

## Stateless Session Bean

Készítsünk egy stateless session bean-t, amellyel összeadhatunk két számot.

### Hozzunk létre egy Dynamic Web projectet

File → New → Other → ... itt a következőket kell választani: Web → Dynamic Web Project

A megjelenő párbeszédablakban adjuk meg a projekt nevét, ez a példában StatelessExample.

**New Dynamic Web Project**

**Dynamic Web Project**  
Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name: StatelessExample

Project location  
 Use default location  
Location: C:\Users\Simon\workspaceEE\StatelessExample Browse...

Target runtime  
JBoss 7.1 Runtime New Runtime...

Dynamic web module version  
3.0

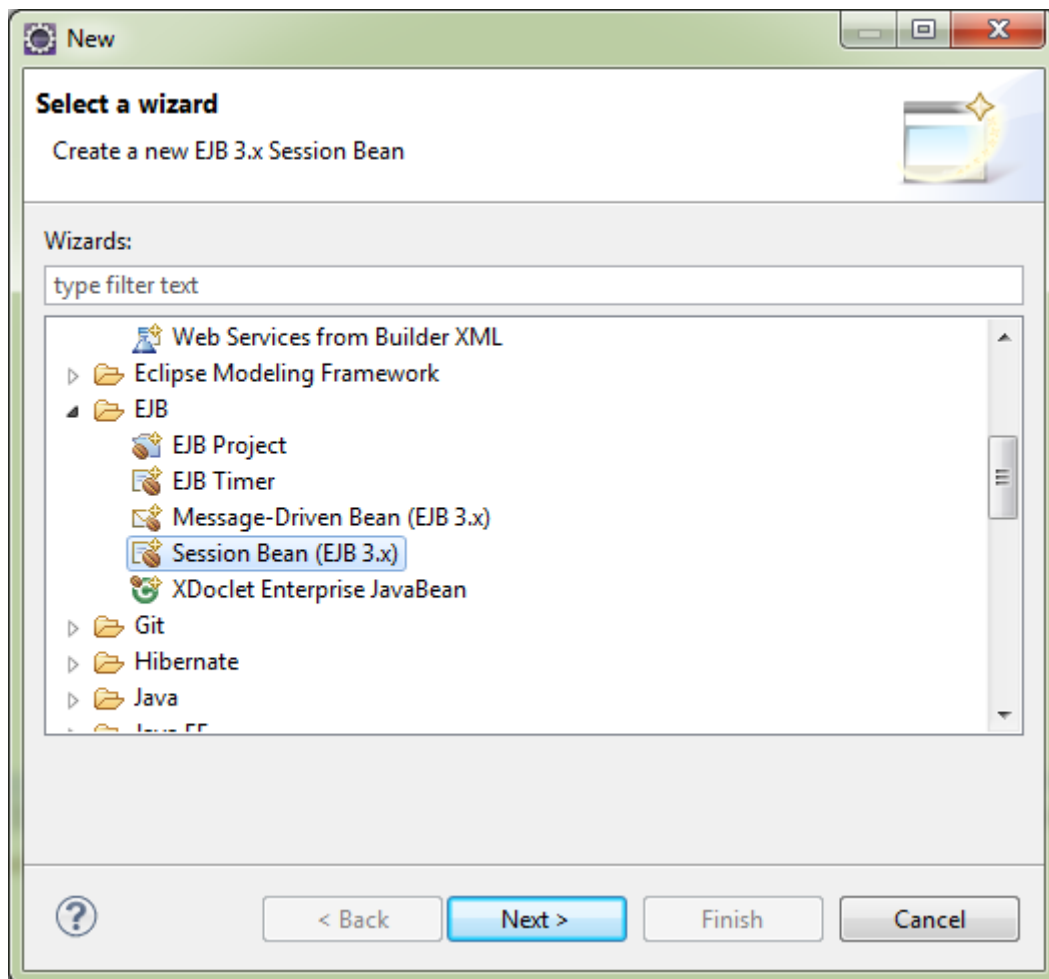
Configuration  
Default Configuration for JBoss 7.1 Runtime Modify...  
A good starting point for working with JBoss 7.1 Runtime runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership  
 Add project to an EAR  
EAR project name: EAR New Project...

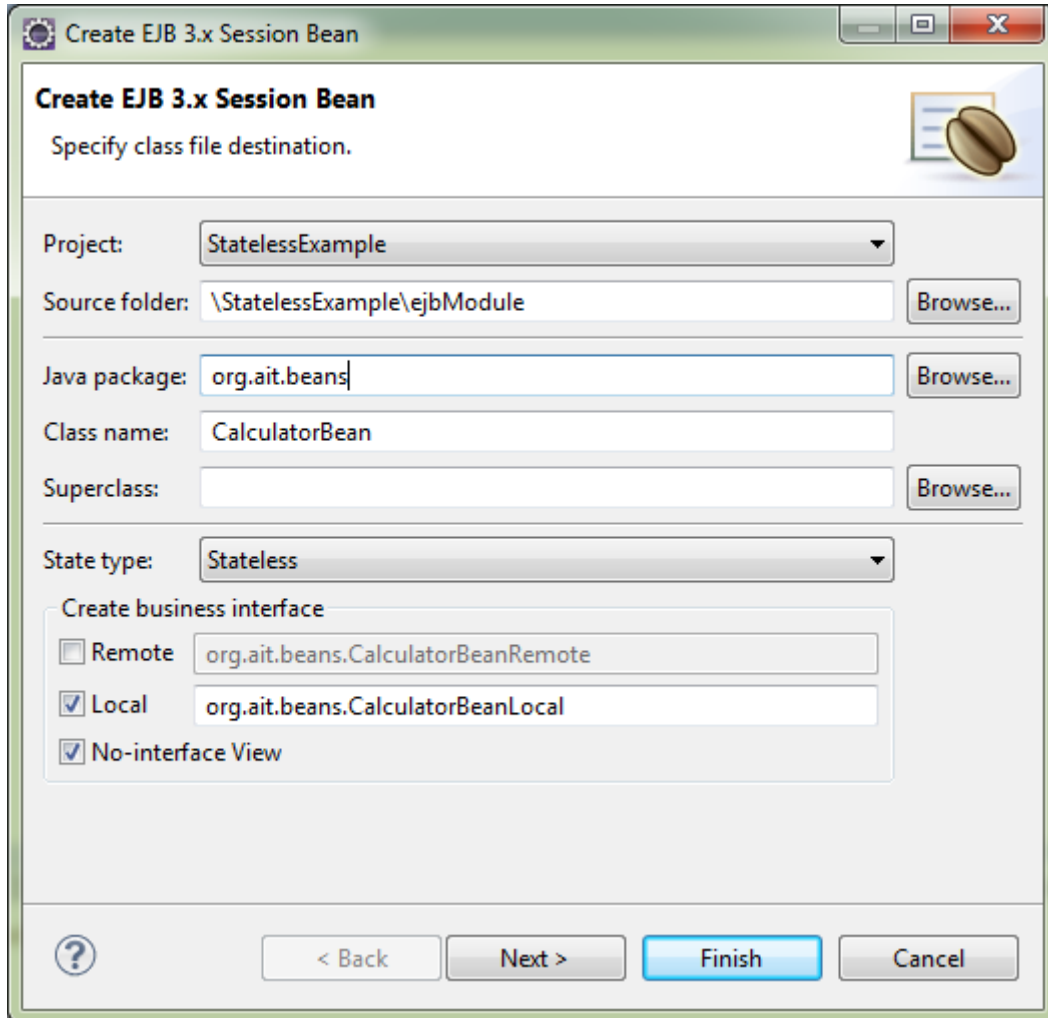
Working sets  
 Add project to working sets  
Working sets: Select...

? < Back Next > Finish Cancel

Új Session Bean létrehozását a következő menüpontban lehet megtenni:  
File → New → Other ... itt a következőket kell választani: EJB → Session Bean



A megjelenő párbeszédablakban a bean paraméterei állíthatók be.  
A példában a csomag név org.ait.beans, az osztály neve pedig CalculatorBean.  
Az állapotok közül válasszuk a *Stateless*-t.  
Az üzleti interfészek részénél jelöljük be a Local interfész létrehozását.  
Majd kattintsunk a Next gombra, majd a Finish-re.



**Create EJB 3.x Session Bean**  
Specify class file destination.

Project: StatelessExample

Source folder: \StatelessExample\ejbModule

Java package: org.ait.beans

Class name: CalculatorBean

Superclass:

State type: Stateless

Create business interface

Remote org.ait.beans.CalculatorBeanRemote

Local org.ait.beans.CalculatorBeanLocal

No-interface View

## Forráskód:

A *CalculatorBeanLocal* interfész törzsében hozzuk létre az összeadó és kivonó metódusok prototípusát.

```
package org.a.it.beans;

import javax.ejb.Local;

@Local
public interface CalculatorBeanLocal {
    public int add(int x, int y);
    public int sub(int x, int y);
}
```

A *CalculatorBean*-ben írjuk meg az interfészben magadott metódusok törzsét.

```
package org.a.it.beans;

import javax.ejb.LocalBean;
import javax.ejb.Stateless;

/**
 * Session Bean implementation class CalculatorBean
 */
@Stateless
@LocalBean
public class CalculatorBean implements CalculatorBeanLocal {

    public CalculatorBean() {

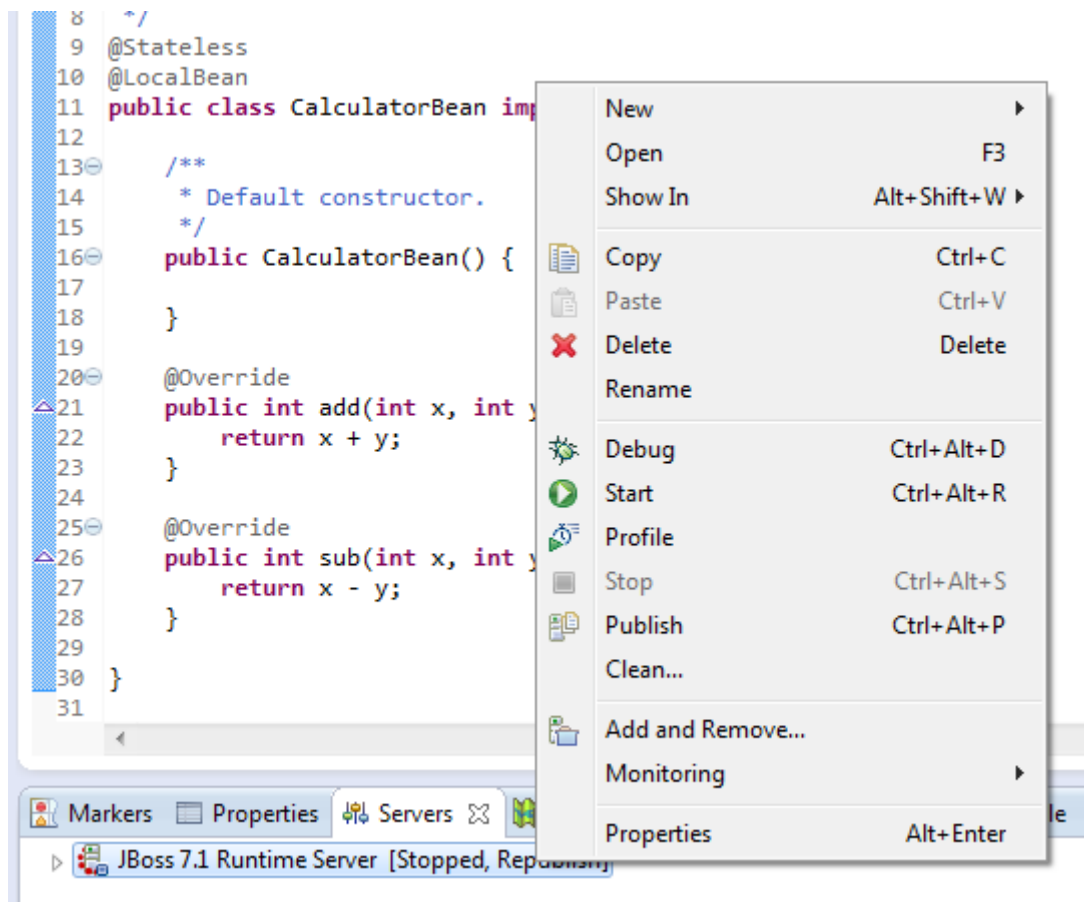
    }

    @Override
    public int add(int x, int y) {
        return x + y;
    }

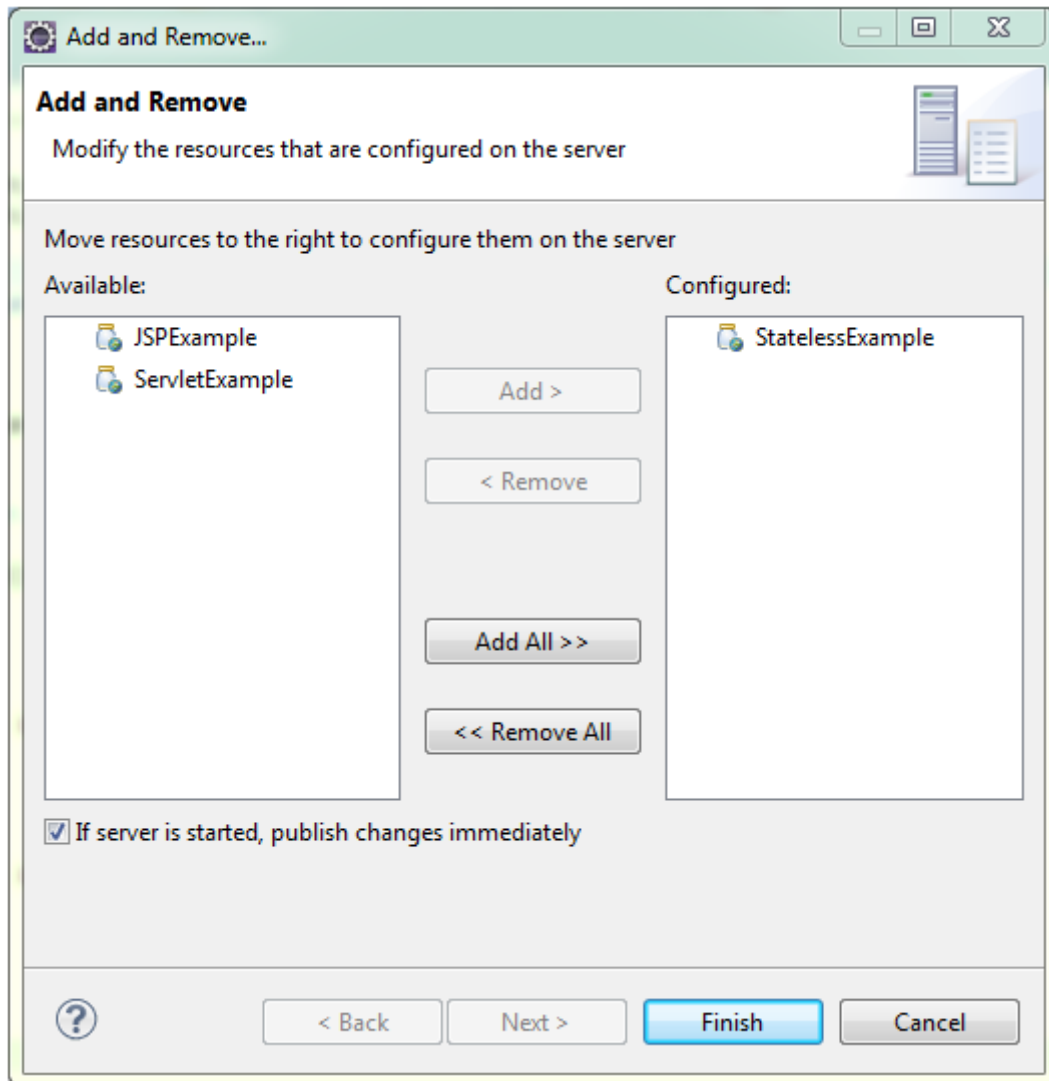
    @Override
    public int sub(int x, int y) {
        return x - y;
    }
}
```

## Bean hozzáadás szerverthez

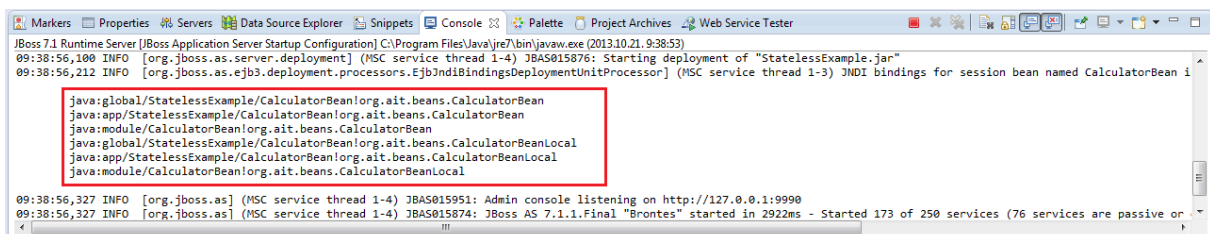
A *Server* fülön klikkeljünk jobb gombbal a szervertre, amelyikhez hozzá akarjuk adni a projektünket.



Itt válasszuk az *Add and Remove...* pontot. Majd a megjelenő ablakban adjuk hozzá a *StatelessExample* projektet.



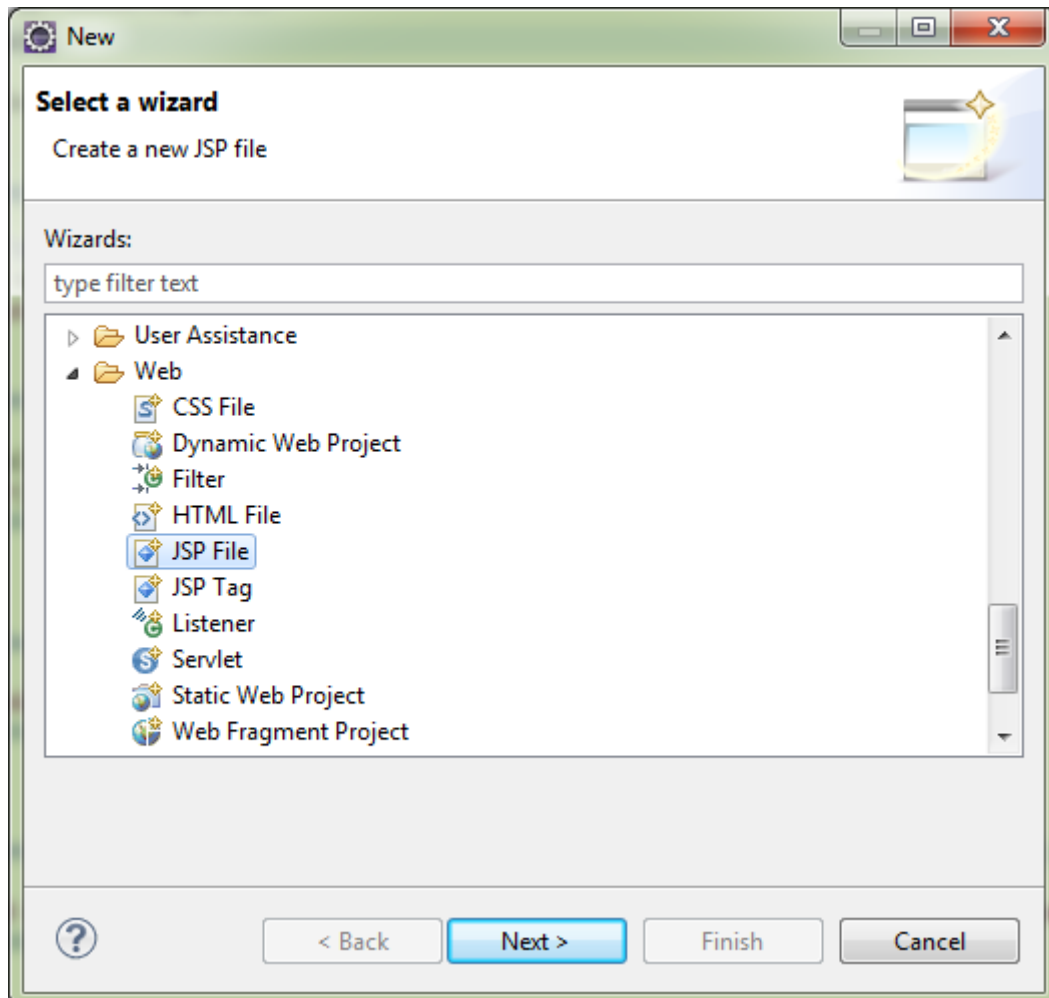
Ezután újra kattinkeljünk jobb gombbal a szerverre és válasszuk a *Start* pontot. Ezzel elindul a szerver és betölti a bean-ünket.



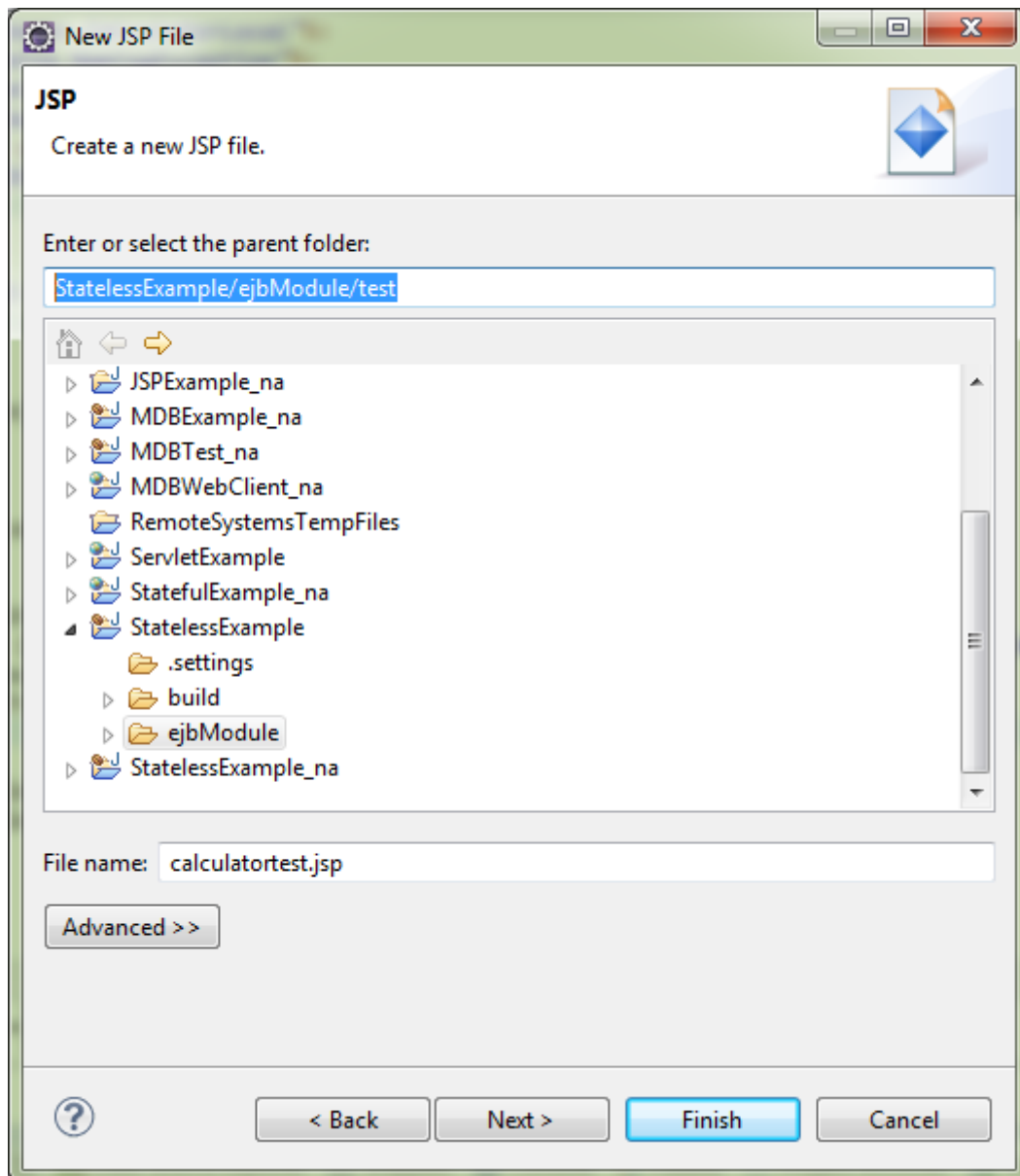
A konzolban megjelenő bean elérési névre szükség van a bean meghívása kor.

## Teszt JSP oldal létrehozása

JSP oldal létrehozásához a menüben válasszuk a New → Other... pontot, majd a Web → JSP File pontot.

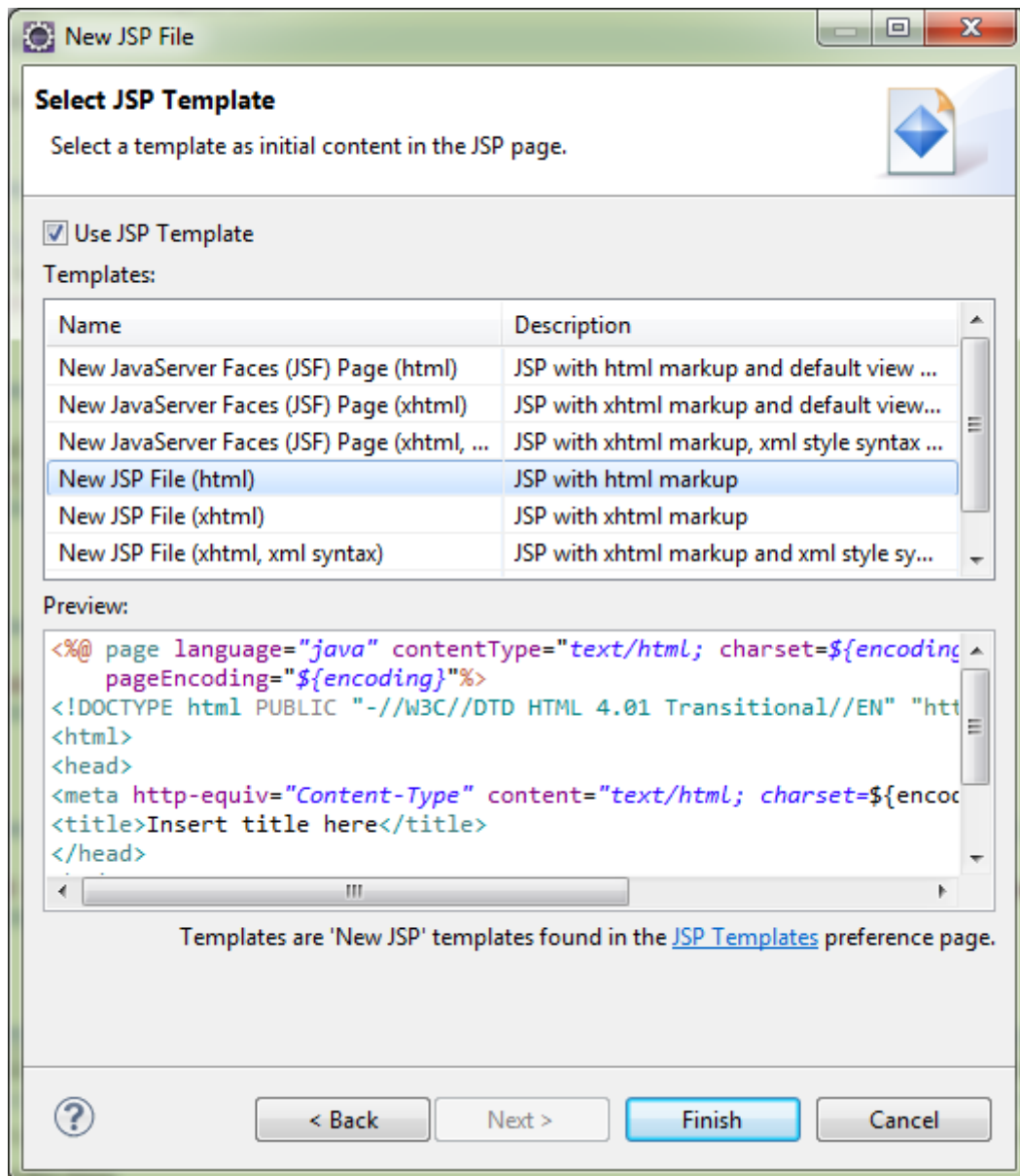


A megjelenő párbeszédablakban adjuk meg szülő könyvtárnak a *StatelessExample/ejbModule/test*-t és a jsp fájl nevét, ez legyen *calulatorctest.jsp*.





A *Next* gombra való kattintás után megjelenő ablakban válasszuk a New JSP File (html) templétet.



A *calculatorTest.jsp* törzse a következő legyen:

```
<%@page import="javax.naming.InitialContext"%>
<%@page import="javax.naming.NamingException"%>
<%@page import="org.ait.beans.CalculatorBeanLocal"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>Insert title here</title>
</head>
<body>
    <%
        InitialContext ctx;
        CalculatorBeanLocal calculator = null;

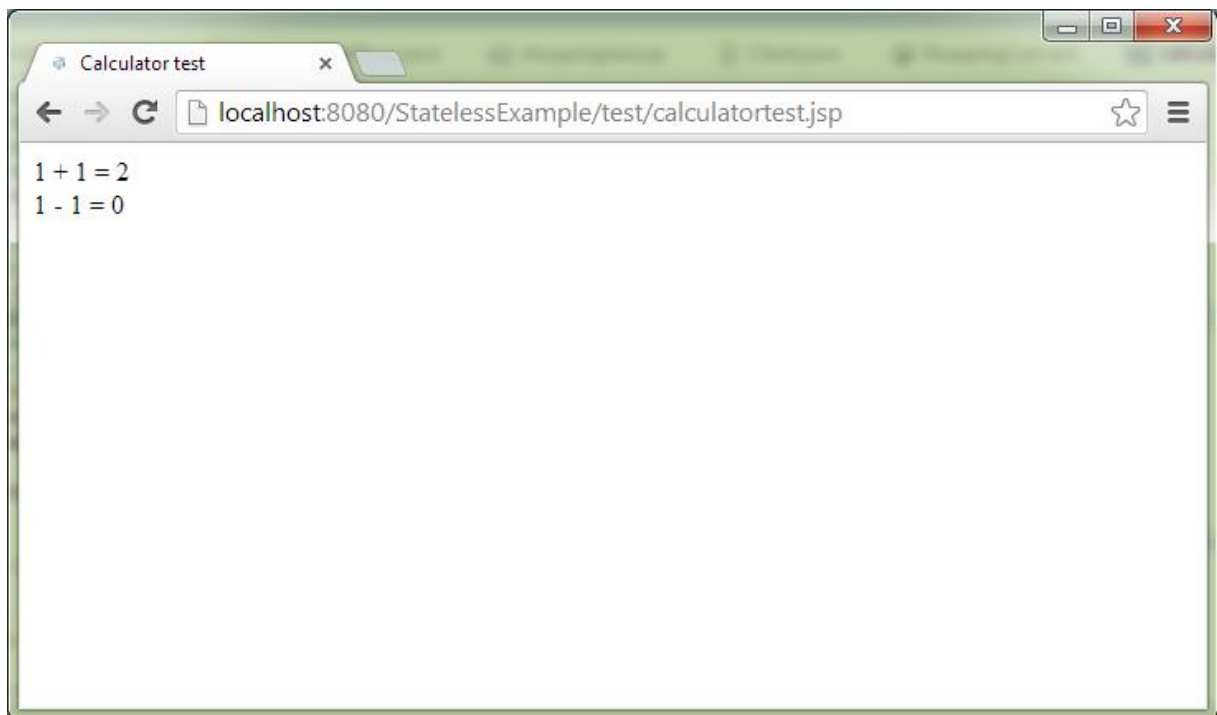
        try {
            ctx = new InitialContext();
            calculator = (CalculatorBeanLocal)
ctx.lookup("java:app/StatelessExampleDinWeb/CalculatorBean!org.ait.beans.CalculatorBeanLocal");
        } catch (NamingException e) {
            e.printStackTrace();
        }
    %>

    1 + 1 = <%= calculator.add(1, 1) %>
    <br>
    1 - 1 = <%= calculator.sub(1, 1) %>

</body>
</html>
```

A *Context lookup* metódusának paraméterében a bean elérési útját kell megadni, ami annak indításakor megjelent a konzolban.

Böngészőben megnézve az oldalt a következő eredményt kapjuk:



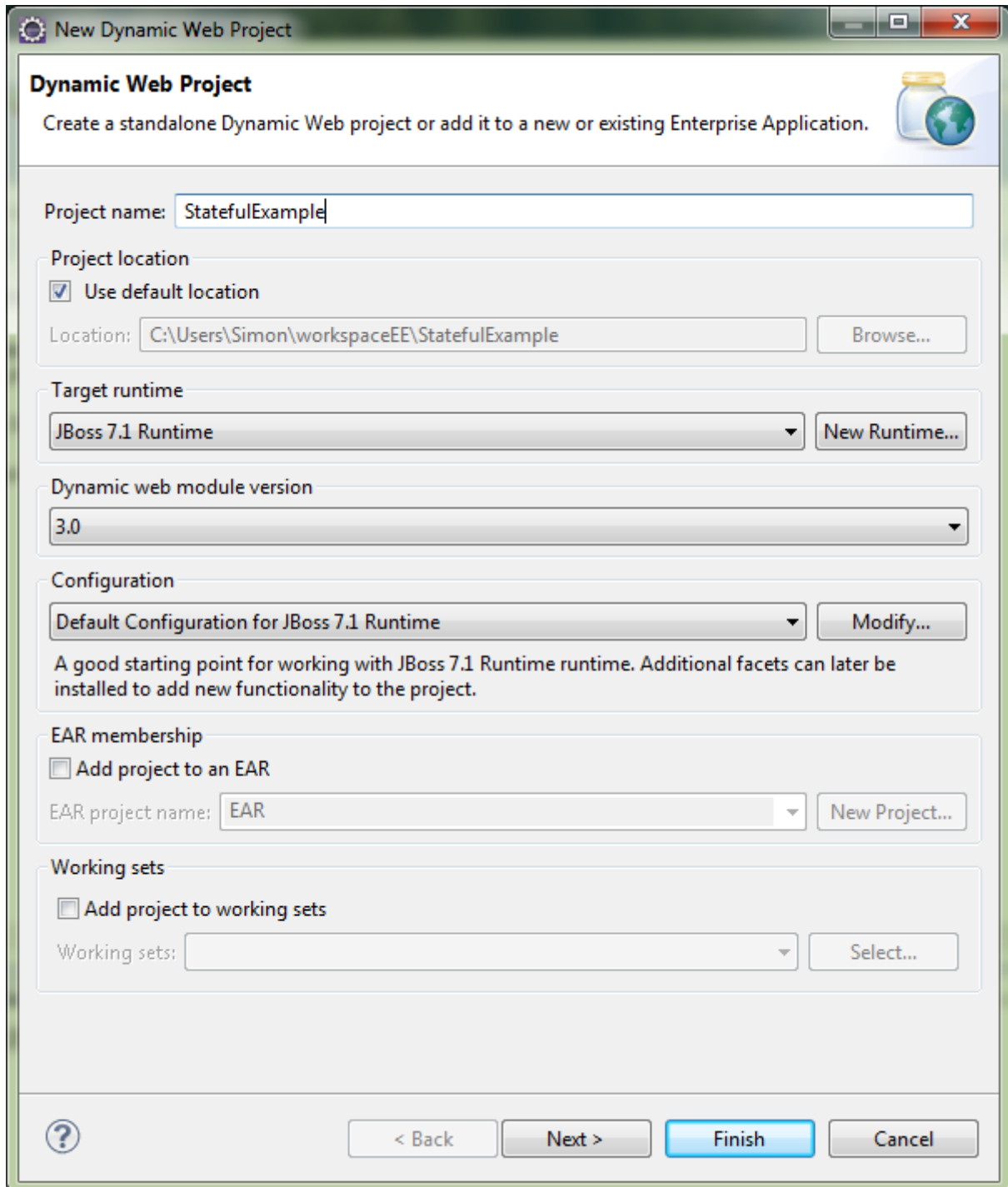
## Stateful Session Bean

Készítsünk egy stateless session bean-t, amely egy bevásárló kosarat modellez.

### Hozzunk létre egy Dynamic Web projectet

File → New → Other → ... itt a következőket kell választani: Web → Dynamic Web Project

A megjelenő párbeszédablakban adjuk meg a projekt nevét, ez a példában StatefulExample.



**New Dynamic Web Project**

Create a standalone Dynamic Web project or add it to a new or existing Enterprise Application.

Project name:

Project location

Use default location

Location:

Target runtime

Dynamic web module version

Configuration

A good starting point for working with JBoss 7.1 Runtime runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

Add project to an EAR

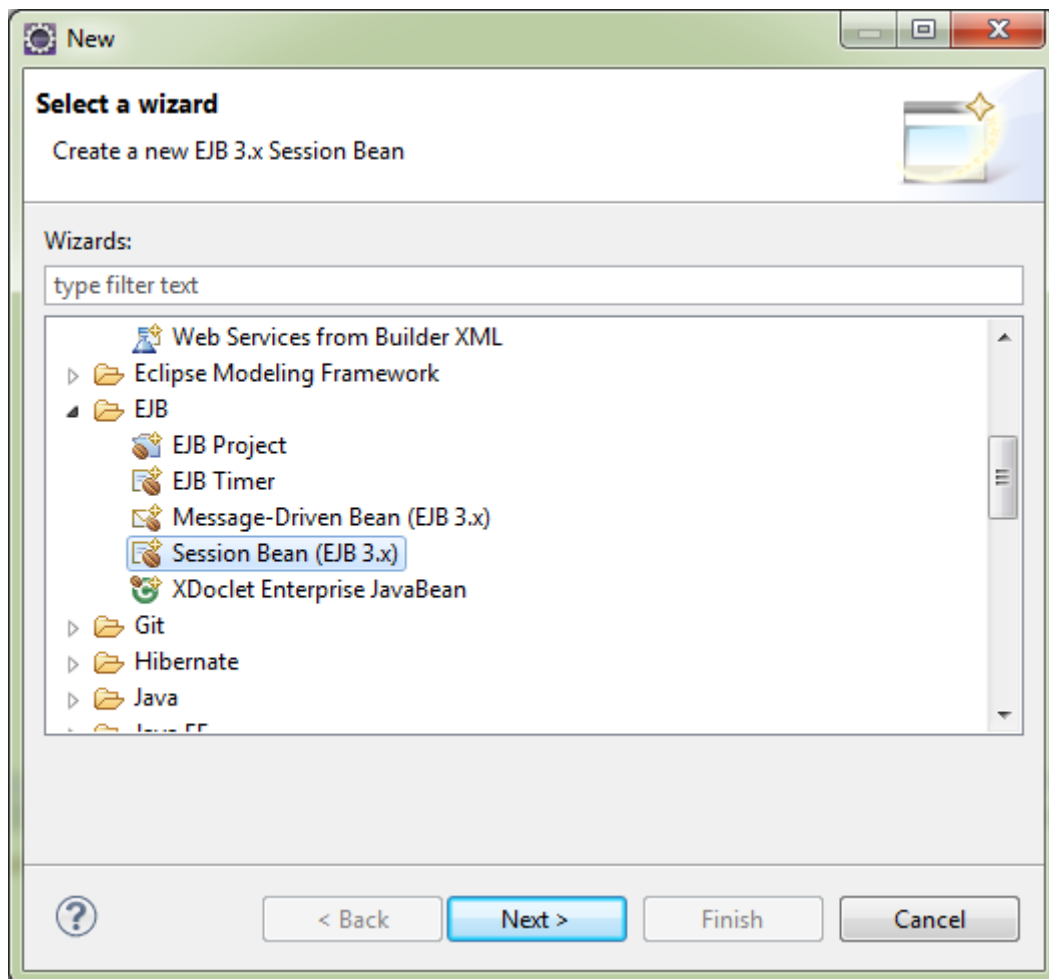
EAR project name:

Working sets

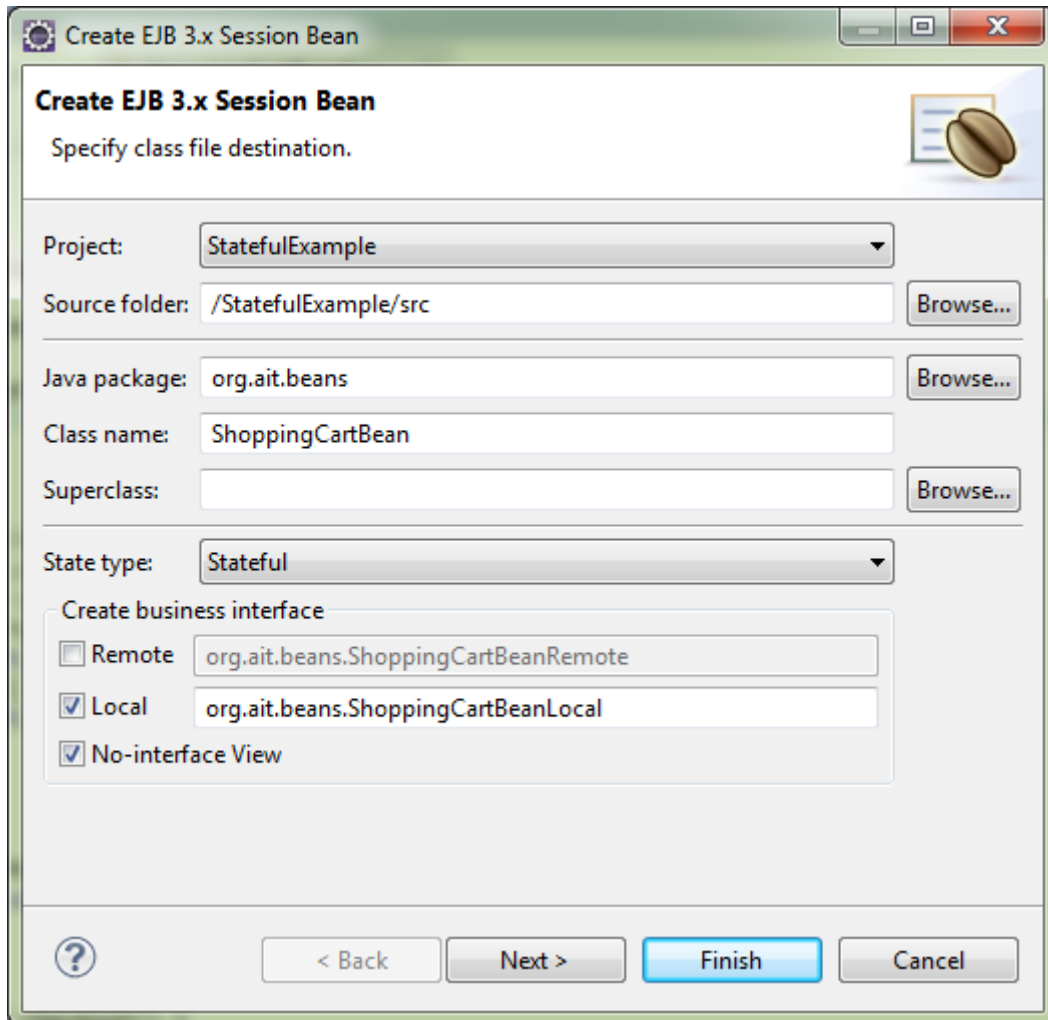
Add project to working sets

Working sets:

Új Session Bean létrehozását a következő menüpontban lehet megtenni:  
File → New → Other ... itt a következőket kell választani: EJB → Session Bean



A megjelenő párbeszédablakban a bean paramétereit állíthatók be.  
A példában a csomag név org.ait.beans, az osztály neve pedig ShoppingCartBean.  
Az állapotok közül válasszuk a *Stateful*-t.  
Az üzleti interfészek részénél jelöljük be a Local interfész létrehozását.  
Majd kattintsunk a Next gombra, majd a Finish-re.



**Create EJB 3.x Session Bean**  
Specify class file destination.

Project: StatefulExample

Source folder: /StatefulExample/src

Java package: org.ait.beans

Class name: ShoppingCartBean

Superclass:

State type: Stateful

Create business interface

Remote org.ait.beans.ShoppingCartBeanRemote

Local org.ait.beans.ShoppingCartBeanLocal

No-interface View

## Forráskód:

A *ShoppingCartBeanLocal* interfész törzse legyen a következő:

```
package org.ait.beans;

import java.util.HashMap;

import javax.ejb.Local;
import javax.ejb.Remove;

@Local
public interface ShoppingCartBeanLocal {

    public void buy(String product, int quality);
    public HashMap<String, Integer> getCartContent();
    @Remove
    public void checkout();
}
```

A *ShoppingCartBean* törzse pedig a következő:

```
package org.ait.beans;

import java.util.HashMap;

import javax.ejb.LocalBean;
import javax.ejb.Remove;
import javax.ejb.Stateful;

/**
 * Session Bean implementation class ShopingCartBean
 */
@Stateful
@LocalBean
public class ShoppingCartBean implements ShoppingCartBeanLocal {

    private HashMap<String, Integer> cart = new HashMap<String, Integer>();

    public ShoppingCartBean() {

    }

    @Override
    public void buy(String product, int quality) {
        if(cart.containsKey(product)) {
            int currqt = cart.get(product);
            currqt += quality;
            cart.put(product, currqt);
        } else {
            cart.put(product, quality);
        }
    }

    @Override
    public HashMap<String, Integer> getCartContent() {
        return cart;
    }
}
```

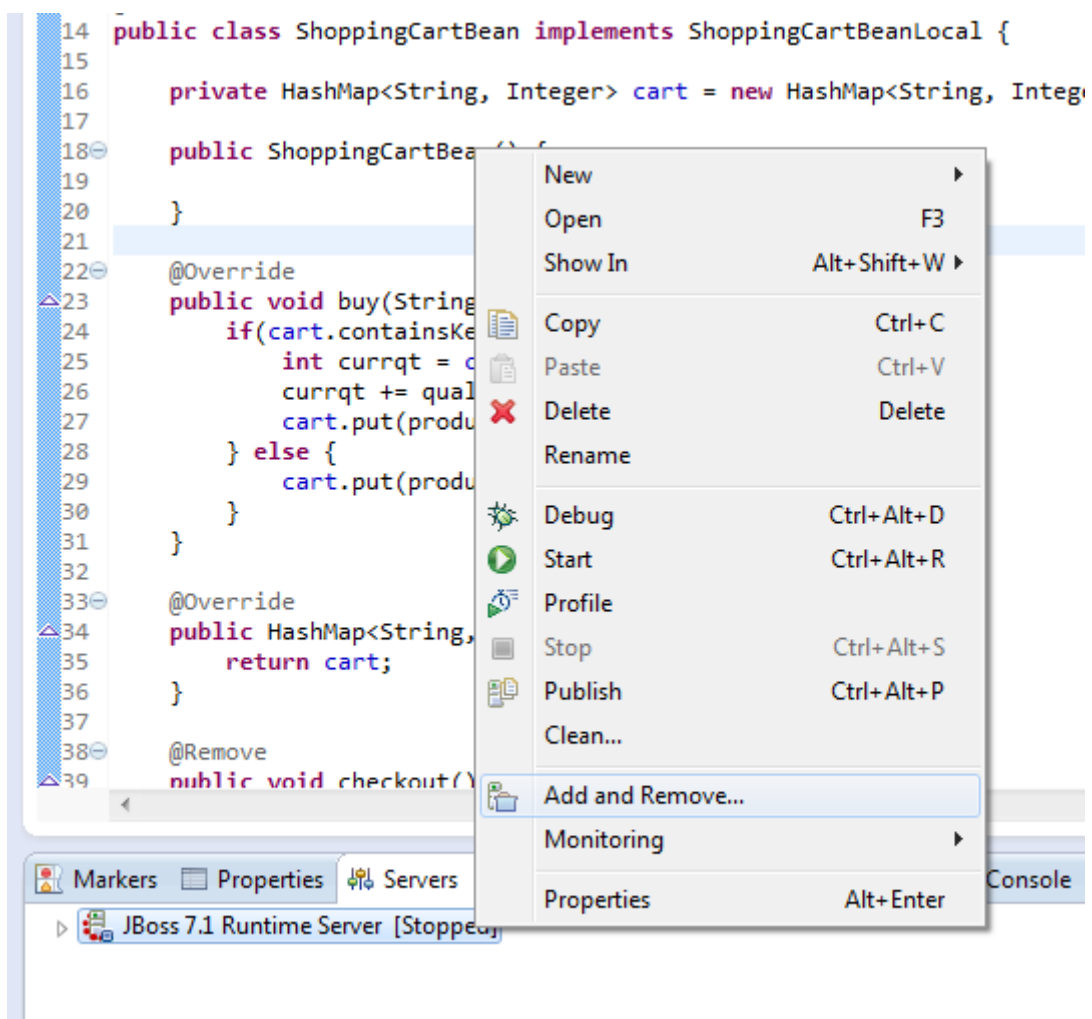
```

@Remove
public void checkout() {
    System.out.println("To be implemented");
}
}

```

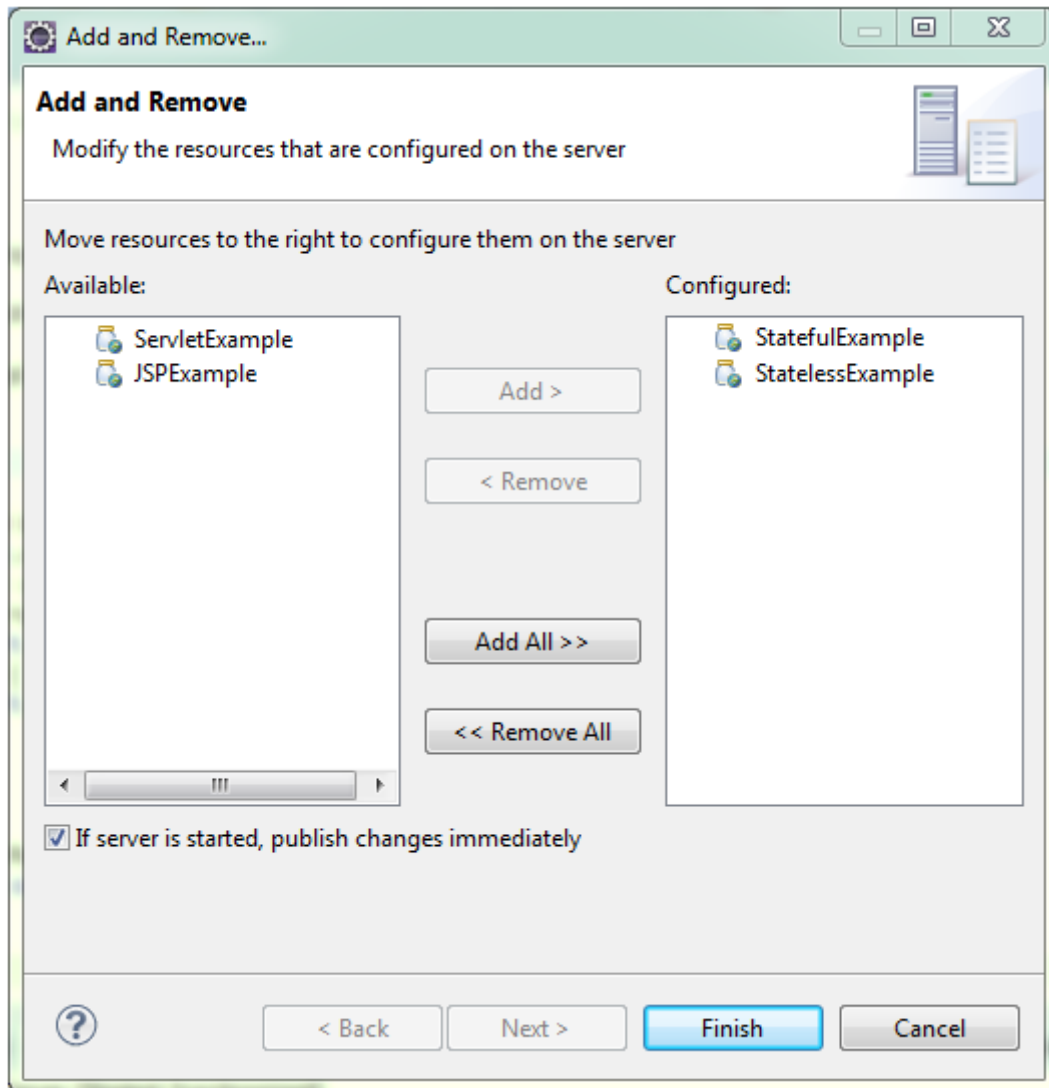
## Bean hozzáadás szerverthez

A *Server* fülön klikkeljünk jobb gombbal a szerverre, amelyikhez hozzá akarjuk adni a projektünket.

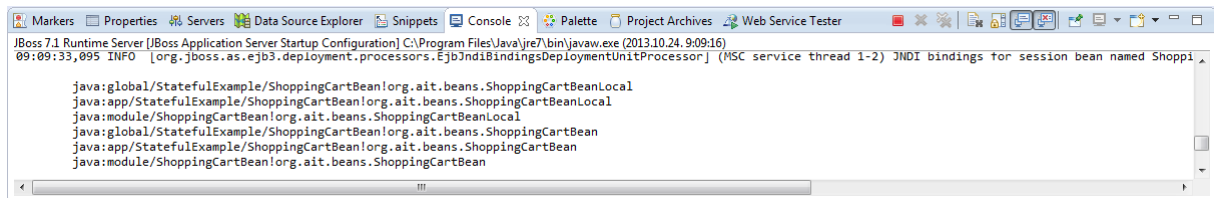




Itt válasszuk az *Add and Remove...* pontot. Majd a megjelenő ablakban adjuk hozzá a *StatefulExample* projektet.



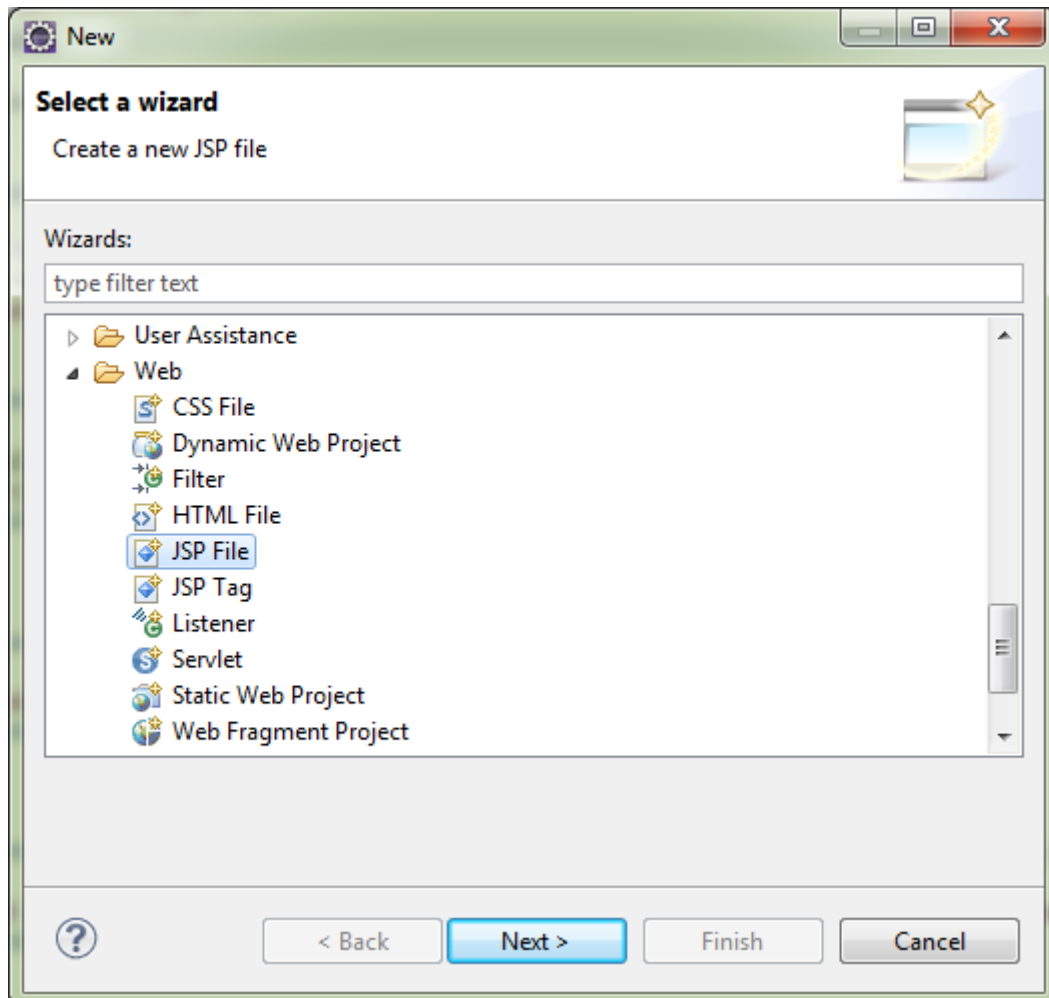
Ezután újra klikkeljünk jobb gombbal a szerverre és válasszuk a *Start* pontot. Ezzel elindul a szerver és betölti a bean-ünket.



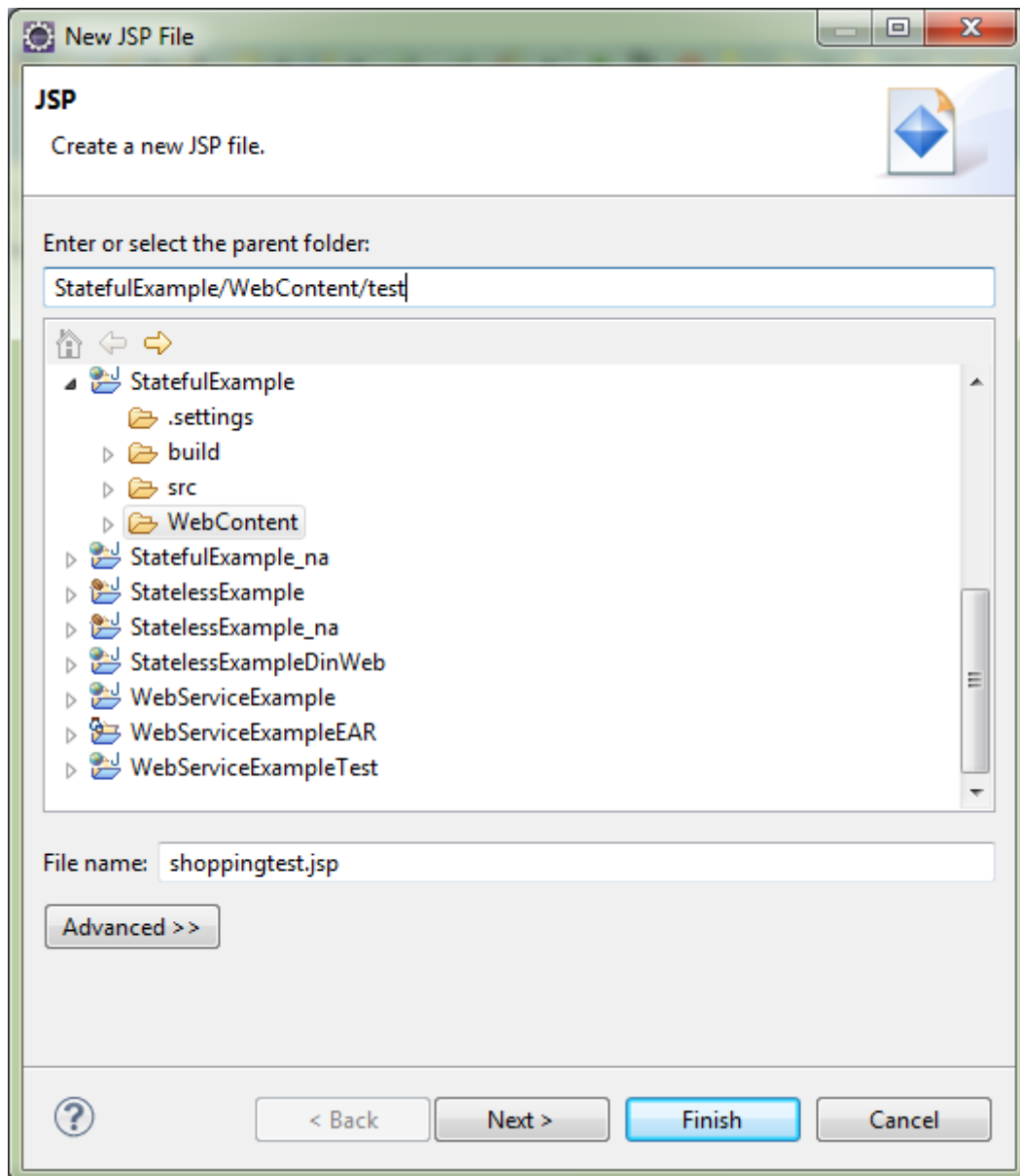
A konzolban megjelenő bean elérési névre szükség van a bean meghívása kor.

## Teszt JSP oldal létrehozása

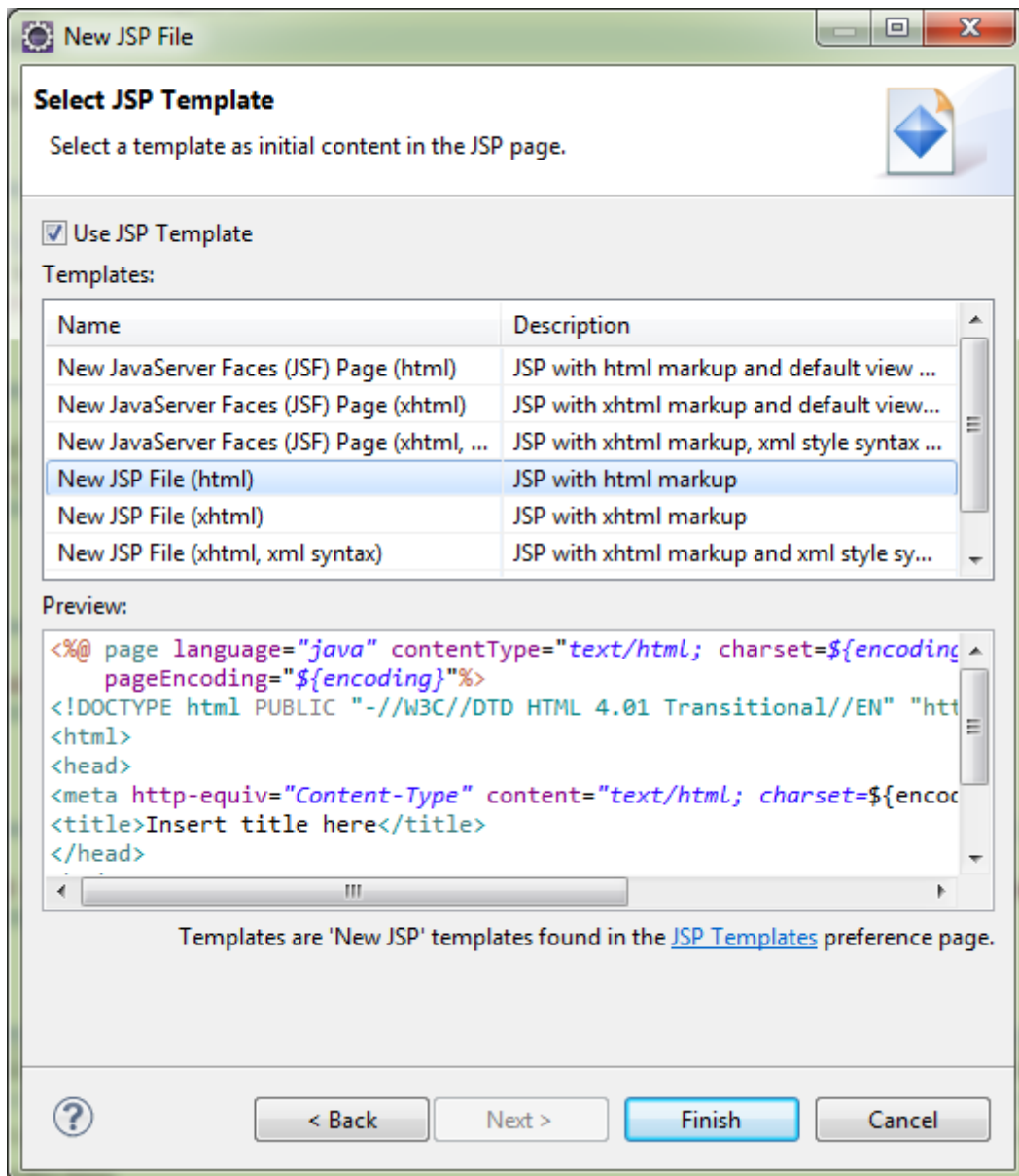
JSP oldal létrehozásához a menüben válasszuk a New → Other... pontot, majd a Web → JSP File pontot.



A megjelenő párbeszédablakban adjuk meg szülő könyvtárnak a *StatefulExample/ejbModule/test* -t és a jsp fájl nevét, ez legyen *shoppingtest.jsp*.



A *Next* gombra való kattintás után megjelenő ablakban válasszuk a New JSP File (html) templétet.



A *shoppingtest.jsp* törzse a következő legyen:

```
<%@page import="javax.naming.InitialContext"%>
<%@page import="javax.naming.NamingException"%>
<%@page import="org.ait.beans.ShoppingCartBeanLocal"%>
<%@page import="java.util.HashMap"%>
<%@ page language="java" contentType="text/html; charset=ISO-8859-1"
    pageEncoding="ISO-8859-1"%>
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
"http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-1">
<title>ShoppingCart test</title>
</head>
<body>
    <%
        InitialContext ctx = new InitialContext();
        ShoppingCartBeanLocal cart = (ShoppingCartBeanLocal)
ctx.lookup("java:app/StatefulExample/ShoppingCartBean!org.ait.beans.ShoppingCartBe
anLocal");
    %>
    Buying 1 memory stick <br>
    <%
        cart.buy("Memory stick", 1);
    %>
    Buying another memory stick <br>
    <%
        cart.buy("Memory stick", 1);
    %>
    Buying a laptop <br>
    <%
        cart.buy("Laptop", 1);
    %>

    <br>
    Print cart: <br>
    <%
        HashMap<String, Integer> fullCart = cart.getCartContent();
        for(String product : fullCart.keySet()) {
    %>
            <%= (String)(fullCart.get(product) + "    " + product) %> <br>
    <%
        }
    %>

    <br>
    Checkout <br>
    <%
        cart.checkout();
    %>

    <br>
    Should throw an object not found exception by invoking on cart after @Remove
method <br>
    <%
        try {
            cart.getCartContent();
        } catch (javax.ejb.NoSuchEJBException e) {
```

```
    %>
        <br>Successfully caught no such object exception. <br>
    <%
    }
    %>
</body>
</html>
```

A *Context lookup* metódusának paraméterében a bean elérési útját kell megadni, ami annak indításakor megjelent a konzolban.

Böngészőben megnézve az oldalt a következő eredményt kapjuk:

