MOMO

? ??

????????? **MOMO ??????? Excel ????**????? SKU ??????????????????????????? Excel ???

? ????

1. **?? Excel ?**
   - ?????? `3841566a` ?? MOMO ????
2. **SKU ????**
   - ?? `sku_momo.json` ????? MOMO ?????+???? ???? SKU?
   - ????????????
3. **????**
   - ?? `plist.json` ??????????????????
   - ????????? ? ???????????
4. **??????**
   - ??????????? + ???????????
   - ??????????????????
5. **??????**
   - ??????????? `MO+□□+□□□` ?
   - ?????? `sku_transfer_log.json` ?????????????
6. **?????**
   - ????????

```
□□□□_SKU□□_YYYYMMDD_HHMMSS.xls
```

? ????

- **????**?
  - MOMO ???? Excel ???.xls?
- **SKU ???**?
  - `\\nas-lianruey\office\sku\sku_momo.json`
- **????**?
  - `\\nas-lianruey\office\sku\plist.json`
- **??????**?
  - `\\nas-lianruey\office\sku\app\sku_transfer_log.json`

? ???????

????????????

- ??+?????????????
- ?SKU
- ???
- ???
- ?????
- ??
- ????
- ?????

?? ????

1. ??????? MOMO ????? Excel ???
2. ?????? ? ???? ? ?????? SKU?
3. ??????????????????
4. ??????????????
5. ???????????????????

? ????

- ??? **SKU ??????????????????????????**?
- ??????????????????

?? ?????

```
import os
import io
import json
import msoffcrypto
import pyexcel as pe
import requests
import re
from tkinter import Tk, filedialog
from datetime import datetime, timedelta
from collections import defaultdict, OrderedDict
```

```
# □□ tkinter □□□
root = Tk()
root.withdraw()

# □□□□□ Excel □□
file_path = filedialog.askopenfilename(title="□□□□□ XLS □□", filetypes=[("Excel Files",
"*.xls")])
if not file_path:
    print("□ □□□□□□□□□□□□□")
    exit()

# SKU □□□
sku_json_path = r"\\nas-lianruey\office\sku\sku_momo.json"
if not os.path.isfile(sku_json_path):
    print(f"□ □□□ SKU □□□□□□{sku_json_path}")
    exit()

with open(sku_json_path, "r", encoding="utf-8") as f:
    sku_map = json.load(f)

# □□□□
plist_url = r'\\nas-lianruey\office\sku\plist.json'



try:
    with open(plist_url, "r", encoding="utf-8") as f:
        temp_data = json.load(f)
except FileNotFoundError:
    print(f"□ □□□□□□□□□□{plist_url}")
    exit()
except json.JSONDecodeError as e:
    print(f"□ □□□□□□□□□□□{e}")
    exit()

sku_temp_map = {}
for item in temp_data:
    sku = item.get("sku")
    temp = item.get("temp", "").strip() or "□□"
    if sku:
```

```
        sku_temp_map[sku] = temp

# 解密 Excel
decrypted = io.BytesIO()
with open(file_path, "rb") as f:
    office_file = msoffcrypto.OfficeFile(f)
    office_file.load_key(password="3841566a")
    office_file.decrypt(decrypted)
decrypted.seek(0)

records = pe.get_array(file_type="xls", file_content=decrypted.read())
next_date = (datetime.today() + timedelta(days=1)).strftime("%Y%m%d")
header = records[0]

# 確保有必要欄位
try:
    order_index = header.index("訂單編號")
    box_index = header.index("併箱編號")

    # 產生轉單編號欄位
    if "轉單編號" not in header:
        header.append("轉單編號")
        for i in range(1, len(records)):
            order_id = str(records[i][order_index])[:14]
            box_id = str(records[i][box_index]).strip()
            mo_id = f"MO{order_id}{box_id}"
            records[i].append(mo_id)
        header = records[0]

    item_index = header.index("品號")
    single_name_index = header.index("單名編號")
    qty_index = header.index("數量")
    address_index = header.index("收件人地址")
    name_index = header.index("收件人姓名")

except ValueError as e:
    print("□ □□□□□□□", e)
    exit()

# 更新欄位
```

```
header[item_index] = "品號+單名編號"
header += ["新SKU", "新數量", "新進價", "新預交貨日", "溫層", "分錄備註", "新轉單編號"]

new_records = [header]
temp_by_order_id = defaultdict(list)
transfer_index = header.index("轉單編號")
temp_index = header.index("溫層")
remark_index = header.index("分錄備註")

# log 檔案位置
log_file_path = r"\\nas-lianruey\office\sku\app\sku_transfer_log.json"
if os.path.exists(log_file_path):
    with open(log_file_path, "r", encoding="utf-8") as f:
        transfer_log = json.load(f)
else:
    transfer_log = {}

today_key = datetime.today().strftime("%Y%m%d")
today_prefix = f"MO{today_key}"
start_seq = transfer_log.get(today_key, 0) + 1
transfer_seq_counter = start_seq

transfer_id_map = OrderedDict()

for row in records[1:]:
    row = row + [""] * (len(header) - 7 - len(row))

    original_product_id = str(row[item_index]).strip()
    single_name = str(row[single_name_index]).strip()
    combined_product_id = "" if original_product_id == "001" else original_product_id +
single_name
    row[item_index] = combined_product_id

    try:
        original_qty = int(row[qty_index])
    except (ValueError, IndexError):
        original_qty = 1

    mo_id = row[transfer_index]
```

```
    if mo_id not in transfer_id_map:
        transfer_id_map[mo_id] = f"{today_prefix}{transfer_seq_counter:03d}"
        transfer_seq_counter += 1

    if combined_product_id in sku_map:
        for sku in sku_map[combined_product_id]:
            try:
                mapped_qty = int(sku.get("新數量", 1))
            except (ValueError, TypeError):
                mapped_qty = 1
            new_qty = mapped_qty * original_qty
            new_sku = sku.get("新SKU", "")
            temp = sku_temp_map.get(new_sku, "冷凍")
            temp_by_order_id[mo_id].append(temp)

            address = str(row[address_index])
            address_cleaned = re.sub(r"[0-9\s]", "", address)[:3]
            name = str(row[name_index])
            remark = address_cleaned + name

            new_row = row + [new_sku, new_qty, sku.get("新進價", ""), next_date, temp, remark,
transfer_id_map[mo_id]]
            new_records.append(new_row)
    else:
        temp_by_order_id[mo_id].append("冷凍")
        address = str(row[address_index])
        address_cleaned = re.sub(r"[0-9\s]", "", address)[:3]
        name = str(row[name_index])
        remark = address_cleaned + name
        new_records.append(row + ["", "", "", next_date, "冷凍", remark,
transfer_id_map[mo_id]])

# 清除重複備註、統一溫層
seen_transfer_ids = set()
final_records = [new_records[0]]
for row in new_records[1:]:
    mo_id = row[transfer_index]
    final_temp = "常溫" if all(t == "常溫" for t in temp_by_order_id[mo_id]) else "冷凍"
    row[temp_index] = final_temp
```

```
    if mo_id in seen_transfer_ids:
        row[remark_index] = ""
    else:
        seen_transfer_ids.add(mo_id)

    final_records.append(row)

# 儲存最新流水號回 log 檔
transfer_log[today_key] = transfer_seq_counter - 1
with open(log_file_path, "w", encoding="utf-8") as f:
    json.dump(transfer_log, f, ensure_ascii=False, indent=2)

# 儲存輸出 Excel
base_dir = os.path.dirname(file_path)
name_part, _ = os.path.splitext(os.path.basename(file_path))
timestamp = datetime.now().strftime("%Y%m%d_%H%M%S")
new_filename = f"{name_part}_SKU轉換_{timestamp}.xls"
new_path = os.path.join(base_dir, new_filename)

pe.save_as(array=final_records, dest_file_name=new_path)
print(f"□ SKU轉換、預交貨日、溫層與分錄備註處理完成，檔案儲存為: {new_path}")
```

Revision #2
Created 1 August 2025 03:01:14 by Wayne
Updated 1 August 2025 03:40:21 by Wayne