『SS7 Op.Python実行』を用いたPythonプログラムを実行ファイル（EXE）に変換する方法

Op.Python実行はPythonの実行環境が必要ですが、プログラムをEXE化することで、Pythonの実行環境がインストールされていない環境でもプログラムを実行することができます。
ここでは、PyInstallerを使用して、Op.Python実行をEXE化する手順を説明します。

1. 実行ファイル変換の注意点

- ファイルサイズが大きくなる

  Pythonの実行環境やライブラリが含まれるため、配布時の容量が大きくなります。
  PyInstallerのオプションを調整することで、ファイルサイズを削減することができる場合があります。

- エラーの原因特定が難しくなる

  ソースコードを確認できなくなるため、エラーの原因を特定するのが難しくなります。
  エラー時にログを出力する仕組みを作成することが有効です。

- 外部ライブラリのバージョン固定

  外部ライブラリのバージョンはEXE化時にインストールしているバージョンに固定されます。
  各ライブラリの更新情報を確認の上、ライブラリのバージョンを更新し再EXE化する必要があります。

2. 実行ファイルに変換する手順

1. PyInstallerのインストール
2. 『SS7』の実行環境管理モジュールの追加
3. EXEファイルの作成

2.1. PyInstallerのインストール

Pythonの実行環境に入り、以下のコマンドを実行して、PyInstallerをインストールします。

```
pip install pyinstaller
```

2.2. 『SS7』の実行環境管理モジュールの追加

プログラムに『SS7』の実行環境管理モジュールである、`ss7_environment_manager.py`を追加し、必要な設定を行います。

以下のコードは、『SS7 Op.Python実行』を用いる際に必要な設定を記述したコードです。
詳細については、第3章『SS7』の実行環境管理モジュールを参照してください。

- バージョンを指定する例

```
from Python import Ss7Python as Cmd
from ss7_environment_manager import SS7EnvironmentManager

# `Ver.1.1.1.20`の『SS7』環境を設定
env_manager = SS7EnvironmentManager("1.1.1.20")

# Op.Python実行の初期化
Cmd.Init()
Cmd.Start(env_manager.get_version(), 1) # Ver.1.1.1.20の『SS7』が起動

# 以下にコードを記述
...
```

- バージョンを指定せずに、最新のバージョンを使用する例

```
from Python import Ss7Python as Cmd
from ss7_environment_manager import SS7EnvironmentManager

# 最新の『SS7』環境を設定（例：Ver.1.1.1.20が最新の場合）
env_manager = SS7EnvironmentManager()

# Op.Python実行の初期化
Cmd.Init()
Cmd.Start(env_manager.get_version(), 1) # Ver.1.1.1.20の『SS7』が起動

# 以下にコードを記述
...
```

2.3. 実行ファイルの作成

PyInstallerのコマンドを実行して、EXEファイルを作成します。
以下は、EXE化するプログラムが `main.py` として保存されている場合のコマンドです。

```
pyinstaller main.py
```

- PyInstallerのオプション

```
--onefile: 1つのEXEファイルにまとめる
--noconsole: コンソールを表示しない
--clean: 一時ファイルを削除
```

以下は、実行ファイルを1つのEXEファイルにまとめ、EXEを実行時にコンソールを表示しない場合のコマンドです。

```
pyinstaller --onefile --noconsole main.py
```

コマンドを実行後、`dist`ディレクトリにEXEファイルが作成されます。

参考: PyInstaller

3.『SS7』の実行環境管理モジュール

3.1. モジュールの概要

実行環境管理モジュール（`ss7_environment_manager.py`）は、EXE化したプログラムで『SS7』を適切に実行するために必要な環境を管理します。

主な機能：

- バージョン管理：インストール済みの『SS7』バージョンの検出と選択
- 環境設定：実行に必要なパスの設定
- エラー処理：詳細なエラーメッセージの提供

3.2. 動作要件

3.2.1. インストール要件

- 『SS7』のインストールパス
  - デフォルト: `C:\Program Files\UsrWeb64\Ss7win`
  - インストール先が異なる場合は、`SS7EnvironmentManager`の`self._base_path`を変更してください。

3.2.2. バージョン要件

- 特定のバージョンを指定することができます。
- 指定がない場合は最新バージョンを使用します。

3.3. 使用方法

3.3.1. 基本的な使い方

最新バージョンの『SS7』を使用する場合：

```
from Python import Ss7Python as Cmd
from ss7_environment_manager import SS7EnvironmentManager

try:
    # 最新バージョンで環境を設定
    env_manager = SS7EnvironmentManager()

    # Op.Python実行の初期化
    Cmd.Init()
    Cmd.Start(env_manager.get_version(), 1)
```

```
except SS7EnvironmentError as e:
    print(f"環境設定エラー: {e}")
```

3.3.2. バージョンを指定する場合

特定のバージョンの『SS7』を使用する場合：

```
try:
    # Ver.1.1.1.20の環境を設定
    env_manager = SS7EnvironmentManager("1.1.1.20")

    # Op.Python実行の初期化
    Cmd.Init()
    Cmd.Start(env_manager.get_version(), 1)

except SS7EnvironmentError as e:
    print(f"環境設定エラー: {e}")
```

著作者

ライセンス