008.使用代理服务器

代理服务器：

1）英文就叫Proxy，即代替用户去连接想要的网站并获取信息；

2）一般主流的商业浏览器都提供代理的功能，即你可以先设置一个代理服务器，

那么接下来所有的上网都是通过该指定的代理服务器完成；

3）一旦设置好了代理服务器，那么你之后所有的上网行为都是通过代理服务器完成的：

i. 你输入一个网址并回车后，会先连接代理服务器；

ii. 然后代理服务器帮你打开那个网址并获取你想要的信息，然后将信息返回给你；

4）为什么需要代理服务器：

i. 处于特殊安全原因需要对外隐藏自己的IP，代理服务器对外只能暴露代理服务器本身的IP，但是可以隐藏你的IP；

ii. 访问国内特定单位、内部团体的资料（也是i.的一种特殊情况）；

iii. 代理服务器可以提供缓存，大大提高访问速度

即你访问一个资源的时候，如果之前有其他人通过Proxy访问过了，

那么Porxy就会把该资源直接放入自己的缓存中，下次再有用户想要获取就直接从缓存中拿而不是去连接这个远程资源了。

e.g. 迅雷下载里面的离线下载/取回本地，之所以速度那么快是因为迅雷服务器已经将该资源保存在服务器的缓存中了，

你下载的时候并不是直接连接远程资源，而是直接从迅雷服务器取回本地，而迅雷可以提供最大带宽接入的服务；

Proxy

1）Java中就用Proxy类来代表代理服务器；

2）在URL的openConnection就有一个版本，其参数就是Proxy对象，包括Socket等的构造器也都有传入Proxy对象的重载版本，一旦调用了以上方法指定了代理服务器，那么接下来的所有对外访问行为都是通过指定的代理服务器进行的；

3）Proxy构造器：Proxy(Proxy.Type type, SocketAddress sa)；

i. Proxy.Type是Proxy类中定义的枚举类型，共有三个值：DIRECT（直连，不使用代理）、

HTTP（高级协议代理，HTTP、FTP等的代理）、SOCKS（SOCKS V4、V5的代理）；

ii. 之前的openConnection以及不带Proxy参数的Socket构造器等都是默认DIRECT直连的，即不使用任何代理的直连；

iii. sa即代理服务器的IP地址/端口号；

4）openConnection时指定代理：URLConnection URL.openConnection(Proxy proxy)；

// 用指定的代理服务器代开连接

5) 指定socket使用代理服务器：Socket(Proxy proxy);

6) 示例：openConnection使用代理

```
public class ProxyDemo {
    // 代理服务器的IP和端口
    private final String PROXY_ADDR = "27.191.234.69";
    private final int PROXY_PORT = 9999;

    public void init() throws IOException {
        URL url = new URL("http://www.baidu.com"); // 想访问的网址
        Proxy proxy = new Proxy(Proxy.Type.HTTP, new InetSocketAddress(PROXY_ADDR,
PROXY_PORT));
        URLConnection conn = url.openConnection(proxy); // 连接时设置代理
        conn.setConnectTimeout(3000);
     Scanner scan = new Scanner(conn.getInputStream());
        while (scan.hasNextLine()) { // 直接将返回的网页代码下载到本地的index.htm文件中
            String line = scan.nextLine();
            System.out.println(line);
        }
    }

    public static void main(String[] args) throws IOException {
        new ProxyDemo().init();
    }
}
```

ProxySelector——自动代理选择器：

1) 直接显示传入Porxy对象的方法未免有点太繁琐

主要问题： 有时候需要根据不同的网址选择不同的代理服务器，

比如访问欧洲的网页时需要用一种代理服务器加速，

而访问美洲的网页时可能用另一个代理更快更好，那么每次都要这样显式地指定代理岂不是很繁琐吗？

2) Java提供了一个抽象类ProxySelector，该类的对象可以根据你要连接的URL自动选择最合适的代理，

但是该类是抽象类，需要自己继承该类**实现个性化代理自动选择；**

3) 选择器实现完毕后就可以创建选择器的对象实例，

并调用ProxySelector类的静态方法**setDefault**将该选择器设置为默认选择器；

那么之后，所有的访问网络的行为（openConnection、Socket构造器）等都不需要指定代理，

直接自动交给代理选择器自动根据你要访问的URL选择一个代理帮你访问网络；

4) **ProxySelector：其中有两个重要的方法需要实现**

i. List<Proxy> select(URI uri); 给定一个URI（URL的父类，但实际中只用URL）

返回一个最适合访问该URI的代理服务器列表，

其中会首选列表的第一个代理，如果不行则用第二个，如果全部不行就会调用connectFailed方法处理连接失败问题

ii. void connectFailed(URI uri, SocketAddress sa, IOException ioe);

a. URI是目标网址；

b. sa是连接失败的那个Proxy的IP地址；

c. ioe是连接失败时所抛出的异常对象；

```
public class ProxyDemo {
    private final String PROXY_ADDR = "27.191.234.69";
    private final int PROXY_PORT = 9999;

    public void init() throws IOException {
        ProxySelector.setDefault(new ProxySelector() {
            @Override
            public List<Proxy> select(URI uri) {
                // 默认任何URI都返回一个代理
                List<Proxy> list = new ArrayList<>();
                list.add(new Proxy(Proxy.Type.HTTP, new InetSocketAddress(PROXY_ADDR,
PROXY_PORT)));
                return list;
            }

            @Override
            public void connectFailed(URI uri, SocketAddress sa, IOException ioe) {
                // 简单地打印信息
                System.out.println("无法连接到代理！");
            }
        });
        URL url = new URL("http://www.baidu.com"); // 想访问的网址
        URLConnection conn = url.openConnection(); // 连接时设置代理
        Scanner scan = new Scanner(conn.getInputStream());
        while (scan.hasNextLine()) { // 直接将返回的网页代码下载到本地的index.htm文件中
            String line = scan.nextLine();
            System.out.println(line);
        }

    }

    public static void main(String[] args) throws IOException {
        new ProxyDemo().init();
    }
}
```