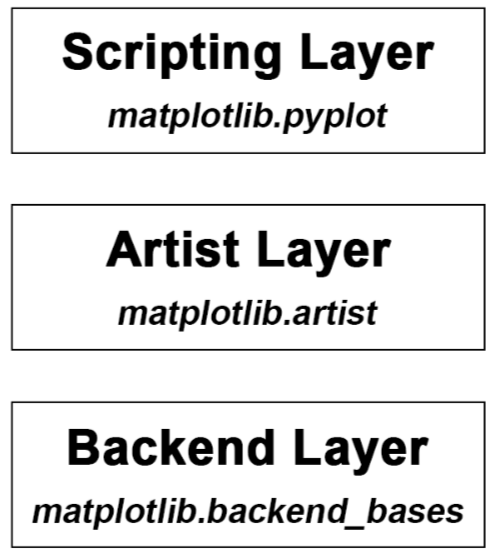
**Matplotlib**

**Project Purpose**

เป็น library ในการ plot กราฟของ python ที่เก่าแก่ที่สุด และยังคงเป็นที่นิยมในการใช้มากที่สุด ถูกสร้างขึ้นในปี ค.ศ. 2003 โดยเป็นส่วนหนึ่งของ SciPy Stack ซึ่งเป็น library เชิงวิทยาศาสตร์คอมพิวเตอร์แบบ open source ที่คล้ายกับ Matlab โดยมีการใช้ Numpy และ extension อื่น ๆ เพื่อทำให้มีประสิทธิภาพที่ดีขึ้น รวมไปถึงการทำงานที่ใช้ arrays ขนาดใหญ่

**Architectural Patterns/Styles**

โครงสร้างของ Matplotlib แบ่งออกเป็น 3 ชั้นหลัก คือ Backend Layer, Artist Layer, Scripting Layer

* Backend Layer: เป็นชั้นที่ซับซ้อนที่สุด เนื่องจาก layer นี้จะทำการสื่อสารไปยังชุดเครื่องมือวาดกราฟ โดยมีการ built-in 3 interface หลักคือ
  + FigureCanvas - [matplotlib.backend\_bases.FigureCanvasBase](https://matplotlib.org/api/backend_bases_api.html#matplotlib.backend_bases.FigureCanvasBase)
  + Renderer - [matplotlib.backend\_bases.RendererBase](https://matplotlib.org/api/backend_bases_api.html#matplotlib.backend_bases.RendererBase) เป็น class พื้นฐานที่ใช้จัดการกับการวาด/การเรนเดอร์ และรับผิดชอบการวาดภาพบน FigureCanvas
  + Event - [matplotlib.backend\_bases.Event](https://matplotlib.org/api/backend_bases_api.html#matplotlib.backend_bases.RendererBase) จัดการกับ input ของ user เช่น การคลิกแป้นพิมพ์และเมาส์

โดยทั่วไปแล้วผู้ใช้แทบจะไม่ยุ่ง หรือจัดการกับ layer นี้เลย

* Artist Layer: เป็น layer ที่อนุญาติให้ควบคุม และปรับแต่งองค์ประกอบได้ โดย layer นี้จะใช้ renderer เพื่อวาดภาพบน FigureCanvas จะทำให้ปรับแต่ง และใช้งานได้ง่ายกว่าเมื่อเทียบกับ Scripting layer และสะดวกต่อการใช้งานมากกว่าสำหรับ advanced plots
* Scripting Layer: เป็น layer ที่อยู่ชั้นบนสุด ที่ออกแบบมาเพื่อให้ Matplotlib สามารถทำงานได้เหมือน MATLAB และเป็นชุดของฟังก์ชันรูปแบบคำสั่งและถือเป็นเลเยอร์ที่ง่ายที่สุดในการใช้งาน

**Quality Attribute Scenario**

1. **Usability**

|  |  |
| --- | --- |
| **Source of stimulus** | End user |
| **Stimulus** | ต้องการวาดกราฟโดยใช้ matplotlib |
| **Artifacts** | matplotlib |
| **Environment** | Runtime |
| **Response** | วาดกราฟสำเร็จ |
| **Response measure** | เวลาที่ใช้ในการวาดสำเร็จ |

1. **Modifiability**

|  |  |
| --- | --- |
| **Source of stimulus** | End user |
| **Stimulus** | ต้องการเพิ่ม/ลบ/แก้ไข/เปลี่ยนฟังก์ชัน |
| **Artifacts** | UI, platform (windows, Unix) |
| **Environment** | Compile time |
| **Response** | เปลี่ยนโดยไม่มีผลกระทบกับการทำงานอื่น ๆ |
| **Response measure** | เวลาที่ใช้ |

1. **Performance**

|  |  |
| --- | --- |
| **Source of stimulus** | Create new event |
| **Stimulus** | เป็นช่วง ๆ ในการทำงาน |
| **Artifacts** | system |
| **Environment** | Overload mode |
| **Response** | Process |
| **Response measure** | ปริมาณ/จำนวนฟังก์ชันที่สามารถผ่านเข้าระบบได้ |

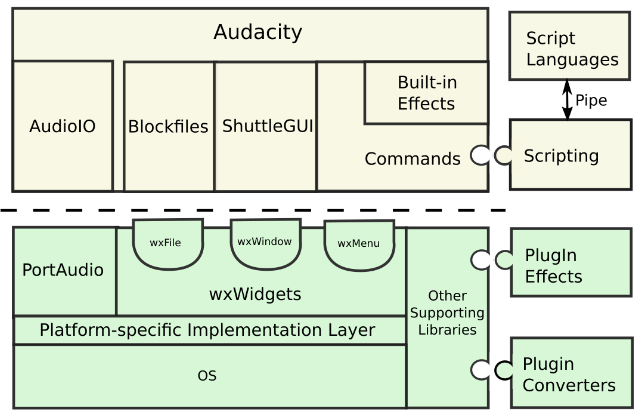
**ที่มา**

* <https://medium.datadriveninvestor.com/data-visualization-with-python-matplotlib-architecture-6b05af533569>
* <https://en.wikipedia.org/wiki/Matplotlib>
* <https://www.aosabook.org/en/matplotlib.html>

**Audacity**

**Project Purpose**

Audacity เป็นโปรแกรม Open-source ที่ใช้ตัดต่อและบันทึกเสียง และสามารถใช้Effect ต่าง ๆ ได้เช่น Noise Reduction (ลดเสียง) Amplify (เพิ่มเสียง) และอื่น ๆ โดยมี user interface 8 คือ wxWidgets ซึ่งสามารถใช้งานได้ในหลากหลาย platform ไม่ว่าจะเป็น Windows MacOS Linux หรือระบบปฏิบัติการอื่น ๆ

**Architectural Patterns/Styles**

โดยโครงสร้างของ audacity จะเห็นได้จากภาพด้านบนว่า แบ่งออกเป็น 2 layers ซึ่งจะเน้นไปที่การใช้ library ในการช่วยการทำงานเป็นจำนวนมาก และเนื่องจาก feature ใน Audacity ส่วนมากจะมาจาก library สะส่วนใหญ่ ส่วน main code มีจำนวนเพียงเล็กน้อยเท่านั้นที่จำเป็นในการใช้งาน

1. lower-level layer จะแบ่งออกเป็น 2 libraries หลักคือ
   1. PortAudio ที่ทำให้ low-level audio interface สำหรับการ cross-platform
   2. wxWidgets ที่ ให้ GUI component สำหรับการข้าม platform

โดยที่ PortAudio และ wxWidgets เป็น OS abstraction layer และมีโค้ดเงื่อนไขที่จะใช้ขึ้นอยู่กับแต่ละ platform

1. higher-level layer จะทำงานโดยมีความสัมพันธ์ที่เกี่ยวข้องกับ lower-level layer ที่ตรงกัน เช่น AudioIO จัดการย้ายข้อมูลเสียงระหว่างการ์ดเสียง หน่วยความจำ และ ฮาร์ดดิสก์ใน PortAudio และ BlockFile ใช้ระบบไฟล์ OS ผ่าน wxFile ของ wxWidgets

**Quality Attribute Scenario**

1. **Usability**

|  |  |
| --- | --- |
| **Source of stimulus** | End user |
| **Stimulus** | การใช้งานอย่างมีประสิทธิภาพ |
| **Artifacts** | System |
| **Environment** | Runtime |
| **Response** | ให้ผู้ใช้ สามารถใช้งานได้อย่างมีประสิทธิภาพ |
| **Response measure** | ระยะเวลา |

1. **Modifiability**

|  |  |
| --- | --- |
| **Source of stimulus** | Developer |
| **Stimulus** | ต้องการเพิ่ม/ลบ/แก้ไข/เปลี่ยน code ในการทำงาน |
| **Artifacts** | Code |
| **Environment** | Design time |
| **Response** | เพิ่ม/ลบ/แก้ไข/เปลี่ยน code โดยไม่มีผลกระทบกับการทำงานอื่น ๆ |
| **Response measure** | เวลาที่ใช้ในการแก้ไข |

1. **Testability**

|  |  |
| --- | --- |
| **Source of stimulus** | System |
| **Stimulus** | สถาปัตยกรรม |
| **Artifacts** | Code |
| **Environment** | ช่วงการ compile |
| **Response** | ค่าการทำงานที่ถูกต้อง |
| **Response measure** | การวัด % ของ statement ที่ทำสำเร็จ |

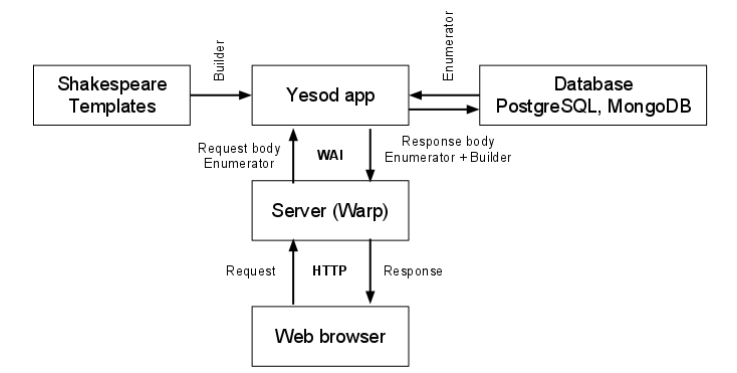
**ที่มา :**

* <https://wiki.audacityteam.org/wiki/ArchitecturalDesign>
* https://www.aosabook.org/en/audacity.html

**Yesod**

**Project Purpose**

Yesod จัดเป็น Model-View-Controller (MVC) และยังเป็น Web framework ที่เขียนด้วยภาษาโปรแกรมมิ่งที่มีชื่อว่า Haskell ในขณะที่ Web framework อื่น ๆ จำนวนมาก ใช้ประโยชน์ความเป็น dynamic ของภาษานั้น ๆ ที่ใช้ใน Web Framework แต่ Yesod ใช้ประโยชน์ความเป็น static ของ Haskell language เพื่อสร้าง produce ที่ ปลอดภัย กระชับ และเร็วกว่า

**Architectural Patterns/Styles**

จากภาพจะเห็นได้ว่า web application ต้องการเส้นทางที่ตนจะสามารถติดต่อสื่อสารกับ server ได้ แต่จริง ๆ แล้วสามารถนำ server ไปใส่รวมกับ frame work ได้โดยตรง แต่วิธีนี้จะทำให้การทำงานมีตัวเลือกที่น้อยลง และทำให้ interface แย่อีกด้วย ซึ่งในหลาย ๆ ภาษาได้สร้าง built-in interface เพื่อนำมาแก้ไขปัญหานี้ โดย python มี  WSGI และ Ruby มี Rack ส่วนในภาษา Haskell เรามี WAI: Web Application Interface

โดย WAI มีข้อดีหลัก ๆ อยู่ 2 ข้อ คือ generality, performance ทำให้ภาษา WAI สามารถช่วยเหลือ backend ของทุก ๆ อย่างได้ตั้งแต่ standalone server , CGI รวมไปถึงการทำงานโดยตรงกับ Webkit เพื่อสร้าง desktop application

**Quality Attribute Scenario**

1. **Usability**

|  |  |
| --- | --- |
| **Source of stimulus** | End user |
| **Stimulus** | ต้องการใช้งานในการเขียน web |
| **Artifacts** | Yesod |
| **Environment** | Runtime |
| **Response** | ให้ผู้ใช้ สามารถใช้งานได้อย่างมีประสิทธิภาพ |
| **Response measure** | ระยะเวลาที่ใช้ในการเขียน |

1. **Modifiability**

|  |  |
| --- | --- |
| **Source of stimulus** | End user |
| **Stimulus** | ต้องการแก้ไข code ที่ใช้ในการเขียน web |
| **Artifacts** | Code |
| **Environment** | Design time |
| **Response** | แก้ไข code สำเร็จ |
| **Response measure** | ระยะเวลาที่ใช้ในการแก้ไข |

1. **Testability**

|  |  |
| --- | --- |
| **Source of stimulus** | System |
| **Stimulus** | Unit test |
| **Artifacts** | Component |
| **Environment** | Compile time |
| **Response** | ผลการ test |
| **Response measure** | % ของข้อผิดพลาดที่เกิดขึ้น |

**ที่มา :**

* <https://www.yesodweb.com/>
* <https://www.aosabook.org/en/yesod.html>