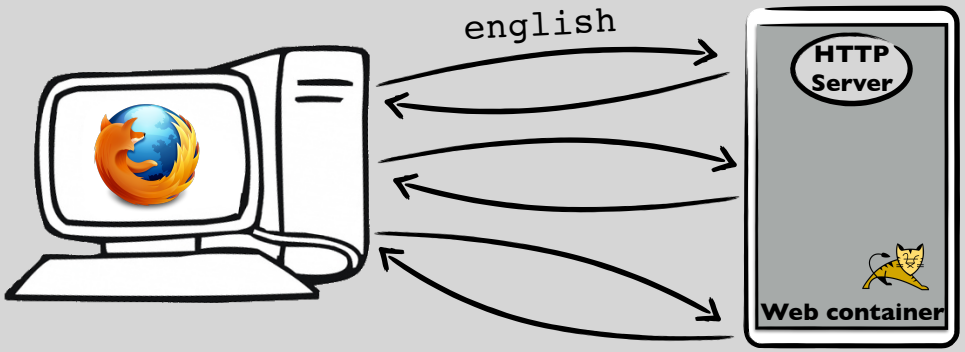
Web 2

Lesson 9: Sessions

AGENDA

?



- How can we save our language choice?

Option 1: Client

1. client chooses a language and sends it to the server
2. server reads chosen language and makes a **cookie** for the language
3. server gives this to the client in the **response header**
4. browser sends the cookie with each request in a **request header**
5. server uses this info in order to show the right language

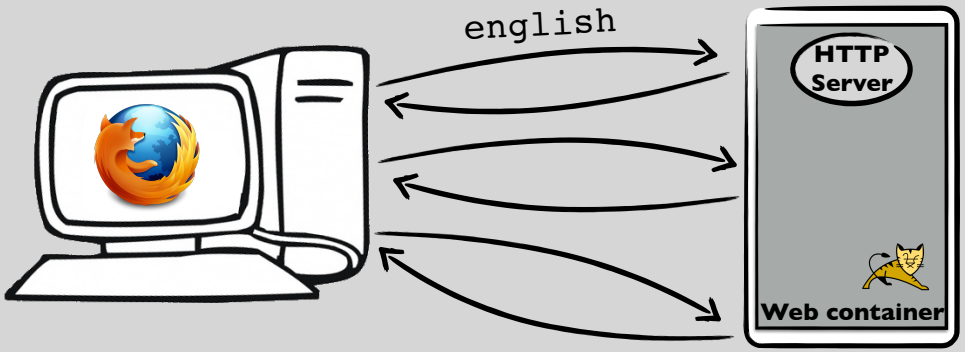
Cookies

AGENDA

EXAM QUESTIONS…

- ☑ Is a session saved at the client side or at the server side?
- ☑ Explain how sessions works.
- ☑ How can you set the expiration time of a session?
- ☑ How can you end a session?
- ☑ Is a session safe? Why or why not?
- ☑ When is it better to use a session, when is it better to use a cookie?
- ☑ …

?



- How can we save our language choice?

Option 2: Server

1. server stores choice
2. server passes a token, id to client
3. client saves this id and sends it in each request
4. server uses this id to find the stored language

Sessions

SESSIONS

- What?
  - **name-value** pairs, eg. `persons=personList`
- Why?
  - to save **temporary data**
  - info about **one client**

SESSIONS

- Where?
  - made by a '**script**' on the **server**
  - saved by **server**
  - id saved by the **client** in its **browser**

SESSIONS

- How?
  - server makes a **session** object
  - ... and **stores the information** in the session
  - ... and gives the **id** of the session to the client in the **response header**
  - browser sends the id within each **request header**
  - server uses the id to **find** the corresponding session
  - server gets the **information** from the session

NAME EXAMPLE

DEMO

- server makes a **session** object:
  request.getSession() method
- ... and **stores the information** in the session:
  session.setAttribute() method
- ... and gives the id of the session to the client in the response header

STEP 1

`name=Jan`

`Set-Cookie:`
`JSESSIONID=D1ACEDFDFC32`
`F4C3910CAC24505E21B0`

String name =
request.getParameter(
"name");

Person user = …;
HttpSession session =
request.getSession();
session.setAttribute
("user", user);

- browser sends the id within each **request header**
- server uses the id to **find** the corresponding session **request.getSession**() method
- server gets the **information** from the session **session.getAttribute()** method

STEP 2

```
Cookie:
JSESSIONID=D1ACEDFDFC32
F4C3910CAC24505E21B0
```

HttpSession session = request.getSession();

String language = session.getAttribute ("user").getName();

CREATE - GET SESSION

- `request.getSession()`
  - returns existing session OR
  - creates a new one if no session exists

Add info to - Get info from Session

- `session.setAttribute(name, object)`
  - if already attribute with same name ⟶ replaced!
- `session.getAttribute(name)`

TILL WHEN?

- until **destroyed**
  - `session.invalidate()`
- or until **timeout**
  - `session.setMaxInactiveInterval(seconds)`
  - web.xml
  - server default
  - closing browser

Invalidate: invalidates session and unbinds all objects

Sessions or Cookies?

COOKIE

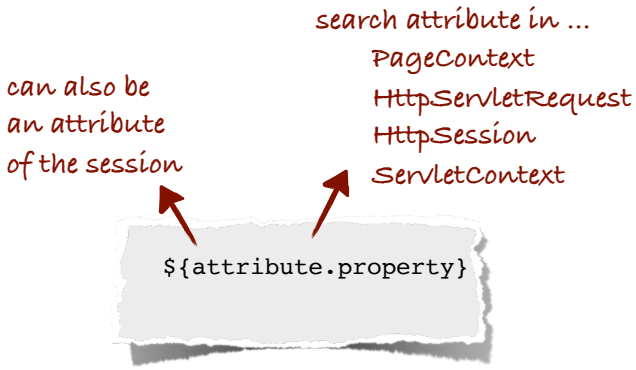
- in browser
- can be removed/blocked
- unsafe → google "edit http cookies"
- expiration time can be set
- only to use when sending few/small data
- example: choice of language → network load!

SESSION

- on server
- safe(r)
  - only sessionID accessible, not data
  - risk: 'session hijacking'
- can be invalidated
- more data possible
- example: shopping cart

REMARK:

EXPRESSION LANGUAGE: . OPERATOR



the container does not only checks the request, but also the other places where an attribute might be stored

REMARK:

IMPLICIT OBJECTS

```
${attribuut.property}

${implicitObject.keyOrProperty}
```

- pageScope
- requestScope
- sessionScope
- applicationScope
- param
- paramValues
- header
- headerValues
- cookie
- initParam
- pageContext


```
${header["host"]}
```

```
${param["id"]}
```

```
${initParam.mainEmail}
```

```
${cookie.JSESSIONID.value}
```

AGENDA