1.请谈一下Android系统的架构。   
答：Android系统采用了分层架构，从高层到低层分别是应用程序层、应用程序框架层、系统运行库层和linux核心层。   
  
2.谈谈android大众常用的五种布局。   
答：在Android中，共有五种布局方式，分别是：FrameLayout(框架布局)，LinearLayout (线性布局)，AbsoluteLayout(绝对布局)，RelativeLayout(相对布局)，TableLayout(表格布局)。   
  
（1）FrameLayout 框架布局,放入其中的所有元素都被放置在最左上的区域，而且无法为这些元素指定一个确切的位置,下一个子元素会重叠覆盖上一个子元素，适合浏览单张图片。   
（2）LinearLayout 线性布局,是应用程序中最常用的布局方式，主要提供控件水平或者垂直排列的模型，每个子组件都是以垂直或水平的方式来定位.(默认是垂直)   
（3）AbsoluteLayout 绝对定位布局,采用坐标轴的方式定位组件，左上角是（0，0）点，往右x轴递增，往下Y轴递增,组件定位属性为android:layout\_x 和 android:layout\_y来确定坐标。   
（4）RelativeLayout 相对布局,根据另外一个组件或是顶层父组件来确定下一个组件的位置。和CSS里面的类似。   
（5）TableLayout 表格布局,类似Html里的Table.使用TableRow来布局，其中TableRow代表一行，TableRow的每一个视图组件代表一个单元格。   
  
3.谈谈android数据存储方式。   
答：Android提供了5种方式存储数据：   
（1）使用SharedPreferences存储数据；它是Android提供的用来存储一些简单配置信息的一种机制，采用了XML格式将数据存储到设备中。只能在同一个包内使用，不能在不同的包之间使用。   
（2）文件存储数据；文件存储方式是一种较常用的方法，在Android中读取/写入文件的方法，与Java中实现I/O的程序是完全一样的，提供了openFileInput()和openFileOutput()方法来读取设备上的文件。   
（3）SQLite数据库存储数据；SQLite是Android所带的一个标准的数据库，它支持SQL语句，它是一个轻量级的嵌入式数据库。   
（4）使用ContentProvider存储数据；主要用于应用程序之间进行数据交换，从而能够让其他的应用保存或读取此Content Provider的各种数据类型。   
（5）网络存储数据；通过网络上提供给我们的存储空间来上传(存储)和下载(获取)我们存储在网络空间中的数据信息。   
  
4.Android中Activity, Intent, Content Provider, Service各有什么区别。   
答：Activity： 活动，是最基本的android应用程序组件。一个活动就是一个单独的屏幕，每一个活动都被实现为一个独立的类，并且从活动基类继承而来。   
Intent： 意图，描述应用想干什么。最重要的部分是动作和动作对应的数据。   
Content Provider：内容提供器，android应用程序能够将它们的数据保存到文件、SQLite数据库中，甚至是任何有效的设备中。当你想将你的应用数据和其他应用共享时，内容提供器就可以发挥作用了。   
Service：服务，具有一段较长生命周期且没有用户界面的程序。   
  
5.View, surfaceView, GLSurfaceView有什么区别。   
答：view是最基础的，必须在UI主线程内更新画面，速度较慢。   
SurfaceView 是view的子类，类似使用双缓机制，在新的线程中更新画面所以刷新界面速度比view快   
GLSurfaceView 是SurfaceView的子类，opengl 专用的   
  
6.Adapter有什么作用？常见的Adapter有哪些？   
答：Adapter是连接后端数据和前端显示的适配器接口。常见的Adapter有ArrayAdapter, BaseAdapter, CursorAdapter, HeaderViewListAdapter, ListAdapter, ResourceCursorAdapter, SimpleAdapter, SimpleCursorAdapter, SpinnerAdapter, WrapperListAdapter等   
  
7.Manifest.xml文件中主要包括哪些信息？   
答：manifest：根节点，描述了package中所有的内容。   
uses-permission：请求你的package正常运作所需赋予的安全许可。   
permission： 声明了安全许可来限制哪些程序能你package中的组件和功能。   
instrumentation：声明了用来测试此package或其他package指令组件的代码。   
application：包含package中application级别组件声明的根节点。   
activity：Activity是用来与用户交互的主要工具。   
receiver：IntentReceiver能使的application获得数据的改变或者发生的操作，即使它当前不在运行。   
service：Service是能在后台运行任意时间的组件。   
provider：ContentProvider是用来管理持久化数据并发布给其他应用程序使用的组件。   
  
8.xml解析实例：http://www.eoeandroid.com/thread-78025-1-1.html   
8.请写一段代码（SAX, DOM, 或者pull ）来解析XML文档。

答：下面是要解析的XML文件：

<?xml version="1.0" encoding="UTF-8"?>

<persons>

<person id="1">

<name>张三</name>

<age>22</age>

</person>

<person id="2">

<name>李四</name>

<age>23</age>

</person>

</persons>

定义一个名为Person的javaBean用于存放上面解析出来的xml内容

public class Person {

private Integer id;

private String name;

private Short age;

public Integer getId() {

return id;

}

public void setId(Integer id) {

this.id = id;

}

public String getName() {

return name;

}

public void setName(String name) {

this.name = name;

}

public Short getAge() {

return age;

}

public void setAge(Short age) {

this.age = age;

}

}

（1）使用SAX读取XML文件；它采用的是事件驱动，并不需要解析完整个文档，速度快并且占用内存少。需要为SAX提供实现ContentHandler接口的类。

PersonDefaultHandler.java

import java.util.ArrayList;

import java.util.List;

import org.xml.sax.Attributes;

import org.xml.sax.SAXException;

import org.xml.sax.helpers.DefaultHandler;

import com.sinber.domain.Person;

public class PersonDefaultHandler extends DefaultHandler {

private List<Person> persons;

private Person person ; //记录当前person

private String perTag; //记录前一个标签的名称

/\*\*

\* 重写父类的开始文档方法。用于初始化

\*/

@Override

public void startDocument() throws SAXException {

persons = new ArrayList<Person>();

}

@Override

public void startElement(String uri, String localName, String qName,

Attributes attributes) throws SAXException {

if("person".equals(localName)){

Integer id = new Integer(attributes.getValue(0)); //取id

person = new Person();

person.setId(id);

}

perTag = localName;

}

/\*\*参数：

\* ch 整个XML字符串

\* start 节点值在整个XML字符串中的索引位置

\* length 节点值的长度

\*/

@Override

public void characters(char[] ch, int start, int length)

throws SAXException {

if(perTag!=null){

String data = new String(ch,start,length);

if("name".equals(perTag)){

person.setName(data);

}else if("age".equals(perTag)){

person.setAge(new Short(data));

}

}

}

@Override

public void endElement(String uri, String localName, String qName)

throws SAXException {

if("person".equals(localName)){

persons.add(person);

person = null;

}

perTag = null;

}

public List<Person> getPersons() {

return persons;

}

}

SAXPerson.java

import java.io.InputStream;

import java.util.List;

import javax.xml.parsers.SAXParser;

import javax.xml.parsers.SAXParserFactory;

import com.sinber.domain.Person;

public class SAXPerson{

public static List<Person> getPerson() throws Exception{

//通过类装载器获取文件

InputStream inStream = SAXPersonService.class.getClassLoader().getResourceAsStream("person.xml");

SAXParserFactory factory = SAXParserFactory.newInstance();

SAXParser saxParser = factory.newSAXParser();

PersonDefaultHandler handler = new PersonDefaultHandler();

saxParser.parse(inStream, handler);

inStream.close();

return handler.getPersons();

}

}

（2）DOM解析XML文件时，会将XML文件的所有内容读取到内存中，然后允许您使用DOM API遍历XML树、检索所需的数据。

DOMPerson.java

import java.io.InputStream;

import java.util.ArrayList;

import java.util.List;

import javax.xml.parsers.DocumentBuilder;

import javax.xml.parsers.DocumentBuilderFactory;

import org.w3c.dom.Document;

import org.w3c.dom.Element;

import org.w3c.dom.Node;

import org.w3c.dom.NodeList;

import com.sinber.domain.Person;

public class DOMPerson {

public static List<Person> getPerson() throws Exception{

List<Person> pers = new ArrayList<Person>();

InputStream inStream = SAXPersonService.class.getClassLoader().getResourceAsStream("person.xml");

DocumentBuilderFactory factory =DocumentBuilderFactory.newInstance();

DocumentBuilder builder = factory.newDocumentBuilder();

Document dom = builder.parse(inStream);

Element root = dom.getDocumentElement();

NodeList persons = root.getElementsByTagName("person");

for(int i=0;i<persons.getLength();i++){

Element personNode =(Element)persons.item(i);

Person person = new Person();

person.setId(new Integer(personNode.getAttribute("id")));

NodeList childNodes = personNode.getChildNodes();

for(int j=0;j<childNodes.getLength();j++){

Node childNode = childNodes.item(j);

if(childNode.getNodeType()==Node.ELEMENT\_NODE){

Element element = (Element)childNode;

if("name".equals(childNode.getNodeName())){

person.setName(new String(element.getFirstChild().getNodeValue()));

}else if("age".equals(childNode.getNodeName())){

person.setAge(new Short(element.getFirstChild().getNodeValue()));

}

}

}

pers.add(person);

}

inStream.close();

return pers;

}

}

（3）使用Pull解析器读取XML文件

PullPerson.java

import java.io.File;

import java.io.FileOutputStream;

import java.io.InputStream;

import java.util.ArrayList;

import java.util.List;

import org.xmlpull.v1.XmlPullParser;

import org.xmlpull.v1.XmlSerializer;

import android.os.Environment;

import android.util.Xml;

import com.sinber.domain.Person;

public class PullPerson {

public static void save(List<Person> persons) throws Exception{

XmlSerializer serializer = Xml.newSerializer();

File file = new File(Environment.getExternalStorageDirectory(),"person.xml");

FileOutputStream outStream = new FileOutputStream(file);

serializer.setOutput(outStream,"UTF-8");

serializer.startDocument("UTF-8", true);

serializer.startTag("", "persons");

for(Person person:persons){

serializer.startTag("", "person"); //person

serializer.attribute("", "id", ""+person.getId());

serializer.startTag("", "name"); //name

serializer.text(person.getName());

serializer.endTag("", "name"); //name

serializer.startTag("", "age"); //age

serializer.text(person.getAge().toString());

serializer.endTag("", "age");//age

serializer.endTag("", "person"); //person

}

serializer.endTag("", "persons");

serializer.endDocument();

outStream.close();

}

public static List<Person> getPersons() throws Exception{

List<Person> persons = null;

Person person = null;

XmlPullParser parser= Xml.newPullParser();

InputStream inStream = PullPersonService.class.getClassLoader().getResourceAsStream("person.xml");

parser.setInput(inStream, "UTF-8");

int eventType = parser.getEventType(); //触发第一个事件

while(eventType!=XmlPullParser.END\_DOCUMENT){

switch(eventType){

case XmlPullParser.START\_DOCUMENT:

persons = new ArrayList<Person>();

break;

case XmlPullParser.START\_TAG: //开始元素事件

if("person".equals(parser.getName())){

person = new Person();

person.setId(new Integer(parser.getAttributeValue(0)));

}else if(person!=null){

if("name".equals(parser.getName())){

person.setName(parser.nextText());

}else if("age".equals(parser.getName())){

person.setAge(new Short(parser.nextText()));

}

}

break;

case XmlPullParser.END\_TAG: //结束元素事件

if("person".equals(parser.getName())){

persons.add(person);

person = null;

}

break;

default:

break;

}

eventType = parser.next();

}

return persons;

}

}

以上三种方式任选其一即可。

9.根据自己的理解描述下Android数字签名。   
答：(1)所有的应用程序都必须有数字证书，Android系统不会安装一个没有数字证书的应用程序   
(2)Android程序包使用的数字证书可以是自签名的，不需要一个权威的数字证书机构签名认证   
(3)如果要正式发布一个Android ，必须使用一个合适的私钥生成的数字证书来给程序签名，而不能使用adt插件或者ant工具生成的调试证书来发布。   
(4)数字证书都是有有效期的，Android只是在应用程序安装的时候才会检查证书的有效期。如果程序已经安装在系统中，即使证书过期也不会影响程序的正常功能

10.已知单链表的头结构head,写一个函数把这个链表逆序。

答： 如下所示

Node.java

public class Node {

private Integer count;

private Node nextNode;

public Node(){

}

public Node(int count){

this.count = new Integer(count);

}

public Integer getCount() {

return count;

}

public void setCount(Integer count) {

this.count = count;

}

public Node getNextNode() {

return nextNode;

}

public void setNextNode(Node nextNode) {

this.nextNode = nextNode;

}

}

ReverseSingleLink.java

public class ReverseSingleLink {

public static Node revSingleLink(Node head){

if(head == null){ //链表为空不能逆序

return head;

}

if(head.getNextNode()==null){ //如果只有一个结点，当然逆过来也是同一个

return head;

}

Node rhead = revSingleLink(head.getNextNode());

head.getNextNode().setNextNode(head);

head.setNextNode(null);

return rhead;

}

public static void main(String[] args){

Node head = new Node(0);

Node temp1 = null,temp2 = null;

for(int i=1;i<100;i++){

temp1 = new Node(i);

if(i==1){

head.setNextNode(temp1);

}else{

temp2.setNextNode(temp1);

}

temp2 = temp1;

}

head = revSingleLink(head);

while(head!=null){

head = head.getNextNode();

}

}

}