Matplotlib

Matplotlib ถูกสร้างขึ้นมาเพื่อใช้พลอตกราฟโดยที่ด้วยคำสั่งไม่กี่คำสั่ง โดยช่วงเริ่มต้นจุดมุ่งหมายของ matplotlib คือ

- กราฟที่ได้จากการพลอตต้องใช้ในการตีพิมพ์ได้ โดยมีเฉพาะตัวหนังสือต้องมีความชัดเจน โดยใช้เทคนิคการลบรอยหยัก, การพลิก, ฯลฯ
- สามารถใช้งานร่วมกับไฟล์ชนิด TEX
- มี GUI สำหรับการพัฒนาเพิ่มเติม
- โค้ดอ่านเข้าใจง่าย
- เป็น Open-Source สามารถโหลด ใช้งาน แจกจ่ายได้ฟรี

โครงสร้างของ matplotlib ประกอบด้วย 3 ส่วนคือ

- Interface สำหรับผู้ใช้งานใช้พลอตผ่าน Command Line
- Front-End หรือ matplotlib API ที่เป็นกลุ่มของ class ที่ใช้สร้างและจัดการ figure, ตัวหนังสือ, เส้น, กราฟ
- Back-End ขึ้นอยู่กับแต่ละเครื่องที่ใช้ โดยใช้เรนเดอร์ผลลัพท์ที่ได้จาก Front-End มาเป็นไฟล์ชนิดต่างๆ หรือแสดงผล

ฟีเจอร์ที่ทำให้ matplotlib ใช้งานง่าย

- ใช้งานร่วมกับ library อื่นใน python ได้ เช่น pandas, numpy
- Plot window ใช้งานง่ายมีระบบซูมเข้าออก เซฟได้
- Command Line ใช้งานง่ายตาม MatLab
- พลอตกราฟได้หลายกราฟและรูปได้หลายรูปในหน้าเดียว
- ใช้งานฟ้อนท์ Truetype/Freetype ซึ่งทำให้อ่านได้ง่ายแม้มีการเพิ่มลดขนาด
- ใช้งาน TEX math mode เมื่อมีการใช้ Truetype
- รูปเปลี่ยนขนาดตามขนาดของ figure
- ใช้งาน Object-Oriented Design

Design Architecture ที่ใช้

Layer

โดยแบ่งเป็น 3 ลำกับ Interface Front-End และ Back-End

Interface จะทำการติดต่อกับผู้ใช้ผ่าน Command Line Interface แล้วส่งข้อมูลไปหา Front-End

Front-End จะทำการจัดการกับ figure, ตัวหนังสือ, กราฟ แล้วส่งให้ Back-End

Back-End จะทำการเรนเดอร์ข้อมูลที่รับมาจาก Front-End ออกมาให้เป็นรูปภาพ

Quality Attribute Scenarios

Integrability จากการใช้งานร่วมกับไฟล์ชนิด TEX, การเซฟรูป figure เป็นไฟล์รูปชนิดมาตรฐาน

- Source of Stimulus : Component marketplace/vendor
- Stimulus : อัพเดทเวอร์ชั่นใหม่ของส่วนประกอบ
- Artifacts : บางส่วนของส่วนประกอบ
- Environment : Development
- Responds : ส่วนที่เกิดการเปลี่ยนแปลงทำงานร่วมกับระบบได้
- Respond Measures : ลดจำนวนการเปลี่ยนแปลง เพิ่ม ลด โค้ดเนื่อง

Modifiability จากการสร้าง GUI สำหรับพัฒนาเพิ่มเติม, การเป็น Open-Source

- Source of Stimulus : คนสร้าง, ผู้ใช้ที่มีส่วนร่วมใน Open Source
- Stimulus : ต้องการที่จะเพิ่ม ลบ หรือแก้ไข
- Artifacts : โค้ด ข้อมูล UI ฯลฯ
- Environment : Runtime, compile time, build time, initiation time, design time
- Responds : เพิ่ม ลบหรือแก้ไข, ทดสอบการผลจากการเปลี่ยนแปลง, Deploy
- Respond Measures : ขนาดหรือความซับซ้อนของแอพพลิเคชั่น, เงิน(เพราะเป็น Open-Source), ความอุตสาหะของ Dev

Useability จากโค้ดที่เข้าใจง่าย, ใช้ Command Line

- Source of Stimulus : ผู้ใช้งาน
- Stimulus : ต้องการใช้ระบบอย่างมีประสิทธิภาพ
- Artifacts : Command Line Interface
- Environment : Runtime, System Configuration Time
- Responds : ให้โค้ดที่ผู้ใช้สามารถเขียนและอ่านได้เข้าใจง่าย
- Respond Measures : ความพึงพอใจของผู้ใช้, เวลาในการทำงาน

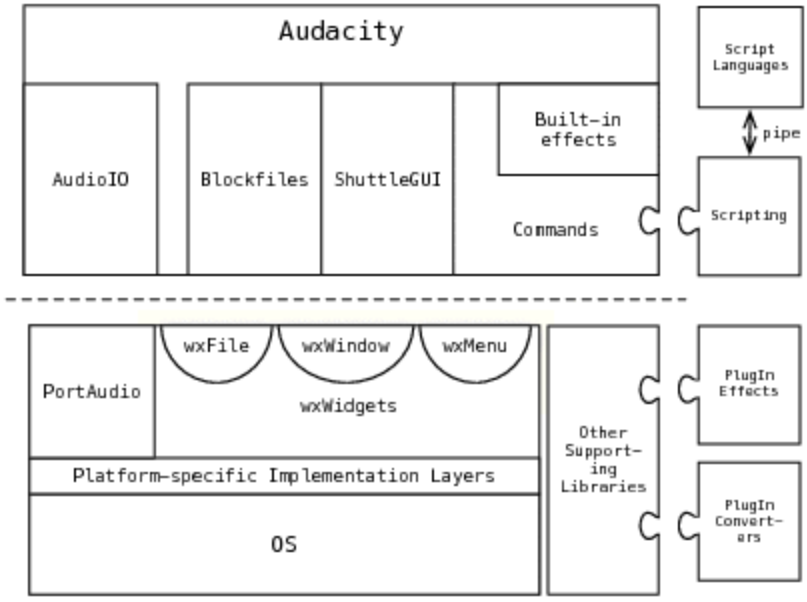
ที่มา : [(PDF) matplotlib -- A Portable Python Plotting Package (researchgate.net)]

Audacity

Audacity เป็นแอพลิเคชั่นอัดเสียงและตัดต่อเสียงซึ่งมีการใช้งานหลากหลายทั้งใน Windows และ MacOS ผู้สร้างแอพพลิเคชั่นตั้งใจที่จะสร้างแพลตฟอร์มที่ใช้พัฒนาและดีบัคอัลกอริทึ่มการประมวลผลเสียง หลังจาก Audacity ได้เปิดเป็น Open-Source ทำให้ผู้พัฒนาได้เข้ามาร่วมพัฒนาและแอพพลิเคชั่นก็ได้เพิ่มระบบใหม่เข้ามามากมายทั้ง ปรับปรุง บำรุงรักษา ทดสอบ อัพเดท รวมถึงทำคู่มือให้ผู้ใช้ และแปลเป็นภาษาอื่นๆ ผู้พัฒนาพยายามต่อยอดด้วยความสม่ำเสมอในด้านของลักษณะการเขียนโค้ด ด้วยการคำนึงถึงโค้ดที่อยู่ในที่ใกล้เคียง

จุดมุ่งหมายของ Audacity คือการทำให้ User Interface ใช้งานง่าย ผู้ใช้งานสามารถใช้งานได้เลยโดยที่ไม่จำเป็นต้องอ่านคู่มือ

โครงสร้างของ Audacity



Audacity มีพื้นฐานจากการใช้ Library จำนวนมาก โดย Library ที่สำคัญคือ PortAudio ที่ให้ Low-level Audio Interface และ wxWidgets ที่ให้ GUI Components

Library อื่นๆที่ใช้ต่อยอดจาก wxWidgets และ PortAudio

- BlockFile ใช้ OS file system ผ่าน wxWidgets เพื่อให้วิธีการเก็บเสียงเป็นกลุ่มเล็กๆ ซึ่งทำให้สามารถตัดต่อปรับแต่งเสียงโดยไม่จำเป็นต้องปรับแต่งทั้งไฟล์
- ShuttleGUI ใช้ wxWidgets ในการจัดการ dialog, ปุ่ม และการควบคุมอื่นๆ เพื่อการเขียนโค้ดซ้ำ ผ่านการเก็บข้อมูลเป็นตัวแปล
- Command จัดการ Bind ปุ่มในคีย์บอร์ด ผ่าน wxWidgets
- AudioIO จัดการการเคลื่อนย้ายเสียงระหว่าง Sound card, memory, hard disk ผ่าน PortAudio

Design Architecture ที่ใช้

Service-Oriented Architecture

จากโครงสร้างตามรูปด้านบนมีการใช้งาน API หลายชนิดที่มีความสัมพันธ์กัน เช่นการใช้งาน BlockFile ผ่าน wxWidget หรือการใช้ AudioIO ผ่าน PortAudio

Quality Attribute Scenarios

Useability จากจุดมุ่งหมายที่ต้องการให้แอพพลิเคชั่นใช้งานง่าย

- Source of Stimulus : ผู้ใช้งาน
- Stimulus : ต้องการเรียนรู้การใช้ระบบ
- Artifacts : GUI
- Environment : Runtime, System Configuration Time
- Responds : ให้การใช้งานที่ง่ายโดยไม่จำเป็นต้องพึ่งคู่มือ
- Respond Measures : ความรู้ของผู้ใช้งาน

Modifiability การเป็น Open-Source, การพยายามเขียนโค้ดในลักษณะเดียวกัน

- Source of Stimulus : คนสร้าง, ผู้ใช้ที่มีส่วนร่วมใน Open Source
- Stimulus : ต้องการที่จะเพิ่ม ลบ หรือแก้ไข
- Artifacts : โค้ด ข้อมูล UI ฯลฯ
- Environment : Runtime, compile time, build time, initiation time, design time
- Responds : เพิ่ม ลบหรือแก้ไข, ทดสอบการผลจากการเปลี่ยนแปลง, Deploy
- Respond Measures : เงิน(เพราะเป็น Open-Source), ความอุตสาหะของDev, เวลาพัฒนาในส่วนนั้นๆลดลง

Performance จากการใช้ BlockFile ให้ไม่จำเป็นต้องแก้ไขไฟล์ทั้งไฟล์

- Source of Stimulus : ผู้ใช้งาน
- Stimulus : ต้องการตัดต่อเสียง
- Artifacts : บางส่วนของระบบที่เกี่ยวกับการตัดต่อและจัดการไฟล์เสียง
- Environment : การทำงานแบบปกติของแอพพลิเคชั่น
- Responds : การใช้งานทรัพยากรของระบบ
- Respond Measures : ใช้เวลาลดลง, ใช้ทรัพยากรลดลง

ที่มา : [The Architecture of Open Source Applications: Audacity (aosabook.org)]

Yesod

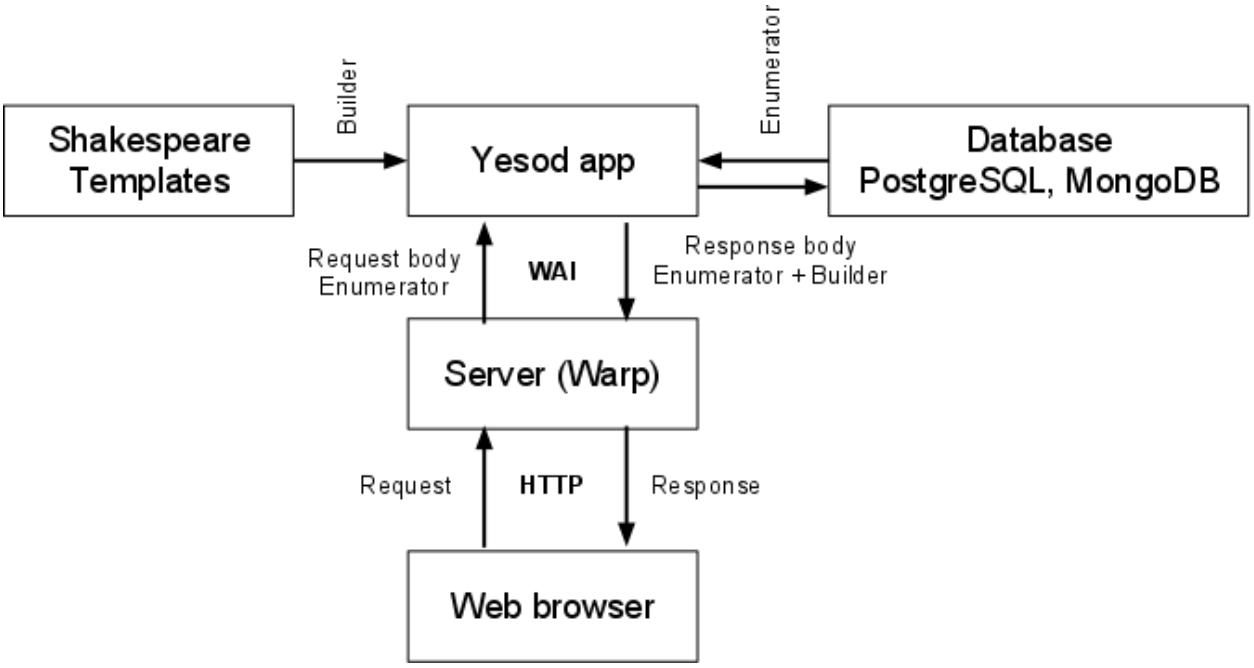
Yesod เป็น web framework ที่เขียนขึ้นในภาษา Haskell ซึ่งเป็นภาษาที่คงที่ (Static) จึงทำให้ Yesod ทำงานได้รวดเร็วกว่าและปลอดภัยกว่า

เป้าหมายของ Yesod คือการใช้ข้อได้เปรียบจากภาษา Haskell ในการทำพัฒนา web Yesod ตั้งใจที่จะทำให้โค้ดมีความรวบรัดมากที่สุดเท่าที่ทำได้ ทุกบรรทัดจะถูกเช็คความถูกต้องตอนที่ compile แทนที่จะใช้ library ภายนอกเพื่อตรวจสอบ Yesod จะใช้ Compiler ตรวจสอบแทน

โครงสร้างของ Yesod

Yesod ใช้โครงสร้างสถาปัตยกรรมตามรูปแบบ Model-View-Controller (MVC) ทำให้มีแม่แบบของระบบของส่วน View ที่แยกออกมาจากส่วนตรรกะ ให้ระบบ Object-Relational Mapping (ORM) และมี Controller ที่ใช้ในการ routing

การเชื่อมต่อกับ server จะใช้ Web Application Interface (WAI) ซึ่งมีจุดมุ่งหมายในความเป็นส่วนใหญ่ โดย WAI จะทำการสนับสนุน Back-End ในทุกๆด้าน และ ประสิทธิภาพที่เกี่ยวข้องกับการใช้งานฟีเจอร์ของภาษา Haskell



รูปของการทำงานแบบ Web Application Interface

Design Architecture ที่ใช้

Model View Controller (MVC)

โดยให้ Model = Database, View = Shakespeare Templates, Yesod app = Controller

Quality Attribute Scenarios

Security จากการใช้ภาษา Haskell ซึ่งเป็นภาษาที่มีช่องโหว่น้อยกว่า

- Source of Stimulus : ผู้ไม่ประสงค์ดีที่ต้องการโจมตี
- Stimulus : การแสดงผลข้อมูลที่ไม่พึงประสงค์จากการโจมตี
- Artifacts : ข้อมูลภายในระบบ
- Environment : ระบบ
- Responds : ข้อมูลถูกป้องกันไม่ให้แสดงผลข้อมูลที่ไม่พึงประสงค์
- Respond Measures : จำนวนการโจมตีที่ป้องกันได้, ข้อมูลใดที่ถูกป้องกันจากการโจมตี

Performance จากการใช้ภาษา Haskell ซึ่งเป็นภาษาชนิดคงที่ทำให้มีประสิทธิภาพที่สูงกว่า

- Source of Stimulus : ส่วนประกอบต้องการใช้งานอีกส่วน
- Stimulus : เหตุการณ์ที่จำเป็นต้องใช้งานส่วนประกอบอีกส่วน
- Artifacts : บางส่วนของระบบที่เกี่ยวข้อง
- Environment : การทำงานแบบปกติของแอพพลิเคชั่น
- Responds : การใช้งานทรัพยากรของระบบ
- Respond Measures : ใช้เวลาลดลง, ใช้ทรัพยากรลดลง

Testability จากการใช้ Compiler ในการตรวจเช็คความถูกต้อง

- Source of Stimulus : System Tester
- Stimulus : ทดสอบฟังก์ชั่น, ทดสอบคุณภาพ, ทดสอบความอันตราย
- Artifacts : ระบบ
- Environment : ทั้งระบบ
- Responds : ผลการตรวจสอบจาก compiler
- Respond Measures : ความอุตสาหะในการทดสอบระบบ

ที่มา : [The Architecture of Open Source Applications (Volume 2): Yesod (aosabook.org)]