;; 錢包操作碼定義及工作鏈識別的內聯函數

;; 繁體中文註解版 by Y.C.

;; 所有中文均經過人腦編譯，不存在 AI 製造的奇怪語法，並可取代原有的 op-code.fc 及 workchain.fc 服用。

;; 此頁代碼本身已經包含 op-code.fc 及 workchain.fc，
;; 請確保同一資料夾中沒有包含以上兩個檔案，並確保引用此頁代碼時不重複引用以上兩者。
#include "stdlib_zh.fc"; ;; 引入 FunC 標準庫的中文註解版本

{-
   op-code.fc 的部份

-}

;; 一般常用的操作碼和錯誤碼

```
const op::transfer = 0xf8a7ea5;
;; 轉帳操作碼，用於觸發智能合約中的資金轉移操作。
const op::transfer_notification = 0x7362d09c;
;; 轉帳通知操作碼，當轉帳完成後，發送通知以確認交易完成。
const op::internal_transfer = 0x178d4519;
;; 內部轉帳操作碼，通常用於智能合約內部的資金轉移。
const op::excesses = 0xd53276db;
;; 過剩資金的操作碼，指當有多餘資金需要退回或重新分配時的操作。
const op::burn = 0x595f07bc;
;; 銷毀操作碼，代表銷毀代幣或資金的操作，通常用於減少代幣供應。
const op::burn_notification = 0x7bdd97de;
;; 銷毀通知操作碼，當代幣銷毀完成後發送的通知。
const op::provide_wallet_address = 0x2c76b973;
;; 提供錢包地址操作碼，通常用於將錢包地址提供給某個智能合約或實體。
const op::take_wallet_address = 0xd1735400;
;; 接收錢包地址操作碼，表示從智能合約中獲取錢包地址。
const op::top_up = 0xd372158c;
;; 充值操作碼，將資金充值到指定的賬戶或合約中。
const error::invalid_op = 72;
;; 無效操作錯誤碼，當智能合約接收到無法識別或不支持的操作碼時會返回此錯誤。
const error::wrong_op = 0xffff;
;; 錯誤操作碼，當發送了錯誤的操作碼或操作類型不符合預期時，返回此錯誤。
```

```
        const error::not_owner = 73;
        ;; 非所有者錯誤碼，當操作需要由智能合約的所有者執行而非當前調用者時，返回此錯誤。
        const error::not_valid_wallet = 74;
        ;; 非有效錢包錯誤碼，當指定的錢包地址無效或不符合智能合約的條件時返回此錯誤。
        const error::wrong_workchain = 333;
        ;; 錯誤工作鏈錯誤碼，當發送的操作發生在錯誤的工作鏈上（非預期的工作鏈）時，返回此錯誤。

;; 用於 Jetton Minter 合約的操作碼

        const op::mint = 0x642b7d07;
        ;; 鑄幣操作碼，用於創建新的 Jetton 代幣並將其分配給指定的地址。
        const op::change_admin = 0x6501f354;
        ;; 更改管理員操作碼，允許更改 Jetton Minter 合約的管理員地址。
        const op::claim_admin = 0xfb88e119;
        ;; 要求管理權操作碼，用於將 Jetton Minter 的管理權限轉移給新管理員。
        const op::drop_admin = 0x7431f221;
        ;; 放棄管理權操作碼，允許當前管理員放棄對 Jetton Minter 的管理權限。
        const op::upgrade = 0x2508d66a;
        ;; 升級操作碼，允許對 Jetton Minter 合約進行升級或修改。
        const op::change_metadata_uri = 0xcb862902;
        ;; 更改元數據 URI 操作碼，允許修改 Jetton 代幣的元數據 URI，以反映更新的代幣信息。

;; 用於 Jetton Wallet 合約的操作碼及錯誤碼

        const op::set_status = 0xeed236d3;
        ;; 設置狀態操作碼，用於設定 Jetton Wallet 的狀態，比如是否啟用或鎖定。
        const error::contract_locked = 45;
        ;; 合約已鎖定錯誤碼，當合約處於鎖定狀態並且無法執行某些操作時，返回此錯誤。
        const error::balance_error = 47;
        ;; 餘額錯誤碼，當合約的餘額不足以完成某些操作時，返回此錯誤。
        const error::not_enough_gas = 48;
        ;; 氣費不足錯誤碼，當執行操作所需的 Gas 費用不足時，返回此錯誤。
        const error::invalid_message = 49;
        ;; 無效消息錯誤碼，當智能合約接收到的消息格式錯誤或不符合合約的預期時，返回此錯誤。
        const op::batch_store_addresses = 0x1111;
        ;; 批量存儲地址操作碼，這是一個自定義操作碼，用於將多個地址批量存儲到智能合約中。



{-
   workchain.fc 的部份
-------------------------------------------------------------------------------
---------
-}
```

```
const MY_WORKCHAIN = BASECHAIN;  ;; 定義常數 MY_WORKCHAIN，表示當前智能合約所在的工作鏈。
                                 ;; BASECHAIN 於 stdlib_zh 的預定義，代表 workchain 0。

;; 檢查地址的工作鏈是否來自與當前帳戶相同
int is_same_workchain(slice addr) inline {
    ;; 使用 `parse_std_addr` 函數從傳入的 `addr` 切片中解析出工作鏈 ID 和地址
    (int wc, _) = parse_std_addr(addr);

    ;; 返回解析出的工作鏈 ID `wc` 是否等於 `MY_WORKCHAIN`
    return wc == MY_WORKCHAIN;
}

;; 定義一個內聯的純函數 `check_same_workchain`，用來在合約中進行工作鏈的一致性檢查
() check_same_workchain(slice addr) impure inline {
    ;; 如果傳入的 `addr` 地址的工作鏈與當前合約的工作鏈不同，則拋出錯誤
`error::wrong_workchain`
    throw_unless(error::wrong_workchain, is_same_workchain(addr));
}
```