

Webanwendungen Grundlagen



Cookies

Prof. DI Dr. Erich Gams
htl-wels.e.gams@eduhi.at



Übersicht ➡ Was lernen wir?

- Vorteile/Nachteile
- Cookies senden/empfangen
- Besuchertracking
- Session/Persistent Cookies
- Benutzereinstellungen merken

Cookies



Typische Verwendung von Cookies

- Session Tracking
 - Übernimmt die Session API
- Benutzernamen und Passwörter vermeiden
- Eine Seite an Benutzer anpassen
- Werbung

Motivation, Cookies nicht zu verwenden ☹

- **Das Problem: Privatsphäre (und nicht die Sicherheit!)**
 - Server speichern die vorangegangenen Schritte
 - Persönliche Informationen die man herausgibt, können gespeichert werden (Ich könnte theoretisch Kreditkarteninfos in einem Cookie speichern!!!!)
 - Server können Cookieinfos über Thirdpartyserver (doubleclick.net) austauschen
 - In früheren Browserversionen konnte man mit JavaScript Cookies auslesen (Bug).
- **Resümee für Servletautoren**
 - Cookies vermeiden, oder Absturz des Servlets vermeiden, sollten Cookies ausgeschaltet sein
 - Keine sensiblen Daten in Cookies schreiben

Cookies am Client erstellen

- **Erzeugen eines Cookie Objekts.**
 - `Cookie c = new Cookie("userID", "a1234");`
- **Die maximale Lebensdauer festlegen**
 - Cookie wird dann auf Platte gespeichert.
 - `c.setMaxAge(60*60*24*7); // 1 Woche`
 - Wenn nicht gesetzt, dann Lebensdauer=Sessiondauer
- **Cookie an den Browser schicken**
 - `response.addCookie(c);`

Cookies lesen

- request.getCookies()
- Rückgabe: ein Array von Cookie Objekten.

```
String cookieName = "userID";
Cookie[] cookies = request.getCookies();
if (cookies != null) {
    for(Cookie cookie: cookies) {
        if (cookieName.equals(cookie.getName())) {
            doSomethingWith(cookie.getValue());
        }
    }
}
```

Beispiel: Neue Besucher ausfindig machen

```
@WebServlet("/repeat-visitor")
public class RepeatVisitor extends HttpServlet {
    public void doGet(HttpServletRequest request,
                      HttpServletResponse response)
        throws ServletException, IOException {
        boolean newbie = true;
        Cookie[] cookies = request.getCookies();
        if (cookies != null) {
            for(Cookie c: cookies) {
                if ((c.getName().equals("repeatVisitor")) &&
                    (c.getValue().equals("yes"))) {
                    newbie = false;
                    break;
                }
            }
        }
    }
}
```


Beispiel: Neue Besucher ausfindig machen

```
String title;
if (newbie) {
    Cookie returnVisitorCookie =
        new Cookie("repeatVisitor", "yes");
    returnVisitorCookie.setMaxAge(60*60*24*365);
    response.addCookie(returnVisitorCookie);
    title = "Welcome Aboard";
} else {
    title = "Welcome Back";
}
response.setContentType("text/html");
PrintWriter out = response.getWriter();
... // (Output page with above title)
```

Cookie Utility

```
public class LongLivedCookie extends Cookie {  
    public static final int SECONDS_PER_YEAR = 60*60*24*365;  
    public LongLivedCookie(String name, String value) {  
        super(name, value);  
        setMaxAge(SECONDS_PER_YEAR);  
    }  
}
```

Cookie Utility

```
public class CookieUtilities {  
    public static String getCookieValue  
        (HttpServletRequest request,  
         String cookieName,  
         String defaultValue) {  
        Cookie[] cookies = request.getCookies();  
        if (cookies != null) {  
            for(Cookie cookie: cookies) {  
                if (cookieName.equals(cookie.getName()))  
                    return(cookie.getValue());  
            }  
        }  
        return(defaultValue);  
    }  
    ...  
}
```

Cookiewerte ändern

- Cookies löschen
 - `setMaxAge(0)`
- Cookiewert ändern
 - Sende denselben Cookienamen mit einem anderen Wert

Cookie Theories





Aufgabe

- 1) Benutzerzugriffe:
 - Willkommen! Du bist das erste Mal da
 - Willkommen zurück! Du warst schon x mal mit dem Browser XY auf unserer Seite
- 2) Erweiterung Login:
 - Vorausgefüllte Werte bei Angabe des Namens und der Adresse
 - (Deine letzten Bestellungen)