

Introduction to MineSight® Scripts

MineSight® is a powerful 3D data modeling and visualization package that offers great flexibility for geologic modeling, mine planning, design, and evaluation. Scripts add to this flexibility by providing access to a wide variety of additional functionality.

Scripts are a set of instructions written and interpreted in the Python™ scripting language and many scripts are included in the MineSight installation.

Scripts can be run from:

- MineSight 3-D (MS3D).
- MineSight Compass™ (MSCompass).
- MineSight Interactive Planner (MSIP).
- Outside the MineSight environment, from the command line or from Windows Explorer.

In this article, we will discuss:

- What types of scripts are available with the MineSight installation.

- MPython and running scripts.
- Setting up an MSIP batch script to run outside of MineSight.

Scripts in MineSight and what their prefixes indicate.

Scripts come in uncompiled form (suffixed .py) or compiled form (suffixed .pyo or .pyc). The uncompiled versions can be viewed in any text editor and contain information indicating the purpose and usage at the top of the script.

All standard MineSight scripts will have one of five different prefixes. Each of these prefixes indicates how the script is intended to be executed. See the table below for a description of each prefix.

Prefix	Description	Location in winexe
No Prefix	Script is executed either as a standalone from Windows Explorer or a Command Prompt, or from MS3D: File Scripts (e.g., shellRpt.pyo)	winexe\scripts
CP	Script is executed as a MSCompass Procedure (e.g., cp-ModelCalcTool.pyc)	winexe\metlib
EM	Script is executed within MS3D: File Scripts (e.g., em-polystats.py)	winexe\scripts
IP	Script is executed from MSIP Cut Design dialog (e.g., ip-accum.py)	winexe\scripts\reserve
Batch IP	Script is executed in batch mode outside MineSight, using data collected from a MineSight Planning Database (MSPD) that has been populated with reserves data from MSIP. (e.g., batch-ip-summary.py)	winexe\scripts\reserve

MPython: Python Version Conflicts Now a Thing of the Past.

In MineSight 4.50, changes were made to how MineSight interacts with Python 2.2.3. In previous versions of MineSight, Python 2.2.3 was installed in the C:\Python22 directory, and the environmental variables **PYTHONPATH** and **PYTHONHOME** pointed to the location of critical Python files.

But because Python is an open-source and very popular scripting language, there was the potential for other software packages, which also use Python, to install or reconfigure Python and thus cause conflicts.

Once this issue became clear, Mintec's Software Development group set out to find an adequate resolution for this problem. Our solution is the Mintec Python Distribution or MPython. Rather than the traditional Python 2.2.3 installation, MPython is a Python 2.2.3 installation that is stored within the MPython folder in the MineSight installation directory (winexe).

This upgrade will be completely transparent to you while running inside the MineSight environment (e.g., from MS3D, MSIP, or MSCompass). However, outside the MineSight environment, MineSight scripts need some indication to use MPython rather than the standard Python installation.

From the Command Line

There are two ways to run scripts from the command line: either running in a special MPython command prompt or using the MPython command file.

The MPython command prompt operates just like the standard Windows command prompt, with the exception that it configures **PYTHONHOME** and **PYTHONPATH** to find MPython. This special version of the Windows command prompt can be run from the Windows **Start** menu (**Start | Programs | MineSight | MPython | MPython Command Line**).

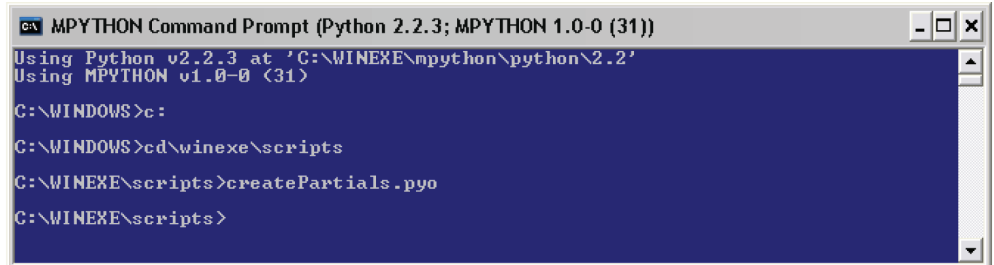


Figure 1. The new MPython Command Prompt. Notice the messages at the top indicating that MPython is being used.

Scripts can also be run from the standard Windows command prompt using the MPython command file (**winexe\mpython.cmd**). The syntax is **mpython scriptName.py** as shown in Figure 2 below:

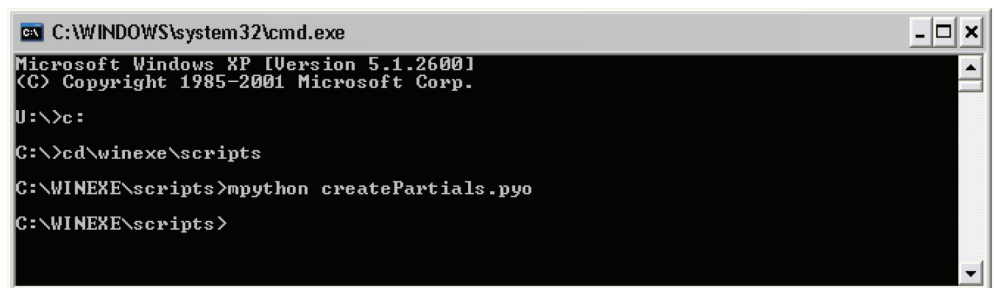


Figure 2. Scripts can be run using MPython from the standard Windows command prompt.

Running Scripts from Explorer

You can also configure Python file extensions to provide the option to “Run with MPYTHON” when right clicked in Windows Explorer.

From Windows Explorer, go to **Tools | Folder Options**, and navigate to the **File Types** tab (Figure 3). In the **Registered File Types** window, scroll down to the ***.py**, ***.pyc**, and ***.pyo** file extensions. These are the file extensions associated with scripts in MineSight. We will configure each of them to run with **mpython.cmd**. The ***.py** extension should already have this option following the MineSight installation, however the ***.pyo** and ***.pyc** formats will not.

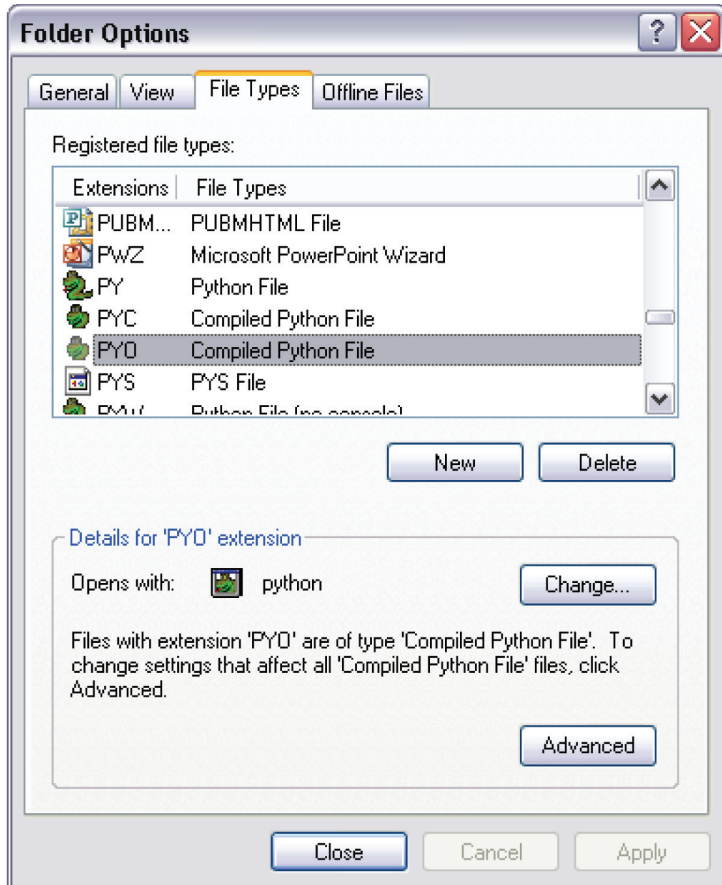


Figure 3. The File Types panel of the Folder Options in Windows Explorer.

Highlight the appropriate file extension (`.pyo` in our example), and click the **Advanced** button.

This will bring up the **Edit File Type** dialog (Figure 4). This dialog allows you to add new options for using or operating on files with a particular extension. In our case we want the option to run files with a `.pyo` extension using MPython (`mpython.cmd`).

Click the **New** button to add the option. The name for the option (entered under the **Action** field) will be “Run with MPYTHON”. Then indicate the application; click the **Browse** button and browse to `mpython.cmd` in the MineSight® directory (`winexe`). Select this, and click **OK**.

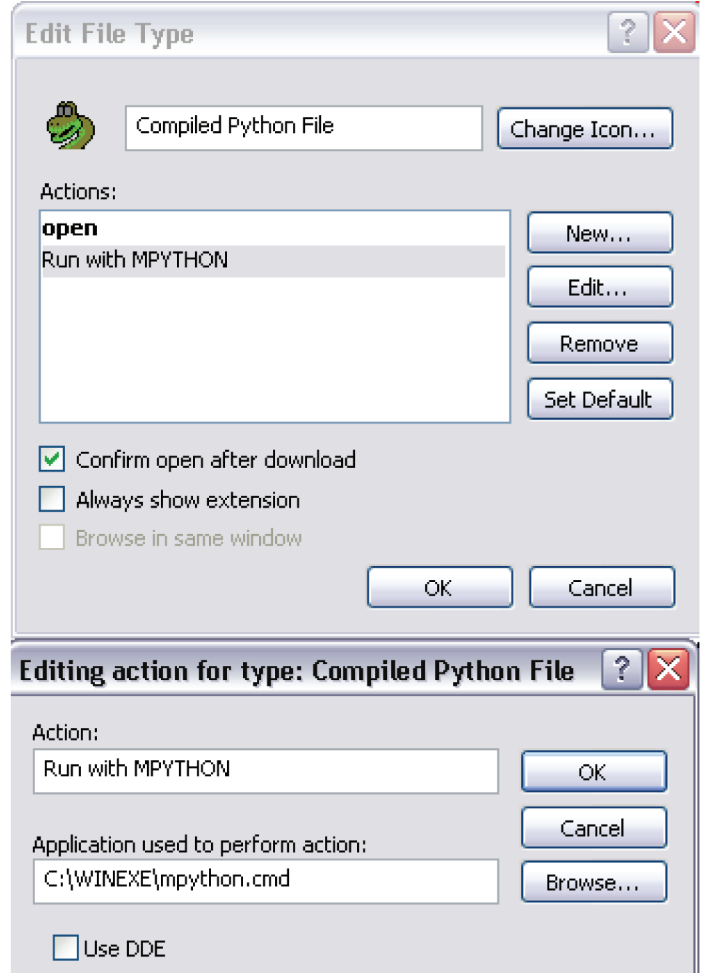


Figure 4. The Edit File Type and Editing Action windows.

Once this is done, when you click right on a file with the edited extension (in this case, a `*.pyo` file), you should see the “Run with MPYTHON” option on the right click menu:



Figure 5. The “Run with MPYTHON” option on the right click menu.

If you are not using any other version of Python, you can configure **all** Python script extensions to open with MPython by default when run from Windows Explorer. This can also be done from the **File Types** panel in the **Folder Options** shown in Figure 6. Select the appropriate file extension (`*.py`, `*.pyo`, or `*.pyc`) and click the **Change** button. This will bring up a dialog that prompts you to choose a program to open the file type. If MPython is an option, select it and click **OK**. Otherwise, click **Browse** and navigate to `mpython.cmd` in the `winexe` directory. Select it, and click **OK** to set `mpython.cmd` as the default for opening each file type.

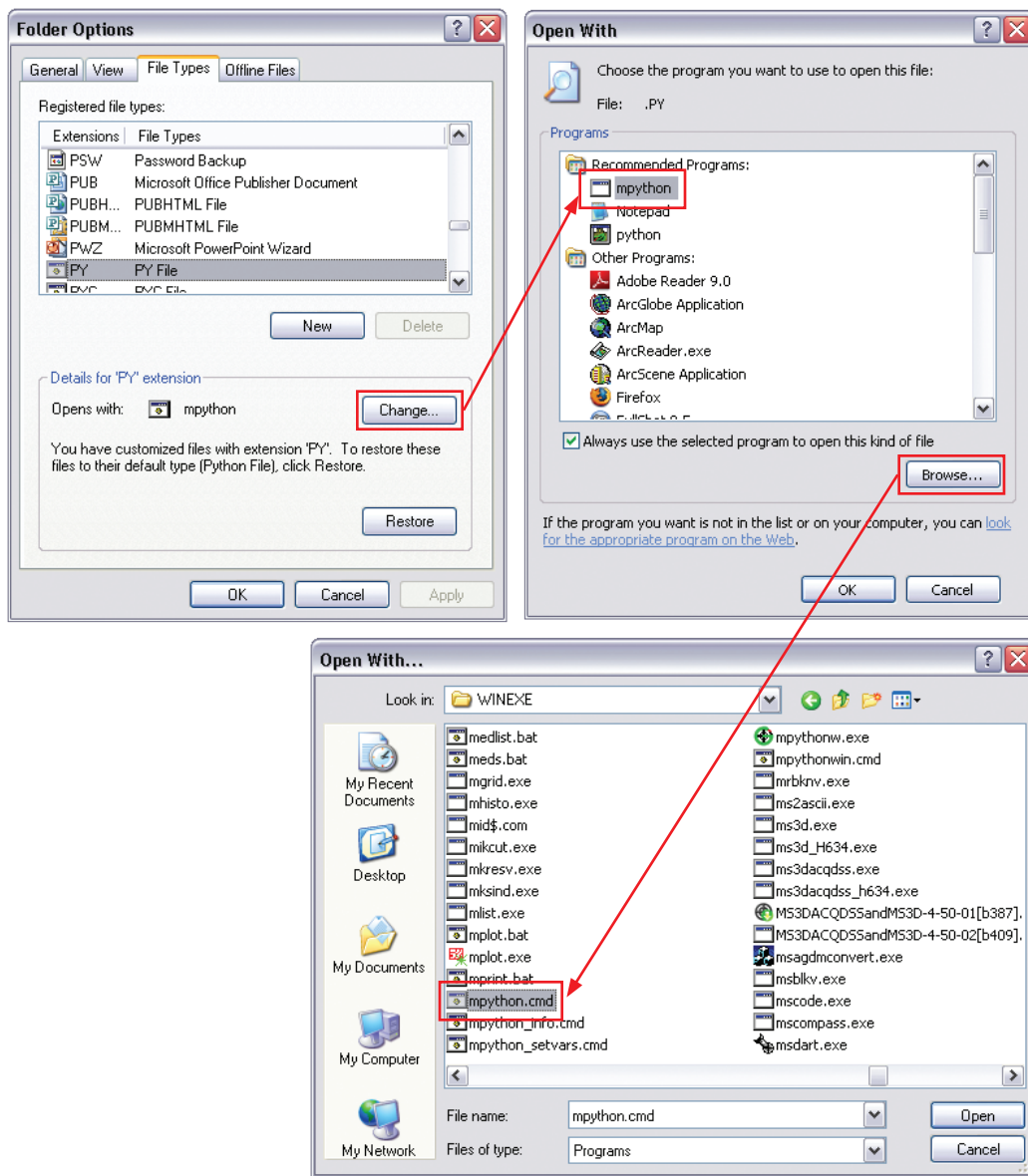


Figure 6. Set the default program for executing all Python scripts. If MPython does not appear in the Open With dialog, add it using the Browse option.

Now, you will be able to simply double click scripts with these extensions from Windows Explorer and run them normally, as it was in earlier versions of MineSight.

Setting up Batch files to run IP scripts outside MineSight.

Several IP scripts can be run outside MineSight from a batch file. In order to run these scripts, you must have previously set up an IP Set (also known as an IP Plan) in MSPD and used MSIP to calculate reserves.

There are currently six standard IP scripts that can be run in this manner, including: **batch-ip-accum.py**, **batch-ip-allflatreport.py**, **batch-ip-flatreport.py**, **batch-ip-period.py**, **batch-ip-report.py**, and **batch-ip-summary.py**. All of these scripts perform the exact same function as their non-batch brethren. The only difference is that these scripts can be run from a batch file outside MineSight. Below, we go through a step-by-step example on how to set up **batch-ip-accum.py**. The script name and batch file name can be altered so that this setup can be used with any batch ip script.

First, create a new text file using Notepad. Save the file as **IP_Accum.bat** (or some other ***.bat** file). The text of the batch file should use the following format:

```
Batch_IP_format.bat - Notepad
File Edit Format View Help
Drive_Name
cd\filepath_of_batch_ip_script
mpython batch_ip_script_name.py "Data_Source_Name" "IP_set_name"
```

Figure 7. Basic format for a batch IP script.

As an example, see Figure 8. Here we use:

- **C:** for the *Drive_Name*,
- **test\WINEXE\scripts\reserve** for the *filepath_of_batch_ip_script*,
- **batch-ip-accum.py** (in c:\test\WINEXE\scripts\reserve) for the *batch_ip_script_name.py*,
- **IPTrainingJan23** for the *Data_Source_Name* (DSN),
- **IPTraining1** for the *IP_set_name*.

```
IP_Accum.bat - Notepad
File Edit Format View Help
C:
cd\test\WINEXE\scripts\reserve
mpython batch-ip-accum.py "IPTrainingJan23" "IPTraining1"
```

Figure 8. Example batch file for running batch-ip-accum.py.

In some cases, the DSN will require a username and password. In those cases, the username and password can be supplied in two ways: written into the batch file or coded into the Python script itself.

To write the DSN username and password into the batch file, the format of the batch file should be matched to the screenshot below, where the username and password are added to the file in the DSN/username/password format:

```
Batch_IP_format.bat - Notepad
File Edit Format View Help
Drive_Name
cd\filepath_of_batch_ip_script
mpython batch_ip_script_name.py "Data_Source_Name/username/password" "IP_set_name"
```

Figure 9. Basic format for a batch file including DSN username and password information.

The username and password can also be specified (along with the DSN and IP Set name) inside the batch script itself, provided you edit the uncompiled version of the script (the one with the **.py** extension). Open the script in any text editor and you will find the following statements near the top of the script:

```
# == START CONFIGURATION OPTIONS ==
# DSN name or DSN/user/pswd string.
DSN_STRING = "MyODBCname/username/password"

# IP planname (IP Set).
PLAN_NAME = "myplan"
```

Figure 10. Basic format for specifying the DSN (with login information) as well as the IP Set in the batch ip script.

With the DSN and IP Set specified within the Python script, you no longer need to include this information in the batch file. The following format can be used for the batch file when the DSN and IP Set are specified in the Python script:

```
Batch_IP_format.bat - Notepad
File Edit Format View Help
Drive_Name
cd\filepath_of_batch_ip_script
mpython batch_ip_script_name.py
```

Figure 11. If the DSN and IP Set are specified in the script, they can be omitted from the batch file.

Now, the batch file can be run from Windows Explorer in any location that has access to the MSPD database through an ODBC connection. Obviously, since cut selection cannot be done outside IP, **ip-accum** will only provide cumulative reserves. All other scripts should operate as normal. Also, in all cases, IP batch scripts will not work for IP Sets containing multiple areas and material sets, as the area and material set cannot be specified in the batch file.

More information on specific MineSight Scripts

Script specific information for standard MineSight scripts can be found in the help documentation as part of the file scripts.chm. This file is stored in the WINEXE\Scripts directory.

For each standard script, this document contains a basic summary of the script's capabilities, dependencies, a revision history, and information regarding user configuration options.