> 文本总结URI、URL、URN、Uri相关知识点

Android中的URI、URL、URN、Uri

版本：2018/9/10-1(18:18)

问题汇总

1. URI是什么?
2. URL是什么?
3. URN是什么?
4. URI、URL、URN三者的关系
5. URI和URL实例的区别?
6. 绝对URI和相对URI
7. 不透明URI和分层URI
8. 相对URL和绝对URL?
9. Java中的URL类不提供对特殊字符的转义
10. URI提供了转义功能
11. URI和URL如何进行对象的互相转换
12. URLEncoder和URLDecoder作用是什么?
13. URI和Uri的区别
14. Uri的特点
15. Uri的组成部分?
16. scheme是什么?
17. authority是什么?
18. host是什么?
19. port是什么?

20. path是什么?
21. fragment是什么?
22. query是什么?
23. scheme-specific-part是什么?
24. getPathSegments()作用
25. getQueryParameter(String key)作用

URI

1、URI是什么?

> 1. 统一资源标识符，uniform resource identifier。
> 2. Java中的类： `java.net.URI`
> 3. URI和URL是采用了两种不同的资源标识方法的URI

2、URL是什么?

> 1. 统一资源定位器, uniform resource locator
> 2. Java中的类： `java.net.URL`
> 3. 是一种具体的URI
> 4. 如：

```
www.baidu.com // 内在就是 = www.baidu.com/index.html
www.baidu.com/index.html
```

3、URN是什么?

> 1. 统一资源命名, uniform resource name
> 2. 通过名字来标识资源
> 3. 如：

```
mailto:java-net@java.sun.com
```

4、URI、URL、URN三者的关系

> 1. URI是一种抽象、高层次的概念。
> 2. URL和URN是具体的标识资源的方法
> 3. URL和URN都是一种URI。

5、URI和URL实例的区别?

> 1. URI实例可以使相对的URI，也可以是绝对的URI。

2. URL作为标识资源的具体方式，不能采用相对的形式，必须要给定 `schema` 等绝对形式需要的信息。
3. URL必须要包含定位该资源的信息。

URI的分类

7、绝对URI和相对URI

1. 绝对URI： `http://fsjohnhuang.cnblogs.com`
2. 相对URI： `fsjohnhuang.cnblogs.com`

8、不透明URI和分层URI

1. 不透明URI: `mailto:fsjohnhuang@xxx.com`
2. 分层URI： `http://fsjohnhuang.com`
3. 不透明的URI就是URN
4. 分层的URI就是URL

URL

9、相对URL和绝对URL?

1. 通常所说的相对URL，也是针对于绝对URL的，本质上还是绝对的。
2. 相对URL，不包含scheme。
3. 绝对URL，包含scheme。

10、Java中的URL类不提供对特殊字符的转义

1. 标准RFC2396规定了需要转义的特殊字符
2. 需要自行进行转义

11、URI提供了转义功能

12、URI和URL如何进行对象的互相转换

1. URI.toURL()： 转换为URL
2. URL.toURI()： 转换为URI

URLEncoder和URLDecoder

1、URLEncoder和URLDecoder作用是什么?

1. 用于特殊字符的转义，比如 请求参数中有中文
2. URLEncoder.encode()： 进行编码
3. URLDecoder.decode()： 进行解码

2、编码一次，服务端解码实例。

```
// 编码一次
String params = URLEncoder.encode("编码测试", "UTF-8");
System.out.println(params);

String result = URLDecoder.decode(params, "UTF-8");
System.out.println(result);

// 结果
%E7%BC%96%E7%A0%81%E6%B5%8B%E8%AF%95
编码测试
```

3、服务端解码出现乱码，需要客户端两次编码的实例。

```
System.out.println("编码测试");
// 编码一次
String params = URLEncoder.encode("编码测试", "UTF-8");
System.out.println(params);
// 编码两次
String doubleParams = URLEncoder.encode(params, "UTF-8");
System.out.println(doubleParams);

// 解码一次
System.out.println("\n====服务端开始解码====");
String result = URLDecoder.decode(doubleParams, "UTF-8");
System.out.println(result);
// 解码两次
result = URLDecoder.decode(params, "UTF-8");
System.out.println(result);
```

Uri

1、URI和Uri的区别

> 所属的包不同：
>
> 1. URI位置在java.net.URI,是Java提供的一个类。
> 2. Uri位置在android.net.Uri,是由Android提供的一个类。
>   作用的不同:
> 3. URI类代表了一个URI（这个URI不是类，而是其本来的意义：通用资源标志符——Uniform Resource Identifier)实例。
> 4. Uri类是一个不可改变的URI引用，包括一个URI和一些碎片，URI跟在“#”后面。建立并且转换URI引用。而且Uri类对无效的行为不敏感，对于无效的输入没有定义相应的行为，如果没有额外制定，它将返回垃圾而不是抛出一个异常。

2、Uri的特点

> 1. Uri是URI的“扩展”以适应Android系统的需要。

3、Uri实例解析

```
http://www.java2s.com:8080/yourpath/fileName.htm?stove=10&path=32&id=4#harvic
```

部分	内容	方法
scheme	http	getScheme()
authority	host+port: www.java2s.com:8080	getAuthority()
host	www.java2s.com	getHost()
port	8080	getPort()
path	authority和query中间部分: /yourpath/fileName.htm	getPath()
fragment	#号后面: harvic	getFragment()
query	参数部分: stove=10&path=32&id=4	getQuery()
scheme-specific-part	http:和frgament之间的内容: //www.java2s.com:8080/yourpath/fileName.htm?stove=10&path=32&id=4	getSchemeSpecificPart()

4、getPathSegments()作用

> 1.将 `Path` 整个部分接取下来，并拆分成字符串数组

```
String mUriStr = "http://www.java2s.com:8080/yourpath/fileName.htm?stove=10&path=32&id=4#harvic
Uri mUri = Uri.parse(mUriStr);
List<String> pathSegList = mUri.getPathSegments();
for (String pathItem:pathSegList){
    Log.d("getPathSegments","pathSegItem:"+pathItem);
}
```

> 2.Path被拆分为“yourpath”和“fileName.htm”

5、getQueryParameter(String key)-获取参数中key对应的value

> 能获取到stove、id、path对应的数值，如果value不存在会返回 `null`

```
String mUriStr = "http://www.java2s.com:8080/yourpath/fileName.htm?stove=10&path=32&id#harvic";
mUri = Uri.parse(mUriStr);
Log.d(tag,"stove="+mUri.getQueryParameter("stove"));
Log.d(tag,"id="+mUri.getQueryParameter("id"));
```

参考资料

1. Uri详解之——Uri结构与代码提取
2. 为什么java的web开发中URLEncoder.encode方法要为什么要调用两次
3. URLEncoder.encode 使用心得