

---

# L8. TUTORIAL Eclipse

## Java Server Faces: Form Simplu Starter

IDE: Eclipse Oxygene JEE [4.7]  
Distribution: **OEPE 12.2.1.6.x**  
**Implementare JSF:** **JSF 2.2 / Apache MyFaces 2.2.x**  
Implementare JPA: JPA 2.1 / EclipseLink 2.7.0  
**Server Web:** **Apache Tomcat 8.5**

---

SGBD: PostgreSQL9/PostgreSQL10  
Driver JDBC: postgresql-42.x.jdbc

---

# Plan

---

- 1. Verificare infrastructură necesară
    - Biblioteci suport: MyFaces
    - Proiect suport JPA
    - Server Tomcat
  - 2. Creare formular start
    - Creare sursă de date formular simplu
    - Creare fișier-pagină gazdă formular simplu
    - Creare rubrici formular
    - Creare butoane de navigare formular
  - 3. Test formular start JSF
-

# 1. Verificare infrastructură necesară

---

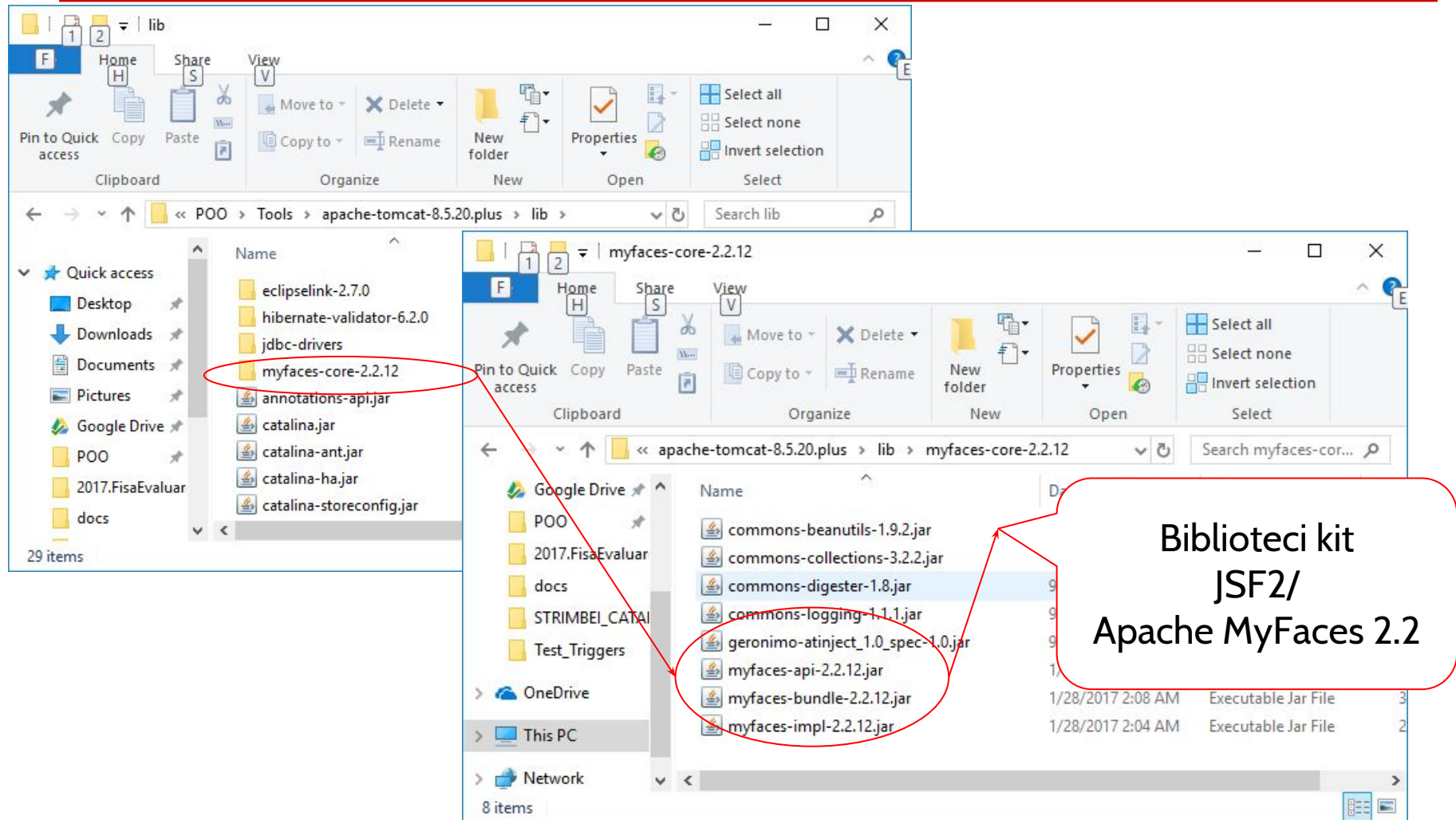
- 1.1 Verificare server Apache Tomcat
  - 1.2 Creare proiect JSF
  - 1.3 Biblioteci suport: MyFaces, JSTL
  - 1.4 Proiect suport JPA
-

# 1.1 Instalare server Apache Tomcat (vezi L7.Tutorial JPA)

---

- *Serverul Apache Tomcat ar trebuie să fie deja configurat din L7.JPA. Doar, în cazul în care Apache Tomcat nu există pașii de urmat ar fi:*
  - 1. In laboratoarele FEAA, serverul ApacheTomcat *preconfigurat* se găsește în folderul local:
    - E:\Programare2\\_apache\\_tomcat
  - Pentru alte locatii (laptopuri, notebookuri personale):
    - Descărcare *arhivă-kit apache-tomcat-7* de pe portal.
    - Dezarhivare kit în directorul-gazdă pentru serverul de aplicații Web Tomcat:
      - E:\Programare2\\_apache\\_tomcat
    - Localizare biblioteci (jars) corespunzătoare JPA în
      - E:\Programare2\\_apache\\_tomcat\lib
  - 2. Adăugare (locație) server Apache Tomcat în contextul workspace-ului Eclipse.

# Localizare biblioteci JSF/Apache\_MyFaces preconfigurate



# 1.2 Creare proiect JSF

---

1. Creare proiect tip *DynamicWebProject*
    - adăugare dependență server de aplicații web Apache Tomcat
  2. Adăugare referință (pentru *compilare*) față de proiectul JPA de persistență
    - in [**Java Build Path**].
  3. Adăugare referință (pentru distribuție/*deploy*) față de proiectul JPA de persistență
    - in [**Deployment Assembly**].
-

File Edit Navigate

1. Alegere tip de proiect Dynamic WebProject

## Dynamic Web Project

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

Project name:

2. Stabilire nume proiect

Project location

☒ Use

Location

ProduseJSF

Browse...

Target runtime

Apache Tomcat v8.5

New Runtime...

Dynamic web module version

3.0

Configuration

JavaServer Faces v2.2 Project

Modify...

Configures a Dynamic Web application to use JSF v2.2

EAR membership

☐ Add project to an EAR

EAR project name:

New Project

3. Alegere tip de server Web

4. Stabilire tip de framework pentru proiect Web: JSF2

Working set

☐ Add project

Working set

?

5. Verificare configurație: JSF Facet **versiunea 2.2**

Ctrl+N

Properties for ScrumJSF2.2

type filter text

- Resource
- AppXray
- Builders
- Deployment Assembly
- Java Build Path
- Java Code Style
- Java Compiler
- Java Editor
- Javadoc Location
- JavaScript
- JSF Facet
- Project Facets
- Project References
- Run/Debug Settings
- Server
- Service Policies
- Targeted Runtimes
- Task Repository
- Task Tags
- Validation
- Web Content Settings

Project Facets


Configuration: <custom>

Project Facet	Version
Axis2 Web Services	1.0
CXF 2.x Web Services	3.0
Dynamic Web Module	3.0
Java	1.7
Java Annotation Processing Support	5.0
JavaScript	1.0
JavaServer Faces	2.2
JAX-RS (REST Web Services)	1.1
JAXB	2.2
JPA	2.1
JSTL	1.2
Oracle Coherence	12.1.3
Struts	1.3
Trinidad	2.0
WebDoclet (XDoclet)	1.2.3

Proces creare proiect JSF

New Dynamic Web Project

### JSF Capabilities

 Library configuration is disabled. Further classpath changes may be required later.

JSF Implementation Library

Type: **Disable Library Configuration**

This facet requires JSF implementation library to be present on project classpath. By disabling library configuration, user takes on responsibility of configuring classpath appropriately via alternate means.

☒ Configure JSF servlet in deployment descriptor

JSF Configuration File: /WEB-INF/faces-config.xml

JSF Servlet Name: Faces Servlet

JSF Servlet Class Name: javax.faces.webapp.FacesServlet

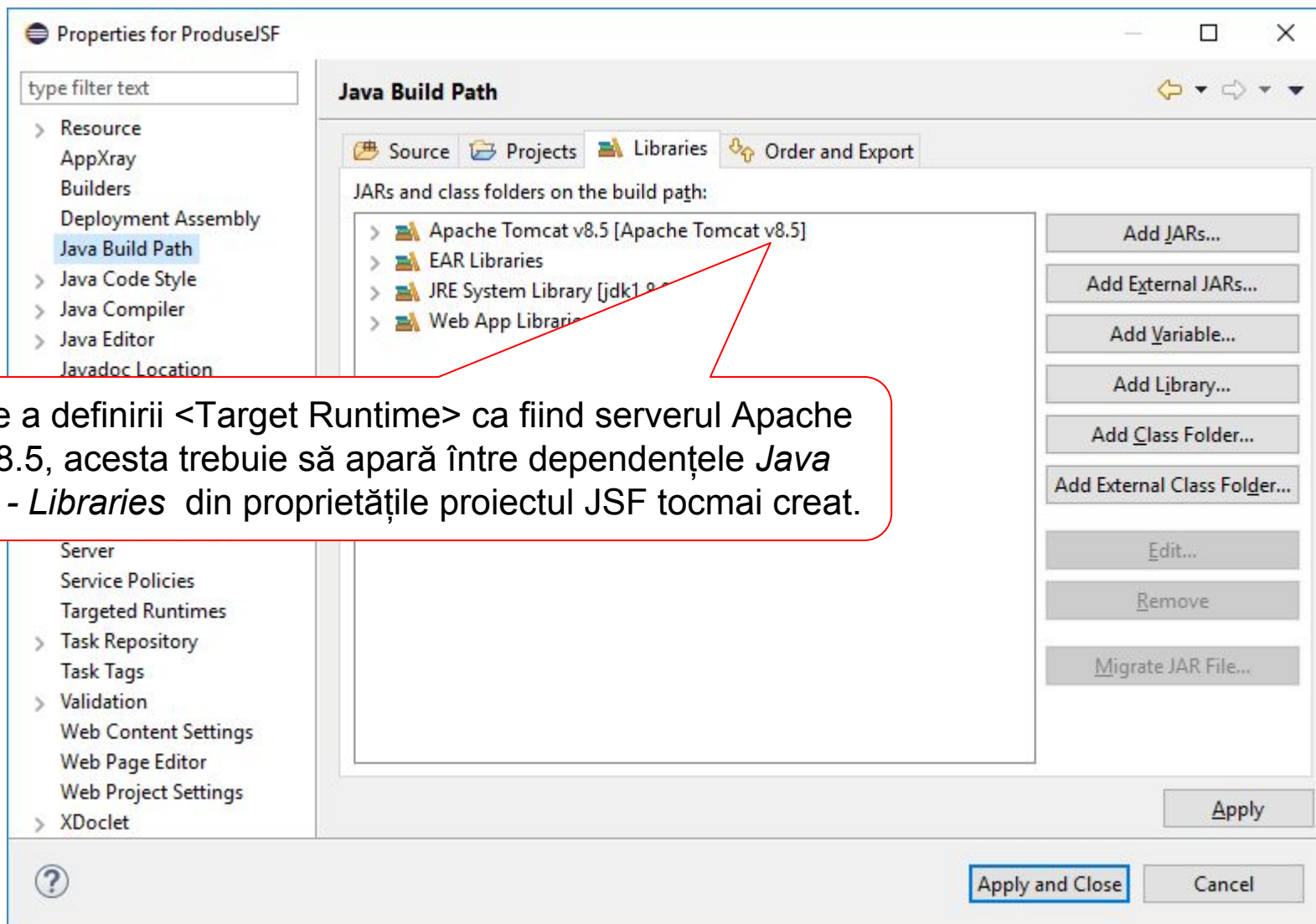
URL Mapping Patterns: /faces/\*

Add...  
Remove

? < Back Next > Finish Cancel

7. Bibliotecile suport JSF se gasesc deja in dir [lib] al ServerRuntime-ului ApacheTomcat7, prin urmare dezactivam definitia explicita a dependentelor JSF



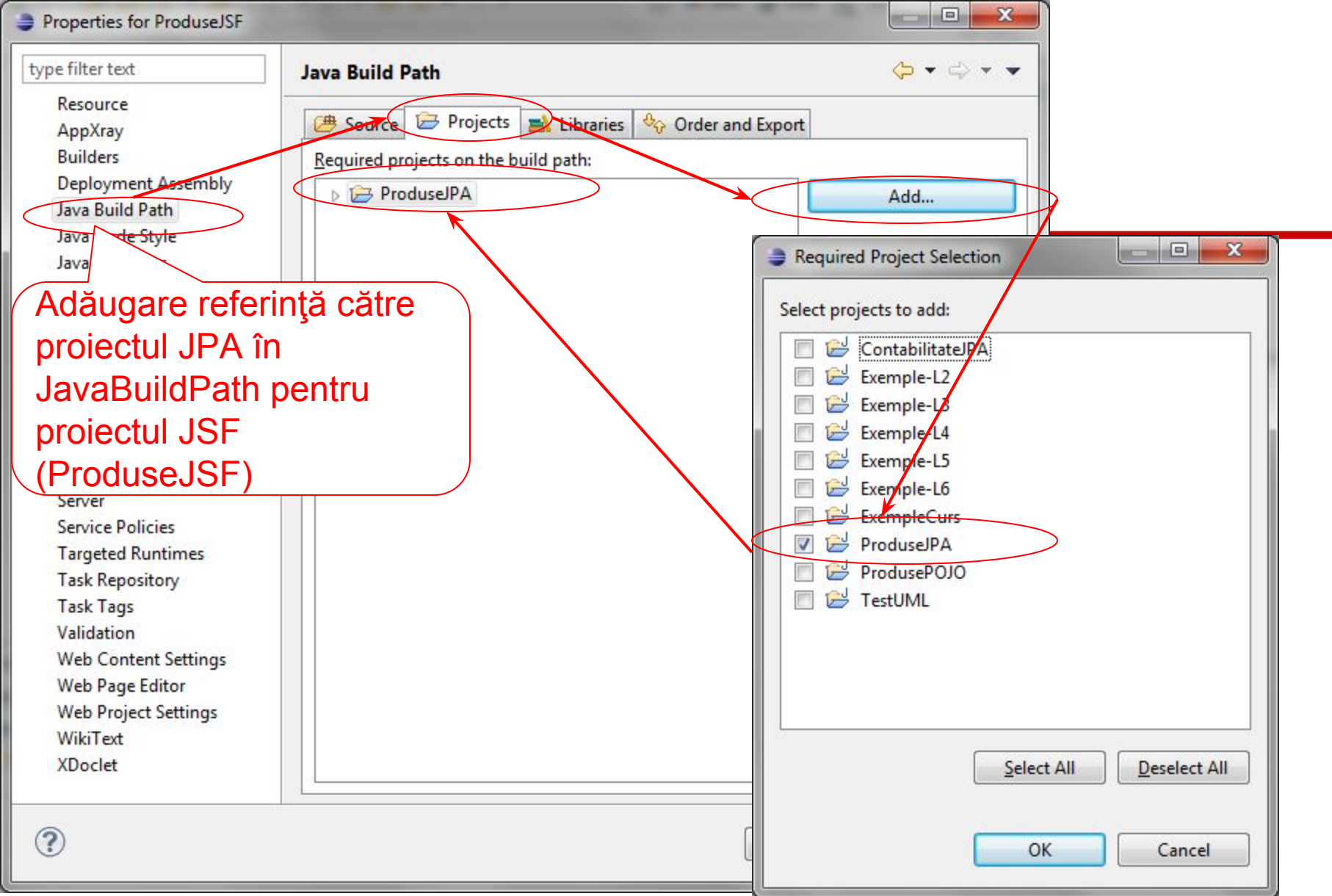


## 1.2.2 Configurare dependențe către proiectul suport JPA

---

Adăugare **dependențe de compilare** către proiectul suport JPA (din fereastra de proprietăți a proiectului JSF):

1. activare secțiune **Java Build Path**,
  2. deschidere pagina **Projects**,
  3. acționare **Add...**;
  4. bifare **nume proiect JPA**.
-



# Adăugare dependențe de distribuție către proiectul suport JPA

---

- Adăugare referințe proiect JPA în *configurația de distribuire/instalare*:
    - a) prin activare secțiune **Deployment Assembly**,
    - b) acționare **Add**,
    - c) selectare directivă **Project**;
    - d) selectare nume proiect JPA;
-

Properties for ProduceJSF

type filter text

- Resource
- AppXray
- Builders
- Deployment Assembly
- Java Build Path
- Java Code Style
- Java Compiler
- Java Editor
- JavaDoc Location
- JavaScript
- JSP Fragment
- Project Facets
- Project Properties
- Run/Debug Settings
- Server
- Service
- Subversion
- Target Runtimes
- Task Runner
- Task Tools
- Validation
- Web Content
- Web Project
- Web Project Facets
- WikiTool
- XDoclet

**Web Deployment Assembly**

Define packaging structure for this Java EE Web Application project.

Source	Deploy Path
/src	WEB-INF/classes
/WebContent	/
ProduceJPA	WEB-INF/lib/ProduceJPA.jar

Advanced

**New Assembly Directive**

Select Directive Type

Add a new assembly directive.

- Archive via Path Variable
- Archives from File System
- Archives from Workspace
- Folder
- Java Build Path Entries
- Project

**New Assembly Directive**

Projects

Select projects to include in the deployment assembly.

- ContabilitateJPA
- Exemple-L2
- Exemple-L3
- Exemple-L4
- Exemple-L5
- Exemple-L6
- ExempleCurs
- ProduceJPA
- ProducePOJO
- TestUML

Adăugare referințe distributie proiect JPA

Adăugare dependență către proiectul JPA în Deployment Assembly pentru proiectul JSF (ProduceJSF)

## 2. Creare formular start

---

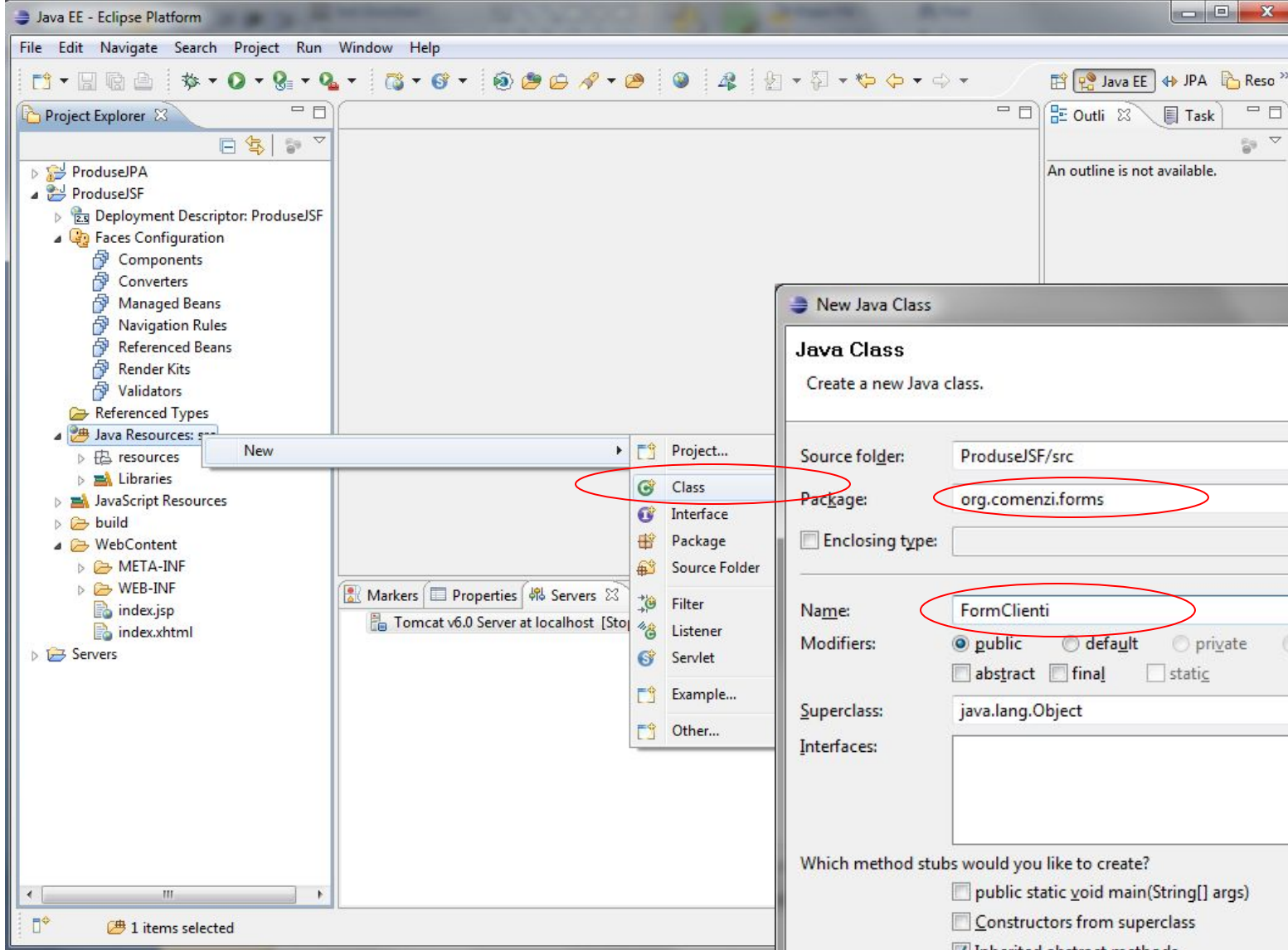
- 2.1 Creare sursă de date formular simplu
  - 2.2 Creare fișier gazdă formular simplu
    - Creare fișier XHTML
    - Creare form JSF cu rubrici legate la sursa de date
    - Creare butoane de navigare formular
      - creare linie de navigare cu butoane
      - declarare acțiuni în fișier formular suport
      - legare butoane la acțiuni
  - 2.3 Test formular simplu JSF
-

## 2.1 Creare sursă de date formular simplu

---

- Creare clasă suport <back-bean> *FormClienti* din secțiunea *Java Resource: src* a perspectivei JavaEE sau Web
- Codificarea clasei *FormClienti*
  - Definirea modelului de date
    - Atribut pentru entitatea curentă
    - Colecție pentru entitățile accesibile
  - Constructorul implicit (fără parametri)
    - Invocarea suportului de persistență pentru a inițializa modelul de date
- Declararea clasei *FormClienti* drept *managed bean* pentru a putea fi invocată din contextul aplicației (în special de cadrul *binding* care o va lega de componentele grafice)





**New Java Class**

Create a new Java class.

**Java Class**

Source folder: ProduceJSF/src Browse...

Package: org.comenzi.forms Browse...

☐ Enclosing type: Browse...

Name: FormClienti

Modifiers: ☒ public ☐ default ☐ private ☐ protected  
☐ abstract ☐ final ☐ static

Superclass: java.lang.Object Browse...

Interfaces: Add... Remove

Which method stubs would you like to create?

☐ public static void main(String[] args)  
☐ Constructors from superclass  
☒ Inherited abstract methods

Do you want to add comments? (Configure templates and default value [here](#))  
☐ Generate comments

? Finish Cancel



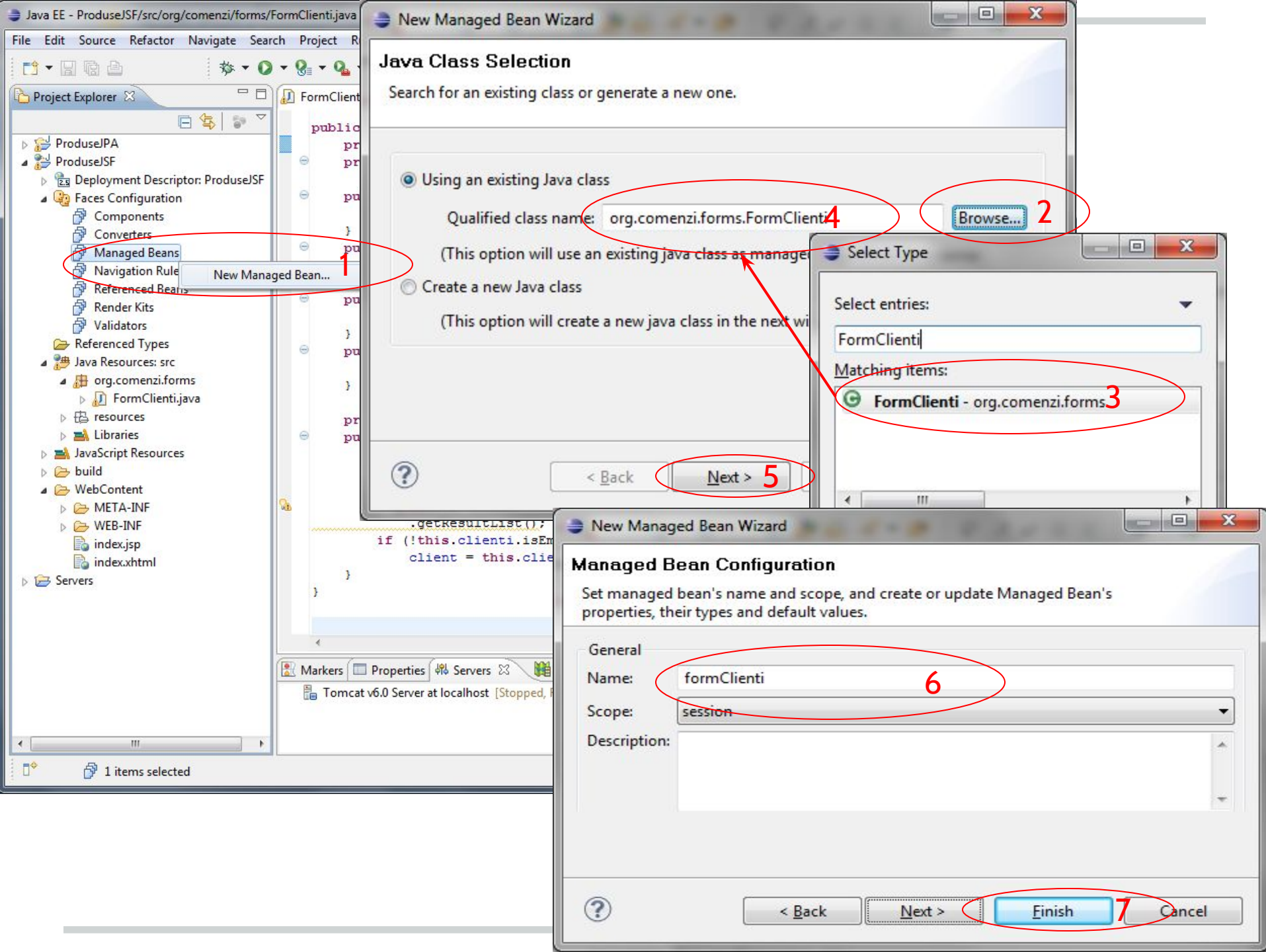
```
FormClienti.java X

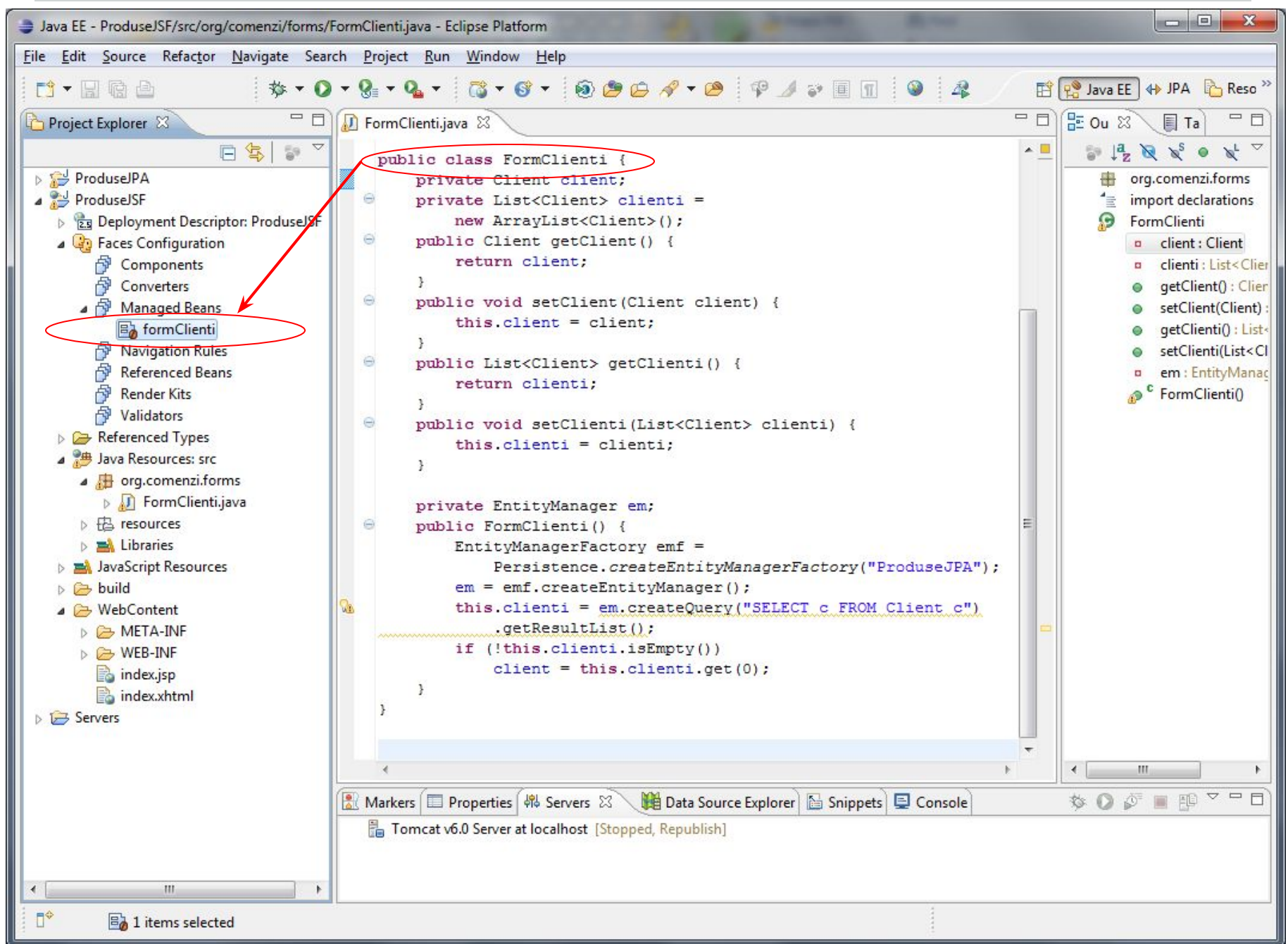
public class FormClienti {
    private Client client;
    private List<Client> clienti =
        new ArrayList<Client>();
    public Client getClient() {
        return client;
    }
    public void setClient(Client client) {
        this.client = client;
    }
    public List<Client> getClienti() {
        return clienti;
    }
    public void setClienti(List<Client> clienti) {
        this.clienti = clienti;
    }

    private EntityManager em;
    public FormClienti() {
        EntityManagerFactory emf =
            Persistence.createEntityManagerFactory("ProduseJPA");
        em = emf.createEntityManager();
        this.clienti = em.createQuery("SELECT c FROM Client c")
            .getResultList();
        if (!this.clienti.isEmpty())
            client = this.clienti.get(0);
    }
}
```

*Model de date*

*Invocare JPA pentru  
Inițializare  
model de date*





## 2.2 Creare fișier gazdă formular simplu

---

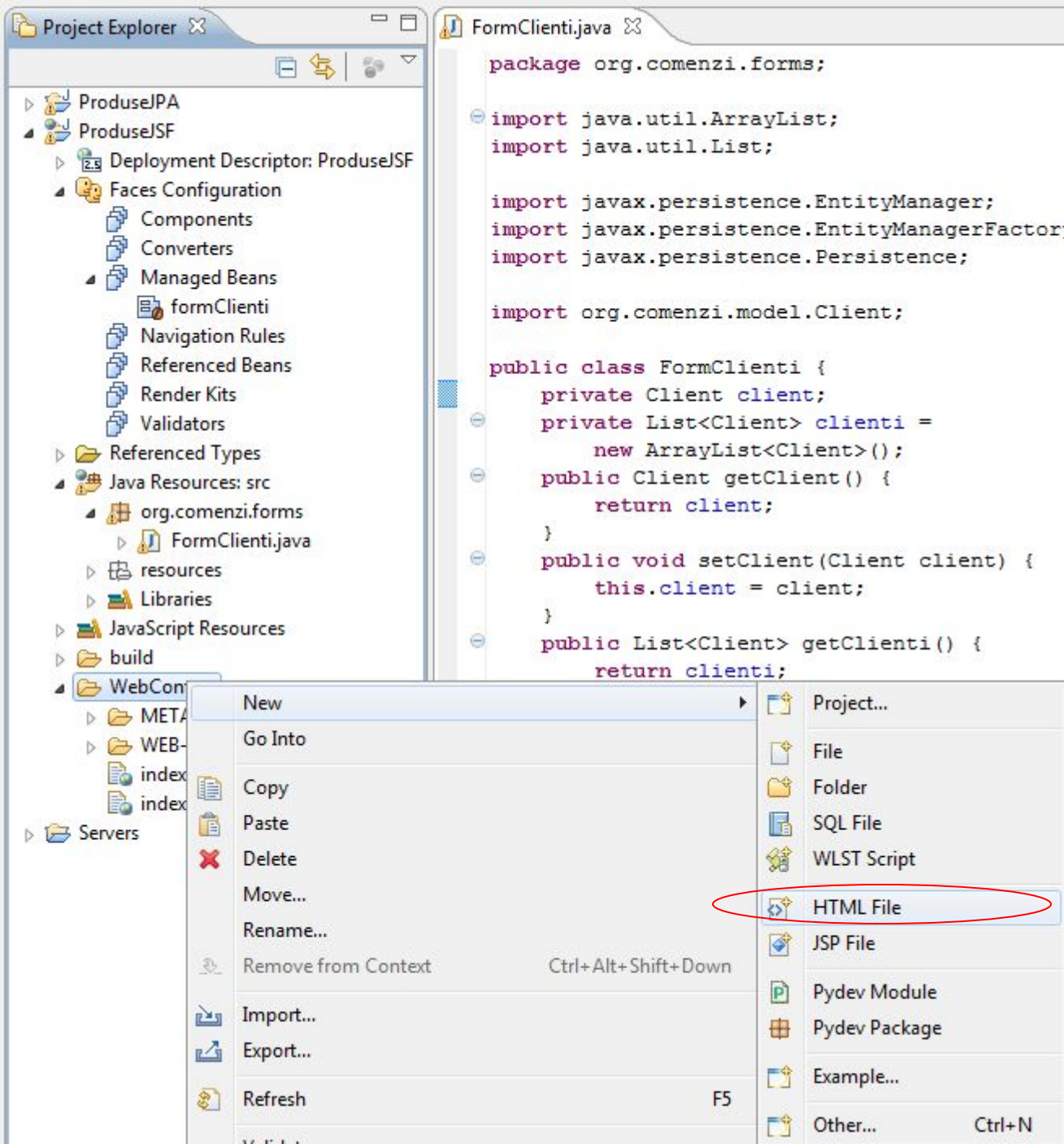
- 2.2.1 Creare fișier XHTML
    - Creare pagină după template Facelet Header
    - Modificare titlu pagină formular
  - 2.2.2 Componente grafice
    - Creare form JSF (drag&drop) legat la sursa de date
  - 2.2.3 Creare butoane de navigare formular
    - creare butoane
    - declarare acțiuni în fișier formular suport
    - Legare butoane la acțiuni
-

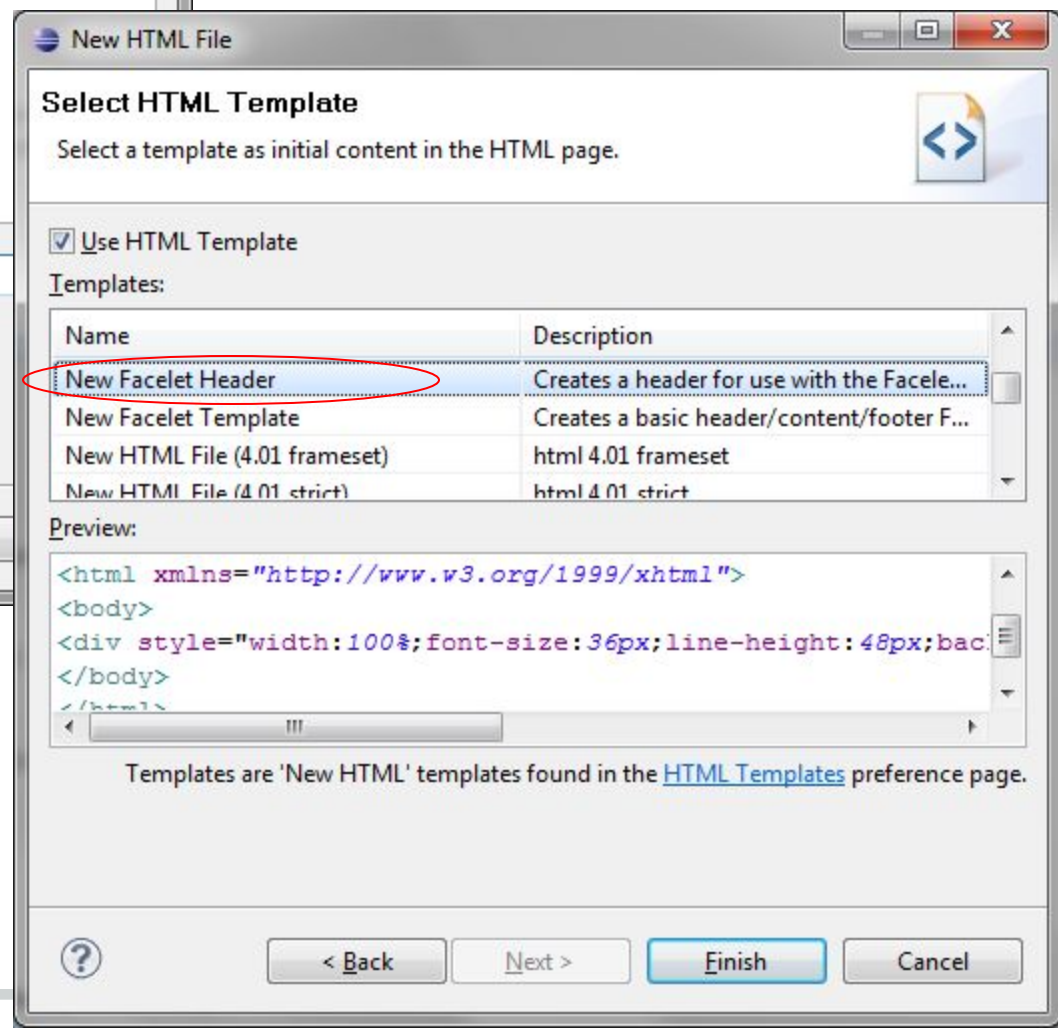
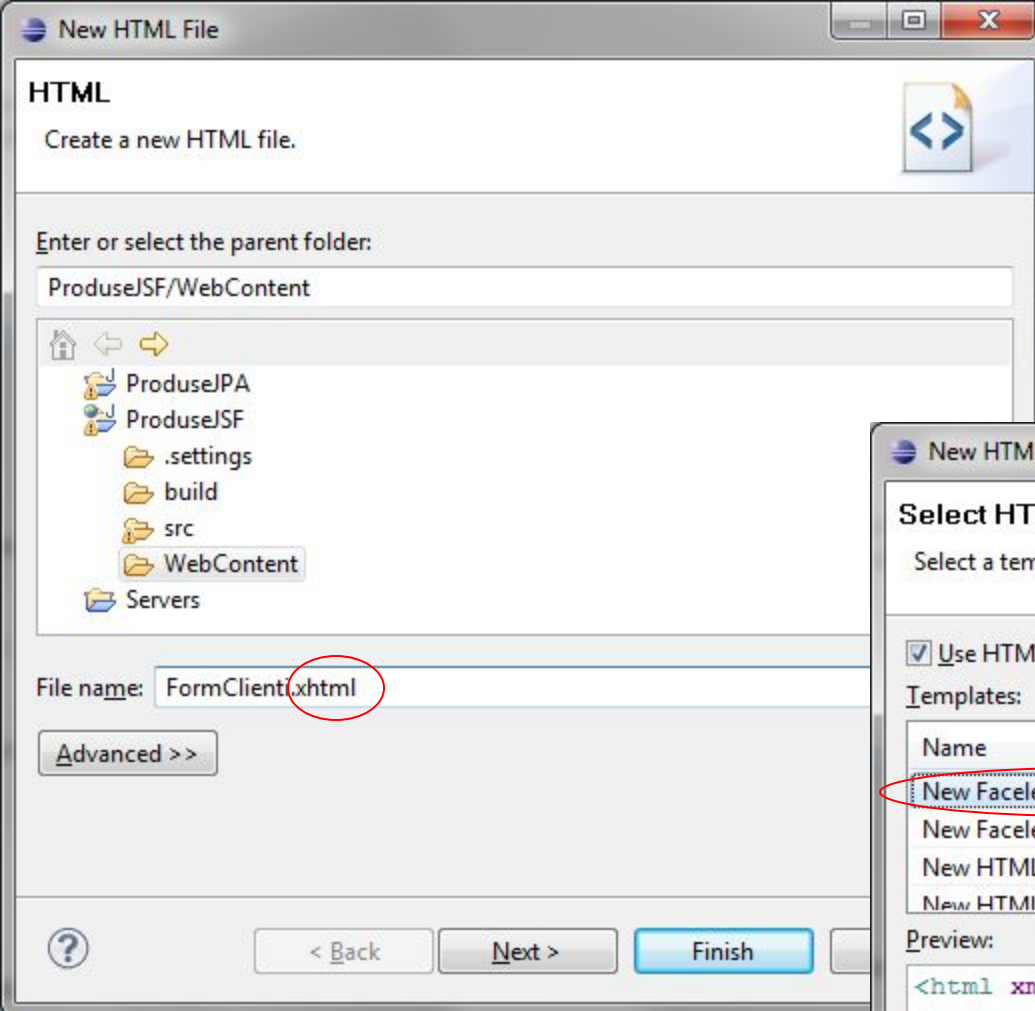
## 2.2.1 Creare fișier XHTML

---

- Creare pagină după template Facelet Header (vezi captura de pe următoarea pagină):
  - Din secțiunea *Web Content* a perspectivei JavaEE sau Web – meniul contextual – opțiunea *New – HTML File*
    - in primul pas se specifică numele paginii (FormClienti) conținând formularul – *adăugând explicit extensia **xhtml***,
    - in al doilea pas se alege șablonul ***New Facelet Header***.
-





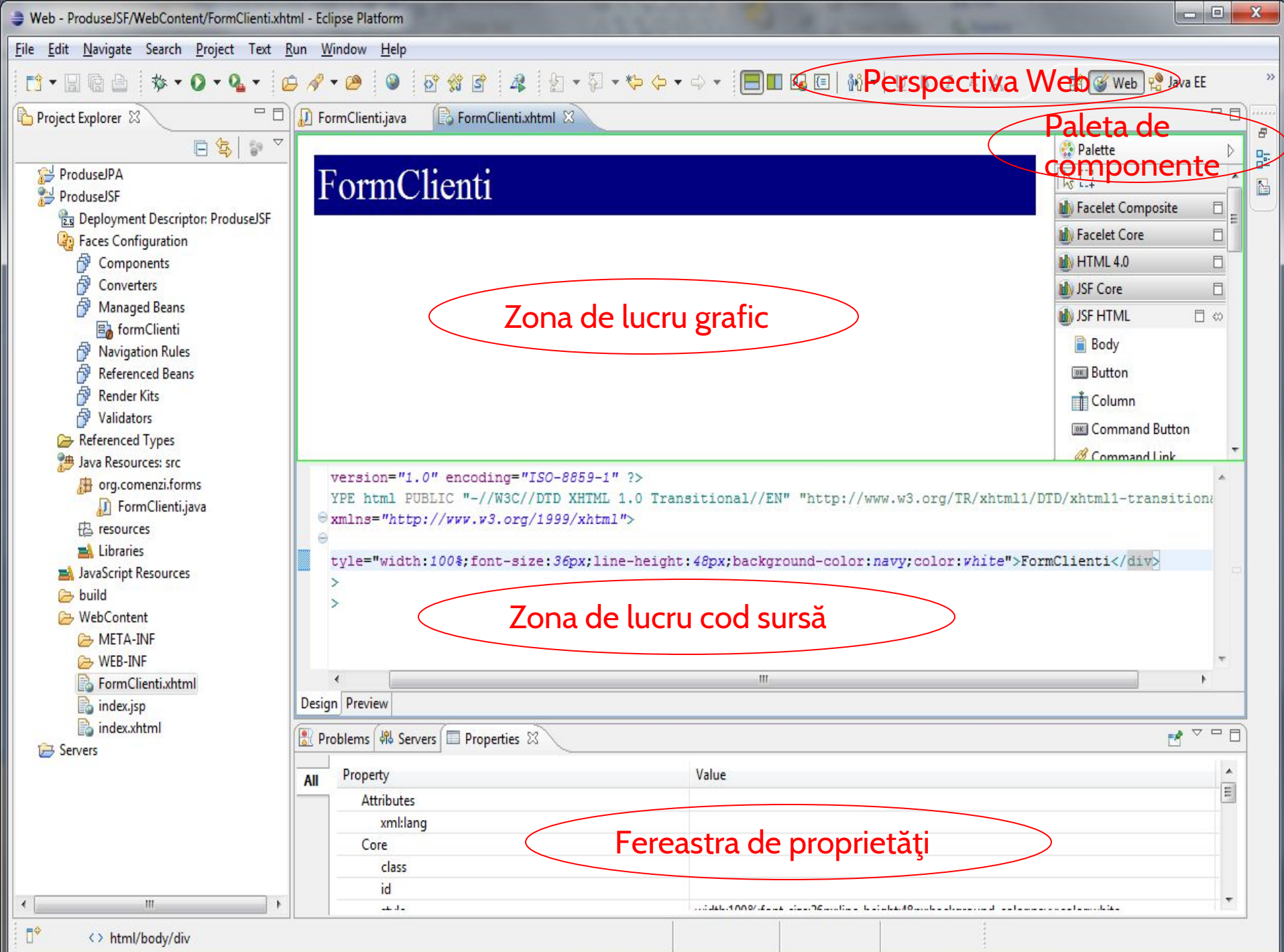


# Creare fișier XHTML

---

- Modificare titlu pagină formular
    - Dacă perspectiva Web nu a fost activată, atunci după finalizarea secvenței de pași de creare a fișierului aceasta va fi activată (vezi captura de pe următoarea pagină).
    - Zona de lucru va fi împărțită astfel
      - O sub-zonă de lucru grafic (drag&drop) care va include o paletă de componente
      - O sub-zonă de lucru în mod cod-sursă
      - O fereastră de proprietăți pentru elementul selectat curent în oricare din cele două secțiuni
    - Modificarea titlului implicit se poate face direct în zona de lucru grafic
-



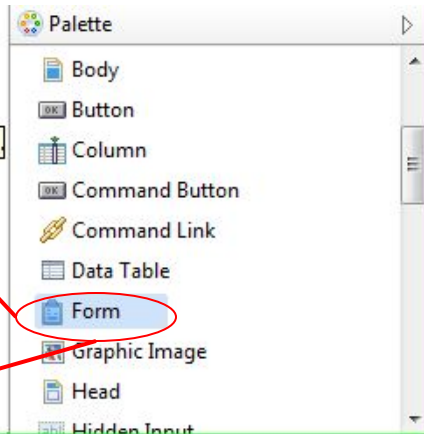


## 2.2.2 Componente grafice legate la sursa de date

---

- Creare form JSF (drag&drop) legat la sursa de date (vezi capturile de pe următoarele pagini):
- Selectare componentă *Form* din secțiunea *JSF HTML* a paletei de componente
- drag&drop în zona de lucru de grafică
- activare secvență asistent configurare form
  - Pasul 1 – selectare sursă de date
    - Alegere opțiune de generare: *Generare a form tag and content from data*
    - Acționare buton selecție *Form Bean*: selectare proprietate *client* din sursa *formClienti*
  - Pasul 2 – selectare proprietăți *client* care vor genera rubricile formularului
  - Pasul 3 – configurare rubrici
    - Poziție-ordine rubrici
    - Text etichete asociat

# Form Clienti



```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
<body>
<div style="width:100%;font-size:36px;line-height:1.2">
</div>
</body>
</html>
```

New Form

### Form Type

Choose the type of form to create.

Generation options: ☒ Configure a form tag ☐ Generate a form tag and content from data

▼ Form

Id:

Binding:  ⚡

? < Back Next > Finish Cancel

New Form f1

### Form f1 Type

Choose the type of form f1 to create.

Generation options: ☐ Configure a form tag ☒ **Generate a form with content from managed beans**

Form bean: formClienti.client

Form action:

**Pasul 1**

Choose Bean/Bean Property

Select a bean or bean property containing fields:

- JSF Managed Beans
  - formClienti
    - client**
    - clienti

OK Cancel

New Form

### Property Selection

Select the properties to use as form fields.

Properties:

- ☒ client
  - ☒ id
  - ☒ nume

**Pasul 2**

New Form

### Configure Form Fields

Order the fields as they should appear in the form. Indicate the rendering for each field.

Form fields:

Name	Label	Type	Rendering
client.id	ID client	java.lang.Integ...	Text Field
<b>client.nume</b>	<b>Nume client</b>	java.lang.String	Text Field

↑ ↓

**Pasul 3**

< Back Next > Finish Cancel

FormClienti.java FormClienti.xhtml

# FormClienti

**formClienti**

Id client:	<input type="text" value="#{formClienti.client.id}"/>	message
Nume client:	<input type="text" value="#{formClienti.client.nume}"/>	message

```
</p:facet>
<h:outputText value="Id client:"></h:outputText>
<h:inputText id="id" value="#{formClienti.client.id}"></h:inputText>
<h:message for="id"></h:message>
<h:outputText value="Nume client:"></h:outputText>
<h:inputText id="nume" value="#{formClienti.client.nume}"></h:inputText>
<h:message for="nume"></h:message>
</h:panelGrid>
</h:form>
</body>
```

Design Preview

Problems Servers Properties

**General**

All

**Input Text id**

Id:	id
Value:	<input type="text" value="#{formClienti.client.id}"/>
Binding:	<input type="text"/>

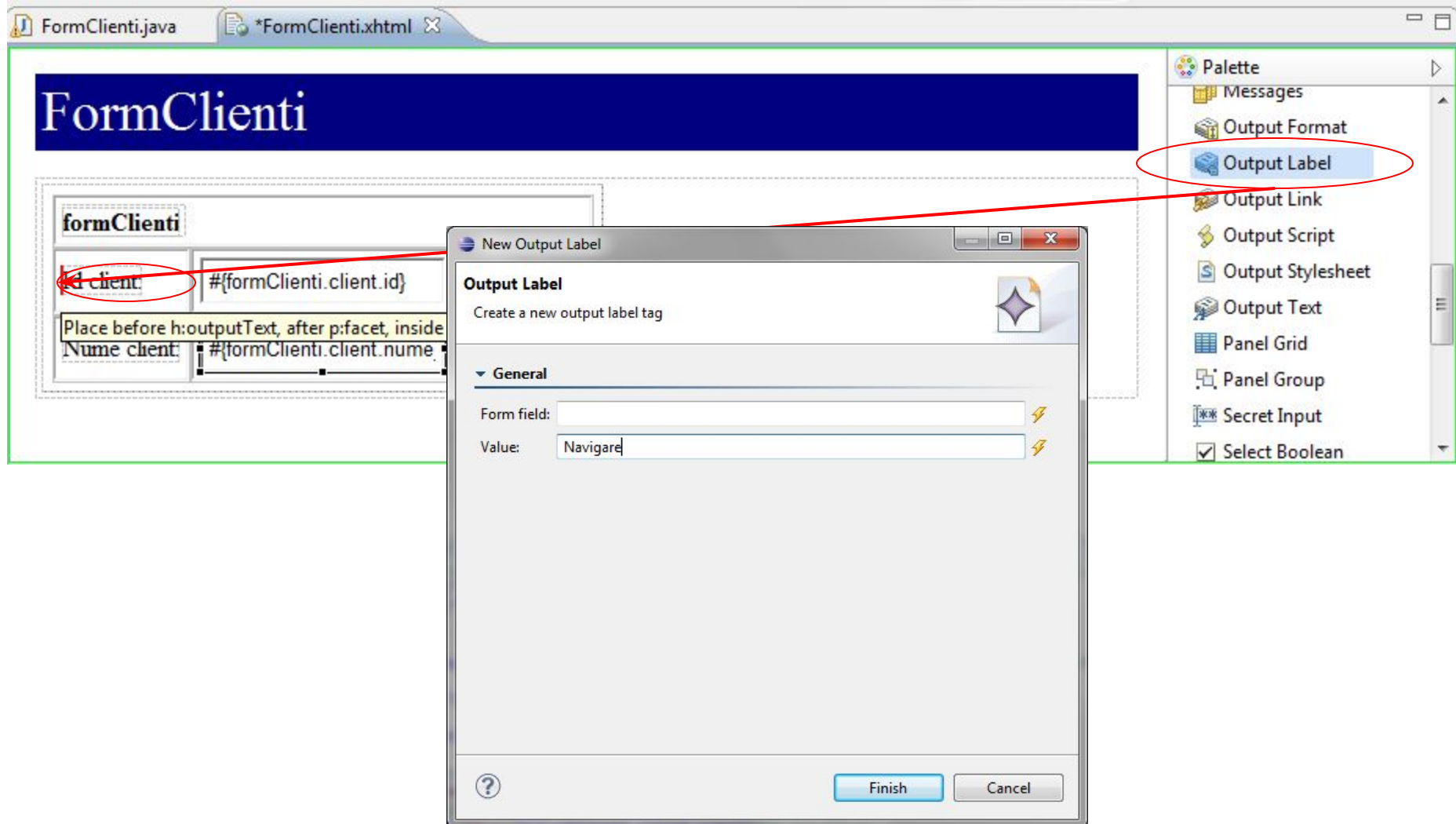


## 2.2.3 Creare butoane de navigare formular

---

- **Așezarea butoane în cadrul formularului:** zona de navigare va constitui o nouă linie deasupra liniilor cu rubricile obișnuite.
- Pentru a obține o nouă linie vom aduce (drag&drop) înaintea etichetei “*Id client*”, linie conținând exact trei componente (vezi capturile de pe următoarele pagini):
  - o nouă etichetă – *Navigare: drag&drop* OutputLabel din secțiunea JSF HTML a paletei
  - un panou cu 2 coloane (câte una pentru fiecare buton): *drag&drop* Panel Grid din secțiunea JSF HTML a paletei
  - un mesaj sau o etichetă finală: *drag&drop* Message sau Output Label din secțiunea JSF HTML a paletei
- În interiorul sub-panoului cu 2 coloane vor fi aduse (*drag&drop*) două componente *Command Button* din secțiunea JSF HTML a paletei, pentru care în fereastra *Properties* vor fi modificate proprietățile:
  - Id, în *cmdPrevious* respectiv *cmdNext*;
  - Value, în *Prev* respectiv *Next*

Componentele care vor forma linia de navigare vor fi *trase* exact înaintea primei rubrici ...



Linia de navigare trebuie formată din exact 3 componente, altfel se “dezechilibrează” așezarea în pagină, panoul principal al formularului fiind definit cu 3 coloane...

The screenshot displays a web form builder interface. The main workspace shows a form titled "formClienti" with a navigation bar at the top. The navigation bar contains three components: "Navigare", a placeholder box, and a "message" component. Below the navigation bar, there are three rows of form fields: "Id client:" with a text input containing "#{formClienti.client.id}", "Nume client:" with a text input containing "#{formClienti.client.num...", and a "message" component. A "message" component is also present in the middle row. A "New Panel Grid p2" dialog box is open in the foreground, showing the "Panel Grid p2 Type" configuration. The dialog has two options: "Configure a panelGrid tag" (selected) and "Generate a panelGrid with content from managed beans". Under the "General" section, the "Columns" property is set to 2. A "Palette" on the right side of the workspace lists various components: "Link", "Message", "Messages", "Output Format", "Output Label", "Output Link", "Output Script", "Output Stylesheet", "Output Text", and "Panel Grid". Red circles and arrows highlight specific elements: a red circle around the "Navigare" component in the navigation bar; a red circle around the placeholder box in the navigation bar; a red circle around the "message" component in the middle row; a red circle around the "message" component in the bottom row; a red circle around the "Columns" property in the "New Panel Grid p2" dialog, which is set to 2; a red circle around the "Message" component in the "Palette"; a red circle around the "Output Label" component in the "Palette"; and a red circle around the "Panel Grid" component in the "Palette".

FormClienti.java \*FormClienti.xhtml

# FormClienti

formClienti

Navigare

Id client: #{formClienti.client.id}

Nume client: #{formClienti.client.num...

message

message

message

New Panel Grid p2

Panel Grid p2 Type

Choose the type of panel grid p2 to create.

Generation options: ☒ Configure a panelGrid tag ☐ Generate a panelGrid with content from managed beans

General

Id: p2

Binding:

Columns: 2

Rendered: ☒

Width:

Palette

Link

Message

Messages

Output Format

Output Label

Output Link

Output Script

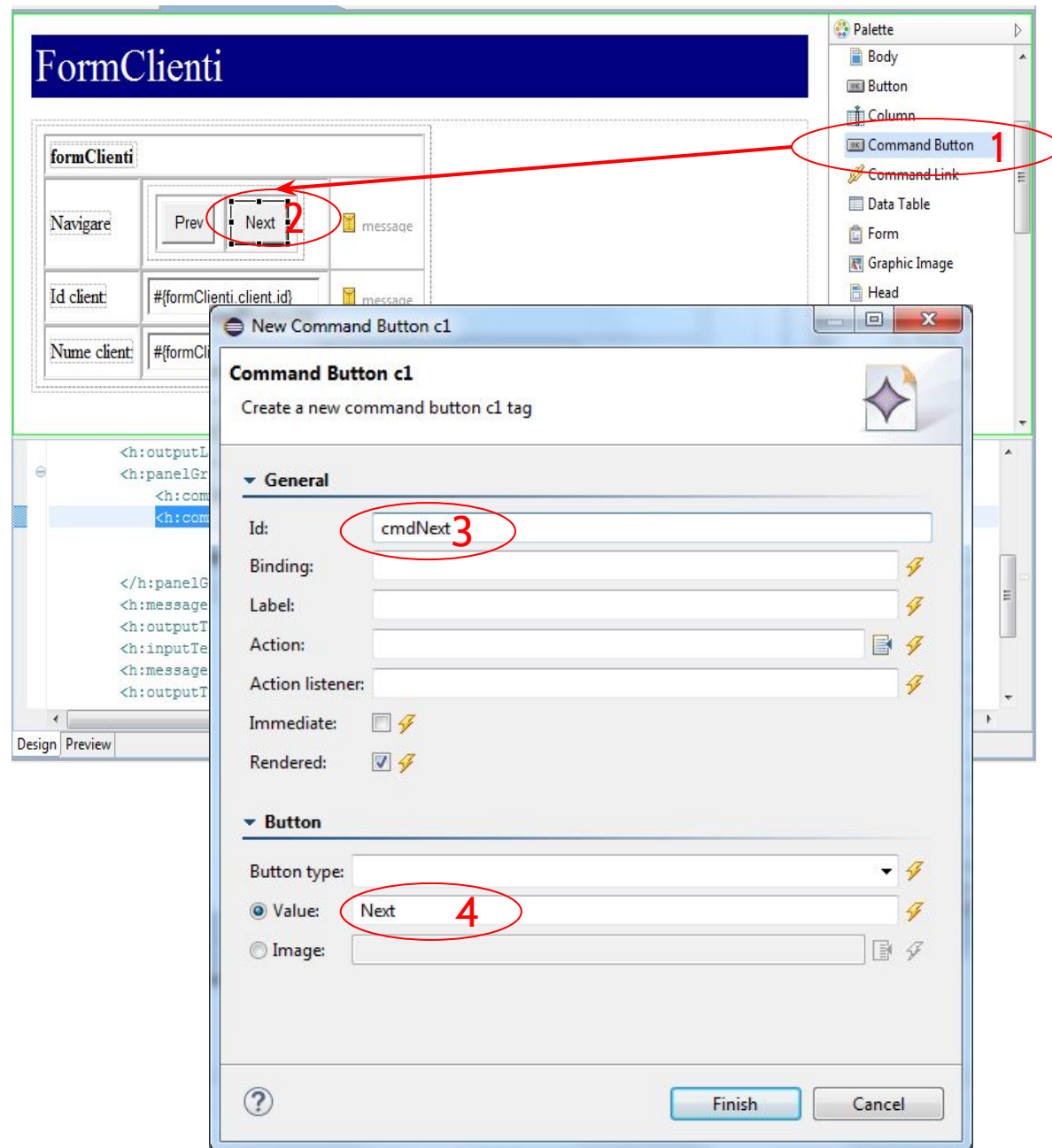
Output Stylesheet

Output Text

Panel Grid



În panoul  
secundar adus  
pe coloana a  
doua din prima  
linie a panoului  
principal al  
formularului vor  
fi “trase” cele  
două butoane de  
navigare  
*cmdPrev* și  
*cmdNext*



## 2.2.3 Creare butoane de navigare formular

---

- Activare butoane de navigare
  - Creare acțiuni de navigare în clasa suport a formularului grafic (vezi cod sursă pe următoarea pagină)
    - în clasa *FormClienti* se adaugă operațiile *nextClient* și *previousClient* care
      - vor fi parametrizate cu *javax.faces.event.Event*;
      - actualizează referința clientului curent relativ la pozițiile din lista de clienți
  - Legare acțiuni de navigare la butoane de navigare (vezi capturile de pe următoarele pagini)
    - se selectează fiecare buton de navigare;
    - se accesează proprietatea *ActionListener* din fereastra de proprietăți și se acționează butonul *Bind to a dynamic value*;
    - din fereastra activată se selectează metoda de navigare din beanul *formClienti*.

```
public class FormClienti {
    private Client client;
    private List<Client> clienti =
        new ArrayList<Client>();
    public Client getClient() {
        return client;
    }
    public void setClient(Client client) {
        this.client = client;
    }
    public List<Client> getClienti() {
        return clienti;
    }
    public void setClienti(List<Client> clienti) {
        this.clienti = clienti;
    }

    private EntityManager em;
    public FormClienti() {
        EntityManagerFactory emf =
            Persistence.createEntityManagerFactory("persistence-unit");
        em = emf.createEntityManager();
        this.clienti = em.createQuery("select c from Client c")
            .getResultList();
        if (!this.clienti.isEmpty())
            client = this.clienti.get(0);
    }

    public void previousClient(ActionEvent evt) {
        Integer idxCurent = this.clienti.indexOf(client);
        if (idxCurent > 0)
            this.client = this.clienti.get(idxCurent - 1);
    }

    public void nextClient(ActionEvent evt) {
        Integer idxCurent = this.clienti.indexOf(client);
        if ((idxCurent+1) < this.clienti.size())
            this.client = this.clienti.get(idxCurent + 1);
    }
}
```

Atentie ActionEvent trebuie  
importat din  
javax.faces.event.ActionEvent

Acțiunile de navigare.  
ActionEvent provine din  
javax.faces.event

# FormClienti

formClienti

Navigare

Prev

Next

1

message

Id client:

`#{formClienti.client.id}`

message

```
</p:facet>
<h:outputLabel value="Navigare"></h:outputLabel>

<h:panelGrid columns="2" border="1">
  <h:commandButton value="Prev" id="cmdPrevious" actionListener="#{formClienti.previousClient}"></h:commandButton>
  <h:commandButton value="Next" id="cmdNext" actionListener="#{formClienti.nextClient}"></h:commandButton>
</h:panelGrid>

<h:message></h:message>
```

Design Preview

Problems Servers Properties

General

All

## Command Properties

Id: cmdNext

Action:

Action listener: `#{formClienti.nextClient}`

Immediate: ☐

## Button Properties

Button type: submit

Value: Next

## Expression Builder

### Expression Builder

Build an EL expression that will be evaluated at runtime.

Expression:

`#{formClienti.previousClient}`

Variables:

type filter text

Managed Beans

formClienti

client

clienti

clientiList

selected

- void previousClient(ActionEvent evt)
- void nextClient(ActionEvent evt)
- void selectClient(ActionEvent evt)
- void adaugareClient(ActionEvent evt)
- void stergereClient(ActionEvent evt)
- void salvareClient(ActionEvent evt)
- void abandonClient(ActionEvent evt)

Operators:

&&

||

>

<

>=

<=

==

!=

!

+

-

\*

/

%

OK

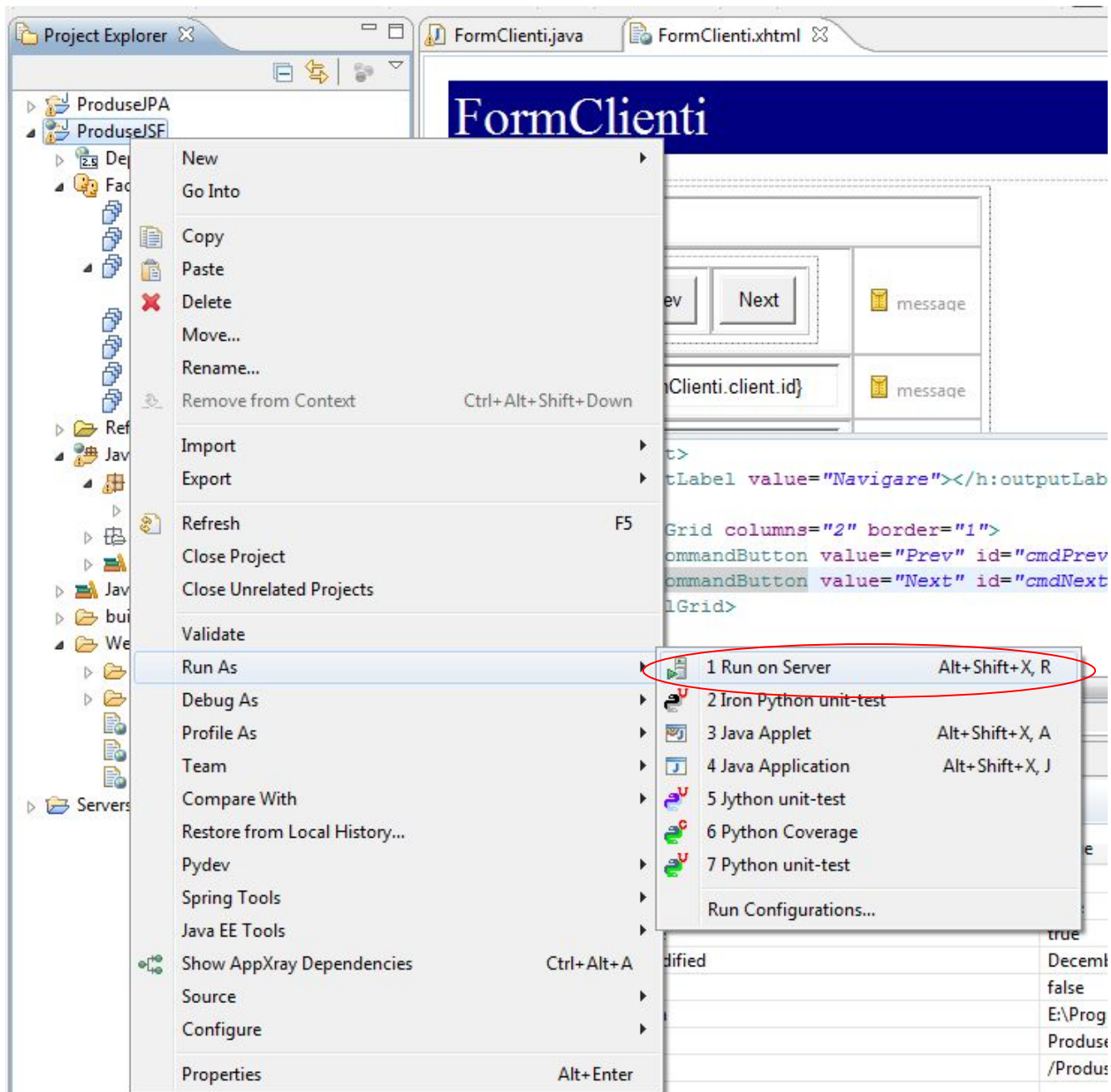
Cancel

# 3. Test formular start JSF

---

- Din meniul contextual al formularului JSF (în perspectiva JavaEE sau Web) se selectează opțiunea *Run* și acțiunea *Run on server*
  - În fereastra consolă se poate observa pornirea serverului Tomcat (dacă nu a fost deja pornit în prealabil) printr-o serie de mesaje dintre care unul va indica calea de *deploy* a aplicației curente
  - Fereastra browserului se va lansa către un URL format astfel
    - URLul generic al aplicație, de exemplu  
<http://localhost:8080/ProduseJSF>
    - La care se adaugă numele formularului urmat de extensia *jsf* și nu *xhtml*, de exemplu  
<http://localhost:8080/ProduseJSF/faces/FormClienti.xhtml>
-





# Lansare Form JSF

