ArcGIS Pro Scripting README

[Client Name]-GISScripts

This is a repository of Python ETL scripts to manage GIS data for TBD

Client:

Remote Servers:

- [DEV ENV] `Server Name`
- [UAT ENV] `Server Name`
- [PROD ENV] `Server Name`

Remote Path:

- [DEV ENV] `None`
- [UAT ENV] `None`
- [PROD ENV] `E:\Scripts\GISScripts`

Task Scheduler Paths:

- [DEV ENV] `None`
- [UAT ENV] `None`
- [PROD ENV] `Task Scheduler (Local)\Task Scheduler Library\TBD`

Table of Contents

Folder and File Structure

- **app** - Folder containing the Python scripts and modules
    - **connections** - Folder to store .SDE and other connection files for each environment.
    - **templates** - Folder containing various templates that are used by the application modules.
        - **email** - Folder containing .TXT file templates used by `util_emailer.py`
    - **utils** - Folder containing reusable Python modules for the scripts.
        - **init.py** - Module that indicates what modules are part of a Python package.
        - **util_database.py** - Module for database connections.
        - **util_emailer.py** - Module for sending emails.
        - **util_gis.py** - Module for GIS functions.
        - **util_logger.py** - Logging module.
    - **init.py** - Module that indicates what modules are part of a Python package.
    - **gis_database_weekly_maintenance.py** - Reconciles versions in the GIS database and performs various database maintenance tasks.
    - **settings.template** - Template file to create a `settings.py` file for each environment.
    - **settings.py** - Must be created from the `settings.template` for each environment. Specifically, excluded from the Git repository.
- **logs** - Folder containing the log files for each of the scripts. This folder is created automatically, if it doesn't exist.
- **task_scheduler_batch_files** - Folder contains .bat files that are used by the Windows Task Scheduler to run Python scripts in the `app` folder.
    - **GISDatabase-WeeklyTasks.bat**
    - **MapServices-WeeklyTasks.bat**
- **temp** - Folder containing the temp files, specifically file geodatabases, for the script processing. This folder is created automatically, if it doesn't exist.
- **unittests** - Folder containing unit testing for the Python script modules. This is not complete.
    - **fixtures**
    - **src**
    - **tests**
- **.gitignore** - File containing entries that will be ignored by Git repositories.
- **README.md** - This document
- **requirements.txt** - Additional Python packages that must be added to the ArcGIS Python installation.

Requirements

- ArcGIS Pro 2.9.3
- JetBrains PyCharm
- Git

Installation, Configuration, and Setup

This section covers the installation, configuration, and environmental modifications to runs these Python scripts. It assumes the user has a basic understanding of Python IDEs, Git functionality, and Windows OS.

This documentation is specific to using JetBrains PyCharm as the Python IDE. All the following steps can be also replicated in VS Code but we find that VS Code's Python Interpreter configuration tends to be more confusing that PyCharm's with the ArcGIS Platform.

Authorizing ArcGIS Pro

When using the ArcPy libraries, the service account MUST be authorized. The following steps will configure ArcGIS Pro operate.

1. Right-click on **ArcGIS Pro** from the **Start** menu and select **More** > **Run as different user**.
2. Login in with the service account credentials
3. In the sign-in windows, click the configure license settings
4. Sign into Portal or ArcGIS Online.

Copying repository from GitHub

This step is only necessary for the initial cloning of the GitHub repository. Once installed, standard Git pushes and pulls should be used. Please refer the PyCharm GitHub Documentation for additional information.

1. Open the PyCharm IDE by running PyCharm under the service account to allow it to traverse the network.
2. In the **Welcome to PyCharm** dialog, click **Get from VCS**
3. In the **Get from Version Control** dialog, enter the following values:
    - **Version Control:** `Git`
    - **URL:** `TBD`
    - **Directory:** `E:\Scripts\`

You may be prompted for credentials to access the GitHub repository, provide the appropriate credentials.

[IMPORTANT] Once PyCharm clones the repository it will open the IDE and display a **Creating Virtual Environment** dialog. It is CRITICAL to select **Cancel**. Using ArcGIS ArcPy in a virtual environment is a horrendously complex process due to DLL dependencies. We will be using ArcGIS Pro to clone its Python environment.

Cloning ArcGIS Pro Python Environments

TO BE COMPLETED https://pro.arcgis.com/en/pro-app/latest/arcpy/get-started/clone-an-environment.htm

Configuring PyCharm IDE with ArcGIS Python Installation

This sections covers how to configure PyCharm to use the ArcGIS Python installation as the interpreter.

1. In PyCharm, select **File** > **Settings**
2. In the **Settings** dialog, expand **Project: Scripts** and select **Python Interpreter**
3. Click the Gear icon on the right side of the **Python Interpreter**
4. In the **Add Python Interpreter** dialog, select **Conda Environment**, click the radio button next to **Existing environment** and click the `...` icon.
5. For the **Interpreter** path, navigate to `E:\Scripts\envs\arcgispro-py3-293\python.exe`. For the **Conda executable**, navigate to `C:\Program Files\ArcGIS\Pro\bin\Python\Scripts\conda.exe` and click **OK**
6. [Optional] Add a checkmark next to **Make available to all projects**
7. Click **OK** to close the **Add Python Interpreter** dialog.
8. Click **OK** to close the **Settings** dialog.

[NOTE #1] PyCharm will detect any missing packages from the `requirements.txt` file and ask if you want to install them. Be sure to install them.

[NOTE #2] PyCharm will indicate that many of the packages need to be updated. Resist the temptation to upgrade these packages until fully testing in the DEV environment. DDS experience is that this typically goes badly due to package changes that are not supported ArcPy. ArcGIS Server uses Python v2.7 which was deprecated in 2021. Eventually, it would be best to migrate to the Python v3 release but there is significant added complexity that is not currently in scope or budget. Documentation on ArcGIS Server and ArcPy.

Creating `settings.py` for each environment

This sections covers using the `settings.template` to create a `settings.py` file for each environment.

1. Copy or rename `settings.template` to `settings.py` in the top level of the `app` folder.
2. Configure the environmental constants appropriately.

[NOTE] The SDE_CONNECTION_FILE_NAME is configured in the next step

Copying in environment-specific .SDE connection files

This section covers copying in environment-specific ArcGIS .SDE connections files into the project.

1. If the `connections` folder does not exist in the `app` folder, create one.
2. Copy in the environment-specific .SDE connection file into the `connections` folder.
3. In the `settings.py`, enter the filename of the .SDE connection file into the constant.

[NOTE] The recommended naming convention for the .SDE connection file is `Server_Database_User.sde`.

Task Scheduler

This section describes the various tasks configured in Windows Task Scheduler on the Remote Servers.

To open **Task Scheduler**, click **Start** and type in `Task Scheduler`. Expand the **Task Scheduler Library** in the tree menu. DDS recommends creating a subfolder in the **Task Scheduler Library**, named "GISScripts" or something to store all tasks associated with the scripts.

Each task should be configured as follows:

[NOTE] These settings need to be verified against existing tasks.

GISDatabase-WeeklyTasks

- **General**
  - Description: Runs `GISDatabase-WeeklyTasks.bat` weekly database maintenance on GIS database.
  - Security Options
    - User: Service Account name
    - `Run whether user is logged on or not`
- **Trigger**
  - Trigger: Weekly
  - Details: Sundays at 00:00 recurring every Sunday.
- **Action**
  - Action: Start a Program
  - Program/Script: `E:\Scripts\[Missing Path]\GISDatabase-WeeklyTasks.bat`
  - Add Arguments: None
  - Start In: `E:\Scripts\`
- **Conditions**

  Use default settings
- **Settings**
  - "Allow task to be run on demand" should be checked
  - "Stop the task if it runs longer than" should be set to "1 day"
  - "If the task is already running, then the following rule applies" should be set to "Stop the existing instance."

Unit Testing Procedures

Unit testing has only been scaffolded out and is not yet implemented.

Leave a comment

Markdown is supported