# 项目中会用到的技术

### 5 . 插件化相关技术，热修补技术是怎样实现的，和插件化有什么区别

相同点:

都使用ClassLoader来实现的加载的新的功能类，都可以使用PathClassLoader与DexClassLoader

不同点：

热修复因为是为了修复Bug的，所以要将新的同名类替代同名的Bug类，要抢先加载新的类而不是Bug类，所以多做两件事：在原先的app打包的时候，阻止相关类去打上CLASS\_ISPREVERIFIED标志，还有在热修复时动态改变BaseDexClassLoader对象间接引用的dexElements，这样才能抢先代替Bug类，完成系统不加载旧的Bug类.

而插件化只是增肌新的功能类或者是资源文件，所以不涉及抢先加载旧的类这样的使命，就避过了阻止相关类去打上CLASS\_ISPREVERIFIED标志和还有在热修复时动态改变BaseDexClassLoader对象间接引用的dexElements.

所以插件化比热修复简单，热修复是在插件化的基础上在进行替旧的Bug类

##### 一. 类的加载过程，Person person = new Person();为例进行说明。

1).因为new用到了Person.class，所以会先找到Person.class文件，并加载到内存中;

2).执行该类中的static代码块，如果有的话，给Person.class类进行初始化;

3).在堆内存中开辟空间分配内存地址;

4).在堆内存中建立对象的特有属性，并进行默认初始化;

5).对属性进行显示初始化;

6).对对象进行构造代码块初始化;

7).对对象进行与之对应的构造函数进行初始化;

8).将内存地址付给栈内存中的p变量

##### 双缓冲技术在绘图中的使用

文章来源：http://www.jianshu.com/u/4d74dd2268af

简单介绍：onDraw方法执行完成之后，才会把数据交给GPU去处理展示。这就是android绘图当中的第一道缓冲，即显示缓冲区。

而所谓的双缓冲，在android绘图中其实就是再创建一个Canvas和对应的Bitmap，然后在onDraw方法里默认的Canvas通过drawBitmap画刚才new的那个bitmap从而实现双缓冲。

##### Retrofit相关

生命周期抽取，通过代理模式：http://andydev.me/2017/07/03/retrofit-lifecycle-management/

##### SDK开发需要考虑的方向

借鉴文章：<http://blog.csdn.net/u010019468/article/details/75425742>

##### 安卓开发过程中的神器App

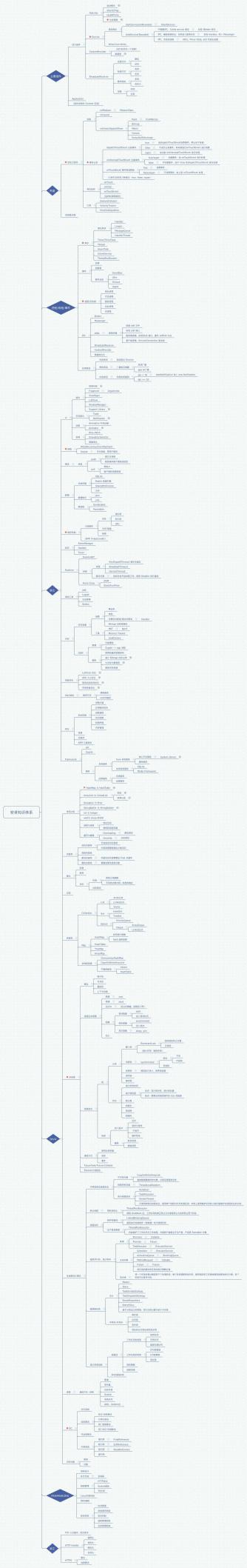
借鉴文章：<https://www.jianshu.com/p/1bdb98c3bf4c>

##### 事件总线

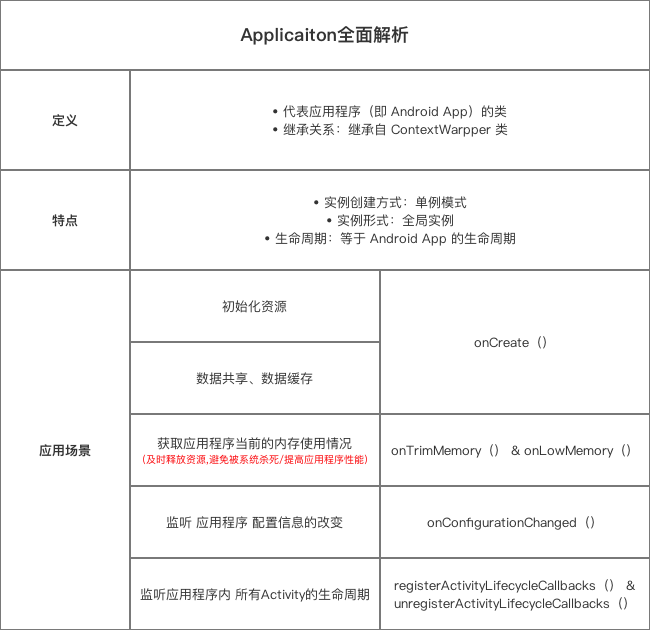
定义：针对事件提供统一订阅，发布以达到组件间通信的解决方案。

原理：观察者模式。发送者把消息发送到事件总线，然后事件总线在接收者中查找哪些方法注册了这个事件，如果某个方法注册了这个事件，就触发该方法。

##### 安卓知识导图

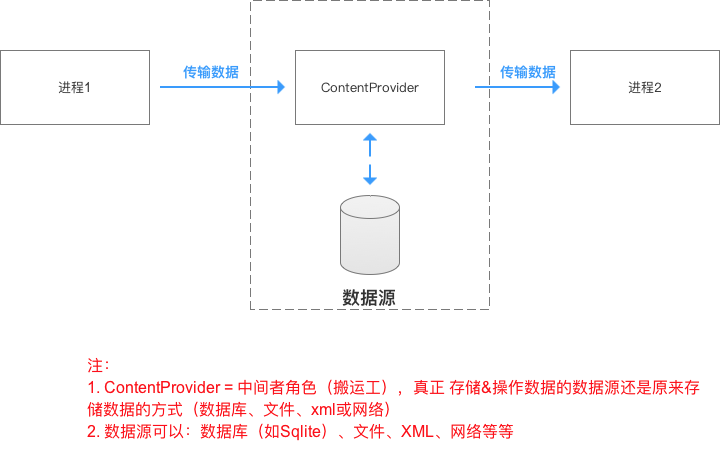


##### Application介绍



##### ContentProvider篇

作用：进程间，进行数据交互&共享，即跨进程通信



原理：Android的Binder机制

##### 时光轴的绘制



借鉴文章：<https://blog.csdn.net/carson_ho/article/details/75005994>

实现方式：通过RecyclerView的ItemDecoration，在偏移里设置装饰

public class TestDividerItemDecoration extends RecyclerView.ItemDecoration {

// 方法1：getItemOffsets（）

// 作用：设置ItemView的内嵌偏移长度（inset）

@Override

public void getItemOffsets(Rect outRect, View view, RecyclerView parent, RecyclerView.State state) {}

// 方法2：onDraw（）

// 作用：在子视图上设置绘制范围，并绘制内容，先于item的onDraw绘制

// 类似平时自定义View时写onDraw()一样

// 绘制图层在ItemView以下，所以如果绘制区域与ItemView区域相重叠，会被遮挡

@Override

public void onDraw(Canvas c, RecyclerView parent, RecyclerView.State state) {}

// 方法3：onDrawOver（）

// 作用：同样是绘制内容，但与onDraw（）的区别是：绘制在图层的最上层

@Override

public void onDrawOver(Canvas c, RecyclerView parent, RecyclerView.State state) {

}

##### getItemOffsets

作用：设置ItemView的内嵌偏移长度（inset）

如图，其实RecyclerView 中的 ItemView 外面会包裹着一个矩形（outRect）

内嵌偏移长度是指：该矩形（outRect）与 ItemView的间隔

