



巨匠線上真人

Java Web OCE JWCD元件系統 開發認證

www.pcschoolonline.com.tw



巨匠線上真人

Java Web OCE JWCD元件系統開發認證

第十三堂：

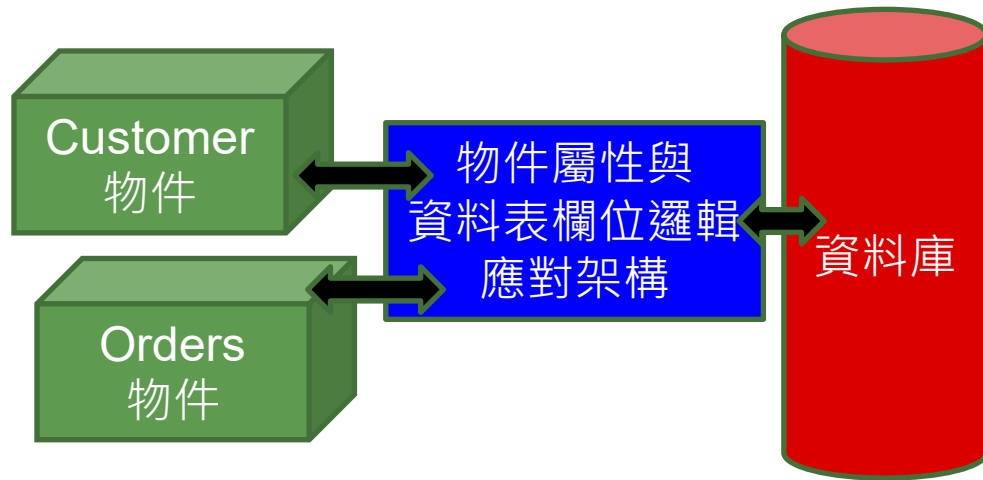
Java網站資料庫存取策略-ORM軟體工程

本堂教學重點

- ◆ JPA實現ORM軟體工程架構說明
- ◆ 配置JPA Config與設計Entity Class進行Table Mapping
- ◆ 使用EntityManager進行資料增修與查詢的DAO元件開發

何謂ORM軟體工程

- ◆ ORM物件關聯對映(Object Relational Mapping)，用於實現物件導向程式語言裡不同類型的資料之間的轉換。
- ◆ 如可建立一個「虛擬物件導向資料庫」，方便使用物件導向設計概念，進行關聯式資料庫的維護。
- ◆ 物件導向是從軟體工程基本原則（如耦合、聚合、封裝）的基礎上發展起來的，而關聯式資料庫則是從數學理論發展而來的，兩套理論存在顯著的區別。為了解決這個不匹配的現象，物件關聯對映技術應運而生。



JPA實現ORM軟體工程

- ◆ JPA(Java Persistence API)，為JEE支援的API，具有三個層面。
 - ◆ API
 - ◆ Java Persistence查詢語言JPQL
 - ◆ OR/M
- ◆ 映射或持久性(Persistence)的階段，包括JPA提供者，可以使用EclipseLink，Toplink，Hibernate等。
- ◆ JPA使用@註解敘述或XML Config描述OR/M映射關係，並且將運行的實體(Entity)對象持久化到資料庫中。

資料庫與物件導向操作

- ◆ 資料庫我們則採用Schem中的Primary Key/Unique Key作為識別，定且使用FK連接架構進行資料之間的Constraints維護作業。
 - ◆ Data Field Property
 - ◆ Constraints-PK/UK/FK...
 - ◆ SQL Statement
- ◆ 物件我們則使用setter and getter表示資料內容的存取，並且透過Method進行資料的維護作業。
- ◆ 其中著重在功能規範上的介面或者類別規劃的DI(Dependency Injection)進行關聯性。
 - ◆ Interface
 - ◆ Setter and Getter-Property
 - ◆ Method操作

配置JPA Config 與設計Entity Class進行Table Mapping

在Eclipse 安裝
TOMEE



建立專案配置
TOMEE



規劃Entity Class

◆ 線上下載apache-tomee-webprofile

◆ <https://tomee.apache.org/download-ng.html>

Apache TomEE		Documentation Community Security Downloads				
						SHA256 SHA512
TomEE plus	8.0.0	16 Sep 2019	58 MB	ZIP		ZIP SHA256 SHA512
TomEE webprofile	8.0.0	16 Sep 2019	41 MB	TAR.GZ		TAR.GZ SHA256 SHA512
TomEE webprofile	8.0.0	16 Sep 2019	41 MB	ZIP		ZIP SHA256 SHA512
TomEE microprofile	8.0.0	16 Sep 2019	44 MB	TAR.GZ		TAR.GZ SHA256 SHA512

Tomcat 與 TomeEE 的差別

- ◆ **Tomcat** 是支援servlet和JSP技術的servlet容器
 - ◆ Java Servlet規範
 - ◆ Java ServerPages (JSP)
 - ◆ 表達語言 (EL)
- ◆ **TomEE** 比 **Tomcat**更廣泛 支援許多其他Java EE技術
 - ◆ **TomEE** 包含：CDI-Apache/OpenwebBeans/EJB-Apache/OpenEJB/JPA-Apache/OpenJPA/JSF-Apache/MyFaces JSP-Apache Tomcat JSTL-/Apache Tomcat/JTA-Apache Geronimo Transaction等

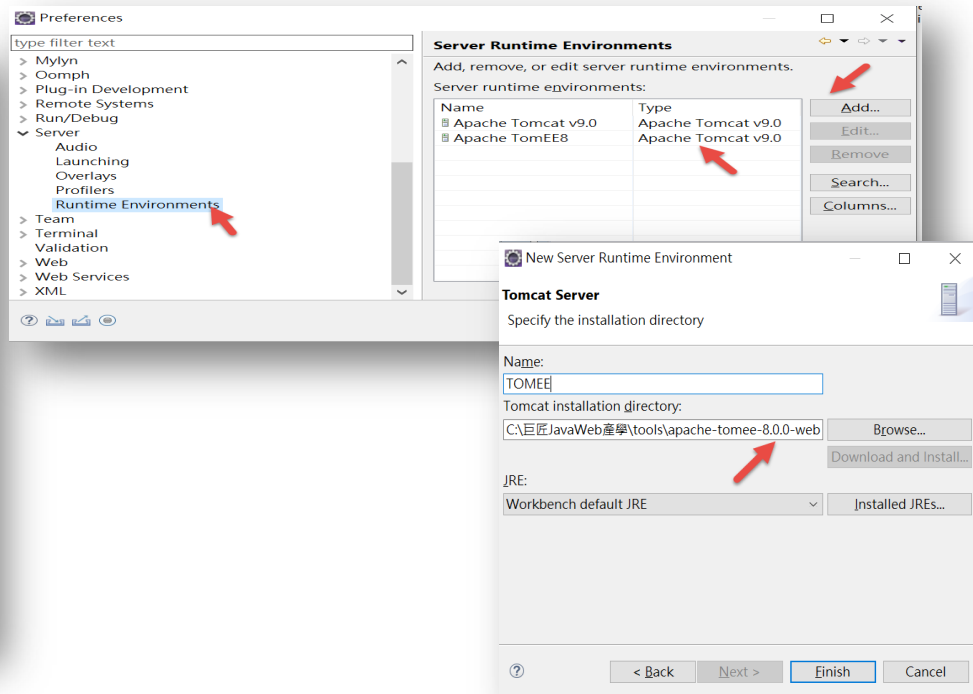
在Eclipse 安裝TOMEE

◆ 解壓下載TOMEE檔案

bin	2019/9/13 下午 12:30
conf	2019/9/13 下午 12:30
lib	2019/9/13 下午 12:30
logs	2019/7/4 下午 03:20
temp	2019/7/4 下午 03:20
<input type="checkbox"/> webapps	2019/9/13 下午 12:30
work	2019/7/4 下午 03:20
BUILDING.txt	2019/7/4 下午 03:20
CONTRIBUTING.md	2019/7/4 下午 03:20
LICENSE	2019/9/13 下午 12:12
NOTICE	2019/9/13 下午 12:12
README.md	2019/7/4 下午 03:20
RELEASE-NOTES	2019/7/4 下午 03:20
RUNNING.txt	2019/7/4 下午 03:20

◆ Eclipse EE安裝TOMEE

◇ 功能表 Window/Preferences/Server



建立Web專案配置TOMEE

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:

Project location
☒ Use default location
Location:

Target runtime

Dynamic web module version

Configuration

A good starting point for working with Apache TomEE8 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:

規劃客戶資料表

- ◆ MySQL範例資料庫sakila，建立customers客戶資料表。
- ◆ 需要設定Primary key:customerid
- ◆ 需要配置欄位是否Mandatory，強制輸入(not or not null)

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
customerid	CHAR(5)	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
companyname	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
address	VARCHAR(80)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
phone	VARCHAR(30)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL
country	VARCHAR(20)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	NULL

- ◆ CREATE TABLE `customers`
(`customerid` char(5) NOT NULL,
`companyname` varchar(45) NOT
NULL, `address` varchar(80) DEFAULT
NULL, `phone` varchar(30) DEFAULT
NULL, `country` varchar(20) DEFAULT
NULL, **PRIMARY KEY (`customerid`))**
ENGINE=InnoDB DEFAULT
CHARSET=utf8;

建立Entity Class Mapping Table

- ◆ 建立一個Entity Class，採用相關的Annotation進行描述，應對相對資料表特徵。
 - ◆ 必須具有一個@Id描述映對資料表 Primary Key或者Unique Key欄位
 - ◆ 需要使用@Entity設定類別應對資料表。如果Table名稱無法應對Class Name，必須使用@Table進行映對描述。
 - ◆ 欄位名稱無法與定義的Attribute一致，需要使用@Column進行欄位名稱映對描述。

Customer.java

```
@Entity
@Table(name = "customers")
@XmlRootElement
public class Customer implements Serializable {

    @Id
    @Column(name = "customerid")
    @NotNull
    private String customerId;

    @Column(name = "companyname")
    @Size(max = 45)
    @NotNull
    private String companyName;

    @Size(max = 80)
    @Column(name = "address")
    private String address;

    @Size(max = 30)
    @Column(name = "phone")
    private String phone;

    @Size(max = 20)
    @Column(name = "country")
    private String country;
```


getter 與 setter 與 Overriding equals/hashCode

- ◆ 將每一個Attribute產生setter and getter。
- ◆ Overriding 識別用的@Id欄位，覆寫equals 與hashCode Method。


```
public String getCountry() {  
    return country;  
}
```

```
public void setCountry(String country) {  
    this.country = country;  
}
```

```
@Override  
public int hashCode() {  
    final int prime = 31;  
    int result = 1;  
    result = prime * result + ((customerId == null) ? 0 : customerId.hashCode());  
    return result;  
}
```

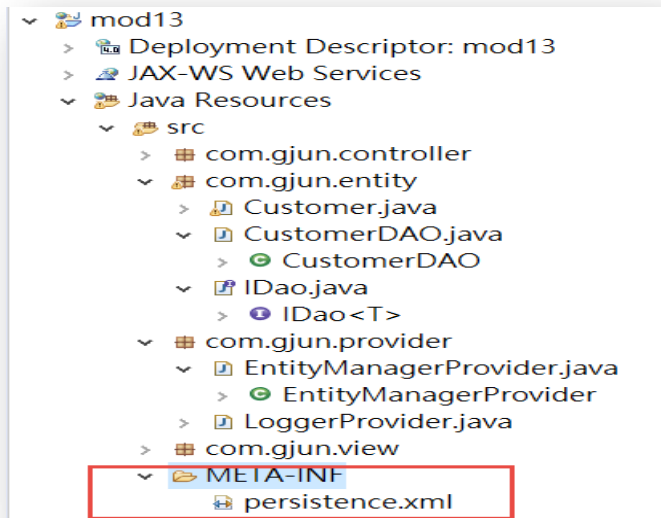


```
@Override  
public boolean equals(Object obj) {  
    if (this == obj)  
        return true;  
    if (obj == null)  
        return false;  
    if (getClass() != obj.getClass())  
        return false;  
    Customer other = (Customer) obj;  
    if (customerId == null) {  
        if (other.customerId != null)  
            return false;  
    } else if (!customerId.equals(other.customerId))  
        return false;  
}
```



佈署JPA 組態檔 persistence.xml

- ◆ 於專案 java source/src建立一個 META-INF資料夾。
- ◆ 建立名稱為persistence.xml檔案



```
<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.0"
  xmlns="http://java.sun.com/xml/ns/persistence"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://java.sun.com/xml/ns/persistence
http://java.sun.com/xml/ns/persistence/persistence_2_0.xsd">
  <persistence-unit name="mod13">
    <class>com.gjun.entity.Customer</class>
    <properties>
      <property name="openjpa.RuntimeUnenhancedClasses"
value="supported"/>
      <property name="javax.persistence.jdbc.url"
value="jdbc:mysql://localhost:3306/sakila?serverTimezone=UTC&useU
nicode=true&characterEncoding=utf8&useSSL=false" />
      <property name="javax.persistence.jdbc.driver"
value="com.mysql.cj.jdbc.Driver" />
      <property name="javax.persistence.jdbc.user" value="root" />
      <property name="javax.persistence.jdbc.password" value="1111" />
      <property name="openjpa.jdbc.DBDictionary" value="mysql" />
      <property name="openjpa.Log" value="SQL=Trace" />
      <property name="openjpa.jdbc.SynchronizeMappings"
value="buildSchema(ForeignKeys=true)/>
    </properties>
  </persistence-unit>
</persistence>
```

persistence.xml

- ◆ Java EE中應用中，Persistence在每個模組中具有唯一的命名。
- ◆ 主要用來給Persistence產生一個EntityManagerFactory配置的一句模組使用。
- ◆ <class>配置Mapping Table描述的Entity Class。
- ◆ <properties>元素，提供了一種指定其他JPA屬性的方法，包括標準JPA屬性（例如JDBC連接字符串，用戶名和密碼或架構生成設置）以及提供程序特定的屬性（例如Hibernate設置）。您可以使用嵌套元素指定一個或多個屬性，每個元素都具有名稱和值屬性。

```
<persistence-unit name="mod13">
  <class>com.gjun.entity.Customer</class>
  <properties>
    <property name="openjpa.RuntimeUnenhancedClasses" value="supported"/>
    <property name="javax.persistence.jdbc.url" value="jdbc:mysql://localhost:3306/sakila?serverTimezone=UTC&us
    <property name="javax.persistence.jdbc.driver" value="com.mysql.cj.jdbc.Driver" />
    <property name="javax.persistence.jdbc.user" value="root" />
    <property name="javax.persistence.jdbc.password" value="1111" />
    <property name="openjpa.jdbc.DBDictionary" value="mysql" />
    <property name="openjpa.Log" value="SQL=Trace" />
    <property name="openjpa.jdbc.SynchronizeMappings" value="buildSchema(ForeignKeys=true)"/>
  </properties>
</persistence-unit>
```

使用EntityManager 進行資料增修與查詢的DAO元件開發

◆ JPA架構

Persistence

- 配合persistence.xml unitName產生一個EntityManagerFactory物件

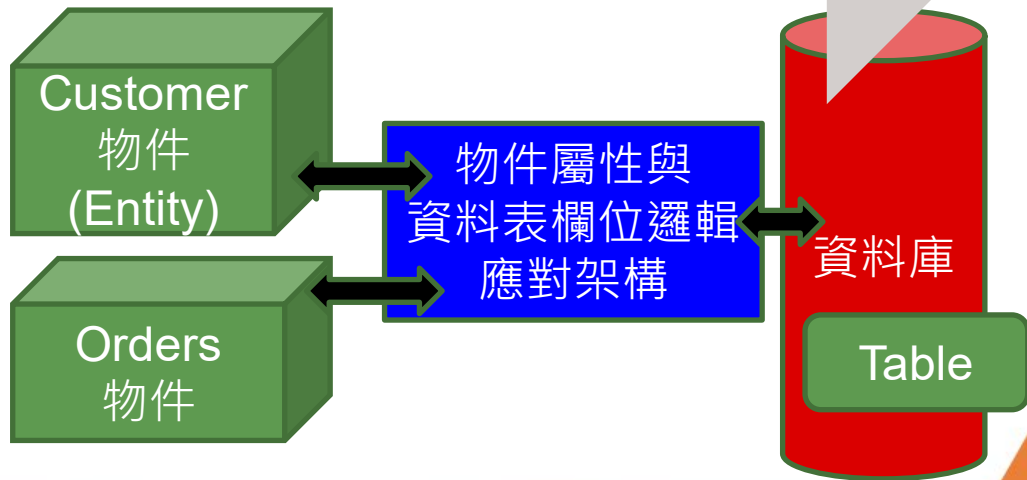
EntityManagerFactory產生
EntityManager物件進行
PersistenceContext管理

EntityManager操作即訂方法操作物件
增刪修至PersistenceContext物件

- 同步異動資料表相對記錄

EntityManager產生Query物件

- 執行JPQL進行PersistenceContext物件維護
- 同步異動資料表相對記錄



JPA類型說明

類型	說明
Persistence	用於在Java SE環境中獲取EntityManagerFactory物件，配合persistence.xml unit name。
EntityManagerFactory	該介面用於實體管理EntityManager工廠，可以產生一個管理持久層環境的個體物件(EntityManager)。
EntityManager	這是一個介面，它管理的持久化(PersistenceContext)。用來操作物件(Entity)與同步資料庫相對資料表記錄的維護。
Query	用於控制查詢執行的介面接口，可以執行JPQL。

ORM DAO設計模式

- ◆ 規劃DAO介面中增刪修與查詢方法規格。
- ◆ 注入EntityManager物件進行物件(Entity)維護與同步資料庫資料更新作業。

IDao.java

```
package com.gjun.entity;

import java.util.List;

public interface IDao<T> {
    //查詢
    public T select(Object key);
    //多筆查詢
    public List<T> selectAll();
    //新增
    public boolean insert(T object);
    //DI 注入EntityManager Interface
    public void setEntityManager(EntityManager entityManager);
}
```

CustomerDAO實作

CustomerDao.java

- ◆ 透過EntityManager操作對PersistenceContext(需擬物件導向資料庫)資料維護與查詢。
- ◆ 注入應用系統配合persisntece.xml unit name產生的EntityManager物件，形成與DAO物件的依賴注入關係(DI)。

```
//實作IDao介面
public class CustomerDAO implements IDao<Customer>{
    //attribute
    private EntityManager entityManager;

    @Override
    public Customer select(Object key) {
        //查詢
        return this.entityManager.find(Customer.class, key.toString());
    }

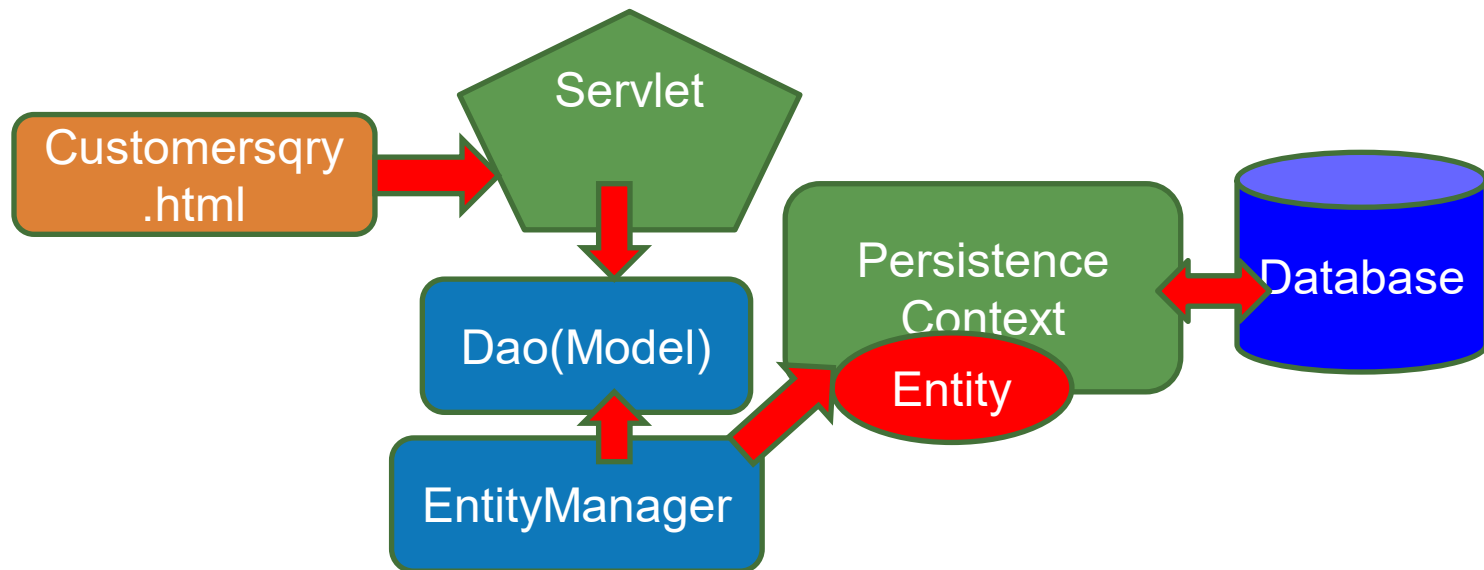
    @Override
    @SuppressWarnings("unchecked")
    public List<Customer> selectAll() {
        List<Customer> result=
            (List<Customer>)this.entityManager.createQuery("Select t from Customer t").getResultList();
        return result;
    }

    @Override
    public boolean insert(Customer object) {
        this.entityManager.persist(object); //新增記錄
        return true;
    }

    @Override
    public void setEntityManager(EntityManager entityManager) {
        this.entityManager=entityManager;
    }
}
```

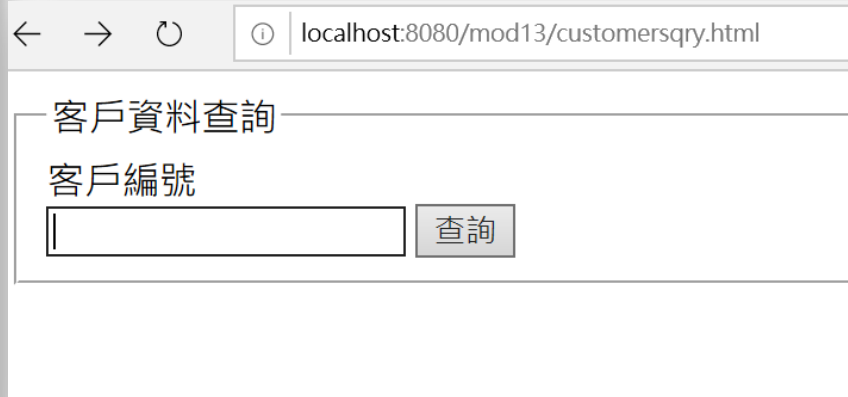
JPA查詢實作

- ◆ 查詢表單頁面設計/使用JPA DAO Model進行查詢的Servlet撰寫



JPA查詢實作-Form Page

```
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>客戶資料查詢</title>
</head>
<body>
  <fieldset>
    <legend>客戶資料查詢</legend>
    <form method="post" action="CustomerQryServLet">
      <div>客戶編號</div>
      <input type="text" name="cid"/>
      <input type="submit" value="查詢"/>
    </form>
  </fieldset>
</body>
</html>
```



The screenshot shows a web browser window with the address bar displaying "localhost:8080/mod13/customersqry.html". The page content includes a title "客戶資料查詢" and a form with the label "客戶編號". The form contains a text input field and a "查詢" (Query) button.

JPA查詢實作-Servlet Controller設計-1

- ◆ 透過Persistence.createEntityManagerFactory("unitname")，建構一個EntityManagerFactory物件。
- ◆ 透過EntityManagerFactory建立一個管理PersistenceContext的EntityManager物件
- ◆ 注入EntityManager物件到自訂的DAO物件裡。

CustomerQryServlet.java

```
@WebServlet("/CustomerQryServlet")
public class CustomerQryServlet extends HttpServlet {
    private static final long serialVersionUID = 1L;

    protected void doPost(HttpServletRequest request, HttpServletResponse response)
        throws ServletException, IOException {
        //使用Persistence 產生工廠 配置UnitName
        String customerid=request.getParameter("cid");
        EntityManagerFactory factory=Persistence.createEntityManagerFactory("mod13");
        //去一個管理這一個PersistenceContext的管理員
        EntityManager entityManager=factory.createEntityManager();
        //建構Dao物件
        IDao<Customer> dao=new CustomerDAO();
        dao.setEntityManager(entityManager);
    }
}
```

JPA查詢實作-Servlet Controller設計-2

showcustomers.jsp

//查詢作業

```
Customer customer=dao.select(customerid);
if(customer!=null) {
    request.setAttribute("result",customer);
    //Dispatcher
    request.getRequestDispatcher("showcustomers.jsp").forward(request, response);
}else
{
    String msg=String.format("查無客戶編號:%s 記錄!",customerid);
    request.setAttribute("msg", msg);
    //Dispatcher
    request.getRequestDispatcher("notfound.jsp").forward(request, response);
}
```

```
<%@ page language="java" contentType="text/html; charset=UTF-8"
    pageEncoding="UTF-8"%>
<%@taglib prefix="c" uri="http://java.sun.com/jsp/jstl/core" %>
<!DOCTYPE html>
<html>
<head>
<meta charset="UTF-8">
<title>查詢結果</title>
</head>
<body>
    <c:if test="${result!=null}">
        <c:if test="${result!=null}">
            <fieldset>
                <legend>查詢結果</legend>
                <table border="1" width="100%">
                    <tr>
                        <td>客戶編號</td>
                        <td>${result.customerId}</td>
                    </tr>
                    <tr>
                        <td>公司行號</td>
                        <td>${result.companyName}</td>
                    </tr>
                    <tr>
                        <td>聯絡地址</td>
                        <td>${result.address}</td>
                    </tr>
                    <tr>
                        <td>連絡電話</td>
                        <td>${result.phone}</td>
                    </tr>
                    <tr>
                        <td>國家別</td>
                        <td>${result.country}</td>
                    </tr>
                </table>
            </fieldset>
        </c:if>
```

Demo

← → ↻ ⓘ localhost:8080/mod13/customersqry.html

客戶資料查詢

客戶編號

ANTON × 查詢



← → ↻ ⓘ localhost:8080/mod13/CustomerQryServlet

查詢結果

客戶編號	ANTON
公司行號	中華電信
聯絡地址	台北市仁愛路
連絡電話	02-34561234
國家別	中華民國

Lab

- ◆ 敘述一下JPA架構以及如何實踐資料存取
- ◆ 使用JPA設計一個客戶資料查詢範例
- ◆

← → ↻ ⓘ localhost:8080/mod13/customersqry.html

客戶資料查詢

客戶編號

ANTON × 查詢



← → ↻ ⓘ localhost:8080/mod13/CustomerQryServlet

查詢結果

客戶編號	ANTON
公司行號	中華電信
聯絡地址	台北市仁愛路
連絡電話	02-34561234
國家別	中華民國