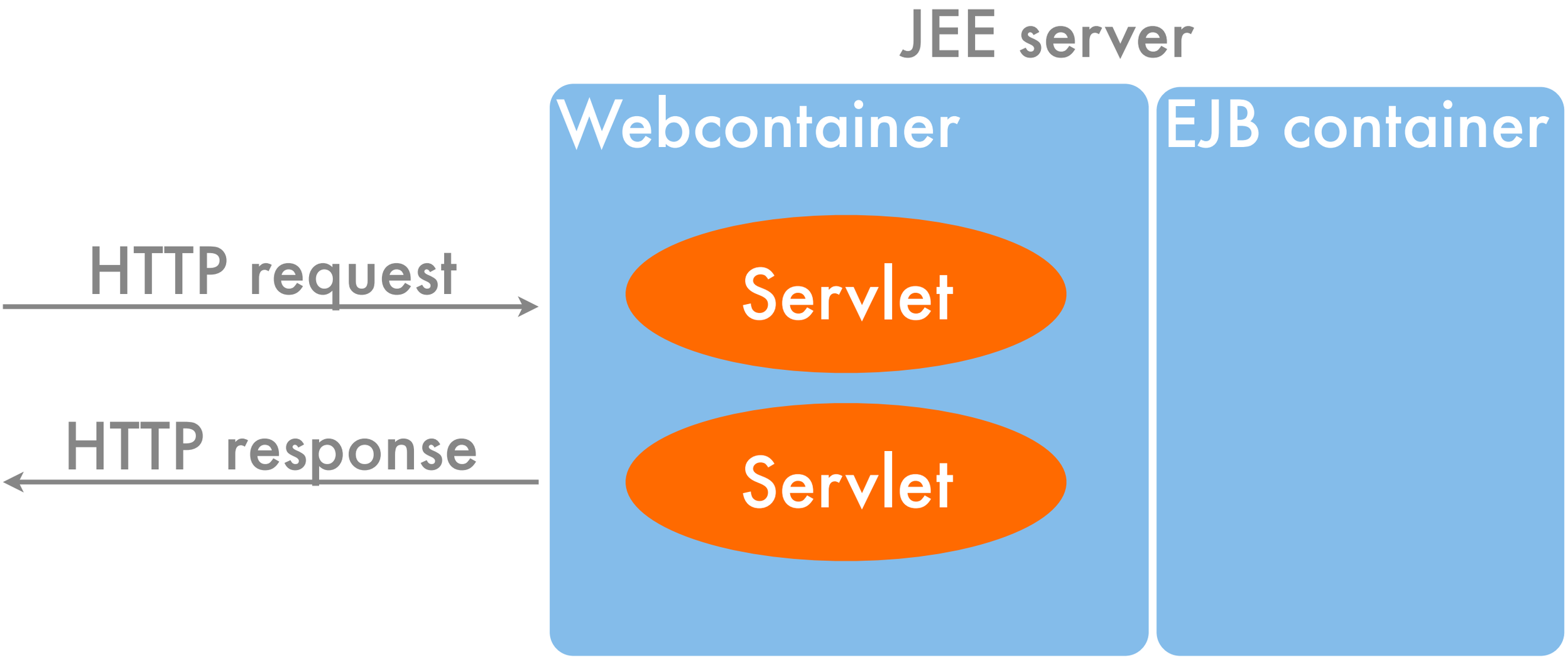
Servlets

joost.vennekens@kuleuven.be

"Plug-in" architectuur



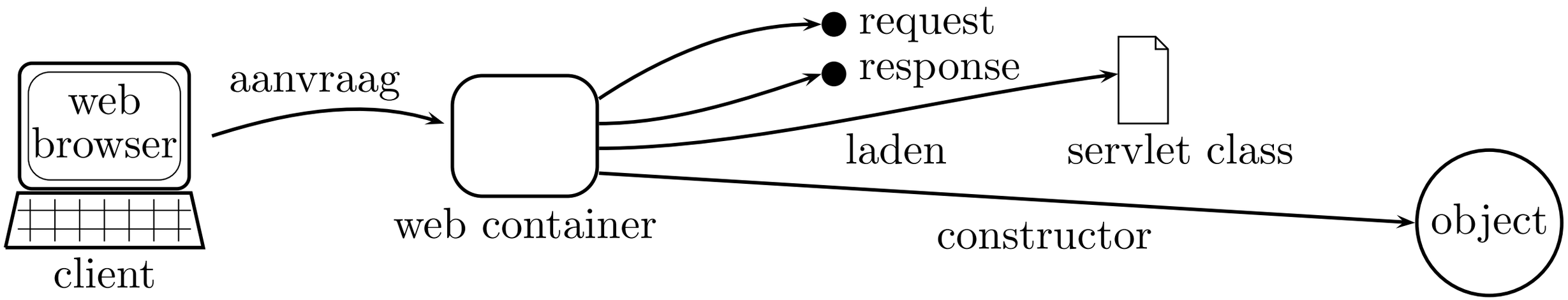
Subklasse van abstracte klasse HttpServlet, die Servlet implementeert

Diensten van webcontainer

- Communicatie via HTTP
- Beheer levenscyclus
- Multithreading
- Beveiliging
- Vertaling van JSPs

Afhandelen van aanvraag

HttpServletRequest HttpServletResponse



Methode oproepen in nieuwe thread

javax.servlet.Servlet

Method Summary

void	**destroy**() Called by the servlet container to indicate to a servlet that the servlet is being taken out of service.
ServletConfig	**getServletConfig**() Returns a ServletConfig object, which contains initialization and startup parameters for this servlet.
java.lang.String	**getServletInfo**() Returns information about the servlet, such as author, version, and copyright.
void	**init**(ServletConfig config) Called by the servlet container to indicate to a servlet that the servlet is being placed into service.
void	**service**(ServletRequest req, ServletResponse res) Called by the servlet container to allow the servlet to respond to a request.

HttpServlet

- javax.servlet.http.HttpServlet implementeert javax.servlet.Servlet

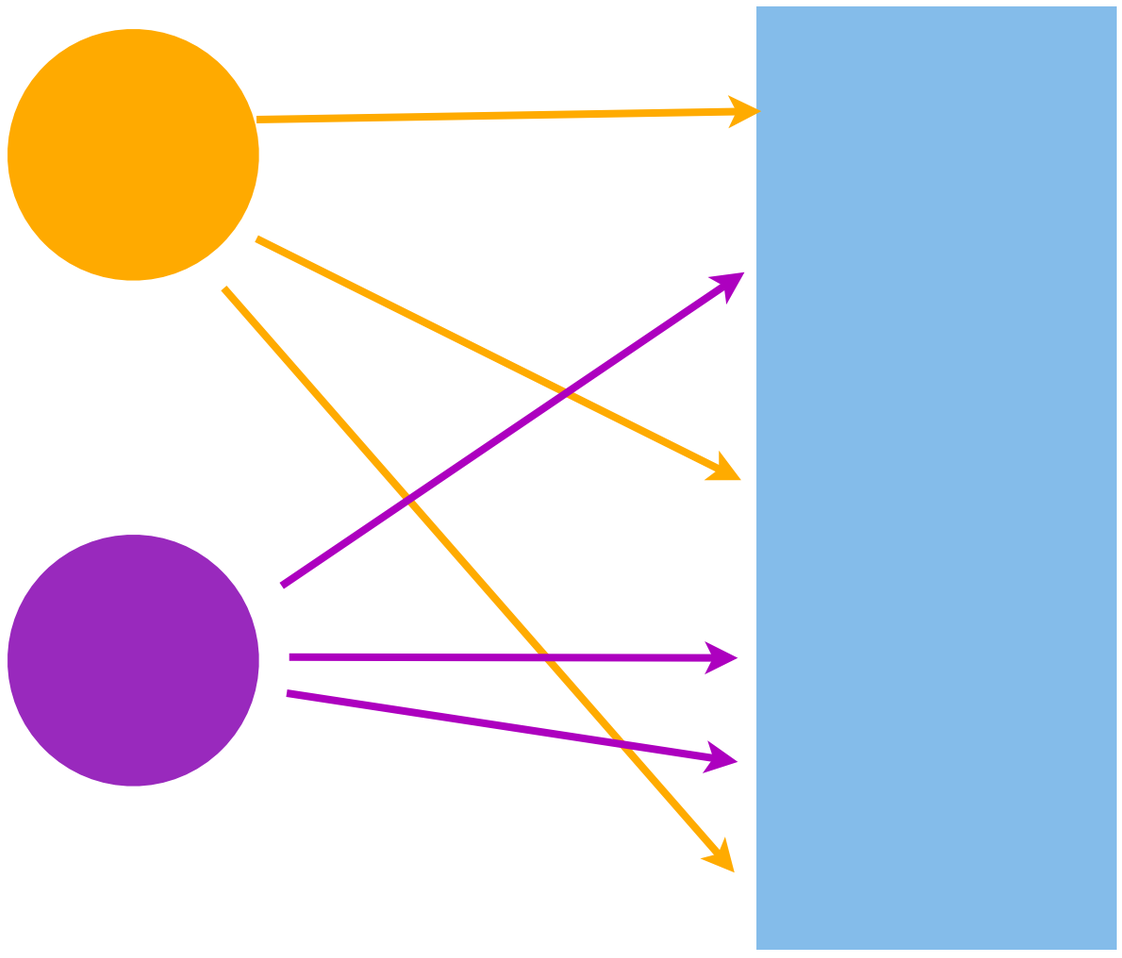
Method Summary

protected void	**doDelete**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a DELETE request.
protected void	**doGet**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a GET request.
protected void	**doOptions**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a OPTIONS request.
protected void	**doPost**(HttpServletRequest req, HttpServletResponse resp) Called by the server (via the service method) to allow a servlet to handle a POST request.

HttpServletRequest

- Parameters:
  - String kleur = request.getParameter(”kleur”);
  - String[] kleur = request.getParameterValues(”kleur”);
- Andere informatie:
  - String agent = request.getHeader(”User-Agent”);
  - String m = request.getMethod();
- Sessie management

Beheer van sessies



- Sessie is opeenvolging requests van zelfde client
- Belangrijk
- Bovenop HTTP

```
HttpSession sessie = request.getSession();
```

Hoe?

- Iedere gebruiker een uniek nummer (JSESSIONID)
- Meesturen met ieder request
  - Met cookies (liefst)
  - Herschrijven van URL

Cookies

- Paar van **naam** en **waarde**
- Op client computer
- Meegestuurd met HTTP response / request
- In JEE:
  - Cookie[] koekjes = request.getCookies();
  - koekjes[0].getName(); koekjes[0].getValue();

Cookies (2)

- Aanmaken: Cookie c = new Cookie(“ch”, “fl”);
- Toevoegen aan response:

  response.addCookie(c);
- Eigenschappen aanpassen:

  c.setMaxAge(7*24*60*60);
- JSESSIONID achter de schermen

Zonder cookies

- URL rewriting:

  out.println(”<a href=\”” +
  response.encodeURL(”BierKeuze.do”) + ”\”> Hier
  klikken ”);

- Resultaat:

  http://localhost/
  Bierkeuze.do;jsessionid=1234567

Sessie in JEE

- Opvragen van HttpRequest:
  - HttpSession sessie = request . getSession ();
- Gegevens bijhouden: sessie.setAttribute(”getal”, ”37”)
- Bij eerste oproep van nieuwe client, wordt Session object aangemaakt
  - if (sessie.isNew()) out. println(”Nieuwe sessie!”); else out.println(”Hello again”);
- Einde: timeout met WDD / setMaxInactiveInterval() of sessie.invalidate();
  - Vb: Sessie

HttpResponse

- MIME type: response.setContentType(”text/html”);
- Hoofdingen:
  - response.setHeader(”content–type”, ”text/html”);
  - response.addHeader(”chips”, ”flippo”);
- PrintWriter voor strings: response.getWriter();
- Stroom voor bytes:
  ServletOutputStream s = response.getOutputStream();
- Pas op: headers en cookies **voor** getWriter()

Redirect

- Doorsturen naar elders
  - response.sendRedirect(” . . . ”) ;
- HTTP 301

Request dispatch

```
RequestDispatcher rd =
                request.getRequestDispatcher(”resul.jsp”)
```

- Forward
  - rd.forward(request, response);
  - Zelf PrintWriter niet gebruiken
- Include
  - rd.include(request, response);
  - Andere pagina geen headers aanpassen

Initializatie

- Run-time parameters in WDD
- Specifiek: in `ServletConfig` object
  - Argument bij `init(ServletConfig)`
    - Implementatie slaat object op
    - Roept `init()` op
  - `getServletConfig()`
- Algemeen: in `ServletContext` object

VB: ConfigServlet.java

Listeners

- JEE Server: plug-in architectuur
- Veel gebeurt achter de schermen
- Soms toch relevant
- Via Listeners ingrijpen op bepaalde momenten

Mogelijke listeners

Object	Ev	Listener Interface	Event Class
web context	(1)	ServletContextListener	ServletContextEvent
	(2)	ServletContextAttributeListener	ServletContextAttributeEvent
sessie	(3)	HttpSessionListener	HttpSessionEvent
	(4)	HttpSessionAttributeListener	HttpSessionBindingEvent
request	(5)	ServletRequestListener	ServletRequestEvent
	(6)	ServletRequestAttributeListener	ServletRequestAttributeEvent

ServletContextListener

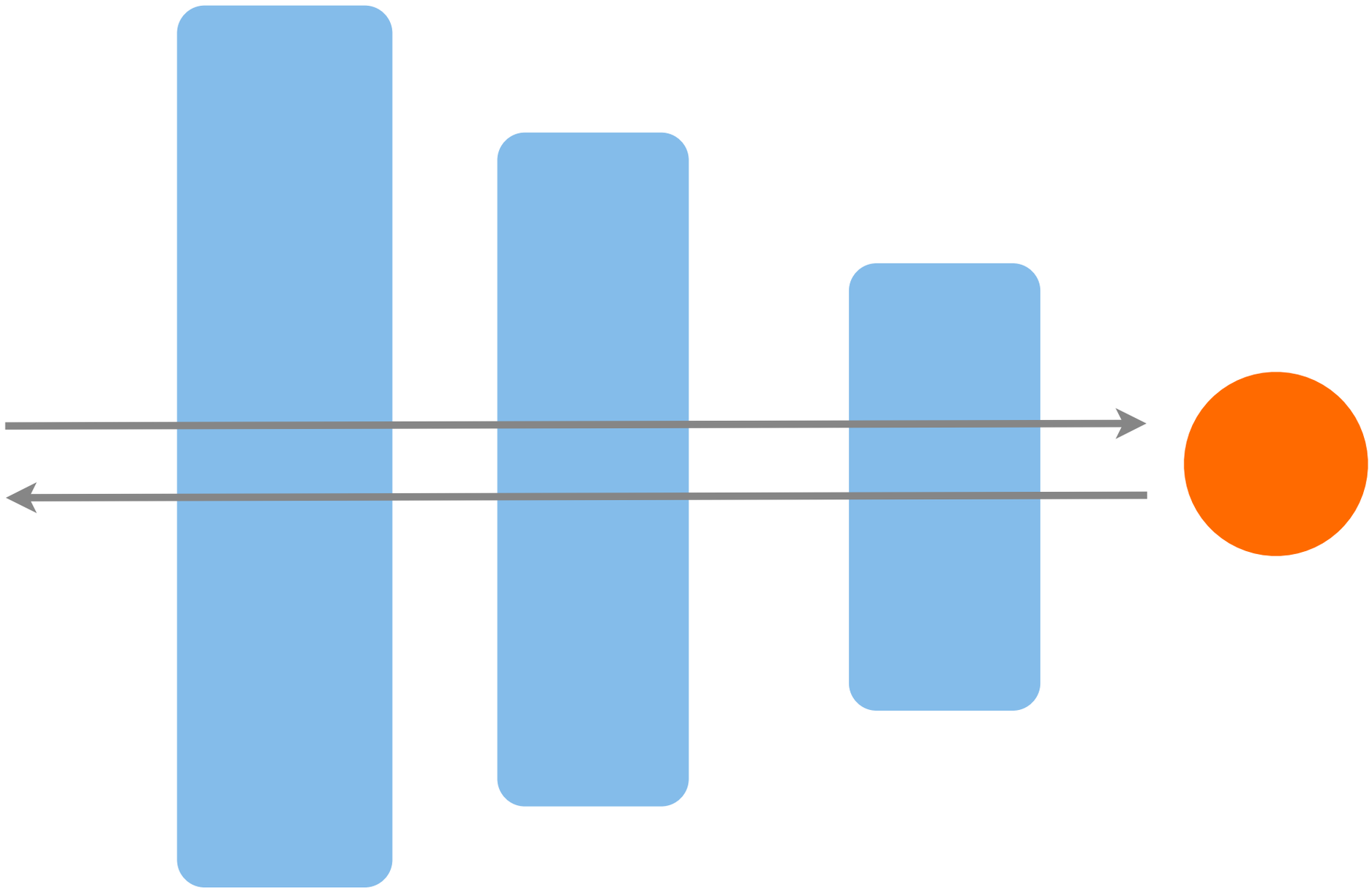
- Interface ServletContextListener
  - public void contextInitialized(ServletContextEvent e)
  - public void contextDestroyed(ServletContextEvent e)

VB: ServletContextListener maken

Filters

- Inspiratie: aspect-oriented programing
- Bijvoorbeeld:
  - Logging is onafhankelijk van toepassing
  - Toepassing is onafhankelijk van logging

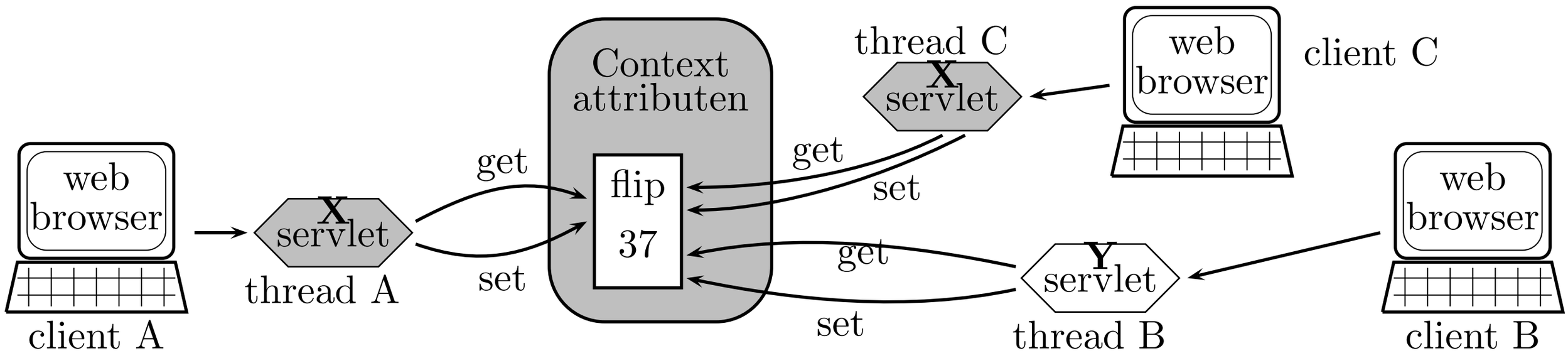
Filterkettingen



VB: Filter maken

Threads

- Elk request krijgt eigen thread
- Mogelijk probleem bij gedeeld geheugen
  - Applicatie niveau en sessie niveau



Oplossing (1)

```
public synchronized void doGet ( . . .  )  throws. . . {
    getServletContext().setAttribute(”flip”, ”37”);
    //  . . .
    out. println(getServletContext (). getAttribute(”flip”);
}
```

Werkt voor requests naar
dezelfde servlet,
maar niet efficient

Oplossing (2)

```
//  . . .
synchronized(getServletContext()) {
  getServletContext().setAttribute(”flip”, ”37”);
  //  . . .
  out. println(getServletContext (). getAttribute(”flip”);
}
//  . . .
```

Conclusie

- Communicatie via HTTP
- Beheer levenscyclus
- Multithreading
- Beveiliging
- Vertaling van JSPs