CyberBiz

? ??

????? **???? Excel ????????**??? SKU ??????????????????????????
??? Python ?????????? Excel ????????????

? ????

1. **??????**
   - ??????????????????????????
2. **SKU ???????**
   - ?? `sku_cb.json` ?????????? SKU ?????????
3. **??????**
   - ????????C2C?????????????C2C??????
4. **???????**
   - ????????????????
   - ?????? × ????????
   - ??????????? = ???? - ??????
5. **????**
   - ?? `plist.json` ? SKU ????????????????
   - ????????????????????????????????
6. **????**
   - ????????????????????? `(□)`?
   - ??????????????????????
7. **?? Excel ??**
   - ????????????????

```
□□□□_Sku□□_20250801_143000.xls
```

? ????

- **????**?
  - ???? Excel?.xls / .xlsx?
- **SKU ???**?
  - `\\nas-lianruey\office\sku\sku_cb.json`
- **????**?
  - `\\nas-lianruey\office\sku\plist.json`

? ????

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

- ?SKU
- ???
- ???
- ?????
- ??
- ????
- ????
- ???
- ???

?? ????

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

? ????

- ??? SKU ????????????????????????
- ????????????????????????????

?? ?????

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


root = Tk()
```

```
root.withdraw()

days_to_add = simpledialog.askinteger("預交貨日設定", "請輸入要加幾天（預設為1）", initialvalue=1)
if not days_to_add:
    days_to_add = 1

file_path = filedialog.askopenfilename(
    title="選擇 Excel 檔案",
    filetypes=[("Excel 檔案", "*.xls *.xlsx")]
)
if not file_path:
    print("□ □□□□□□□□□□□□")
    exit()

sku_json_path = r"\\nas-lianruey\office\sku\sku_cb.json"
if not os.path.isfile(sku_json_path):
    print(f"□ □□□ SKU 對照表檔案: {sku_json_path}")
    exit()

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

with open(file_path, "rb") as f:
    file_content = f.read()

file_ext = os.path.splitext(file_path)[1].lower()
file_type = "xlsx" if file_ext == ".xlsx" else "xls"
records = pe.get_array(file_type=file_type, file_content=file_content)

next_date = (datetime.today() + timedelta(days=days_to_add)).strftime("%Y%m%d")

header = records[0]
try:
    order_index = header.index("訂單編號")
    status_index = header.index("訂單狀態")
    item_index = header.index("SKU")
    qty_index = header.index("數量")
    price_index = header.index("商品售價")
    address_index = header.index("收件人地址")
    name_index = header.index("收件人名稱")
```

```
    if "轉單編號" not in header:
        header.append("轉單編號")
        for i in range(1, len(records)):
            records[i].append(str(records[i][order_index]))
    transfer_index = header.index("轉單編號")
except ValueError as e:
    print("□ □□□□□□□□", e)
    exit()

# □□替換付款方式
if "付款方式" in header:
    payment_index = header.index("付款方式")
    for i in range(1, len(records)):
        if str(records[i][payment_index]).strip() == "全家冷凍C2C貨到付款":
            records[i][payment_index] = "全冷C2C貨到付款"

total_amount_col_index = -1
try:
    total_amount_col_index = header.index("總額")
    print("□ □□□□□□□□□□□□")
except ValueError:
    print("□◌ 未找到原始「總額」欄位，將根據「數量」和「商品售價」計算。")

order_total_map = {}
if total_amount_col_index != -1:
    for row in records[1:]:
        order_no = str(row[order_index])
        try:
            total_value = float(row[total_amount_col_index])
            order_total_map[order_no] = total_value
        except (ValueError, TypeError):
            pass

original_total_amount_by_order = defaultdict(float)
unique_order_nos = {str(row[order_index]) for row in records[1:]}

for order_no in unique_order_nos:
    if order_no in order_total_map:
        original_total_amount_by_order[order_no] = order_total_map[order_no]
```

```
    else:
        order_total_calculated = 0
        for row in records[1:]:
            if str(row[order_index]) == order_no:
                try:
                    qty = int(row[qty_index])
                    price = float(row[price_index])
                    order_total_calculated += qty * price
                except (ValueError, TypeError):
                    pass
        original_total_amount_by_order[order_no] = order_total_calculated

total_amount_by_order = defaultdict(float)
for row in records[1:]:
    order_no = str(row[order_index])
    product_id = str(row[item_index])
    try:
        original_qty = int(row[qty_index])
    except:
        original_qty = 0

    if product_id in sku_map:
        for sku in sku_map[product_id]:
            try:
                mapped_qty = int(sku["□□□"])
            except:
                mapped_qty = 0
            final_qty = mapped_qty * original_qty
            try:
                new_price = float(sku["□□□"])
            except ValueError:
                new_price = 0.0
            item_total = final_qty * new_price
            total_amount_by_order[order_no] += item_total
    else:
        try:
            new_price = float(row[price_index])
        except ValueError:
            new_price = 0.0
        item_total = original_qty * new_price
```

```
        total_amount_by_order[order_no] += item_total

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 "冷凍"
    if sku:
        sku_temp_map[sku] = temp

fill_value_columns = [
    "收件人名稱", "收件人電話", "收件人地址", "收貨超商代號",
    "收件人國家", "收件人省份", "收件人城市", "收件人區",
    "購買人名稱", "購買人電話", "購買人地址", "時間", "取消時間",
    "訂單編號", "訂單狀態", "訂單來源", "備註", "收款方", "會員名稱",
    "Email", "會員手機號碼", "會員資料備註", "會員標籤", "會員生日", "會員ID"
]

fill_value_index = {}
for col in fill_value_columns:
    if col in header:
        fill_value_index[col] = header.index(col)

first_row_map = {}
for row in records[1:]:
    order_no = str(row[order_index])
    if order_no not in first_row_map:
        first_row_map[order_no] = {}
```

```
        for col, idx in fill_value_index.items():
            first_row_map[order_no][col] = row[idx]

# 確保備註欄位存在
if "備註" not in header:
    header.append("備註")
    for i in range(1, len(records)):
        records[i].append("")
remark_index = header.index("備註")

# 確認額外資訊欄位，不存在就新增空欄
if "額外資訊" not in header:
    header.append("額外資訊")
    for i in range(1, len(records)):
        records[i].append("")
extra_info_index = header.index("額外資訊")

# 新增欄位
header += ["新SKU", "新數量", "新進價", "新預交貨日", "溫層", "分錄備註", "新總金額", "新折扣"]
header.insert(header.index("新折扣") + 1, "新備註")

new_total_amount_index = header.index("新總金額")
new_discount_index = header.index("新折扣")
new_remark_index = header.index("新備註")
temp_col_index = header.index("溫層")
remark_col_index = header.index("分錄備註")

new_records = [header]
temp_by_order = defaultdict(list)

for row in records[1:]:
    row = row + ["" for _ in range(len(header) - len(records[0]))]

    if not row[item_index] or str(row[item_index]).strip() == "":
        row[item_index] = "A"

    order_no = str(row[order_index])
    for col, idx in fill_value_index.items():
        if not row[idx] or str(row[idx]).strip() == "":
            row[idx] = first_row_map[order_no][col]
```

```
    product_id = str(row[item_index])
    try:
        original_qty = int(row[qty_index])
    except:
        original_qty = 0

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

    buyer_name = str(row[fill_value_index.get("購買人名稱", "")] if "購買人名稱" in fill_value_index
else "")
    receiver_name = str(row[name_index])
    current_remark = str(row[remark_index]) if row[remark_index] else ""

    if buyer_name and receiver_name and buyer_name != receiver_name:
        row[remark_index] = current_remark + f"({buyer_name}寄)"
    else:
        row[remark_index] = current_remark

    original_remark = str(row[remark_index]) if row[remark_index] else ""
    extra_info = str(row[extra_info_index]) if row[extra_info_index] else ""
    new_remark = original_remark + extra_info
    new_remark = new_remark.replace("如有送禮需求請幫忙註明送禮", "").strip()

    if product_id in sku_map:
        for sku in sku_map[product_id]:
            try:
                mapped_qty = int(sku["新數量"])
            except:
                mapped_qty = 0
            final_qty = mapped_qty * original_qty
            new_sku = sku["新SKU"]
            try:
                new_price = float(sku["新進價"])
            except ValueError:
                new_price = 0.0
```

```
            temp = sku_temp_map.get(new_sku, "□□")
            temp_by_order[transfer_no].append(temp)
            new_row_data = [new_sku, str(final_qty), new_price, next_date, temp, split_remark,
"", "", new_remark]
            new_row = row[:len(records[0])] + new_row_data
            new_records.append(new_row)
    else:
        new_sku = row[item_index]
        try:
            new_price = float(row[price_index])
        except ValueError:
            new_price = 0.0
        temp = sku_temp_map.get(new_sku, "□□")
        temp_by_order[transfer_no].append(temp)
        new_row_data = [new_sku, str(original_qty), new_price, next_date, temp, split_remark,
"", "", new_remark]
        new_row = row[:len(records[0])] + new_row_data
        new_records.append(new_row)

final_records = [new_records[0]]
seen_transfers = set()

for row in new_records[1:]:
    transfer_no = row[transfer_index]
    order_no = str(row[order_index])
    new_total = total_amount_by_order.get(order_no, 0)
    original_total = original_total_amount_by_order.get(order_no, 0)

    row[new_total_amount_index] = new_total
    row[new_discount_index] = original_total - new_total

    print(f"Order: {order_no}, Original Total (□□): {original_total}, New Total (□□□□):
{new_total})")
    print(f"Calculated New Discount (□□□): {original_total - new_total})")

    temps = temp_by_order[transfer_no]
    row[temp_col_index] = "□□" if all(t == "□□" for t in temps) else "□□"
    if transfer_no in seen_transfers:
        row[remark_col_index] = ""
    else:
```

```
        seen_transfers.add(transfer_no)
    final_records.append(row)

base_dir = os.path.dirname(file_path)
name_part, ext_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、分錄備註、總金額與新折扣加總完成，檔案儲存為: {new_path}")
```

Revision #4
Created 1 August 2025 02:49:27 by Wayne
Updated 1 August 2025 02:56:28 by Wayne