新手必备的常用 Android 代码片段整理（2）

2016-01-21 安卓应用频道 安卓应用频道

(点击上方公众号，可快速关注)

来源：赵凯强（@裸奔的凯子哥）

链接：http://blog.csdn.net/zhaokaiqiang1992/article/details/44724687

本系列：
新手必备的常用 Android 代码片段整理（1）

以下内容来自多个开源项目的整理和自己的项目积累

1.收集设备信息，用于信息统计分析

```
public static Properties collectDeviceInfo(Context context) {
        Properties mDeviceCrashInfo = new Properties();
        try {
            PackageManager pm = context.getPackageManager();
            PackageInfo pi = pm.getPackageInfo(context.getPackageName(),
                    PackageManager.GET_ACTIVITIES);
            if (pi != null) {
                mDeviceCrashInfo.put(VERSION_NAME,
                        pi.versionName == null ? "not set" : pi.versionName);
                mDeviceCrashInfo.put(VERSION_CODE, pi.versionCode);
            }
        } catch (PackageManager.NameNotFoundException e) {
            Log.e(TAG, "Error while collect package info", e);
        }
        Field[] fields = Build.class.getDeclaredFields();
        for (Field field : fields) {
            try {
                field.setAccessible(true);
                mDeviceCrashInfo.put(field.getName(), field.get(null));
            } catch (Exception e) {
                Log.e(TAG, "Error while collect crash info", e);
            }
        }

        return mDeviceCrashInfo;
    }

public static String collectDeviceInfoStr(Context context) {
        Properties prop = collectDeviceInfo(context);
        Set deviceInfos = prop.keySet();
        StringBuilder deviceInfoStr = new StringBuilder("{\n");
        for (Iterator iter = deviceInfos.iterator(); iter.hasNext();) {
            Object item = iter.next();
            deviceInfoStr.append("\t\t\t" + item + ":" + prop.get(item)
                    + ", \n");
        }
        deviceInfoStr.append("}");
        return deviceInfoStr.toString();
    }
```

2.是否有SD卡

```
public static boolean haveSDCard() {
        return android.os.Environment.getExternalStorageState().equals(
                android.os.Environment.MEDIA_MOUNTED);
    }
```

3.动态隐藏软键盘

```
@TargetApi(Build.VERSION_CODES.CUPCAKE)
    public static void hideSoftInput(Activity activity) {
        View view = activity.getWindow().peekDecorView();
        if (view != null) {
            InputMethodManager inputmanger = (InputMethodManager) activity
                    .getSystemService(Context.INPUT_METHOD_SERVICE);
            inputmanger.hideSoftInputFromWindow(view.getWindowToken(), 0);
        }
    }

    @TargetApi(Build.VERSION_CODES.CUPCAKE)
public static void hideSoftInput(Context context, EditText edit) {
        edit.clearFocus();
        InputMethodManager inputmanger = (InputMethodManager) context
                .getSystemService(Context.INPUT_METHOD_SERVICE);
        inputmanger.hideSoftInputFromWindow(edit.getWindowToken(), 0);
    }
```

4.动态显示软键盘

```
@TargetApi(Build.VERSION_CODES.CUPCAKE)
public static void showSoftInput(Context context, EditText edit) {
        edit.setFocusable(true);
        edit.setFocusableInTouchMode(true);
        edit.requestFocus();
        InputMethodManager inputManager = (InputMethodManager) context
                .getSystemService(Context.INPUT_METHOD_SERVICE);
        inputManager.showSoftInput(edit, 0);
    }
```

5.动态显示或者是隐藏软键盘

```
@TargetApi(Build.VERSION_CODES.CUPCAKE)
public static void toggleSoftInput(Context context, EditText edit) {
        edit.setFocusable(true);
        edit.setFocusableInTouchMode(true);
        edit.requestFocus();
        InputMethodManager inputManager = (InputMethodManager) context
                .getSystemService(Context.INPUT_METHOD_SERVICE);
        inputManager.toggleSoftInput(InputMethodManager.SHOW_FORCED, 0);
    }
```

6.主动回到Home，后台运行

```
public static void goHome(Context context) {
        Intent mHomeIntent = new Intent(Intent.ACTION_MAIN);
        mHomeIntent.addCategory(Intent.CATEGORY_HOME);
        mHomeIntent.addFlags(Intent.FLAG_ACTIVITY_NEW_TASK
                | Intent.FLAG_ACTIVITY_RESET_TASK_IF_NEEDED);
        context.startActivity(mHomeIntent);
    }
```

7.获取状态栏高度

注意，要在onWindowFocusChanged中调用，在onCreate中获取高度为0

```
@TargetApi(Build.VERSION_CODES.CUPCAKE)
public static int getStatusBarHeight(Activity activity) {
    Rect frame = new Rect();
    activity.getWindow().getDecorView().getWindowVisibleDisplayFrame(frame);
        return frame.top;
    }
```

8.获取状态栏高度＋标题栏(ActionBar)高度

(注意，如果没有ActionBar，那么获取的高度将和上面的是一样的，只有状态栏的高度)

```
public static int getTopBarHeight(Activity activity) {
        return activity.getWindow().findViewById(Window.ID_ANDROID_CONTENT)
                .getTop();
    }
```

9.获取MCC+MNC代码 (SIM卡运营商国家代码和运营商网络代码)

仅当用户已在网络注册时有效, CDMA 可能会无效（中国移动：46000 46002, 中国联通：46001,中国电信：46003）

```
    public static String getNetworkOperator(Context context) {
        TelephonyManager telephonyManager = (TelephonyManager) context
                .getSystemService(Context.TELEPHONY_SERVICE);
        return telephonyManager.getNetworkOperator();
    }
```

10.返回移动网络运营商的名字

(例：中国联通、中国移动、中国电信) 仅当用户已在网络注册时有效, CDMA 可能会无效)

```
    public static String getNetworkOperatorName(Context context) {
        TelephonyManager telephonyManager = (TelephonyManager) context
                .getSystemService(Context.TELEPHONY_SERVICE);
        return telephonyManager.getNetworkOperatorName();
    }
```

11.返回移动终端类型

PHONE_TYPE_NONE :0 手机制式未知
PHONE_TYPE_GSM :1 手机制式为GSM，移动和联通
PHONE_TYPE_CDMA :2 手机制式为CDMA，电信
PHONE_TYPE_SIP:3

```
    public static int getPhoneType(Context context) {
        TelephonyManager telephonyManager = (TelephonyManager) context
                .getSystemService(Context.TELEPHONY_SERVICE);
        return telephonyManager.getPhoneType();
    }
```

12.判断手机连接的网络类型(2G,3G,4G)

联通的3G为UMTS或HSDPA，移动和联通的2G为GPRS或EGDE，电信的2G为CDMA，电信的3G为EVDO

```
public class Constants {
  /**
   * Unknown network class
   */
  public static final int NETWORK_CLASS_UNKNOWN = 0;

  /**
   * wifi net work
   */
  public static final int NETWORK_WIFI = 1;

  /**
   * "2G" networks
   */
  public static final int NETWORK_CLASS_2_G = 2;

  /**
   * "3G" networks
   */
  public static final int NETWORK_CLASS_3_G = 3;

  /**
   * "4G" networks
   */
  public static final int NETWORK_CLASS_4_G = 4;

}

public static int getNetWorkClass(Context context) {
    TelephonyManager telephonyManager = (TelephonyManager) context
        .getSystemService(Context.TELEPHONY_SERVICE);

    switch (telephonyManager.getNetworkType()) {
    case TelephonyManager.NETWORK_TYPE_GPRS:
    case TelephonyManager.NETWORK_TYPE_EDGE:
    case TelephonyManager.NETWORK_TYPE_CDMA:
    case TelephonyManager.NETWORK_TYPE_1xRTT:
    case TelephonyManager.NETWORK_TYPE_IDEN:
      return Constants.NETWORK_CLASS_2_G;

    case TelephonyManager.NETWORK_TYPE_UMTS:
    case TelephonyManager.NETWORK_TYPE_EVDO_0:
```

```
    case TelephonyManager.NETWORK_TYPE_EVDO_A:
    case TelephonyManager.NETWORK_TYPE_HSDPA:
    case TelephonyManager.NETWORK_TYPE_HSUPA:
    case TelephonyManager.NETWORK_TYPE_HSPA:
    case TelephonyManager.NETWORK_TYPE_EVDO_B:
    case TelephonyManager.NETWORK_TYPE_EHRPD:
    case TelephonyManager.NETWORK_TYPE_HSPAP:
      return Constants.NETWORK_CLASS_3_G;

    case TelephonyManager.NETWORK_TYPE_LTE:
      return Constants.NETWORK_CLASS_4_G;

    default:
      return Constants.NETWORK_CLASS_UNKNOWN;
    }
  }
```

13.判断当前手机的网络类型(WIFI还是2,3,4G)

需要用到上面的方法

```
public static int getNetWorkStatus(Context context) {
    int netWorkType = Constants.NETWORK_CLASS_UNKNOWN;

    ConnectivityManager connectivityManager = (ConnectivityManager) context
        .getSystemService(Context.CONNECTIVITY_SERVICE);
    NetworkInfo networkInfo = connectivityManager.getActiveNetworkInfo();

    if (networkInfo != null && networkInfo.isConnected()) {
      int type = networkInfo.getType();

      if (type == ConnectivityManager.TYPE_WIFI) {
        netWorkType = Constants.NETWORK_WIFI;
      } else if (type == ConnectivityManager.TYPE_MOBILE) {
        netWorkType = getNetWorkClass(context);
      }
    }

    return netWorkType;
  }
```

举报