**YESOD**

Yesod เป็นเว็บเฟรมเวิร์คถูกเขียนด้วยภาษาฮาสเกล ( Haskell programming language ) ในขณะที่เว็บเฟรมเวิร์คที่โด่งดังใช้ประโยชน์จากลักษณะแบบไดนามิกของภาษาโฮสต์ yesod ใช้ประโยน์จากลักษณะที่เป็นรูปแบบคงที่ ( static ) ของภาษาฮาสเกลเพื่อให้การเขียนเป็นรูปแบบที่ปลอดภัยและเขียนโค้ดได้รวดเร็ว

การพัฒนาได้เริ่มต้นขึ้นมาตั้งแต่ปี ค.ศ. 2010 และก็ได้พัฒนามาเรื่อย ๆ yesod ได้ฟันฝ่าในโครงการชีวิตจริงด้วยกับฟีเจอร์ที่เริ่มต้นทั้งหมดที่เกิดจากของจริงความต้องการในชีวิตจริง แรกเริ่มการพัฒนา yesod เป็นในรูปแบบทำคนเดียวทั้งหมดหลังจากนั้นประหนึ่งปีให้หลังก็ได้มีกลุ่มคนคนได้เข้ามามีส่วนร่วมทำให้ yesod ได้เบ่งบานในโครงการ thriving open source project

ทำไมต้องใช้ภาษาฮาสเกล? ดูเหมือนว่าในโลกนี้จะมีรูปแบบสองแบบด้วยกัน

* Statically typed language เช่นภาษาจาวา ภาษาซีชาร์ป ภาษาซีพลัส ๆ ภาษาเหล่านี้ให้ความเร็วและความปลอยภัยของ type แต่ยากต่อการเขียนโค้ด
* Dynamically typed language เช่นภาษารูบี้ ภาษาไพธอน ภาษาเหล่านี้เพิ่มผลผลิตอย่างมากอย่างแต่รันได้ช้าและคอมไพเลอร์ไม่ค่อยสนับสนุนในการเช็คความถูกต้อง

คำตอบคือนี้เป็นแบ่งขั้วที่ผิด ไม่มีเหตุผลใดที่ว่าทำไมภาษาที่เป็น static typed ต้องเป็นภาษาที่งุ่มง่ามเชื่องช้า ฮาสเกลสามารถที่จะตรวจจับการแสดงออกจำนวนมากของ Ruby และ Python ในขณะที่ยังคงที่เป็น strongly typed language ในความเป็นจริงระบบของ Haskell type ตรวจจับบั๊คมากกว่าภาษาจาวาเช่น Null pointer exception ถูกกำจัดไว้เรียบร้อยแล้ว โครงสร้างข้อมูลที่ไม่เปลี่ยนรูปทำให้การให้เหตุผลเกี่ยวกับโค้ดของคุณง่ายขึ้น ลดความซับซ้อนของการเขียนโปรแกรมแบบขนานและพร้อมกัน

ทำไมต้องใช้ภาษาฮาสเกล? ฮาสเกลที่ภาษาที่มีประสิทธิภาพและภาษาที่เป็นมิตรกับนักพัฒนาซึ่งให้การตรวจสอบความถูกต้องของโปรแกรมคอมไพล์เวลารวบรวมมากมาย

**วัตถุประสงค์**

เป้าหมายของ Yesod คือการขยายจุดแข็งของ Haskell ไปสู่การพัฒนาเว็บ Yesod พยายามทำให้โค้ดของคุณกระชับที่สุด ทุกบรรทัดของโค้ดของคุณจะถูกตรวจสอบความถูกต้อง ณ เวลาคอมไพล์ให้มากที่สุดเท่าที่เป็นไปได้ แทนที่จะต้องใช้ไลบรารีขนาดใหญ่ของการทดสอบหน่วย ( unit testing ) เพื่อทดสอบคุณสมบัติพื้นฐาน คอมไพเลอร์ทำทุกอย่างเพื่อคุณ โดยเบื่องหลัง Yesod ใช้เทคนิคประสิทธิภาพขั้นสูงมากเท่าที่เราสามารถรวบรวมเพื่อให้โค้ดระดับสูงของคุณทำงาน

**Architectural Patterns / Styles**

โดยที่สถาปัตยกรรมซอฟต์แวร์ที่ yesod ใช้เป็นรูปแบบ Model-View-Controller ( MVC ) หนึ่งในเป้าหมายของ MVC คือแยกส่วนของ logic ออกจากส่วนติดต่อผู้ใช้ view เพื่อที่จะทำการแยกได้โดยใช้ภาษาเทมเพลต ( template language ) อย่างไรก็ตาม มีหลายวิธีในการแก้ไขปัญหานี้ ที่ปลายด้านหนึ่งของสเปกตรัม ตัวอย่างเช่น PHP/ASP/JSP จะอนุญาตให้คุณฝังโค้ดใดก็ได้ภายในเทมเพลตของคุณ อีกด้านหนึ่ง คุณมีระบบ เช่น StringTemplate และ QuickSilver ซึ่งผ่าน argument และไม่มีการโต้ตอบกับส่วนที่เหลือของโปรแกรม

แต่ละระบบมีข้อดีและข้อเสีย การมีระบบเทมเพลตที่มีประสิทธิภาพยิ่งขึ้นสามารถอำนวยความสะดวกได้อย่างมาก หากต้องการแสดงเนื้อหาของตารางฐานข้อมูลหรือไม่? ไม่มีปัญหา ดึงเข้าไปด้วยเทมเพลต อย่างไรก็ตาม วิธีการดังกล่าวสามารถนำไปสู่โค้ดที่ซับซ้อนได้อย่างรวดเร็ว การอัพเดตเคอร์เซอร์ฐานข้อมูลแบบสลับกันด้วยการสร้าง HTML สิ่งนี้สามารถเห็นได้ทั่วไปในโครงการ ASP ที่เขียนไม่ดี

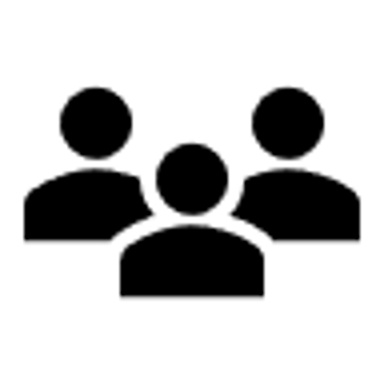
แม้ว่าระบบเทมเพลตที่อ่อนแอจะสร้างโค้ดง่ายๆ แต่ก็มีแนวโน้มที่จะทำงานซ้ำซาก คุณมักจะต้องไม่เพียงแค่เก็บค่าดั้งเดิมไว้ในประเภทข้อมูลเท่านั้น แต่ยังต้องสร้าง dictionary ของค่าเพื่อส่งต่อไปยังเทมเพลตด้วย การดูแลรักษาโค้ดดังกล่าวไม่ใช่เรื่องง่าย และโดยปกติแล้วจะไม่มีทางที่คอมไพเลอร์จะช่วยคุณได้

ภาษาเทมเพลตของตระกูล Yesod ภาษาเช็คสเปียร์ ( Shakespear ) มุ่งมั่นเพื่อความเป็นกลางโดยใช้ประโยชน์จากความโปร่งใสในการอ้างอิงมาตรฐานของ Haskell เราสามารถมั่นใจได้ว่าเทมเพลตของเราไม่มีผลข้างเคียง อย่างไรก็ตาม พวกเขายังคงสามารถเข้าถึงตัวแปรและฟังก์ชันทั้งหมดที่มีอยู่ในโค้ด Haskell ของคุณได้อย่างเต็มที่ นอกจากนี้ เนื่องจากได้รับการตรวจสอบอย่างครบถ้วนสำหรับทั้งรูปร่างที่ดี ความละเอียดที่เปลี่ยนแปลงได้ และความปลอดภัยในการพิมพ์ ณ เวลา compile time การพิมพ์ผิดจึงมีโอกาสน้อยมากที่คุณจะต้องค้นหาโค้ดของคุณที่พยายามจะปักหมุดจุดบกพร่อง

**รูปภาประกอบ MVC**

* View แสดงผลค่าในโมเดลในรูปแบบที่เหมาะสมต่อการปฏิสัมพันธ์กับผู้ใช้ ในแต่ละโมเดลสามารถมีวิวได้หลายแบบ เพื่อใช้ในจุดประสงค์ที่ต่างกันโดยใน yesod ใช้ Shakespeare
* Controller รับข้อมูลจากผู้ใช้เข้ามา แล้วดำเนินการตอบสนองต่อข้อมูลนั้น โดยเรียกใช้ logic ต่างๆจากอ็อบเจกต์ในโมเดล และส่งข้อมูลผลลัพธ์นั้นกลับไปยังส่วนแสดงผล เพื่อตอบกลับไปยังผู้ใช้ได้อย่างถูกต้อง
* Model หมายถึง ส่วนของซอฟต์แวร์ที่ใช้ในการแปลการทำงานของระบบ ไปสู่สิ่งที่ระบบซอฟต์แวร์ได้ถูกออกแบบเอาไว้ ตรรกะเนื้อหาใช้เพื่อให้ความหมายแก่ข้อมูลดิบ (ยกตัวอย่างเช่น การคำนวณว่าวันนี้เป็นวันเกิดของผู้ใช้หรือไม่, หรือจำนวนเงินรวม ภาษี และค่าส่งสินค้า ในตะกร้าสินค้า) เมื่อโมเดลมีการเปลี่ยนแปลง จะมีการส่งคำเตือนให้แก่ วิว ที่เกี่ยวข้องเพื่อปรับค่าระบบซอฟต์แวร์หลายระบบใช้การเก็บข้อมูลถาวร เช่น ฐานข้อมูล เพื่อเก็บข้อมูลเหล่านี้ MVC ไม่ได้กำหนดถึงระดับการเข้าถึงข้อมูล เพราะเป็นที่เข้าใจกันว่าส่วนนี้จะอยู่ภายใต้ หรือถูกครอบคลุมด้วยโมเดล โมเดลไม่ได้เป็นเพียงอ็อบเจกต์ที่ใช้เข้าถึงข้อมูล แต่ในระบบซอฟต์แวร์เล็กๆ ซึ่งมีความซับซ้อนน้อยจะไม่เห็นความแตกต่างมากนักโดยใน yesod ใช้ Persistent เป็น model

**Quality Attribute Scenarios**



Source:

Component

marketplace

Stimulus:

New data filtering component become available

Artifact:

System

Environment:

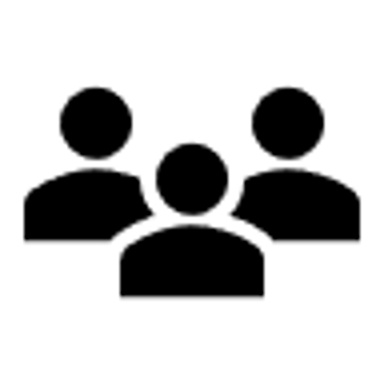
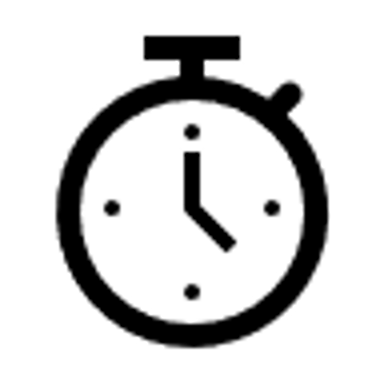
Development

Response:

The new component is integrated and deployed

Response measure:

1 month



Source:

Component

marketplace

Stimulus:

New data filtering component become available

Artifact:

System

Environment:

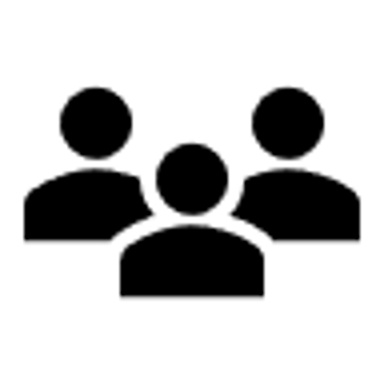
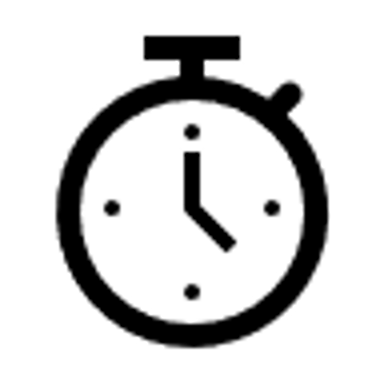
Development

Response:

The new component is integrated and deployed

Response measure:

1 month



Source:

Component

marketplace

Stimulus:

New data filtering component become available

Artifact:

System

Environment:

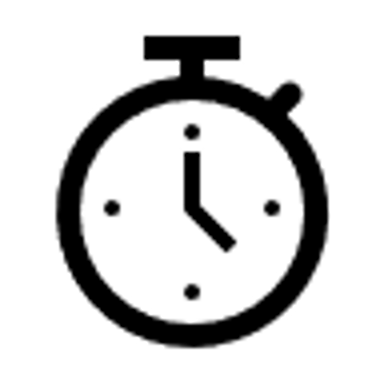
Development

Response:

The new component is integrated and deployed

Response measure:

1 month



**แหล่งอ้างอิง**

* <https://www.google.com/search?q=mvc&hl=en&source=lnms&tbm=isch&sa=X&ved=2ahUKEwjPs76q8ov6AhUUiOYKHZW9C-0Q_AUoAXoECAIQAw&biw=1920&bih=947&dpr=1#imgrc=Y9TvT6_QwerjoM>
* <https://en.wikipedia.org/wiki/Yesod_%28web_framework%29#MVC_architecture>
* <https://stackoverflow.com/questions/23239236/yesod-architecture>
* <https://www.aosabook.org/en/yesod.html>
* <https://www.yesodweb.com/>
* <https://th.wikipedia.org/wiki/%E0%B9%82%E0%B8%A1%E0%B9%80%E0%B8%94%E0%B8%A5-%E0%B8%A7%E0%B8%B4%E0%B8%A7-%E0%B8%84%E0%B8%AD%E0%B8%99%E0%B9%82%E0%B8%97%E0%B8%A3%E0%B8%A5%E0%B9%80%E0%B8%A5%E0%B8%AD%E0%B8%A3%E0%B9%8C#:~:text=Model%2DView%2DController%20(MVC,%E0%B8%8A%E0%B9%88%E0%B8%A7%E0%B8%A2%E0%B9%83%E0%B8%AB%E0%B9%89%E0%B8%81%E0%B8%B2%E0%B8%A3%E0%B8%9E%E0%B8%B1%E0%B8%92%E0%B8%99%E0%B8%B2%20%E0%B8%81%E0%B8%B2%E0%B8%A3>