
5
Most Useful Design Patterns

Objectives

After completing this lesson, you should be able to do the following:

- Most Useful Design Patterns
- DAO
- MVC

Course Roadmap

Java Design Patterns

- Lesson 1: Introduction to Design Patterns
- Lesson02: Creational Design Patterns
- Lesson03: Structural Design Patterns
- Lesson04: Behavioral Design Patterns
- **Lesson 5: Most Useful Design Patterns** — **You are here!**

Most Useful Design Patterns

Example

```
// Driver code
    public static void main(String args[])
    {
        int arr[] = { 2, 3, 4, 10, 40 };
        int x = 10;

        // Function call
        int result = search(arr, x);
        if (result == -1)
            System.out.print("Element is not present in array");
        else
            System.out.print("Element is present at index “ + result);
    }
}
```

- There are a lot of design patterns that don’t come under Gang of Four design patterns. Let’s look at some of these popular design patterns.

1. DAO Design Pattern

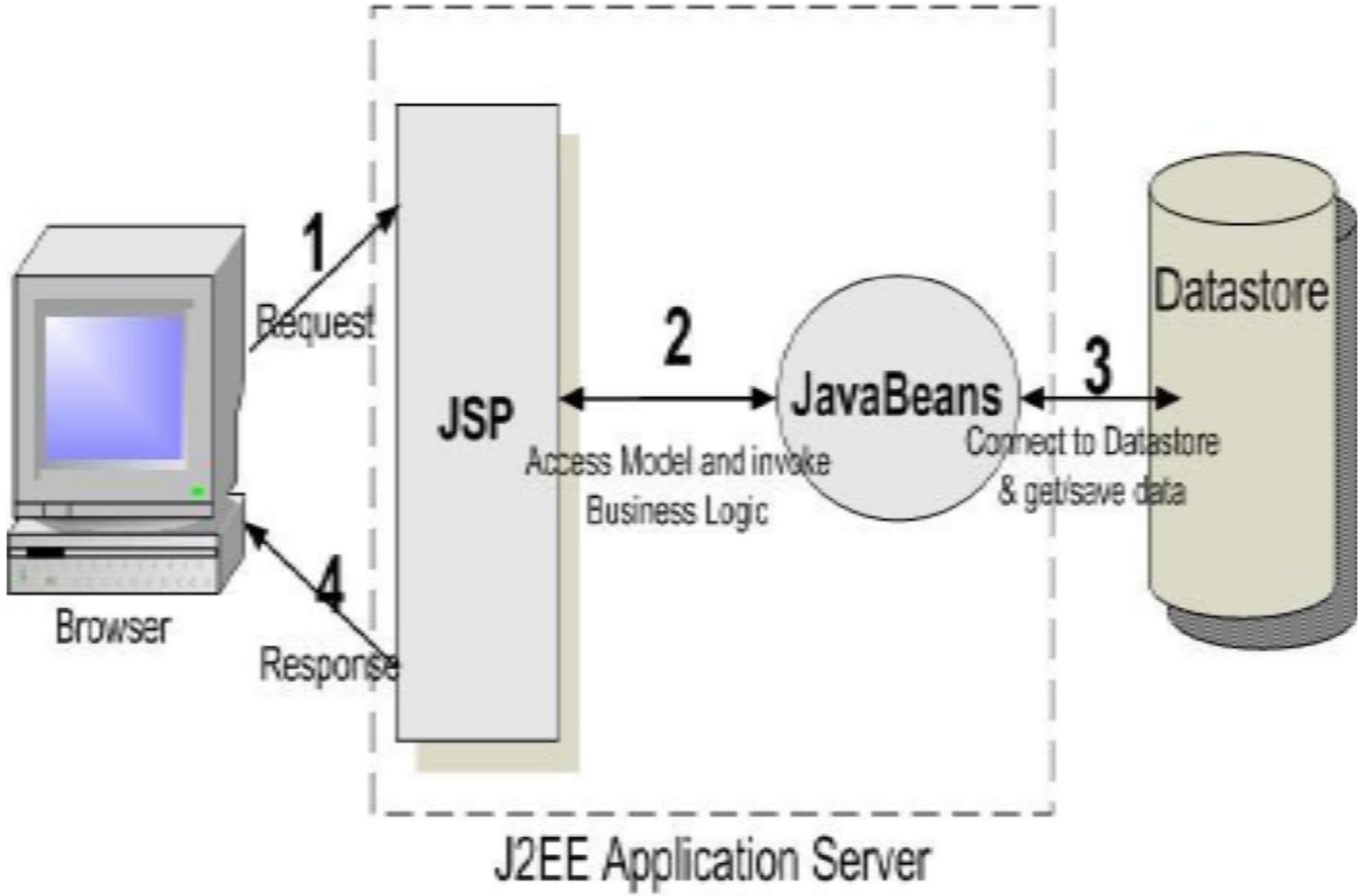
- The Data Access Object (DAO) design pattern is used to decouple the data persistence logic to a separate layer. DAO is a very popular pattern when we design systems to work with databases. The idea is to keep the service layer separate from the data access layer. This way we implement the separation of logic in our application.

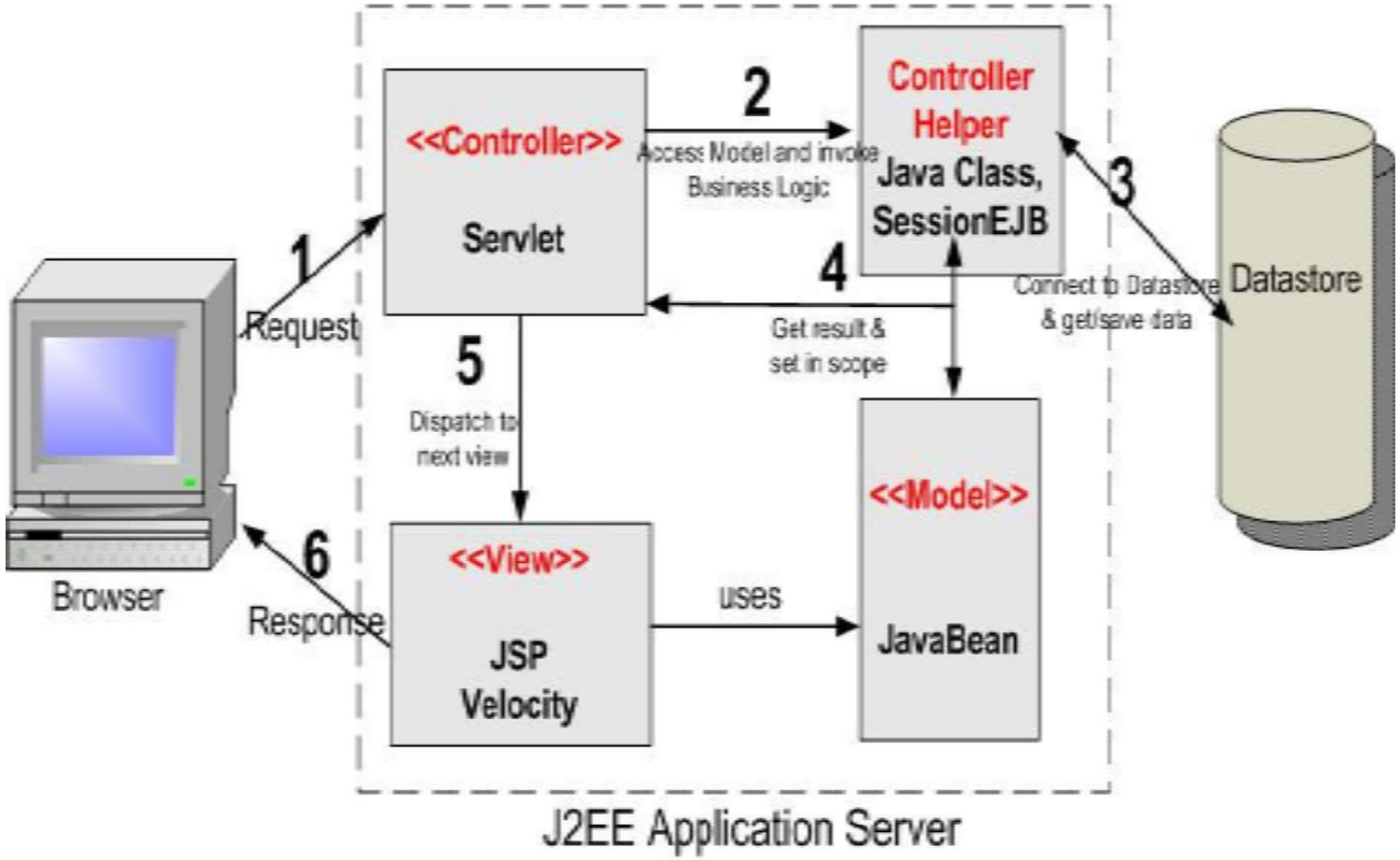
2. Dependency Injection Pattern

- The dependency injection pattern allows us to remove the hard-coded dependencies and make our application loosely-coupled, extendable, and maintainable. We can implement dependency injection in Java to move the dependency resolution from compile-time to runtime. Spring framework is built on the principle of dependency injection.

3. MVC Pattern

- Model-View-Controller (MVC) Pattern is one of the oldest architectural patterns for creating web applications.


1
Request
JSP
2
Access Model and invoke
Business Logic
JavaBeans
3
Connect to Datastore
& get/save data
Datastore
Browser
4
Response
J2EE Application Server


1
Request
<<Controller>>
Servlet
2
Access Model and invoke
Business Logic
Controller
Helper
Java Class,
SessionEJB
3
Connect to Datastore
& get/save data
Datastore
4
Get result &
set in scope
5
Dispatch to
next view
<<Model>>
JavaBean
Browser
6
Response
<<View>>
JSP
Velocity
uses
J2EE Application Server

Summary

In this lesson, you should have learned how to:

- Most Useful Design Patterns
- DAO
- MVC