LineBot 交接文件 Puzzle 功能

注意，此份 Code 有部分使用其他功能（題目等等）部分，會在該功能時解釋，此處就不重複陳述。

詳細解釋以註解及後續視訊影片解釋。

註：可以用前後有#TODO、#TODO END 的註解來辨識 獨立於 Puzzle 的 Code。

以 requirement.txt 文件安裝整合版本所需套件環境。

先設定 Channel Access Token 和 Channel Secret

```
#Channel Access Token
line_bot_api = LineBotApi('請填LineBotApi')
#Channel Secret
handler = WebhookHandler('請填WebhookHandler')
```

初始抓資料＆資料處理

```
##解謎   初始抓資料＆資料處理
GDriveJSON = 'JSON.json'
GSpreadSheet_P = 'cilab_ChatBot_puzzle'
gc_Q= pygsheets.authorize(service_account_file='JSON.json')
survey_url_P = 'https://docs.google.com/spreadsheets/d/1nVIgWGQJRIQtMtZSv1HxyDb5FvthBNc0duN4Rlra8to/edit#gid=1732714016'
sh_P = gc_Q.open(GSpreadSheet_P)
sh_P.worksheet_by_title('d0').export(filename='d0')
sh_P.worksheet_by_title('r0').export(filename='r0')
sheet_d0 = pd.read_csv('d0.csv') #type: <class 'pandas.core.frame.DataFrame'>
sheet_r0 = pd.read_csv('r0.csv')
sh_P.worksheet_by_title('d1').export(filename='d1')
sh_P.worksheet_by_title('r1').export(filename='r1')
sheet_d1 = pd.read_csv('d1.csv')
sheet_r1 = pd.read_csv('r1.csv')
sh_P.worksheet_by_title('d2').export(filename='d2')
sh_P.worksheet_by_title('r2').export(filename='r2')
sheet_d2 = pd.read_csv('d2.csv')
sheet_r2 = pd.read_csv('r2.csv')
sh_P.worksheet_by_title('d3').export(filename='d3')
sh_P.worksheet_by_title('r3').export(filename='r3')
sheet_d3 = pd.read_csv('d3.csv')
sheet_r3 = pd.read_csv('r3.csv')
```

#根據階級設定對應資料表單

```python
##-----------------------------------------------------
#根據階級設定對應資料表單
def getSheet_P(level):
    if(level == 3):
        print("level == 3")
        sheet_d = sheet_d3
        sheet_r = sheet_r3
    elif(level == 2):
        print("level == 2")
        sheet_d = sheet_d2
        sheet_r = sheet_r2
    else:
        print("level == 1")
        sheet_d = sheet_d1
        sheet_r = sheet_r1

    return sheet_d, sheet_r
```

#使用者變數

```python
class userVar():
    def __init__(self,_id):
        self._id = _id
        #QA
        self.level_Q = 1 # 預設level 1
        self.data_Voc, self.data_Reading, self.data_Cloze = getSheetQA(self.level_Q) #預設傳level = 1
        self.isVoc = False
        self.VocQA = []
        #Listen
        self.level_L = 1 # 預設level 1
        self.data_pho, self.data_word, self.data_sen = getSheet(self.level_L)
        self.isWord = False
        self.word_list = []
        #speech
        self.L1_qa = []
        self.L2_qa = []
        self.L3_qa = []
        self.stt_mes = ''
        self.QA_ = []

#TODO ------------------------------------
        #puzzle
        self.name = '???'#預設名字
        self.next_id = 0
        self.level_P = 1
        self.index_P = 0 #第幾題
        self.isInit_P = False #是否初始遊戲
        self.isChangingLevel_P = False #是否觸發設定階級功能
        self.isChooseHelp = False #是否觸發選單功能
        self.isLoad_P = False #是否觸發載入階級表單
        self.isPreStory_P = False #是否正在題目前故事劇情中
        self.isStart_P = False #是否開始出題
        self.isAsked_P = False #是否正在題前故事中 或 答題中
```

#重置使用者變數

```
#puzzle
self.name = '???'#預設名字
self.next_id = 0
self.level_P = 1
self.index_P = 0 #第幾題
self.isInit_P = False #是否初始遊戲
self.isChangingLevel_P = False #是否觸發設定階級功能
self.isChooseHelp = False #是否觸發選單功能
self.isLoad_P = False #是否觸發載入階級表單
self.isPreStory_P = False #是否正在題目前故事劇情中
self.isStart_P = False #是否開始出題
self.isAsked_P = False #是否正在題前故事中 或 答題中
self.levelsheet_d = sheet_d0 #階級表單 初始設為d0 r0
self.levelsheet_r = sheet_r0
self.text_sheet_P = self.data_Cloze #題目表單 預設為克漏字表單
self.test_type_list = np.zeros(10) #隨機七種類題目 共十題
self.subindex_P = 0 #子題號 (隨機取表單之題號)
self.count_P = 2 #答題次數
self.star_num_P = 0 #計分
self.count_type_P = 2 #可答題次數 (因應選項可能為兩題或三題)
self.isPuzzle_P = True  #判斷語音辨識中是目前是Puzzle還是Speech功能使用到
#speech變數
self.sheet_word_s = []
self.sheet_sen_s = []
self.L1_sen_s = []
self.L2_sen_s = []
self.L3_sen_s = []
```

```
# 監聽所有來自 /callback 的 Post Request
@app.route("/callback", methods=['POST'])
def callback():
    # get X-Line-Signature header value
    signature = request.headers['X-Line-Signature']
    # get request body as text
    body = request.get_data(as_text=True)
    app.logger.info("Request body: " + body)
    # handle webhook body
    try:
        handler.handle(body, signature)
    except InvalidSignatureError:
        abort(400)
    return 'OK'
```

#處理訊息

```python
#處理訊息
@handler.add(MessageEvent, message=TextMessage)
def handle_message(event):
    user = getUser(event.source.user_id)
#TODO ----------------------------------------
    if event.message.text =='#puzzle':
        reset(user) #初始遊戲變數
        user.isInit_P = True
    if(user.isInit_P == True):
        user.isInit_P = False
        smallpuzzle(event,'d00000',sheet_d0, user)
    if user.next_id == 'd00002':
        if event.message.type == 'text':
            user.name = event.message.text #設定user name
            print(event.message.text)
            print(user.name)
        smallpuzzle(event, user.next_id , user.levelsheet_d, user)
#TODO END----------------------------------------
```

#將使用者資訊更新至資料庫

```python
#將使用者資訊更新至資料庫
def getUser(user_ID):
    global allUser
    user = next((item for item in allUser if item._id == user_ID), None)
    if user is None:
        user = userVar(user_ID)
        allUser.append(user)
        print("Alluser",allUser)
    return user
```

#Postback Event 判斷

-

LineBot 交接文件 Puzzle 功能

```python
#Postback Event判斷
@handler.add(PostbackEvent)
def handle_postback(event):
    user = getUser(event.source.user_id)
#TODO ------------------------------------
    pb_event = event.postback.data
    print("postbackData = ",pb_event )
    if (pb_event == 'Next'):
        #載入題號與敘述
        if user.isLoad_P == True:
            user.isLoad_P = False
            message = LoadTestIndex(user)
            line_bot_api.reply_message(event.reply_token, message)
            user.isPreStory_P = True
        #載入題目前故事
        elif user.isPreStory_P == True:
            if user.isAsked_P == False :
                user.isAsked_P = True
                test_type = user.test_type_list[user.index_P] #判斷題型
                print("test_type = ", test_type)
                print('--TestPreStory--'+'d'+ str(user.level_P) + str(test_type) + '000')
                smallpuzzle(event, 'd' + str(user.level_P) + str(test_type) + '000', user.levelsheet_d, user)
            else:
                smallpuzzle(event, user.next_id , user.levelsheet_d, user)
        #開始出題
        elif(user.isStart_P == True):
            print("load_Q")
            bubble = Question_P(event, user)
            message = FlexSendMessage(alt_text="bubble", contents = bubble)
            line_bot_api.reply_message(event.reply_token, message)

        else:
            if(user.next_id != 'd00002'): #若非需等待使用者輸入名字
                smallpuzzle(event, user.next_id , user.levelsheet_d, user)

    elif user.isChangingLevel_P == True: #設定階級
        setLevel_P(pb_event, user)
        smallpuzzle(event,'d00202',sheet_d0, user)

    elif user.isChooseHelp == True: #功能選單
        #--Game State-----------------------------------
        user.isChooseHelp = False
        if pb_event == 'f1':
            #了解背景故事
            smallpuzzle(event,'d00100',sheet_d0, user)
        elif pb_event == 'f2':
            #開始遊戲
            smallpuzzle(event,'d00200',sheet_d0, user)
        elif pb_event == 'f3':
            #結束遊戲
            reset(user)
            print("End!")
        else:
            pass

    elif user.isStart_P == True: #判斷答案對錯
        print("---Ans feedback---")
        #取得正確答案
        if user.isVoc == True:
            correctAns = str(user.VocQA[2])
        elif user.isWord == True:
            correctAns = str(user.word_list[2])
        else: #非字彙題型
            correctAns = str(user.text_sheet_P[user.subindex_P][4])
        print("correct answer",correctAns)
        checkAnswer(pb_event, correctAns, user, event)
#TODO END-------------------------------------
```

#判斷答案對錯

```
#TODO ------------------------------------
#判斷答案對錯
def checkAnswer(pb_event, correctAns, user, event):
    if pb_event != correctAns:
        print("answer",pb_event," != correctAns",correctAns)
        if(user.count_P != user.count_type_P - 1): #第一次答錯
            print("Wrong 1")
            user.isStart_P = False
            user.count_P -= 1
            user.next_id = 'd'+ str(user.level_P) + str(user.test_type_list[user.index_P]) + '200'
            smallpuzzle(event, user.next_id, user.levelsheet_d, user)

        elif(user.count_P == user.count_type_P - 1):#第二次答錯
            user.isStart_P = False
            print("Wrong 2")
            user.next_id = 'd'+ str(user.level_P) + str(user.test_type_list[user.index_P]) + '300'
            user.count_P = 2
            smallpuzzle(event, user.next_id, user.levelsheet_d, user)

    else:#答對
        user.isStart_P = False
        user.star_num_P += user.count_P
        print('Correct Answer!')
        user.next_id = 'd'+ str(user.level_P) + str(user.test_type_list[user.index_P]) + '100'
        if(user.count_P == user.count_type_P):
            smallpuzzle(event, user.next_id, user.levelsheet_d, user)

        elif(user.count_P == user.count_type_P - 1):
            smallpuzzle(event, user.next_id, user.levelsheet_d, user)

        user.count_P = 2
    print('after count_P: ', user.count_P)
```

#設定階級與取得相對應題目表單

```
def setLevel_P(levelinput, user):
    print("---Changing Level---")
    #global user.level_P, user.isChangingLevel_P

    if (levelinput=='L'):
        user.level_P = 1
        user.isChangingLevel_P = False

    elif (levelinput=='M'):
        user.level_P = 2
        user.isChangingLevel_P = False

    elif (levelinput=='H'):
        user.level_P = 3
        user.isChangingLevel_P = False

    else:
        user.isChangingLevel_P = True

    user.data_pho, user.data_word, user.data_sen = getSheet(user.level_P)
    user.data_Voc, user.data_Reading, user.data_Cloze = getSheetQA(user.level_P) #預設傳level = 1
    getSheet_S(user.level_P, user)
```

#表單 id、種類、內容處理

```python
#表單id、種類、內容處理
def smallpuzzle(event,id, sheet, user):
    print("----------id----------",id)
    #檢查表單中是否有此id
    id_index = sheet["a-descriptionID"].index[sheet["a-descriptionID"] == id]

    if len(id_index) > 0: #有此id
        id_index = id_index[0]
        #print("id_index",id_index)

        user.next_id = id[0:3]+ str( int(id[3:6]) + 1).zfill(3) #下一號
        #print("next id = ", user.next_id)

        sheet_type = sheet["type"][id_index] #id種類
        #print("sheet_type",sheet_type)

        #依id種類對應訊息格式
        if sheet_type == 'image':
            sheet_text = sheet["text"][id_index]
            message = ImageBubble(sheet_text)
            line_bot_api.reply_message(event.reply_token, message)

        elif sheet_type == 'text':
            sheet_text = sheet["text"][id_index]
            if '$username' in sheet_text:   # 使用in運算子檢查
                sheet_text = sheet_text.replace('$username', user.name) #判斷字串中有'$username 取代為user.name
            message = TextBubble(sheet_text)
            line_bot_api.reply_message(event.reply_token, message)

        elif sheet_type == 'button':
            if id == 'd00003':
                user.isChooseHelp = True
            if id == 'd00201':
                user.isChangingLevel_P = True
            sheet_title = sheet["title"][id_index]
            sheet_text = sheet["text"][id_index]
            sheet_reply_list = []
            #抓取表單選項資料
            for i in range (3):
                if (str(sheet.iloc[id_index][4 + i]) != "") :
                    sheet_reply_list.append((str(sheet.iloc[id_index][4 + i])))
            replylist = ButtonPuzzle(sheet_reply_list)
            button_bubble = ButtonBubble(sheet_title, sheet_text, replylist)
            line_bot_api.reply_message(event.reply_token, button_bubble)

        elif sheet_type == 'confirm':
            sheet_text = sheet["text"][id_index]
            sheet_reply_list = []
            #抓取表單選項資料
            for i in range (2):
                if (str(sheet.iloc[id_index][4 + i]) != "") :
                    sheet_reply_list.append((str(sheet.iloc[id_index][4 + i])))
            replylist = CofirmPuzzle(sheet_reply_list, user)
            confirm_bubble = ConfirmBubble(sheet_text, replylist)
            line_bot_api.reply_message(event.reply_token, confirm_bubble)

        elif sheet_type == 'input':
            sheet_text = sheet["text"][id_index]
            line_bot_api.reply_message(event.reply_token,TextSendMessage(text = sheet_text))
```

#id 不存在 判斷

```
else: #id不存在
    print("Do Not Find ID in Sheet! ")
    if id =='d00102': #重複詢問可以幫您什麼？
        smallpuzzle(event,'d00003',sheet_d0, user)

    elif id =='d00208': #設定階級
        user.levelsheet_d, user.levelsheet_r = getSheet_P(user.level_P) #取得階級表單
        print("level = ",user.level_P)
        setLevelStory(event, user) #開始階級表單

    #-----------------------------------------------

    #剛開始答題
    elif id == 'd10030' or id == 'd20025' or id == 'd30022':
        RandomTest(user) #取得隨機十題型
        user.isLoad_P = True #載入題號
    elif (int(id[1:2]) == (user.level_P)):# 判斷第一碼是否為d1/d2/d3表單
        if(int(id[2:3]) == (user.test_type_list[user.index_P])):  #判斷第二碼為題型碼
            #答對
            if id[3:4] == '1':
                if  user.index_P < 9:
                    print("答對 繼續isLoad_P")
                    user.index_P += 1
                    user.isAsked_P = False
                    user.isLoad_P = True
                else:
                    smallpuzzle(event,'d'+ str(user.level_P) + '0100', user.levelsheet_d, user)
            #第一次答錯
            elif id[3:4] == '2':
                if user.index_P < 9:
                    print("第一次答錯 再一次 isStart_P，Load題目")
                    user.isStart_P = True
                else:
                    smallpuzzle(event,'d'+ str(user.level_P) + '0100', user.levelsheet_d, user)
            #第二次答錯
            elif id[3:4] == '3':
                if user.index_P < 9:
                    user.index_P += 1
                    user.isAsked_P = False
                    user.isLoad_P = True
                    print("第二次答錯 新題目PreStory")
                else:
                    smallpuzzle(event,'d'+ str(user.level_P) + '0100', user.levelsheet_d, user)
```

```
    #-----------------------------------------------
    #----計算最後答題結果
    #是否大於六題
    elif id[2:4] == '01':
        if user.star_num_P >= 6:
            smallpuzzle(event,'d'+ str(user.level_P) + '0200', user.levelsheet_d, user)
        else:
            smallpuzzle(event,'d'+ str(user.level_P) + '0300', user.levelsheet_d, user)
    #結尾故事
    elif id[2:4] == '02' or id[2:4] == '03':
        smallpuzzle(event,'d'+ str(user.level_P) + '0400', user.levelsheet_d, user)

    #結束 回到最初功能選擇
    elif id[2:4] == '04':
        reset(user)
        smallpuzzle(event,'d00003',sheet_d0, user)

    if user.isPreStory_P == True: #題目前故事結束
        print("PreStory End! Strat Testing!")
        user.isStart_P = True #開始出題
        user.isAsked_P = False
        user.isPreStory_P = False

pass
```

#依據格式取得表單內容

```
#取得表單內容
def ButtonPuzzle(sheet_reply_list):
    replylist = []
    print("ButtonPuzzle",sheet_reply_list)
    for i in range(len(sheet_reply_list)):
        id_index = sheet_r0["a-replyID"].index[sheet_r0["a-replyID"] == sheet_reply_list[i]]
        replylist.append(([sheet_r0["label"][id_index[0]], sheet_r0["text"][id_index[0]], sheet_r0["data"][id_index[0]]]))
    print("replylist",replylist)
    return replylist

#取得表單內容
def CofirmPuzzle(sheet_reply_list, user):
    print("CofirmBubble",sheet_reply_list)
    replylist = []
    for i in range(len(sheet_reply_list)):
        id_index = user.levelsheet_r["a-replyID"].index[user.levelsheet_r["a-replyID"] == sheet_reply_list[i]]
        replylist.append(([user.levelsheet_r["label"][id_index[0]], user.levelsheet_r["text"][id_index[0]], user.levelsheet_r["data"][id_index[0]]]))
    print("--Cofirm replylist",replylist)
    return replylist
```

#載入階級表單

```
606    #載入階級表單
607    def setLevelStory(event, user):
608        print("setLevelStory")
609
610        if user.level_P == 1:
611            smallpuzzle(event,'d10000' , user.levelsheet_d, user)
612
613        elif user.level_P == 2:
614            smallpuzzle(event,'d20000' , user.levelsheet_d, user)
615
616        elif user.level_P == 3:
617            smallpuzzle(event,'d30000' , user.levelsheet_d, user)
```

#取隨機七種題型十題

```
619    #取隨機七種題型十題
620    def RandomTest(user):
621        user.test_type_list = [random.randint(1,7) for _ in range(10)]
622        print("-----*** Quiz type = ",user.test_type_list)
623
```

#載入題號與敘述

```
625    def LoadTestIndex(user):
626        print("-----LoadTestIndex----", user.index_P)
627        #題數引文
628        if user.level_P == 1 :
629            test_pretext = "（第" + str(user.index_P+1) + "題）\n【Silas】：\n勇者" + user.name + "，現在是 "+ str(8+user.index_P) +":00，Ariel 希望我們在傍晚18:00前完成。"
630            print(test_pretext)
631            message = TextBubble(test_pretext)
632
633        elif user.level_P == 2:
634            test_pretext = "（第" + str(user.index_P+1) + "題）\n【Keith】：\n勇者" + user.name + "，現在是 "+ str(8+user.index_P) +":00，Faun 希望我們在傍晚18:00前完成。"
635            print(test_pretext)
636            message = TextBubble(test_pretext)
637
638        elif user.level_P == 3:
639            test_pretext = "（第" + str(user.index_P+1) + "題）\n【Cynthia】：\n真是太好了！剛好每天晚上Helena都會在他的閣樓唱歌給大家聽，我們趕緊去找，18:00拿去給領主吧！\n勇者，Let's go！"
640            print(test_pretext)
641            message = TextBubble(test_pretext)
642        return message
```

```python
#出題目
def Question_P(event, user):
    user.isVoc = False
    user.isWord = False
    user.count_type_P = 2

    if user.test_type_list[user.index_P] == 1:
        print("sheet_L_pho & word")
        test_type1 = random.randint(1, 2)
        if test_type1 == 1: #題型 聽力pho
            print("--sheet_pho--")
            if user.level_P != 3:
                user.count_type_P = 1

                if user.count_P == 2:
                    user.count_P = 1

                if user.count_P == user.count_type_P and user.isAsked_P == False : #是否為第一次答此題 隨機取題，若否 subindex_P維持
                    print("random QA_Tail subindex")
                    user.isAsked_P = True
                    user.text_sheet_P = user.data_pho
                    user.subindex_P = random.randrange(1,len(np.transpose([user.text_sheet_P])[0]))
                user.text_sheet_P = user.data_pho
                bubble = QA.QA_Tail(user.text_sheet_P,user.index_P,user.subindex_P)
            else: #聽力pho高級 題目不同 音檔選句子
                print("---level 3 pho  依據音檔選句子---")
                if user.count_P == user.count_type_P and user.isAsked_P == False:
                    user.isAsked_P = True
                    user.text_sheet_P = user.data_pho
                    user.subindex_P = random.randrange(1,len(np.transpose([user.text_sheet_P])[0]))
                bubble = QA.QA_Sentence(user.text_sheet_P,user.index_P,user.subindex_P,'依據音檔，選出最適當的答案')
        else: #題型 聽力單字題
            print("--sheet_word--",test_type1)
            user.isWord = True
            if user.count_P == user.count_type_P and user.isAsked_P == False:
                user.isAsked_P = True
                user.text_sheet_P = getVoc.editSheet(user.data_word) #編輯單字表單
                q_index, q_chinese, q_english = getVoc.getVoc(user.text_sheet_P) #取得題目
                option_english,option_english2 = getVoc.getOption(user.text_sheet_P, q_index) #取得選項
                option, answer = getVoc.getQA(q_english, option_english,option_english2) #編輯選項答案成list
                q_audio = getVoc.getAudio(user.text_sheet_P, q_index) #取得音檔
                user.word_list = [q_audio, option, answer]
            print("user.word_list",user.word_list)
            bubble = QA.QA_Word(user.index_P, user.word_list)

    elif user.test_type_list[user.index_P] == 2:#題型 聽力句子
        print("sheet_L_sen")
        user.text_sheet_P = user.data_sen
        if user.count_P == user.count_type_P and user.isAsked_P == False:
            user.isAsked_P = True
            print("random subindex_P")
            user.subindex_P = random.randrange(1,len(np.transpose([user.text_sheet_P])[0]))
        print("user.subindex_P",user.subindex_P)
        bubble = QA.QA_Sentence(user.text_sheet_P,user.index_P,user.subindex_P,'選出正確的聽對句子')

    elif user.test_type_list[user.index_P] == 3:#題型 口說單字
        print("sheet_speaking_word")
        bubble = QA_S(user.sheet_word_s[user.index_P][0], user.sheet_word_s[user.index_P][1], user, user.index_P)

    elif user.test_type_list[user.index_P] == 4:#題型 口說句子
        print("sheet_speaking_sen")
        bubble = QA_S(user.sheet_sen_s[user.index_P][0], user.sheet_sen_s[user.index_P][1], user, user.index_P)

    elif user.test_type_list[user.index_P] == 5:#題型 出題單字
        print("sheet_Q_voc")
        user.isVoc = True
        if user.count_P == user.count_type_P and user.isAsked_P == False:
            user.isAsked_P = True
            user.text_sheet_P = getVoc.editSheet(user.data_Voc)#編輯單字表單
            q_index, q_chinese, q_english = getVoc.getVoc(user.text_sheet_P)
            option_english,option_english2 = getVoc.getOption(user.data_Voc, q_index)
            option, answer = getVoc.getQA(q_english, option_english,option_english2)
            user.VocQA = [q_chinese, option, answer]
        print(user.VocQA)
        bubble = QA_Bubble.Voc(user.index_P, user.VocQA)

    elif user.test_type_list[user.index_P] == 6:#題型 出題克漏字
        print("sheet_Q_cloze")
        user.text_sheet_P = user.data_Cloze
        if user.count_P == user.count_type_P and user.isAsked_P == False:
            user.isAsked_P = True
            user.subindex_P = random.randrange(1,len(np.transpose([user.text_sheet_P])[0]))
            print("data_Cloze subindex_P", user.subindex_P)
        if (user.level_P != 3):
            bubble = QA_Bubble.Cloze(user.text_sheet_P, user.index_P, user.subindex_P)
        else: #高級 格式不同
            bubble = QA_Bubble.Cloze_L3(user.text_sheet_P, user.index_P, user.subindex_P)

    elif user.test_type_list[user.index_P] == 7 :#題型 出題閱測
        print("sheet_Q_reading")
        if(user.count_P == user.count_type_P and user.isAsked_P == False):
            user.isAsked_P = True
            user.text_sheet_P = user.data_Reading
            print("reading", len( np.transpose( [user.text_sheet_P])[0] ) )
            user.subindex_P = random.randrange(1, len(np.transpose([user.text_sheet_P])[0]), 3)
            QA_bubble_article = QA_Bubble.Article( user.text_sheet_P, user.subindex_P )
            article = FlexSendMessage(alt_text="QA_bubble", contents = QA_bubble_article)
            line_bot_api.push_message(event.source.user_id, article)

        bubble = QA_Bubble.Reading(user.data_Reading, user.index_P, user.subindex_P)
```

#Bubble 格式處理

```
def ButtonBubble(sheet_title, sheet_text, replylist)
def TextBubble(sheet_text)
def ImageBubble(sheet_text)
def ConfirmBubble(sheet_text, replylist)
def QA_S(address, ques, user, index)
```