

项目总结

最近在公司做了一个非常轻量级别的app，不过里面还是有一些知识点，是查了资料之后才会的，现在app基本做完了，整体总结一下。

1.获取当前app的一些基础信息:

```
public static final boolean DEBUG = BuildConfig.DEBUG;

//以下是能获取到的信息
public static final boolean DEBUG = Boolean.parseBoolean("true");
public static final String APPLICATION_ID = "com.fuyizhulao.daily_service";
public static final String BUILD_TYPE = "debug";
public static final String FLAVOR = "";
public static final int VERSION_CODE = 1;
public static final String VERSION_NAME = "1.0.0";
```

2.高德地图(具体用法见官网吧，文档很详细了)

3.个推(一个推送服务，很好用，同时它的别名机制也很人性化，省着后台再维护一套映射了)

4.Bugly(腾讯的一款崩溃统计，异常上报的SDK，非常好配置，非常的好用)

5.配置回调类的时候，可以配制成泛型的，非常的方便、灵活

```
public interface NetWorkListener<T> {

    void onSuccess(T netWorkModel);

    void onFail(T netWorkModel);

}
```

6.在Fragment中想和Activity通信，可以通过EventBus进行通信，但是也不要过分依赖这个吧，因为当过分依赖之后，整个代码的逻辑会变得异常的复杂，同时发生异常之后，也是非常不好检查的。

7.在app中无论一个网络接口出现过几次，最好还是统一的封装起来会比较方便一点。

8.将Model专程Json字符串可以用Gson:

```
new Gson().toJson(new GetOrderList(
    Integer.valueOf(MyApplication.id)
    , state
    , pageNum
    , pageSize))
```

9.得到相机的View和控制闪光灯:

```
public class CameraView extends SurfaceView implements SurfaceHolder.Callback, Camera.PreviewCallback {

    private Camera mCamera;

    private int mPreviewRotation = 90;
    private int mCamId = Camera.CameraInfo.CAMERA_FACING_BACK;
    private PreviewCallback mPrevCb;
    private byte[] mYuvPreviewFrame;
    private int previewWidth;
    private int previewHeight;

    private Camera.Parameters params;

    public interface PreviewCallback {
        void onGetYuvFrame(byte[] data);
    }

    public CameraView(Context context) {
        this(context, null);
    }

    public CameraView(Context context, AttributeSet attrs) {
        super(context, attrs);
    }
}
```

```

        super(context, attrs);
    }

    public void setPreviewRotation(int rotation) {
        mPreviewRotation = rotation;
    }

    public void setCameraId(int id) {
        mCamId = id;
    }

    public int getCameraId() {
        return mCamId;
    }

    public void setPreviewCallback(PreviewCallback cb) {
        mPrevCb = cb;
        getHolder().addCallback(this);
    }

    public void setPreviewResolution(int width, int height) {
        previewWidth = width;
        previewHeight = height;
    }

    public boolean startCamera() {
        if (mCamera != null) {
            return false;
        }
        if (mCamId > (Camera.getNumberOfCameras() - 1) || mCamId < 0) {
            return false;
        }

        mCamera = Camera.open(mCamId);

        params = mCamera.getParameters();
        Camera.Size size = mCamera.new Size(previewWidth, previewHeight);

        mYuvPreviewFrame = new byte[previewWidth * previewHeight * 3 / 2];

        params.setPreviewSize(previewWidth, previewHeight);

        params.setPreviewFormat(ImageFormat.NV21);

        List<String> supportedFocusModes = params.getSupportedFocusModes();

        if (!supportedFocusModes.isEmpty()) {
            if (supportedFocusModes.contains(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE)) {
                params.setFocusMode(Camera.Parameters.FOCUS_MODE_CONTINUOUS_PICTURE);
            }
        }

        mCamera.setParameters(params);

        mCamera.setDisplayOrientation(mPreviewRotation);

        mCamera.addCallbackBuffer(mYuvPreviewFrame);
        mCamera.setPreviewCallbackWithBuffer(this);
        try {
            mCamera.setPreviewDisplay(getHolder());
        } catch (IOException e) {
            e.printStackTrace();
        }
        mCamera.startPreview();

        return true;
    }

    public void stopCamera() {
        if (mCamera != null) {
            mCamera.setPreviewCallback(null);
            mCamera.stopPreview();
            mCamera.release();
            mCamera = null;
        }
    }

    @Override
    public void onPreviewFrame(byte[] data, Camera camera) {
        mPrevCb.onGetYuvFrame(data);
        camera.addCallbackBuffer(mYuvPreviewFrame);
    }

```

```

@Override
public void surfaceChanged(SurfaceHolder holder, int format, int width, int height) {
}

@Override
public void surfaceCreated(SurfaceHolder arg0) {
    if (mCamera != null) {
        try {
            mCamera.setPreviewDisplay(getHolder());
        } catch (IOException e) {
            e.printStackTrace();
        }
    }
}

@Override
public void surfaceDestroyed(SurfaceHolder arg0) {
}

public void isOpenLight(boolean isOpen) {
    if (isOpen) {
        params.setFlashMode(Camera.Parameters.FLASH_MODE_TORCH);
        mCamera.setParameters(params);
    } else {
        params.setFlashMode(Camera.Parameters.FLASH_MODE_OFF);
        mCamera.setParameters(params);
    }
}
}

```

10.Notification

```

NotificationManager mNotificationManager = (NotificationManager) getSystemService(NOTIFICATION_SERVICE);
NotificationCompat.Builder mBuilder = new NotificationCompat.Builder(this);
mBuilder.setContentTitle(title) //设置通知栏标题
    .setContentIntent(getDefalutIntent(Notification.FLAG_AUTO_CANCEL)) //设置通知栏点击意图
    .setNumber(number) //设置通知集合的数量
    .setTicker(title) //通知首次出现在通知栏，带上升动画效果的
    .setWhen(System.currentTimeMillis()) //通知产生的时间，会在通知信息里显示，一般是系统获取到的时间
    .setPriority(Notification.PRIORITY_DEFAULT) //设置该通知优先级
    .setAutoCancel(true) //设置这个标志当用户单击面板就可以让通知将自动取消
    .setOngoing(false) //ture，设置他为一个正在进行的通知。他们通常是用来表示一个后台任务，用户积极参与（如播放音乐）
    .setDefaults(Notification.DEFAULT_VIBRATE) //向通知添加声音、闪灯和振动效果的最简单、最一致的方式是使用当前系统默认设置
    //Notification.DEFAULT_ALL Notification.DEFAULT_SOUND 添加声音 // requires VIBRATE permission
    .setSmallIcon(R.mipmap.ic_launcher) //设置通知小ICON
mNotificationManager.notify(1, mBuilder.build());

```

11.沉浸式状态栏

这是一个第三方库，用起来挺方便的，有空应该撸一遍它的源码

```
compile 'com.readystatesoftware.systembartint:systembartint:1.0.3'
```

```

if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.KITKAT) {
    //透明状态栏
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_STATUS);
    //透明导航栏
    getWindow().addFlags(WindowManager.LayoutParams.FLAG_TRANSLUCENT_NAVIGATION);
    SystemBarTintManager tintManager = new SystemBarTintManager(this);
    // 激活状态栏
    tintManager.setStatusBarTintEnabled(true);
    // enable navigation bar tint 激活导航栏
    //tintManager.setNavigationBarTintEnabled(true);
    //设置系统栏设置颜色
    //tintManager.setTintColor(R.color.red);
    //给状态栏设置颜色
    tintManager.setStatusBarTintResource(R.color.normal_blue);
    //Apply the specified drawable or color resource to the system navigation bar.
    //给导航栏设置资源
    //tintManager.setNavigationBarTintResource(R.color.normal_blue);
}

```

```

android:fitsSystemWindows="true"
android:clipToPadding="true"

```

12.利用反射动态调节tablayout的长度

```
public void setIndicator(TabLayout tabs, int leftDip, int rightDip) {
    Class<?> tabLayout = tabs.getClass();
    Field tabStrip = null;
    try {
        tabStrip = tabLayout.getDeclaredField("mTabStrip");
    } catch (NoSuchFieldException e) {
        e.printStackTrace();
    }

    tabStrip.setAccessible(true);
    LinearLayout llTab = null;
    try {
        llTab = (LinearLayout) tabStrip.get(tabs);
    } catch (IllegalAccessException e) {
        e.printStackTrace();
    }

    int left = (int) TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP, leftDip, Resources.ge
    int right = (int) TypedValue.applyDimension(TypedValue.COMPLEX_UNIT_DIP, rightDip, Resources.

    for (int i = 0; i < llTab.getChildCount(); i++) {
        View child = llTab.getChildAt(i);
        child.setPadding(0, 0, 0, 0);
        LinearLayout.LayoutParams params = new LinearLayout.LayoutParams(0, LinearLayout.LayoutPa
        params.leftMargin = left;
        params.rightMargin = right;
        child.setLayoutParams(params);
        child.invalidate();
    }
}
```

13.验证码倒计时

```
mTimeCount = new TimeCount(60000, 1000);
mTimeCount.start();
```

```
class TimeCount extends CountDownTimer {
    public TimeCount(long millisInFuture, long countDownInterval) {
        super(millisInFuture, countDownInterval);
    }

    @Override
    public void onFinish() { // 计时完毕
        mGetCode.setText("获取验证码");
        mGetCode.setClickable(true);
    }

    @Override
    public void onTick(long millisUntilFinished) { // 计时过程
        mGetCode.setClickable(false); // 防止重复点击
        mGetCode.setText(String.valueOf(millisUntilFinished / 1000) + "秒后重发");
    }
}
```

14.downloadManager工具类

```
public class DownloadUtils {

    private DownloadManager mDownloadManager;
    private Context mContext;
    private long downloadId;
    private String apkName;

    public DownloadUtils(Context context) {
        mContext = context;
    }

    public void download(String url, String name) {
        final String packageName = "com.android.providers.downloads";
        int state = mContext.getPackageManager().getApplicationEnabledSetting(packageName);
        // 检测下载管理器是否被禁用
        if (state == PackageManager.COMPONENT_ENABLED_STATE_DISABLED
            || state == PackageManager.COMPONENT_ENABLED_STATE_DISABLED_USER
            || state == PackageManager.COMPONENT_ENABLED_STATE_DISABLED_UNTIL_USED) {
```

```

AlertDialog.Builder builder = new AlertDialog.Builder(mContext).setTitle("温馨提示").setMessage(
    ("系统下载管理器被禁止，需手动打开").setPositiveButton("确定", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
            try {
                Intent intent = new Intent(Settings.ACTION_APPLICATION_DETAILS_SETTINGS);
                intent.setData(Uri.parse("package:" + packageName));
                mContext.startActivity(intent);
            } catch (ActivityNotFoundException e) {
                Intent intent = new Intent(Settings.ACTION_MANAGE_APPLICATIONS_SETTINGS);
                mContext.startActivity(intent);
            }
        }
    }).setNegativeButton("取消", new DialogInterface.OnClickListener() {
        @Override
        public void onClick(DialogInterface dialog, int which) {
            dialog.dismiss();
        }
    }));
builder.create().show();
} else {
    // 正常下载流程
    apkName = name;
    DownloadManager.Request request = new DownloadManager.Request(Uri.parse(url));
    request.setAllowedOverRoaming(false);

    // 通知栏显示
    request.setNotificationVisibility(DownloadManager.Request.VISIBILITY_VISIBLE_NOTIFY_COMPLETED);
    request.setTitle("test");
    request.setDescription("正在下载中...");
    request.setVisibleInDownloadsUi(true);

    // 设置下载的路径
    request.setDestinationInExternalPublicDir(Environment.DIRECTORY_DOWNLOADS, apkName);

    // 获取DownloadManager
    mDownloadManager = (DownloadManager) mContext.getSystemService(Context.DOWNLOAD_SERVICE);
    downloadId = mDownloadManager.enqueue(request);

    mContext.registerReceiver(mReceiver, new IntentFilter(DownloadManager.ACTION_DOWNLOAD_COMPLETE));
}

private BroadcastReceiver mReceiver = new BroadcastReceiver() {
    @Override
    public void onReceive(Context context, Intent intent) {
        checkStatus();
    }
};

/**
 * 检查下载状态
 */
private void checkStatus() {
    DownloadManager.Query query = new DownloadManager.Query();
    query.setFilterById(downloadId);
    Cursor cursor = mDownloadManager.query(query);
    if (cursor.moveToFirst()) {
        int status = cursor.getInt(cursor.getColumnIndex(DownloadManager.COLUMN_STATUS));
        switch (status) {
            // 下载暂停
            case DownloadManager.STATUS_PAUSED:
                break;
            // 下载延迟
            case DownloadManager.STATUS_PENDING:
                break;
            // 正在下载
            case DownloadManager.STATUS_RUNNING:
                break;
            // 下载完成
            case DownloadManager.STATUS_SUCCESSFUL:
                installAPK();
                break;
            // 下载失败
            case DownloadManager.STATUS_FAILED:
                Toast.makeText(mContext, "下载失败", Toast.LENGTH_SHORT).show();
                break;
        }
    }
    cursor.close();
}
}

```

```
/**
 * 7.0兼容
 */
private void installAPK() {
    File apkFile =
        new File(Environment.getExternalStoragePublicDirectory(Environment.DIRECTORY_DOWNLOAD
Intent intent = new Intent(Intent.ACTION_VIEW);
intent.setFlags(Intent.FLAG_ACTIVITY_NEW_TASK);
if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.N) {
    Uri apkUri = FileProvider.getUriForFile(mContext, mContext.getPackageName() + ".provider"
intent.addFlags(Intent.FLAG_GRANT_READ_URI_PERMISSION);
intent.setDataAndType(apkUri, "application/vnd.android.package-archive");
} else {
    intent.setDataAndType(Uri.fromFile(apkFile), "application/vnd.android.package-archive");
}
mContext.startActivity(intent);
}
}
```

15.还有一些简单的工具类:

```

public class Util {

    /**
     * 将时间戳转换为时间
     */
    public static String stampToDate(String s) {
        String res;
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("yyyy-MM-dd HH:mm");
        long lt = new Long(s);
        Date date = new Date(lt);
        res = simpleDateFormat.format(date);
        return res;
    }

    /**
     * 这个工具类是为了展示出首页那种持续的效果
     * 例如: 12月21日 09:00 - 10:00
     */
    public static String stampAndDurationToDate(long serviceTime, long duration) {
        SimpleDateFormat simpleDateFormat = new SimpleDateFormat("MM月dd日 HH:mm");
        Date date = new Date(serviceTime);
        String myData = simpleDateFormat.format(date);
        Date date2 = new Date(serviceTime + duration * 60 * 1000);
        String myData2 = simpleDateFormat.format(date2);
        return myData + "-" + myData2.split(" ")[1];
    }

    /**
     * 这个工具类是为了将时间戳转换成订单上面的时间
     * 例如: 2016-12-21 09:00-10:00
     */
    public static String stampAndDurationToDateForOrder(long serviceTime, long duration) {
        String date1 = stampToDate(String.valueOf(serviceTime));
        String date2 = stampToDate(String.valueOf(serviceTime + duration * 60 * 1000));
        return date1 + "-" + date2.split(" ")[1];
    }

    /**
     * bitmap转file
     */
    public static File saveBitmapFile(Bitmap bitmap) {
        File file = new File(Environment.getExternalStorageDirectory().getAbsolutePath()+"/img.png");
        try {
            BufferedOutputStream bos = new BufferedOutputStream(new FileOutputStream(file));
            bitmap.compress(Bitmap.CompressFormat.PNG, 100, bos);
            bos.flush();
            bos.close();
        } catch (IOException e) {
            e.printStackTrace();
        }
        return file;
    }
}

```

16.cardView的点击效果

```
android:foreground="@drawable/card_view_select"
```

```

<?xml version="1.0" encoding="utf-8"?>
<selector xmlns:android="http://schemas.android.com/apk/res/android">

    <item android:drawable="@color/normal_white" android:state_pressed="false"> </item>
    <item android:drawable="@color/press_white" android:state_pressed="true"> </item>

</selector>

```