第九章

看看精彩的世界--使用网络技术

主讲：王海

本章目标

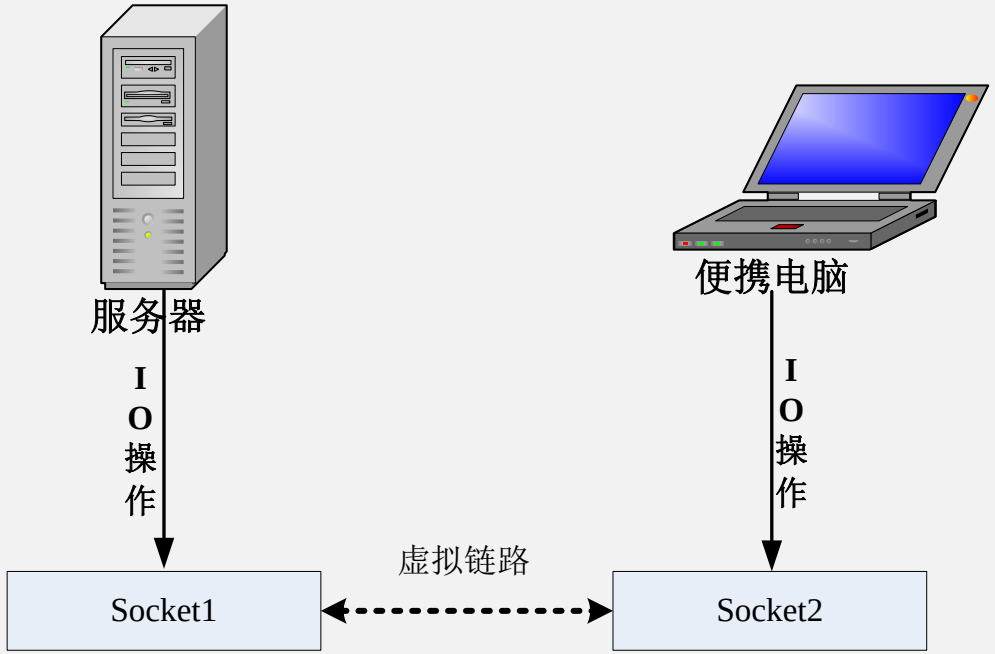
- 了解网络编程原理
- 了解基于TCP协议的网络通信机制
- 能够熟练使用HttpURLConnection进行网络通信
- 能够使用WebView组件浏览网页

网络编程

- Android完全支持JDK本身所提供的TCP、UDP网络通信的API，也支持URL、URLConnection等网络通信API
- Android中常用的网络编程方式：
  - ✓ 针对TCP/IP协议的Socket和ServerSocket
  - ✓ 针对HTTP协议的网络编程，如HttpURLConnection和HttpClient
  - ✓ 直接使用WebKit访问网络

TCP协议

- TCP/IP协议规范了网络上的所有通信设备之间的数据往来格式以及传送方式
- TCP/IP协议提供一种数据打包和寻址的标准方法，可以在Internet中无差错地传送数据
- TCP/IP通信协议是一种可靠的、双向的、持续的、点对点的网络协议



网络通信的两种Socket

- java.net包中包含了网络编程所需的类型，其中基于TCP协议的网络编程主要使用以下两种Socket：
  - ✓ ServerSocket是服务器套接字，用于监听并接收来自客户端的Socket连接
  - ✓ Socket是客户端套接字，用于实现两台计算机之间的通信

Socket

- Socket的构造方法两种：
  - ✓ Socket(String host,int port)
  - ✓ Socket (String host,int port,InetAddress localAddr,int localPort)

– 【示例】创建Socket对象

```
try{
   Socket s= new Socket("192.168.1.128" , 28888);
   ...//Socket通信
}catch (IOException e) {
   e.printStackTrace();
}
```

Socket类常用方法

方法	功能描述
public InetAddress getInetAddress()	返回连接到远程主机的地址，如果连接失败则返回以前连接的主机
public int getPort()	返回Socket连接到远程主机的端口号
public int getLocalPort()	返回本地连接终端的端口号
public InputStream getInputStream()	返回一个输入流，从Socket读取数据
public OutputStream getOutputStream()	返回一个输出流，往Socket中写数据
public synchronized void close()	关闭当前Socket连接

ServerSocket

- ServerSocket是服务器套接字，运行在服务器端，在指定的端口上主动监听来自客户端的Socket连接
- ServerSocket类的构造方法：
  - ✓ ServerSocket(int port)
  - ✓ ServerSocket(int port,int backlog)
  - ✓ ServerSocket(int port,int backlog,InetAddress localAddr)

– 【示例】创建ServerSocket对象

```
try {
   ServerSocket server = new ServerSocket(28888);
} catch (IOException e) {
   e.printStackTrace();
}
```

ServerSocket类常用的方法

方法	功能说明
public Socket accept()	接收客户端Socket连接请求，并返回一个与客户端Socket对应的Socket实例，该方法是一个阻塞方法，如果没有接收到客户端发送的Socket，则一直处于等待状态，线程也会被阻塞
public InetAddress getInetAddress()	返回当前ServerSocket实例的地址信息
public int getLocalPort()	返回当前ServerSocket实例的服务端口
public void close()	关闭当前ServerSocket实例

用ServerSocket进行网络通信的步骤

① 根据指定端口实例化一个ServerSocket对象
② 调用ServerSocket对象的accept()方法接收客户端发送的Socket对象
③ 调用Socket对象的getInputStream()/getOutputStream()方法建立与客户端进行交互的IO流
④ 服务器与客户端根据一定的协议进行交互，直到关闭连接
⑤ 关闭服务器端的Socket
⑥ 回到第2步，继续监听下一次客户端发送的Socket请求连接

授权应用程序能够访问网络

- 要让客户端能够访问服务器，必须在AndroidManifest.xml配置文件中增加如下权限

```
<uses-permission
android:name="android.permission.INTERNET">
</uses-permission>
```

URL

- URL（Uniform Resource Locator，统一资源定位器）用于表示互联网上资源的唯一地址

方法	功能描述
public URL(String spec)	构造方法，根据指定的字符串创建一个URL对象
public URL(String protocol,String host,int port,String file)	构造方法，根据指定的协议、主机名、端口号和文件资源创建一个URL对象
public URL(String protocol, String host, String file)	构造方法，根据指定的协议、主机名、和文件资源创建URL对象
public String getProtocol()	返回协议名
public String getHost()	返回主机名
public int getPort()	返回端口号，如果没有设置端口，则返回-1
public String getFile()	返回文件名
public String getRef()	返回URL的锚
public String getQuery()	返回URL的查询信息
public String getPath()	返回URL的路径
public URLConnection openConnection()	返回一个URLConnection对象
public final InputStream openStream()	返回一个用于读取该URL资源的InputStream流

URLConnection

方法	功能描述
public int getContentLength()	获得文件的长度
public String getContentType()	获得文件的类型
public long getDate()	获得文件创建的时间
public long getLastModified()	获得文件最后修改的时间
public InputStream getInputStream()	获得输入流，以便读取文件的数据
public OutputStream getOutputStream()	获得输出流，以便输出数据
public void setRequestProperty(String key,String value)	设置请求属性值

HttpURLConnection

- Java提供了java.net.HttpURLConnection类专门用于处理HTTP的请求和响应
- HttpURLConnection继承自URLConnection类

方法	功能描述
InputStream getInputStream()	返回从此处打开的连接读取的输入流
OutputStream getOutputStream()	返回写入到此连接的输出流
String getRequestMethod()	获取请求方法
int getResponseCode()	获取状态码，如HTTP_OK、HTTP_UNAUTHORIZED
void setRequestMethod(String method)	设置URL请求的方法
void setDoInput(boolean doinput)	设置输入流，如果使用URL连接进行输入，则将DoInput标志设置为true（默认值）；如果不打算使用，则设置为false
void setUseCaches(boolean usecaches)	设置连接是否使用任何可用的缓存
void disconnect()	关闭连接

HttpURLConnection

- HttpURLConnection是一个抽象类，无法直接实例化
- 使用URL的openConnection()方法获得HttpURLConnection实例
- 【示例】获取HttpURLConnection对象

```
//创建URL
URL url=new URL("http://www.google.com/");
//获取HttpURLConnection连接
HttpURLConnection
urlConn=(HttpURLConnection)url.openConnection();
```

- 【示例】设置HttpURLConnection属性

```
//设置输出、输入流
urlConn.setDoOutput(true);
urlConn.setDoInput(true);
//设置方式为POST
urlConn.setRequestMethod("POST");
//请求不能使用缓存
urlConn.setUseCaches(false);
```

- 【示例】关闭HttpURLConnection连接

```
urlConn.disconnect();
```

WebView组件

- WebView专门用来浏览网页的视图组件
- WebView优点：
  - ✓ 功能强大，支持CSS、JavaScript和HTML，并很好地融入布局，使页面更加美观
  - ✓ 能够对浏览器控件进行详细的设置，例如字体、背景颜色、滚动条样式
  - ✓ 能够捕捉到所有浏览器的操作，例如单击、打开或关闭URL

方法	功能描述
loadUrl(String url)	打开一个指定的Web资源页面
loadData(String data, StringmimeType,String encoding)	显示HTML格式的网页内容
getSettings()	获取WebView的设置对象
addJavascriptInterface()	将一个对象添加到JavaScript的全局对象Window中，这样可以通过Window.*XXX*进行调用，与JavaScript进行交互
clearCache()	清除缓存
destory()	销毁WebView

WebView组件的基本步骤

- 在AndroidManifest.xml中配置访问权限
- 在布局文件中创建WebView元素
- 在代码中加载网页

本章总结

- Java中的网络编程经验完全适用于Android应用的网络编程
- Android完全支持JDK本身的TCP、UDP网络通信的API，也支持URL、URLConnection等网络通信API
- 通讯协议是用来管理数据通信的一组规则，用于规范传输速率、传输代码、代码结构、传输控制步骤、出错控制等
- 通信协议规定了通信的内容、方式和通信时间，其核心要素由语义、语法和时序三部分组成
- TCP/IP（Transmission Control Protocol/Internet Protocol，传输控制协议/互联网络协议）是最基本的通信协议，也是网络中最常用的协议
- TCP/IP通信协议是一种可靠的、双向的、持续的、点对点的网络协议

本章总结

- ServerSocket是服务器套接字，用于监听并接收来自客户端的Socket连接
- Socket是客户端套接字，用于实现两台计算机之间的通信
- URL（Uniform Resource Locator，统一资源定位器）表示互联网上某一资源的地址
- URL的openConnection()方法返回一个URLConnection对象
- HttpURLConnection继承URLConnection，每个HttpURLConnection实例都可用于生成单个请求，可以透明地共享连接到HTTP服务器的基础网络
- WebView是专门用来浏览网页的视图组件，为用户提供了一系列的网页浏览、用户交互接口，通过这些接口显示和处理请求的网络资源