Отправка через ExtSDK2

Подготовительные действия

Подготовка к отправке запросов включает три шага:

- подключение к Плагину
- подключение к ExtSDK2
- аутентификация (там поправьте адрес хоста (fix, test) и логин/пароль на свои значения)

Почитать можно на хелпе.

Код в данном файле поправлен относительно хелпа: там зацикливания в циклах, лучше скопируйте отсюда.

```
import win32com.client
import json
import uuid


"""Шаг 1. Подключение к СБИС3.Плагин"""


ole = win32com.client.Dispatch("Tensor.SbisPluginClientCOM")  # Получаем COM объект

# При получении COM объекта плагин генерирует событие типа 'Event' с
eventName=connected,
# поэтому если среди полученных ответов от ReadAllObject() есть событие стаким
eventName, то вам удалось подключиться
# к плагину, иначе стоит сделать таймаут 100 - 500мс и повторить вызов
ReadAllObject().Пример цикла.

while True:
    events = json.loads(ole.ReadAllObject())  # получаем события от плагина и
преобразуем в массив json объектов
    # в цикле проходим по каждому событию и ищем нужное с eventName=connected
    for event in events:
        if event['type'] == 'Event' and event['data']['eventName'] == 'connected':
            print("Подключение к Плагину выполнено")
            # если нужное событие пришло выходим из цикла
            break
    else:
    # если ожидаемое событие не найдено, то делаем таймаут в 300мс
        ole.Sleep(300)
        continue
    break
# После подключения к плагину вы можете подключаться к ExtSdk2 и звать методы
данного модуля


"""Шаг 2. Подключение к ExtSDK2"""


# Получаем uuid модуля ExtSdk2 через ole.GetModule. Если uuid получен, значит
подключение к ExtSdk2 успешно и можно совершать запросы в модуль.
guid_module = ole.GetModule("ExtSdk2")  # Получаем uuid модуля ExtSdk2
print("Подключение к ExtSDK2 выполнено, uuid: ", guid_module)


"""Шаг 3. Аутентификация по логину и паролю"""


query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по которому
будем ожидать ответ
```

```
module_method = "ExtSdk2.AuthByPassword"  # метод, который хотим позвать
parameters_module_method = json.dumps({"Login": "введи_логин", "Password":
"введи_пароль"},
                                      ensure_ascii=True)  # формируем параметры
метода ExtSdk2.AuthByPassword
host = "test-online.sbis.ru"  # хост, на который хотим пойти online.sbis.ru.

# Вызов метода аутентификации без авторизаии, чтобы получить идентификатор сессии
ole.CallMethodWithoutAuth(query_id, guid_module, module_method,
parameters_module_method, host)

# Ожидаем ответ через ReadAllObject в цикле по query_id
session_id = ""  # Переменная для сохранения идентификатора сессии для вызов
методов ExtSdk2 с пройденной аутентификацией
while True:
    events = json.loads(ole.ReadAllObject())  # получаем события от плагина и
преобразуем в массив json объектов
    # в цикле проходим по каждому событию и ищем нужное с ожидаемым query_id
    for event in events:
        if event['type'] == 'Message' and event['queryID'] == query_id:
            # если нужное событие пришло, то записываем идентификатор сессии в
переменную
            session_id = event['data']['Result']
            break
    else:
        # если ожидаемое событие не найдено, то делаем таймаут в 300мс
        ole.Sleep(300)
        continue
    break
print("Получен идентификатор сессии: ", session_id)
```

Вспомогательная информация

Как проверять, что все ушло

Надо смотреть логи Плагина с дебагом.

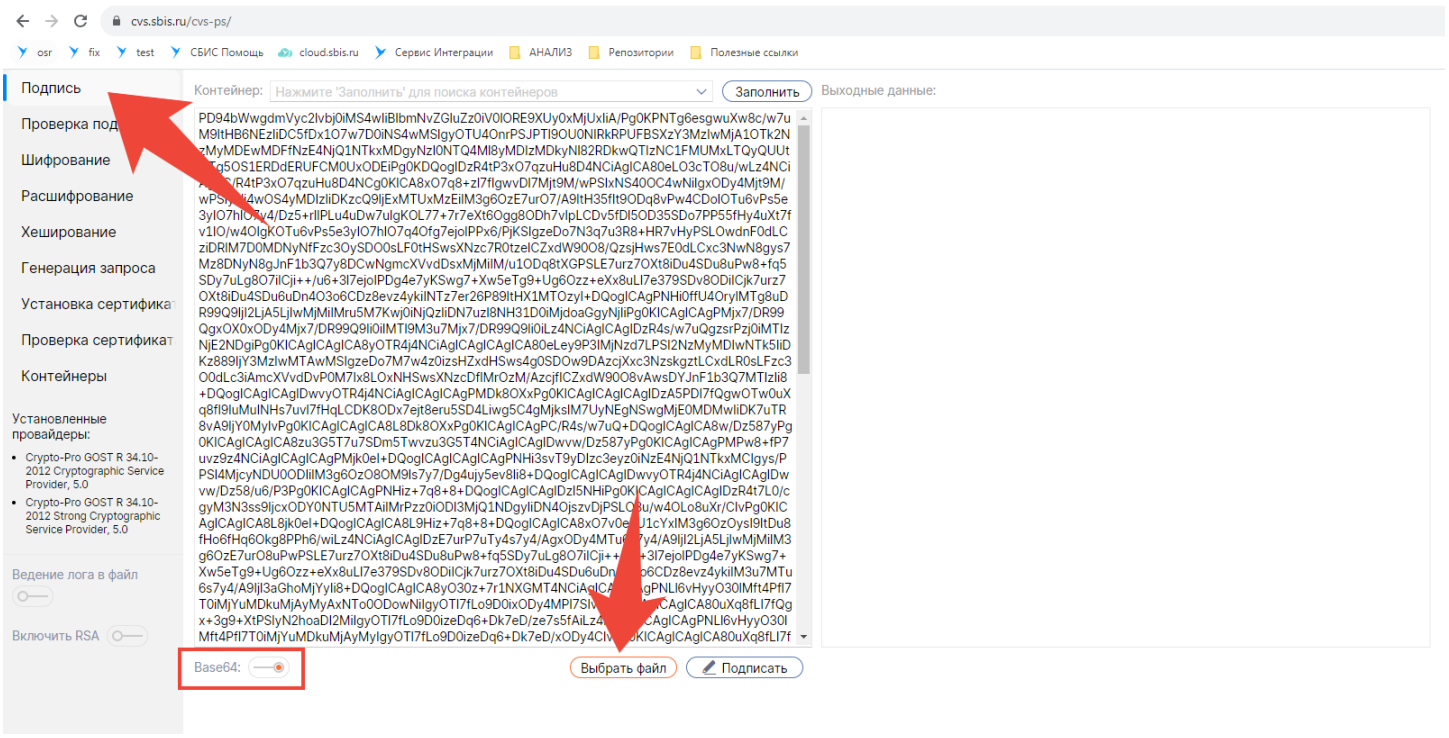
1. Перейти к расположению Плагина.
2. В адресной строке проводника вбить cmd вместо C:\Program Files (x86)\Tensor Company Ltd\SBIS3Plugin, откроется командная строка в директории с Плагином
3. Вбить туда sbis3plugin.exe --debug
4. Перейти в %localappdata%, там папка Sbis3Plugin, в ней папка logs, в ней папка с сегодняшней датой, смотреть можно, например в ExtSdk2-client.log.

Про получение данных xml и подписи в виде строки BASE64

Я моделировала ситуацию клиента, а он отправлял документ и подпись в виде строки BASE64. Способов перевести данные в такой формат много. Привожу самый простой.

Воспользуйтесь https://cvs.sbis.ru/.

Выбираете "Подпись", подгружаете файл подписываемого xml-документа, подписываете отсоединенной подписью. Результат — и файл, и подпись представлены в виде строки BASE64.



Отправка запросов

Метод ExtSDK2.WriteDocument

Отправляла документ и подпись в кодировке Base64.

Сам документ должен содержать сведения о подписанте, совпадающие с данными подписи.

Пример:

```
<Подписант ОблПолн="5" Статус="1" ОснПолн="Должностные обязанности">
      <ЮЛ ИННЮЛ="6732020599" НаимОрг="ОБЩЕСТВО С ОГРАНИЧЕННОЙ ОТВЕТСТВЕННОСТЬЮ
&amp;quot;ПРОИЗВОДСТВЕННАЯ КОМПАНИЯ &amp;quot;ЛАВАШ&amp;quot;123" Должн="">
        <ФИО Фамилия="Бельский" Имя="Александр" Отчество="Николаевич"/>
      </ЮЛ>
    </Подписант>
```

Пример сформированного запроса и вызова метода:

```
"""Функция, зовущая метод ExtSDK2.WriteDocument"""


def write_document():
    document_id = "d8e74588-f9e9-4a50-a4f8-82753ab" + str(random.randint(10000,
99999)) #генерирую рандомный ИД документа
    query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по
которому будем ожидать ответ
    module_method = "ExtSdk2.WriteDocument"  # метод, который хотим позвать
    parameters_module_method = json.dumps(
        {
        "Document": {
            "Вложение": [
                {
                "Идентификатор": "8b8d54e2-44ae-4de3-8433-f4a7c86c2c5e",
                "Файл": {
                    "ДвоичныеДанные":
"PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0id2luZG93cy0xMjUxIiA/Pgo81ODp6yDC5fDxz/Du4z0i0c
```

```
                    "Имя":
"ON_NSCHFDOPPR_2BE24e62d2f2ba54942bfe49025c553cc62_2BE8f3178e7ae8c416790175e95e9aed524_
DA6D-4285-A69D-187EDE268D0F.xml"
                },
                    "Подпись": [
                        {
                        "Файл": {
                        "Имя":
"ON_NSCHFDOPPR_2BE24e62d2f2ba54942bfe49025c553cc62_2BE8f3178e7ae8c416790175e95e9aed524_
DA6D-4285-A69D-187EDE268D0F.xml.sgn",
                        "ДвоичныеДанные":
"MIIfjQYJKoZIhvcNAQcCoIIffjCCH3oCAQExDjAMBggqhQMHAQECAgUAMAsGCSqGSIb3DQEHAaCCClowggpWMI

                                }
                        }
                            ]
                }
            ],
            "ДопПоля":"ЭтапВернутьВсеСертификаты",
            "Дата": "27.09.2023",
            "Номер": "OLA27092023",
            "Идентификатор": document_id,
            "Контрагент": {
                "СвЮЛ": {
                "ИНН": "2600303385",
                "КПП": "260001001",
                "Название": "Новый получатель"
                }
            },
            "НашаОрганизация": {
                "СвЮЛ": {
                "ИНН": "6732020599",
                "КПП": "673201001"
                }
            },
            "Примечание": "Тест ExtSDK2 Степанова",
            "Тип": "ДокОтгрИсх"
        }
        }
        )
    # формируем параметры метода ExtSdk2.WriteDocument.

    # Вызов метода ExtSdk2.WriteDocument
    ole.CallMethod(query_id, guid_module, module_method, parameters_module_method,
session_id)

    # Ожидаем ответ через ReadAllObject в цикле по query_id
    result = {}  # Переменная для сохранения результата ExtSdk2.WriteDocument
    result_checked = False

    while not result_checked:

        events = json.loads(ole.ReadAllObject())  # получить события от плагина и
преобразовать в массив json объектов

        # в цикле пройти по каждому событию и найти нужное с ожидаемым query_id
        for event in events:
            if event['type'] == 'Event':
                # если нужное событие пришло, то записать результат в нужную
переменную
                print('Запрос ExtSDK2.WriteDocument ушел, смотри логи Плагина')
                response_writedocument = event['data']['data']['Result']
                result_checked = True
                break

        # если ожидаемое событие не найдено, то сделать таймаут в 300мс
        ole.Sleep(300)
    return response_writedocument
```

Чтобы функция запустилась на выполнение, не забудьте ее позвать:

```
response_writedocument = write_document()
```

Ответ на запрос ляжет в переменную response_writedocument. Если что-то упало, смотрите логи Плагина, оберток на ошибки не писала.

Метод ExtSDK2.ExecuteAction

Здесь в запрос перекладываю ИД документа и вложения из ответа на WriteDocument.

Для вызова используем что-то вроде:

```
response_writedocument = write_document() #сначала зовем ЗаписатьДокумент
execute_action(response_writedocument) #теперь зовем выполнить действие для
документа с ИД, пришедшим в ответе на прошлый запрос
```

Я пропустила ПодготовитьДействие сознательно: в ЗаписатьДокумент у меня уже есть подпись.

Сама функция:

```
"""Функция, зовущая метод ExtSDK2.ExecuteAction"""


def execute_action(response_writedocument):
    query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по
которому будем ожидать ответ
    module_method = "ExtSdk2.ExecuteAction"  # метод, который хотим позвать
    parameters_module_method = json.dumps(
                {
                "Document": {
                    "Идентификатор": response_writedocument['Идентификатор'],
                        "Этап": {
                        "Вложение": [
                            {
                                "Идентификатор": response_writedocument['Вложение']
[0]['Идентификатор'],
                                "Подпись": [
                                    {
                                        "Файл": {
                                            "Имя":
"ON_NSCHFDOPPR_6732020599673201001_7186455910827245482_20230926_6D90A234-E1C1-42AE-
9899-DD7DEAB3E181.xml.sgn",
                                            "ДвоичныеДанные":
"MIIfjQYJKoZIhvcNAQcCoIIffjCCH3oCAQExDjAMBggqhQMHAQECAgUAMAsGCSqGSIb3DQEHAaCCClowggpWMI

                                                },
                                    }
                                ]
                            }
                        ],
                        "Действие": [
                            {
                                "Комментарий": "",
                                "Название": "Отправить",
                                "Сертификат": {
                                    "Отпечаток":
"FEC6BAE4A3069DF3E80049EFD1D1AD2364F51E14"
                                }
                            }
                        ],
                        "Название": "Отправка"
                        }
                            }
```

```
                }
        )

    # Вызов метода ExtSdk2.ExecuteAction
    ole.CallMethod(query_id, guid_module, module_method, parameters_module_method,
session_id)

    # Ожидаем ответ через ReadAllObject в цикле по query_id
    result = {}  # Переменная для сохранения результата ExtSdk2.CallSabyApi
    result_checked = False
    while not result_checked:

        events = json.loads(ole.ReadAllObject())  # получить события от плагина и
преобразовать в массив json объектов

        # в цикле пройти по каждому событию и найти нужное с ожидаемым query_id
        for event in events:
            if event['type'] == 'Message':
                # если нужное событие пришло, то записать результат в нужную
переменную
                print('Запрос ExtSDK2.ExecuteAction ушел, смотри логи Плагина')
                response_executeaction = event['data']['Result']
                result_checked = True
                break

        # если ожидаемое событие не найдено, то сделать таймаут в 300мс
        ole.Sleep(300)
    return response_executeaction
```

Метод ExtSDK2.ExecuteAction

Функция для отправки запроса:

```
def WriteDocumentEx():
    query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по
которому будем ожидать ответ
    module_method = "ExtSdk2.WriteDocumentEx"  # метод, который хотим позвать
    document_id = "d8e74588-f9e9-4a50-a4f8-82753ab" + str(random.randint(10000,
99999)) #генерирую рандомный ИД документа
    parameters_module_method = json.dumps(
                                            {
                                            "Document": {
                                                "Вложение": [
                                                    {
                                                    "Идентификатор": "8b8d54e2-
44ae-4de3-8433-f4a7c86c2c5e",
                                                    "Файл": {
                                                        "ДвоичныеДанные":
"PD94bWwgdmVyc2lvbj0iMS4wIiBlbmNvZGluZz0id2luZG93cy0xMjUxIiA/Pgo81ODp6yDC5fDxz/Du4z0i0c

                                                        "Имя":
"ON_NSCHFDOPPR_2BE24e62d2f2ba54942bfe49025c553cc62_2BE8f3178e7ae8c416790175e95e9aed524_
DA6D-4285-A69D-187EDE268D0F.xml"
                                                    },
                                                        "Подпись": [
                                                            {
                                                            "Файл": {
                                                            "Имя":
"ON_NSCHFDOPPR_2BE24e62d2f2ba54942bfe49025c553cc62_2BE8f3178e7ae8c416790175e95e9aed524_
DA6D-4285-A69D-187EDE268D0F.xml.sgn",
                                                            "ДвоичныеДанные":
"MIIfjQYJKoZIhvcNAQcCoIIffjCCH3oCAQExDjAMBggqhQMHAQECAgUAMAsGCSqGSIb3DQEHAaCCClowggpWMI

                                                                    },
                                                            "Сертификат":
```

```
                                                                    {

    "Доверенность": [

    {"ИдентификаторМЧД": "832fa66c-0494-4a09-bf1e-c80def4b6805"}
                                                                        ]
                                                                    }
                                                                }
                                                                    ]
                                                            }
                                                        ],

    "ДопПоля":"ЭтапВернутьВсеСертификаты",
                                                        "Дата": "27.09.2023",
                                                        "Номер": "OLA27092023",
                                                        "Идентификатор": document_id,
                                                        "Контрагент": {
                                                            "СвЮЛ": {
                                                            "ИНН": "2600303385",
                                                            "КПП": "260001001",
                                                            "Название": "Новый получатель"
                                                            }
                                                        },
                                                        "НашаОрганизация": {
                                                            "СвЮЛ": {
                                                            "ИНН": "6732020599",
                                                            "КПП": "673201001"
                                                            }
                                                        },
                                                        "Примечание": "Тест ExtSDK2
    Степанова",
                                                        "Тип": "ДокОтгрИсх"
                                                                }
                                                    }
                                                )

        # Вызов метода ExtSdk2.ExecuteAction
        ole.CallMethod(query_id, guid_module, module_method, parameters_module_method,
    session_id)

        # Ожидаем ответ через ReadAllObject в цикле по query_id
        result = {}  # Переменная для сохранения результата ExtSdk2.CallSabyApi
        result_checked = False
        while not result_checked:

            events = json.loads(ole.ReadAllObject())  # получить события от плагина и
    преобразовать в массив json объектов

            # в цикле пройти по каждому событию и найти нужное с ожидаемым query_id
            for event in events:
                if event['type'] == 'Event':
                    # если нужное событие пришло, то записать результат в нужную
    переменную
                    print('Запрос ExtSDK2.WriteDocumentEx ушел, смотри логи Плагина')
                    response_writedocumentex  = event['data']['data']['Result']
                    result_checked = True
                    break

            # если ожидаемое событие не найдено, то сделать таймаут в 300мс
            ole.Sleep(300)
        return response_writedocumentex
```

Не забудьте ее позвать, например, так:

```
WriteDocumentEx()
```

Метод ExtSDK2.CallSabyAPI

Этот метод позволяет позвать любой API-запрос через ExtSDK2. Описание есть на хелпе.

Я использовала метод для получения списка МЧД по организации и чтения конкретной МЧД. Вообще, конечно, для чтения существует ExtSDK2.ReadDocument, который используется, например, в обработке. Но метод недокументирован, клиентам не даем.

Функция для отправки запроса СБИС.СписокДокументов через CallSabyAPI:

```
"""Функция, зовущая СБИС.СписокДокументов через CallSabyApi"""


def list_of_documents():
    query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по
которому будем ожидать ответ
    module_method = "ExtSdk2.CallSabyApi"  # метод, который хотим позвать
    parameters_module_method = json.dumps({
                                           "Method": "СБИС.СписокДокументов",
                                           "Params": {
                                                      "Тип": "Доверенность",
                                                  }},
                          ensure_ascii=True)  # сформировать параметры метода
ExtSdk2.CallSabyApi.

    # Вызов метода ExtSdk2.CallSabyApi с пройденной аутентификацией
    ole.CallMethod(query_id, guid_module, module_method, parameters_module_method,
session_id)

    # Ожидаем ответ через ReadAllObject в цикле по query_id
    result = {}  # Переменная для сохранения результата ExtSdk2.CallSabyApi
    result_checked = False
    while not result_checked:

        events = json.loads(ole.ReadAllObject())  # получить события от плагина и
преобразовать в массив json объектов

        # в цикле пройти по каждому событию и найти нужное с ожидаемым query_id
        for event in events:
            if event['type'] == 'Message' and event['queryID'] == query_id:
                # если нужное событие пришло, то записать результат в нужную
переменную
                result = event['data']['Result']
                result_checked = True
                print('Метод СБИС.СписокДокументов отработал, смотри логи Плагина')
                break

        # если ожидаемое событие не найдено, то сделать таймаут в 300мс
        ole.Sleep(300)
    return result['Документ']
```

Не забудьте ее позвать, например, так:

```
list_of_poas = list_of_documents()
```

Функция для отправки запроса СБИС.ПрочитатьДокумент через CallSabyAPI:

```
"""Функция, зовущая СБИС.СписокДокументов через CallSabyApi"""


def list_of_documents():
    query_id = str(uuid.uuid4())  # генерируем идентификатор запроса (UUID) по
которому будем ожидать ответ
```

```
    module_method = "ExtSdk2.CallSabyApi"  # метод, который хотим позвать
    parameters_module_method = json.dumps({
                                           "Method": "СБИС.СписокДокументов",
                                           "Params": {
                                                      "Тип": "Доверенность",
                                                  }},
                          ensure_ascii=True)  # сформировать параметры метода
ExtSdk2.CallSabyApi.

    # Вызов метода ExtSdk2.CallSabyApi с пройденной аутентификацией
    ole.CallMethod(query_id, guid_module, module_method, parameters_module_method,
session_id)

    # Ожидаем ответ через ReadAllObject в цикле по query_id
    result = {}  # Переменная для сохранения результата ExtSdk2.CallSabyApi
    result_checked = False
    while not result_checked:

        events = json.loads(ole.ReadAllObject())  # получить события от плагина и
преобразовать в массив json объектов

        # в цикле пройти по каждому событию и найти нужное с ожидаемым query_id
        for event in events:
            if event['type'] == 'Message' and event['queryID'] == query_id:
                # если нужное событие пришло, то записать результат в нужную
переменную
                result = event['data']['Result']
                result_checked = True
                print('Метод СБИС.СписокДокументов отработал, смотри логи Плагина')
                break

        # если ожидаемое событие не найдено, то сделать таймаут в 300мс
        ole.Sleep(300)
    return result['Документ']
```

Аналогично, не забываем ее позвать:

```
current_poa = read_document_via_callsabyapi()
```

Обратите внимание.

Если что-то упало, значит в ответе на вызываемый метод пришло что-то неожидаемое, не Event, не Message (зависит от метода), а Error, к примеру. Или структура ответа не та, что я ожидала, и надо поправить код. Оберток не писала на такие случаи, все можно увидеть в логах Плагина.

+ еще информация:

при вызове методов, относящихся к записи документа, приходит Event с типом Error:

```
{'type': 'Error', 'queryID': '955a5bd8-b91b-4f30-a513-2cd587dc4a38', 'data':
{'code': 308, 'detail': '#ASYNC', 'dump': {}, 'message': 'Ожидайте загрузки
документа', 'method_name': 'ExtSdk2.WriteDocumentEx', 'stack': [...]}}
```

Это не ошибка, надо просто в цикле подождать следующие события. Ну это если вы собрались править код или писать сами.