Persistent State in HTTP Servlets

Vittayasak Rujivorakul

Lecture 6

Persistent State in HTTP Servlets

Vittayasak Rujivorakul

Lecture 6

Saving Client State

- Servlet API กำหนด 2 ช่องทางสำหรับเข้าถึงสถานะของ Client คือ
  - Session
  - Cookies

Session Tracking

- คือกลไกที่ Servlet ใช้บริหารสถานะเกี่ยวกับลำดับของการ request จากผู้ใช้บางคน ในบางช่วงเวลา
- Session ของผู้ใช้หนึ่ง ๆ จะถูกจัดสรรให้กับ servlets หลาย ๆ ตัว ใช้ร่วมกันได้ ซึ่งวิธีการใช้ Session Tracking เราต้อง
  - รับค่า Session (Http Session object) จากผู้ใช้
  - บันทึก และอ่านข้อมูลจาก Object ของ Http Session
  - กำหนดอายุของ Session (optional)

Obtaining a Session

- getSession method ของ HttpServletRequest จะคืน users session กลับมาให้เมื่อเราเรียกใช้ method นี้ด้วยการส่ง argument เป็น true
- การเรียกใช้ getSession ต้องเรียกก่อนจะมีการเขียน Output ไปยัง response (ก่อนจะมีการ Access Writer)

ตัวอย่าง getSession

```
public class CatalogServlet extends HttpServlet {

    public void doGet (HttpServletRequest request,
                       HttpServletResponse response)
    throws ServletException, IOException
    {
       // Get the user's session and shopping cart
       HttpSession session = request.getSession(true);
       // ...
       out = response.getWriter();
       //...
    }
}
```


Store and Getting Data from a Session

Http Session interface มี method สำหรับ บันทึกและอ่านข้อมูล

- คุณสมบัติของ Session ทั่ว ๆ ไป เช่น session identifier
- Application data ซึ่งบันทึกในรูปของ name-value pair โดยที่ name จะเป็น String และ Value คือ object ของ Java
- เนื่องจากจะมี Servlets จำนวนมากเข้ามาใช้ Session เราควรยอมรับระเบียบการกำหนดชื่อ เพื่อเลี่ยงอุบัติเหตุจากการที่ข้อมูลของ Servlet ซ้อนทับกัน ซึ่งมีหลักการกำหนดดังนี้
  - ควรระบุเป็น servletname.name โดย servletname คือชื่อเต็ม ๆ เช่น com.acme.WidgetServlet.state โดยมี state เป็น name

ตัวอย่างการอ่านเขียน application data ของ session

```
public class CatalogServlet extends HttpServlet {
    public void doGet (HttpServletRequest request,
                       HttpServletResponse response)
    throws ServletException, IOException
    {
       // Get the user's session and shopping cart
       HttpSession session = request.getSession(true);
                          ShoppingCart cart =
         (ShoppingCart)session.getValue(session.getId());

    // If the user has no cart, create a new one
    if (cart == null) {
       cart = new ShoppingCart();
              session.putValue(session.getId(), cart);
    }
      // ...
    }
}
```


Object & Session

- ข้อมูลต่าง ๆ ที่จะส่งไปกับ session จะเป็น object เมื่อรับ object มาแล้ว ก็จะสามารถกระทำการใด ๆ กับ object นั้นได้

```
public void doGet (HttpServletRequest request,
                   HttpServletResponse response)
    throws ServletException, IOException
{
    HttpSession session = request.getSession(true);
    ShoppingCart cart =
            (ShoppingCart)session.getValue
            (session.getId());
    //...
    // Check for pending adds to the shopping cart
    String bookId = request.getParameter("Buy");
```

```
//If the user wants to add a book, add it and print the
result
      String bookToAdd = request.getParameter
("Buy");
      if (bookToAdd != null) {
         BookDetails book = database.getBookDetails
(bookToAdd);

         cart.add(bookToAdd, book);
         out.println("<p><h3>" + ...);
      }
   }
```

New Session

- Session สามารถกำหนด เป็น new ได้ ซึ่ง New session จะทำให้ method is New ของ HttpSession return ค่า true เช่น client ไม่รู้จัก session เป็นต้น
- new session จะไม่มีข้อมูลภายในตัวเอง

Invalidating the Session

- เราสามารถกำหนดการหมดอายุของ session ได้เอง หรือกำหนดเวลาหมดอายุของ Servlet ได้ เช่น ใน Java Web Server จะทำการยกเลิก session เมื่อไม่มีใครติดต่อ เป็นเวลา 30 นาที
- การหมดอายุของ session หมายถึง การนำ Http Session object และข้อมูลของมันออกไปจากระบบ
- การกำหนดการหมดอายุของ session เราจะใช้ method ที่ชื่อว่า invalidate ( )

ตัวอย่าง inbalidate

```
public class ReceiptServlet extends HttpServlet {
      public void doPost(HttpServletRequest request,
                         HttpServletResponse response)
        throws ServletException, IOException {

        //...
        ShoppingCart scart = (ShoppingCart)session.getValue(session.getId());
        //...
        // Clear out shopping cart by invalidating the session
        session.invalidate();

          // set content type header before accessing the Writer
          response.setContentType("text/html");
          out = response.getWriter();
          ...
    }
}
```


Using Cookies

- Cookies คือทางเดียวสำหรับ server (หรือ Servlet ที่เป็นส่วนของ Server) ที่จะส่งข้อมูลไปเก็บ ที่ Client
- Server สามารถอ่านข้อมูลจาก cookies ของ clients
- Servelet จะส่ง cookies ไปยัง clients โดยเพิ่มส่วนที่เป็น HTTP response headers
- Client จะส่ง cookies คืนโดยอัตโนมัติ โดยเพิ่มเนื้อที่ไปยังส่วนที่เป็น HTTP request header
- แต่ละ HTTP request และ response header จะมีชื่อและข้อมูลค่าหนึ่ง ๆ

Using Cookies

- Server สามารถส่ง cookies ได้ตั้งแต่ 1 ตัวขึ้นไป ไปยัง client software เช่น web browser สามารถสนับสนุน cookies ได้ถึง 20 ตัวต่อ host โดยมีความยาวของ cookies แต่ละตัวได้ถึง 4 k bytes
- Cookies ที่ Client เก็บไว้สำหรับ server จะถูกส่งกลับ โดย Client ไปยังเฉพาะ Server เจ้าของ cookies นั้น ๆ
- Server สามารถที่จะมี servlets ได้หลาย ๆ ตัว ดังนั้น Servlet ที่ทำงานภายใน server จะใช้ cookies ร่วมกัน

การส่ง cookie

- Instantiate Cookie object
- Set atribute ต่าง ๆ
- Send cookie

การรับข้อมูลจาก cookie

1. รับ cookies ทุกตัวกลับจาก request ของผู้ใช้
2. ค้นหา cookies ซึ่งมีชื่อตามที่สนใจ โดยใช้เทคนิค ของโปรแกรม
3. รับข้อมุ,ของ cookies ที่ต้องการ

Creating a Cookies

- Constructor ของ javax.servlet.http.cookie จะสร้าง cookie ซึ่งจะทำการ initial ชื่อและ value เราสามารถเปลี่ยนค่า ของ cookie ภายหลังโดยใช้ method ชื่อ set Value
- ชื่อของ cookie จะต้องเป็นไปตามกฎของ HTTP / 1.1 token ซึ่งต้องไม่มีตัวอักษรพิเศษดังที่กำหนดไว้ใน RFC 2068
- ค่าของ cookies สามารถเป็น string ได้บางชนิด และไม่ควรใช้ whitespace หรือตัวอักษรต่อไปนี้คือ

  [ ] ( ) = , " / ? @ : ;
- ถ้า servlet ส่ง response ไปยังผู้ใช้ด้วย Writer ต้องสร้าง cookie ก่อนที่จะมีการทำงานกับ Writer

ตัวอย่าง การ Create Cookies

```
public void doGet (HttpServletRequest request, HttpServletResponse response) throws ServletException,
    IOException
  {
  // Check for pending adds to the shopping cart
     String bookId = request.getParameter("Buy");

   //If the user wants to add a book, remember it by adding a cookie
     if (bookId != null) {
        Cookie getBook = new Cookie("Buy", bookId);
        ...
     }

     // set content-type header before accessing the Writer
     response.setContentType("text/html");

     // now get the writer and write the data of the response
     PrintWriter out = response.getWriter();
     out.println("<html>" +
            "<head><title> Book Catalog </title></head>" + ...);
     //...
  }
```


Setting Cookie Attribute

- Class Cookie กำหนด method สำหรับ set values และ attribute หรือจะ set comment

```
public void doGet (HttpServletRequest request,
                   HttpServletResponse response)
      throws ServletException, IOException
   {
     // ...
    //If the user wants to add a book, remember it by adding a
cookie
      if (values != null) {
         bookId = values[0];
         Cookie getBook = new Cookie("Buy", bookId);
         getBook.setComment("User wants to buy this book "
+
                  "from the bookstore.");
      }
     // ...
   }
```


Delete Cookie

- เราสามารถกำหนด อายุของ cookie ได้ และเมื่อกำหนดเป็น 0 จะหมายถึง การลบ cookie นั้น

```
public void doGet (HttpServletRequest request,
                   HttpServletResponse response)
      throws ServletException, IOException
   {
      ...
      /* Handle any pending deletes from the shopping
cart */
      String bookId = request.getParameter("Remove");
      ...
      if (bookId != null) {
         // Find the cookie that pertains to the book to
remove
         //...
         // Delete the cookie by setting its maximum age to
zero
         thisCookie.setMaxAge(0);
         //...
      }
      // also set content type header before accessing
the Writer
      response.setContentType("text/html");
      PrintWriter out = response.getWriter();

      //Print out the response
      out.println("<html> <head>" +
            "<title>Your Shopping Cart</title>" +
...);
```


Sending the Cookie

- Cookies จะถูกส่งไปในฐานะของ header ของ response ไปยัง client ซึ่งจะเพิ่มเข้าไปด้วย method add Coolie ของ Http Servlet Response

```
public void doGet (HttpServletRequest request, HttpServletResponse response) throws ServletException,
IOException
   {
      ...
   //If the user wants to add a book, remember it by adding a cookie
      if (values != null) {
         bookId = values[0];
         Cookie getBook = new Cookie("Buy", bookId);
         getBook.setComment("User has indicated a desire " +
                  "to buy this book from the bookstore.");
         response.addCookie(getBook);
      }
            // ...
   }
```

Retrieving Cookies

- Clients จะคืนค่า cookies เข้าเป็นส่วนหนึ่งของ HTTP request headers การกู้ cookie บางตัวคืน เราต้อง รับทุก ๆ cookies
- Method getCookies จะคืนค่า array ของ object Cookies ซึ่งเราสามารถค้นหา cookie ที่เราต้องการได้
- การรับชื่อของ cookies หนึ่ง ๆ เราจะใช้ method ชื่อว่า getName

get Cookie

```
public void doGet (HttpServletRequest request,
                   HttpServletResponse response)
    throws ServletException, IOException
  {
    //...

    /* Handle any pending deletes from the shopping cart */
    String bookId = request.getParameter("Remove");
    ...
    if (bookId != null) {

      // Find the cookie that pertains to the book to remove
      Cookie[] cookies = request.getCookies();
     // ...
// Delete the book's cookie by setting its maximum age to zero
      thisCookie.setMaxAge(0);
    }

    // also set content type header before accessing
the Writer
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    //Print out the response
    out.println("<html> <head>" +
          "<title>Your Shopping Cart</title>" + ...);
```


Getting the Value of a Cookie

- การค้นหาข้อมูลของ Cookie ใช้ method getValue

```
public void doGet (HttpServletRequest request,
                   HttpServletResponse response)
    throws ServletException, IOException
  {
    //...
    /* Handle any pending deletes from the shopping
cart */
    String bookId = request.getParameter
("Remove");
    //...
    if (bookId != null) {
      // Find the cookie that pertains to that book
      Cookie[] cookies = request.getCookies();
      for(i=0; i < cookies.length; i++) {
        Cookie thisCookie = cookie[i];
        if (thisCookie.getName().equals("Buy") &&
thisCookie.getValue().equals
(bookId)) {
// Delete the cookie by setting its maximum
          // age to zero
          thisCookie.setMaxAge(0);
        }
      }
    }

    // also set content type header before accessing the
Writer
    response.setContentType("text/html");
    PrintWriter out = response.getWriter();

    //Print out the response
    out.println("<html> <head>" +
          "<title>Your Shopping Cart</title>" + ...);
```


Handling All Browser

- ในกรณีที่บาง browser อาจไม่สนับสนุน cookies หรือผู้ใช้ยกเลิกการใช้ cookies เราต้องใช้ URL rewriting แทน
- เมื่อใช้ URL rewriting เราต้อง เรียกใช้ method ซึ่งจะต้องมี session ID ใน Link เราต้องเรียก method นี้สำหรับทุก ๆ link ใน servlet response
- method ซึ่งติดต่อกับ session ID ด้วย URL คือ

  Http Servlet Response.encodeUrl (JSDk 2.0)

  Http Servlet Response.encodeUrl (JSDk 2.1)
- ถ้า server พบว่า browser สนับสนุน cookies URL นั้น จะไม่มีการ rewrite

ตัวอย่าง URL rewriting

```
public class CatalogServlet extends HttpServlet {


    public void doGet (HttpServletRequest request,
                       HttpServletResponse response)
      throws ServletException, IOException
    {
     // Get the user's session and shopping cart, the Writer,
etc.
     //...
     // then write the data of the response
      out.println("<html>" + ...);
     //...
     // Get the catalog and send it, nicely formatted
      BookDetails[] books = database.getBooksSortedByTitle
();
     //...
```

```
for(int i=0; i < numBooks; i++) {
            ...
            //Print out info on each book in its own two rows
            out.println("<tr>" + ... "<a href=\"" +
            response.encodeURL("/servlet/bookdetails?
bookId="+
            bookId) +
            "\"> <strong>" + books[i].getTitle() +
            "  </strong></a></td>" + ..."<a href=\"" +
            response.encodeURL("/servlet/catalog?
Buy="+bookId)
            + "\">   Add to Cart  </a></td></tr>" +
        }
      }
    }
```


URL rewriting

- ถ้าผู้ใช้ client บน link ซึ่งใช้ rewrite URL Servlet จะจดจำ session ID เมื่อ method getSession ถูกเรียกใช้ ก็จะใช้ ID นี้จาก Http Session ของผู้ใช้
- หาก browser ของผู้ใช้ ไม่ support cookies และ ผู้ใช้เลือกบน URL ที่ไม่มีการ rewrite session ของผู้ใช้ก็จะสูญหาย servelet ก็จะสร้าง session ใหม่ ซึ่งไม่มีข้อมูลที่เกี่ยวข้องกับ session เดิม