第7章 PyAEDT

7.1 簡介

PyAEDT是一個Python庫，直接與Ansys Electronics Desktop (AEDT)的API進行交互，使腳本編寫更簡單。PyAEDT的架構可以被所有AEDT 3D產品（如HFSS, Icepak, Maxwell 3D,和Q3D Extractor）、2D工具，以及Ansys Mechanical重用。PyAEDT也支援像Nexxim這樣的電路工具和像Twin Builder這樣的系統模擬工具。最後，PyAEDT為Ansys的佈局工具（如HFSS 3D Layout和EDB）提供了腳本編寫能力。PyAEDT的類和方法結構簡化了操作，同時在API中盡可能地重用信息。

PyAEDT可以在CPython3.10上從PyPI安裝，也可以從Conda-Forge安裝。PyAEDT仍然與IronPython兼容，並且可以在AEDT框架中使用。

PyAEDT是更大的PyAnsys努力的一部分，旨在直接從Python使用Ansys技術。PyAEDT旨在整合並擴展所有現有的關於AEDT腳本的功能，以允許重用現有的代碼，分享最佳實踐，並增加協作。

PyAEDT是在MIT許可下授權的。PyAEDT包括與以下AEDT工具和Ansys產品交互的功能：

- HFSS和HFSS 3D Layout
- Icepak
- Maxwell 2D, Maxwell 3D, 和 RMXprt
- 2D Extractor 和 Q3D Extractor
- Mechanical
- Nexxim
- EDB
- Twin Builder

要運行PyAEDT，您必須有一個本地授權的AEDT副本。PyAEDT支持AEDT版本2022 R2及更高版本。PyAEDT支持AEDT學生版本2022 R2及更高版本。

PyAEDT提供的主要優點包括：

- 自動初始化所有AEDT對象，如編輯器、邊界等
- 錯誤管理
- 日誌管理
- 變量管理
- 與IronPython和CPython的兼容性
- 使用數據對象簡化複雜的API語法，同時保持PEP8兼容性
- 在不同的求解器之間重用代碼
- 對函數和API的清晰文檔
- 對代碼的單元測試，以提高在不同的AEDT版本中的質量

7.2 ANSYS Python Manager

Ansys Python Manager 是一款由 Ansys 團隊開發、以 MIT 授權發布的開源 Python QT 應用。它為 Ansys 的開發者提供一個輕便的 GUI 界面，方便管理 Python 安裝、虛擬環境、常用 Python 包，以及 PyAnsys 包。

主要功能包括：

1. Python 安裝：輕鬆選擇並安裝所需的 Python 版本。
2. 虛擬環境：創建與管理 Python 的虛擬環境。
3. IDE 和開發環境：啟動與管理開發工具。
4. 包管理：安裝 PyAnsys 包和管理依賴關係。
5. 文檔訪問與更新：提供文檔資源和應用更新選項。

使用方法：

1. 下載安裝：訪問 Ansys Python Manager GitHub 發布頁面 ，從 Assets 區域下載並運行安裝程序。
2. 啟動與使用：安裝後從開始菜單啟動，選擇所需 Python 版本，創建並管理虛擬環境。

界面功能：

- 「Python 安裝」選項：選擇標準 Python 或 Condaforge Python（社區驅動版）進行安裝。
- 「Create Virtual Environments」標籤頁：輕鬆創建虛擬環境，指定版本和名稱。
- 「Manage Virtual Environments」標籤頁：查看、管理虛擬環境，進行包安裝或刪除等操作。

總結來說，Ansys Python Manager 是一款功能全面的 Python 管理工具，特別適合 Ansys 開發者使用。

7.3 PyAEDT的編程流程

使用PyAEDT進行程式開發的流程與傳統的編程方式大相逕庭，它更像是一種交互式的體驗。開發者能夠邊撰寫程式碼，邊實時觀察到AEDT的變化反應。例如，你只需輸入一行指令來創建新的專案和設計並運行程式碼，就能立即在AEDT中看到新的專案和設計被創建。接著，你可以返回編輯器，新增新的程式碼塊，在該塊中，向所創建的設計中加入新的變數。此時，只需運行該程式碼塊，你就可以在設計中看到新變數的加入。

在以往，當我們使用AEDT API進行程式設計時，需要從頭到尾運行整個程式。若程式的前半部有一些耗時的操作，每次調試都需等待這些操作完成，對於開發人員來說，這無疑是一種巨大的壓力。然而，現在的PyAEDT結合編輯器的程式碼塊執行功能，大幅減輕了這種壓力，使我們可以逐步地開發和擴展程式。如果需要，我們甚至可以返回到之前的程式碼塊，修改某些條件，然後再回到最新的程式碼塊繼續執行。例如，若在程式碼的後半部分發現忘記修改材料屬性，我們可以返回到先前的程式碼塊，加入並運行修改材料屬性的程式碼，並可以在AEDT環境中確認修改是否正確，然後再回到最新的程式碼塊繼續開發。

與傳統的AEDT API開發流程相比，由於PyAEDT的指令不能透過錄製功能生成，一開始可能會認為使用PyAEDT進行開發將非常耗時。然而，實際上，這種互動式的自動化編程方式可以讓我們在小範圍內進行修改和測試，再加上編輯器的智能感知（Intellisense）功能可以幫助我們選擇適當的物件方法，這些都將大幅縮短程式開發的時間。

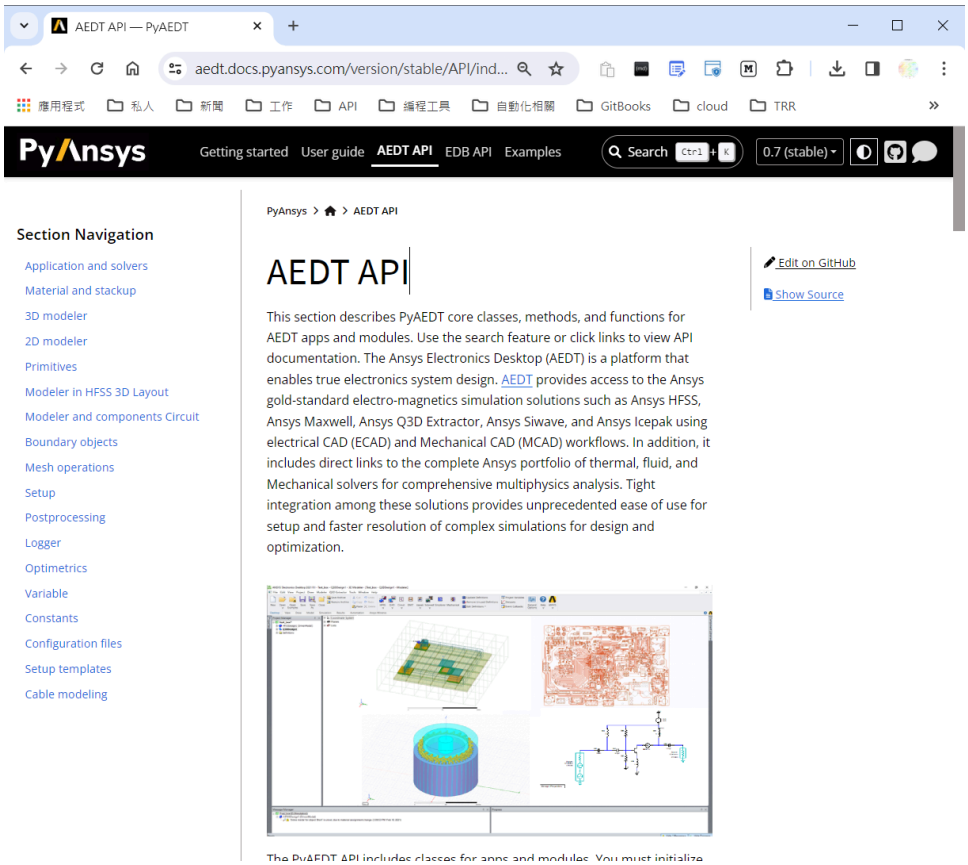
在pyaedt中，對屬性的CRUD操作（創建，讀取，更新和刪除）通常涉及到特定的ANSYS模塊，例如HFSS或Maxwell。以下是一些基本的方法：

- 創建(Create)：使用相關的API方法創建新的對象或屬性。例如，如果你正在使用HFSS，你可能會使用pyaedt.Hfss.create_object()方法來創建新的3D對象。你也可以創建新的屬性，比如物料屬性。
- 讀取(Read)：可以使用pyaedt.Hfss.get_property()或類似的方法讀取特定的屬性值。例如，你可能需要讀取一個電場的值或者一個對象的幾何參數。
- 更新(Update)：可以使用pyaedt.Hfss.set_property()或類似的方法更新特定的屬性值。例如，你可能需要更改一個對象的幾何參數或者更改一個物料的屬性。

- 刪除(Delete)：可以使用pyaedt.Hfss.delete_object()或類似的方法刪除特定的對象或屬性。

7.4 AEDT的API查詢

PyAEDT完整的API文件可以在 API文檔網頁查詢。這個頁面提供了詳細的信息和指南，幫助用戶理解和使用PyAEDT提供的各種功能。文檔涵蓋了從應用程序和求解器的初始化到材料和堆疊管理、模型建構、邊界設置、網格操作、後處理等眾多方面。這些內容對於希望利用PyAEDT進行電磁場仿真的工程師和研究人員來說非常有用。



PyAEDT API查詢頁面

文檔的主要部分包括：

1. **應用程序和求解器** ：介紹了如何使用不同的應用程序和求解器，例如HFSS、Maxwell、Q3D、Icepak等。
2. **材料和堆疊管理** ：提供了關於如何管理仿真中使用的材料和堆疊的信息。
3. **3D和2D模型建構** ：介紹了模型建構的基本操作和方法。
4. **邊界對象和源** ：說明了如何設置仿真的邊界條件和激勵源。
5. **網格操作和設置** ：提供了關於如何進行網格細分和仿真設置的指南。
6. **後處理** ：介紹了如何使用後處理器分析和可視化仿真結果。
7. **優化和變數管理** ：涉及了參數化設計和優化以及變數管理。
8. **配置文件和模板** ：提供了有關配置仿真的模板和選項的詳細信息。

這些文檔對於想要深入了解PyAEDT的用戶來說是一個非常寶貴的資源。通过這些文檔，用戶可以更好地理解如何有效地使用PyAEDT進行複雜的電磁仿真和多物理場分析。

7.5 利用PyAEDT腳本建立一個新的設計

這裡我們開始一個簡單腳本的練習, 首先在IDE當中輸入以下程式碼並執行：

```
import pyaedt

print(dir(pyaedt))
```

pyaedt模組中導入了所有可用的類別和方法。在這個模組中，你可以看到對應於各種ANSYS產品的類別，如Hfss（高頻結構仿真系統）、Edb（電路板資料庫）、Maxwell3d（電磁場仿真軟體）等等。

```
['Circuit', 'Desktop', 'Edb', 'Emit', 'Hfss', 'Hfss3dLayout', 'Icepak', 'Maxwel
```

這些類別可以讓你使用Python來操作和自動化相對應的ANSYS軟體。每一個類別都提供了一套方法，讓你可以設定模擬、運行模擬、分析結果等等。如果我們只需要使用到單一一套軟體，比方說HFSS，我們可以刪除原本代碼，並輸入以下代碼：

```
from pyaedt import Hfss

# 創建一個新的Hfss物件
hfss = Hfss(specified_version='2023.2')
```

Hfss是pyaedt套件中的一個類別，這個類別用來對應ANSYS HFSS的軟體功能。在Python中，類別名稱通常會使用駝峰命名法，也就是說，類別的名字開頭會是大寫字母。這是Python的一個慣例，雖然不是強制的，但大部分的程式設計師都會遵守。所以你會看到Hfss這個類別的H是大寫的。

使用這個類別，你可以創建一個新的Hfss物件(物件名為hfss)，並使用它來控制和自動化ANSYS HFSS的功能。創建Hfss物件時，你可以指定你想要使用的ANSYS HFSS軟體的版本。在代碼中指定了使用'2023.2'這個版本。執行上面代碼，成功的話會開啟新的AEDT專案，並在專案當中建立一個新的HFSS設計。

最後輸入以下代碼並執行：

```
hfss.save_project('c:/demo/test.aedt')

hfss.close_desktop()
```

這個方法將會將你的HFSS專案儲存到c:/demo/test.aedt這個路徑。你需要確保這個c:/demo路徑是存在的，否則可能會發生錯誤。close_desktop方法用來關閉ANSYS的Desktop應用。呼叫這個方法後，ANSYS的Desktop應用將會被關閉，所有的未儲存的專案也將會被關閉。

斷開程式與AEDT之連結

當程式完成工作，可以用save_project()來儲存檔案。最後可以用release_desktop()斷開與AEDT連結，可以用close_desktop參數來關閉AEDT。

```
hfss.save_project()
hfss.release_desktop(close_desktop=False)
```

程式碼與AEDT連結模式

當使用 PyAEDT 來操作 AEDT (Ansys Electronic Desktop) 時，了解如何正確連接到不同的3D結構和設計是關鍵。PyAEDT 提供了多種方式來建立與 AEDT 的連接，以滿足不同的使用場景，如創建新設計、修改現有物件、批量處理模型，或在遠端電腦上進行模擬等。以下是一些主要的連接方法：

1. **連結全新檔案** ：

- 當 AEDT 未在運行時，使用 `Hfss()` 可以啟動 AEDT 並創建一個全新的專案與設計。例如：

```
hfss = Hfss()
```

這會自動開啟 AEDT 並創建一個新的專案。

2. **連結存在且未開啟的檔案** ：

- 當指定專案和設計名稱，如果這些文件已存在，PyAEDT 會開啟並連接到它們。例如：

```
hfss = Hfss(projectname='c:/xxx.aedt', designname='yyy')
```

這行代碼會尋找並開啟指定路徑和名稱的專案。

3. **連接已開啟的檔案** ：

- 如果 AEDT 已經在運行，且特定的專案和設計處於活動狀態，執行 `Hfss()` 時，會自動連接到該活動設計。例如：

```
hfss = Hfss()
```

若 AEDT 正在運行且有活動設計，這行代碼會連接到該設計。

4. **連接多個檔案** ：

- 通過指定 `designname` 參數，可以連接到特定的設計。這對於同時處理多個設計非常有用。

這些方法讓 PyAEDT 成為一個靈活且強大的工具，可以應對各種 AEDT 相關的自動化和腳本化任務。

7.6 連結AEDT Classical API

pyAEDT是對ANSYS AEDT的傳統API進行重新包裝，使之更符合Python的編程風格。這裏解釋一下為什麼這樣做對於使用者和開發者都是有好處的。

1. **Python風格的接口** ：原始的ANSYS AEDT API由於歷史原因，在函數命名和參數定義上可能不太符合Python的慣例。Python社區有一套廣為接受的編程標準（如PEP 8），這些標準使得代碼更易於理解和維護。pyAEDT通過將這些原始API重新包裝，使之符合Python的編程風格，從而提高了代碼的可讀性和可維護性。
2. **舊API函數的再利用** ：儘管pyAEDT提供了Python風格的接口，但它並沒有完全丟棄原始的ANSYS AEDT API。這意味著，如果有現有的基於舊API的代碼或函數，它們仍然可以在pyAEDT中被調用和利用，這樣就無需從頭開始重寫這些功能，節省了大量的開發時間和資源。
3. **代碼錄製與嵌入的靈活性** ：ANSYS AEDT的舊API支持錄製用戶的操作，將其轉化為可再次執行的代碼。在pyAEDT中，這些錄製的代碼可以被直接嵌入和利用，這為開發帶來了額外的彈性。開發者可以快速錄製一些操作過程，然後將這些過程整合到更大的Python項目中，這樣可以快速實現複雜功能，同時保持了代碼的整潔性和組織性。

總的來說，pyAEDT通過將傳統的ANSYS AEDT API轉化為更符合Python風格的形式，不僅提高了代碼的可讀性和可維護性，同時也保留了對舊API的兼容性，為開發者提供了更大的靈活性和效率。

> PEP 8，全稱是Python Enhancement Proposal 8，是一份Python社區廣泛認可和遵循的風格指南。它規定了如何格式化Python代碼，目的是提高代碼的可讀性和一致性。

以下這段代碼是一個例子，這是由AEDT的舊API構成的函數。這個特定的函數 createBoxArray 旨在創建一個由方塊組成的矩陣，其中方塊的位置和尺寸是根據輸入的參數來確定的，該函式是存放在classical.py檔案當中：

```
def createBoxArray(m, n, spacing=1):
    for i in range(m):
        for j in range(n):
            oEditor.CreateBox(
                [
                    "NAME:BoxParameters",
                    "XPosition:="     , "{}mm".format(i*spacing),
                    "YPosition:="     , "{}mm".format(j*spacing),
                    "ZPosition:="     , "0mm",
                    "XSize:="     , "0.3mm",
                    "YSize:="     , "0.3mm",
                    "ZSize:="     , "0.4mm"
                ],
                [
                    "NAME:Attributes",
                    "Name:="      , "Box1",
                    "Flags:="     , "",
                    "Color:="     , "(143 175 143)",
                    "Transparency:="   , 0,
                    "PartCoordinateSystem:=", "Global",
                    "UDMId:="     , "",
                    "MaterialValue:="  , "\"vacuum\"",
                    "SurfaceMaterialValue:=", "\"\"",
                    "SolveInside:="       , True,
                    "ShellElement:="   , False,
                    "ShellElementThickness:=", "0mm",
                    "IsMaterialEditable:=" , True,
                    "UseMaterialAppearance:=", False,
                    "IsLightweight:="  , False
                ])
```

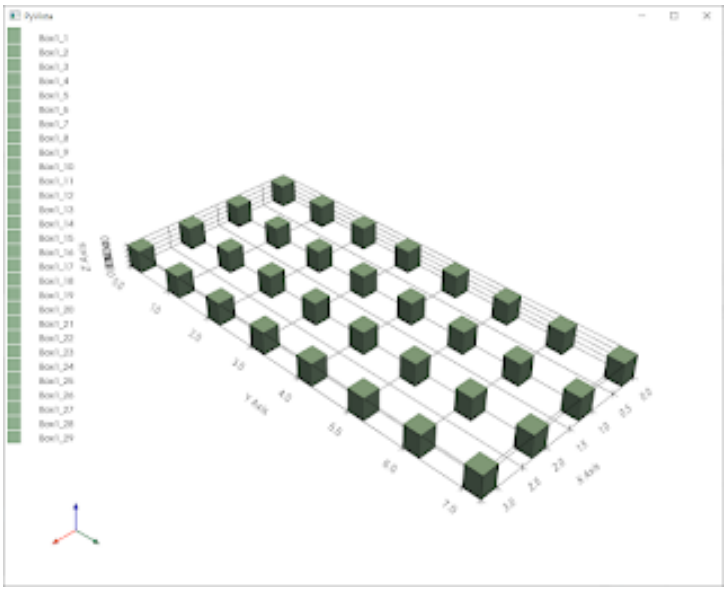
下面這段代碼展示了如何在pyAEDT中調用classical API：

```
from pyaedt import Hfss

hfss = Hfss(specified_version='2024.1', non_graphical=True)

import classical
classical.oEditor = hfss.modeler.oeditor
classical.createBoxArray(4, 8, 1)

hfss.plot()
```



輸出結果

以下解釋pyAEDT程式碼功能

1. **導入pyAEDT並創建HFSS對象** ：

- `from pyaedt import Hfss`：這行代碼導入了pyAEDT中的Hfss類。
- `hfss = Hfss(specified_version='2022.1', non_graphical=True)`：創建一個Hfss對象，指定了使用的ANSYS版本和非圖形化模式（適合腳本運行和服務器應用）。

2. **整合classical API** ：

- `import classical`：導入包含classical API函數的模塊。
- `classical.oEditor = hfss.modeler.oeditor`：這行將pyAEDT中的oeditor對象賦值給classical API的oEditor變量。這樣做可以讓classical API的函數在pyAEDT的環境中運行。

3. **使用classical API函數** ：

- `classical.createBoxArray(4, 8, 1)`：調用之前定義的`createBoxArray`函數，創建一個4x8的盒子矩陣，每個盒子之間的間距為1單位。

4. **可視化結果** ：

- `hfss.plot()`：這行代碼調用了pyAEDT中的繪圖功能，用於可視化建模結果。

> 因為使用non_graphical模式，AEDT GUI不須開啟便完成建模工作。因為沒有開啟GUI，pyAEDT提供額外介面用來檢視建出來的模型是否正確

這個例子很好地展示了pyAEDT如何使得使用ANSYS AEDT的classical API更加直觀和高效，尤其是在整合現有的基於classical API的代碼時。通過將classical API的函數嵌入到pyAEDT的環境中，可以在保持原有功能的同時，享受到Python語言和pyAEDT所提供的便利和靈活性。

7.7 PyAEDT的GUI設計

PyAEDT程式碼開發完成之後，可以使用 Python 的 GUI 框架來封裝 pyAEDT 的功能，從而為一般用戶提供一個圖形操作介面。這樣做的優點是使得軟件的使用變得更加直觀和易於操作，特別是對於那些不熟悉命令行或編程的用戶。這裡是筆者常用的 GUI 設計模組及其官方網頁連結：

Tkinter

Tkinter 是 Python 的標準 GUI (圖形用戶界面) 程式庫，它是基於 Tk GUI 工具集的 Python 界面。Tkinter 是 Python 中最常用的 GUI 框架之一，因為它簡單易用，並且通常作為 Python 標準庫的一部分預裝在大多數 Python 安裝中。這意味著對於絕大多數 Python 用戶來說，使用 Tkinter 不需要額外安裝任何東西。

Tkinter 的主要特點包括：

1. **易於學習和使用** ：對於初學者來說，Tkinter 是學習基礎 GUI 編程的理想選擇。
2. **跨平台** ：Tkinter 程序可以在 Windows、macOS 和 Linux 等主要操作系統上運行，無需修改代碼。
3. **輕量級** ：它不需要大量的系統資源，非常適合快速開發小型應用程序。
4. **可擴展性** ：雖然 Tkinter 提供的控件和功能有限，但它足以應對多數基本的 GUI 需求。此外，它也可以與其他庫結合，以擴展其功能。

Tkinter 通過提供各種標準的 GUI 控件，如按鈕、標籤、文本框、菜單、畫布等，使得創建窗口應用變得簡單。你可以通過組合這些控件來構建用戶界面。

由於 Tkinter 是 Python 的一部分，所以你可以直接從 Python 官方文檔中找到相關的資料和指南：Tkinter 官方文檔 。这里提供了豐富的參考資料，包括基本概念、可用控件、範例代碼等，非常適合初學者學習和參考。

PySide

PySide 是一個 Python GUI 框架，它提供了 Qt 框架的 Python 綁定。與 PyQt 類似，PySide 也是基於 Qt 框架的，但它由 The Qt Company 官方支持。PySide 被稱為 PySide2（用於 Qt 5）和 PySide6（用於 Qt 6）。

主要特點包括：

1. **跨平台** ：PySide 支持 Windows、macOS 和 Linux 等多個操作系統。
2. **豐富的控件** ：PySide 提供了豐富的控件和工具，可以創建專業和現代的 GUI 應用程序。
3. **強大的功能** ：包括支持 2D 和 3D 圖形、動畫、網絡功能和數據庫集成。
4. **開源和商業支持** ：PySide 遵循 LGPL（較為寬鬆）協議，對商業應用更友好。
5. **社區和官方支持** ：作為官方支持的 Qt 綁定，PySide 擁有穩定的發展路線和活躍的社區。

PySide 是開發跨平台桌面應用的強大選擇，特別適合需要豐富界面和高度互動性的應用程序。如果你對 PyQt 已經比較熟悉，那麼轉向使用 PySide 會很容易，因為它們有很多相似的 API。

PySide 的官方網站提供了詳細的文檔和教程，可以通過以下連結訪問：PySide 官方網站 。

Streamlit

Streamlit 是一款專為數據科學家和機器學習工程師設計的開源 Python 框架，它讓用戶能夠快速創建和分享美觀、互動式的 Web 應用程序。使用 Streamlit，開發者可以用少量的 Python 代碼，就能夠構建和部署數據驅動的 Web 應用程序。

Streamlit 的主要特點包括：

1. **易於使用** ：Streamlit 的 API 簡單直觀，即使是對網頁開發不熟悉的用戶也能快速上手。
2. **快速原型開發** ：它允許用戶寫少量代碼來創建原型，並實時查看更改，這使得迭代開發過程更加高效。
3. **豐富的控件和元件** ：Streamlit 提供了多種內置的控件，如滑動條、按鈕、圖表等，方便用戶構建互動式元件。
4. **數據科學集成** ：它可以輕鬆集成 Python 的數據科學生態系統，如 Pandas、NumPy、Matplotlib、Plotly 等。
5. **部署方便** ：Streamlit 應用可以輕鬆部署到各種平台，如 Heroku、AWS、Google Cloud Platform 等。

6. **開源社區** ：作為一個開源項目，Streamlit 有一個活躍的開發者和用戶社區，提供豐富的資源和支持。

Streamlit 特別適合於快速創建數據探索和機器學習原型的應用程序，並且對於希望將數據科學項目轉化為互動式 Web 應用的用戶來說，它是一個絕佳的選擇。

Streamlit 的官方網站提供了詳細的文檔和教程，你可以訪問以下連結來獲取更多信息：Streamlit 官方網站 。