Enterprise Java Beans (EJB)

EJB – OVERVIEW

- EJB تعتبر server side component تحتوي على business logic .
- هي عبارة عن POJO تم عمل deployment لها في container لتقدم لها بعض الخدمات مثل:
- Transaction mangament
- Concurrency control
- يمكن إعادة استخدامها.
- يمكن استخدامها ك webservice endpoint .

Types of EJBs

- Session Bean
- Message Driven Bean

Types of Session Beans

- يوجد ثلاث أنواع من Session beans وهي:
- Stateless
- Stateful
- Singlton

الخدمات التي تقدمها container لل EJB.

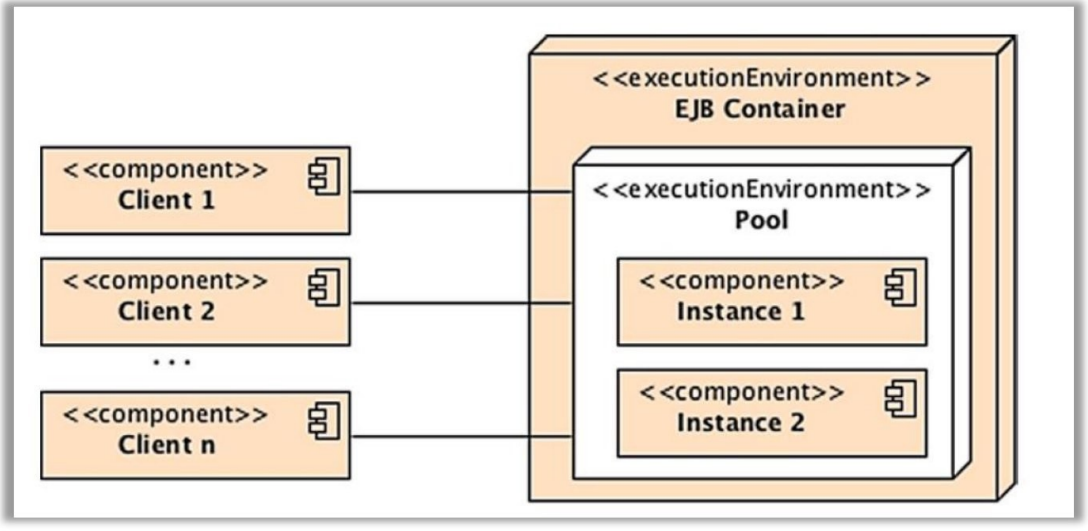
- Remote client communication
- Dependency Injection
- State management
- Pooling
- Component life cycle
- Transaction management
- Security
- Concurrency support

Writing Enterprise Java Beans

```
@Stateless
public class BookEJB {

  @PersistenceContext(unitName = "chapter07PU")
  private EntityManager em;

  public Book findBookById(Long id) {
    return em.find(Book.class, id);
  }

  public Book createBook(Book book) {
    em.persist(book);
    return book;
  }
}
```

Stateless EJB

- تستخدم في المهام التي يمكن تنفيذها في one method call ولا تحتوي على conversational state وأي instance منها يمكن استخدامه من قبل أي client .



Example

- المثال التالي يبين كيفية انشاء آلة حاسبة بسيطة باستخدام Stateless Session Bean .

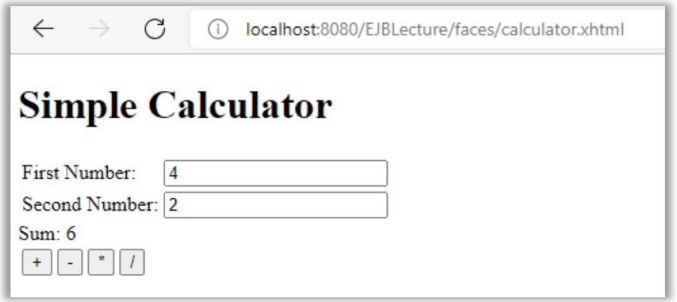
```
@Stateless
public class CalculatorEJB {

    public int addition(int n1, int n2) {
        return n1 + n2;
    }

    public int subtraction(int n1, int n2) {
        return n1 - n2;
    }

    public int multiplication(int n1, int n2) {
        return n1 * n2;
    }

    public int devision(int n1, int n2) {
        return n1 / n2;
    }
}
```

Example

```java
@SessionScoped
public class CalculatorBean implements Serializable {

    private int firstNumber;
    private int secondNumber;
    private int sum;

    @Inject
    CalculatorEJB calculatorEJB;

    public CalculatorBean() {
    }

    public int getFirstNumber() {
        return firstNumber;
    }

    public void setFirstNumber(int firstNumber) {
        this.firstNumber = firstNumber;
    }

    public int getSecondNumber() {
        return secondNumber;
    }

    public void setSecondNumber(int secondNumber) {
        this.secondNumber = secondNumber;
    }

    public int getSum() {
        return sum;
    }

    public void addition() {
        this.sum = calculatorEJB.addition(firstNumber, secondNumber);

    }

    public void subtraction() {
        this.sum = calculatorEJB.subtraction(firstNumber, secondNumber);

    }

    public void multyplication() {
        this.sum = calculatorEJB.multiplication(firstNumber, secondNumber);

    }

    public void devision() {
        this.sum = calculatorEJB.devision(firstNumber, secondNumber);

    }
```

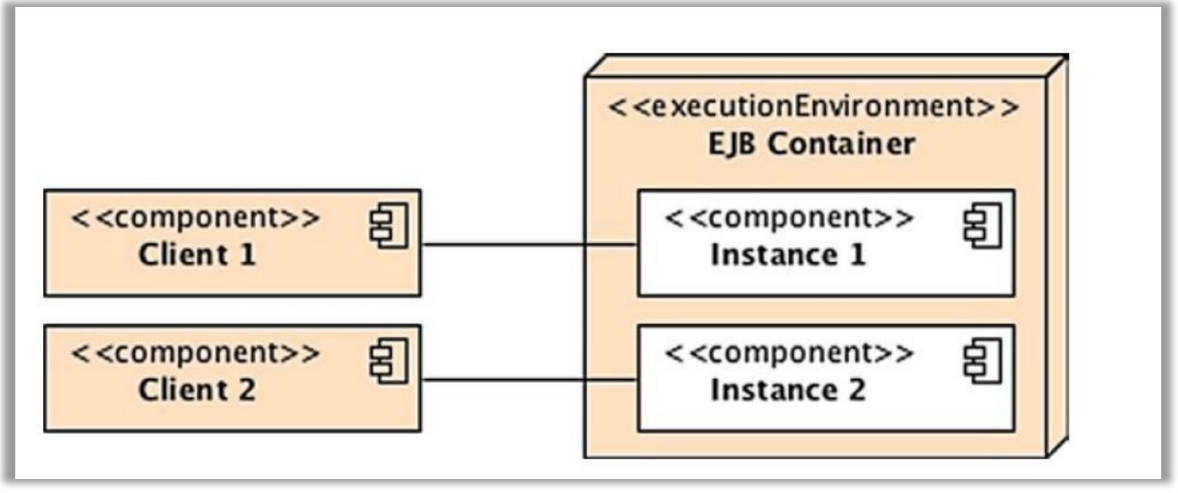
Example

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets">
    <h:head>
        <title>Facelet Title</title>
    </h:head>
    <h:body>
        <h1>
            <h:outputText  value="Simple Calculator"/>
        </h1>
        <h:messages />
        <h:form>
            <h:panelGrid columns="2">
                <h:outputText value="First Number: "    />
                <h:inputText value="#{calculatorBean.firstNumber}" />
                <h:outputText value="Second Number: " />
                <h:inputText value="#{calculatorBean.secondNumber}" />
            </h:panelGrid>

            <h:outputText value="Sum: #{calculatorBean.sum}"    />


            <h:panelGrid columns="4">
                <h:commandButton  value="+" action="#{calculatorBean.addition()}"   />
                <h:commandButton  value="-" action="#{calculatorBean.subtraction()}"  />
                <h:commandButton  value="*" action="#{calculatorBean.multyplication()}"  />
                <h:commandButton  value="/" action="#{calculatorBean.devision()}"  />
            </h:panelGrid>

        </h:form>
    </h:body>
</html>
```



Stateful EJB

- تستخدم في المهام التي تحتوي على conversational state ونحتاج الاحتفاظ بها خلال عدة methods call لمستخدم واحد وكل instance يستخدمه client واحد.



Example

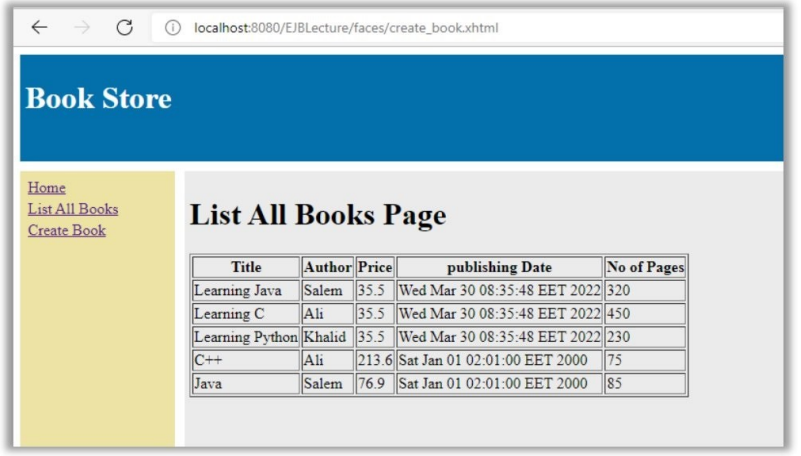
- المثال التالي يبين كيفية استخدام Stateful Session Bean .

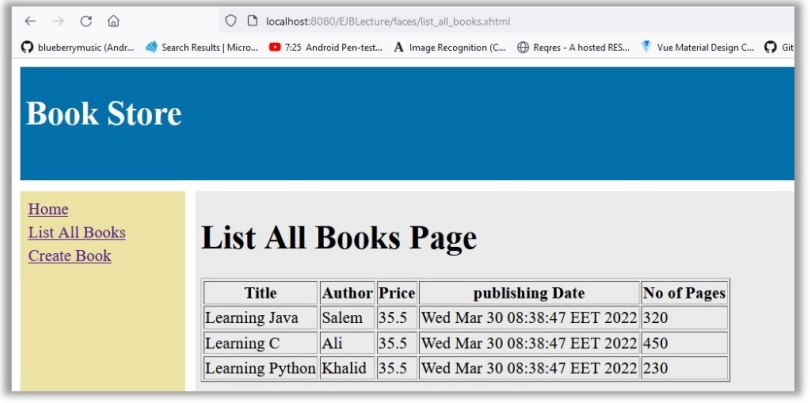
```
@Stateful
public class BookEJB {

    private List<Book> books;

    @PostConstruct
    public void init() {
        books = new ArrayList<>();
        books.add(new Book("Learning Java ", "Salem", 35.5f, new Date(), 320, "History"));
        books.add(new Book("Learning C ", "Ali", 35.5f, new Date(), 450, "Comics"));
        books.add(new Book("Learning Python ", "Khalid", 35.5f, new Date(), 230, "Scifi"));

    }

    public List<Book> getBooks() {
        return books;
    }

    public void createBook(Book book) {
        books.add(book);
    }

     public void deleteBook(Book book){
        books.remove(book);
    }

}
```

Example

```
@Named(value = "bookListBean")
@SessionScoped
public class BookListBean implements Serializable{


    @Inject
    BookEJB bookEJB;


    public BookListBean() {
    }

    public List<Book> getBooks() {
        return bookEJB.getBooks();
    }


    public String createBook(Book book){
        bookEJB.createBook(book);
        return"list_all_books";
    }

    public String deleteBook(Book book){
        bookEJB.deleteBook(book);
        return"list_all_books";
    }
}
```

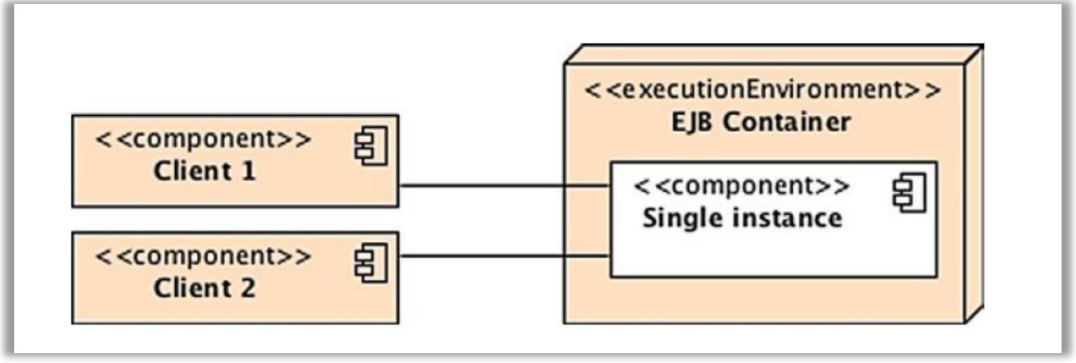
Example


localhost:8080/EJBLecture/faces/create_book.xhtml
Book Store
Home
List All Books
Create Book
List All Books Page
Title Author Price publishing Date No of Pages
Learning Java Salem 35.5 Wed Mar 30 08:35:48 EET 2022 320
Learning C Ali 35.5 Wed Mar 30 08:35:48 EET 2022 450
Learning Python Khalid 35.5 Wed Mar 30 08:35:48 EET 2022 230
C++ Ali 213.6 Sat Jan 01 02:01:00 EET 2000 75
Java Salem 76.9 Sat Jan 01 02:01:00 EET 2000 85


localhost:8080/EJBLecture/faces/list_all_books.xhtml
Book Store
Home
List All Books
Create Book
List All Books Page
Title Author Price publishing Date No of Pages
Learning Java Salem 35.5 Wed Mar 30 08:38:47 EET 2022 320
Learning C Ali 35.5 Wed Mar 30 08:38:47 EET 2022 450
Learning Python Khalid 35.5 Wed Mar 30 08:38:47 EET 2022 230

Singlton EJB

- تتم مشاركتها بين clients وهي تدعم concurrency ويوجد منها one instance في التطبيق الواحد.



Example

- المثال التالي يبين كيفية استخدام Singleton Session Bean .

```java
@Singleton
public class CityEJB {

    private List<String> cities;

    @PostConstruct
    public void init() {
        cities = new ArrayList<>();
        cities.add("Tripoli");
        cities.add("Benghazi");
        cities.add("Sabha");
        cities.add("Zawia");
        cities.add("Musrata");

    }

    public List<String> getCities() {
        return cities;
    }

}
```

Example

```
@Named()
@RequestScoped
public class CityBean implements Serializable {



    @Inject
    CityEJB cityEJB;

    public CityBean() {
    }

    public List<String> getCities() {
        return cityEJB.getCities();
    }
}
```

Example

```
<html xmlns="http://www.w3.org/1999/xhtml"
      xmlns:h="http://xmlns.jcp.org/jsf/html"
      xmlns:ui="http://xmlns.jcp.org/jsf/facelets"
      xmlns:f="http://xmlns.jcp.org/jsf/core">
    <h:head>
        <title>Facelet Title</title>
    </h:head>
    <h:body>
        <h1>
            <h:outputText  value="Show Cities List Page"/>
        </h1>
        <h:form>
            <h:panelGrid columns="2">

                <h:outputLabel value="City: "/>
                <h:selectOneMenu value="">
                    <f:selectItems value="#{cityBean.cities}"  />
                </h:selectOneMenu><br/>
            </h:panelGrid>
        </h:form>
    </h:body>
</html>
```