

MODULE 6: Networking

Nội dung thực hành

- Tương tác với trang web sử dụng URL, URLConnection, HttpURLConnection
- Lấy thông tin trạng thái kết nối mạng của thiết bị android
- Tương tác với SOAP Web Services
- Tương tác với REST Web Services

Bài tập 1**Mục đích:**

- Ôn lại cách dùng kỹ thuật asynchronous để làm việc với các tài nguyên trên mạng.
- Sử dụng đối tượng URL để mở một luồng cho việc tương tác.

Yêu cầu:

1. Hiện thị một image từ một url lên một ImageView

Hướng dẫn

```
package vovanhai.wordpress.com;
public class DisplayImageActivity extends Activity {
    private ImageView imgView;
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_display_image);
        imgView=(ImageView) findViewById(R.id.imageView1);
    }
    private boolean isNetworkAvailable() {
        ConnectivityManager cm = (ConnectivityManager)
            getSystemService(Context.CONNECTIVITY_SERVICE);
        NetworkInfo networkInfo = cm.getActiveNetworkInfo();
        // if no network is available networkInfo will be null
        // otherwise check if we are connected
        if (networkInfo != null && networkInfo.isConnected()) {
            return true;
        }
        return false;
    }
    //nút nhấn hiển thị image
    public void displayImage(View v){
        String url="http://10.0.3.2/asp_test/images/my01.jpg";
        if(isNetworkAvailable())
            new ImageLoad(imgView).execute(url);
        else
            Toast.makeText(this, "not connect to internet",
```

```

        Toast.LENGTH_LONG).show();
    }
}
//Tiến trình nạp hình ảnh từ internet về ImageView
class ImageLoad extends AsyncTask<String, Void, Bitmap>{
    private ImageView imgView;
    public ImageLoad(ImageView imgView) {
        this.imgView = imgView;
    }
    protected Bitmap doInBackground(String... params) {
        Bitmap bmp=null;
        try {
            URL url=new URL(params[0]);
            InputStream is = url.openStream();
            bmp=BitmapFactory.decodeStream(is);
        } catch (Exception e) {
            e.printStackTrace();
        }
        return bmp;
    }
    protected void onPostExecute(Bitmap result) {
        imgView.setImageBitmap(result);
    }
}

```

Kết quả hiển thị



Bài tập 2

Mục đích

- Sử dụng HttpURLConnection để tương tác với tài nguyên web.

Yêu cầu

- Tạo một trang web động (Servlet chẳng hạn) cho việc kiểm tra tài khoản đăng nhập có hợp lệ không. Servlet này kết nối với một cơ sở dữ liệu có tên là **logondb** đặt trên SQLServer (Ví dụ có bảng **logon** (*userName*, *password*, *role*) với dữ liệu tự nhập). Khi người dùng cung cấp đúng username-password thì trả về quyền của đối tượng đăng nhập.
- Tạo một Activity cho việc đăng nhập trước khi thực hiện một công việc nào đó. Màn hình cho phép cung cấp username-password và một nút logon. Nếu việc đăng nhập thành công và quyền là admin thì sẽ hiển thị ra một Admin Activity. Nếu không thì báo lỗi để người dùng đăng nhập lại.

Hướng dẫn

- Tạo một servlet và triển khai nó lên một Web Server: tham khảo ở môn Software Architecture, hoặc xem thêm trên blog <http://vovanhai.wordpress.com/>.

```
package vovanhai.wordpress.com;
import java.io.IOException;
import java.io.PrintWriter;
import javax.servlet.ServletException;
import javax.servlet.annotation.WebServlet;
import javax.servlet.http.HttpServlet;
import javax.servlet.http.HttpServletRequest;
import javax.servlet.http.HttpServletResponse;
/**
 * Servlet implementation class Logon
 */
@WebServlet("/Logon")
public class Logon extends HttpServlet {
    private static final long serialVersionUID = 1L;
    public Logon() {
        super();
    }
    protected void doPost(HttpServletRequest request, HttpServletResponse
response) throws ServletException, IOException {
        try {
            String username=request.getParameter("us");
            String password=request.getParameter("ps");

            ConnectDB db=ConnectDB.getInstance();
            PrintWriter out=response.getWriter();
            String role=db.logon(username, password);
            if(role=="")
                out.println("<h1>Invalid User & password</h1>");
            else
                out.println(role);
        } catch (Exception e) {
```

```

        e.printStackTrace();
    }
}

package vovanhai.wordpress.com;
import java.sql.Connection;
import java.sql.DriverManager;
import java.sql.PreparedStatement;
import java.sql.ResultSet;
import com.microsoft.sqlserver.jdbc.SQLServerDriver;

public class ConnectDB {
    private static Connection con;
    private ConnectDB() throws Exception{
        Class.forName(SQLServerDriver.class.getName());
        String url="jdbc:sqlserver://localhost:1433;databaseName=logondb";
        con=DriverManager.getConnection(url, "sa", "12345678");
    }

    public static ConnectDB getInstance()throws Exception{
        ConnectDB condb=null;
        if(condb==null)
            condb=new ConnectDB();
        return condb;
    }

    public String logon(String username,String password)throws Exception{
        String sql="select * from logon where username=? and password=?";
        PreparedStatement ps = con.prepareStatement(sql);
        ps.setString(1, username);
        ps.setString(2,password);
        ResultSet rs=ps.executeQuery();
        if(rs.next())
            return rs.getString("role");
        return "";
    }
}

```

2. Tạo một activity với code cho việc đăng nhập như sau

```

package vovanhai.wordpress.com;
import java.io.BufferedReader;
import java.io.InputStream;
import java.io.InputStreamReader;
import java.io.OutputStreamWriter;
import java.net.HttpURLConnection;
import java.net.URL;
import java.net.URLEncoder;
import java.util.Locale;
import android.content.Context;
import android.os.AsyncTask;

public class PostingData extends AsyncTask<String, Void, String>{
    private Context ctx;
    public PostingData(Context ctx) {
        this.ctx = ctx;
    }
}

```

```

    }
    private InputStream sendRequest(URL url, String method,
        String params)throws Exception{
        HttpURLConnection con=(HttpURLConnection)url.openConnection();
        con.setRequestMethod(method.toUpperCase(Locale.US));
        con.setDoOutput(true);

        OutputStreamWriter ow=new
OutputStreamWriter(con.getOutputStream(),"UTF-8");
        ow.write(params);//send data to server
        ow.flush(); ow.close();

        int rc=con.getResponseCode();//response code from server
        if(rc==HttpURLConnection.HTTP_OK){
            return con.getInputStream();
        }
        return null;
    }
    private String getResponse(InputStream is)throws Exception{
        BufferedReader br=new BufferedReader(new InputStreamReader(is));
        String line="",html="";
        while((line=br.readLine())!=null){
            html+=line+"\n";
        }
        br.close();
        return html;
    }
    protected String doInBackground(String... params) {
        String html="";
        try {
            String us=URLEncoder.encode(params[0],"UTF-8");
            String ps=URLEncoder.encode(params[1],"UTF-8");
            URL url=new URL("http://10.0.3.2:8080/SampleWeb/Logon");
            String thamso="ten="+us+"&matkhau="+ps;
            InputStream is = sendRequest(url, "post", thamso);
            if(is!=null)
                html=getResponse(is);
            else
                html="nothing";
        } catch (Exception e) {
            e.printStackTrace();
        }
        return html;
    }
    @Override
    protected void onPostExecute(String result) {
        if(result.equals("admin")){
            //do your work here
            /*Intent it=new Intent(ctx,AdminActivity.class);
            ctx.startActivity(it);*/
        }
    }
}

```

Bài tập 3

Mục đích

- Tương tác giữa ứng dụng Android và SOAP web service

Yêu cầu

1. Xây dựng một SOAP web service đơn giản kiểm tra một dãy số có phải là một số thẻ credit card hợp lệ hay không
2. Xây dựng một ứng dụng chạy trên android gửi một chuỗi số đến web service để kiểm tra việc hợp lệ này

Hướng dẫn

Cách kiểm tra 1 số có phải là 1 số credit card hợp lệ theo wikipedia (https://en.wikipedia.org/wiki/Luhn_algorithm):

The formula verifies a number against its included check digit, which is usually appended to a partial account number to generate the full account number. This number must pass the following test:

1. From the rightmost digit, which is the check digit, moving left, double the value of every second digit; if the product of this doubling operation is greater than 9 (e.g., $8 \times 2 = 16$), then sum the digits of the products (e.g., 16: $1 + 6 = 7$, 18: $1 + 8 = 9$) or alternatively subtract 9 from the product (e.g., 16: $16 - 9 = 7$, 18: $18 - 9 = 9$).
2. Take the sum of all the digits.
3. If the total modulo 10 is equal to 0 (if the total ends in zero) then the number is valid according to the Luhn formula; else it is not valid.

Assume an example of an account number "7992739871" that will have a check digit added, making it of the form 7992739871x:

Account number	7	9	9	2	7	3	9	8	7	1	x
Double every other	7	18	9	4	7	6	9	16	7	2	x
Sum digits	7	9	9	4	7	6	9	7	7	2	x

The sum of all the digits in the third row is **67+x**.

The check digit (x) is obtained by computing the sum of the non-check digits then computing 9 times that value modulo 10 (in equation form, $(67 \times 9 \bmod 10)$). In algorithm form:

1. Compute the sum of the non-check digits (67).
2. Multiply by 9 (603).

3. The last digit, 3, is the check digit. Thus, $x=3$.

```
package vovanhai.wordpress.com;
public class Luhn {
    /**
     * Validate a number string using Luhn algorithm
     *
     * @param numberString
     * @return
     */
    public static boolean validate(String numberString) {
        return checkSum(numberString) == 0;
    }
    /**
     * Generate check digit for a number string. Assumes check digit or a place
     * holder is already appended at end of the string.
     *
     * @param numberString
     * @return
     */
    public static int checkSum(String numberString) {
        return checkSum(numberString, false);
    }
    /**
     * Generate check digit for a number string.
     * @param numberString
     * @param noCheckDigit
     *      Whether check digit is present or not. True if no check Digit
     *      is appended.
     * @return
     */
    public static int checkSum(String numberString, boolean noCheckDigit) {
        int sum = 0, checkDigit = 0;
        if(!noCheckDigit)
            numberString = numberString.substring(0, numberString.length()-1);
        boolean isDouble = true;
        for (int i = numberString.length() - 1; i >= 0; i--) {
            int k = Integer.parseInt(String.valueOf(numberString.charAt(i)));
            sum += sumToSingleDigit((k * (isDouble ? 2 : 1)));
            isDouble = !isDouble;
        }
        if ((sum % 10) > 0)
            checkDigit = (10 - (sum % 10));
        return checkDigit;
    }
    private static int sumToSingleDigit(int k) {
        if (k < 10)
            return k;
        return sumToSingleDigit(k / 10) + (k % 10);
    }
}
```

Tạo Webservice

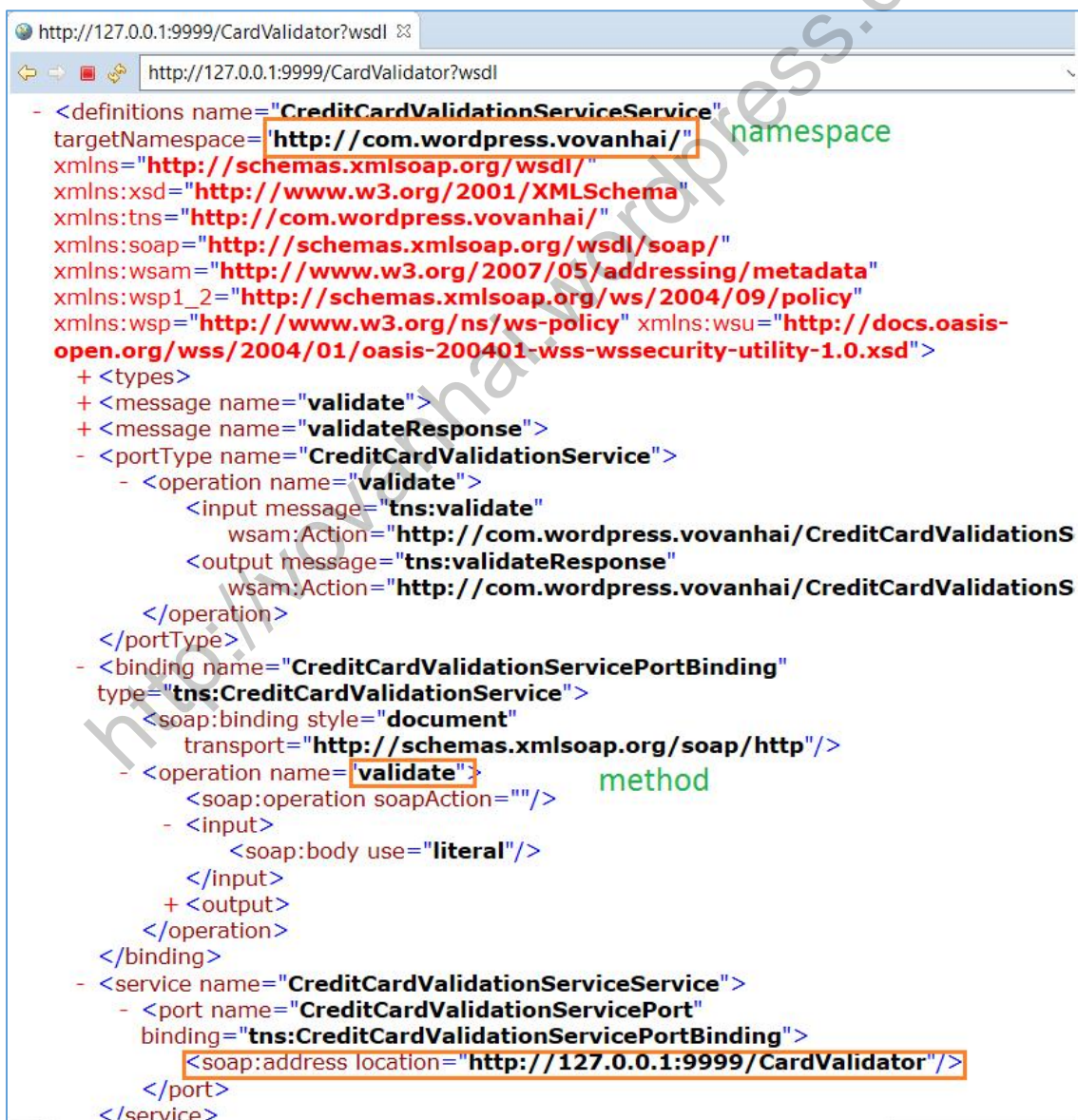
```
package vovanhai.wordpress.com;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
```



```
import javax.xml.ws.Endpoint;
@WebService
public class CreditCardValidationService {
    @WebMethod
    public boolean validate(@WebParam(name="numberString")String numberString){
        return Luhn.validate(numberString);
    }
    public static void main(String[] args) {
        System.out.println("server is running...");
        CreditCardValidationService service= new CreditCardValidationService();
        Endpoint.publish("http://127.0.0.1:9999/CardValidator", service);
    }
}
```

Chạy chương trình, web service sẽ được bind vào đại chỉ: <http://127.0.0.1:9999/CardValidator>

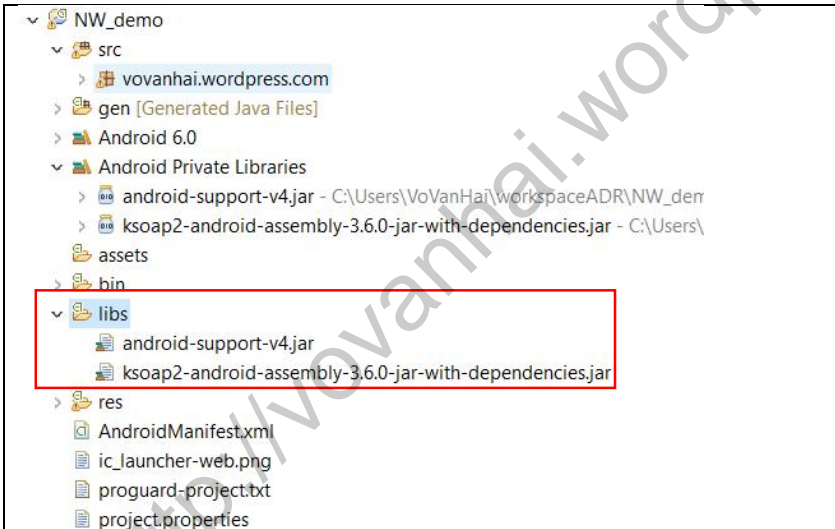
Để xem wsdl file, mở trình duyệt ở địa chỉ: <http://127.0.0.1:9999/CardValidator?wsdl>



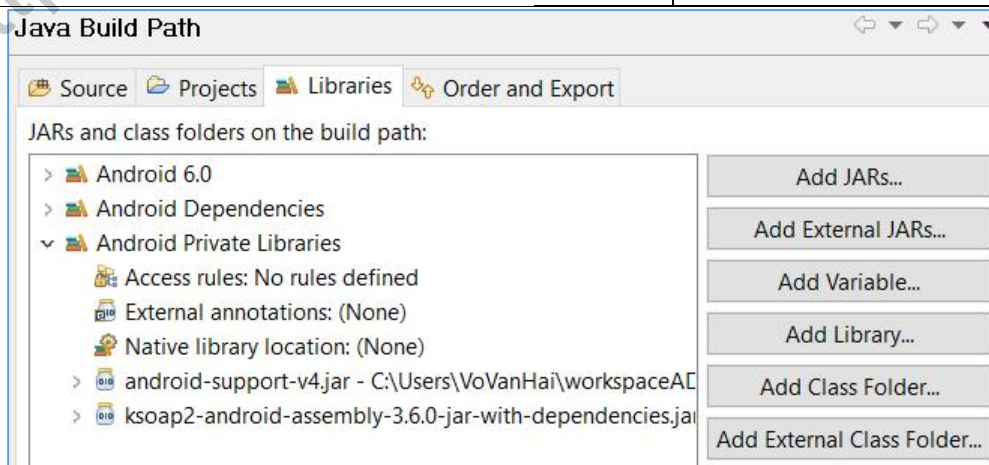
Download thư viện kSOAP tại <https://oss.sonatype.org/content/repositories/ksoap2-android-releases/com/google/code/ksoap2-android/ksoap2-android-assembly/3.6.0/>. Bản mới nhất hiện nay (tháng 4/2016) là 3.6

Name	Last Modified	Size	Description
Parent Directory			
ksoap2-android-assembly-3.6.0-jar-with-dependencies.jar	Thu Oct 15 18:45:09 UTC 2015	163800	
ksoap2-android-assembly-3.6.0-jar-with-dependencies.jar.asc	Thu Oct 15 18:45:12 UTC 2015	473	
ksoap2-android-assembly-3.6.0-jar-with-dependencies.jar.md5	Thu Oct 15 18:45:11 UTC 2015	32	
ksoap2-android-assembly-3.6.0-jar-with-dependencies.jar.sha1	Thu Oct 15 18:45:11 UTC 2015	40	
ksoap2-android-assembly-3.6.0.jar	Thu Oct 15 18:45:05 UTC 2015	2015	
ksoap2-android-assembly-3.6.0.jar.asc	Thu Oct 15 18:45:11 UTC 2015	473	

Trên android project, thêm tham chiếu đến thư viện ksoap bằng cách copy rồi dán vào thư mục libs của projects



Copy rồi Paste ở đây



Lớp dùng làm một tiến trình triệu gọi web service

```
package vovanhai.wordpress.com;
import org.ksoap2.SoapEnvelope;
import org.ksoap2.serialization.PropertyInfo;
import org.ksoap2.serialization.SoapObject;
import org.ksoap2.serialization.SoapSerializationEnvelope;
import org.ksoap2.transport.HttpTransportSE;
import android.os.AsyncTask;
import android.widget.TextView;
public class CallSampleWebService extends AsyncTask<String, Double, SoapObject>{
    private String URL = "http://10.0.3.2:9999/CardValidator?wsdl";
    private String NAMESPACE = "http://com.wordpress.vovanhai/";
    private String METHOD_NAME = "validate";
    private String SOAP_ACTION = NAMESPACE+METHOD_NAME;
    private TextView tvDisplay;
    public CallSampleWebService(TextView tvDisplay) {
        this.tvDisplay = tvDisplay;
    }
    protected SoapObject doInBackground(String... params) {
        SoapObject result=null;
        //Initialize soap request + add parameters
        SoapObject request = new SoapObject(NAMESPACE, METHOD_NAME);
        //Use this to add parameters for 3.6
        PropertyInfo p = new PropertyInfo();
        p.setName("numberString");p.setValue(params[0]);p.setType(String.class);
        request.addProperty(p);
        //Declare the version of the SOAP request
        SoapSerializationEnvelope envelope =
            new SoapSerializationEnvelope(SoapEnvelope.VER11);
        envelope.setOutputSoapObject(request);
        try {
            HttpTransportSE androidHttpTransport = new HttpTransportSE(URL);
            //this is the actual part that will call the webservice
            androidHttpTransport.call(SOAP_ACTION, envelope);
            // Get the SoapResult from the envelope body.
            result = (SoapObject)envelope.bodyIn;
        } catch (Exception e) {
            e.printStackTrace();
        }
        return result;
    }
    protected void onPostExecute(SoapObject result) {
        String kq=result.getPropertyAsString(0);
        if(kq.equalsIgnoreCase("true"))
            tvDisplay.setText("Your credit card is valid");
        else
            tvDisplay.setText("Your credit card is valid");
    }
}

public void callService(View v){
    String []params={"79927398713"};
    new CallSampleWebService(tvDisplay).execute(params);
}
```


Bài tập 4

Mục đích

- Tương tác giữa ứng dụng Android và SOAP web service

Yêu cầu

3. Xây dựng một SOAP web service cơ bản với các hành vi CRUD trên một cơ sở dữ liệu MS SQL Server có tên Customers với một bảng dữ liệu như sau

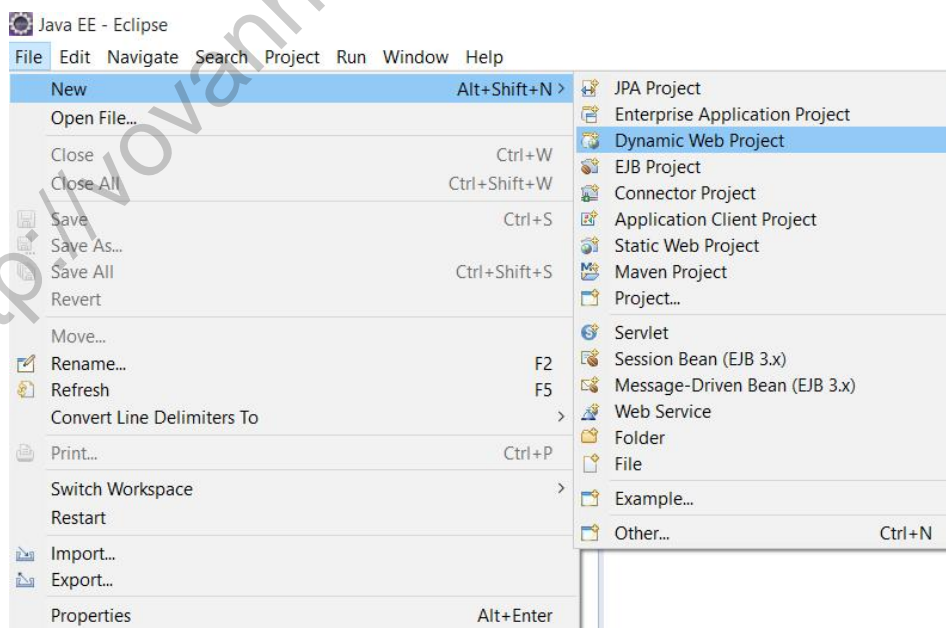
	Column Name	Condensed Type	Allow Nulls
	id	bigint	<input type="checkbox"/>
	address	varchar(255)	<input type="checkbox"/>
	fullName	varchar(150)	<input type="checkbox"/>
	phone	varchar(12)	<input type="checkbox"/>

4. Xây dựng một ứng dụng chạy trên android để tương tác với các chức năng được cung cấp trên web service

Hướng dẫn

Làm một web service: vui lòng xem lại ở môn software architecture (SA) hoặc blog.

Tạo SOAP Web Service - JPA - với eclipse và JBossEAP



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: WS-JPA-JBossEAP

Project location

☒ Use default location

Location: C:\Users\VoVanHai\workspaceMarsLite\WS-JPA-JBossEAP

Target runtime

JBoss EAP 6.4 Runtime

Dynamic web module version

3.0

Configuration

Default Configuration for JBoss EAP 6.4 Runtime

A good starting point for working with JBoss EAP 6.4 Runtime runtime. Additional facets can later be installed to add new functionality to the project.

EAR membership

☐ Add project to an EAR

EAR project name: DHKTPM9A-EJB-EAR

Working sets

☐ Add project to working sets

Working sets:

< Back Next > Finish Cancel

New Dynamic Web Project

Web Module

Configure web module settings.

Context root: WS-JPA_JBossEAP

Content directory: WebContent

☒ Generate web.xml deployment descriptor

< Back Next > Finish Cancel

Nhấn Finish

Tạo package **ws.jpa.entities** chứa các entities. Nhớ thêm các named queries để xử lý các tình huống đọc dữ liệu

```
package ws.jpa.entities;
import java.io.Serializable;
import javax.persistence.Entity;
import javax.persistence.GeneratedValue;
import javax.persistence.GenerationType;
import javax.persistence.Id;
import javax.persistence.NamedQueries;
import javax.persistence.NamedQuery;
@Entity
@NamedQueries({
    @NamedQuery(name="Customer.findAll", query="select c from Customer c"),
    @NamedQuery(name="Customer.findByCity",
        query="select c from Customer c where c.address like :city")
})
public class Customer implements Serializable{
    @Id @GeneratedValue(strategy=GenerationType.IDENTITY)
    private long id;
    private String fullName;
    private String address;
    private String phone;
    public Customer() {
    }
    public Customer(String fullName, String address, String phone) {
        this.fullName = fullName;
        this.address = address;
        this.phone = phone;
    }
    public String getFullName() {
        return fullName;
    }
    public void setFullName(String fullName) {
        this.fullName = fullName;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    public String getPhone() {
        return phone;
    }
    public void setPhone(String phone) {
        this.phone = phone;
    }
    public long getId() {
        return id;
    }
    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + (int) (id ^ (id >>> 32));
    }
}
```

```

        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 (id != other.id)
            return false;
        return true;
    }
    @Override
    public String toString() {
        return "Customer [id=" + id + ", fullName=" + fullName + ", address=" +
address + ", phone=" + phone + "]\n";
    }
}

```

Thêm file persistence.xml cho khai báo cấu hình JPA. Nhớ cấu hình driver và DataSource có tên CustomerDS cho JBossEAP

```

<?xml version="1.0" encoding="UTF-8"?>
<persistence version="2.1" xmlns="http://xmlns.jcp.org/xml/ns/persistence"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://xmlns.jcp.org/xml/ns/persistence
http://xmlns.jcp.org/xml/ns/persistence/persistence_2_1.xsd">
    <persistence-unit name="TeoPU" transaction-type="JTA">
        <provider>org.hibernate.ejb.HibernatePersistence</provider>
        <jta-data-source>CustomerDS</jta-data-source>
        <properties>
            <property name="hibernate.dialect"
                value="org.hibernate.dialect.SQLServerDialect"/>
            <property name="hibernate.hbm2ddl.auto" value="update"/>
            <property name="hibernate.show_sql" value="true"/>
        </properties>
    </persistence-unit>
</persistence>

```

Tyrong file web.xml, thêm khai báo sau để AS hiểu được Persistence Unit

```

<?xml version="1.0" encoding="UTF-8"?>
<web-app xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns="http://java.sun.com/xml/ns/javaee"
xsi:schemaLocation="http://java.sun.com/xml/ns/javaee
http://java.sun.com/xml/ns/javaee/web-app_3_0.xsd"
version="3.0">
    <welcome-file-list>
        <welcome-file>index.html</welcome-file>
    </welcome-file-list>

```



```
<persistence-context-ref>
    <persistence-context-ref-name>persistence/em</persistence-context-ref-name>
    <persistence-unit-name>TeoPU</persistence-unit-name>
</persistence-context-ref>
</web-app>
```

Tạo package **ws.jpa.facades** để chứa các lớp facade cho các jpa entities. Bạn thêm các method cần thiết của bạn

```
package ws.jpa.facades;
import java.util.List;
import javax.ejb.LocalBean;
import javax.ejb.Stateless;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import javax.persistence.PersistenceContextType;
import ws.jpa.entities.Customer;
@Stateless
@LocalBean
public class CustomerController {
    @PersistenceContext(unitName="TeoPU", type=PersistenceContextType.TRANSACTION)
    private EntityManager em;
    public CustomerController() {
    }
    public void Insert(Customer c){
        em.persist(c);
    }
    public void Remove(Customer c){
        em.remove(em.merge(c));
    }
    public void Update(Customer c){
        em.merge(c);
    }
    @SuppressWarnings("unchecked")
    public List<Customer>getAll(){
        return em.createNamedQuery("Customer.findAll").getResultList();
    }
    @SuppressWarnings("unchecked")
    public List<Customer>getByCity(String city){
        return em.createNamedQuery("Customer.findByCity").
            setParameter("city", "%"+city+"%").getResultList();
    }
}
```

Tạo package **ws.services** cho việc tạo các SOAP services

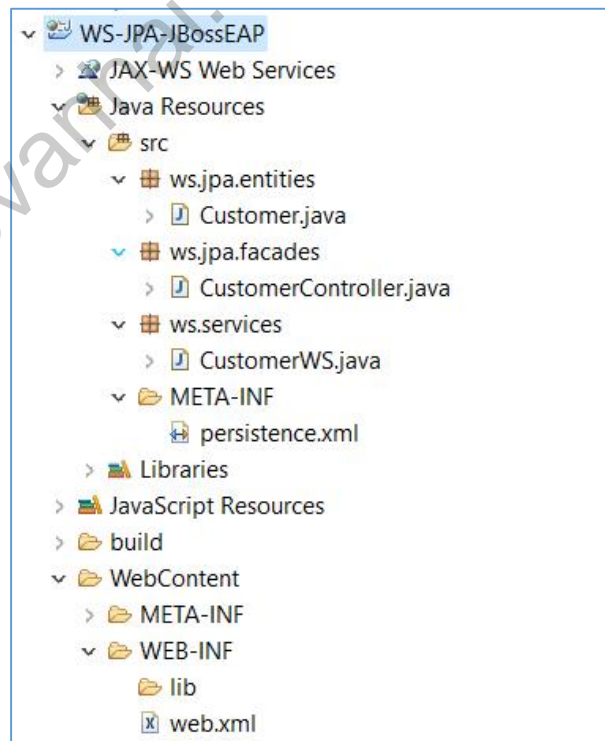
```
package ws.services;
import java.util.List;
import javax.ejb.Stateless;
import javax.jws.WebMethod;
import javax.jws.WebParam;
import javax.jws.WebService;
import javax.persistence.EntityManager;
import javax.persistence.PersistenceContext;
import ws.jpa.entities.Customer;
//missing will generate error about transaction
```

```

@Stateless
@WebService()
public class CustomerWS {
    @PersistenceContext(unitName="TeoPU")
    private EntityManager em;
    public CustomerWS() {
    }
    @WebMethod
    @SuppressWarnings("unchecked")
    public List<Customer>getAll(){
        return em.createNamedQuery("Customer.findAll").getResultList();
    }
    @WebMethod()
    public void Insert(@WebParam(name="cust")Customer c){
        em.persist(c);
    }
    @WebMethod()
    public void update(@WebParam(name="cust")Customer c){
        em.merge(c);
    }
    @WebMethod()
    public void Delete(@WebParam(name="cust")Customer c){
        em.remove(em.merge(c));
    }
    //====Add your wishes methods here
}

```

Structure of project



WSDL file: <http://localhost:8080/WS-JPA-JBossEAP/CustomerWS?wsdl>

http://localhost:8080/WS-JPA-JBossEAP/CustomWS?wsdl

```
<?xml version="1.0" encoding="UTF-8"?>
- <wsdl:definitions targetNamespace="http://services.ws/" name="CustomerWSService" xmlns:ns1="http://schemas.xmlsoap.org/wsdl/soap/" xmlns:tns="http://services.ws/" xmlns:wsdl="http://www.w3.org/2001/XMLSchema">
  - <wsdl:types>
    - <xs:schema targetNamespace="http://services.ws/" xmlns:tns="http://services.ws/" version="1.0" xmlns:xs="http://www.w3.org/2001/XMLSchema">
      <xs:element name="Delete" type="tns:Delete"/>
      <xs:element name="DeleteResponse" type="tns:DeleteResponse"/>
      <xs:element name="Insert" type="tns:Insert"/>
      <xs:element name="InsertResponse" type="tns:InsertResponse"/>
      <xs:element name="getAll" type="tns:getAll"/>
      <xs:element name="getAllResponse" type="tns:getAllResponse"/>
      <xs:element name="update" type="tns:update"/>
      <xs:element name="updateResponse" type="tns:updateResponse"/>
      + <xs:complexType name="Insert">
      + <xs:complexType name="customer">
      + <xs:complexType name="InsertResponse">
      + <xs:complexType name="update">
      + <xs:complexType name="updateResponse">
      + <xs:complexType name="Delete">
      + <xs:complexType name="DeleteResponse">
      + <xs:complexType name="getAll">
      + <xs:complexType name="getAllResponse">
    </xs:schema>
  </wsdl:types>
  - <wsdl:message name="Insert">
```

Code của lớp

Bài tập 5**Mục đích**

- Hiểu và tạo được một RESTful Web Service
- Có thể triệu gọi REST từ ứng dụng android

Yêu cầu

1. Tạo RESTful Web Service quản lý thông tin của một đối tượng Person mô tả cho người.
2. Triệu gọi REST từ ứng dụng android

Hướng dẫn**1. Tạo REST**

Trong ví dụ này, chúng ta dùng JavaSE để tạo REST. Đây là vấn đề khá đơn giản. Tuy nhiên để dễ dàng hơn ta nên dùng Maven build: thêm 2 dependencies vào maven project (file pom.xml). Ở đây ta dùng jersey (<https://jersey.java.net>), version hiện tại (09/04/2015) là 2.22.

```
<dependency>
  <groupId>org.glassfish.jersey.containers</groupId>
  <artifactId>jersey-container-jdk-http</artifactId>
  <version>2.22</version>
</dependency>
<dependency>
  <groupId>org.glassfish.jersey.media</groupId>
  <artifactId>jersey-media-moxy</artifactId>
  <version>2.22</version>
</dependency>
```

Nếu không dùng maven, ta dùng Ant build thì phải add các thư viện như sau

```
> jersey-container-jdk-http-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-container-jdk-http\2.22\jersey-container-jdk-http-2.22.jar
> jersey-common-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-common\2.22\jersey-common-2.22.jar
> javax.annotation-api-1.2.jar - C:\Users\VoVanHai\m2\repository\javax\annotation\api\1.2\javax.annotation-api-1.2.jar
> jersey-guava-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-guava\2.22\jersey-guava-2.22.jar
> hk2-api-2.4.0-b31.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\hk2-api\2.4.0-b31\hk2-api-2.4.0-b31.jar
> hk2-utils-2.4.0-b31.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\hk2-utils\2.4.0-b31\hk2-utils-2.4.0-b31.jar
> aopalliance-repackaged-2.4.0-b31.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\aopalliance-repackaged\2.4.0-b31\aopalliance-repackaged-2.4.0-b31.jar
> javax.inject-2.4.0-b31.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\javax.inject\2.4.0-b31\javax.inject-2.4.0-b31.jar
> hk2-locator-2.4.0-b31.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\hk2-locator\2.4.0-b31\hk2-locator-2.4.0-b31.jar
> javassist-3.18.1-GA.jar - C:\Users\VoVanHai\m2\repository\org\javassist\javassist\3.18.1-GA\javassist-3.18.1-GA.jar
> osgi-resource-locator-1.0.1.jar - C:\Users\VoVanHai\m2\repository\org\osgi\org.osgi.resource\1.0.1\org.osgi.resource-1.0.1.jar
> jersey-server-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-server\2.22\jersey-server-2.22.jar
> jersey-client-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-client\2.22\jersey-client-2.22.jar
> jersey-media-jaxb-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-media-jaxb\2.22\jersey-media-jaxb-2.22.jar
> validation-api-1.1.0.Final.jar - C:\Users\VoVanHai\m2\repository\javax\validation\validation-api\1.1.0.Final\validation-api-1.1.0.Final.jar
> javax.ws.rs-api-2.0.1.jar - C:\Users\VoVanHai\m2\repository\javax\ws\rs\api\2.0.1\javax.ws.rs-api-2.0.1.jar
> jersey-media-moxy-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-media-moxy\2.22\jersey-media-moxy-2.22.jar
> jersey-entity-filtering-2.22.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\jersey-entity-filtering\2.22\jersey-entity-filtering-2.22.jar
> org.eclipse.persistence.moxy-2.6.0.jar - C:\Users\VoVanHai\m2\repository\org\eclipse.persistence\moxy\2.6.0\org.eclipse.persistence.moxy-2.6.0.jar
> org.eclipse.persistence.core-2.6.0.jar - C:\Users\VoVanHai\m2\repository\org\eclipse.persistence\core\2.6.0\org.eclipse.persistence.core-2.6.0.jar
> org.eclipse.persistence.asm-2.6.0.jar - C:\Users\VoVanHai\m2\repository\org\eclipse.persistence\asm\2.6.0\org.eclipse.persistence.asm-2.6.0.jar
> javax.json-1.0.4.jar - C:\Users\VoVanHai\m2\repository\org\glassfish\javax.json\1.0.4\javax.json-1.0.4.jar
```

Lớp biểu diễn cho đối tượng

```
package vovanhai.wordpress.com.dao;
import java.util.List;
import javax.xml.bind.annotation.XmlRootElement;

@XmlRootElement
public class Person {
    private long id;
    private String fullName;
    private String address;
    private List<String> phone;

    public Person() {
    }

    public Person(long id, String fullName, String address) {
        this.id = id;
        this.fullName = fullName;
        this.address = address;
    }

    public Person(long id, String fullName, String address, List<String> phone) {
        this.id = id;
        this.fullName = fullName;
        this.address = address;
        this.phone = phone;
    }
}
```

```

    public List<String> getPhone() {
        return phone;
    }

    public void setPhone(List<String> phone) {
        this.phone = phone;
    }

    public long getId() {
        return id;
    }
    public void setId(long id) {
        this.id = id;
    }
    public String getFullName() {
        return fullName;
    }
    public void setFullName(String fullName) {
        this.fullName = fullName;
    }
    public String getAddress() {
        return address;
    }
    public void setAddress(String address) {
        this.address = address;
    }
    @Override
    public int hashCode() {
        final int prime = 31;
        int result = 1;
        result = prime * result + (int) (id ^ (id >>> 32));
        return result;
    }
    @Override
    public boolean equals(Object obj) {
        if (this == obj)
            return true;
        if (obj == null)
            return false;
        if (getClass() != obj.getClass())
            return false;
        Person other = (Person) obj;
        if (id != other.id)
            return false;
        return true;
    }
    @Override
    public String toString() {
        return fullName;
    }
}

```

Giả sử trong trường hợp này, ta truy xuất dữ liệu. Nhưng để đơn giản, ta có dữ liệu tĩnh được xây dựng như sau:

```

package vovanhai.wordpress.com.dao;
import java.util.ArrayList;

```



```

import java.util.HashMap;
import java.util.List;
import java.util.Map;

public enum PersonDao {
    instance;

    private Map<Long, Person> persons = new HashMap<Long, Person>();
    private PersonDao(){
        List<String> phones=new ArrayList<>();
        phones.add("0903456789");
        phones.add("01686547893");
        phones.add("0986834927");

        persons.put(1L,new Person(1, "teo", "12 nguyen van bao",phones));
        persons.put(2L,new Person(2, "ty", "12/3 Nguyen Kiem"));
    }
    public Map<Long, Person> getPersons() {
        return persons;
    }
}

```

Các dịch vụ CRUD liên quan được code như sau:

```

package vovanhai.wordpress.com.dao;

import java.util.ArrayList;
import java.util.List;
import java.util.Map;

public class PersonService {
    private PersonDao personDao;
    public PersonService() {
        personDao=PersonDao.instance;
    }
    //Create
    public void createPerson(Person p){
        personDao.getPersons().put(p.getId(), p);
    }
    //Get by id
    public Person getPerson(Long id){
        return personDao.getPersons().get(id);
    }
    //Get all
    public Map<Long, Person>getPersons(){
        return personDao.getPersons();
    }
    //get all as list
    public List<Person>getPersonsAsList(){
        List<Person> personList = new ArrayList<Person>();
        personList.addAll(personDao.getPersons().values());
        return personList;
    }
    //delete
    public Person deletePerson(Long id){
        return personDao.getPersons().remove(id);
    }
}

```

```
//update
public void updatePerson(long id, Person p){
    Map<Long, Person> dao = personDao.getPersons();
    if(dao.containsKey(id))
        dao.put(id, p);
}
}
```

Bây giờ ta tạo REST

```
package vovanhai.wordpress.com;

import java.util.List;
import javax.ws.rs.Consumes;
import javax.ws.rs.DELETE;
import javax.ws.rs.GET;
import javax.ws.rs.POST;
import javax.ws.rs.PUT;
import javax.ws.rs.Path;
import javax.ws.rs.PathParam;
import javax.ws.rs.Produces;
import javax.ws.rs.core.GenericEntity;
import javax.ws.rs.core.MediaType;
import javax.ws.rs.core.Response;
import jersey.repackaged.com.google.common.collect.Lists;
import vovanhai.wordpress.com.dao.Person;
import vovanhai.wordpress.com.dao.PersonService;

@Path("/persons")
public class RersonResources {
    private PersonService personService;

    public RersonResources() {
        personService=new PersonService();
    }

    @PUT
    @Consumes({MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML})
    public void addPerson(Person p){
        personService.createPerson(p);
    }

    @GET
    @Path("/{id}")
    @Produces({MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML})
    public Response getPerson(@PathParam(value="id") long id){
        Person p=personService.getPerson(id);
        return Response.ok(p).build();
    }

    @GET
    @Produces({MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML})
    public Response getPersonList(){
        List<Person> lst = personService.getPersonsAsList();
        GenericEntity<List<Person>> entity = new GenericEntity<List<Person>>(<
            Lists.newArrayList(lst)) {});
        return Response.ok(entity).build();
    }
}
```

```

    }

    @DELETE
    @Path("/{id}")
    @Produces({MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML})
    public Response delete(@PathParam(value="id") long id){
        Person p = personService.deletePerson(id);
        if(p!=null)
            return Response.ok(p).build();
        return null;
    }

    @POST
    @Path("/{id}")
    @Consumes({MediaType.APPLICATION_JSON, MediaType.APPLICATION_XML})
    public void update(@PathParam(value="id") Long id, Person newInfos){
        personService.updatePerson(id, newInfos);
    }
}

```

Lớp sau dùng để giả lập một Http Server bên ngoài một container của JavaEE

```

package vovanhai.wordpress.com;

import java.net.URI;
import org.glassfish.jersey.jdkhttp.JdkHttpServerFactory;
import org.glassfish.jersey.server.ResourceConfig;

public class StartRestServer {
    public static void main(String[] args) {
        System.out.println("Http Server is start on http://localhost:8888/");
        URI baseUri=URI.create("http://localhost:8888/");
        ResourceConfig config = new ResourceConfig(RersonResources.class);
        JdkHttpServerFactory.createHttpServer(baseUri, config);
    }
}

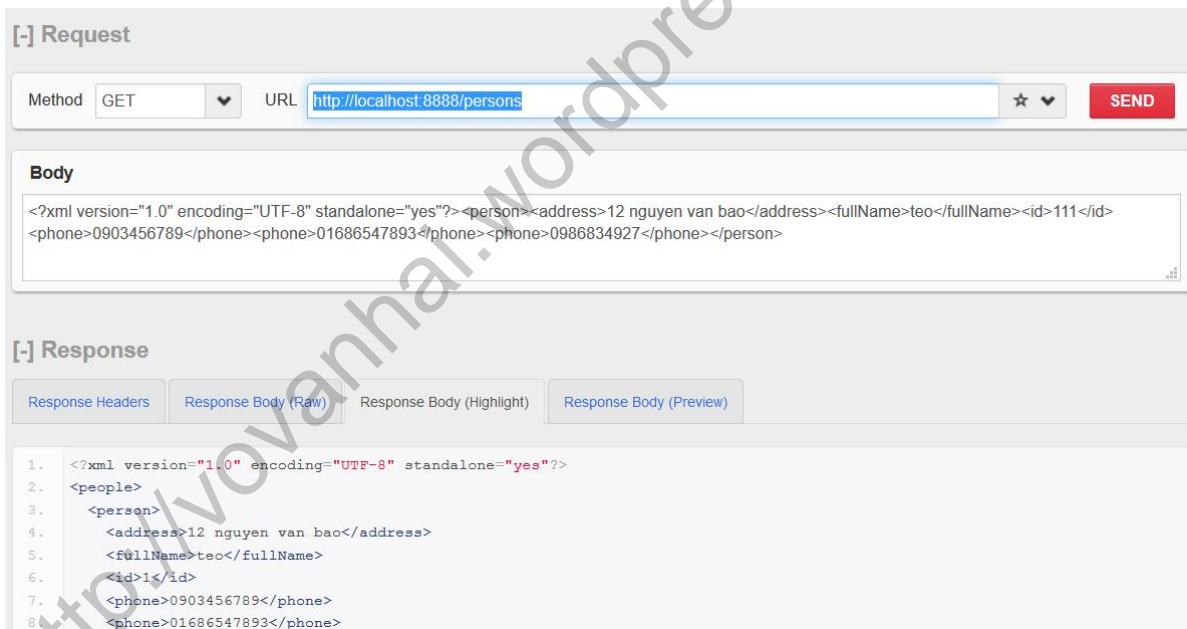
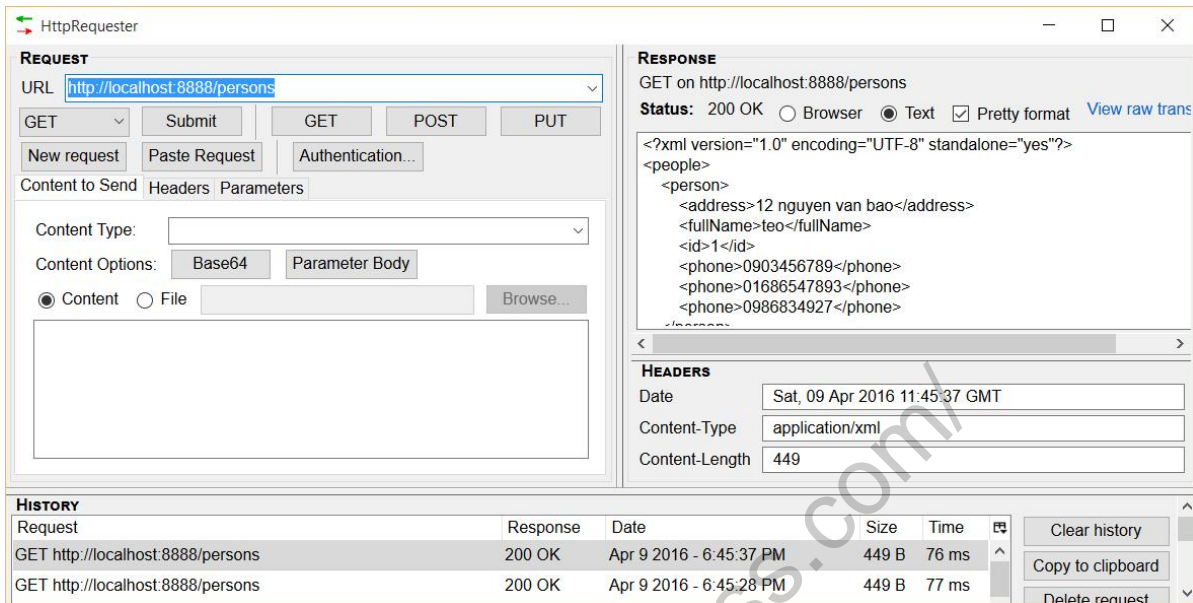
```

Chạy ứng dụng này

2. Thử

Ta có thể dùng các plugin của firefox như:

- RESTclient: <http://www.restclient.net/>
- HttpRequest: <https://github.com/tommut/HttpRequester>.



3. Viết client trên android

Code truy xuất REST có thể dùng HttpURLConnection. Code sau cho việc load một danh sách các person đang có

```
public void getAll(View v) throws Exception{
    String []params={
        "http://10.0.3.2:8888/persons", //url
        "application/json",             //media type
        "GET"                           //method
    };
    String json=new CallRestService().execute(params).get();
    JSONArray array=new JSONArray(json);
    for (int i = 0; i < array.length(); i++) {
        JSONObject obj = array.getJSONObject(i);
        String id = obj.get("id").toString();
    }
}
```


```

        String name = obj.get("fullName").toString();
        String add = obj.get("address").toString();
        List<String> phones=new ArrayList<String>();
        if(!obj.isNull("phone")){
            JSONArray phoneArr = obj.getJSONArray("phone");
            if(phoneArr!=null){
                for (int j = 0; j < phoneArr.length(); j++) {
                    phones.add(phoneArr.getString(i));
                }
            }
        }
        Person p=new Person(Long.parseLong(id), name, add, phones);
        adapter.add(p);
    }
    adapter.notifyDataSetChanged();
}
class CallRestService extends AsyncTask<String, Void, String>{
    @Override
    protected String doInBackground(String... params) {
        String result="";
        try {
            //get all params
            url=new URL(params[0]);
            String mediaType=params[1];
            String method=params[2].toUpperCase(Locale.ENGLISH);

            HttpURLConnection
con=(HttpURLConnection)url.openConnection();
            con.setRequestMethod(method);
            con.setDoInput(true);
            con.setRequestProperty("Content-Type",mediaType);
            con.connect();
            InputStream is = con.getInputStream();
            Scanner in=new Scanner(is);
            while(in.hasNextLine()){
                result+=in.nextLine();
            }
            in.close();con.disconnect();
        } catch (Exception e) {
            e.printStackTrace();
            result="[]";
        }
        return result;
    }
}

```

Thiết kế app như hình sau


PersonRestActivity

1

teo

12 nguyen van bao

[0903456789, 0903456789, 0903456789]

Get all

Add

Update

Remove

teo

ty