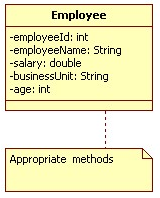
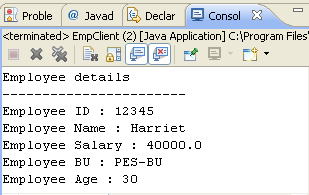
1. Injecting dependencies into a Spring application

|  |  |
| --- | --- |
| **Goals** | * Using IoC to integrate disparate systems in a loosely coupled manner. |
| **Time** | 180 minutes |

**Problem statement-1.1: Injecting dependencies**

Write an Employee bean. Inject values into bean using DI and display all values. Refer the class diagram below

The output would look as shown below:

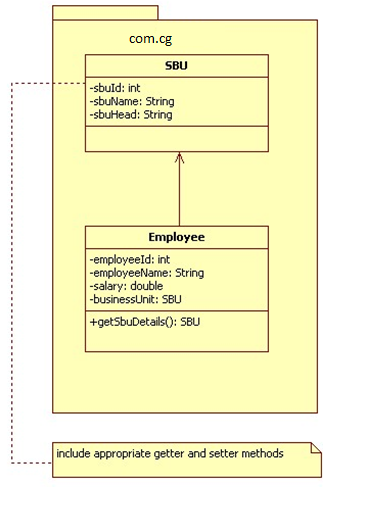


Class Diagram 1: Employee

**** Keep each of the lab solutions separate, preferably in different packages/source folders

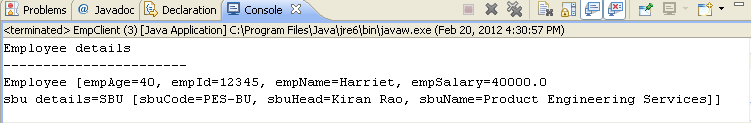
**Problem statement-1.2: Injecting dependencies**

Code SBU bean. Revisit the Employee bean and provide a method to retrieve SBU details (getSBUDetails()) for the employee. You will need to inject the SBU bean to the Employee bean as shown in the Class diagram below:



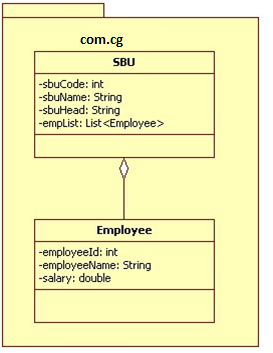
Class Diagram 2: **SBU and Employee**

The output would look as shown below:



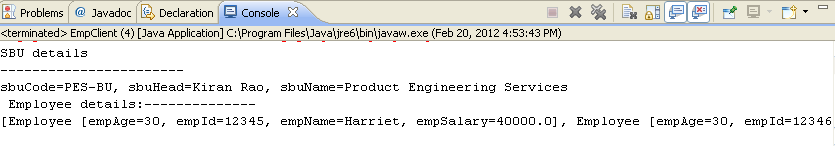
**Problem statement-1.3: Injecting dependencies**

Revisit the SBU bean. Create a new property called empList which will contain a list of all employees in the PES BU. Display the SBU details, followed by a list of all employees in that BU. To inject employee objects into the SBU bean, use “List” collection. Allocate two employees to PES. Refer Class diagram below



Class Diagram 3: **SBU and Employee (Ver -2)**

The output would look as shown below:



**Problem statement-1.4: Injecting dependencies**

Develop a console based spring application where main method of client class will retrieve employee information from Employee collection and displays info in the console as shown:

**Input:**

Employee ID : 100

**Output:**

Employee Info:

Employee ID :100

Emplpoyee NAME :Rama

Employee SALARY :12345.67

Refer diagram below for implementation details:

beans.xml

Collection

main()

EmployeeService

EmployeeDao

Employee

Flow Diagram - 1

**Note**: implement above application using

* Setter Injection
* Constructor Injection (use index and type attribute with constructor arg tag)
* Use different bean wiring mechanism like..
* By Name
* By Type
* Auto Wiring

**Change Request: Now Change the application so that all components are autowired and components are automatically scanned.Use Spring boot API.**

**Problem statement-1.5: Injecting dependencies**

Implement the above lab using Javabase configuration and Spring Boot Features.