Servlet Life Cycle

Vittayasak Rujivorakul

Lecture 3

Servlet Life Cycle

Vittayasak Rujivorakul

Lecture 3

Four Step Life cycle

1. Loading and instantiation
2. init
3. service
4. destroy

Loading and Instantiating

- Class loaders ทำการ load และสร้าง instance ของ servlet class
- load system class จาก CLASSPATH
- สำหรับ servlet ที่อยู่ใน server_root / servlet / ใช้ class loader พิเศษของ Java webserver
- Remote servlets การ load class ข้ามเครื่องโดยการอ้างถึง codebase

init Method

- ถูกเรียกโดย service เมื่อ servlet ถูก load
- ถูกเรียกในครั้งแรก เมื่อ load servlet นั้น
- ไม่สามารถเรียกใช้อีก เป็นครั้งที่สองจนกว่าจะมีการทำลายโดย destroy method ก่อน
- มักใช้กับการรับค่าจาก requests ของ client
- ถึงแม้จะมีการ run servlet เป็นแบบ multithreaded ก็จะมีการเรียก init เพียงครั้งเดียวเท่านั้น นอกจากจะมีการ reload servlet ใหม่

init Method

```
public class MyServlet extends HttpServlet {
    public void init(ServletConfig conf) throws ServletException {
        super.init(conf);
        log("PhoneServlet.initialize: enter");
        String fileName = getInitParameter("phonelist");
        if(fileName == null){
            return;
        }
        log("PhoneServlet.initialize: filename = " + fileName);
    }
}
```


Example init Method

- การ initial servlet ทำได้โดยการ override method ชื่อ init ( )
- หากการ initialization มีสิทธิที่จะเกิดข้อผิดพลาด ซึ่งจะทำให้ servlet ไม่สามารถตอบสนอกต่อ ได้ให้ throw Unavaliable Exception เช่น ไม่สามารถ connect network
- ห้ามเรียกใช้ System.exit method

Retriving Initialization Parameter

- getInitParameter method รับ parameter name เข้ามาเป็น argument และ return เป็น String
- getParameter Names method ในกรณีที่ต้องการรู้ชื่อของ parameter
- initialization parameters ขึ้นอยู่กับข้อกำหนดของ server จะอยู่ใน file properties ซึ่งก็คือ "servlet properties" นั่นเอง

Service Method

- Process request ของ client ถูกแบ่งเป็น Thread
- รับ object 2 ตัวคือ
  - ServletRequest : รับ information จากการ request เช่น parameter, Hostname, IP
  - Servlet Response : จัดการ output Stream และทำการส่ง reply data
- ทุก servlet ที่ run จะไม่มีความเกี่ยวข้องกัน (อยู่คนละ Thread) แต่อาจมีปัญหาในการใช้งานตัวแปรร่วมกัน เช่น class variable

Multithreaded - Safe Servlets

- เมื่อมีการ access data ตัวเดียวกัน จาก thread หลาย ๆ ตัวอาจทำให้เกิดปัญหาค่าของตัวแปรนั้น ๆ เปลี่ยนแปลงไปมาไม่มีความน่าเชื่อถือ
- การแก้ปัญหาทำได้หลายวิธีดังนี้
  - ทำการ Define access types (public, protected, private) ให้กับ methods และ variables
  - ทำการ Synchronize ทุก ๆ instance methods ที่คิดว่าน่าจะเกิดความผิดพลาดได้
  - สร้าง accesser methods สำหรับการ access class variables

Synchronized Access Method

```
public class LogServlet extends HttpServlet{
    public void service(HttpServletRequest req, HttpServletResponse res) throws
    IOException {
        String data = “Hello World”;
        this.writeToFile(data);
    }
    private syncronized writeToFile(String data){
        //code that writes to the file
    }
}
```


SingleThreadModel

- เราสามารถสร้าง service ที่ไม่ run แบบ multiple thread เพื่อความปลอดภัยของข้อมูลและเพื่อให้แน่ใจว่า ณ ขณะใดขณะหนึ่งจะมีเพียง thread เดียวเท่านั้นที่ใช้ method หรือ เข้าถึงตัวแปรนี้

```
public class FormServlet extends HttpServlet implements SingleThreadModel{
    ….
}
```

- จะมีเพียงหนึ่ง thread ต่อหนึ่ง instance เท่านั้น เมื่อมี thread ใหม่เข้ามาจะทำการสร้าง instance ใหม่ (จองพื้นที่ memory ใหม่)

Destroy Method

- destroy method ถูกเรียกเมื่อมีการ unload servlet
- โดยจะทำสิ่งต่อไปนี้
  - Undo ทุกอย่างที่เกิดจากการ Initialization
  - Save ข้อมูลที่อยู่ใน memory ลง Hard Device
- นิยมใช้ในการปิด connection ของ Database หรือ Network
- ทำการเก็บบันทึกลง log files

Example destroy Method

```
public void destroy( ){
    try{
        con.close( );
    }catch(SQLException e) {
        while(e != null){
            getServletContext.log("SQLException: " + e.getSQLState( ) + '\t' +
            e.getMessage( ) + '\t' + e.getErrorCode( ) + '\t');
            e = e.getNextException( );
        }
    }
}
```


การ set Servlet Properties

- การเก็บใช้ ชื่อ servlet.properties
- วิธีการ set properties

  alias servlet.code = <servlet Name>

  alias servlet.initparame = <name = value>,

  <name = value>
- เวลาเรียกใช้บน Browser สามารถเรียกใช้ชื่อ alias ได้