esri ジャパン

ArcPy 入門
～ArcGIS Pro の作業を自動化！～

2024 年 5 月 22 日 (水)
ESRI ジャパン株式会社
プラットフォームプロダクトグループ
廣川 智也

本セッションについて

- 概要
  - Python に関する基本的な解説や ArcPy の基本的な機能に加え、ArcPy を用いた ArcGIS Pro の業務の自動化の例を紹介

- 対象者 (おすすめのユーザー)
  - ArcGIS Pro ユーザー
  - ArcPy で自動化を実施したいユーザー
  - ArcGIS Pro の機能を Python で扱いたいユーザー

- 到達目標
  - ArcPy を使って ArcGIS Pro の自動化ができる
  - Python で ArcGIS Pro の機能を使用できる

目次

1. **Python/ArcPy とは？**
   - Python とは
   - ArcPy とは
   - ArcPy サイトパッケージの内容
     - 関数
     - クラス
     - モジュール
   - Python/ArcPy を使うとできること
   - ArcPy を使用するメリット
   - ArcPy を実行する環境

目次

1. Python/ArcPy とは？

Python とは

シンプルで扱いやすく、多くの用途に適したプログラミング言語

- プログラミング言語
  - タスクを自動化する
  - シンプルなコーディングで可読性の高いコードが記述できる
  - 世界中で親しまれているスクリプト言語

- 豊富なライブラリが提供されている
  - 数学・科学技術計算ライブラリが使用できる
  - ビッグデータ解析や機械学習にも使用されている

- オープンソース

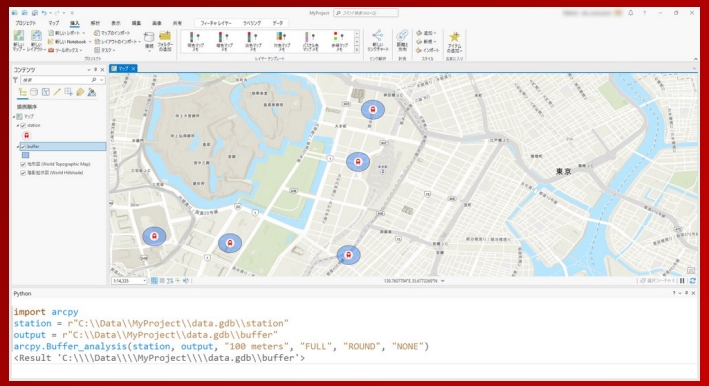
ArcPy とは

ArcGIS 製品群 のさまざまなタスクを自動化

- Python スクリプトから地理的データの解析、変換、管理などを実行するための便利な関数およびクラスの集まり (パッケージ)
- ジオプロセシング ツールを拡張するための標準ライブラリ
  - ArcGIS Pro
  - ArcGIS Enterprise

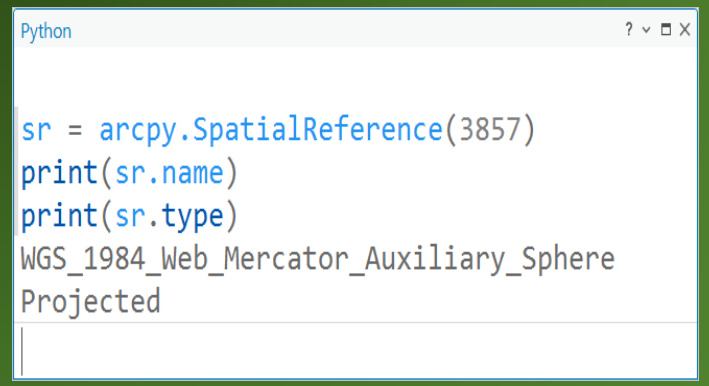
ArcPy サイトパッケージの内容

ArcPy 関数



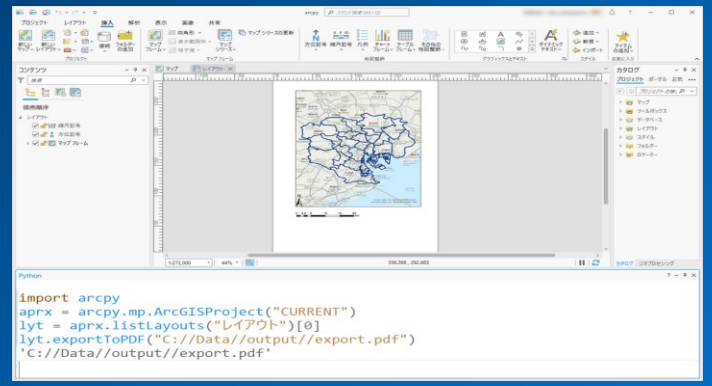

```python
# ArcPy のインポート
import arcpy

# 入出力パラメーターの設定
station = r"C:¥Data¥MyProject¥data.gdb¥station"
output = r"C:¥Data¥MyProject¥data.gdb¥buffer"

#上で設定したパラメーターを使ってバッファーツールを実行
arcpy.Buffer_analysis(roads, output, "100 meters“, "FULL“, "ROUND“, "NONE“)
```

ArcPy クラス




```python
#SpatialReference クラスを Web メルカトル (3857) で設定
sr = arcpy.SpatialReference(3857) # arcpy.<クラス名> で使用

# 設定した SpatialReference のプロパティ (name と type) を出力
print(sr.name)
print(sr.type)
```

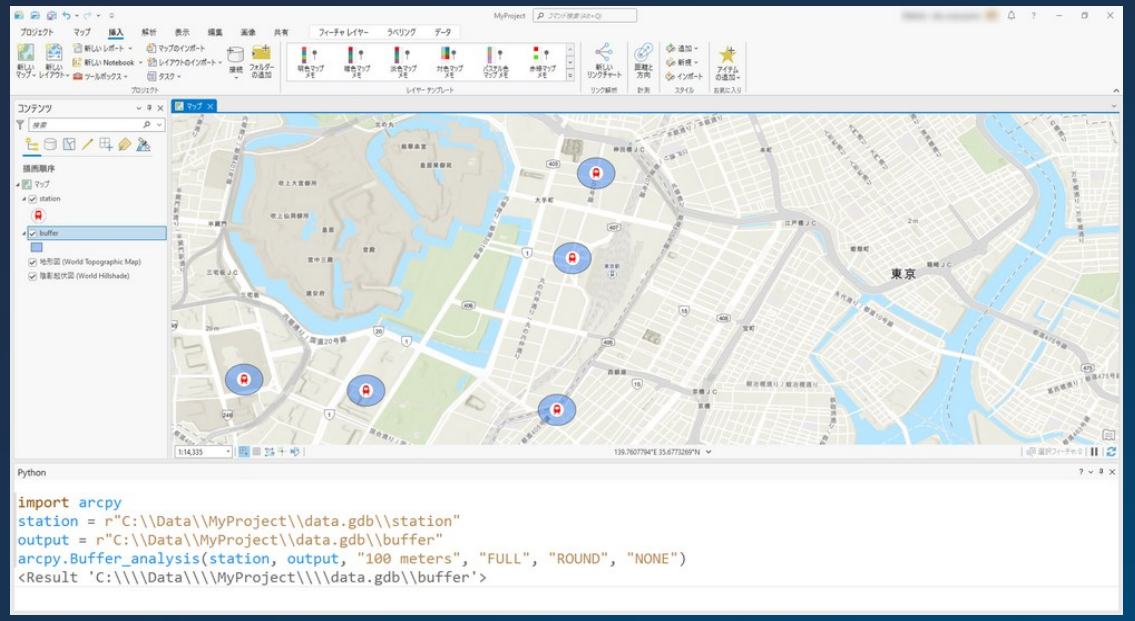
ArcPy モジュール




```python
# ArcPy のインポート
import arcpy

# 現在開いているプロジェクトファイルを参照
aprx = arcpy.mp.ArcGISProject("CURRENT”)

# プロジェクトファイルの中のレイアウトを検索し取得
lyt = aprx.listLayouts("レイアウト")[0]

# PDFにエクスポート
lyt.exportToPDF(r"C:¥Data¥output¥export.pdf")
```

ArcPy サイトパッケージの内容

関数

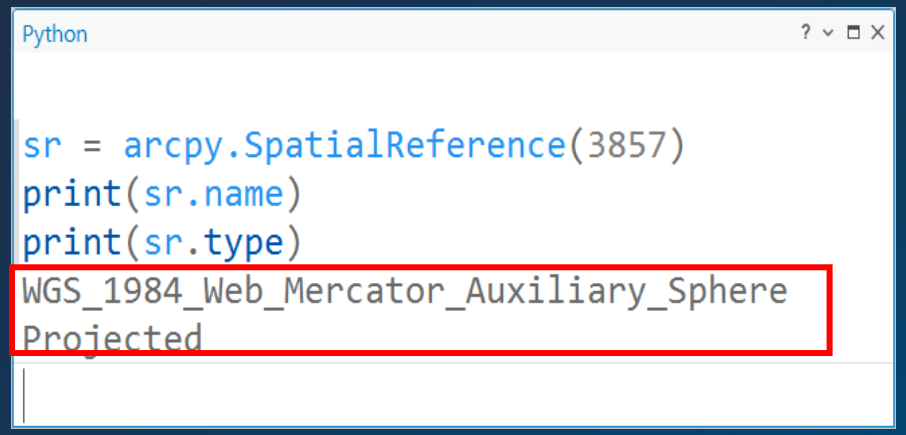



```
# ArcPy のインポート
import arcpy

# 入出力パラメーターの設定
station =
r"C:¥Data¥MyProject¥data.gdb¥station"
output =
r"C:¥Data¥MyProject¥data.gdb¥buffer"

#上で設定したパラメーターを使ってバッファー
ツールを実行
arcpy.Buffer_analysis(roads, output, "100
meters“, "FULL“, "ROUND“, "NONE“)
```

全てのジオプロセシング ツールを関数として提供

ArcPy サイトパッケージの内容

クラス



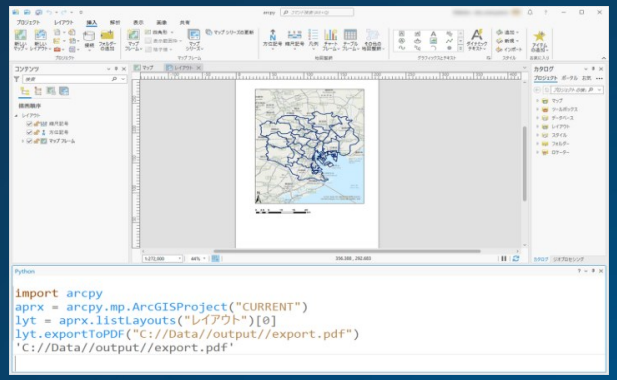

```
#SpatialReference クラスを Web メルカトル (3857) で設定
sr = arcpy.SpatialReference(3857) # arcpy.<クラス名> で使用

# 設定した SpatialReference のプロパティ (name と type) を出力
print(sr.name)
print(sr.type)
```

まとまったデータや一律に設定した値を処理

ArcPy サイトパッケージの内容

モジュール

モジュール名	機能
データ アクセス モジュール (arcpy.da)	データの操作 (編集操作、カーソル サポートの向上 (高速化など)、テーブルおよびフィーチャクラス 等)
マッピング モジュール (arcpy.mp)	マップ作成の自動化 (エクスポート、印刷の自動化 等)
Spatial Analyst モジュール (arcpy.sa)	ラスターおよびベクター データの解析 (クラス、関数、ジオプロセシング機能、演算子)
Network Analyst モジュール (arcpy.nax)	ネットワーク解析 (ルート、最寄り施設の検出、到達圏 等)




```python
# ArcPy のインポート
import arcpy

# 現在開いているプロジェクトファイルを参照
aprx = arcpy.mp.ArcGISProject("CURRENT")

# プロジェクトファイルの中のレイアウトを検索し取得
lyt = aprx.listLayouts("レイアウト")[0]

# PDFにエクスポート
lyt.exportToPDF(r"C:¥Data¥output¥export.pdf")
```

役割に応じたいくつかのモジュールを保有

Python/ArcPy を使うとできること

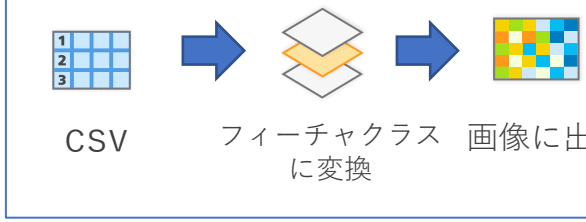
- ArcGIS 上での一連の処理をスクリプト化
  - プロパティの設定
  - ジオプロセシング ツールの実行
  - マップ操作
- 条件分岐を含む処理
  - if（もし～なら）

```
if format == "csv": #ファイル形式が CSV なら
    処理 1 を実行

if format == "shp": #ファイル形式がシェープファイルなら
    処理 2 を実行
```

- 繰り返し (反復) 処理
  - for (～条件範囲で繰り返す)

```
for layer in layers: #レイヤーの数分繰り返す
    繰り返し処理を実行
```

- ジオプロセシング ツール化し汎用化
  - 一連の処理を ArcPy で記述しツールとして登録
  - 作成したツールの保存・再配布も可能



× 100

1 回の命令で実行！

ジオプロセシング ツール

ArcPy を使用するメリット

通常は…

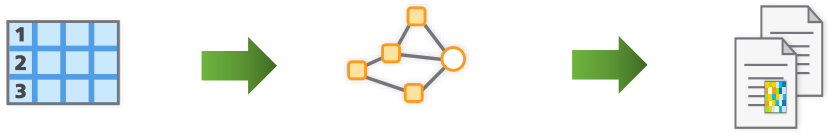
①ポイント データ化　②シンボル/レイアウトなどの設定　③画像ファイルの出力

利用例

【利用者】〇〇市保健所
【業務内容】週ごとに報告されるインフルエンザ罹患者の位置情報を地図上に視覚化し、レポートを作成する
【必要な作業】
1) CSV で報告される罹患者の住所を GIS データとしてポイント化
2) 罹患者の年齢でポイントのシンボルを設定
3) 報告用に地図や作成日時、シンボルの凡例などのレイアウトを設定し、画像ファイルとして出力

ArcPy を使用すると

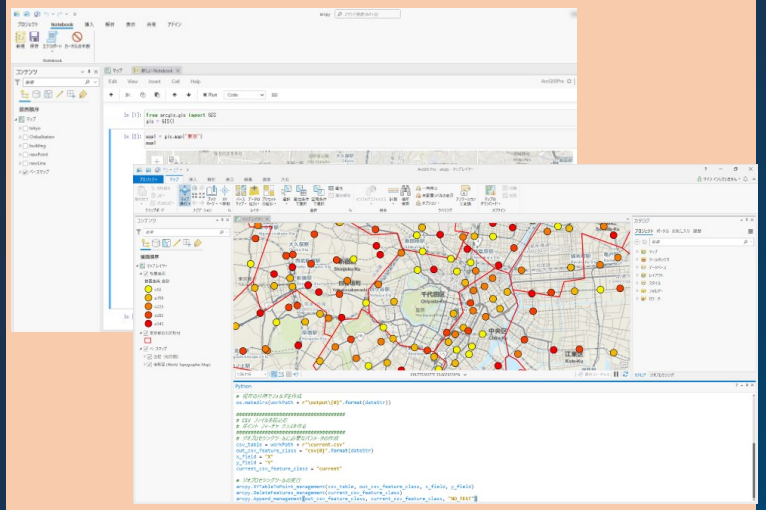
①CSV をフォルダに配置し、Python スクリプトを実行　②画像ファイルが自動で出力



ArcPy を実行する環境

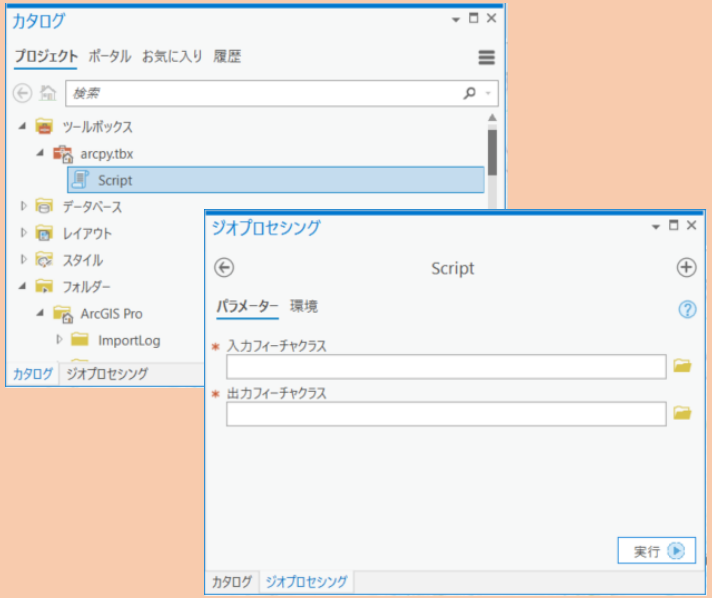
目的に応じて実行方法を選択

Python ウィンドウ/ノートブック



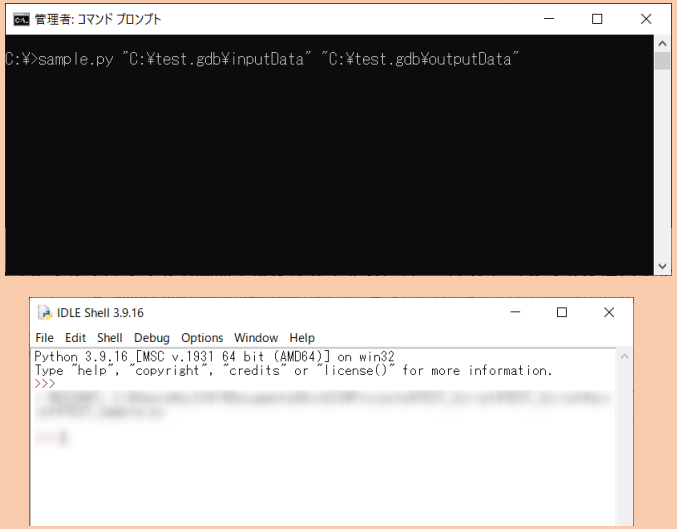
マップを見ながら
対話的に処理ができる

スクリプト ツール



ジオプロセシング
ツールと同様に使える

コマンド/開発ツール



アプリケーションの
起動が不要

2. 処理の自動化

業務の想定

定期的にファイル入手、GIS で加工しレポート作成

xxxx 年 xx 月分の
ファイル受領

- CSV ファイル
- シェープファイル
- …

↓

GIS で加工

- 値の集計
- 主題図作成
- …

↓

xxxx 年 xx 月の
レポート作成

- PDF 出力
- お客様へ提出
- …

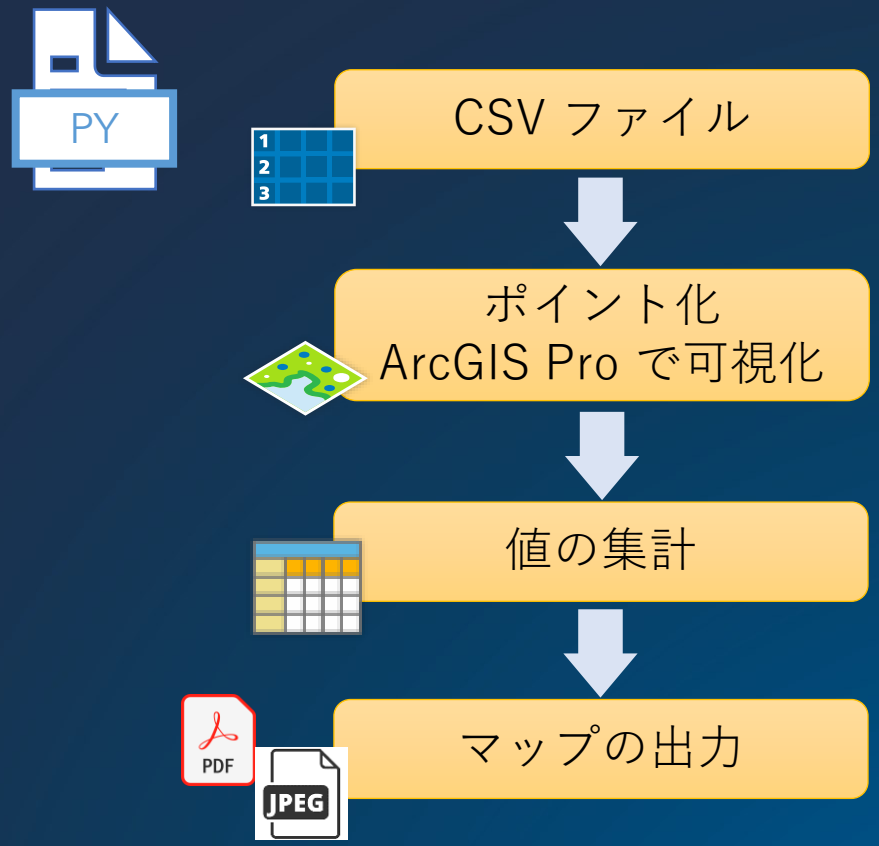
毎月同じ作業を実施

↓

ArcPy で作業を効率化！

デモの内容

東京都の鉄道各駅の放置車両台数をマップ上に可視化して、
ファイル形式で出力！



・CSV の位置情報をポイント化！
・シンボル設定されたポイントデータを
ファイル形式で出力！

今回の使用環境:
ArcGIS Pro 3.2.2
Python 3.9.18

デモの内容

- デモで使用する CSV (サンプル データ)
  - 経度 (X)
  - 緯度 (Y)
  - 駅名
  - 放置自転車の台数 (bicycle)
  - 放置原付の台数 (scooters)
  - 放置自動二輪の台数 (motorcycle)
  - 放置台数の合計 (total)
    - ArcPy のカーソル機能で作成

	OBJECTID *	Shape *	X	Y	駅名	bicycle	scooters	motorcycle	total
1	1	ポイント	139.839999	35.747602	お花茶屋駅	95	7	1	103
2	2	ポイント	139.778647	35.629825	お台場海浜公園駅	2	0	0	2
3	3	ポイント	139.481864	35.517121	すずかけ台駅	0	0	0	0
4	4	ポイント	139.484984	35.527165	つくし野駅	0	0	0	0
5	5	ポイント	139.798637	35.712376	つくばエクスプレス浅草駅	187	11	4	202
6	6	ポイント	139.574951	35.657928	つつじヶ丘駅	17	1	0	18
7	7	ポイント	139.779411	35.617559	テレコムセンター駅	23	0	0	23
8	8	ポイント	139.808884	35.710317	とうきょうスカイツリー駅	11	0	1	12
9	9	ポイント	139.688812	35.758822	ときわ台駅	21	1	0	22
10	10	ポイント	139.545805	35.751474	ひばりヶ丘駅	2	0	0	2
11	11	ポイント	139.308404	35.64357	めじろ台駅	2	0	0	2
12	12	ポイント	139.635682	35.704881	阿佐ヶ谷駅	73	0	0	73

- 参考
  - 東京都生活文化スポーツ局 都内における駅前放置自転車の現況 令和5年度 https://www.seikatubunka.metro.tokyo.lg.jp/tomin_anzen/kotsu/jitensha/houchi/0000001962.html

処理の流れ

CSV ファイルを読み込む

ポイント フィーチャクラスを作る

各駅ごとの放置車両数を集計する

マップの出力

プラス α:
Web へ レイヤーを共有

使用する ArcPy の機能

①ジオプロセシング ツール

②カーソル

③マッピング モジュール

プラス α:
シェアリング モジュール

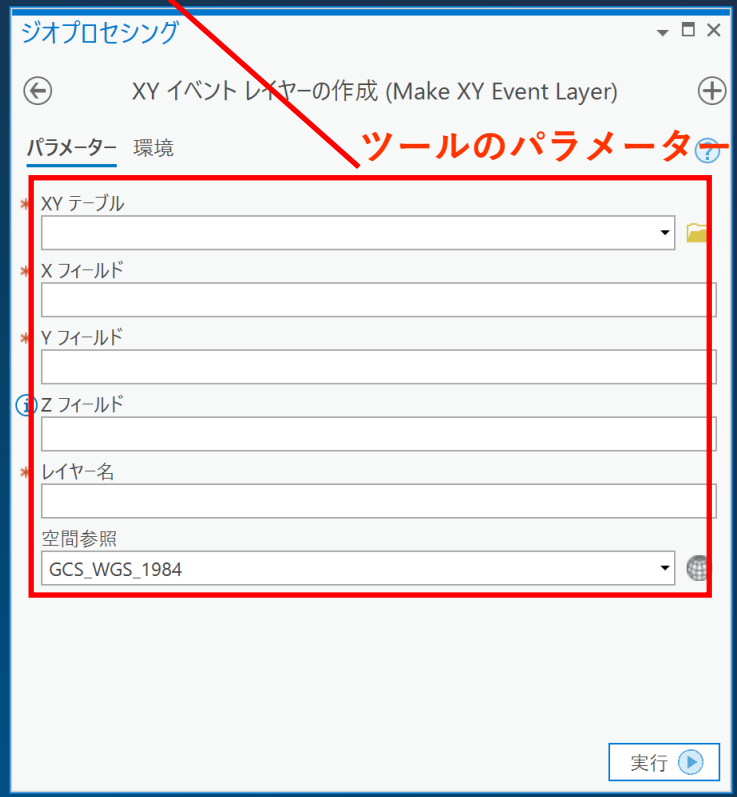
ArcPy の主要機能

①ジオプロセシング ツール

ArcPy からジオプロセシング ツールを実行可能

```
# ジオプロセシング ツールの基本的な呼び出し方
arcpy.management.MakeXYEventLayer(table, x_field , y_field , out_layer )
```

ツールボックスの
エイリアス名

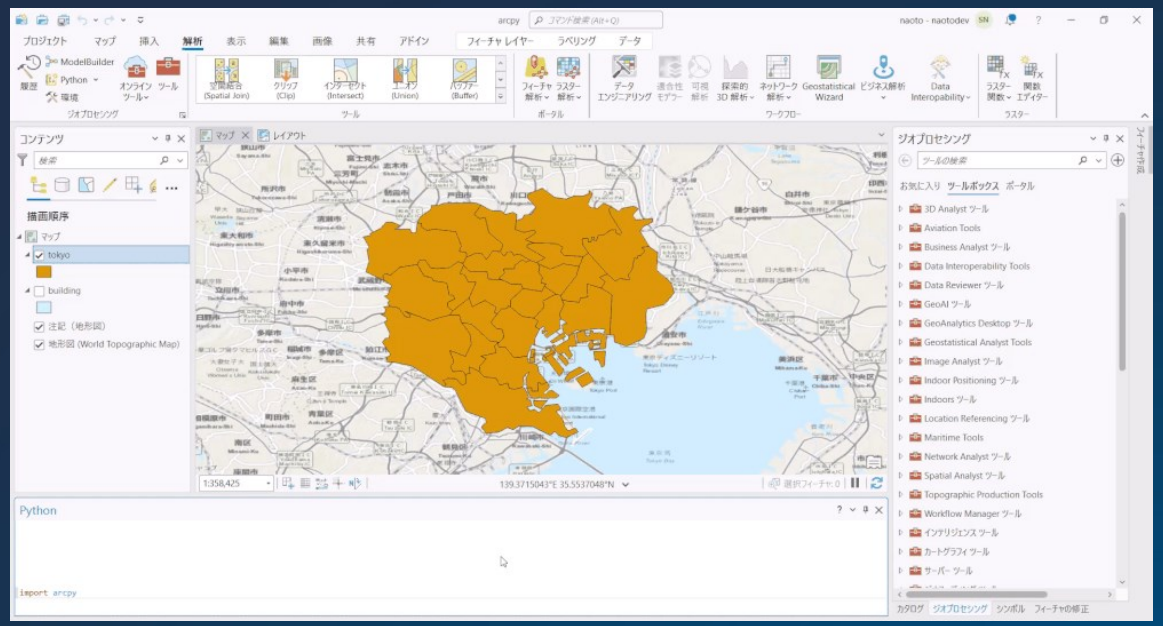


[データ管理ツール] > [レイヤーとテーブル ビュー] > [XY イベント レイヤーの作成] ツールの例

ArcPy の主要機能

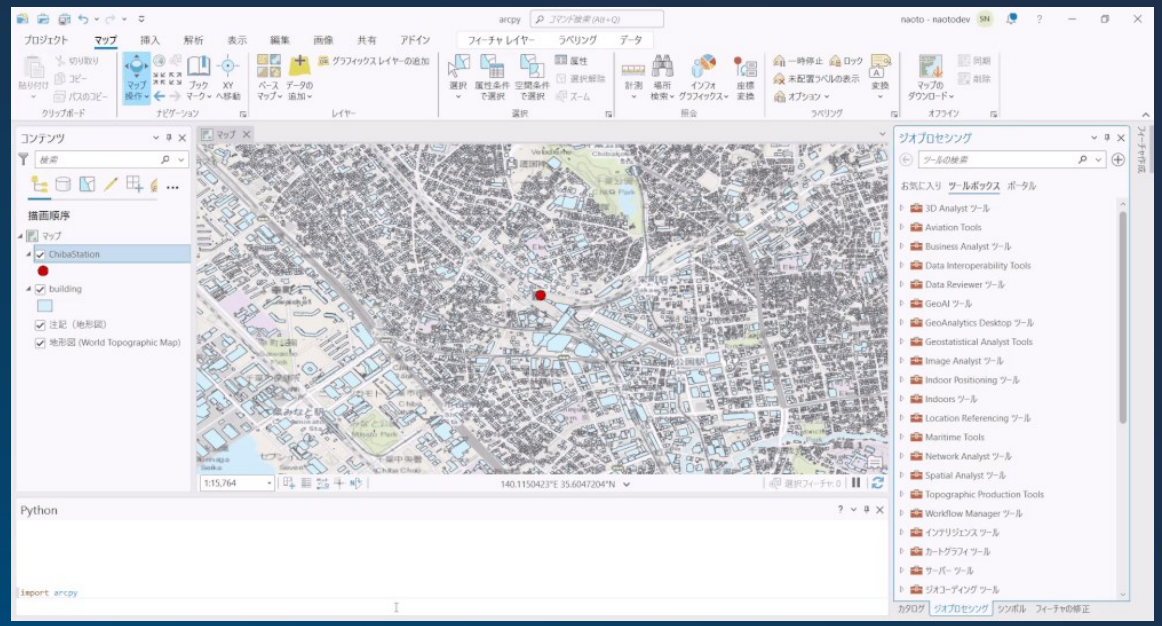
①ジオプロセシング ツール – その他の処理

- 属性検索



```
# tokyo レイヤーから新宿区のフィーチャを選択
arcpy.management.SelectLayerByAttribute ("tokyo","NEW_SELECTION", "CITY1 = '新宿区'")
```

- 空間検索



```
# 千葉駅から 500 メートル以内にある建物を検索
arcpy.management.SelectLayerByLocation("building","WITHIN_A_DISTANCE","ChibaStation", "500 meters", "NEW_SELECTION")
```

ArcPy の主要機能

①ジオプロセシング ツール – デモ

```
# ArcPy サイト パッケージをインポートします。
import arcpy

# ジオプロセシング ツールで利用する変数を定義
csv = r"C:¥data¥HandsOn.csv"
x = "X"
y = "Y"
layer = "PointLayer"
outpath = r"C:¥data¥ArcGISPro¥arcpy.gdb"

# CSV ファイルから XY イベント レイヤー （テンポラリ のオブジェクト） を作成します。
arcpy.management.MakeXYEventLayer(csv, x, y, layer)

# XY イベント レイヤーをフィーチャ クラスとして保存します。
arcpy.Conversion.FeatureClassToFeatureClass(layer, outpath, "csv001")

# XY イベント レイヤーを削除します。
arcpy.management.Delete(layer)
```

※PDF のため、動画は省略しています。

ArcPy の主要機能
②カーソル

- テーブルの操作
  - SearchCursor：読み取り専用でアクセス
  - InsertCursor：行の挿入
  - UpdateCursor：行の更新、削除など

ID	駅	自転車	原付	自動二輪	合計
1	永田町駅	10	2	6	
2	六本木駅	33	4	2	
3	赤坂見附駅	25	6	0	
4	池袋駅	53	11	1	

ID	駅	自転車	原付	自動二輪	合計
1	永田町駅	10	2	6	18
2	六本木駅	33	4	2	39
3	赤坂見附駅	25	6	0	31
4	池袋駅	53	11	1	65

特定フィールドの値を更新する例

```
with arcpy.da.UpdateCursor(<フィーチャクラス>, <フィールド名>) as cursor:
    for row in cursor:
        row[フィールド インデックス] = 属性値
        cursor.updateRow(row)
```

フィーチャクラスの各行（フィーチャ）に対して繰り返し処理を行う

ArcPy の主要機能

②カーソル – デモ

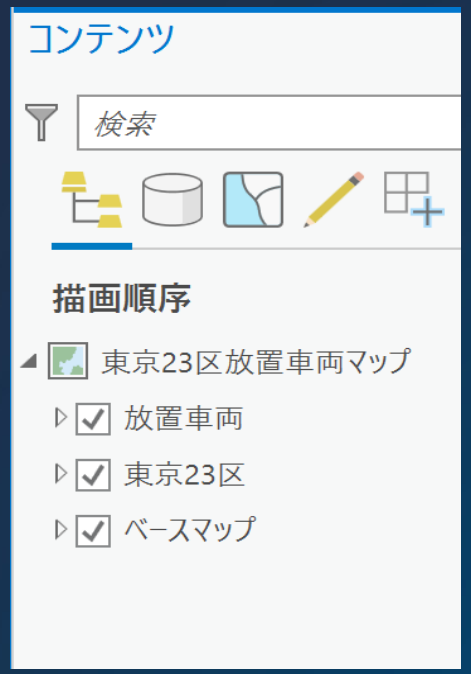
```
# ArcPy サイト パッケージをインポートします。
import arcpy

# csv001 フィーチャ クラス（既に作成）に LONG 型の total フィールドを追加します。
arcpy.management.AddField("csv001","total","LONG")

# csv001 フィーチャ クラスから取得するフィールド名のリストを作成
fields = [‘total’, ‘bicycle‘,  'scooters’, ’motorcycle']

# フィーチャ クラス (csv001) に対してカーソルを取得
cursor = arcpy.da.UpdateCursor("csv001", fields)

# for xx in :で、カーソルを移動しながら値を更新
for row in cursor:
    row[0] = row[1] + row[2] + row[3] # bicycle （インデックス番号:1 ）～ motorcycle （インデックス番号:3 ）の合計値
    cursor.updateRow(row) # 合計値の値を適用

# オブジェクトを削除して参照を解放します。
del cursor, row
```

※PDF のため、動画は省略しています。

ArcPy の主要機能

③マッピング モジュール

- マップ/レイヤーの操作
  - メタ データやシンボル等、各種プロパティの設定
  - 画像/PDF 出力

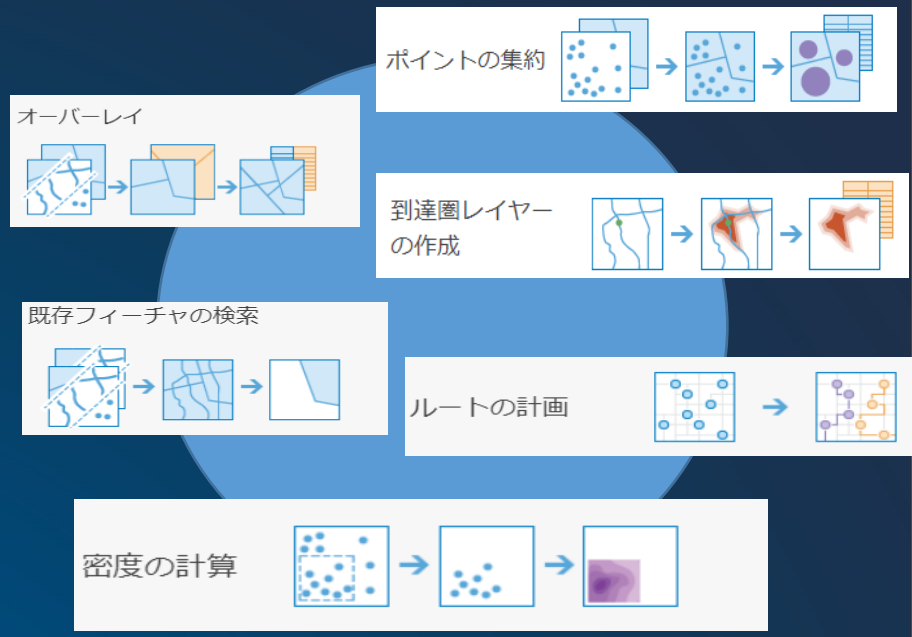


＝プロジェクト ファイル
（＊.aprx）

＝マップ[0]
＝レイヤー[0]
＝レイヤー[1]
＝レイヤー[2]

レイヤーのリスト

レイヤー
オブジェクト

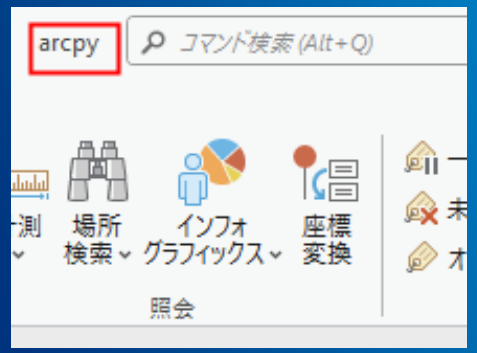


解析処理や設定

ArcPy の主要機能

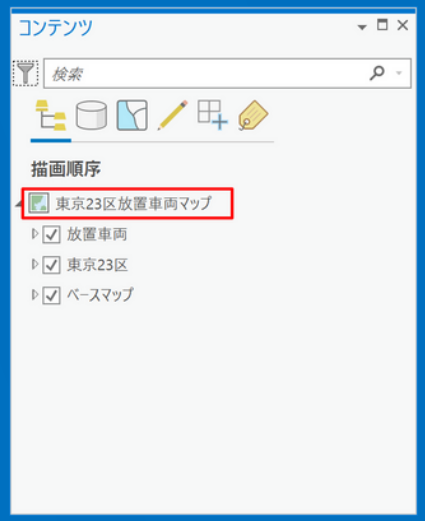
マッピング モジュールを使ったマップの操作方法①

プロジェクト ファイルの取得



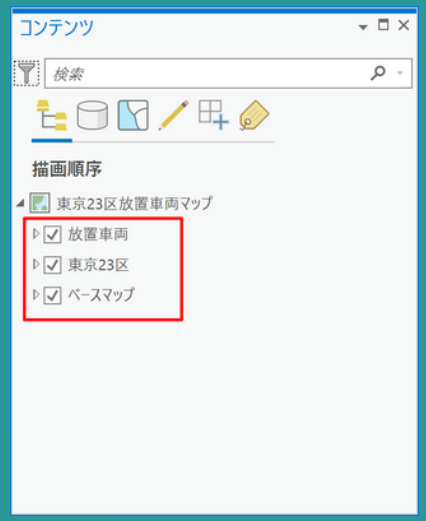

```
# 現在開いているプロジェクトファイルを取得
aprx = arcpy.mp.ArcGISProject("CURRENT“)
```

マップの取得



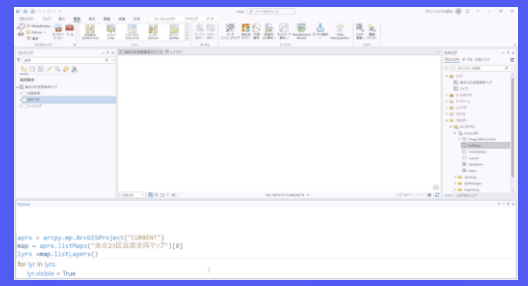

```
# プロジェクト ファイルのマップを取得
maps = aprx.listMaps(“マップ")[0]
```

レイヤーの取得




```
# マップに含まれるレイヤーの一覧（リスト）を取得
lyrs =maps.listLayers()
```

レイヤーの操作

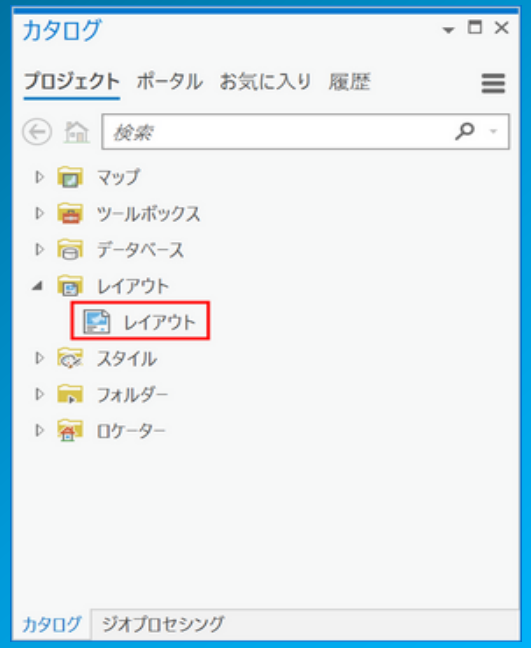


```
# ArcPyをインポート
import arcpy
# プロジェクト ファイルのオブジェクト取得
aprx = arcpy.mp.ArcGISProject(“CURRENT”)
# マップのオブジェクト取得
maps = aprx.listMaps()[0]
# レイヤーのオブジェクト取得
lyrs =maps.listLayers()
# レイヤーの数だけ繰り返し
for lyr in lyrs:
    # レイヤーの表示設定
    lyr.visible = True
```

ArcPy の主要機能

マッピング モジュールを使ったマップの操作方法②

レイアウトの出力




```
# 現在開いているプロジェクトファイルを取得
aprx = arcpy.mp.ArcGISProject("CURRENT")
# レイアウトのオブジェクトを取得
lyt = aprx.listLayouts()[0]
```

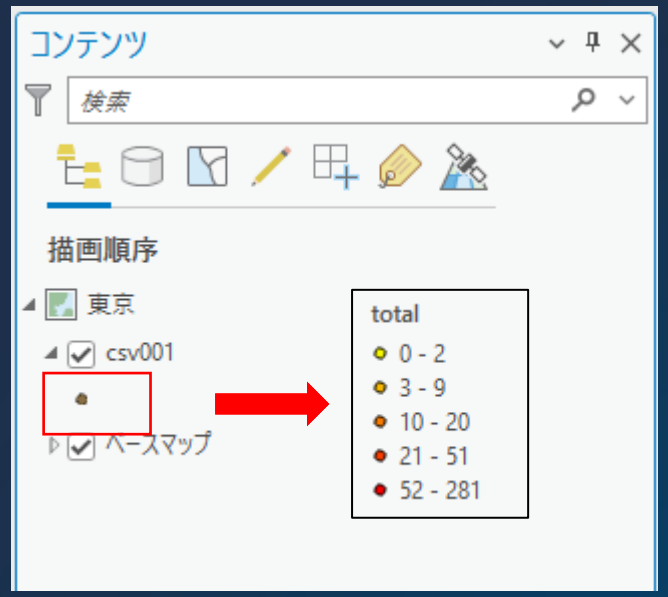
マップの出力


```
aprx = arcpy.mp.ArcGISProject("CURRENT")
lyt = aprx.listLayouts()[0]
# PDFへエクスポート
lyt.exportToPDF(r"C:¥data¥output¥Sample.pdf")
```

```
# JPEGへエクスポート
lyt.exportToJPEG(r"C:¥data¥output¥Sample.jpg")
```

ArcPy の主要機能

シンボルの更新 – updateRenderer 関数




```
# csv001 レイヤーのシンボル更新
for lyr in lyrs:
    if lyr.name == “csv001”:
        sym = lyr.symbology
        sym.updateRenderer(“GraduatedColorsRenderer”) # 等級色
        sym.renderer.classificationField = “total”  #分類するフィールド名
        sym.renderer.classificationMethod = “Quantile” # 等比間隔
        sym.renderer.breakCount = 5 # クラス
        lyr.symbology = sym
```

UpdateRenderer 関数

構文： updateRenderer (renderer_name)

パラメータ	説明	データタイプ
renderer_name	使用するレンダラーの名称	String

ArcPy の主要機能

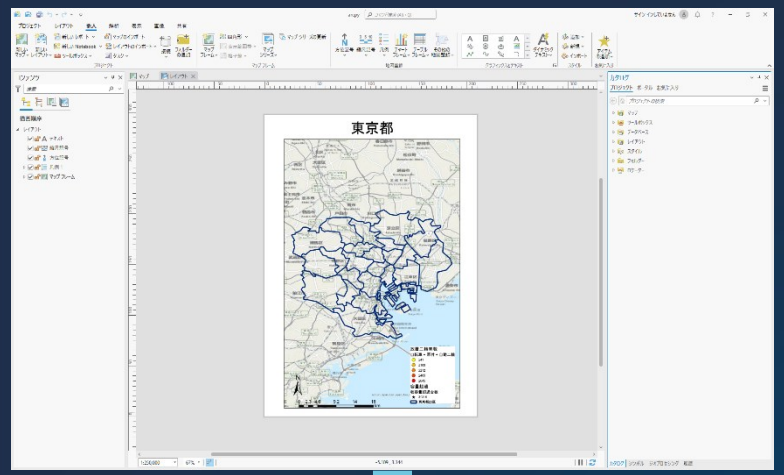
③マッピング モジュール – シンボルの更新 デモ

```
# ArcPyをインポート
import arcpy
# プロジェクト ファイルのオブジェクト取得
aprx = arcpy.mp.ArcGISProject("CURRENT")
# マップのオブジェクト取得
maps = aprx.listMaps()[0]
# レイヤーのオブジェクト取得
lyrs =maps.listLayers()

# csv001 レイヤーのシンボル更新
for lyr in lyrs:
    if lyr.name == “csv001”:
        sym = lyr.symbology
        sym.updateRenderer(“GraduatedColorsRenderer”) # 等級色
        sym.renderer.classificationField = “total”  #分類するフィールド名
        sym.renderer.classificationMethod = “Quantile” # 等比間隔
        sym.renderer.breakCount = 5 # クラス
        lyr.symbology = sym
```

※PDF のため、動画は省略しています。

ArcPy の主要機能

マップの出力 - exportToPDF 関数



PDF へエクスポート

```
#　すべてのオプションのデフォルト値を使用してページ レイアウトをPDF出力
aprx = arcpy.mp.ArcGISProject("CURRENT")
lyt = aprx.listLayouts()[0]
lyt.exportToPDF(r"C:¥data¥output¥Sample.pdf")
```

exportToPDF 関数

PDF ファイルの出力先

構文：exportToPDF (out_pdf, {resolution}, {image_quality}, {compress_vector_graphics}, {image_compression}, {embed_fonts}, {layers_attributes}, {georef_info}, {jpeg_compression_quality}, {clip_to_elements}, {output_as_image})

パラメータ	説明	データタイプ
out_pdf	出力エクスポート ファイルのパスとファイル名を表す文字列	文字列

ArcPy の主要機能

③マッピング モジュール – マップの出力 デモ

```
# ArcPy サイト パッケージをインポートします。
import arcpy

# プロジェクト ファイルのオブジェクト取得
aprx = arcpy.mp.ArcGISProject("CURRENT")

# レイアウト[0]を取得
layout = aprx.listLayouts()[0]

#   ページ レイアウト設定内容を PDF へ出力
layout.exportToPDF(r"C:¥data¥output¥Sample.pdf")
```

※PDF のため、動画は省略しています。

ArcPy の主要機能

番外：シェアリング モジュール

- サービスの公開
  - ArcGIS OnlineやArcGIS Enterpriseへフィーチャ レイヤー等を公開

ArcGIS Online公開の流れ

サービス定義ドラフトの作成

↓ ステージング

サービス定義ファイルの作成

↓ サービス定義のアップロード

レイヤー・サービスの公開 ArcGIS Online

```
project = arcpy.mp.ArcGISProject(<aprxのファイルパス>)
aprx = project.listMaps(<マップ名>)[0]
lyrs = []
lyrs.append(aprx.listLayers(<レイヤー名>)[0])
sharing_draft = aprx.getWebLayerSharingDraft(<サーバータイプ>,<サービスタイプ>,<サービス名>, lyrs)
sharing_draft.exportToSDDraft(<サービス定義ドラフトファイル名>)
arcpy.StageService_server(<サービス定義ドラフトファイル名>, <サービス定義ファイル名>)
arcpy.UploadServiceDefinition_server(<サービス定義ファイル名>,<サーバータイプ>)
```

3. おすすめ情報

初めての方へ

トレーニング

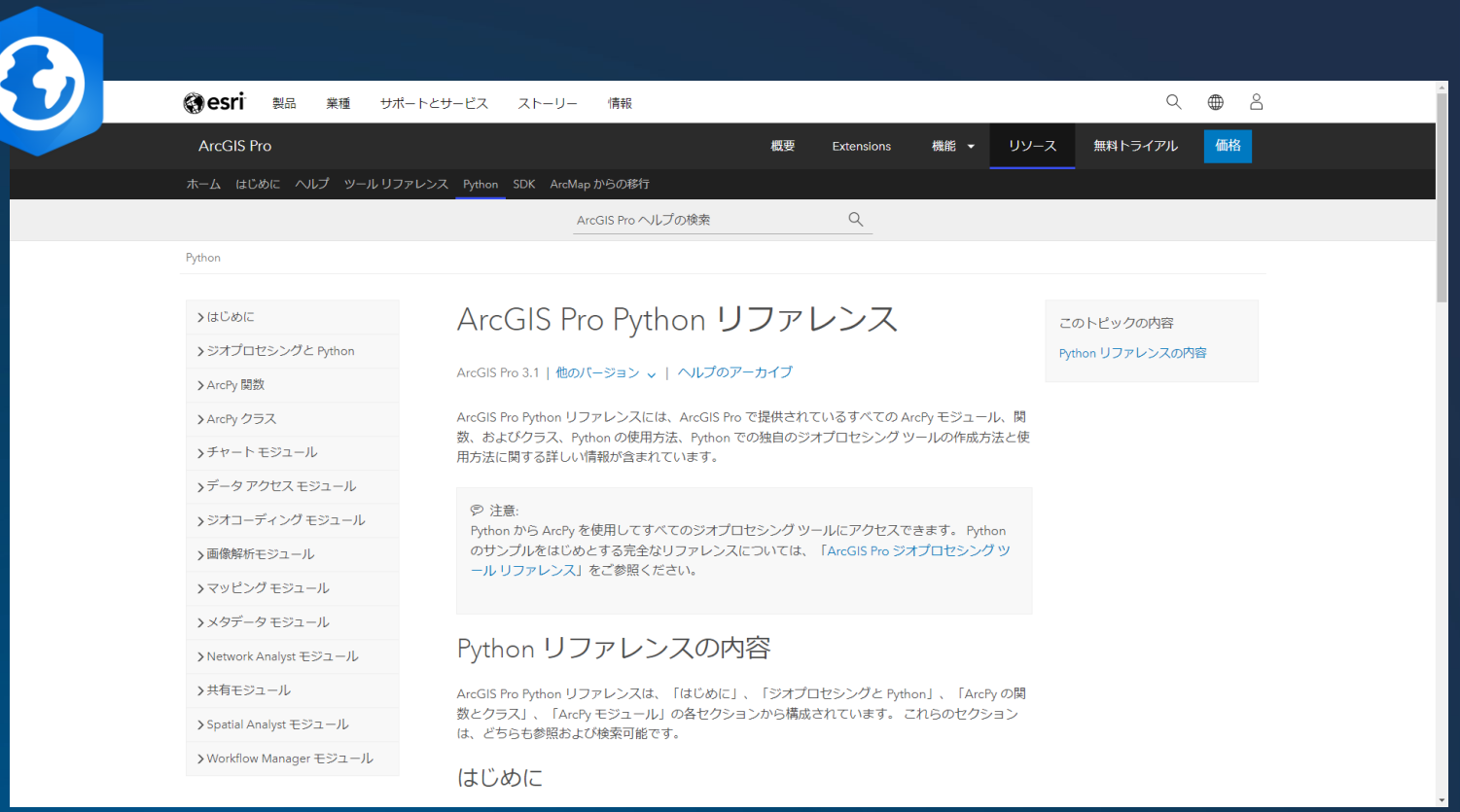
- ArcGIS Pro: Python スクリプト入門
  https://www.esrij.com/training/courses/developer/et30/

  - **内容:** データの編集やジオプロセシング ツールの実行などの処理を
    自動化するための基礎的な Python スクリプトの記述方法を習得する
  - **対象者:** Python を使用してどんなことができるか知りたい方
    処理の自動化を考えている方
  - **前提知識:** ArcGIS Pro 経験者
    Python の基本的な知識を有する方

Web ヘルプ

- 機能解説、リファレンス、サンプル コード

- ArcGIS Pro
  - https://bit.ly/2poH7cS



4. まとめ

まとめ

- Python/ArcPy の基礎知識
  - Python/ArcPy で出来ること
  - 実行方法

- Python/ArcPy を使用した自動化処理
  - ジオプロセシング ツールの呼び出し
  - テーブルの操作 (カーソル)
  - マップの出力 (マッピング モジュール)

- おすすめ情報
  - 初めて使う方に向けた学習方法など

参考資料

- ESRIジャパン株式会社
  - ArcPy | ESRIジャパンArcGIS の基礎知識 | GIS 基礎解説 | ESRIジャパン (esrij.com)
- ESRI ジャパン GitHub
  - GitHub - EsriJapan/arcpy-resources: ArcPy 学習用資料
- Esri Community
  - ArcGIS 開発者コミュニティ
    - 【ArcGIS Pro 版】Python を使って作業の効率化を図ろう!① : ArcPy の基礎 - Esri Community
    - 【ArcGIS Pro 版】Python を使って作業の効率化を図ろう!② : マップ・レイヤーの操... - Esri Community
    - 【ArcGIS Pro 版】Python を使って作業の効率化を図ろう!③ : ジオプロセシング ツ... - Esri Community
    - 【ArcGIS Pro 版】Python を使って作業の効率化を図ろう!④ : データの操作 - Esri Community

ご参加いただき誠にありがとうございました。

アンケートのご協力をお願い致します。

https://arcg.is/1zz5fD0

esri ジャパン