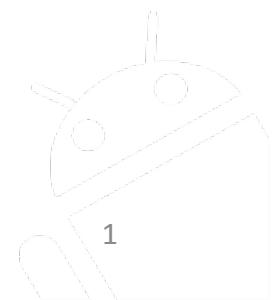




#13 数据存储





学习目标

- 1、掌握sharedPreferences的使用方法；
- 2、掌握各种文件存储的区别与适用情况；
- 3、SQLite数据库的使用；
- 4、ContentProvider的使用。





数据存取

Android有4种数据存取的方式：

- SharedPreferences

轻量级**NVP** (Name/Value Pair, 名称/值对) 方式存储，以**XML**的文件方式保存；

- 文件

采用**java.io.***库提供的I/O接口读写文件；

- SQLite数据库

轻量级嵌入式内置**数据库**；

- ContentProvider

封装各种数据源（文件、数据库、网络），**共享**给多个应用。





13.1 SharedPreferences存储

应用场景：

- 保存播放位置：用手机播放器播放音乐，我们希望重启播放器时，播放器能从上次停止的那首曲目开始播放，如何实现？
- 自动登录：记住登录用户名（密码）
- 其他应用？





13.1 SharedPreferences存储

什么是SharedPreferences？

- 一种轻量级的数据保存方式。
- 类似于我们常用的ini文件，用来保存应用程序的一些属性设置、较简单的参数设置。
- 保存现场：保存用户所作的修改或者自定义参数设定，当再次启动程序后回复上次退出时的状态。





13.1 SharedPreferences存储

什么是SharedPreferences？

- 将**NVP** (Name/Value Pair, 名称/值对) 保存在Android的文件系统中 (XML文件)，完全屏蔽的对文件系统的操作过程。
 - NVP举例: (姓名,张三), (性别,男), (年龄,30), ...
- 开发人员仅是通过调用SharedPreferences的API对NVP进行保存和读取。





13.1 SharedPreferences存储

什么是SharedPreferences？

- 除数据保存，还提供数据共享功能。
- 主要支持3种数据访问模式（读写权限）
 - 私有（MODE_PRIVATE）：仅创建程序可读、写
 - 全局读（MODE_WORLD_READABLE）：创建程序可读写，其他程序可读不可写
 - 全局写（MODE_WORLD_WRITEABLE）：创建程序和其他程序都可写，但不可读！

Ps：后两种模式可组合，用+号或|号。





13.1 SharedPreferences存储

使用方法：

– 第1步：定义访问模式

- 下面的代码将访问模式定义为私有模式

```
public static int MODE = MODE_PRIVATE
```

- 访问模式可组合：既可以全局读，也可以全局写，将两种模式组合（+号或|号）成下面的方式：

```
public static int MODE =
    Context.MODE_WORLD_READABLE +
    Context.MODE_WORLD_WRITEABLE
```





13.1 SharedPreferences存储

使用方法：

- 第2步：定义SharedPreferences的名称
 - 该名称与Android文件系统中保存的XML文件同名。
 - (保存在:/data/data/<package name>/shared_prefs/)
 - 相同名称的NVP内容，都会保存在同一个文件中。

```
public static final String PREFERENCE_NAME = "SaveSetting";
```





13.1 SharedPreferences存储

使用方法：

- 第3步：创建SharedPreferences对象
 - 将**访问模式**和**名称**作为参数，传递到getSharedPreferences()函数，并获得SharedPreferences对象

```
SharedPreferences sharedPreferences =  
    getSharedPreferences ( PREFERENCE_NAME, MODE );
```





13.1 SharedPreferences存储

使用方法：

- 第4步：修改与保存
 - 通过**SharedPreferences.Editor类**进行修改
 - 调用**commit()**函数保存修改内容
- 支持数据类型：**整型、布尔型、浮点型和长整型等**

```
SharedPreferences.Editor editor = sharedPreferences.edit();
editor.putString("Name", "Tom");
editor.putInt("Age", 20);
editor.putFloat("Height", (float)163.00 );
editor.commit();
```





13.1 SharedPreferences存储

使用方法：

- 第5步：读取数据(可紧接第3步)
 - (先调用getSharedPreferences()函数获得对象)
 - 通过get<Type>()函数获取NVP
 - 第1个参数是NVP的名称(Name)
 - **第2个参数是在无法获取到数值的时候使用的缺省值**

```
SharedPreferences sharedPreferences =
    getSharedPreferences(PREFERENCE_NAME, MODE);
String name = sharedPreferences.getString("Name", "Default Name");
int age = sharedPreferences.getInt("Age", 20);
```





13.1 SharedPreferences存储

示例：

- 我们已经知道了用Preferences来存取数据，那么这些数据究竟被保存在手机的什么地方了呢？
- 每安装一个应用程序时，SharedPreferences文件就保存在
/data/data/<package name>/shared_prefs/
目录下，其中的就是我们的数据文件。





13.1 SharedPreferences存储

示例：

- 如何读取程序SharesPreferences的数据？
- 可通过DDMS的FileExplorer查看/data/data下的数据，Android为每个应用程序建立了与包同名的目录，用来保存应用程序产生的数据，这些数据包括文件、SharedPreferences文件和数据库等





13.1 SharedPreferences存储

示例：

- 名为edu.hrbeu.SimplePreferenceDemo的程序，其shared_prefs目录生成了一个SaveSetting.xml文件

[-] edu.hrbeu.SimplePreferenceDemo	2009-07-10 02:18	drwxr-xr-x
[+] lib	2009-07-10 02:18	drwxr-xr-x
[+] shared_prefs	2009-07-10 03:01	drwxrwx--x
SaveSetting.xml	170 2009-07-15 08:45	-rw-rw-rw-
[+] edu.hrbeu.SimpleRandomServiceDemo	2009-06-30 12:17	drwxr-xr-x
[+] edu.hrbeu.SpinnerDemo	2009-06-21 07:01	drwxr-xr-x

该文件名取决于之前第2步的代码：

public static final String PREFERENCE_NAME = "SaveSetting";

- 这个文件就是保存SharedPreferences的文件，文件大小为170字节，在Linux下的权限为“-rw-rw-rw”





13.1 SharedPreferences存储

示例：DDMS > File Explorer

The screenshot shows the Android Device Monitor (DDMS) interface. The top bar has icons for file operations (copy, paste, delete), an Android device icon (circled in red), and other tools. The menu bar includes File, Edit, Run, Window, and Help. The tab bar shows Devices, Threads, Heap, Allocation Tracker, Network Statistics, File Explorer (circled in red), Emulator Control, and System Information. The main area displays a file tree on the left and a detailed file list on the right. The file list table has columns for Name, Size, Date, Time, Permissions, and Info. The permissions column shows standard Unix-style permissions like drwxr-xr-x. The file tree shows directories like acct, cache, config, d, data, default.prop, dev, etc, init, init.goldfish.rc, init.rc, init.trace.rc, init.usb.rc, mnt, proc, root, sbin, sdcard, sys, and system. The file list table contains many entries, including several 'init.' files and 'ueventd.' files.

Name	Size	Date	Time	Permissions	Info
acct		2015-09-05	15:25	drwxr-xr-x	
cache		2015-09-03	14:58	drwxrwx---	
config		2015-09-05	15:25	dr-x-----	
d		2015-09-05	15:25	lrwxrwxrwx	-> /sys/ker...
data	116	1970-01-01	00:00	drwxrwx--x	
default.prop		2015-09-03	13:12	-rw-r--r--	
dev		2015-09-05	15:25	drwxr-xr-x	
etc		2015-09-05	15:25	lrwxrwxrwx	-> /system...
init	234929	1970-01-01	00:00	-rwxr-x---	
init.goldfish.rc	2344	1970-01-01	00:00	-rwxr-x---	
init.rc	17057	1970-01-01	00:00	-rwxr-x---	
init.trace.rc	1637	1970-01-01	00:00	-rwxr-x---	
init.usb.rc	3915	1970-01-01	00:00	-rwxr-x---	
mnt		2015-09-05	15:25	drwxrwxr-x	
proc		2015-09-05	15:25	dr-xr-xr-x	
root		2012-08-23	04:23	drwx-----	
sbin		1970-01-01	00:00	drwxr-x---	
sdcard		2015-09-05	15:25	lrwxrwxrwx	-> /mnt/sd...
sys		2015-09-05	15:25	drwxr-xr-x	
system		2015-03-24	00:43	drwxr-xr-x	
ueventd.goldfish.rc	272	1970-01-01	00:00	-rw-r--r--	
ueventd.rc	3879	1970-01-01	00:00	-rw-r--r--	



13.1 SharedPreferences存储

示例：文件权限说明

- 在Linux系统中，文件权限描述符每3个字符分别描述了**创建者（第1–3字符）**、**同组用户（第4–6字符）**和**其他用户（第7–9字符）**对文件的操作限制（权限）；
- 每字符意义：x表示可执行，r表示可读，w表示可写，d表示目录，-表示普通文件(无操作权限)。
- 例如：“-rw-rw-rw” 表示SaveSetting.xml可以被创建者、同组用户和其他用户进行读取和写入操作，但不可执行。





13.1 SharedPreferences存储

示例：文件权限说明

- 产生这样的文件权限与程序人员设定的 SharedPreferences 的访问模式有关，“-rw-rw-rw” 的权限是“全局读+全局写”的结果
- 如果将SharedPreferences的访问模式设置为私有，则文件权限将成为“-rw-rw ---”，表示仅有创建者和同组用户具有读写文件的权限





13.1 SharedPreferences存储

示例：数据文件格式

- SaveSetting.xml文件是以XML格式保存的信息，信息内容如下：

```
<?xml version='1.0' encoding='utf-8' standalone='yes' ?>
<map>
<float name="Height" value="1.81" />
<string name="Name">Tom</string>
<int name="Age" value="20" />
</map>
```

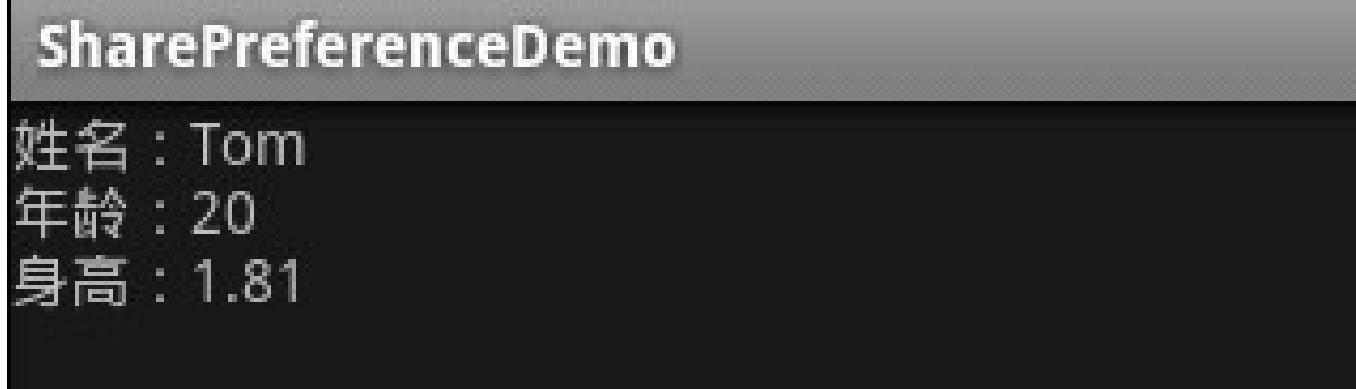




13.1 SharedPreferences存储

示例：共享的实现

- 如何读取其他应用程序保存的SharedPreferences数据
- 示例将读取程序名为SimplePreferenceDemo保存的信息，在程序启动时显示在用户界面上





13.1 SharedPreferences存储

示例：SharePreferenceDemo示例的核心代码

```
public static final String PREFERENCE_PACKAGE =  
    "edu.hrbeu.SimplePreferenceDemo";  
  
public static int MODE = Context.MODE_WORLD_READABLE +  
    Context.MODE_WORLD_WRITEABLE;  
public static final String PREFERENCE_NAME = "SaveSetting";  
  
public void onCreate(Bundle savedInstanceState) {  
    Context c = null;  
    try {  
        //获取SimplePreferenceDemo示例的Context  
        c = this.createPackageContext(PREFERENCE_PACKAGE,  
            Context.CONTEXT_IGNORE_SECURITY);  
    }  
    catch (NameNotFoundException e) {e.printStackTrace();}  
    //将正确的SharedPreferences名称传递给函数  
    SharedPreferences sharedPreferences =  
        c.getSharedPreferences( PREFERENCE_NAME, MODE );  
    //读取NVP  
    String name = sharedPreferences.getString("Name", "Tom");  
    int age = sharedPreferences.getInt("Age", 20);  
    float height = sharedPreferences.getFloat("Height", );  
}
```



(新)定义要访问的应用的名称

第1—2步: 定义模式、名称

(新)创建要访问的应用的上下文

第3步：得到上下文的SP

第5步: 读NVP



13.1 SharedPreferences存储

示例：访问共享设置的额外步骤

- 第1行**定义要访问的应用的包名**
- 第8行代码调用了createPackageContext()**获取到了要访问的应用的上下文Context**
 - 第1个参数是要访问的应用的包名称
 - 第2个参数Context.CONTEXT_IGNORE_SECURITY表示忽略所有可能产生的安全问题。
 - 这段代码可能引发异常，因此必须放在try/catch中。
- 在代码第11行，通过Context**得到要访问应用的SharedPreferences对象**。
 - （有别于访问自己配置过程的第3步）
 - 同样在getSharedPreferences()函数中，需要将正确的SharedPreferences名称、访问模式传递给函数。





13.1 SharedPreferences存储

示例：访问其他应用程序的共享设置必须满足三个条件

- 共享者需要将SharedPreferences的访问模式设置为全局读或全局写
- 访问者需要知道共享者的包名称和SharedPreferences的名称，以通过Context获得SharedPreferences对象
- 访问者需要确切知道每个数据的名称和数据类型，用以正确读取数据

