

# OSPP 项目申请书

项目名称: 为 treeland 实现锁屏管理协议

项目导师: rewine

申 请 人: 陈俊甫

邮 箱: miruku@miruku.moe

## 目录

1 项目背景 .....	3
1.a 项目基本要求 .....	3
1.b 项目相关仓库 .....	3
2 技术方法及可行性 .....	3
2.a C++语言程序设计 .....	3
2.b Qt/QML 基础 .....	3
2.c Linux / 窗管（合成器）相关 .....	3
3 项目实施细节梳理 .....	3
3.a 锁屏全过程行为梳理 .....	3
3.b 代码结构梳理 .....	3
3.b.a waylib 部分 .....	3
3.b.b treeland 部分 .....	4
3.c 测试流程梳理 .....	4
3.c.a 单元测试 .....	4
3.c.b 兼容性测试 .....	4
3.c.c 稳定性（安全性）测试 .....	4
3.c.d 主线合入 .....	4
4 时间规划 .....	4
4.a 准备阶段（7月1日 – 7月5日） .....	4
4.b 实施阶段（7月6日 – 8月20日） .....	4
4.c 完善阶段（8月1日 – 9月30日） .....	5
5 期望 .....	5

# 1 项目背景

## 1.a 项目基本要求

- 为 treeland 实现完整的 ext-session-lock-v1 协议支持
- 测试锁屏功能，确保支持主流锁屏客户端，功能正确无异常行为

## 1.b 项目相关仓库

- treeland: <https://github.com/linuxdeepin/treeland>
- waylib (treeland 核心开发库) : <https://github.com/vioken/waylib>

# 2 技术方法及可行性

## 2.a C++语言程序设计

waylib 与 treeland 仓库使用 Qt 框架，几乎全部使用 C++ 语言实现。

作为多年的算法竞赛选手，申请人精通 C++ 语言程序设计。与算法竞赛选手的广泛习惯不同的是，申请人**非常注重**代码习惯与代码风格，日常要求自己写出可读、优美、简介且效率不打折扣的代码。

## 2.b Qt/QML 基础

Qt 和 GTK 为 Linux 桌面最主流的两个 GUI 框架，作为多年的 Linux Power User，申请人对 Qt 框架十分熟悉，曾上手对许多基于 Qt 的应用程序进行客制化。

除此之外，申请人**最近使用 QML 实现**了一个多功能的桌面状态栏，其额外支持通知中心、消息弹窗、系统托盘、应用选择、蓝牙选择等功能。为实现这些功能，申请人用到了 QML 的大部分特性，故**对 QML 开发相当熟悉**。

## 2.c Linux / 窗管（合成器）相关

申请人是多年桌面 Linux 用户，多年独立窗管用户，经常自己构建客制化桌面环境，并且正在日用使用 Hyprland 的 NixOS 桌面系统，故对桌面/服务器 Linux 与窗管均**非常熟悉**。

申请人的相关经历请参见个人简历。

# 3 项目实施细节梳理

## 3.a 锁屏全过程行为梳理

ext-session-lock-v1 协议允许受信任的锁屏客户端在会话锁定时显示自定义图形。客户端通过 ext\_session\_lock\_manager\_v1 请求锁屏，创建 ext\_session\_lock\_v1 对象，然后为每个输出创建 ext\_session\_lock\_surface\_v1 并提交缓冲。合成器在发送 locked 事件后必须停止渲染普通客户端，使用不透明背景遮蔽所有输出，只显示锁定表面。用户验证成功后，客户端发送 unlock\_and\_destroy 请求解锁；如果锁屏客户端崩溃，会话保持锁定。

## 3.b 代码结构梳理

### 3.b.a waylib 部分

在 waylib 中为 ext-session-lock 协议添加封装类，包括 ExtSessionLockManagerV1、ExtSessionLockV1 和 ExtSessionLockSurfaceV1 等，映

射协议定义的请求和事件接口，以便在 QtQuick 环境中使用。这样可以利用 waylib 的信号槽机制，在 Treeland 中方便地管理锁屏对象。

### 3.b.b treeland 部分

在 Treeland 中实现完整的锁屏流程。当收到锁屏请求时，Treeland 会生成对应的 ExtSessionLockV1 对象，并为所有当前输出创建锁定表面（ExtSessionLockSurfaceV1）。在接收到 locked 事件后，Treeland 切换到锁屏模式：停止更新普通窗口层，只绘制锁定表面内容。用户通过输入密码解锁后，Treeland 将收到 unlock\_and\_destroy 请求并恢复正常桌面模式。整个流程需严格遵守 configure/ack 等帧同步机制，确保锁屏界面平滑无闪烁。

## 3.c 测试流程梳理

### 3.c.a 单元测试

用于开发中快速发现并解决问题。

### 3.c.b 兼容性测试

使用常见 Wayland 锁屏工具进行测试，包括 swaylock、waylock、gtklock 等。

### 3.c.c 稳定性（安全性）测试

使用虚拟化技术，模拟用户输入，测试锁屏协议性能表现；构造竞争条件、多输出与锁屏客户端异常事件，验证 Treeland 在多输出、重复锁屏、锁屏客户端异常退出等场景下行为是否符合预期。

### 3.c.d 主线合入

代码通过测试后，应及时准备合并。先向 waylib 提交 PR，确保接口合并，再更新 treeland 仓库代码实现 ext-session-lock-v1 支持。

注意编写简明扼要的注释与文档，提供充分的使用例。

项目全程与导师保持流畅沟通，及时回应并评审反馈并进行更正，确保项目落地。

## 4 时间规划

### 4.a 准备阶段（7 月 1 日 – 7 月 5 日）

- 审计 treeland 与 waylib 仓库代码结构
- 阅读 Wayland 协议规范，检查项目行为细节符合规范
- 阅读 deepin 贡献指南
- 改进并最终确定代码实现细节方案，形成书面计划与检查单

### 4.b 实施阶段（7 月 6 日 – 8 月 20 日）

按照准备阶段提出的方案完成锁屏协议实现

- 实现 ext-session-lock-v1 完整协议支持
- 实现锁屏行为（拦截用户输入，屏蔽图形输出）
- 实现异常处理（处理锁屏客户端异常退出，etc.）

#### 4.c 完善阶段（8 月 1 日 – 9 月 30 日）

完善协议实现，测试代码。

- 进行兼容性、稳定性测试，保障锁屏功能正确，符合规范，满足安全假设。
- 总结项目实施，完善相关注释与文档。
- 在现有基础上提出并实现可能的行为/性能改进
- 满足社区要求，将代码合入主线

### 5 期望

作为多年的 Linux 桌面端/服务器端用户，我深受 FOSS 软件生态与社区之益。Deepin 社区是国内富有影响力的社区，我希望借此机会初次正式为开源项目贡献功能代码，跻身开源贡献者的大家庭中。

都说 Wayland 是桌面 Linux 的未来，我也希望能更加了解 Wayland 协议及其实现，参与到亲手搭建这一未来的工作中去。