# 插件钱包

以浏览器插件的形式提供一款钱包，实现代签和代扣。

## 核心思想

1. 传统的插件钱包（metamask，coinbase），在玩游戏过程中，每次和合约交互都会调起插件钱包进行签名确认，大量的确认操作，严重验证影响游戏的体验。

2. 资产在layer1主链上面，在layer2链上的账户需要将原生代币兑换到layer2上才能支付发送合约调用交易的手续费。操作步骤较多，造成体验较差。

**解决方案**

1. 鉴于目前链的架构。layer2链上的游戏具有一定的生命周期。在短暂的生命周期内构造一个临时私钥代替工作私钥签名。由于临时私钥在生命周期结束之后会销毁，安全性可以弱化。在游戏合约运行期间。弱化为不需要每次签名合约交易都弹出对话框进行确认。整个游戏运行期间在后台完成签名，实现用户的无感签名，带来游戏的丝滑体验。

2. 签名交易的手续费统一由游戏发行方提供，在代签名交易执行过程中将gas费用的扣款方更改为游戏的发行方。

## 关键名词

**游戏发行方**：游戏合约的部署者，需要在layer2链上由足够的费用代付游戏合约调用的手续费。

**临时私钥**：代替工作私钥签名的临时性私钥。具有一定的游戏阶段性。阶段开始时候生成，阶段结束销毁。只在阶段中具有有效性。

**工作私钥**：游戏资产需要从layer1转移到layer2上面。在layer2上面具有有效游戏资产的账户私钥，即游戏过程中实际动账的账户私钥。

## 临时私钥生成

在有些阶段开始时，插件钱包内部生成一个工作私钥此阶段对应的临时私钥。可以是 HD 钱包推导路径上的下一个私钥，调用layer2链的系统合约设置工作私钥和临时私钥之间的绑定关系。由工作私钥签名交易。

| 名称 | 类型 | 说明 |
| --- | --- | --- |
| gameContractAddress | address | 游戏合约地址 |
| TempAddress | address | 临时私钥对应的地址 |
| workAddress | address | 工作私钥对应的地址 |
| period | []byte | 阶段描述信息 |

## 更改临时私钥

功能完备性考虑，出现异常需要更改某一阶段的临时私钥。调用layer2链的系统合约设置工作私钥和临时私钥之间的绑定关系。由工作私钥签名交易。

| 名称 | 类型 | 说明 |
| --- | --- | --- |
| gameContractAddress | address | 游戏合约地址 |
| TempAddress | address | 临时私钥对应的地址 |
| workAddress | address | 工作私钥对应的地址 |
| period | []byte | 阶段描述信息 |

## 游戏过程中签名

游戏过程中签名由临时私钥代替工作私钥签名，此过程直接在后台运行，不需要频繁弹出窗口，确认交易信息。

| 名称 | 类型 | 说明 |
| --- | --- | --- |
| gameContractAddress | address | 游戏合约地址 |
| period | []byte | 阶段描述信息 |
| callData | []byte | 合约调用的参数信息 |
| signedData | []byte | 临时私钥签名后的消息 |

1. 从签名信息中解析出来签名的临时私钥，根据gameContractAddress和period，获取 workAddress。
2. 进入 vm 之前将游戏发行方授信给游戏玩家的剩余费用加给 workAddress，进入 vm 执行合约逻辑，vm 执行结束，将 workAddress 的费用恢复到进入 vm 之前，获取实际消耗 gas，将游戏发行方的 账户减少 gas 花费。

## 删除临时私钥

某一阶段游戏结束之后，需要销毁临时私钥，销毁之后临时私钥不可再代理工作私钥签名。

| 名称 | 类型 | 说明 |
| --- | --- | --- |
| gameContractAddress | address | 游戏合约地址 |
| TempAddress | address | 临时私钥对应的地址 |
| workAddress | address | 工作私钥对应的地址 |
| period | []byte | 阶段描述信息 |

## 问题和平衡

**问题一**；简化操作需要，用户的临时私钥处理在 layer2 处理，解决在 layer1 和 layer2 之间切换网络。

要解决这个问题需要用户在 layer1 上进行游戏资产兑换到 layer2 上时，需要同时兑换一定的原生代币到 layer2.

**问题二**：游戏gas费代付，游戏玩家可能进行频繁重复操作，耗尽游戏发行方费用。

游戏发行方，对游戏玩家也就是 workAddress 有一定授信额度，此额度可以设置为无限也就是持续代付，也可以设置为有限值，也就是有限额度代付。

**问题三**：避免游戏临时私钥权限过大，造成游戏资产不必要的损失。

提出抽象的阶段概念，游戏合约更具自己业务需要是否要在合约方法内进行阶段的判断。

# 游戏合约基类

```
contract GameBase {
    // 获取合约发行方
    function issuer() external view returns(address);

    // 获取授信额度
    function lineOfCredit() external view returns (uint256);
}
```