# **Deploying and running JPA application client**

The Java Persistence API (JPA) is a new programming model under EJB3.0 specification (JSR220) for the management of persistence and object /relational mapping with Java EE and Java SE. With JPA, developers can easily develop java applications that perform operations on relational database management systems using java objects and mapping. In that way, java applications developed using JPA are not only portable across different platforms, but also applications can be easily developed using simple yet powerful programming model provided by JPA. This greatly improves application maintainability against ever changing database world. JPA insulates applications from all the complexity and non-portable boilerplate code involved in database connectivity and operations.

Apache Geronimo uses Apache OpenJPA for providing Java Persistence API implementation to Java EE applications deployed in the server. Even though JPA is a part of EJB3 spec, it is independent of it. Hence, JPA can be used in JavaSE, web and ejb applications in the same uniform way. Think of a JPA implementation, e.g. Apache OpenJPA as a driver for JPA (JPA driver) similarly to what a JDBC driver is for JDBC.

The below section illustrates developing and deploying a JEE application client that uses JPA to access and manipulate database data residing in an embedded derby database.

In order to develop, deploy and run the application, the following environment is required.

- Sun JDK 5.0+ (J2SE 1.5)
- Eclipse 3.3.1.1 (Eclipse Classic package of Europa distribution), which is platform specific
- Web Tools Platform (WTP) 2.0.1
- Data Tools Platform (DTP) 1.5.1
- Eclipse Modeling Framework (EMF) 2.3.1
- Graphical Editing Framework (GEF) 3.3.1

The article has the following sections.

- Setting up Eclipse for Application development
- Developing Entities and Client
- Preparing Deployment Descriptors and Deployment Plans
- Deploying the application client
- Running the application client

To download the complete application, click on link

# Setting Eclipse for Application development

1. Start the Eclipse IDE and create ApplicationClientJPA java project. Right click on the *Project Explorer* Window and click on *New > Java Project*.

Edit Source Re	factor Navigate Seard ≫ •	h Project Run Wir	ndow Help
	New	,	/ Java Project
	Show In	Alt+Shift+W	🏫 Project
	Copy	Ctrl+C	Package
	Paste	Ctrl+V Delete	Interface     Enum
	Build Path	:►	Annotation
	≥ Import ≧ Export		Folder File
	🔗 Refresh	F5	
	Properties	Alt+Enter	JUnit Test Case
			Example
			= the other

 $\label{eq:provide ApplicationClientJPA} \text{ as the name for the Java Project}.$ 

Create a Java project in the workspace or in an external location.  Project name: ApplicationClientJPA Contents O Create new project in workspace O Create project from existing source Directory: C:\bm\edipse201\ws-temp\ApplicationClientJPA Br JRE O Use default JRE (Currently 'SDK-May-31-2007') O Use a project specific JRE: SDK-May-31-2007 O Use an execution environment JRE: J2SE-1.5	the state of the s		
Project name: ApplicationClientJPA Contents O Create new project in workspace O Create project from existing source Directory: C:\jbm\edipse201\ws-temp\ApplicationClientJPA Br JRE O Use default JRE (Currently 'SDK-May-31-2007) O Use a project specific JRE: SDK-May-31-2007 O Use an execution environment JRE: J2SE-1.5	ite a Java project in the workspace or in an e	external location.	
Contents         Image: Create new project in workspace         Create project from existing source         Directory:       C:\bm\edipse201\ws-temp\ApplicationClientJPA         JRE         Image: Ouse default JRE (Currently 'SDK-May-31-2007')         Configure         Ouse a project specific JRE:         SDK-May-31-2007         Ouse an execution environment JRE:	ect name: ApplicationClientJPA		
<ul> <li>Create new project in workspace</li> <li>Create project from existing source</li> <li>Directory: C:\bm\eclipse201\ws-temp\ApplicationClientJPA</li> <li>Br</li> <li>JRE</li> <li>Use default JRE (Currently 'SDK-May-31-2007')</li> <li>Configure</li> <li>Use a project specific JRE: SDK-May-31-2007</li> <li>Use an execution environment JRE: J2SE-1.5</li> </ul>	intents		
O Create project from existing source         Directory:       C:\bm\edipse201\ws-temp\ApplicationClientJPA         JRE         ③ Use default JRE (Currently 'SDK-May-31-2007')         ○ Use a project specific JRE:         SDK-May-31-2007         ○ Use an execution environment JRE:	) Create new project in workspace		
Directory:       C:\bm\edipse201\ws-temp\ApplicationClientJPA       Br         JRE       Ouse default JRE (Currently 'SDK-May-31-2007')       Configure         Ouse a project specific JRE:       SDK-May-31-2007       V         Ouse an execution environment JRE:       J2SE-1.5       V	) Create project from existing source		
JRE         ③ Use default JRE (Currently 'SDK-May-31-2007')         〇 Use a project specific JRE:         SDK-May-31-2007         〇 Use an execution environment JRE:	rectory: C:\bm\edipse201\ws-temp\Applic	cationClientJPA	Browse
Ouse default JRE (Currently 'SDK-May-31-2007')       Configure         Ouse a project specific JRE:       SDK-May-31-2007         Ouse an execution environment JRE:       J2SE-1.5	E		
O Use a project specific JRE:     SDK-May-31-2007       O Use an execution environment JRE:     J2SE-1.5	) Use default JRE (Currently 'SDK-May-31-20	007')	Configure default
O Use an execution environment JRE: J2SE-1.5	) Use a project specific JRE: SDK	-May-31-2007 😪	
	) Use an execution environment JRE: J2SE	E-1.5	
Project layout	oject layout		
O Use project folder as root for sources and class files	) Use project folder as root for sources and	class files	
Create separate folders for sources and class files <u>Configure</u>	) Create sep <mark>a</mark> rate folders for sources and da	ass files	Configure default
Working sets	orking sets		
Add project to working sets	Add project to working sets		

Click on Finish.

Right click on the Project and click on the Properties option which is the last option in the menu.



This will open up Properties Window. Click on Java Build Path => Libraries tab on the right hand side.



Click on Add External JARs button and browse for geronimo-jpa\_3.0\_spec-1.1.1. jar file which should be in <geronimo\_home>/repository /org/apache/geronimo/specs/geronimo-jpa\_3.0\_spec/1.1.1. The jar file is only to compile entity and entity client classes you're about to create. Geronimo provides the jar at runtime. Select the file and click on the *Open* button.

ype filter text	Java B	uild Path	(p + c)
Resource BeanInfo Path	🗁 So	urce 🔁 Projects 🛋 Libraries 😽 Ord	der and Export
Builders	JARs a	nd class folders on the build path:	
∃ava Build Path ⊕ Java Code Style	Đ 🔁	🛔 JRE System Library [SDK-May-31-2007]	Add JARs
∃ava Compiler ∃ava Editor			Add External JARs
Java JAR Selection			? 🔀
Profi Proje Look in:	1.1.1	▼ (3)	🌶 📂 🛄+
Refa	eronimo-jpa	3.0 spec-1, 1, 1, jar	
- Serv			
Valid Documents			
1			
Desktop			
-			
My Documents			
-			
My Computer			
My Computer	0		
My Computer	File name:	geronimo-jpa_3.0_spec-1.1.1.jar	V Open

This will add the library to the build path. Click on OK.

Properties for ApplicationC	lientJPA	
type filter text	Java Build Path	4 4
Resource BeanInfo Path	😕 Source 🗁 Projects 🛋 Libraries 🗞 Order and Ex	cport
Java Build Path Java Code Style Java Compiler Java Editor Javadoc Location Profile Compliance and Validatic Project References Refactoring History Run/Debug Settings Server Task Tags Validation	Image: Series and class force is on the ball page.         Image: Series and page.         I	Add <u>J</u> ARs Add E <u>x</u> ternal JARs Add <u>V</u> ariable Add Libr <u>a</u> ry Add <u>C</u> lass Folder <u>E</u> dit <u>R</u> emove <u>Migrate JAR File</u>
K		
0		DK Cancel

2. Create META-INF folder in the project. Right click on the project and click on New > Folder

🛢 Java - Ec	lipse SDK	-	
File Edit So	urce Refactor Navigate S	Gearch Project F	Run Window Help
i 📬 • 🖫	<u>⊜</u>   ☆ • Q • Q •	: 😕 🖶 🞯 •	· 1 🥭 🛷 1 🎯 1 🗍 1 🖗
📕 Package E	xplor 🛛 🍃 Hierarchy		
	4 4 Q 🖻 🕯	• ▽	
🗆 🖂 Applic	ationClientJPA		-
🗁 🖽 s	New	)	🕨 🏄 Java Project
Ē 🛃 J	Go Into	🌱 Project	
	Open in New Window Open Type Hierarchy Show In	F4 Alt+Shift+W	Package Class
	Copy	Ctrl+C	Enum     Minotation
	Paste	Ctrl+V	😰 Source Folder
	💢 Delete	Delete	😋 Folder
	Build Path Source Refactor	Alt+Shift+S Alt+Shift+T	File Untitled Text File DbUnit Test Case

Provide META-INF for the folder name and click on *Finish*.

New Folder		
Folder Create a new folder resource.		
Enter or select the parent folder:		
ApplicationClientJPA		
ApplicationClientJPA		
Folder name: META-INF		

That's all! the setup required for developing the application client is over.

# Developing Entities and the Client

1.Right click on the project and create a  ${\tt Account}$  java class as follows.

Janetter		-		
- Application-	New		Þ	/ Java Project
🗉 🛋 JRE Sy	Go Into			Project
META-	Open in New Window			Package
	Open Type Hierarchy	F4		Class
-	Show In Alt	Alt+Shift+W	1	🕜 Interface
	Сору	Ctrl+C	3	🕼 Enum
	Copy Qualified Name			@ Annotation
	💼 Paste	Ctrl+V		😰 Source Folder
	💢 Delete	Delete		😂 Folder
-	Build Path		F	😭 File
	Source	Alt+Shift+S	F	😭 Untitled Text File
	Refactor	Alt+Shift+T	•	E DbUnit Test Case
	Minport			📑 JUnit Test Case
	Export			📬 Example
-	🔗 Refresh	F5		😭 Other
	Close Project		85	

 $Provide \ name \ for \ the \ class \ as \ {\tt Account} \ with \ a \ package \ name \ as \ {\tt sample.jpa.appclient.} \ Click \ on \ Finish.$ 

Create a new Java	a dass.	C
ource folder:	ApplicationClientJPA/src	Browse
ackage:	Sample.jpa.appclient	Browse
Enclosing type:		Browse
lame:	Account	]
lodifiers:	public Odefault Oprivate Oprotected     abstract Innal static	
uperdass:	java.lang.Object	Browse
nterfaces:		Add
		Remove
Vhich method stub	s would you like to create?  public static void main(String] args)  Constructors from superclass  Inherited abstract methods	1
o you want to add	d comments as configured in the <u>properties</u> of the current proj Generate comments	ect?

2. Copy the below contents to the  ${\tt Account.java}.$ 

Account.java		

```
package sample.jpa.appclient;
import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.Id;
import javax.persistence.Table;
@Entity
@Table(name="Account1")
public class Account implements Serializable{
private static final long serialVersionUID = 1L;
@Id
public int accountNo;
public String name;
public String address;
public int branchCode;
public double balance;
public Account(){
 this.accountNo = 0;
 this.name = "DUMMY";
 this.address = "DUMMY";
 this.branchCode = 0;
}
public int getAccountNo() {
 return accountNo;
}
public void setAccountNo(int accountNo) {
 this.accountNo = accountNo;
}
public String getName() {
 return name;
}
public void setName(String name) {
 this.name = name;
 }
public String getAddress() {
 return address;
}
public void setAddress(String address) {
 this.address = address;
}
public int getBranchCode() {
 return branchCode;
}
public void setBranchCode(int branchCode) {
 this.branchCode = branchCode;
 }
public double getBalance() {
 return balance;
}
public void setBalance(double balance) {
 this.balance = balance;
}
}
```

3. Similarly, right click on the project and create another java class by name AccountClient.java. Also, provide the same package name sample.jpa.appclient.

4.Copy the below contents to the AccountClient.java.

```
AccountClient.java
package sample.jpa.appclient;
import java.util.List;
import javax.persistence.EntityManager;
import javax.persistence.EntityManagerFactory;
import javax.persistence.Persistence;
import javax.persistence.Query;
import javax.persistence.EntityTransaction;
public class AccountClient {
   private EntityManager em;
   public AccountClient() {
       EntityManagerFactory emf = Persistence.createEntityManagerFactory("JPA-App-Client");
       if (emf == null) {
           throw new IllegalStateException("EntityManagerFactory is unavailable");
        }
        em = emf.createEntityManager();
       if (em == null) {
           throw new IllegalStateException("EntityManager is unavailable");
        }
    }
   public static void main(String[] args) {
       AccountClient client = new AccountClient();
       String opt = args[0];
       if ("create".equals(opt)) {
           if (args.length != 6) {
                System.err.println("No values for accountNo, name, address, branchCode and balance. Exiting.");
                System.exit(0);
           } else {
               int accNo = Integer.parseInt(args[1]);
                String name = args[2];
                String address = args[3];
                int branchCode = Integer.parseInt(args[4]);
                double balance = Double.parseDouble(args[5]);
                client.createAccount(accNo, name, address, branchCode, balance);
           }
        } else if ("list".equals(opt)) {
           List<Account> accounts = client.listAccounts();
           for (Account account: accounts) {
                System.out.println("_
                                                                    __");
                System.out.println(account.getAccountNo());
                System.out.println(account.getName());
                System.out.println(account.getAddress());
                System.out.println(account.getBranchCode());
                System.out.println(account.getBalance());
                                                                        _____");
                System.out.println("
                System.out.println("");
            }
        } else if ("update".equals(opt)) {
           if (args.length != 3) {
                System.out.println("No values for accountNo and new balace value. Exiting.");
                System.exit(0);
           } else {
                int accNo = Integer.parseInt(args[1]);
```

```
double newbalance = Double.parseDouble(args[2]);
            client.updateAccountBalance(accNo, newbalance);
        }
    } else if (opt.equals("delete")) {
        if (args.length != 2) {
            System.out.println("No values for accountNo for delete. Exiting.");
            System.exit(0);
        } else {
            int accNo = Integer.parseInt(args[1]);
            client.deleteAccount(accNo);
        }
    }
    else {
        System.err.println("Unknown option (" + opt + ") selected");
    }
}
public Account createAccount(int accNo, String name, String address, int branchCode, double balance) {
    Account acc1 = em.find(Account.class, accNo);
    if (acc1 != null) {
        throw new IllegalArgumentException("Account already exists - account Number (" + accNo + ")");
    }
    Account acc = new Account();
    acc.setAccountNo(accNo);
    acc.setAddress(address);
    acc.setBalance(balance);
    acc.setBranchCode(branchCode);
    acc.setName(name);
    System.out.println("Persisting account entity (accNo = " + accNo + ")");
    EntityTransaction et = em.getTransaction();
    et.begin();
    em.persist(acc);
    et.commit();
    System.out.println("Persisted successfully account entity (accNo = " + accNo + ")");
    return acc;
}
@SuppressWarnings("unchecked")
public List<Account> listAccounts() {
    Query q = em.createQuery("SELECT a FROM Account a");
    List<Account> currList = q.getResultList();
    return currList;
}
public Account updateAccountBalance(int accNo, double newBalance) {
    Account acc = em.find(Account.class, accNo);
    if (acc == null) {
        throw new IllegalArgumentException("Account not found : Account Number (" + accNo + ")");
    }
    EntityTransaction et = em.getTransaction();
    et.begin();
    acc.setBalance(newBalance);
    et.commit();
    return acc;
}
public void deleteAccount(int accNo) {
    EntityTransaction et = em.getTransaction();
    et.begin();
    em.remove(em.getReference(Account.class, accNo));
   et.commit();
}
```

}

∕∖∖

The AccountClient obtains EntityManagerFactory object first and then creates an EntityManager object from the factory. The Entity Manager object thus obtained is Application Managed EntityManager object. The Persistence scope of the EntityManager is by default Extend ed. Since the application client runs in a different JVM from the server, the transaction type is RESOURCE\_LOCAL. Hence the AccountClient must use EntityTransaction to demarcate the transactions.

## Preparing Deployment Descriptors and Deployment Plans

1. Create persistence.xml file in the META-INF folder as follows. Right click on the META-INF folder and click on New => File.



Provide persistence.xml as the file name and click on Finish.



Copy the below contents to the persistence.xml.

persistence.xml
xml version="1.0" encoding="UTF-8"?
<pre><persistence <="" pre="" xmlns="http://java.sun.com/xml/ns/persistence"></persistence></pre>
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_1_0.xsd"
version="1.0" >
<pre><persistence-unit name="JPA-App-Client"></persistence-unit></pre>
<pre><description>JPA Application Client</description></pre>
<provider>org.apache.openjpa.persistence.PersistenceProviderImpl</provider>
<class>sample.jpa.appclient.Account</class>
<properties></properties>
<property name="openjpa.ConnectionURL" value="jdbc:derby://localhost/AccountDB"></property>
<property name="openjpa.ConnectionDriverName" value="org.apache.derby.jdbc.ClientDriver"></property>
<property name="ConnectionUserName" value="app"></property>
<property name="openjpa.jdbc.SynchronizeMappings" value="false"></property>

2. Similarly, create application-client.xml file in the META-INF folder as given the previous step and copy the below contents into the file.

#### application-client.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<application-client xmlns="http://java.sun.com/xml/ns/javaee"
    xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
    xsi:schemaLocation="http://java.sun.com/xml/ns/javaee http://java.sun.com/xml/ns/javaee/application-client_5.
xsd"
    version="5">
</application-client>
```

#### Since we are using JPA annotations to declare the resources, we do not require any entries in the file.

3. Similarly, create geronimo-application-client.xml file under META-INF file and copy the below contents.

#### geronimo-application-client.xml

```
<?xml version="1.0" encoding="UTF-8"?>
<application-client xmlns="http://geronimo.apache.org/xml/ns/j2ee/application-client-2.0"
 xmlns:sys="http://geronimo.apache.org/xml/ns/deployment-1.2"
 xmlns:naming="http://geronimo.apache.org/xml/ns/naming-1.2"
 xmlns:security="http://geronimo.apache.org/xml/ns/security-2.0"
 xmlns:connector="http://geronimo.apache.org/xml/ns/j2ee/connector-1.2">
 <sys:client-environment>
   <svs:moduleId>
     <sys:groupId>AccountJPA</sys:groupId>
     <sys:artifactId>AccountJPA-app-client</sys:artifactId>
     <sys:version>3.0</sys:version>
     <sys:type>jar</sys:type>
   </sys:moduleId>
    <sys:dependencies>
     <sys:dependencv>
       <sys:groupId>org.apache.geronimo.configs</sys:groupId>
       <sys:artifactId>transaction</sys:artifactId>
       <sys:version>2.1</sys:version>
       <sys:type>car</sys:type>
     </sys:dependency>
   </sys:dependencies>
 </sys:client-environment>
 <sys:server-environment>
   <sys:moduleId>
     <sys:groupId>AccountJPA</sys:groupId>
     <sys:artifactId>AccountJPA-app-client-server</sys:artifactId>
     <sys:version>3.0</sys:version>
     <sys:type>jar</sys:type>
   </sys:moduleId>
 </sys:server-environment>
</application-client>
```

### Deploying the application client

1. Export the Java Project to a jar file Start the geronimo server and open the admin console <a href="http://localhost:8080/console">http://localhost:8080/console</a>. Click on Embedded DB => DB Manager on the Console Navigation portlet. This will open up DB Viewer and Run SQL portlets on the right hand side as follows.

#### **DB Viewer Database List** Databases View Ta AccountDB Application ActiveMRCDB Application Application ArchiveMRCDB PhaniDB Application SystemDatabase Application Application UddiDatabase VehicleDB Application

### Run SQL

create DD.		
Delete DB:	SystemDatabase 💌	Delete
Use DB:	SystemDatabase 💌	Run SQL
	SQL Command/s:	

Enter AccountDB in the Create DB text box and click on Create button. This will create a new Database AccountDB and will be listed in the DB Viewer portlet.

In the textarea SQL Command/s: which is on the Run SQL portlet, enter the below SQL command, and select AccountDB in the Use DB combo box, and click on Run SQL button. This creates the Account1 table in the AccountDB database.

Account1 table
create table ACCOUNT1 (ACCOUNTNO integer, NAME varchar(50), ADDRESS varchar(225), BRANCHCODE integer, BALANCE decimal(15,2));

Insert some sample rows in the table using the below SQL statements by following the same outlined procedure above.

#### Account1 table

```
insert into Account1 values (1111, 'Joe', 'NewYork', 10, 30000.0);
insert into Account1 values (2222, 'John', 'NewJersy', 11, 31000.0);
insert into Account1 values (3333, 'Jane', 'Raleigh', 13, 32000.0);
```

2. Export the Java project by name <code>ApplicationClientJPA.jar</code> as follows. Right click on the java project and export it as jar file.



Select Java => jar

ject Run Window Help

Export	
Select Export resources into a JAR file on the local file system.	N
Select an export destination:	
General     General     General     General     General     General     General     General     General     JJEE     Java     Java     Java     Java     Javadoc     Plug-in Development     General     General     Web     General     Web     General     Web Services	
(?) < Back Next > Fin	ish Cancel

Click on Next and provide the jar file name and destination folder as below and click on Next.

JAR Ex	cport	
JAR File	Specification ich resources should be exported int	to the JAR.
Select the	resources to export:	
±	ApplicationClientJPA	<ul> <li>✓ X .classpath</li> <li>✓ X .project</li> </ul>
Expor	t generated class files and resources t all output folders for checked proje t java source files and resources	cts
Select the	export destination:	Selectrefactorings
JAR file:	C:\temp\Appclient.jar	Browse
Options: Comp Add d Overv	ress the contents of the JAR file rectory entries rrite existing files without warning	
0		< Back Next > Finish Cancel

In the next screen click Next. In the Jar Manifest Specification wizard, select sample.jpa.appclient.AccountClient using Browse button. This is the class that has public static void main(String[] args) method which is the application entry point.

JAR Manif Customize t	est Specification ne manifest file for the JAR file	2.		
Specify the n	nanifest:			
📀 Generat	e <mark>the</mark> manifest file			
Save	e the manifest in the workspac	ce		
Reu	se and save the manifest in th	ie workspace		
Manifes	t file:			Browse
O Use exis	ting manifest from workspace			
Manifes	t file:			Browse
Seal content	5:			
O Seal the	JAR			Details
Seal som	e packages		Nothing seal	ed Details
Select the cla	ass of the application entry po	int:		
Main class:	sample.jpa.appclient.Accoun	ntClient		Browse
0		< Back	Next > Finish	Cancel

Click Finish.

3. Deploy the application client as follows.

Change the directory to <code><geronimo\_home>\bin.</code> Submit the below command

# Running the application client

1. Run the application client as follows.

The application performs the following operations. Use the script client (.bat or .sh) to run the application client.

- List : This option lists the accounts curently in the database. The command to list is as follows.
- Create : This option creates an account in the database. The command to create is as follows.

- Update : This option updates an account with a new balance in the database. The command to update is as follows.
- Delete : This option deletes an account in the database. The command to delete is as follows.