Java technológiák - 2. előadás

ANTAL Margit

Sapientia - EMTE

2010

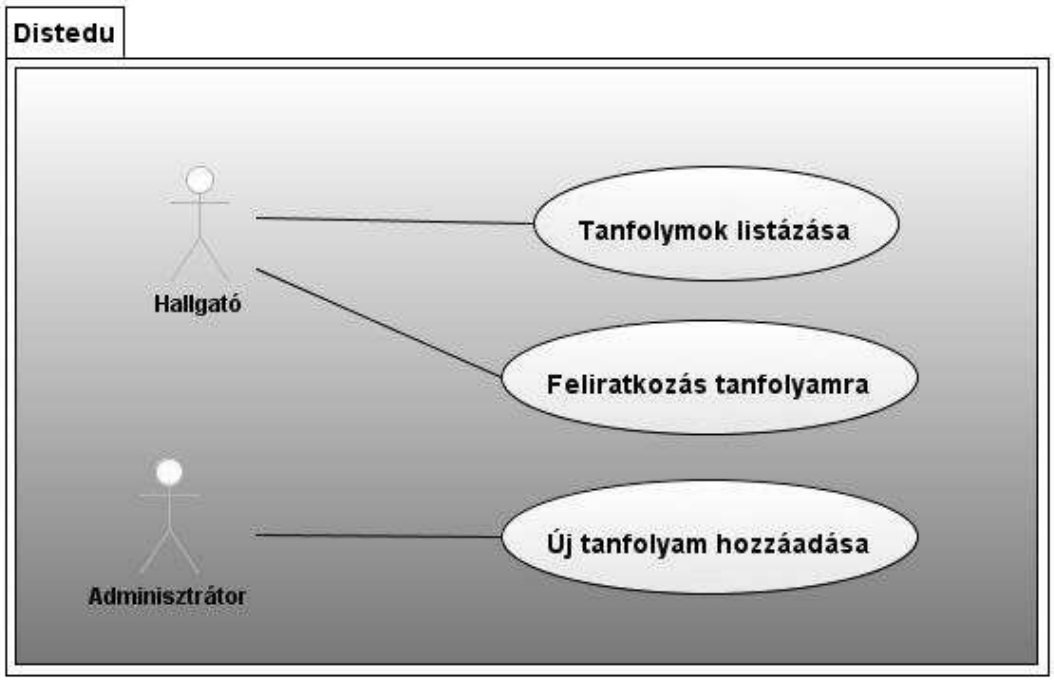
A 2. előadás célja

- Megjelenítési komponensek tervezése
- A HTTP protokoll leírása
- A Webkonténerek viselkedése
- Egyszerű HTTP szervlet készítése és telepítése

Megjelenítés komponenstípusok

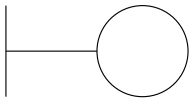
- Adatok megjelenítése: grafikonok, táblázatok
- Űrlapok
- Navigációs elemek: menük, hiperlinkek
- Információs képernyők: útbaigazítások, segítség, hibaüzenetek, dialógusdobozok

Esettanulmány - On-line tanfolyamok

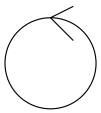


Az analízis modell komponensei

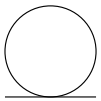
Boundary-Service-Entity



HATÁR BOUNDARY



SZOLGÁLTATÁS SERVICE



EGYED ENTITY

Az analízis modell komponensei (2)

Határ analízis kompenensek

- View - Megjelenítési
- Controller - Vezérlési

Megjelenítési komponensek

View components

- Adat-megjelenítési komponensek: grafikonok, táblázatok...
- Adatbeviteli űrlapok
- Navigációs elemek: menük, hiperlinkek,...
- Információs képernyők: utasítások, segítség képernyők...

Az első webalkalamazás szerkezete

```
__
  |
  |___index.html
  |
  |___list-courses.html
```

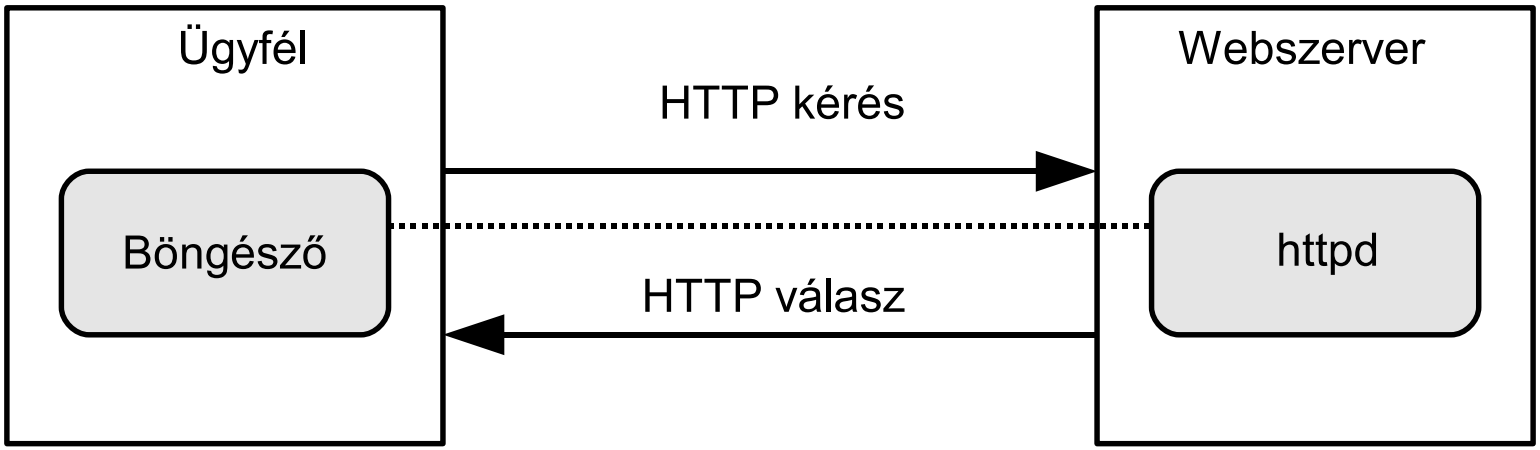
list-courses.html

```
<html>
<head>
    <title>Osszes tanfolyam </title>
</head>
<body>
<H3>Osszes tanfolyam</H3>
<ul>
    <li>J2SE</li>
    <li>J2EE</li>
    <li>C++</li>
</ul>
</body>
</html>
```

index.html

```
<html>
<H1>On line tanfolyamok</H1>
<body>
<ul>
    <H3>HALLGATO</H3>
    <ul>
        <li><a href="list-courses.html">
        Osszes tanfolyam</a></li>
        <li>Regisztracio</li>
    </ul>
    <H3>ADMINISZTRACIO<H3>
    <ul>
        <li>Uj tanfolyam felvitele</li>
    </ul>
</ul>
</body>
</html>
```

HTTP kliens-szerver architektúra



HTTP metódusok

restclient.jar!!!

HTTP metódus	Leírás
OPTIONS	A szerver által támogatott metódusok lekérdezése.
GET	Statikus erőforrás kérés az URL definiálásával.
HEAD	Ugyanaz mint a GET, de csak a fejlécét kéri le
POST	Hasonló a GET-hez, kivéve az üzenet testet. Jellemzően kulcs-érték párokat tartalmaz
PUT	Fájlok feltöltése egy megadott URL-re a szerveren
DELETE	URL-el megadott erőforrás törlése a szerveren

HTTP GET metódus

- Minden egyes hiperlinkre való kattintáskor ez történik.
- Minden egyes média típusú fájlra is hasonló kérést küld a böngésző
- HTTP kérés:

```
<HTTP metodus> <URL> <HTTP verzio>
```

- HTTP kérésre példa:

```
GET /list-courses.html HTTP/1.0
```

HTTP kérés fejléc elemek

- Accept - A kliens (ügyfél) által elfogadható médiatípusokat(MIME) adja meg

  ```
  Accept: image/gif, image/jpeg, text
  ```

- Accept-Language - Az ügyfél által elfogadott nyelvek fontossági sorrendben
- Host - A szerver nevét és a portszámot tartalmazza
- Referer - Annak az oldalnak a webcímét tartalmazza, ahonnan a kérés jött
- User-Agent - Információ a kliensszoftverről (böngésző, nyelv, op. rendszer)

HTTP válasz

A válasz szerkezete:

1. Státusz sor
2. Fejléc sorok
3. Üres sor
4. Üzenet blokkja

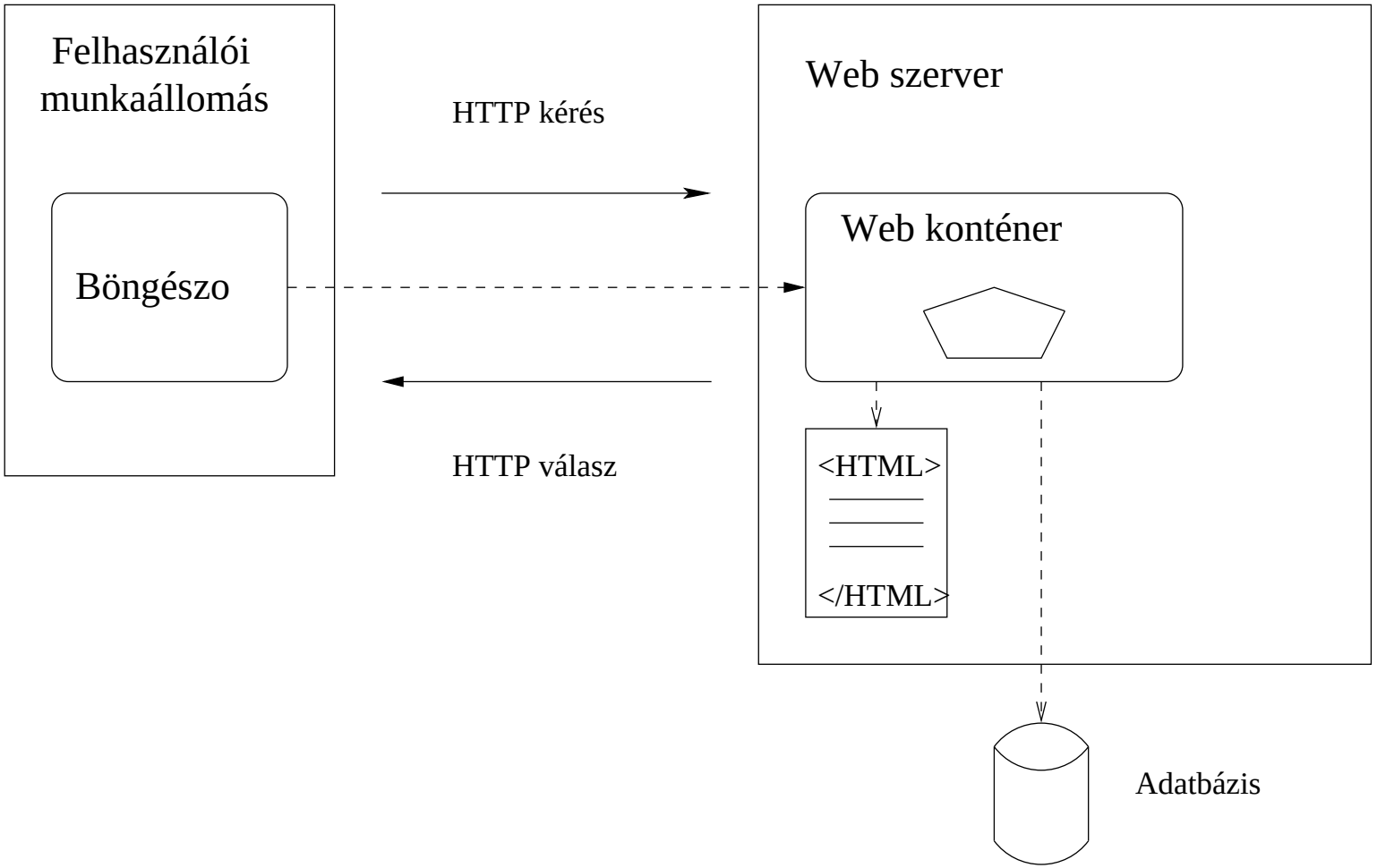
```
HTTP/1.0 200 OK
Content-Type: text/html
Date: Tue, 10 Apr. 2001 10:00:03 GMT
Server: Apache Tomcat/4.0-b1

<HTML>
...
</HTML>
```

HTTP válasz fejléc

- Content-Type: MIME típus
- Content-Length: A válasz hossza (bájtban)
- Server: A kiszolgáló típusa

Webkonténerek architektúrája



Szervletek

- A szervlet egy Java osztály, melynek feladata egy kérés kiszolgálása
- A szervlet impementálja a Servlet interfészt
- A szervlet életciklusát az a webkonténer kezeli, amelybe telepítve volt az illető szerver

A szervlet életciklusa

Ha egy kérés érkezik a szervlethez:

- Ha a szervletnek még nem létezik példánya, akkor a webkonténer: betölti az osztályt, példányosítja majd inicializálja (init metódus hívása)
- Meghívja a service metódusát
- Ha a konténernek el kell távolítania a szervletet, meghívja a destroy metódusát

FONTOS!!!

Az init és a destroy csak egyszer hívódik meg

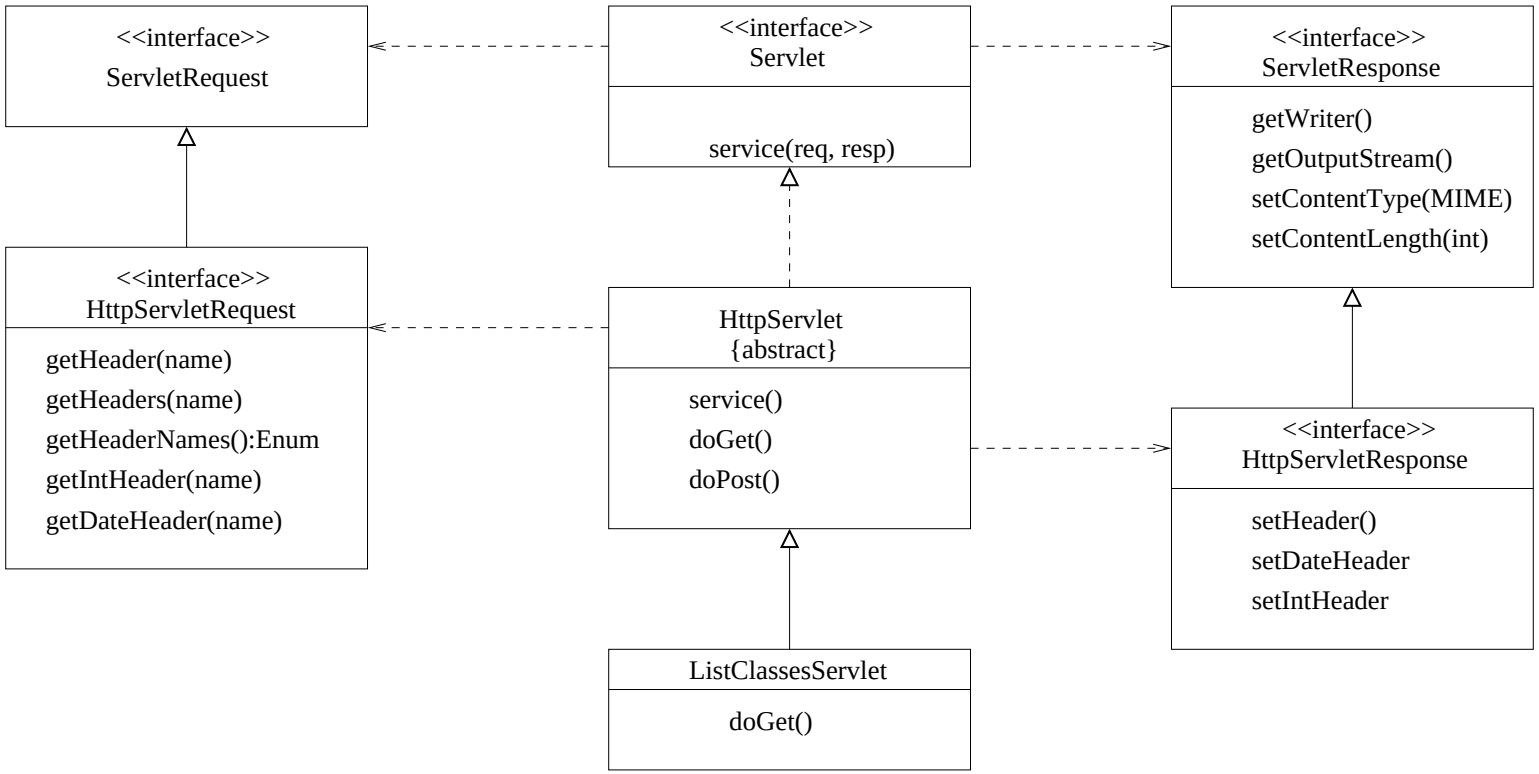
A service annyiszor hívódik ahány kérés érkezik az adott szervlethez

A service metódus

A service metódus

1. Két paramétere van: egy kérés és egy válasz objektum, amelyeket a webkonténer hoz létre
2. A kérés `HttpServletRequest` típusú
3. A válasz `HttpServletResponse` típusú

HttpServlet API



HttpServlet metódusok

HTTP metódus	HttpServlet metódus
OPTIONS	doOptions()
GET	doGet()
HEAD	doHead()
POST	doPost()
PUT	doPut()
DELETE	doDelete()
TRACE	doTrace()
CONNECT	doConnect()

HttpServletRequest metódusok

- `getHeaderNames()`: az összes név lekérése
- `getHeader()`: valamely névhez rendelt érték lekérése
- `getIntHeader()`: a névnek megfelelő érték átalakítása egész típusúvá
- `getDateHeader()`: a névnek megfelelő érték átalakítása dátum típusúvá

Példa

```
boolean xhtml= false;
String user=
       request.getHeader("User-Agent");
if( user.startsWith("Mozilla/5.0"))
  xhtml = true;
```

HttpServletResponse metódusok

- `setHeader()`
- `setIntHeader()`
- `setDateHeader()`
- `getOuputStream()`: bináris tartalom írása
- `getWriter()`: szöveges tartalom írása, pl. HTML

Példa

```
response.setContentType("text/html");
response.setHeader
      ("Cache-Control","no-cache");
```

Egyszerű HTTP szervlet készítése

- Elkészítjük a `Course` osztályt
- Elkészítjük a `ListCoursesServlet` osztályt

Model: Course.java I

```
package distedu.model;
public class Course{
  private String name;
  private String description;
  private double price;
  public Course(){
  }
  public Course( String name,
                 String description,
                 double price){
     ...
  }
  //get es set metodusok
  //minden attributumra
  public String toString(){ ... }
}
```

View: ListCoursesServlet.java I

```
package distedu.view;

import java.io.*;
import java.net.*;
import java.util.*;

import javax.servlet.*;
import javax.servlet.http.*;

public class ListCoursesServlet extends
             HttpServlet {
  private List<Course> courseList=null;
  private String[] courseNames  = {...};
  private double[] coursePrices = {...};

  protected void doGet(
```

View: ListCoursesServlet.java II

```
    HttpServletRequest request,
    HttpServletResponse response)
    throws ServletException, IOException {
      courseList = new LinkedList<Course>();

      //courseList feltoltese
      String pageTitle="Distance Courses";
      response.setContentType("text/html");
      PrintWriter out=response.getWriter();

      //HTML valasz
      out.println("<html>");
      out.println("<head>");
      out.println("<title>"+
        pageTitle+"</title>");
      out.println("</head>");
      out.println("<body>");
```

View: ListCoursesServlet.java III

```
        //Tablazat tartalma
        out.println("<ul>");
        Iterator items=courseList.iterator();
        while( items.hasNext()){
              out.println(
              "<li> "+ items.next()+ "</li>");
        }
        out.println("</ul>");
        out.println("</body>");
        out.println("</html>");
    }
    ...
}
```

Szervlet definíció: `web.xml`

- Szervlet definíció

```
<servlet>
  <servlet-name>
    ListCoursesServlet
  </servlet-name>
  <servlet-class>
    distedu.view.ListCoursesServlet
  </servlet-class>
</servlet>
```

- Szervlet leképzés (mapping)

```
<servlet-mapping>
  <servlet-name>
    ListCoursesServlet
  </servlet-name>
  <url-pattern>
    /list-courses.view
  </url-pattern>
</servlet-mapping>
```

WAR (Web ARchive) fájlok

- Helyettesíti a kézi telepítést
- Egy JAR fájl, amely egy web alkalmazás struktúráját tartalmazza
  - WAR létrehozása: `jar cvf distedu.war distedu`
  - WAR kicsomagolása: `jar xvf distedu.war`

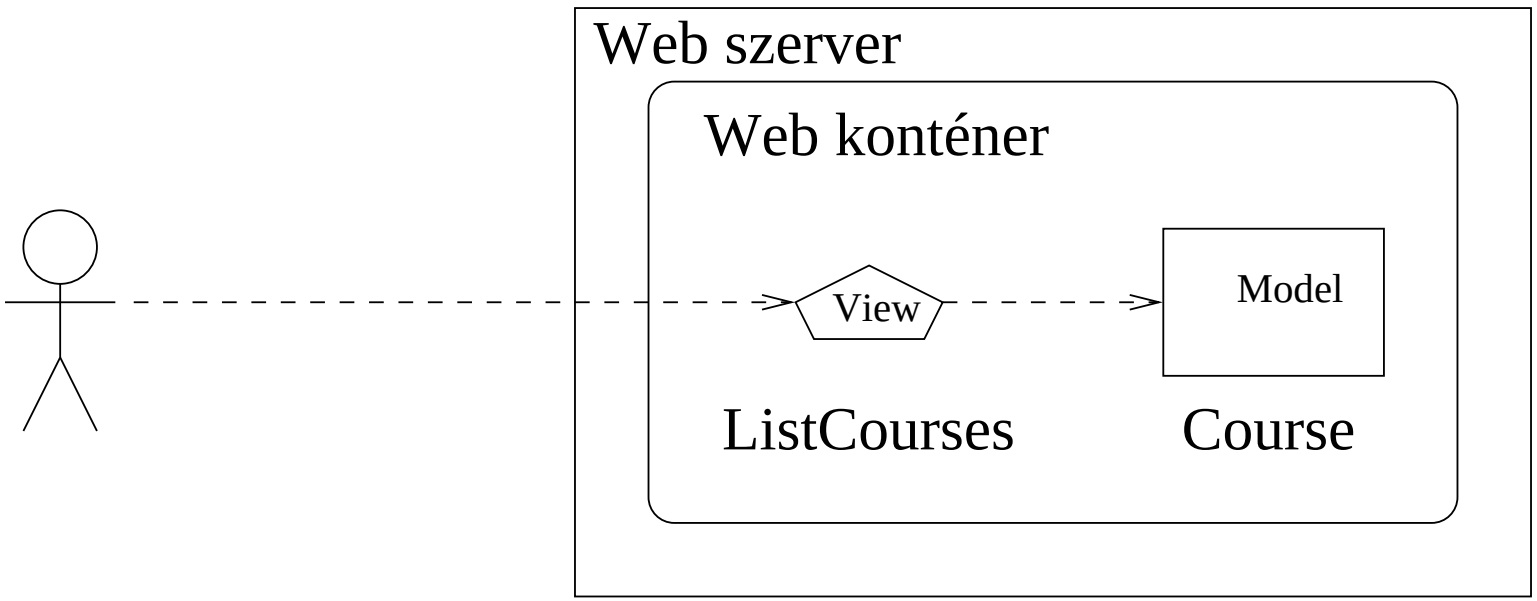
Szervlet aktiválása (1) I

- Felhasználó-böngésző: `http://localhost:8080/distedu/`
- Webkonténer:`index.html` fájl

```
<h3>Distance Courses</h3>
<ul>
  <li>
    <a href='list-courses.view'>
      List all courses
    </a>
  </li>
  <li>Register for a course</li>
</ul>
```

Szervlet aktiválása (2)

Kiválasztva a ”List all courses” listapontot aktiválódik a szervlet: http://localhost:8080/distedu/list-courses.view



Összefoglalás

- A megjelenítési komponensek statikus tartalmat jeleníthetnek meg, illetve űrlap megjelenítésére is használhatók
- A HTTP protokoll biztosítja a megjelenítés lekérését
- A webkonténer fogadja a HTTP kérést és aktiválja a megfelelő szervletet
- A szervelet a `doGet()` metódus végrehajtásával válaszol a kérésre
- A kérésben található adatokat a webkonténer által biztosított `request` objektumtól lehet lekérni
- A megjelenítést úgy állítjuk elő, hogy a `response` objektumból kinyert kimeneti adatfolyamba adatokat írunk