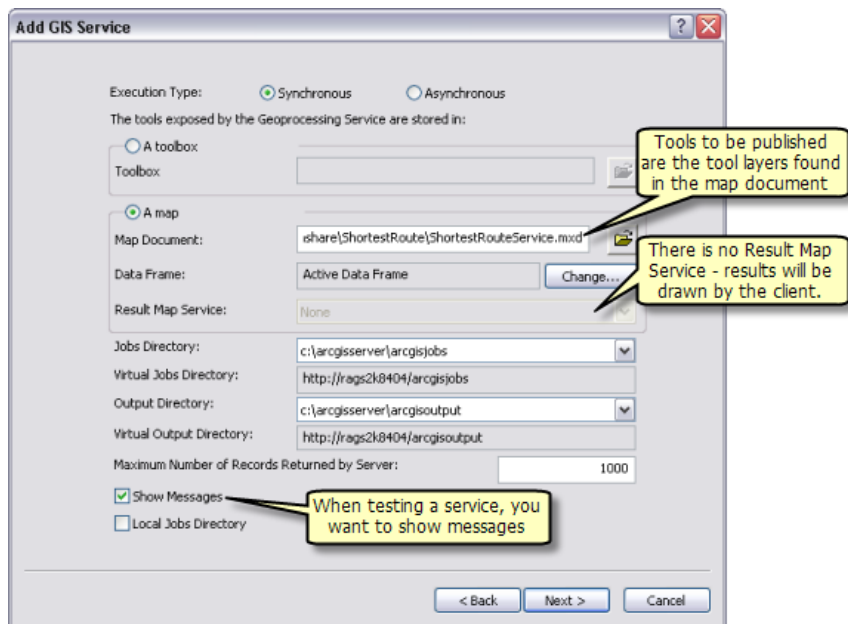
Symbology for the tool layer

## Publishing

SanFranciscoBaseMap.mxd is published as a map service. ShortestRouteService.mxd is published as a geoprocessing service with no result map service, as follows:

1. In the **Catalog** window, right-click SanFranciscoBaseMap.mxd and click **Publish to ArcGIS Server**.
2. Accept all defaults.

3. In the **Catalog** window, navigate to your server administrative connection under the **GIS Servers** node, right-click, then choose **Add New Service**. Name the service `ShortestRouteService` and choose **Geoprocessing Service** as the type.
4. Click **Next**.
5. In the next panel, choose **Synchronous** for **Execution Type**. For **The tools exposed by the Geoprocessing Service are stored in** option, choose **A map** and specify `ShortestRouteService.mxd` for **Map Document**. Since you will test your service, check **Show Messages**.

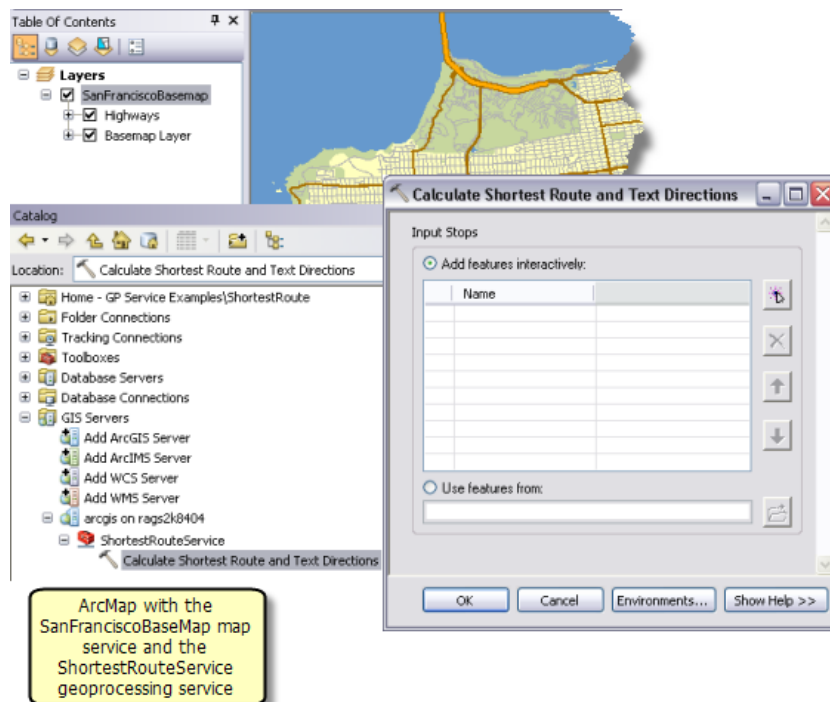


Publishing ShortestRouteService

6. Click **Next**. From this point on, you can accept the default values provided by the wizard and create the service.

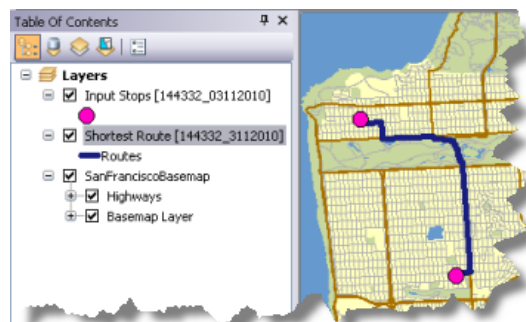
## Using

1. Start ArcMap with a blank document.
2. Create a user connection to ArcGIS Server from the **Catalog** window if one does not exist.
3. Add the `SanFranciscoBaseMap` map service to the ArcMap table of contents.
4. In the **Catalog** window, under your **GIS Servers** user connection node, expand the `ShortestRouteService` toolbox and open the `Calculate Shortest Route and Text Directions` tool. The illustration below shows the result of these steps:



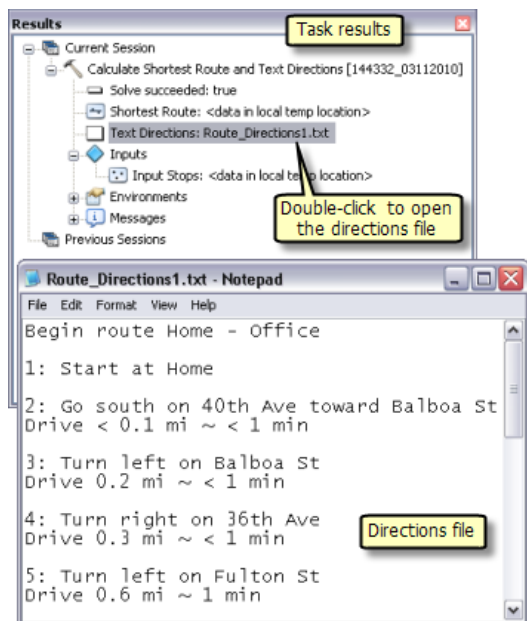
5. Add two or more points to create stops. Optionally, specify names for each stop and click **OK** to run the task.

After the task completes, the table of contents contains the Shortest Route output layer, as illustrated below. The input stops are not output from the task but are added to the table of contents from the **Inputs** node in the **Results** window.



Task result

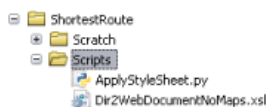
6. The text file containing the directions is copied from the jobs directory on the server to the scratch workspace for the current ArcMap session. The text file containing the directions can be viewed by double-clicking it in the **Results** window.



Viewing the directions file

## Generating HTML directions

The Directions tool used in the Calculate Shortest Route and Text Directions model can generate directions in text or XML format. The directions in XML format can be converted to a nicely formatted HTML file by applying a style sheet using a Python script. The script, `ApplyStyleSheet.py`, found in the Scripts folder within the ShortestRoute folder, can be used to convert directions from XML to HTML format. The scripts folder also contains the style sheet file, `Dir2WebDocumentNoMaps.xml`, that specifies the format of the HTML file.



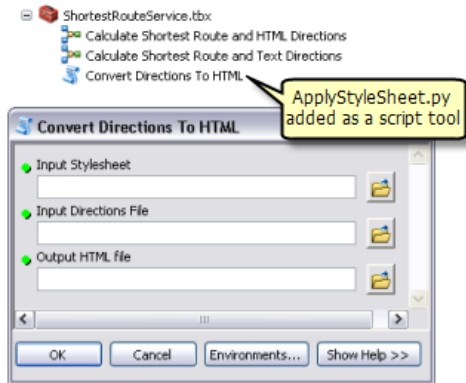
Scripts folder contents

## Installing the external Python library

The `ApplyStyleSheet.py` script uses an external Python library, `libxml2`. This library is not provided with ArcGIS and is not supported by ESRI. It is external, third-party software. You will need to download and install this library on your ArcGIS Server SOC machines. Navigate to <http://xmlsoft.org/sources/win32/python/>, download the latest `libxml2` setup for Python 2.6 (for example, `libxml2-python-2.7.4.win32-py2.6.exe`—earlier versions may not work). After downloading, double-click the executable file to install the library.

## Creating the script tool

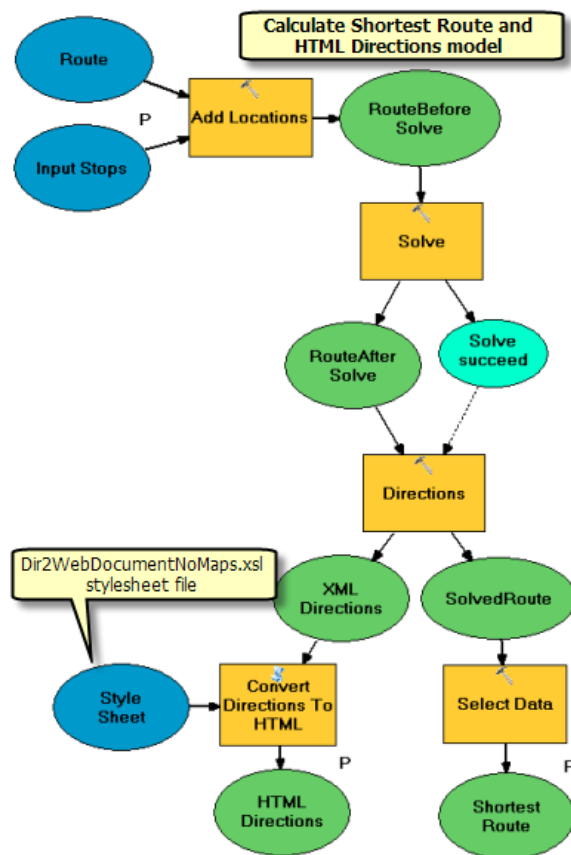
To use the `ApplyStyleSheet.py` script in the model, it is added to `ShortestRouteService` toolbox as a script tool called `Convert Directions to HTML`. This script tool takes the style sheet file and the XML file as input and generates an output HTML file.



Adding the script tool

## Adding the script tool to the model

The Generate Shortest Route and Text Directions model is renamed and saved as Generate Shortest Route and HTML Directions so as to add the Convert Directions to HTML script tool. In this model, the output file type parameter for the Directions tool is changed to XML. This XML file and the `Dir2WebDocumentNoMaps.xml` file are used as input variables for the script tool. The output HTML file is written to the jobs directory on the server using the `%scratchworkspace%` inline variable.

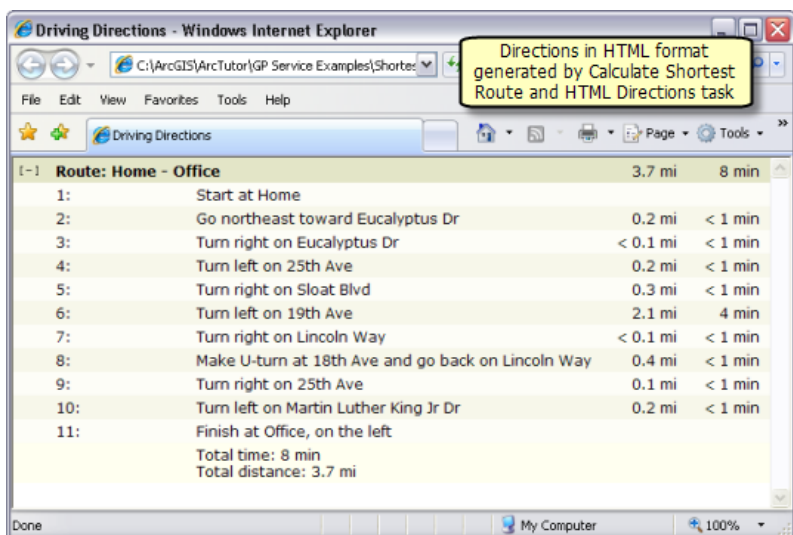


## Publishing and using

The Calculate Shortest Route and HTML Directions model can be published as a new task within the ShortestRouteService geoprocessing service created earlier, as described below:

1. Create the Calculate Shortest Route and HTML Directions tool layer in ShortestRouteService.mxd by dragging the Calculate Shortest Route and HTML Directions model from the **Catalog** window to the ArcMap table of contents.
2. Save ShortestRouteService.mxd.
3. In the **Catalog** window, stop and start the ShortestRouteService geoprocessing service.

The ShortestRouteService service should now have a second task called Calculate Shortest Route and HTML Directions. This task is similar to the Calculate Shortest Route and Text Directions task except that it generates the directions in HTML format.



Directions in HTML format

# GP service example: Finding nearby features over a street network

**Complexity:**  
Advanced

**Data Requirement:**  
ArcGIS Tutorial Data Setup

**Data Path:**  
C:\ArcGIS\ArcTutor\GP Service  
Examples\ClosestFacilities

**Goal:**  
Author, publish, and use a geoprocessing service that finds features closest to a given location based on shortest route on a street network.

Folder	ClosestFacilities
Purpose	Finds a given number of closest libraries from a starting location based on the travel time along a street network, calculates the shortest route to each of the closest library, and generates driving directions in a text file.
Services	<ul style="list-style-type: none"> <li>SanFranciscoBaseMap (map service)</li> <li>ClosestFacilitiesService (geoprocessing service)</li> </ul>
Geoprocessing task	Find Nearby Libraries
Inputs	One or more user-digitized points and the number of closest libraries to find.
Outputs	<ol style="list-style-type: none"> <li>The closest libraries.</li> <li>The shortest routes between the user-specified points and each of the closest libraries based on the travel time.</li> <li>A text file containing driving directions for each route.</li> </ol>
Data	Uses a street network dataset and a feature class containing all the library locations in the San Francisco area provided in the ToolData folder.
Extensions	<a href="#">Network Analyst</a>
Of note	Demonstrates how to use a solved network analysis layer for further analysis.

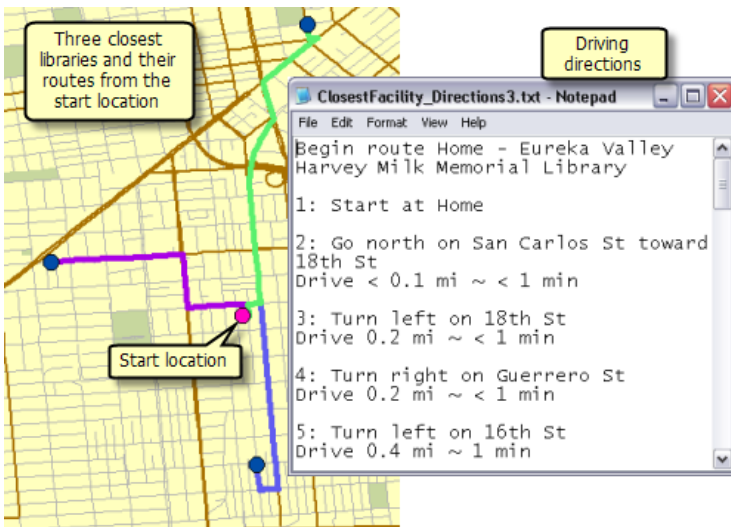
About this example

## Corresponding folder

C:\arcgis\ArcTutor\GP Service Examples\ClosestFacilities contains the completed models and data.

## About this example

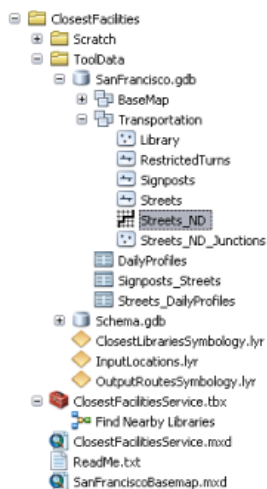
The `ClosestFacilitiesService` service created in this example shows you how to publish a geoprocessing task that performs a closest facility analysis on a street network. The `Find Nearby Libraries` task determines the shortest routes to a user-specified number of closest libraries from the given points based on travel time along the street network. The task outputs the routes and driving directions to the closest libraries.



Example output from Find Nearby Libraries task

## Data

The data for this example comes from C:\arcgis\ArcTutor\GP Service Examples\ClosestFacilities.



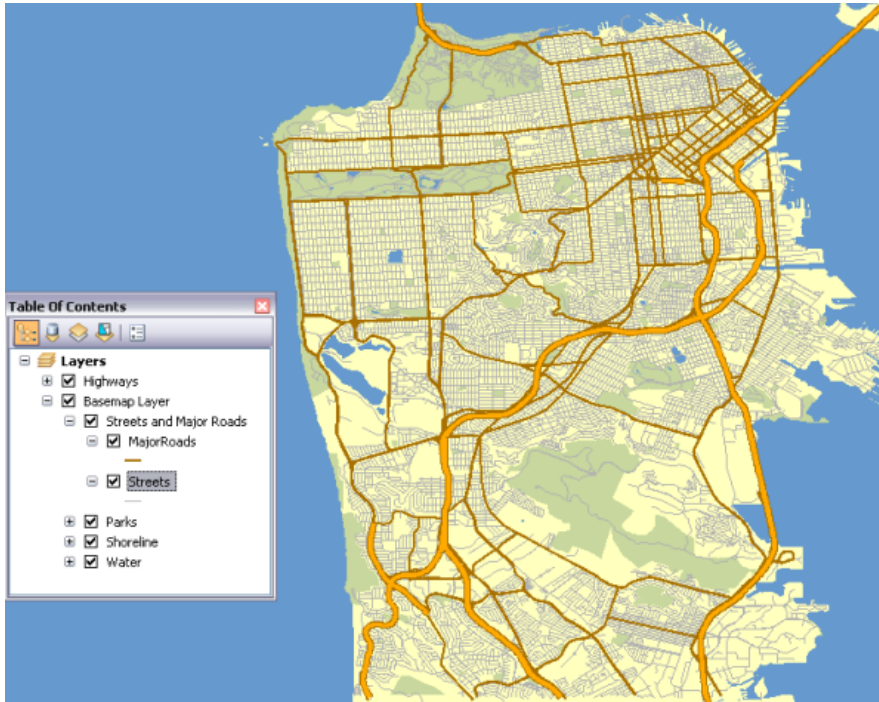
ClosestFacilities folder contents

## Network dataset

The ToolData folder contains a file geodatabase, SanFrancisco.gdb. This geodatabase contains a network dataset, Streets\_ND, within the Transportation feature dataset. This [network dataset](#) models the street network for the San Francisco area. It provides a [network attribute](#), TravelTime, which indicates the time taken to travel each street segment.

## Basemap

The basemap layer within `SanFranciscoBaseMap.mxd` has a layer, `Streets`, as illustrated below. This layer shows the extent of the network dataset. This means that this task can be used to determine the nearby libraries only in this extent.



San Francisco basemap showing the network dataset extent

`SanFranciscoBaseMap` is published as a map service.

## Toolbox and map document

The toolbox for the geoprocessing service is `ClosestFacilitiesService`, and the source map document for the service is `ClosestFacilitiesService.mxd`. `ClosestFacilitiesService.mxd` contains the following four source data layers and the Find Nearby Libraries tool layer:

- `Streets_ND`—The network dataset.
- `Library`—Feature layer containing the locations of all the libraries in the San Francisco area
- `ClosestLibrariesSymbology`—Feature layer that defines the symbology for the output libraries
- `OutputRoutesSymbology`—Feature layer that defines the symbology for the output routes

## Model

## Model overview

The Find Nearby Libraries model is illustrated below. There are two input variables:

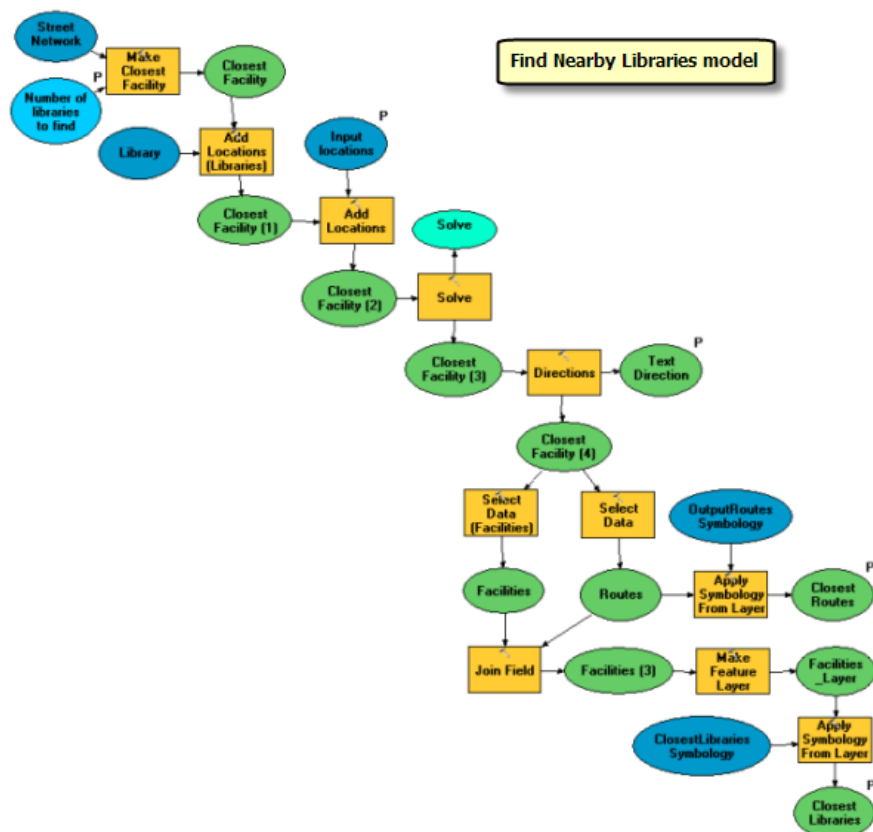
- Input Locations, which are the user-specified points from which the closest libraries are found
- Number of libraries to find

The model creates a closest facility network analysis layer; adds the library locations from the library feature layer as facilities; adds the user-specified locations as incidents; and performs a solve to determine the routes to the closest libraries, generate driving directions, and find only those libraries that are included in the routes from all the libraries that are loaded as facilities.

Element	Type	Description
Street Network	Network dataset layer	The network dataset layer.
Number of libraries to find	Long, input parameter	The number of libraries to find for each input location.
<a href="#">Make Closest Facility Layer</a>	Tool	Creates a closest facility network analysis layer. This layer contains both data and properties that determine how the closest facilities will be calculated, along with the results of the calculation.
Closest Facility	Network Analyst layer	Closest facility layer.
Library	Feature Layer	The point feature class containing all the library locations. The positions of these points on the street network are already calculated as described in the section <a href="#">Precalculating network locations for libraries</a> .
<a href="#">Add Locations (Libraries)</a>	Tool	Adds the library locations as facilities to the closest facility layer.
Closest Facility (1)	Network Analyst layer	Closest facility layer with facilities.
Input Locations	Feature set (points), input parameter	Point features from which the closest libraries are determined.
Add Locations	Tool	Adds the input locations as incidents to the closest facility layer.
Closest Facility (2)	Network Analyst layer	Closest facility layer with facilities and incidents.
<a href="#">Solve</a>	Tool	Calculates the closest facilities and determines the shortest route to each facility.
Closest Facility (3)	Network Analyst layer	Closest facility layer containing all the facilities and the shortest route to the closest facilities.
SolveSucceeded	Boolean	The derived output from the Solve tool that indicates if the solve was successful.
<a href="#">Directions</a>	Tool	Generates the driving directions for the routes to the closest facilities.
Text Directions	File, output parameter	The text file containing driving directions.
Closest Facility (4)	Network Analyst layer	Closest facility layer containing all the facilities and the shortest route to the closest facilities.

Select Data	Tool	Selects the Routes sublayer from the closest facility layer.
Routes	Feature layer	The routes layer from Closest Facility (3) Network Analyst layer.
OutputRoutesSymbology	Layer	The symbology layer used to apply symbology to the Routes feature layer.
Apply Symbology From Layer	Tool	Applies symbology to the Routes layer from the OutputRoutesSymbology layer.
Closest Routes	Feature layer, output parameter	The Routes layer with appropriate symbology.
Select Data (Facilities)	Tool	Selects the facilities sublayer from the closest facility layer.
Facilities	Feature layer	The facilities layer from the Closest Facility (3) Network Analyst layer.
Join Field	Tool	Joins the FacilityID, FacilityRank, Total_TravelTime, and Total_Meters fields from the Routes layer to the Facilities layer.
Facilities (3)	Table view	The derived facilities layer containing the joined fields.
Make Feature Layer	Tool	Selects only the facilities for which the FacilityID value is not null. Only the fields required in the output facilities are set to be visible.
Facilities_Layer	Feature layer	The facilities feature layer containing only the facilities that are included in the routes.
ClosestLibrariesSymbology	Layer	The symbology layer used to apply symbology to Facilities_Layer.
Apply Symbology From Layer (1)	Tool	Applies symbology to Facilities_layer from the ClosestLibrariesSymbology layer.
Closest Libraries	Feature layer, output parameter	The Facilities_Layer with appropriate symbology.

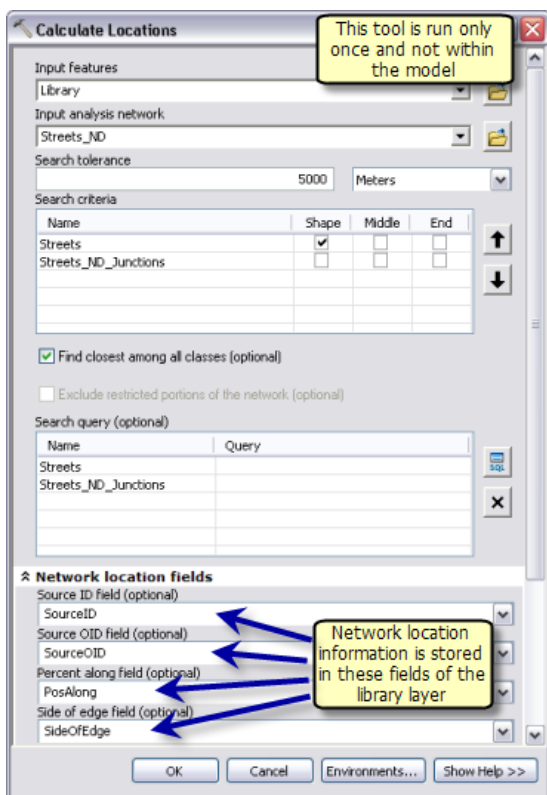
Model elements



## Precalculating network locations for libraries

The library locations used in the closest facility analysis are not transient—their locations on the network remain constant. Hence it is more efficient to calculate their **network locations** only once instead of calculating them every time they are added as facilities.

The **Calculate Locations** tool can be used to determine the network locations for the libraries and store the information in the fields SourceID, SourceOID, PosAlong, and SideOfEdge. This information can then be used by the Add Locations tool to load the libraries as facilities in the new closest facility layer. This is considerably faster than using Add Locations to first determine the network locations for libraries, then load them as facilities. For the Library layer, the network locations were determined based on the Streets\_ND network dataset layer using the Calculate Locations tool.

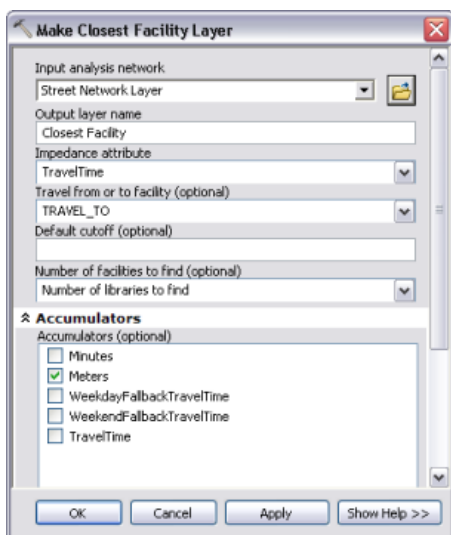


Calculating network locations for libraries

Note that if in another scenario the facilities **are** transient, their network locations will have to be determined every time they are added as facilities. So precalculating their network locations using the Calculate Locations tool will not provide any performance benefit.

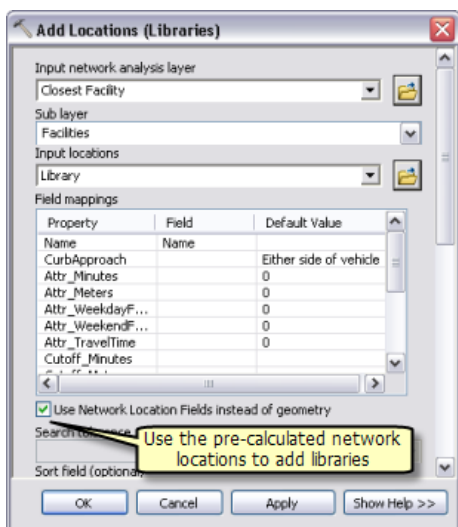
## Model processes

The [Make Closest Facility Layer](#) tool creates a new Network Analyst layer, Closest Facility, that stores the analysis properties, references the Streets\_ND network dataset layer used for the analysis, and stores the input facilities and incidents and the output routes. The network dataset has a network cost attribute called TravelTime, which specifies the travel time required to traverse each street segment in minutes. This attribute is used as an impedance attribute. The Number of libraries to find variable specifies the number of facilities to find.



Make Closest Facility Layer tool parameters

The [Add Locations \(Libraries\)](#) tool adds the library locations as facilities to the closest facility layer. Since the network locations for the libraries were already calculated using the Calculate Locations tool, the **Use Network Location fields instead of geometry** option was checked.



Using network location fields to add facilities

The Add Locations tool adds the user-digitized points as incidents to the closest facility layer. The Input Locations parameter is a feature set data type so that the model can interactively accept the user-digitized

points as incidents. The schema and symbology for the feature set are derived from the `InputLocations.lyr` file found within the ToolData folder.

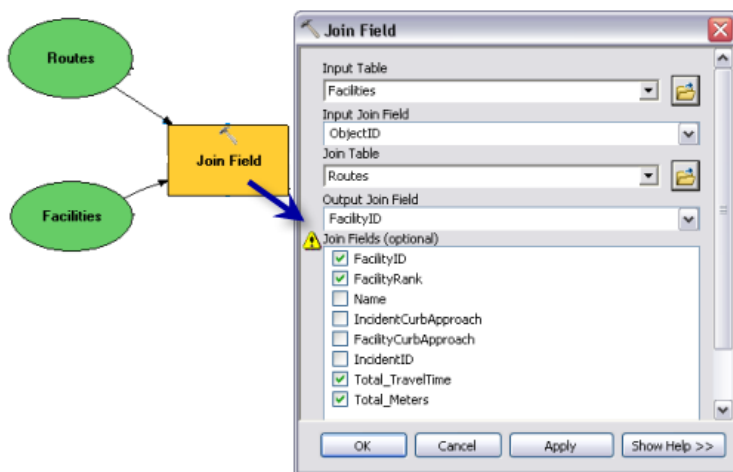
The [Solve](#) tool searches the given number of closest facilities from each incident and calculates a shortest route to each facility based on the TravelTime network attribute. The calculated routes are written to the Routes sublayer in the output closest facility layer.

The Network Analyst layer is not a [supported output parameter data type](#) for ArcGIS Server clients. So the [Select Data](#) tool is used to get the Routes sublayer from the Closest Facility Network Analyst layer.

The Routes sublayer uses the symbology of the Network Analyst layer. To apply a different symbology such that each route has a unique color, the [Apply Symbology From Layer](#) tool is used to apply symbology to the Routes sublayer from the OutputRoutesSymbology layer.

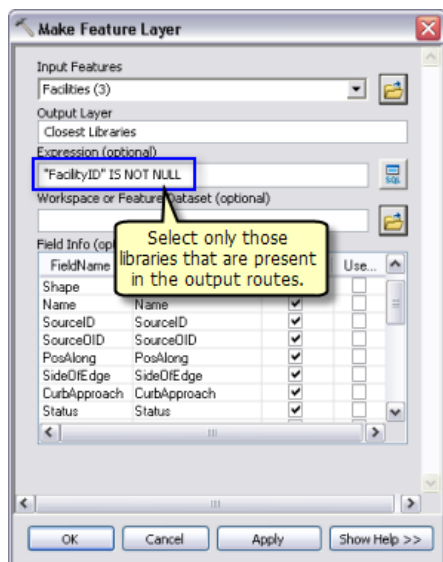
The [Directions](#) tool is used to generate the driving directions and output them to a text file. The output text file containing driving directions is created in the jobs directory on the server using the `%scratchworkspace%` [inline variable](#).

The Routes sublayer in the Closest Facility layer contains a FacilityID field that indicates the object ID of the facility visited by the route. This information can be used to select only the facilities that are visited by routes from all the facilities. The [Join Field](#) tool joins the Routes sublayer to the facilities layer using the FacilityID field. The tool joins the FacilityID, FacilityRank, Total\_TravelTime, and Total\_Meters fields to the facilities sublayer based on FacilityID in routes and ObjectID in facilities.



Join Field tool parameters

The output of the Join Field tool contains the FacilityID field in the facilities sublayer. This field has a value of null for all the facilities that are not visited by the routes. Using the [Make Feature Layer](#) tool, only those facilities for which the value of the FacilityID field is not null are selected and output to a new layer.



Make Feature Layer tool parameters

The symbology for the facilities layer is set from the OutputLibrariesSymbology layer using the Apply Symbology From Layer tool.

## Tool layer

The Find Nearby Libraries tool layer is created by dragging the Find Nearby Libraries model into the ArcMap table of contents.

Since the model outputs are in-memory feature layers, the Closest Libraries and Routes sublayer within the tool layer will have a broken data source when you first open ClosestFacilitiesService.mxd. The map document will publish as is. However, you should rerun the tool layer and verify that the model works before publishing the service.

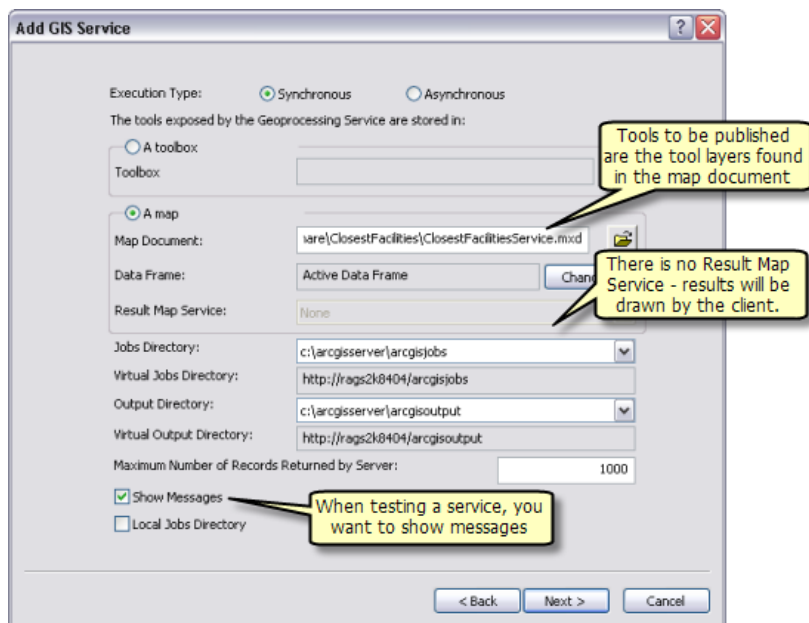
Note that if you change the symbology for any of the output layers in the tool layer, the new symbology will not be used. This is because the outputs of this model are feature layers and their symbologies have already been defined using the Apply Symbology To Layer tool in the model.

## Publishing

SanFranciscoBaseMap.mxd is published as a map service. ClosestFacilitiesService.mxd is published as a geoprocessing service with no result map service, as follows:

1. In the **Catalog** window, right-click SanFranciscoBaseMap.mxd and click **Publish to ArcGIS Server**.
2. Accept all defaults.
3. In the **Catalog** window, navigate to your server administrative connection under the **GIS Servers** node, right-click, then choose **Add New Service**. Name the service ClosestFacilitiesService and choose **Geoprocessing Service** as the type.
4. Click **Next**.

- In the next panel, choose **Synchronous** for **Execution type**. For **The tools exposed by the Geoprocessing service are stored in** option, choose **A map** and specify `ClosestFacilitiesService.mxd` for **Map Document**. Since you will test your service, check **Show Messages**.

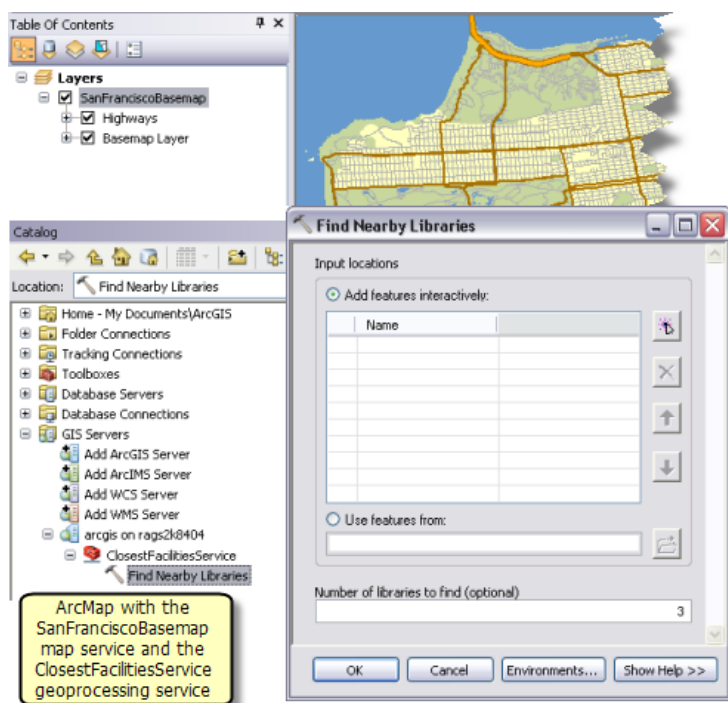


Publishing the ClosestFacilitiesService

- Click **Next**. From this point on, you can accept the default values provided by the wizard and create the service.

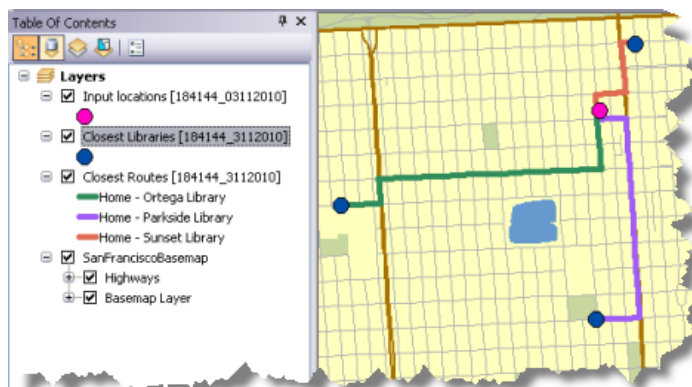
## Using

- Start ArcMap with a blank document.
- Create a user connection to ArcGIS Server from the **Catalog** window if one does not exist.
- Add the `SanFranciscoBaseMap` map service to the ArcMap table of contents.
- In the **Catalog** window, under your **GIS Servers** user connection node, expand the `ClosestFacilitiesService` toolbox and open the Find Nearby Libraries tool. The illustration below shows the result of these steps:



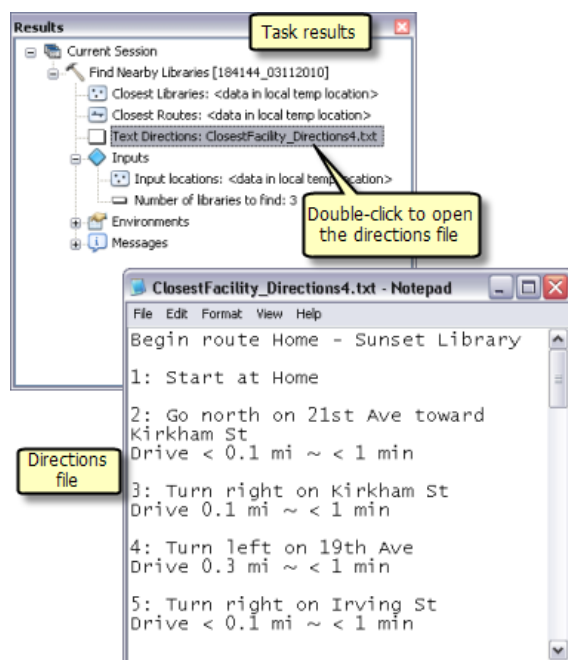
5. Add a point to create an input location. Specify 3 for number of libraries to find and click **OK** to run the task.

After the task completes, the table of contents contains the Closest Libraries and Routes output layer, as illustrated below. The input locations are not output from the task but are added to the table of contents from the **Inputs** node in the **Results** window.



Completed task

6. The text file containing the directions is copied from the jobs directory on the server to the scratch workspace for the current ArcMap session. This file can be viewed by double-clicking it in the **Results** window.



Viewing the directions file