*** 기존 : System Service add방식**
ex> public class VibratorService extends IVibratorService.Stub
...
ServiceManager.addService("vibrator", new VibratorService(context));
*** L 에서는 기존의 방식 + LocalService 와 SystemService 개념을 추가하여, 각각을 구분하여 처리함.**
A. 개념적으로 LocalService 란?
"System Service 의 api 들 중, system_server Process 에서만 사용이 허용되는 internal API 들의 set을 정의하고 있는 Internal class 이다."
여기서 말하는 LocalService 는 실제로 LocalService 라는 이름의 Calss 가 존재한다는 것이 아니라, 개념적으로 말하는 LocalService 이다.
그러므로 LocalService 는 정확히 표현하자면, 첫줄에서 말한대로,
"System Service 의 api 들 중, system_server Process 에서만 사용이 허용되는 internal API 들의 set을 정의하고 있는 단순히 System Service 의 Internal class 인 것이다"
B. LocalServices 란?
@hide 된 final class 로써, LocalService object 들을 ArrayMap 형태로 저장하는 역할을 하는 Storage class.
실제로 ArrayMap 에 저장되는 LocalService object 는 아래 List 참조
어떤 형태의 LocalService object 들을 ArrayMap 에 저장할수 있어야 하므로, ArrayMap 의 key는 <Class> 이고, value 는 <Object> 인 것이다.

C. SystemService 란?
abstract class 로써 System Service 들 중에 Life cycle 에 의해 동작되고, 아래의 Callback 들이 호출될때 noti 를 받아야만 하는 Service.
< Callback 들 : onStart(),onBootPhase(),onStartUser(),onSwitchUser(),onStopUser(),onCleanupUser() >

ServiceManager 에서 등록되어 사용되는개념 과는 직접적인 관련이 없다.
가) SystemService 를 ServiceManager 에서 등록하기 위해서는,
1. 내부적으로 IInterface를 구현하고, Binder 를 상속받은 Remote Service 를 만든뒤, 이를 기존의 방법대로 ServiceManager.addService 를 사용하던지,
2. 내부적으로 ServiceManager.addService API 를 호출해주는 interface api 인 publishBinderService API 를 호출해야 한다.

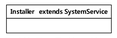
나) SystemService 를 LocalServices 에 등록하기 위해서는
1. 내부적으로 적당한 InternalClass 를 만든뒤, LocalServices.addService API 를 호출하던지,
2. 내부적으로 LocalServices.addService API 를 호출해주는 interface api 인 publishLocalService API 를 호출해야 한다.

D. SystemServiceManager 란?
SystemService 들의 List 를 가지면서, Service 들의 Life cycle 에 따라 Callback 의 수행을 관리함.

E. Class 구조의 모습이 달라진다.<아래 그림 비교 참조>
<Lollipop 이전> : ex> AlarmManagerService 자체가 IAlarmManager.Stub 를 상속 받는 구조로 표현되어 개념상 아래 그림처럼 표현된다.

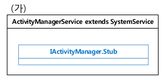
<Lollipop> : ex> AlarmManagerService 가 SystemService 를 상속 받는 구조이므로, IAlarmManager.Stub 를 상속받은 Stub class 를 반드시 Inner class 구조로 가야 한다.

단, Stub 을 제공하지 않는(즉, IPC call 을 허용하지 않는) Installer 같은 경우는 아래와 같은 구조임

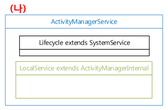


특이한 구조를 하고 있는 경우도 있음(ex> ActvityManagerService)
AlarmManagerService 처럼 (가)와 같은 구조를 유지해야 하지만, 기존에 구현된 내용에 대한 수정이 너무많이 이뤄져야 하므로, (나) 와 같은 구조를 선택한것으로 보임

(가)

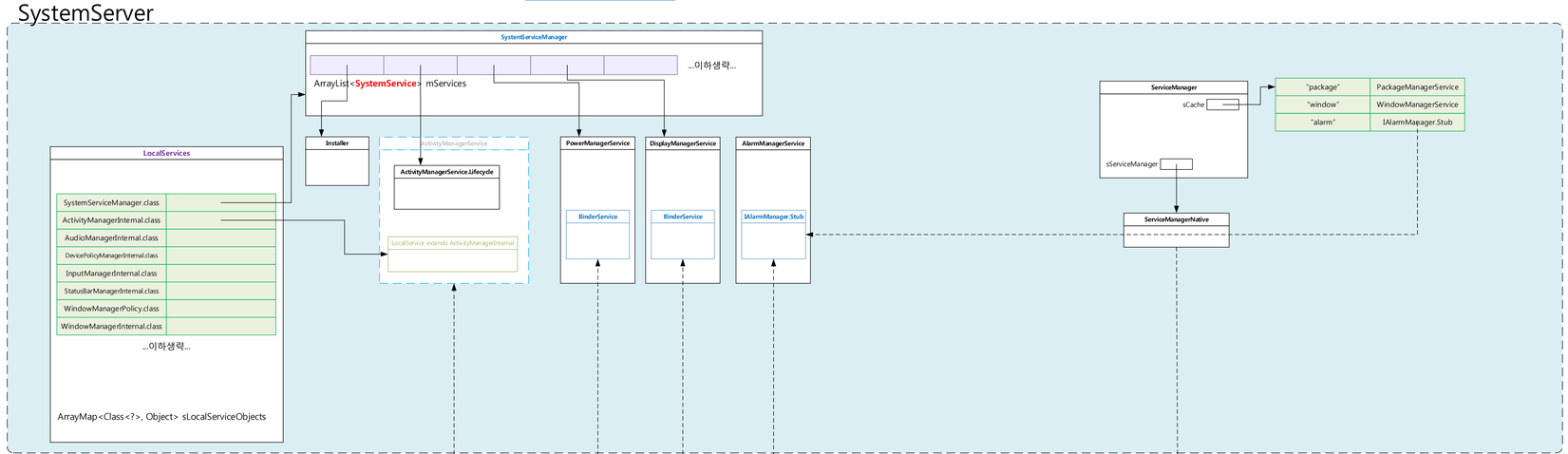


(나)



PID system_server

SystemServer



PID servicemanager