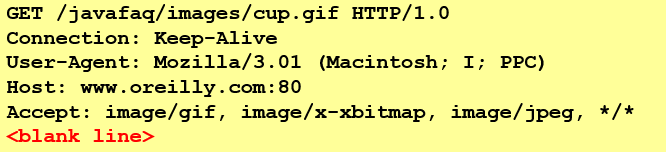
**Simple HTTP Server**

**Web Server**

เว็บเซิร์ฟเวอร์ (Web Server) หรือ HTTP Server คือ โปรแกรมให้บริการที่เก็บเว็บไซต์บนเครื่องผู้ให้บริการ (Server) แล้วให้ผู้ใช้ (Client) เรียกชมหน้าเว็บไซต์ได้โดยใช้โปรโทคอล HTTP ผ่านทางเว็บเบราว์เซอร์

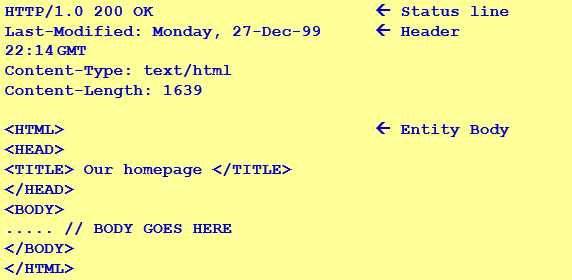
โดย HTTP โปรโตคอล คือ โปรโตคอลที่ใช้สื่อสารแลกเปลี่ยนข้อมูล(เช่น เอกสารHTML) ระหว่าง Web Server และ Web Client (Browser) โดยในการสื่อสารนั้นจะใช้ HTTP Request ในการร้องขอเนื้อหาจาก Web Server และ Web Server จะตอบกลับมาในรูปแบบ HTTP Response ซึ่ง HTTP Protocol มีหน้าตาดังนี้

**HTTP Request**

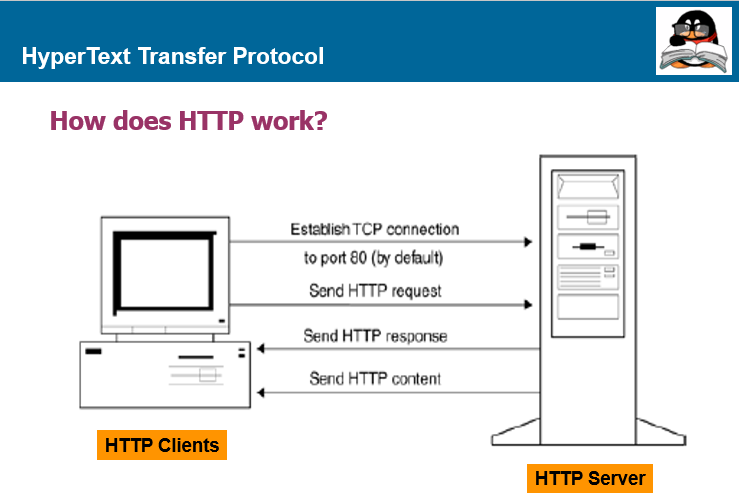


โดยบรรทัดแรกคือ Request Line ประกอบด้วย HTTP Method, Resource Path จาก URL และ HTTP Protocol Version และบรรทัดต่อๆมาคือ Header บ่งบอกข้อมูลเพิ่มเติม และ Body บ่งบอกเนื้อหาที่แนบมาสำหรับ Method อื่นๆอย่าง Post Method

**HTTP Response**

****

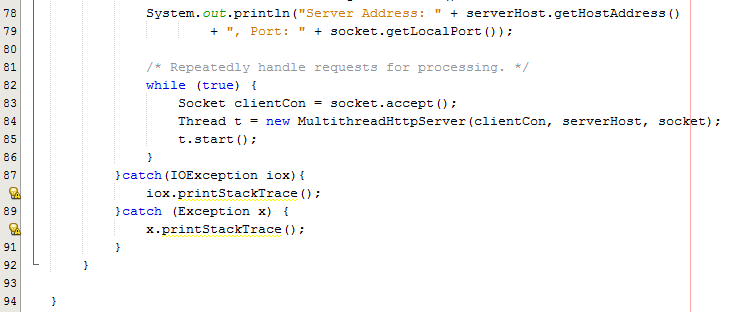
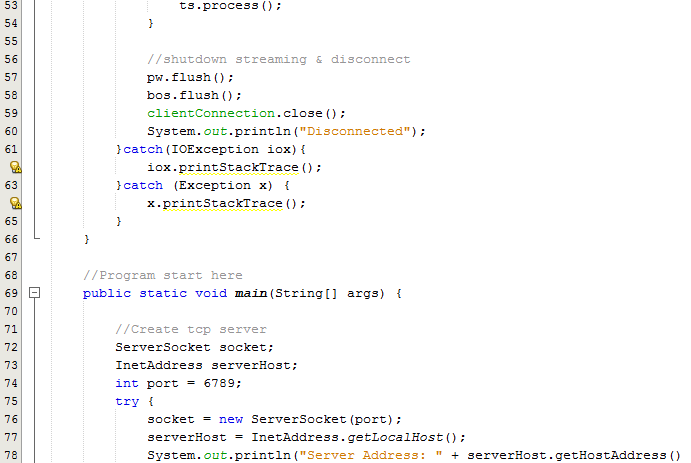
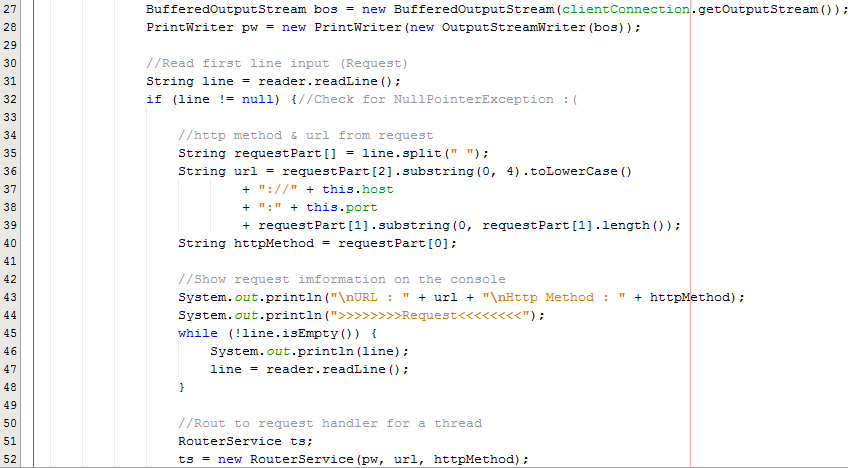
โดยบรรทัดแรกคือ Status line ประกอบด้วย HTTP Prorocol vesion, Status code และ ความหมาย ใช้ในการบ่งบอกสถานะต่างๆเช่น 4XX คือเกิดความผิดพลาดจากฝั่ง Client 5XX, คือเกิดความผิดพลาดจากฝั่ง Server, 2XX คือการตอบกลับ OK ไม่มีปัญหา เป็นต้น บรรทัดต่อๆมาคือ Header ใช้บ่งบอกข้อมูลเพิ่มเติม จากนั้นเว้นบรรทัดแล้วต่อด้วย Entity Body คือ Content ที่ Client ต้องการ

ซึ่ง HTTP Protocol ทำงานอยู่ในระดับ Application Layer บนโปรโตคอล TCP/IP ซึ่งเป็น Layer ระดับล่าง ดังนั้น การสร้าง HTTP Server จึงสร้างบน TCP Server (Socket Server) โดย HTTP Server มีการทำงานดังนี้

1. Client และ Server จะสร้างการเชื่อมต่อกันด้วย TCP/IP Protocol ที่ IP Address ของ Host และ Port ที่อยู่ของ Web Server ซึ่งปกติจะตั้งอยู่ที่ Port 80
2. ทำการรับส่งข้อมูล โดยการรับส่งข้อมูลนี้คือการสื่อสารด้วย HTTP Protocol โดยจะเริ่มจากฝั่ง Client ส่ง HTTP Request มาที่ฝั่ง Server จากนั้น Server จะทำการแปลความหมายของ HTTP Request นั้นและประมวลผลแล้วตอบกลับด้วย HTTP Response และ Content ที่ Client ต้องการ

**สร้าง HTTP Server ด้วยภาษา Java**

**Class MultithreadHttpServer**



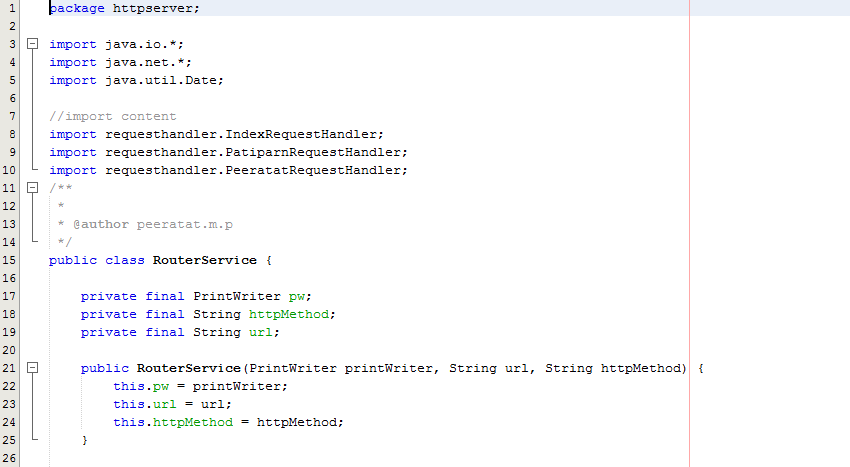
เนื่องจาก HTTP Protocol ทำงานในระดับ Application Layer จึงต้องสร้าง TCP Server ก่อน

บรรทัดที่ 69 : เริ่มต้นโปรแกรม  
บรรทัดที่ 72-79 : สร้าง Socket Server และพิมพ์ข้อความบอก IP Address และ Port Address ทาง Console  
บรรทัดที่ 82-86 : วนลูป โดย Server จะรอรับการเชื่อมต่อจาก Client จากนั้นก็จะแตก Thread แล้วส่งการเชื่อมต่อนั้น ไปให้ Thread นั้นๆจัดการจากนั้นก็กลับมารอรับการขอการเชื่อมต่อจาก Client ถัดไป

บรรทัดที่ 9 : เริ่มต้น Class MultithreadHttpServer  
บรรทัดที่ 11-13 : ตัวแปลของ Class ใช้ในการจัดเก็บการเชื่อมต่อและ IP Address กับ Port Address  
บรรทัดที่ 15-19 : Constructor method ใช้กำหนดค่าเริ่มต้นให้กับตัวแปลของ Class (ตอนที่สร้าง Instance บรรทัดที่ 84)  
บรรทัดที่ 22 : จุดเริ่มต้นการทำงานของ Thread (method run() บรรทัดที่22-66)

บรรทัดที่ 25-28 : สร้างท่อสำหรับการส่งข้อมูลจากการเชื่อมต่อโดยมี Instance ของ Class PrintWriter ชื่อว่า pw ใช้ในการส่งข้อมูล และมี Instance ของ Class BufferedReader ชื่อว่า reader ใช้ในการอ่านข้อมูลที่รับเข้ามา  
บรรทัดที่ 31-32 : อ่านข้อความที่ส่งมาบรรทัดแรกและตรวจสอบว่าเป็นค่าว่างหรือไม่  
บรรทัดที่ 35-44 : เนื่องจากบรรทัดแรกของ HTTP Request นั้นบ่งบอกถึง HTTP Method และ Path จึงนำบรรทัดแรกและ Host Address กับ Port Address มาทำการตัดต่อเพื่อสร้าง URL เพื่อให้สะดวกในการแปลความหมาย HTTP Request และนำไปใช้ในอนาคต จากนั้นแสดงผลที่ Console  
บรรทัดที่ 45-48 : วนลูปเพื่ออ่านข้อความ HTTP Request บรรทัดที่เหลือแล้วแสดงข้อความออกทาง Console  
บรรทัดที่ 51-53 : สร้าง Instance ของ Class RouterService เพื่อส่งต่อการทำงานของ Thread โดยส่ง Instance ของ Class PrintWriter และ String URL และ HTTP Method ไป

บรรทัดที่ 57-60 : หลังจากที่งานของ Thread เสร็จเรียบร้อยแล้วจึงปิดท่อส่งข้อมูล และปิดการเชื่อมต่อระหว่าง Client กับ Server

**Class RouterService**  
Class นี้มีหน้าที่แปลความหมายของ HTTP request ว่า Client ต้องการอะไรแล้วทำการส่งต่อการทำงานไปให้ส่วนที่ทำหน้าที่จัดการและตอบสนองต่อ Request นั้นๆ  


บรรทัดที่ 15 : เริ่มต้น Class RouterService  
บรรทัดที่ 17-19 : ตัวแปลของ Class ใช้เก็บ PrintWriter object, HTTP Method และ URL  
บรรทัดที่ 21-25 : Constructor Method ใช้กำหนดค่าเริ่มต้นให้กับตัวแปลต่างๆของ class  
บรรทัดที่ 27-65 : Method process() เป็น method สำหรับการทำงานของ Class RouterService  
บรรทัดที่ 30 : สร้าง Instance ของ Class URL จาก URL String เพื่อให้ใช้งาน URL ได้สะดวกขึ้น  
บรรทัดที่ 31-34 : ตรวจสอบว่าเป็น HTTP Protocol หรือไม่ ถ้าไม่ให้ส่งข้อความ Error และจบการทำงานของ Method  
บรรทัดที่ 35-64 : ตรวจสอบ HTTP method ของ Request ว่าเป็น GET Method หรือไม่ ถ้าไม่ให้ส่ง Response เพื่อบอกว่าไม่รองรับ Method นี้ (Status code 501 Not Implemented) ถ้าใช่ให้เข้าสู่การทำงานของ Switch-Case Statement เพื่อนำ Path จาก URL มาตรวจสอบว่า Client ต้องการอะไรแล้วจึงส่งต่อการทำงานไปให้ Class RequestHandler ที่ต้องจัดการกับ Request นั้นๆ

บรรทัดที่ 42-46 : ถ้าหากมี Request มาที่ Path เป็น page1 ก็จะส่งต่อการทำงานไปที่ Object ของ Class PatiparnRequestHnadler (ส่ง PrintWriter instance ไปให้ใช้ด้วย)  
บรรทัดที่ 52-59 : ถ้ามี Request ที่มี Path ไม่ตรงกับที่กำหนดไว้ให้ทำการส่ง HTTP Response กลับไปโดย Status code เป็น 404 Not Found

**Class PatiparnRequestHandler**  

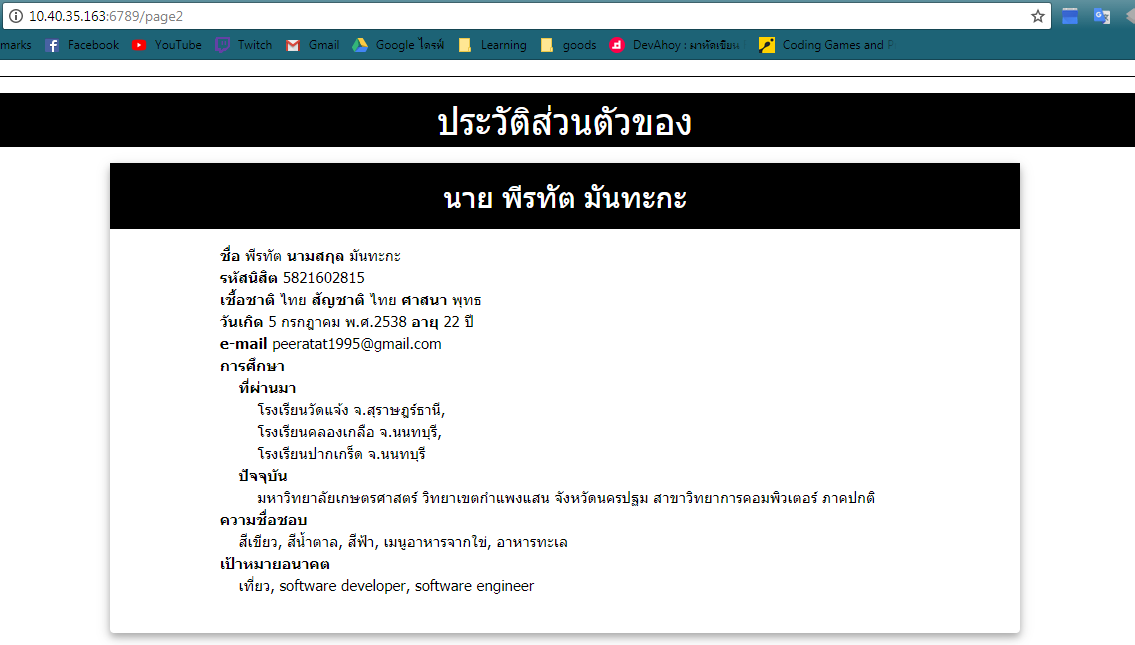

บรรทัดที่ 9 : จุดเริ่มต้นของ Class PatiparnRequestHandler  
บรรทัดที่ 10 : ตัวแปลชื่อ content ใช้เก็บเนื้อหาที่จะส่งให้แก่ Client  
บรรทัดที่ 60-62 : Constructor Method ใช้กำหนดค่าเริ่มต้นให้กับตัวแปล PrintWriter pw  
บรรทัดที่ 65-67 : Method getContent() ใช้ get ข้อมูลจากตัวแปล content  
บรรทัดที่ 70-73 : Method execute() ใช้ส่งข้อความ HTTP Response โดยเรียกใช้งาน sendHeader() เพื่อส่งข้อความ Status line : HTTP/1.0 200 OK และ Header ต่างๆ จากนั้นจะทำงานบรรทัดที่72เพื่อส่ง Content

เสร็จสิ้นการสร้าง Web Server อย่างง่ายด้วยภาษา Java

**ทดสอบ**

Run:   
Server Address: 10.40.35.163, Port: 6789

**Path: /  
**

**Path: /page2  
**