D-Bus是一种消息总线系统，是应用程序之间相互通信的一种简单方式。除了进程间通信，D-Bus还有助于协调进程生命周期；它使编写“单实例”应用程序或守护程序变得简单可靠，并在需要应用程序和守护程序的服务时按需启动它们。可以实现：

- 其他应用退出时通知应用

- 按需启动服务

- 支持单实例应用

D-Bus的主要特点是快速和轻量级

libdbus是dbus的一部分，是D-Bus协议的参考实现。是low-level D-Bus C API，指的也就是ibdbus.so里面的接口（因为很多实用功能被集成在了高级库中）

安装及环境配置记录在了csdn：https://blog.csdn.net/oZuoZuo789/article/details/123276920

安装完成后：

这是dbus测试方法的官方文档：<https://dbus.freedesktop.org/doc/dbus-test-plan.html>

Dbus各个函数详细解释：<https://dbus.freedesktop.org/doc/api/html/group__DBus.html>

Dbus简易官方教程：<https://dbus.freedesktop.org/doc/dbus-tutorial.html>

可以从DBusConnection and DBusMessage 开始学习

<https://dbus.freedesktop.org/doc/api/html/index.html>

dbus分为两种总线，一种叫SystemBus，一种叫SessionBus。SystemBus就只有一条，SessionBus是一个用户会话时会产生一条。至于这两种的区别，SystemBus一般是用于权限较高的系统级（root）进程与其他进程（可以是普通进程）的通信，而SessionBus是用于普通的用户进程之间的交流。

dbus是单对单的通信，其实和C/S架构差不多，一个server端接收消息和发布信号，多个client端发送消息和接收信号。

dbus通信的话有5个值需要注意：

1. Address：dbus是通过本地socket来通信，所有会有socket文件。可以直接连接这个sokcet文件的地址来通信。

2. Bus Name：当使用总线守护进程时，只用通过一个Bus Name就可以直接将消息路由到你想要的地址。server端想要Bus Name需要向SystemBus或SessionBus申请。如果不申请连接到dbus，它会自动被分配一个唯一的名字，就是1.45之类的，这数字没什么意义，只是为了名字唯一。名字除了路由消息还有第二种用途，就是当一个程序退出，断开连接，消息总线就会提醒其他连接程序该名字失去了所有者。这样就容易管理其他程序了

3. Path：这个路径是指在进程里的路径，可以按模块来划分，比如NetwrokManager 有 无线和有线这两模块。

4. Interface：这个就像是一组功能的集合名字，可以按功能来划分。

5. Method/Signals：方法和信号，方法其实就是进程里的函数名，你发消息给这个函数名，这个函数就会被调用，并返回结果。信号就是当server端主动调用这个信号函数的时候，便会发出这个信号（信号名就是函数名），其他连接在同一总线上的程序，如果谁感兴趣就会接收处理。

例程代码参考：<https://www.cnblogs.com/tlam/p/15582727.html> （写的很详细！！很厉害）

程序执行流程详解：https://www.cnblogs.com/wangshenyang/p/5781103.html

