Google Git

Sign in

android / platform / frameworks / base / master / . / services / core / java / com / android / server /
SystemService.java

blob: 94397d07e1a247fa6b0ad5379bb6c7479b5cc813 [file] [log] [blame]

```
/*
 * Copyright (C) 2013 The Android Open Source Project
 *
 * Licensed under the Apache License, Version 2.0 (the "License");
 * you may not use this file except in compliance with the License.
 * You may obtain a copy of the License at
 *
 *      http://www.apache.org/licenses/LICENSE-2.0
 *
 * Unless required by applicable law or agreed to in writing, software
 * distributed under the License is distributed on an "AS IS" BASIS,
 * WITHOUT WARRANTIES OR CONDITIONS OF ANY KIND, either express or implied.
 * See the License for the specific language governing permissions and
 * limitations under the License.
 */

package com.android.server;

import android.app.ActivityThread;
import android.content.Context;
import android.os.IBinder;
import android.os.ServiceManager;
import android.os.UserManager;

/**
 * The base class for services running in the system process. Override and implement
 * the lifecycle event callback methods as needed.
 * <p>
 * The lifecycle of a SystemService:
 * </p><ul>
 * <li>The constructor is called and provided with the system {@link Context}
 * to initialize the system service.
 * <li>{@link #onStart()} is called to get the service running.  The service should
 * publish its binder interface at this point using
 * {@link #publishBinderService(String, IBinder)}.  It may also publish additional
 * local interfaces that other services within the system server may use to access
 * privileged internal functions.
 * <li>Then {@link #onBootPhase(int)} is called as many times as there are boot phases
 * until {@link #PHASE_BOOT_COMPLETED} is sent, which is the last boot phase. Each phase
 * is an opportunity to do special work, like acquiring optional service dependencies,
 * waiting to see if SafeMode is enabled, or registering with a service that gets
 * started after this one.
 * </ul><p>
 * NOTE: All lifecycle methods are called from the system server's main looper thread.
 * </p>
 *
 * {@hide}
 */
public abstract class SystemService {
    /*
     * Boot Phases
     */
    public static final int PHASE_WAIT_FOR_DEFAULT_DISPLAY = 100; // maybe should be a dependency?

    /**
     * After receiving this boot phase, services can obtain lock settings data.
     */
```