Сервер Модуля Узгодження

Інструкція користувача

Київ, 2022

ЗМІСТ

Перелік скорочень

АС	–	автоматизована система
СМУ	–	Сервер Модуля Узгодження
АРМ ВВ	–	автоматизоване робоче місце вантажовідправника
ЕПД	–	електронний перевізний документ
ЕД	–	електронні дані документа
ЕЦП	–	електронно-цифровий підпис
ЦСК	–	центр сертифікації ключів
XML	–	extensible markup language
DLL	–	динамічно лінкуєма бібліотека
УЗ	–	Українська залізниця
ПІ	–	попереднє інформування

1. Концепція інформаційної взаємодії

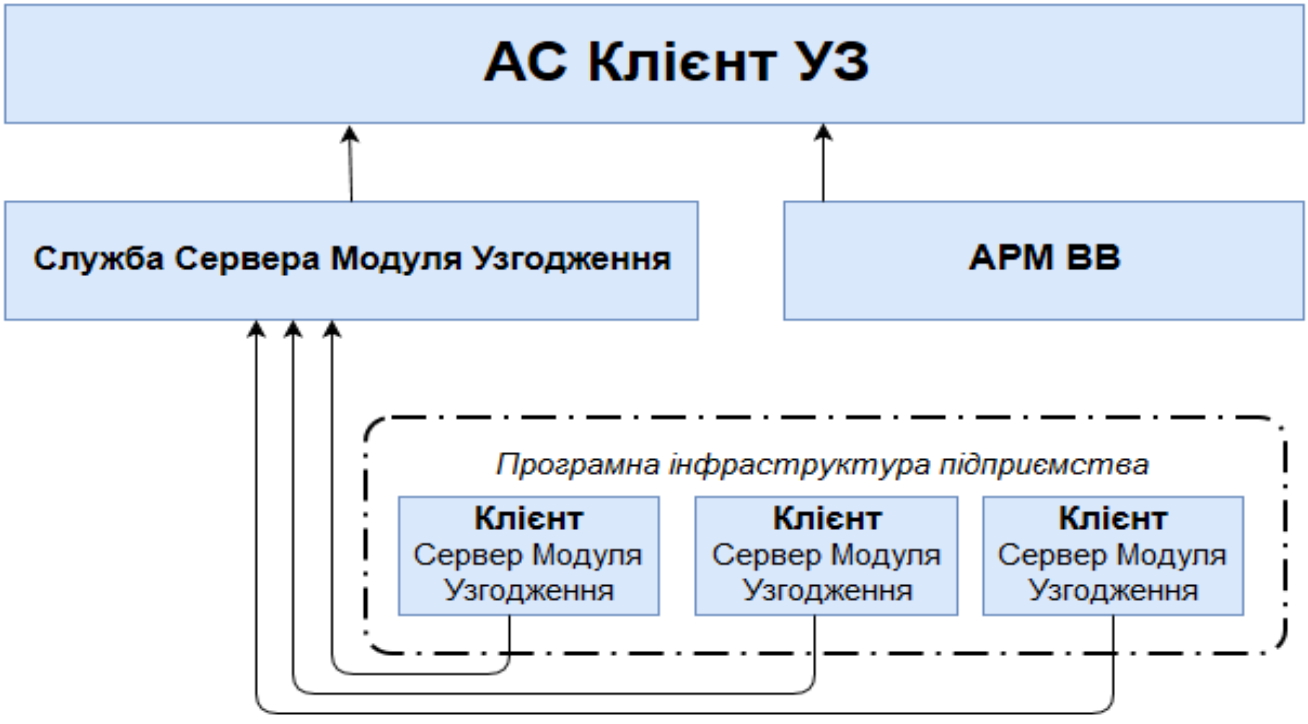
1.1. Загальні положення

Сервер Модуля Узгодження – модуль, що дозволяє прикладному програмному забезпеченню взаємодіяти з АС «Клієнт УЗ». Користувачі отримують можливість формувати перевізні документи за допомогою власних програм з подальшою передачею їх в АС «Клієнт УЗ». Крім того, *Сервер Модуля Узгодження* дозволяє призначеним для користувача програмам синхронізувати стан документів (ЕПД) електронних даних (ЕД) і чернеток користувальницької програмної системи зі станом відповідних про'єктів системи АС «Клієнт УЗ».

Фізично *Сервер Модуля Узгодження* є службою Windows, що запущена на одному ПК, до якої підключаються *Клієнти СМУ*.

Для полегшення реалізації взаємодії прикладних програм користувачів з *Клієнтом СМУ* до складу поставки включено кілька варіантів бібліотек *Клієнта*, які надають прикладним програмам дружній програмний інтерфейс для наступних мов програмування: C, C ++, COM / OLE, C #, ін. мови платформи *.NET*.

На малюнку 1 надано приблизну схему взаємодії прикладної програми користувача з АС «Клієнт УЗ» з використанням *Сервера Модуля Узгодження.*



Малюнок 1

1.2. Функції модуля узгодження

- передача перевізних та супровідних документів, створених та призначених для користувача в АС «Клієнт УЗ» в програми користувача;
- отримання інформації про зміни (і списку змін), що відбулися в ЕПД, ЕД, чернетках і супровідних документах з конкретного моменту;
- отримання окремих ЕПД, ЕД, чернеток і супровідних документів за ідентифікатором з АС «Клієнт УЗ»;
- передача дозаповнених в призначених для користувача програмах ЕПД та ЕД по прибуттю в АС «Клієнт УЗ»;
- виконання накладення ЕЦП;
- виконання перевірки ЕЦП;
- відкликання документів та супровідних документів;
- видалення чернеток та супровідних документів;
- підключення файлів електронних ключів і управління підключеними ключами;
- отримання заявок з АС «Месплан»;
- отримання добових переліків з АС «Клієнт УЗ».

2. Керівництво по встановленню та налагодженню

2.1. Інструкція по встановленню *служби Сервера Модуля Узгодження*

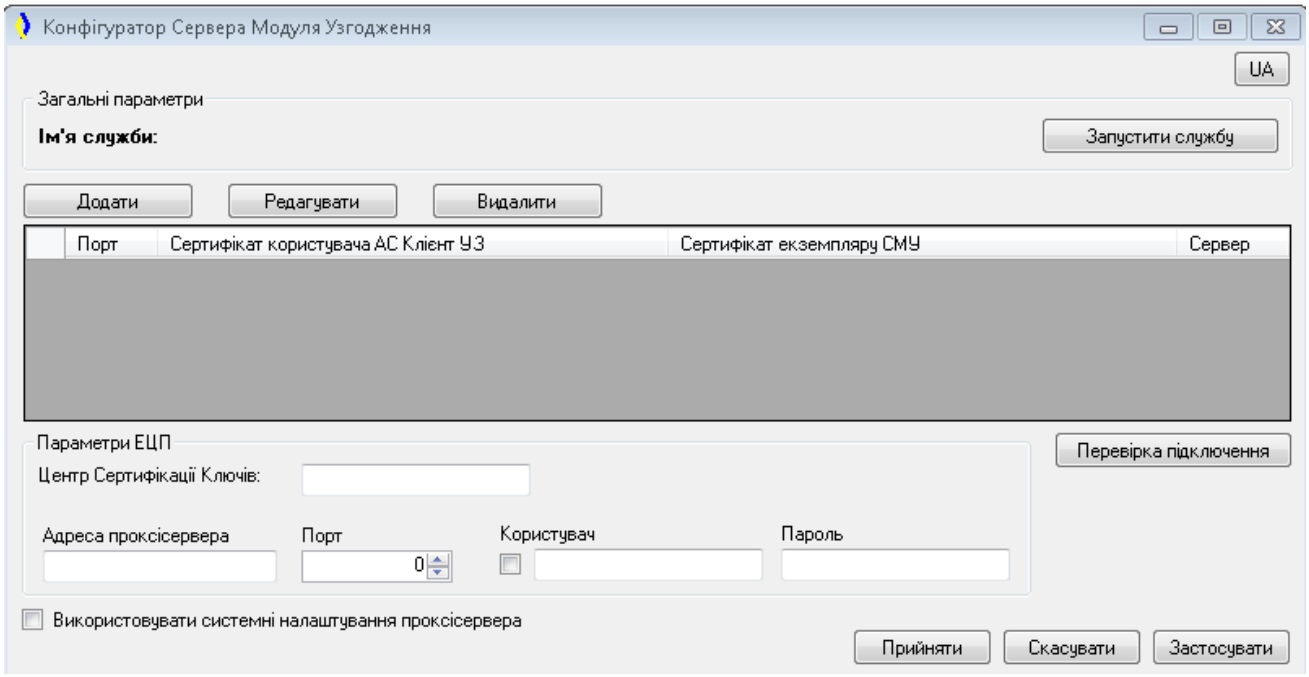
Для встановлення *служби Сервера Модуля Узгодження* потрібно запустити інсталятор *tmsoft.gohub.service.setup.msi*

В процесі інсталяції буде запропоновано вибрати директорію установки, за замовчуванням - *"C:\Program Files\KPD-UZ\Сервер Модуля Узгодження"*.

В результаті установки *Служба СМУ* буде інстальована в систему як служба Windows з ім'ям *TMSoft.Gohub (в переліку служб відображається з ім'ям «Модуль Узгодження»).* Засобами операційної системи можна вибрати, чи запускати служби *(ручний або автоматичний, за замовчуванням - автоматичний)* і стартувати службу.

2.2. Налаштування *Служби Сервера Модуля Узгодження*

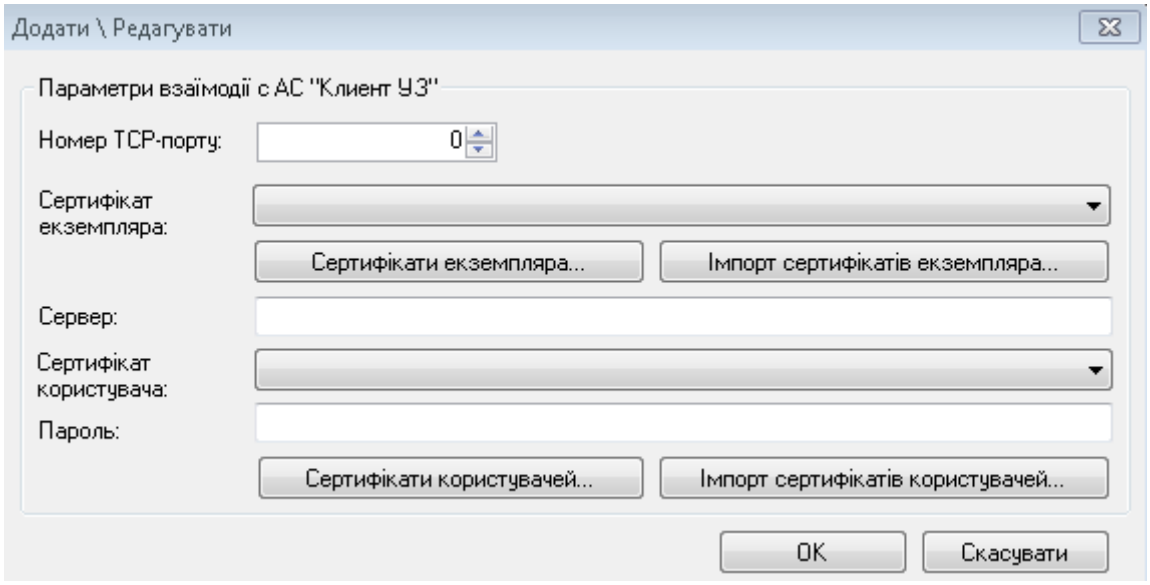
Для налаштування *Служби СМУ* в комплект поставки включена утиліта *Конфігуратор Сервера Модуля Узгодження,* яка запускається після завершення інсталяції, а також доступна в системному меню: *Пуск -> KPD-UZ -> Сервер Модуль Узгодження -> Конфігуратор.*



Вікно: *Конфігуратор Сервера Модуля Узгодження*

Конфігуратор дозволяє налаштувати наступні властивості:

- Запуск Служби СМУ: кнопка «Запустити службу» / «Зупинити службу» запускає / зупиняє Службу СМУ;
- Створювати, редагувати і видаляти дані про підключення Сервера Модуля Узгодження до АС «Клієнт» УЗ;
- Установки для перегляду ЕЦП:
    - Адреса сервера Центру Сертифікації Ключів (по замовчуванням - *«csk. uz. gov. ua »);*
    - Налаштування проксісервера - адреса і порт;
    - Налаштування користувача локальної мережі - ім'я та пароль.



Вікно додавання / редагування даних про підключення

- Номер TCP-порту, по якому здійснюється взаємодія *Служби* і *Клієнта СМУ;*
- Параметри взаємодії з АС «Клієнт УЗ»:
    - Сертифікат екземпляру - сертифікат надає можливість роботи Службі СМУ і взаємодіяти з АС «Клієнт УЗ»;
    - Кнопка «Імпорт ...» надає можливість внести дані про сертифікат екземпляра з файлу сертифіката;
    - В поле «Сервер» задається адреса сервера АС «Клієнт УЗ», який автоматично встановлюється з сертифіката користувача АС «Клієнт УЗ»;
    - В поле «Ім'я користувача» обирається користувач АС «Клієнт УЗ», який встановлюється з сертифіката користувача АС «Клієнт УЗ»;
    - В поле «Пароль користувача» задається пароль доступу до АС «Клієнт УЗ»;
    - Кнопка «Сертифікати ...» відкриває браузер дозволів, які стосуються користувачів Служби СМУ, від яких йде звернення до АС «Клієнт УЗ» і зберігаються в операційній системі;
    - Кнопка «Імпорт сертифікатів користувачів ...» надає можливість внести дані про користувача, від якого йде звернення до АС «Клієнт УЗ» з файлу сертифіката;

2.3. Опис установки *Клієнта Сервера Модуля Узгодження*

Для встановлення *Клієнта Сервера Модуля Узгодження* потрібно запустити інсталятор *tmsoft.gohub.client.setup.msi*.

В процесі інсталяції буде запропоновано вибрати директорію установки, за замовчуванням - *C:\Program File\ KPD-UZ\Клієнт Модуля Узгодження.*

В результаті установки *Клієнт СМУ* в теці установки розміщуються наступні файли:

- Тека ***bin*** містить підготовлений до запуску файл консолі (його можна побачити і в розділі «Пуск / Всі програми / *KPD-UZ* / Клієнт модуля узгодження» як *«Консоль Модуля Узгодження»)* і три файли різних варіантів бібліотеки Клієнта:
    - ***TMSoft.gohub.client.dll*** - *динамічно лінкуємая бібліотека* (DLL) для родини операційних систем Windows, починаючи з Windows 2000 і вище;
    - ***TMSoft.gohub.client.com.dll*** - *COM / OLE об'єкт;*
    - ***TMSoft.Gohub.Client.Net.dll*** - *.NET версія бібліотеки.*

    Примітка: бібліотеки можна вільно переміщувати, копіювати і т.д., окрім бібліотеки COM / OLE, яка при установці реєструється в системі із зазначенням шляху розміщення файлу.
- Тека ***doc*** містить - Керівництво користувача і *Changes.txt* з описами змін в попередніх версіях;
- Тека ***inc*** містить файли для лінкування бібліотек - заголовки *gohub.client.h* і *gohub.client.errors.h,* для динамічно лінкуємой бібліотеки і файл *gohubclientcom.idl* з оголошенням інтерфейсів для *COM / OLE* версії бібліотеки;
- Тека ***lib*** містить два файли: статична бібліотека TMSoft.gohub.client.lib і бібліотека типів TMSoft.gohub.client.com.tlb для додатків, що використовують *COM / OLE* версію.
- Тека ***samples*** містить xml-файли зразків повідомлень для формування електронних перевізних документів різних видів відправлень, а також файл *Gohub.Client.Test.xls,* з макросами-прикладами використання *COM / OLE* версії бібліотеки;
- Тека ***schemas*** містить файли *xml-Схема* (*. Xsd)* з формалізованим описом структур xml-повідомлень електронних перевізних документів.

3. Загальний опис бібліотеки

Прикладна бібліотека *«Клієнт Сервера Модуля Узгодження»* (далі - *Бібліотека)* призначена для спрощення взаємодії прикладних програм із *Службою Сервера Модуля Узгодження. Бібліотека* реалізована відповідно до вимог, викладених в «Концепції інформаційної взаємодії» *Сервера Модуля Узгодження.*

Фізично *Бібліотека* складається з декількох варіантів бібліотеки *TMSoft.gohub.client,* під різними мовами програмування:

- *TMSoft.gohub.client.dll - динамічно лінкуемой бібліотекою* (DLL) для родини операційних систем Windows, починаючи з Windows 2000 і вище;
- *TMSoft.gohub.client.com.dll - COM / OLE об'єкт;*
- *TMSoft.Gohub.Client.Net.dll - .NET версія бібліотеки.*

Бібліотека надає набір простих і в той же час ефективних засобів діагностики помилок, що виникають при її використанні.

Програмний інтерфейс *Бібліотеки* надає прикладним програмам можливість вирішувати такі завдання:

- Відправляти в АС «Клієнт УЗ» перевізні документи, створені поза системою АС «Клієнт УЗ».
- Відслідковувати появу в АС «Клієнт УЗ» нових документів та модифікацію існуючих.
- Запитувати з АС «Клієнт УЗ» окремі документи по їх унікальним ідентифікаторам.
- Виконувати перекодування документів і отримувати детальну інформацію про можливі помилки, що виникають при роботі з *Бібліотекою.*
- Відкликати документи і видаляти чернетки.
- Накладати ЕЦП на перевізні документи.
- Виробляти перевірку ЕЦП, накладеної на перевізні *документи.*
- Отримувати заявки з АС «Месплан».
- Отримувати добові переліки з АС «Клієнт УЗ»

Для взаємодії з АС «Клієнт УЗ» прикладна програма повинна підключитися до *Сервера Модуля Узгодження* за допомогою функції *«Створити_з'єднання».* Результатом виклику цієї функції є дескриптор (покажчик) об'єкта або об'єкт з'єднання, який буде використовуватися у всіх операціях взаємодії з АС «Клієнт УЗ»: відправлення та запиту перевізних документів, а також запиту списків модифікацій.

Для відправлення чернетки перевізного документа в АС «Клієнт УЗ» прикладна програма спочатку повинна створити об'єкт документа за допомогою функції `«Створити_документ»`, або завантажити документ з файлу за допомогою функції

«Завантажити_документ». Після цього можна відправити документ в АС «Клієнт УЗ» за допомогою функції «Відправити_документ». Формат чернеток перевізних документів, що відправляються в АС «Клієнт УЗ», повинен відповідати вимогам викладеним в документі «Структура, склад та формат реквізитів та атрибутів електронного перевізного документа», опублікованому на офіційному сайті УЗ.

В результаті відправлення в АС «Клієнт УЗ» документ отримує унікальний ідентифікатор по якому можна в майбутньому запросити документ з АС «Клієнт УЗ» протягом усього його життєвого циклу. Для запиту документа з АС «Клієнт УЗ» служить функція «Запросити_документ» .

Отримані документи можна зберегти в файл за допомогою функції «Зберегти_документ», або отримати текст документа у вигляді рядка за допомогою функції «Текст_документа». Ця функція повертає текст документа (без xml-заголовка) в поточному кодуванні, яке може бути задано за допомогою функції «Встановити_кодування» (за замовчуванням - 1251).

Документи, отримані з АС «Клієнт УЗ» за замовчуванням мають кодування *utf-8* (кодова сторінка 65001). При збереженні документа в файл за допомогою функції «Зберегти_документ» ви можете вказати кодову сторінку в якій хочете зберегти документ .

Крім унікального ідентифікатора, кожен документ в системі АС «Клієнт УЗ» має таку важливу властивість як номер ревізії. Номер ревізії, так само як і ідентифікатор, призначаються документу при першому попаданні в систему. Однак, на відміну від ідентифікатора, номер ревізії не є постійною властивістю і змінює своє значення при кожній модифікації документа. Всі модифікації по всім документам системи АС «Клієнт УЗ» мають наскрізну нумерацію.

Номери ревізії дозволяють зовнішнім системам відслідковувати зміни в документах, що знаходяться в АС «Клієнт УЗ». За допомогою функції «Завантажити_наступний_документ» додатка можуть отримувати документи відповідно до послідовності їх послідовності їх модифікацій.

Всі назви функцій наведені українською мовою спеціально тому, що семантика в різних *бібліотеках* відрізняється, але сутність логіки дій і команд не змінюється.

4. Динамічно лінкуєма бібліотека tmsoft.gohub.client.dll

4.1. Короткий опис бібліотеки

Gohub.client.dll є *динамічно лінкуємою бібліотекою* (DLL) для сімейства операційних систем Windows, починаючи з Windows 2000 і вище. Для роботи перших версій *Бібліотеки* будуть потрібні також платформа *.NET Framework* версії не нижче 2.0, а також бібліотеки часу виконання *Microsoft CRT* . Дистрибутиви цих додаткових компонентів входять в комплект поставки *Бібліотеки* .

Додатково в комплект поставки включені: статична бібліотека *gohub.client.lib* для лінковки з *Бібліотекою* програм, написаних мовою *Microsoft Visual C / C ++* , та заголовки *gohub.client.h* , що містить декларації всіх функцій, типів і констант, що становлять програмний інтерфейс *Бібліотеки* .

У *Бібліотеці* є ряд функцій, що працюють з малими даними виключно в юнікодовому кодуванні UTF-16 Little Indian *(далі - UTF-16)* , всі вони містять в назві букву ***W*** .

Більш детальний опис програмного інтерфейсу бібліотеки описано в наступних пунктах. Так само можна переглянути заголовки gohub.client.h (Додаток А) і gohub.client.errors.h (Додаток Б).

Для полегшення і прискорення знайомства з програмним інтерфейсом Бібліотеки служать приклади її використання, наведені в п.4.5.

4.2. Типи даних

`GohubBool`	Булева змінна
`GohubWChar`	Символ представлений в кодуванні UTF-16
`GohubConnection`	З'єднання з *Модулем Узгодження*
`GohubDocument`	Перевізні документи
`GohubAttachment`	Супровідні документи
`GohubEData`	Електронні дані попереднього інформування
`GohubPiPackage`	Пакети попереднього інформування
`GohubPiPackageToEData`	Інформація про електронні даних в пакеті попереднього інформування
`GohubFdu92`	Документи ФДУ-92
`GohubGu46`	Документи ГУ-46
`GohubGu45`	Документи ГУ-45
`GohubInformServicesDoc`	Документи інформаційних послуг
`GohubDispatchInfo`	Перелік № Замовлень для перевізного документу
`GohubPSTDInfo`	Перелік № замовлення ПСТД

GohubBool

Булева змінна.

Об'явлення:

```
typedef int GohubBool;
```

GohubWChar

Рядкова змінна в кодуванні UTF-16.

Об'явлення:

```
typedef unsigned char GohubWChar;
```

GohubConnection

Об'єкти цього типу служать для подання підключення прикладної програми до служби *Модуля Узгодження*

Об'явлення:

```
typedef struct GohubConnection GohubConnection;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можлива тільки через відкриті функції *Бібліотеки,* які оперують покажчиками на ці про'єкти.

GohubDocument

Об'єкти цього типу служать для подання електронних перевізних документів.

Об'явлення:

```
typedef struct GohubDocument GohubDocument;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції *Бібліотеки,* які оперують покажчиками на ці про'єкти.

GohubAttachment

Об'єкти цього типу служать для подання електронних супровідних документів.

Об'явлення:

```
typedef struct GohubAttachment GohubAttachment;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubEData

Об'єкти цього типу служать для подання електронних даних попереднього інформування.

Об'явлення:

```
typedef struct GohubEData GohubEData;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubPiPackage

Об'єкти цього типу служать для подання пакетів попереднього інформування .

Об'явлення:

```
typedef struct GohubPiPackage GohubPiPackage;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubPiPackageToEData

Об'єкти цього типу служать для представлення інформації про електронні даних в пакеті попереднього інформування.

Об'явлення:

```
typedef struct GohubPiPackageToEData GohubPiPackageToEData;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubFdu92

Об'єкти цього типу служать для подання електронних накопичувальних карток ФДУ-92.

Об'явлення:

```
typedef struct GohubFdu92 GohubFdu92;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubGu46

Об'єкти цього типу служать для подання електронних відомостей про користування вагонами / контейнерами ГУ-46 .

Об'явлення:

```
typedef struct GohubGu46 GohubGu46;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubGu45

Об'єкти цього типу служать для подання електронних пам'яток про подачу / прибирання вагонів і видачу / прийом контейнерів ГУ-45.

Об'явлення:

```
typedef struct GohubGu45 GohubGu45;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubInformServicesDoc

Об'єкти цього типу служать для подання документів інформаційних послуг.

Об'явлення:

```
typedef struct GohubInformServicesDoc GohubInformServicesDoc;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubDispatchInfo

Об'єкти цього типу служать для отримання переліку інформації про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати)**.**

Об'явлення:

```
typedef struct GohubDispatchInfo GohubDispatchInfo;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

GohubPSTDInfo

Об'єкти цього типу служать для отримання переліку № замовлення ПСТД

Об'явлення:

```
typedef struct GohubPSTDInfo GohubPSTDInfo;
```

Примітки:

Повне визначення цього типу недоступно прикладній програмі. Об'єкти цього типу, а також процедури їх створення та знищення вміщені в Бібліотеці. Взаємодія з про'єктами цього типу можливо тільки через відкриті функції Бібліотеки, які оперують покажчиками на ці про'єкти.

4.3. Статуси документів

4.3.1. Стани перевізного документа

Коди стану перевізного документа в *Бібліотеці* представлені перерахуванням (*enum*) `Gohub DocumentStatus`, що містить набір констант.

Зведена таблиця кодів станів перевізного документа

`gohub_document_status_unknown = 0`	Статус невідомий

gohub_document_status_draft = 1	Чернетка
gohub_document_status_sending = 2	Документ передається товарному касиру
gohub_document_status_registered = 3	Документ переданий товарному касиру
gohub_document_status_reclaiming = 4	Документ відкликається від товарного касира
gohub_document_status_accepted = 5	Вантаж прийнято до перевезення
gohub_document_status_delivered = 6	Вантаж прибув
gohub_document_status_recieved = 7	Вантаж отримано одержувачем
gohub_document_status_uncredited = 8	Документ розкредитовано товарним касиром
gohub_document_status_recieved_draft = 9	Вантаж отримано одержувачем і редагується
gohub_document_status_recieved_sending = 10	Вантаж отримано одержувачем і переданий товарному касиру
gohub_document_status_recieved_reclaiming = 11	Вантаж отримано одержувачем і відкликається від товарного касира
gohub_document_status_canceled = 12	Документ зіпсований товарним касиром
gohub_document_status_locked = 13	Документ заблокований

Більш детальний опис програмного інтерфейсу бібліотеки можна отримати з файлу gohub.client.h (Додаток А).

4.3.2. Стани накопичувальної картки ФДУ-92

Коди стану накопичувальної картки ФДУ-92 в *Бібліотеці* представлені перерахуванням (*enum*) `GohubFdu92Status`, що містить набір констант.

Зведена таблиця кодів станів накопичувальної картки ФДУ-92

gohub_fdu92_status_unknown = -1	Статус невідомий
gohub_fdu92_status_approoving = 0	Документ отримано на узгодження
gohub_fdu92_status_approving_modified = 1	Документ отримано на узгодження і відредагований користувачем
gohub_fdu92_status_confirmed = 2	Документ узгоджений та підписаний
gohub_fdu92_status_canceled = 3	Документ скасований
gohub_fdu92_status_agreed_noted_sending = 4	Документ погоджений з правками, відправляється
gohub_fdu92_status_agreed = 5	Документ погоджений
gohub_fdu92_status_agreed_noted = 6	Документ погоджений з правками
gohub_fdu92_status_expired = 7	Документ не узгоджений вчасно
gohub_fdu92_status_agreed_sending = 8	Документ погоджений, відправляється
gohub_fdu92_status_confirmed_paper = 9	Документ підтверджений товарним касиром
gohub_fdu92_status_rejecting = 10	Документ ініційований паперовий, надсилається
gohub_fdu92_status_rejected = 11	Документ ініційований паперовий
gohub_fdu92_status_paper = 60	Паперовий документ

4.3.3. Стани відомостей про користування вагонами / контейнерами ГУ-46

Коди стану відомостей про користування вагонами / контейнерами ГУ-46 в *Бібліотеці* представлені перерахуванням *(enum)* `GohubGU46Status`, що містить набір констант.

Зведена таблиця кодів станів відомості користування вагонами / контейнерами ГУ-46

gohub_gu46_status_unknown = -1	Статус невідомий
gohub_gu46_status_approoving = 0	Документ отримано на узгодження
gohub_gu46_status_approving_modified = 1	Документ отримано на узгодження і відредагований користувачем
gohub_gu46_status_confirmed = 2	Документ узгоджений та підписаний
gohub_gu46_status_canceled = 3	Документ скасований
gohub_gu46_status_agreed_noted_sending = 4	Документ погоджений з правками, відправляється
gohub_gu46_status_agreed = 5	Документ погоджений
gohub_gu46_status_agreed_noted = 6	Документ погоджений з правками
gohub_gu46_status_expired = 7	Документ не узгоджений вчасно
gohub_gu46_status_agreed_sending = 8	Документ погоджений, відправляється
gohub_gu46_status_confirmed_paper = 9	Документ підтверджений товарним касиром
gohub_gu46_status_rejecting = 10	Документ ініційований паперовий, відправляється
gohub_gu46_status_rejected = 11	Документ ініційований паперовий
gohub_gu46_status_paper = 60	Паперовий документ

4.3.4. Стани пам'ятки про подачу / прибирання вагонів і видачу / прийом контейнерів ГУ-45

Коди стану пам'яток про подачу / прибирання вагонів і видачу / прийом контейнерів ГУ-45 в *Бібліотеці* представлені перерахуванням *(enum)* `GohubGU45Status`, що містить набір констант.

Зведена таблиця кодів станів пам'ятки про подачу / прибирання вагонів і видачу / прийом контейнерів ГУ-45

gohub_gu45_status_unknown = -1	Статус невідомий
gohub_gu45_status_confirmed = 2	Документ узгоджений та підписаний
gohub_gu45_status_canceled = 3	Документ скасований
gohub_gu45_status_confirmed_paper = 9	Документ підтверджений товарним касиром
gohub_gu45_status_paper = 60	Паперовий документ

4.3.5. Стан перевізного документу при запиті статусу документа в підписі за індексом

Коди стану перевізного документа в *Бібліотеці* представлені перерахуванням (*enum*) `UzRwcDocStatus`, що містить набір констант.

Зведена таблиця кодів станів перевізного документа на момент накладання підпису

none = 0	Статус невідомий
project = 1	ЕПД, що передаються вантажовідправником
accepted = 2	ЕПД, що прийняті до перевезення
resend = 3	ЕПД, що зазнали змін під час прямування вантажу
arrived = 4	ЕПД, що прибули на станцію призначення
reviewed = 5	ЕПД, що передаються вантажоодержувачем
uncredited = 6	Розкредитовані ЕПД
foreign = 7	ЕПД, отримані від іноземної залізниці
entered = 8	ЕПД, оброблені по входу на УЗ
exited = 9	ЕПД, оброблені по виходу з УЗ

Більш детальний опис програмного інтерфейсу бібліотеки можна отримати з файлу gohub.client.h (Додаток А).

4.4. Основні функції

Більш детальний опис програмного інтерфейсу бібліотеки можна отримати з файлу gohub.client.h (Додаток А).

4.4.1. Робота з кодовими сторінками

`gohub_set_codepage`	Встановити кодування по її числовому позначенню
`gohub_encoding_codepage`	Отримати кодування по її назві
`gohub_encoding_codepage_w`	Отримати кодування по її назві `(параметр в UTF-16)`

gohub_set_codepage

Встановити поточну кодову сторінку для всіх малих параметрів *(крім тих, що в UTF-16)*.

Об'явлення:

```
int gohub_set_codepage(int codepage);
```

Параметри:

`codepage`

Кодування сторінки;

Результат:

Встановлює кодувальну сторінку для переданих рядком даних.

gohub_encoding_codepage

Отримати цифрове позначення кодової сторінки по її текстовому назвою.

Об'явлення:

```
int gohub_encoding_codepage(const char* encodingName);
```

Параметри:

`encodingName`

Назва кодової сторінки;

Результат:

У разі успіху - Повертає цифрове кодування, що позначає запитану сторінку. У разі помилки - повертає нуль. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_encoding_codepage_w

Отримати цифрове позначення кодової сторінки по її текстовому назвою в кодуванні UTF-16.

Об'явлення:

```
int gohub_encoding_codepage_w(
const GohubWChar* encodingName);
```

Параметри:

`encodingName`

Назва кодової сторінки;

Результат:

У разі успіху - Повертає цифрове кодування, що позначає запитану сторінку. У разі помилки - повертає нуль. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.2. Підключення до Модуля Узгодження

`gohub_connect`	Створити з'єднання з *Модулем Узгодження*
`gohub_connect_w`	Створити з'єднання з *Модулем Узгодження* `(параметр в UTF-16)`
`gohub_disconnect`	Закрити з'єднання з *Модулем Узгодження*

gohub_connect

Встановити з'єднання з *Модулем Узгодження.*

Об'явлення:

```
GohubConnection* gohub_connect(
      const char* host,
      int port);
```

Параметри:

`host`

DNS-ім'я або IP адресаа комп'ютера, на якому встановлений *Модуль Узгодження*;

`port`

Номер порту для встановлення TCP з'єднання з *Модулем Узгодження.*

Результат:

У разі успіху - Покажчик з'єднання з Модулем Узгодження . У разі помилки - нульовий покажчик. Інформацію про помилку можна отримати за допомогою функції gohub_last_error та ін .

Примітки:

Після закінчення роботи зі з'єднанням, його необхідно закрити за допомогою функції gohub_disconnect .

gohub_connect_w

Встановити з'єднання з Модулем Узгодження.

Об'явлення:

```
GohubConnection* gohub_connect(
      const GohubWChar* host,
      int port);
```

Параметри:

`host`

DNS-ім'я або IP адресаа комп'ютера, на якому встановлений *Модуль Узгодження* *`(в UTF-16)`*;

`port`

Номер порту для встановлення TCP з'єднання з *Модулем Узгодження* .

Результат:

У разі успіху - Покажчик з'єднання з Модулем Узгодження . У разі помилки - нульовий покажчик. Інформацію про помилку можна отримати за допомогою функції gohub_last_error та ін .

Примітки:

Після закінчення роботи зі з'єднанням, його необхідно закрити за допомогою функції gohub_disconnect.

gohub_disconnect

Закрити з'єднання з Модулем Узгодження.

Об'явлення:

```
GohubBool gohub_disconnect(
    GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.3. Робота з документами

gohub_load_document	Завантажити документ
gohub_load_document_w	Завантажити документ (в UTF-16)
gohub_create_document	Створити документ
gohub_create_document_w	Створити документ (в UTF-16)
gohub_query_document	Запросити документ
gohub_query_document_w	Запросити документ (в UTF-16)
gohub_query_next_document	Запросити наступний документ
gohub_document_id	Запросити ID документа
gohub_document_id_w	Запросити ID документа (в UTF-16)
gohub_document_revision	Запросити ревізію документа
gohub_document_text	Запросити текст документа
gohub_document_text_w	Запросити текст документа (в UTF-16)
gohub_document_data_text	Запросити текст електронних даних документа
gohub_document_data_text_w	Запросити текст електронних даних документа (в UTF-16)
gohub_document_status	Запросити статус документа
gohub_document_size	Запросити передбачуваний розмір документа представлений в поточній кодової сторінці
gohub_document_measure_equip_num	Отримання відомостей вагонівимірювальної техніки
gohub_document_measure_equip_num_w	Отримання відомостей вагонівимірювальної техніки (в UTF-16)
gohub_document_set_measure_equip_num_w	Встановлення відомостей вагонівимірювальної техніки
gohub_document_set_verified_empty_weight_for_wagon	Встановлення уточненої ваги тари вагона
gohub_document_get_verified_empty_weight_for_wagon	Отримання уточненої ваги тари вагона
gohub_document_business_unit_num	Отримання номера філії ПрАТ УЗ
gohub_document_business_unit_num_w	Отримання номера філії ПрАТ УЗ (в UTF-16)
gohub_document_set_business_unit_num	Встановлення номера філії ПрАТ УЗ
gohub_document_set_business_unit_num_w	Встановлення номера філії ПрАТ УЗ (в UTF-16)
gohub_document_get_foreign_not_accept	Отримання відмітки повернення

	неприйнятих прикордонними станціями іноземних залізниць вагонів на територію України
gohub_send_document	Надіслати документ
gohub_save_document	Зберегти документ
gohub_save_document_w	Зберегти документ (в UTF-16)
gohub_save_document_data	Зберегти електронні дані документа
gohub_save_document_data_w	Зберегти електронні дані документа (в UTF-16)
gohub_close_document	Закрити документ
gohub_reclaim_document	Відкликати документ
gohub_reclaim_document_w	Відкликати документ (в UTF-16)
gohub_delete_document	Видалити документ
gohub_delete_document_w	Видалити документ (в UTF-16)
gohub_send_received_document	Надіслати документ по прибуттю
gohub_send_received_document_w	Надіслати документ по прибуттю (в UTF-16)
gohub_document_get_otpr	Зберегти актуальний текст документа
gohub_document_get_otpr_w	Зберегти актуальний текст документа (у кодуванні UTF-8)
gohub_document_get_otpr_string	Отримати актуальний текст перевізного документа
gohub_document_get_otpr_string_w	Отримати актуальний текст перевізного документа (у кодуванні UTF-8)
gohub_document_warning	Отримати текст попередження після відправки документу
gohub_document_warning_w	Отримати текст попередження після відправки документу(у кодуванні UTF-8)
gohub_query_and_save_document_printable_form	Запитати друковану форму документа
gohub_query_and_save_document_printable_form_w	Запитати друковану форму документа (в UTF-16)
gohub_query_and_save_document_archive	Запит архіву з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД
gohub_query_and_save_document_archive_w	Запит архів з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД (в UTF-16)

gohub_load_document

Створення про'єкта документа (GohubDocument) з xml-файлу.

Об'явлення:

```
GohubDocument* gohub_load_document(
      const char* path);
```

Параметри:

`path`

шлях до xml-файлу, з якого завантажується документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін .

gohub_load_document_w

Створення про'єкта документа (GohubDocument) з xml-файлу.

Об'явлення:

```
GohubDocument* gohub_load_document_w(
      const GohubWChar* path);
```

Параметри:

`path`

шлях до xml-файлу в кодуванні UTF-16, з якого завантажується документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_document

Створення про'єкта документа (GohubDocument) з рядка з xml-структурою.

Об'явлення:

```
GohubDocument* gohub_create_document(
      const char* content);
```

Параметри:

`content`

рядок з xml-структурою з вмістом документа;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_document_w

Створення про'єкта документа (GohubDocument) з рядка з xml-структурою.

Об'явлення:

```
GohubDocument* gohub_create_document_w(
      const GohubWChar* content);
```

Параметри:

`content`

рядок з xml-структурою з вмістом документа у кодуванні UTF-16;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_document

Запит документа з сервера АС «Клієнт УЗ» за його ID.

Об'явлення:

```
GohubDocument* gohub_query_document(
      GohubConnection* connection,
      const char* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа, що запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_document_w

Запит документа з сервера АС «Клієнт УЗ» за його ID.

Об'явлення:

```
GohubDocument* gohub_query_document_w(
      GohubConnection* connection,
      const GohubWChar* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа (в кодуванні UTF-16), який запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_document

Запит документа з сервера АС «Клієнт УЗ» наступного за списком ревізій, починаючи від ревізії переданої параметром.

Об'явлення:

```
GohubDocument* gohub_query_next_document(
      GohubConnection* connection,
      int lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

ревізія документа, від якої починається пошук наступного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_document2

Запит документа з сервера АС «Клієнт УЗ» наступного за списком ревізій, починаючи від ревізії переданої параметром.

Об'явлення:

```
GohubDocument* gohub_query_next_document2(
      GohubConnection* connection,
      int lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

ревізія документа, від якої починається пошук наступного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_id

Запит унікального ID документа з сервера АС «Клієнт УЗ»..

Об'явлення:

```
GohubDocument* gohub_query_document(
      GohubConnection* connection,
      const char* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа, що запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_id_w

Запит документа з сервера АС «Клієнт УЗ» за його ID.

Об'явлення:

```
GohubDocument* gohub_query_document_w(
      GohubConnection* connection,
      const GohubWChar* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа (в кодуванні UTF-16), який запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_revision

Запит ревізії документа.

Об'явлення:

```
int gohub_document_revision(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (GohubDocument), ревізія якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_text

Запит тексту документа з про'єкта документ (`GohubDocument`).

Об'явлення:

```
const char* gohub_document_text(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_text_w

Запит тексту документа з про'єкта документ (`GohubDocument`) у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_text_w(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`, текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_data_text

Запит тексту електронних даних документа з про'єкта документ (`GohubDocument`) в заданій версії ЕПД.

Об'явлення:

```
const char* gohub_document_text(
      GohubDocument* document,
      int epdVersion);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), текст електронних даних якого запитується;

`epdVersion`

версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_data_text_w

Запит тексту електронних даних документа (у кодуванні UTF-16) з про'єкту документ (`GohubDocument`) в заданій версії ЕПД.

Об'явлення:

```
const GohubWChar* gohub_document_text_w(
      GohubDocument* document,
      int epdVersion);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), текст електронних даних якого запитується;

`epdVersion`

версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_status

Запит статусу документа.

Об'явлення:

```
GohubDocumentStatus gohub_document_status(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), статус якого запитується;

Результат:

У разі успіху - значення з перерахування GohubDocumentStatus . У разі помилки - «-1». Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_size

Запит передбачуваного розміру файлу, який вийде при збереженні документа у поточній кодуванні (GohubDocument).

Об'явлення:

```
int gohub_document_size(
      GohubDocument* document);
```

Параметри:

document

Об'єкт документа (GohubDocument), текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_measure_equip_num

Отримання відомостей вагонівимірювальної техніки

Об'явлення:

```
const char* gohub_document_measure_equip_num(
      GohubDocument* document);
```

Параметри:

document

Об'єкт документа (GohubDocument), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_measure_equip_num_w

Отримання відомостей вагонівимірювальної техніки у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_measure_equip_num_w(
      GohubDocument* document);
```

Параметри:

document

Об'єкт документа (GohubDocument), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_measure_equip_num

Встановлення відомостей вагонівимірювальної техніки.

Об'явлення:

```
GohubBool gohub_document_set_measure_equip_num(
      GohubDocument* document, const char* value);
```

Параметри:

document

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`value`

значення відомостей вагонівимірювальної технікі, що потрібно встановити;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_measure_equip_num_w

Встановлення відомостей вагонівимірювальної техніки у кодуванні UTF-16.

Об'явлення:

```
GohubBool gohub_document_set_measure_equip_num_w(
      GohubDocument* document, const GohubWChar* value);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`value`

значення відомостей вагонівимірювальної технікі, що потрібно встановити (у кодуванні UTF-16);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_verified_empty_weight_for_wagon

Отримання уточненої ваги тари вагона

Об'явлення:

```
int gohub_document_ get_verified_empty_weight_for_wagon (
      GohubDocument* doc, int wagonIndex);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`wagonIndex`

Індекс вагона уточнену вагу тари якого необхідно отримати. Індексація вагонів починається з нуля;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_verified_empty_weight_for_wagon

Встановлення уточненої ваги тари вагона

Об'явлення:

```
GohubBool gohub_document_ set_verified_empty_weight_for_wagon (
      GohubDocument* doc, int wagonIndex, int weight);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`wagonIndex`

Індекс вагона уточнену вагу тари якого необхідно змінити. Індексація вагонів починається з нуля;

`weight`

Уточнена вага тари що необхідно встановити

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_warrant_type

Дозволяє визначити тип підстав для отримання вантажу

Об'явлення:

```
int gohub_document_warrant_type(GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - -1. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_warrant_type

Дозволяє задати тип підстав для отримання вантажу

Об'явлення:

```
GohubBool gohub_document_set_warrant_type(GohubDocument* document,
int val);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`val`

Тип підстав для отримання вантажу (0 - довіреність, 1 - наказ)

Результат:

У разі успіху - true . У разі помилки - false . Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_business_unit_num

Отримання номера філії ПрАТ УЗ

Об'явлення:

```
const char* gohub_document_ business_unit_num(
```

```
        GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_business_unit_num_w

Отримання номера філії ПрАТ УЗ у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_business_unit_num_w(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_business_unit_num

Встановлення номера філії ПрАТ УЗ.

Об'явлення:

```
GohubBool gohub_document_set_business_unit_num(
      GohubDocument* document, const char* value);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`value`

значення номера філії ПрАТ УЗ, що встановлюється;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_set_business_unit_num_w

Встановлення номера філії ПрАТ УЗ у кодуванні UTF-16.

Об'явлення:

```
GohubBool gohub_document_set_business_unit_num_w(
      GohubDocument* document, const GohubWChar* value);
```

Параметри:

`document`

Об'єкт документа (GohubDocument), що відправляється на сервер;

value

значення номера філії ПрАТ УЗ, що встановлюється (у кодуванні UTF-16);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_foreign_not_accept

Отримання відмітки повернення неприйнятих прикордонними станціями іноземних залізниць вагонів на територію України

Об'явлення:

```
bool gohub_document_get_foreign_not_accept(GohubDocument* document);
```

Параметри:

document

Об'єкт документа (GohubDocument)

Результат:

У разі успіху - значення true або false . У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_document

Відправлення документа (GohubDocument) на сервер АС «Клієнт УЗ» і далі - товарному касиру.

Об'явлення:

```
GohubBool gohub_send_document(
      GohubConnection* connection,
      GohubDocument* document);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

document

Об'єкт документа (GohubDocument), що відправляється на сервер;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_document

Збереження документа (GohubDocument) на диск, за вказаним шляхом та зі зазначеним кодуванням.

Об'явлення:

```
GohubBool gohub_save_document(
      GohubDocument* document,
      const char* path,
      int codePage);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`path`

шлях, куди зберігається файл;

`codePage`

числове позначення сторінки кодування, в якій зберігається файл;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_document_w

Збереження документа (`GohubDocument`) на диск, за вказаним шляхом та зі зазначеним кодуванням.

Об'явлення:

```
GohubBool gohub_save_document_w(
      GohubDocument* document,
      const GohubWChar* path,
      int codePage);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), що відправляється на сервер;

`path`

шлях, куди зберігається файл (у кодуванні UTF-16);

`codePage`

числове позначення сторінки кодування, в якій зберігається файл;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_document_data

Збереження електронних даних документа (`GohubDocument`) на диск, за вказаним шляхом та зі зазначеним кодуванням.

Об'явлення:

```
GohubBool gohub_save_document_data(
      GohubDocument* document,
      const char* path,
      int codePage,
      int epdVersion);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

`path`

шлях, куди зберігається файл;

`codePage`

числове позначення сторінки кодування, в якій зберігається файл;

`epdVersion`

версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_document_data_w

Збереження електронних даних документа (`GohubDocument`) на диск, за вказаним шляхом та зі зазначеним кодуванням.

Об'явлення:

```
GohubBool gohub_save_document_data_w(
      GohubDocument* document,
      const GohubWChar* path,
      int codePage,
      int epdVersion);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

`path`

шлях, куди зберігається файл (у кодуванні UTF-16);

`codePage`

числове позначення сторінки кодування, в якій зберігається файл;

`epdVersion`

версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_document

Закриття документа (`GohubDocument`).

Об'явлення:

```
GohubBool gohub_close_document(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reclaim_document

Відкликання документа.

Об'явлення:

```
GohubBool gohub_reclaim_document(
      GohubConnection* connection,
      const char* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа, що запитується;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reclaim_document_w

Відкликання документа.

Об'явлення:

```
GohubBool gohub_reclaim_document_w(
      GohubConnection* connection,
      const GohubWChar* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа (в кодуванні UTF-16), який запитується;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_document

Видалення документа.

Об'явлення:

```
GohubBool gohub_delete_document(
      GohubConnection* connection,
      const char* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа, що запитується.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_document_w

Видалення документа.

Об'явлення:

```
GohubBool gohub_delete_document_w(
      GohubConnection* connection,
      const GohubWChar* char documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

унікальний ідентифікатор документа (в кодуванні UTF-16), який запитується.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_received_document

Відправлення документа по прибуттю (`GohubDocument)` на сервер АС «Клієнт УЗ» і далі - товарному касиру.

Об'явлення:

```
GohubBool gohub_send_document(
      GohubConnection* connection,
      GohubDocument* document
      const char* documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`document`

Об'єкт документа (`GohubDocument)`, що відправляється на сервер;

`documentId`

унікальний ідентифікатор документа (у кодуванні UTF-16), що відправляється.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_received_document_w

Відправлення документа по прибуттю (`GohubDocument)` на сервер АС «Клієнт УЗ» і далі - товарному касиру.

Об'явлення:

```
GohubBool gohub_send_document(
      GohubConnection* connection,
      GohubDocument* document
      const GohubWChar* char documentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`document`

Об'єкт документа (`GohubDocument)`, що відправляється на сервер;

`documentId`

унікальний ідентифікатор документа (у кодуванні UTF-16), що відправляється.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_otpr

Збереження актуального тексту перевізного документа (`GohubDocument)` у форматі xml на диск, за вказаним шляхом.

Об'явлення:

```
GohubBool gohub_document_get_otpr(
      GohubDocument* document,
      const char* path);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument)`, електронні дані якого запитуються;

`path`

шлях, куди зберігається файл;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_otpr_w

Збереження актуального тексту перевізного документа (`GohubDocument)` у форматі xml на диск, за вказаним шляхом (у кодуванні UTF-8).

Об'явлення:

```
GohubBool gohub_document_get_otpr_w(
      GohubDocument* document,
      const GohubWChar* path);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

`path`

шлях, куди зберігається файл (у кодуванні UTF-16);

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_otpr_string

Дозволяє отримати актуальний текст перевізного документа (`GohubDocument`) у вигляді рядку.

Об'явлення:

```
const char* gohub_document_get_otpr_string(GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_get_otpr_string_w

Дозволяє отримати актуальний текст перевізного документа (`GohubDocument`) у вигляді рядку (у кодуванні UTF-8).

Об'явлення:

```
GohubBool const GohubWChar* gohub_document_get_otpr_string_w(
GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_warning

Дозволяє отримати актуальний текст попереджень перевізного документа (`GohubDocument`) у вигляді рядку після відправки документа.

Об'явлення:

```
const char* gohub_document_warning(GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_warning_w

Дозволяє отримати актуальний текст попереджень перевізного документа (`GohubDocument`) у вигляді рядку після відправки документа (у кодуванні UTF-8).

Об'явлення:

```
const GohubWChar* gohub_document_warning_w(GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`), електронні дані якого запитуються;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_document_printable_form

Запитати друковану форму документа за його ID та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_document_printable_form(
      GohubConnection* connection,
      const char* documentId,
      const char* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

Унікальний ідентифікатор документа;

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_document_printable_form_w

Запитати друковану форму документа за його ID (у кодуванні UTF-16) та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_document_printable_form_w(
      GohubConnection* connection,
      const GohubWChar* documentId,
      const GohubWChar* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

Унікальний ідентифікатор документа (у кодуванні UTF-16);

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_ query_and_save_document_archive

Запитати архів з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД.

Об'явлення:

```
GohubBool gohub_query_and_save_document_archive(GohubConnection*
connection, onst char* documentId, const char* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

Унікальний ідентифікатор документа (у кодуванні UTF-16);

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

отримати за допомогою функції gohub_last_error і ін.

gohub_ query_and_save_document_archive _w

Запитати архів з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД (в UTF-16).

Об'явлення:

```
GohubBool gohub_query_and_save_document_archive _w(GohubConnection*
connection, const GohubWChar* documentId, const GohubWChar* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentId`

Унікальний ідентифікатор документа (у кодуванні UTF-16);

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.4. Робота зі супровідними документами

`gohub_load_attachment`	Завантажити супровідний документ, для бланка ГУ та ЦІМ
`gohub_load_attachment_with_user_data`	Завантаження супровідного документа (для бланків ГУ та ЦІМ) з електронними даними користувача.
`gohub_load_attachment_w`	Завантажити супровідний документ для бланка ГУ та ЦІМ `(в UTF-16)`
`gohub_load_attachment_with_user_data_w`	Завантаження супровідного документа (для бланків ГУ та ЦІМ) з електронними даними користувача `(в UTF-16)`
`gohub_load_smgs_attachment`	Завантажити супровідний документ для бланка СМГС та ЦІМ/СМГС
`gohub_load_smgs_attachment_with_user_data`	Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС) з електронними даними користувача.
`gohub_load_smgs_attachment_w`	Завантажити супровідний документ для бланка СМГС та ЦІМ/СМГС `(в UTF-16)`
`gohub_load_smgs_attachment_with_user_data_w`	Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС) з електронними даними користувача. `(в UTF-16)`
`gohub_send_attachment`	Відправити супровідний документ
`gohub_query_attachment`	Запитати супровідний документ
`gohub_query_attachment_w`	Запитати супровідний документ `(в UTF-16)`
`gohub_save_attachment`	Зберегти супровідний документ
`gohub_save_attachment_with_user_data`	Збереження електронних даних супровідного документа користувача
`gohub_save_attachment_w`	Зберегти супровідний документ `(в UTF-16)`
`gohub_save_attachment_with_user_data_w`	Збереження електронних даних супровідного документа користувача (в UTF-16).
`gohub_delete_attachment`	Видалити супровідний документ
`gohub_delete_attachment_w`	Видалити супровідний документ `(в UTF-16)`
`gohub_close_attachment`	Закрити супровідний документ
`gohub_attachment_id`	Запитати ID супровідного документа
`gohub_attachment_id_w`	Запитати ID супровідного документа `(в UTF-16)`
`gohub_attachment_type_code`	Запитати код типу супровідного документа
`gohub_attachment_type_code_w`	Запитати код типу супровідного документа `(в`

	UTF-16)
gohub_attachment_name	Запитати ім'я супровідного документа
gohub_attachment_name_w	Запитати ім'я супровідного документа (в UTF-16)
gohub_attachment_reg_number	Запитати реєстраційний номер супровідного документа
gohub_attachment_reg_number_w	Запитати реєстраційний номер супровідного документа (в UTF-16)
gohub_attachment_reg_date	Запитати дату реєстрації супровідного документа
gohub_attachment_reg_date_w	Запитати дату реєстрації супровідного документа (в UTF-16)
gohub_attachment_valid_from	Запитати дату початку дії супровідного документа
gohub_attachment_valid_from_w	Запитати дату початку дії супровідного документа (в UTF-16)
gohub_attachment_valid_to	Запитати дату припинення дії супровідного документа
gohub_attachment_valid_to_w	Запитати дату припинення дії супровідного документа (в UTF-16)
gohub_attachment_description	Запитати дескриптор супровідного документа
gohub_attachment_description_w	Запитати дескриптор супровідного документа в UTF-16)
gohub_attachment_count	Запитати кількість доданих до перевізного документу супровідних документів
gohub_attachment_id_by_index	Запитати ID супровідного документа, за його індексом у переліку доданих до перевізного документу супровідних документів
gohub_attachment_id_by_index_w	Запитати ID супровідного документа, за його індексом у переліку доданих до перевізного документу супровідних документів (в UTF-16)

gohub_load_attachment

Завантаження супровідного документа (для бланків ГУ та ЦІМ).

Об'явлення:

```
GohubAttachment* gohub_load_attachment(
      const char* typeCode,
      const char* name,
      const char* regNumber,
      const char* regDate,
      const char* validFrom,
      const char* validTo,
      const char* path);
```

Параметри:

typeCode

Код типу супровідного документа;

name

ім'я супровідного документа;

regNumber

реєстраційний номер супровідного документа;

regDate

дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’);

`validFrom`

дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`validTo`

дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`path`

шлях до файлу з відсканованим текстом супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_attachment_with_user_data

Завантаження супровідного документа (для бланків ГУ та ЦІМ) з електронними даними користувача.

Об'явлення:

```
GohubAttachment* gohub_load_attachment_with_user_data(
      const char* typeCode,
      const char* name,
      const char* regNumber,
      const char* regDate,
      const char* validFrom,
      const char* validTo,
      const char* path ,
      const char* pathUserData);
```

Параметри:

`typeCode`

Код типу супровідного документа;

`name`

ім'я супровідного документа;

`regNumber`

реєстраційний номер супровідного документа;

`regDate`

дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’);

`validFrom`

дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`validTo`

дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`path`

шлях до файлу з відсканованим текстом супровідного документа.

`pathUserData`

шлях до файлу з електронними даними супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_attachment_w

Завантаження супровідного документа (для бланків ГУ та ЦІМ) `(в UTF-16)`.

Об'явлення:

```
GohubAttachment* gohub_load_attachment_w(
      const GohubWChar* typeCode,
      const GohubWChar* name,
      const GohubWChar* regNumber,
      const GohubWChar* regDate,
      const GohubWChar* validFrom,
      const GohubWChar* validTo,
      const GohubWChar* path);
```

Параметри:

`typeCode`

Код типу супровідного документа у кодуванні UTF-16;

`name`

Ім'я супровідного документа у кодуванні UTF-16;

`regNumber`

Реєстраційний номер супровідного документа у кодуванні UTF-16;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`path`

Шлях до файлу з відсканованим текстом супровідного документа у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_attachment_with_user_data_w

Завантаження супровідного документа (для бланків ГУ та ЦІМ) з електронними даними користувача (в UTF-16).

Об'явлення:

```
GohubAttachment* gohub_load_attachment_with_user_data_w(
      const GohubWChar* typeCode,
      const GohubWChar* name,
      const GohubWChar* regNumber,
      const GohubWChar* regDate,
      const GohubWChar* validFrom,
      const GohubWChar* validTo,
      const GohubWChar* path,
      const GohubWChar* pathUserData);
```

Параметри:

typeCode

Код типу супровідного документа у кодуванні UTF-16;

name

ім'я супровідного документа у кодуванні UTF-16;

regNumber

реєстраційний номер супровідного документа у кодуванні UTF-16;

regDate

дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

validFrom

дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

validTo

дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

path

шлях до файлу з відсканованим текстом супровідного документа у кодуванні UTF-16.

pathUserData

шлях до файлу з електронними даними супровідного документа у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_smgs_attachment

Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС).

Об'явлення:

```
GohubAttachment* gohub_load_smgs_attachment(
      const char* smgsTypeCode,
      const char* name,
      const char* regNumber,
      const char* regDate,
      const char* validFrom,
      const char* validTo,
      const char* path);
```

Параметри:

`smgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС;

`name`

Ім'я супровідного документа;

`regNumber`

Реєстраційний номер супровідного документа;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’);

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`path`

Шлях до файлу з відсканованим текстом супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_smgs_attachment_with_user_data

Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС) з електронними даними користувача.

Об'явлення:

```
GohubAttachment* gohub_load_smgs_attachment_with_user_data(
      const char* smgsTypeCode,
      const char* name,
      const char* regNumber,
      const char* regDate,
      const char* validFrom,
      const char* validTo,
      const char* path,
      const char* pathUserData);
```

Параметри:

`smgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС;

`name`

Ім'я супровідного документа;

`regNumber`

Реєстраційний номер супровідного документа;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’);

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`path`

Шлях до файлу з відсканованим текстом супровідного документа.

`pathUserData`

Шлях до файлу з електронними даними супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_smgs_attachment_w

Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС) в UTF-16.

Об'явлення:

```
GohubAttachment* gohub_load_smgs_attachment_w(
      const GohubWChar* smgsTypeCode,
      const GohubWChar* name,
      const GohubWChar* regNumber,
      const GohubWChar* regDate,
      const GohubWChar* validFrom,
      const GohubWChar* validTo,
      const GohubWChar* path);
```

Параметри:

`smgsTypeCode`

Код типу супровідного документа у кодуванні UTF-16 згідно інформаційного керівництва СМГС;

`name`

Ім'я супровідного документа у кодуванні UTF-16;

`regNumber`

Реєстраційний номер супровідного документа у кодуванні UTF-16;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`path`

Шлях до файлу з відсканованим текстом супровідного документа у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_smgs_attachment_with_user_data_w

Завантаження супровідного документа (для бланків СМГС та ЦІМ/СМГС) з електронними даними користувача (в UTF-16).

Об'явлення:

```
GohubAttachment* gohub_load_smgs_attachment_with_user_data_w(
      const GohubWChar* smgsTypeCode,
      const GohubWChar* name,
      const GohubWChar* regNumber,
      const GohubWChar* regDate,
      const GohubWChar* validFrom,
      const GohubWChar* validTo,
      const GohubWChar* path,
      const GohubWChar* pathUserData);
```

Параметри:

`smgsTypeCode`

Код типу супровідного документа у кодуванні UTF-16 згідно інформаційного керівництва СМГС;

`name`

Ім'я супровідного документа у кодуванні UTF-16;

`regNumber`

Реєстраційний номер супровідного документа у кодуванні UTF-16;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`path`

Шлях до файлу з відсканованим текстом супровідного документа у кодуванні UTF-16.

`pathUserData`

Шлях до файлу з електронними даними супровідного документа у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_attachment

Відправлення супровідного документа.

Об'явлення:

```
GohubBool gohub_send_attachment(
      GohubConnection* connection,
      GohubAttachment* attachment);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachment`

Об'єкт документа (`GohubAttachment`), що відправляється на сервер.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_attachment

Запит супровідного документа.

Об'явлення:

```
GohubAttachment* gohub_query_attachment(
      GohubConnection* connection,
      const char* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

унікальний ідентифікатор супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_attachment_w

Запит супровідного документа.

Об'явлення:

```
GohubAttachment* gohub_query_attachment_w(
      GohubConnection* connection,
      const GohubWChar* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

унікальний ідентифікатор супровідного документа (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_attachment_with_user_data

Запит супровідного документа з електронними даними користувача.

Об'явлення:

```
GohubAttachment* gohub_query_attachment_with_user_data(
      GohubConnection* connection,
      const char* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

унікальний ідентифікатор супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_attachment_with_user_data_w

Запит супровідного документа.

Об'явлення:

```
GohubAttachment* gohub_query_attachment_with_user_data_w(
      GohubConnection* connection,
      const GohubWChar* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

унікальний ідентифікатор супровідного документа (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_attachment

Збереження супровідного документа.

Об'явлення:

```
GohubBool gohub_save_attachment(
      GohubAttachment* attachment,
      const char* path);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`), що зберігається на диск;

`path`

шлях, куди зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_attachment_with_user_data

Збереження електронних даних супровідного документа користувача.

Об'явлення:

```
GohubBool gohub_save_attachment_with_user_data(
      GohubAttachment* attachment,
      const char* path);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`), електронні дані якого зберігаються на диск;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_attachment_w

Збереження супровідного документа (в UTF-16).

Об'явлення:

```
GohubBool gohub_save_attachment_w(
      GohubAttachment* attachment,
      const GohubWChar* path);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_attachment_with_user_data_w

Збереження електронних даних супровідного документа користувача (в UTF-16).

Об'явлення:

```
GohubBool gohub_save_attachment_with_user_data_w(
      GohubAttachment* attachment,
      const GohubWChar* path);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`), електронні дані якого зберігаються на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_attachment

Видалення супровідного документа.

Об'явлення:

```
GohubBool gohub_delete_attachment(
      GohubConnection* connection,
      const char* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

Унікальний ідентифікатор супровідного документа, що видаляється.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_attachment_w

Запит супровідного документа.

Об'явлення:

```
GohubBool gohub_delete_attachment_w(
      GohubConnection* connection,
      const GohubWChar* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

унікальний ідентифікатор супровідного документа (у кодуванні UTF-16), що видаляється.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_attachment

Закриття супровідного документа.

Об'явлення:

```
GohubBool gohub_close_attachment(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_id

Запит ID супровідного документа.

Об'явлення:

```
const char* gohub_attachment_id(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_id_w

Запит ID супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_id_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_type_code

Запит кода типу супровідного документа.

Об'явлення:

```
const char* gohub_attachment_type_code(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_type_code_w

Запит кода типу супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_type_code_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_name

Запит імені супровідного документа.

Об'явлення:

```
const char* gohub_attachment_name(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_name_w

Запит імені супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_name_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_reg_number

Запит реєстраційного номера супровідного документа.

Об'явлення:

```
const char* gohub_attachment_reg_number(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_reg_number_w

Запит реєстраційного номера супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_reg_number_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_reg_date

Запит дати реєстрації супровідного документа.

Об'явлення:

```
const char* gohub_attachment_reg_date(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_reg_date_w

Запит дати реєстрації супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_reg_date_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_valid_from

Запит дати початку дії супровідного документа.

Об'явлення:

```
const char* gohub_attachment_valid_from(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_valid_from_w

Запит дати початку дії супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_valid_from_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_valid_to

Запит дати припинення дії супровідного документа.

Об'явлення:

```
const char* gohub_attachment_valid_to(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_valid_to_w

Запит дати припинення супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_valid_to_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_description

Запит дескриптора супровідного документа. Повертається рядок що містить назву типу документа, його реєстраційного номера та дати.

Об'явлення:

```
const char* gohub_attachment_description(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_description_w

Запит дескриптора супровідного документа. Повертається рядок що містить назву типу документа, його реєстраційного номера та дати (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_description_w(
      GohubAttachment* attachment);
```

Параметри:

`attachment`

Об'єкт документа (`GohubAttachment`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_count

Запитати кількість доданих супровідних документів до перевізного документу.

Об'явлення:

```
int gohub_attachment_count(
      GohubDocument* document);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`).

Результат:

У разі успіху - не негативна число. У разі помилки - «-1». Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_id_by_index

Запитати ID супровідного документа за його індексом у переліку доданих супровідних документів до перевізного документу.

Об'явлення:

```
const char* gohub_attachment_id_by_index(
      GohubDocument* document,
      int attachmentIndex);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`);

`attachmentIndex`

Індекс документа у переліку доданих до перевізного документу.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_attachment_id_by_index_w

Запитати ID супровідного документа за його індексом у переліку доданих супровідних документів до перевізного документу (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_attachment_id_by_index_w(
```

```
GohubDocument* document,
int attachmentIndex);
```

Параметри:

`document`

Об'єкт документа (`GohubDocument`);

`attachmentIndex`

Індекс документа у переліку доданих до перевізного документу.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.5. Робота з електронними даними і пакетами попереднього інформування (ПІ)

`gohub_load_edata`	Завантажити електронні дані ПІ
`gohub_load_edata_w`	Завантажити електронні дані ПІ (в UTF-16)
`gohub_load_edata_simple`	Завантажити електронні дані ПІ без супровідного документа
`gohub_load_edata_simple_w`	Завантажити електронні дані ПІ без супровідного документа (в UTF-16)
`gohub_send_edata`	Відправити електронні дані ПІ
`gohub_update_edata`	Оновити наявні електронні дані ПІ
`gohub_query_edata`	Запитати електронні дані ПІ
`gohub_query_edata_w`	Запитати електронні дані ПІ (в UTF-16)
`gohub_query_next_edata`	Запитати електронні дані ПІ за ревізією
`gohub_save_edata`	Зберегти електронні дані ПІ
`gohub_save_edata_w`	Зберегти електронні дані ПІ (в UTF-16)
`gohub_edata_load_data`	Завантажити дані з файлу у наявні електронні дані ПІ
`gohub_edata_load_data_w`	Завантажити дані з файлу у наявні електронні дані ПІ (в UTF-16)
`gohub_close_edata`	Закрити електронні дані ПІ
`gohub_edata_id`	Запитати ID електронних даних ПІ
`gohub_edata_id_w`	Запитати ID електронних даних ПІ (в UTF-16)
`gohub_edata_revision`	Запитати ревізію електронних даних ПІ
`gohub_edata_revision_date`	Запитати дату ревізії електронних даних ПІ
`gohub_edata_revision_date_w`	Запитати дату ревізії електронних даних ПІ (в UTF-16)
`gohub_edata_doc_type`	Запитати код типу електронних даних ПІ (190 рахунок-фактура, 320 пакувальний лист)
`gohub_edata_status`	Запитати статус електронних даних ПІ
`gohub_edata_version`	Запитати версію електронних даних ПІ
`gohub_edata_version_w`	Запитати версію електронних даних ПІ (в UTF-16)
`gohub_edata_attachment_id`	Запитати ID супровідного документа електронних даних ПІ
`gohub_edata_attachment_id_w`	Запитати ID супровідного документа електронних даних ПІ (в UTF-16)
`gohub_query_pi_package`	Запитати пакет ПІ
`gohub_query_pi_package_w`	Запитати пакет ПІ (в UTF-16)
`gohub_query_next_pi_package`	Запитати пакет ПІ за ревізією
`gohub_save_pi_package`	Зберегти пакет ПІ
`gohub_save_pi_package_w`	Зберегти пакет ПІ (в UTF-16)
`gohub_close_pi_package`	Закрити пакет ПІ
`gohub_pi_package_id`	Запитати ID пакета ПІ
`gohub_pi_package_id_w`	Запитати ID пакета ПІ (в UTF-16)
`gohub_pi_package_revision`	Запитати ревізію пакета ПІ
`gohub_pi_package_revision_date`	Запитати дату ревізії пакета ПІ
`gohub_pi_package_revision_date_w`	Запитати дату ревізії пакета ПІ (в UTF-16)
`gohub_pi_package_consignment_id`	Запитати ID перевізного документа на який посилається пакет ПІ
`gohub_pi_package_consignment_id_w`	Запитати ID перевізного документа на який посилається пакет

	ПІ (в UTF-16)
gohub_pi_package_status	Запитати статус пакета ПІ
gohub_pi_package_pipacktoed_count	Запитати кількість елементів GohubPiPackageToEData що містяться у пакеті ПІ
gohub_pi_package_pipacktoed	Запитати елемент GohubPiPackageToEData у пакеті ПІ за індексом
gohub_pipacktoed_id	Запитати ID про'єкту GohubPiPackageToEData
gohub_pipacktoed_id_w	Запитати ID про'єкту GohubPiPackageToEData (в UTF-16)
gohub_pipacktoed_edata_id	Запитати ID електронних даних в про'єкті GohubPiPackageToEData
gohub_pipacktoed_edata_id_w	Запитати ID електронних даних в про'єкті GohubPiPackageToEData (в UTF-16)
gohub_pipacktoed_pi_package_id	Запитати ID пакета ПІ в про'єкті GohubPiPackageToEData
gohub_pipacktoed_pi_package_id_w	Запитати ID пакета ПІ в про'єкті GohubPiPackageToEData (в UTF-16)
gohub_pipacktoed_note	Запитати примітки в про'єкті GohubPiPackageToEData
gohub_pipacktoed_note_w	Запитати примітки в про'єкті GohubPiPackageToEData (в UTF-16)
gohub_pipacktoed_status	Запитати статус про'єкту GohubPiPackageToEData
gohub_pipacktoed_edata_version	Запитати версію електронних даних в про'єкті GohubPiPackageToEData
gohub_pipacktoed_edata_version_w	Запитати версію електронних даних в про'єкті GohubPiPackageToEData (в UTF-16)
gohub_add_edata_to_pi_package	Додати електронні дані до пакету ПІ
gohub_add_edata_to_pi_package_w	Додати електронні дані до пакету ПІ (в UTF-16)

gohub_load_edata

Завантаження електронних даних ПІ.

Об'явлення:

```
GohubEData* gohub_load_edata(
      unsigned int attachmentSmgsTypeCode,
      const char* xmlPath,
      const char* name,
      const char* regNumber,
      const char* regDate,
      const char* validFrom,
      const char* validTo,
      const char* pdfPath);
```

Параметри:

`attachmentSmgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС. Можна вибрати зі значень 325 Рахунок-проформа (інвойс), 380 (інвойс) рахунок-фактура, 935 Рахунок-фактура;

`xmlPath`

Шлях до xml-файлу що містять електронні дані ПІ;

`name`

Ім'я супровідного документа;

`regNumber`

Реєстраційний номер супровідного документа;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’);

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’);

`pdfPath`

Шлях до pdf-файлу з відсканованим текстом супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_edata_w

Завантаження електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubEData* gohub_load_edata_w(
      unsigned int attachmentSmgsTypeCode,
      const GohubWChar* xmlPath,
      const GohubWChar* name,
      const GohubWChar* regNumber,
      const GohubWChar* regDate,
      const GohubWChar* validFrom,
      const GohubWChar* validTo,
      const GohubWChar* pdfPath);
```

Параметри:

`attachmentSmgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС. Можна вибрати зі значень 325 Рахунок-проформа (інвойс), 380 (інвойс) рахунок-фактура, 935 Рахунок-фактура;

`xmlPath`

Шлях до xml-файлу у кодуванні UTF-16 що містять електронні дані ПІ;

`name`

Ім'я супровідного документа у кодуванні UTF-16;

`regNumber`

Реєстраційний номер супровідного документа у кодуванні UTF-16;

`regDate`

Дата реєстрації супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validFrom`

Дата початку терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`validTo`

Дата припинення терміну дії супровідного документа (формат дати ‘dd.MM.yyyy’) у кодуванні UTF-16;

`pdfPath`

Шлях до pdf-файлу з відсканованим текстом супровідного документа у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_edata_simple

Завантажити електронні дані ПІ без супровідного документа.

Об'явлення:

```
GohubEData* gohub_load_edata(
      unsigned int attachmentSmgsTypeCode,
      const char* xmlPath
      );
```

Параметри:

`attachmentSmgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС. Можна вибрати зі значень 325 Рахунок-проформа (інвойс), 380 (інвойс) рахунок-фактура, 935 Рахунок-фактура;

`xmlPath`

Шлях до xml-файлу що містять електронні дані ПІ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_edata_simple_w

Завантажити електронні дані ПІ без супровідного документа (у кодуванні UTF-16).

Об'явлення:

```
GohubEData* gohub_load_edata_w(
      unsigned int attachmentSmgsTypeCode,
      const GohubWChar* xmlPath);
```

Параметри:

`attachmentSmgsTypeCode`

Код типу супровідного документа згідно інформаційного керівництва СМГС. Можна вибрати зі значень 325 Рахунок-проформа (інвойс), 380 (інвойс) рахунок-фактура, 935 Рахунок-фактура;

`xmlPath`

Шлях до xml-файлу у кодуванні UTF-16 що містять електронні дані ПІ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_edata

Відправлення електронних даних ПІ.

Об'явлення:

```
GohubBool gohub_send_edata(
      GohubConnection* connection,
      GohubEData* eData);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що відправляється на сервер.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_update_edata

Оновити наявні електронні дані ПІ.

Об'явлення:

```
GohubBool gohub_update_edata(
      GohubConnection* connection,
      GohubEData* eData);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що буде оновлюватись на сервері.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_edata

Запит електронних даних ПІ.

Об'явлення:

```
GohubEData* gohub_query_edata(
      GohubConnection* connection,
      const char* eDataId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eDataId`

Унікальний ідентифікатор електронних даних ПІ.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_edata_w

Запит електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubEData* gohub_query_edata_w(
      GohubConnection* connection,
      const GohubWChar* eDataId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eDataId`

Унікальний ідентифікатор електронних даних ПІ (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_edata_for_attachment

Запит електронних даних ПІ за ідентифікатором супровідного документа.

Об'явлення:

```
GohubEData* gohub_query_edata(
      GohubConnection* connection,
      const char* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

Унікальний ідентифікатор супровідного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_edata_for_attachment_w

Запит електронних даних ПІ (у кодуванні UTF-16) за ідентифікатором супровідного документа.

Об'явлення:

```
GohubEData* gohub_query_edata_w(
      GohubConnection* connection,
      const GohubWChar* attachmentId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`attachmentId`

Унікальний ідентифікатор супровідного документа (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_edata

Запит електронних даних ПІ за ревізією.

Об'явлення:

```
GohubEData* gohub_query_next_edata(
      GohubConnection* connection,
      unsigned __int64 lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія, з якої починати пошук електронних даних ПІ.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_edata

Збереження електронних даних ПІ.

Об'явлення:

```
GohubBool gohub_save_edata(
      GohubEData* eData,
      const char* path);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що зберігається на диск;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_edata_w

Збереження електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_save_edata_w(
      GohubEData* eData,
      const GohubWChar* path);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_load_data

Завантажити дані з файлу у наявні електронні дані ПІ.

Об'явлення:

```
GohubBool gohub_save_edata(
      GohubEData* eData,
      const char* path);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що зберігається на диск;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_load_data_w

Завантажити дані з файлу у наявні електронні дані ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_save_edata_w(
      GohubEData* eData,
      const GohubWChar* path);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_edata

Закриття електронних даних ПІ.

Об'явлення:

```
GohubBool gohub_close_edata(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_id

Запит ID електронних даних ПІ.

Об'явлення:

```
const char* gohub_edata_id(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_id_w

Запит ID електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_edata_id_w(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_revision

Запит номера ревізії електронних даних ПІ.

Об'явлення:

```
unsigned __int64 gohub_edata_revision(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_revision_date

Запит дати ревізії електронних даних ПІ.

Об'явлення:

```
const char* gohub_edata_revision_date(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_revision_date_w

Запит дати ревізії електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_edata_revision_date_w(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_doc_type

Запит типу електронних даних ПІ.

Об'явлення:

```
unsigned int gohub_edata_doc_type(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_status

Запит статусу електронних даних ПІ.

Об'явлення:

```
int gohub_edata_status(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_version

Запит версії електронних даних ПІ.

Об'явлення:

```
const char* gohub_edata_version(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_version_w

Запит версії електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_edata_version_w(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_attachment_id

Запит ID супровідного документа електронних даних ПІ.

Об'явлення:

```
const char* gohub_edata_attachment_id(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_edata_attachment_id_w

Запит ID супровідного документа електронних даних ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_edata_attachment_id_w(
      GohubEData* eData);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_pi_package

Запит пакет ПІ.

Об'явлення:

```
GohubPiPackage* gohub_query_pi_package(
      GohubConnection* connection,
      const char* piPackageId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`piPackageId`

Унікальний ідентифікатор електронних даних ПІ.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_pi_package_w

Запит пакет ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubPiPackage* gohub_query_pi_package_w(
      GohubConnection* connection,
      const GohubWChar* piPackageId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`piPackageId`

Унікальний ідентифікатор супровідного документа (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_pi_package

Запит пакета ПІ за ревізією.

Об'явлення:

```
GohubPiPackage* gohub_query_next_pi_package(
      GohubConnection* connection,
      unsigned __int64 lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія, з якої починати пошук пакета ПІ.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_pi_package

Збереження пакета ПІ.

Об'явлення:

```
GohubBool gohub_save_pi_package(
      GohubPiPackage* piPackage,
      const char* path);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`), що зберігається на диск;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_pi_package_w

Збереження пакета ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_save_pi_package_w(
      GohubPiPackage* piPackage,
      const GohubWChar* path);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_pi_package

Закриття пакета ПІ.

Об'явлення:

```
GohubBool gohub_close_pi_package(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_id

Запит ID пакета ПІ.

Об'явлення:

```
const char* gohub_pi_package_id(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_id_w

Запит ID пакета ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pi_package_id_w(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_revision

Запит номера ревізії пакета ПІ.

Об'явлення:

```
unsigned __int64 gohub_pi_package_revision(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_revision_date

Запит дати ревізії пакета ПІ.

Об'явлення:

```
const char* gohub_pi_package_revision_date(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_revision_date_w

Запит дати ревізії пакета ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pi_package_revision_date_w(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_status

Запит статусу електронних даних ПІ.

Об'явлення:

```
int gohub_pi_package_status(
      GohubPiPackage* piPackage);
```

Параметри:

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_consignment_id

Запитати ID перевізного документа на який посилається пакет ПІ.

Об'явлення:

```
const char* gohub_pi_package_consignment_id(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_consignment_id_w

Запитати ID перевізного документа на який посилається пакет ПІ (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pi_package_consignment_id_w(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pi_package_pipacktoed_count

Запитати кількість елементів GohubPiPackageToEData що містяться у пакеті ПІ.

Об'явлення:

```
int gohub_pi_package_pipacktoed_count(
      GohubPiPackage* piPackage);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

Результат:

кількість елементів GohubPiPackageToEData що містяться у пакеті ПІ.

gohub_pi_package_pipacktoed

Запит елементу GohubPiPackageToEData у пакеті ПІ за індексом.

Об'явлення:

```
GohubPiPackageToEData* gohub_pi_package_pipacktoed(
      GohubPiPackage* piPackage,
      int index);
```

Параметри:

`piPackage`

Об'єкт пакета ПІ (`GohubPiPackage`).

`index`

Номер індексу. Може мати значення від 0 до gohub_pi_package_pipacktoed_count()-1.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_id

Запитати ID про'єкту GohubPiPackageToEData.

Об'явлення:

```
const char* gohub_pipacktoed_id(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_id_w

Запитати ID про'єкту GohubPiPackageToEData (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pipacktoed_id_w(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_edata_id

Запитати ID електронних даних в про'єкті GohubPiPackageToEData.

Об'явлення:

```
const char* gohub_pipacktoed_edata_id(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_edata_id_w

Запитати ID електронних даних в про'єкті GohubPiPackageToEData (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pipacktoed_edata_id_w(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_pi_package_id

Запитати ID пакета ПІ в про'єкті GohubPiPackageToEData.

Об'явлення:

```
const char* gohub_pipacktoed_pi_package_id(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_pi_package_id_w

Запитати ID пакета ПІ в про'єкті GohubPiPackageToEData (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pipacktoed_pi_package_id_w(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_note

Запитати примітки в про'єкті GohubPiPackageToEData.

Об'явлення:

```
const char* gohub_pipacktoed_note(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_note_w

Запитати Примітки в про'єкті GohubPiPackageToEData (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pipacktoed_note_w(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_edata_version

Запитати версію електронних даних в про'єкті GohubPiPackageToEData.

Об'явлення:

```
const char* gohub_pipacktoed_edata_version(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pipacktoed_edata_version_w

Запитати версію електронних даних в про'єкті GohubPiPackageToEData (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_pipacktoed_edata_version_w(
      GohubPiPackageToEData* piPackageToEData);
```

Параметри:

`piPackageToEData`

Об'єкт `GohubPiPackageToEData`.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_add_edata_to_pi_package

Додати електронні дані до пакету ПІ.

Об'явлення:

```
GohubPiPackage* gohub_add_edata_to_pi_package(
      GohubConnection* connection,
      GohubEData* eData,
      const char* piPackageId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

`piPackageId`

Унікальний ідентифікатор електронних даних ПІ.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_add_edata_to_pi_package_w

Додати електронні дані до пакету ПІ (у кодуванні UTF-16).

Об'явлення:

```
GohubPiPackage* gohub_add_edata_to_pi_package_w(
      GohubConnection* connection,
      GohubEData* eData,
```

```
        const GohubWChar* piPackageId);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`eData`

Об'єкт електронних даних ПІ (`GohubEData`).

`piPackageId`

Унікальний ідентифікатор супровідного документа (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.6. Робота з накопичувальними картками ФДУ-92

gohub_query_fdu92	Запитати накопичувальну картку
gohub_query_fdu92_w	Запитати накопичувальну картку (в UTF-16)
gohub_query_fdu92_by_number	Запитати накопичувальну картку за номером станції та номером накопичувальної картки
gohub_query_fdu92_by_number_w	Запитати накопичувальну картку за номером станції та номером накопичувальної картки (в UTF-16)
gohub_query_next_fdu92	Запитати наступну накопичувальну картку
gohub_create_fdu92	Створити накопичувальну картку
gohub_create_fdu92_w	Створити накопичувальну картку (в UTF-16)
gohub_load_fdu92	Завантажити накопичувальну картку з файлу xml
gohub_load_fdu92_w	Завантажити накопичувальну картку з файлу xml (UTF-16)
gohub_save_fdu92	Зберегти накопичувальну картку
gohub_save_fdu92 _w	Зберегти накопичувальну картку (в UTF-16)
gohub_send_fdu92	Відправити накопичувальну картку
gohub_fdu92_id	Запитати ID накопичувальної картки
gohub_fdu92_id_w	Запитати ID накопичувальної картки (в UTF-16)
gohub_fdu92_status	Запитати статус накопичувальної картки
fdu92_revision	Запитати номер ревізії накопичувальної картки
gohub_fdu92_text	Запитати текст накопичувальної картки
gohub_fdu92_text_w	Запитати текст накопичувальної картки (в UTF-16)
fdu92_size	Запитати передбачуваний розмір накопичувальної картки представлений в поточній кодової сторінці
gohub_fdu92_signer_info	Запитати інформацію про підписантів
gohub_fdu92_signer_info_w	Запитати інформацію про підписантів (в UTF-16)
gohub_fdu92_sign_time	Запитати інформацію про час підписання
gohub_fdu92_sign_time_w	Запитати інформацію про час підписання (в UTF-16)
gohub_fdu92_signer_name_w	Запитати ім'я підписанта (в UTF-16)
gohub_fdu92_has_signature	Перевірити чи підписаний документ
gohub_save_fdu92	Зберегти накопичувальну картку в файл
gohub_save_fdu92_w	Зберегти накопичувальну картку в файл (в UTF-16)
gohub_reject_fdu92	Відкликати накопичувальну картку
gohub_reject_fdu92_w	Відкликати накопичувальну картку (в UTF-16)
gohub_close_fdu92	Закрити накопичувальну картку

gohub_query_and_save_fdu92_printable_form	Запитати друковану форму накопичувальної картки
gohub_query_and_save_fdu92_printable_form_w	Запитати друковану форму накопичувальної картки (в UTF-16)

gohub_query_fdu92

Запитати накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubFdu92* gohub_query_fdu92(
GohubConnection* connection,
const char* fdu92Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_fdu92_w

Запитати накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubFdu92* gohub_query_fdu92_w(
GohubConnection* connection,
const GohubWChar* fdu92Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_fdu92_by_number

Запитати накопичувальну картку за номером станції та номером накопичувальної картки

Об'явлення:

```
GohubFdu92* gohub_query_fdu92_number(
GohubConnection* connection,
const char* registration_esr, const char* registration_num);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`registration_esr`

Номер станції.

`registration_num`

Номер накопичувальної картки.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_fdu92_by_number_w

Запитати накопичувальну картку за номером станції та номером накопичувальної картки (в UTF-16)

Об'явлення:

```
GohubFdu92* gohub_query_fdu92_w(
GohubConnection* connection,
const GohubWChar* registration_esr,
const GohubWChar* registration_num);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`registration_esr`

Номер станції. (у кодуванні UTF-16).

`registration_num`

Номер накопичувальної картки. (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_fdu92

Створити накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubFdu92* gohub_create_fdu92(
const char* content);
```

Параметри:

`content`

Рядок з xml-структурою з вмістом документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_fdu92_w

Створити накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubFdu92* gohub_create_fdu92_w(
const GohubWChar* content);
```

Параметри:

`content`

Рядок з xml-структурою з вмістом документа у кодуванні UTF-16;.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_fdu92

Завантажити накопичувальну картку ФДУ-92 з файлу xml.

Об'явлення:

```
GohubFdu92* gohub_load_fdu92(
const char* id,
const char* path);
```

Параметри:

`id`

Ідентифікатор картки ФДУ-92 на сервері.

`path`

Шлях до файлу.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_fdu92_w

Завантажити накопичувальну картку ФДУ-92 з файлу xml.

Об'явлення:

```
GohubFdu92* gohub_load_fdu92_w(
const GohubWChar* id,
const GohubWChar* path);
```

Параметри:

`id`

Ідентифікатор картки ФДУ-92 на сервері.

`path`

Шлях до файлу у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_fdu92

Відправити накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubBool gohub_send_fdu92(
GohubConnection* connection,
GohubFdu92* fdu92);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect

`fdu92`

Об'єкт документа (`GohubFdu92`).;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_fdu92

Запитати наступну накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubFdu92* gohub_query_next_fdu92(
GohubConnection* connection,
int lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія документа, від якої починається пошук наступної накопичувальної картки.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_id

Запит ID накопичувальної картки ФДУ-92.

Об'явлення:

```
const char* gohub_fdu92_id(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_id_w

Запит ID накопичувальної картки ФДУ-92 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_fdu92_id_w(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_status

Запит статусу накопичувальної картки ФДУ-92.

Об'явлення:

```
GohubFdu92Status gohub_fdu92_status(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`), статус якого запитується;

Результат:

У разі успіху - значення з перерахування GohubFdu92Status . У разі помилки - «-1». Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_revision

Запит номера ревізії накопичувальної картки ФДУ-92.

Об'явлення:

```
int gohub_fdu92_revision(
GohubFdu92* fdu92);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_text

Запит тексту накопичувальної картки ФДУ-92.

Об'явлення:

```
const char* gohub_fdu92_text(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_text_w

Запит тексту накопичувальної картки ФДУ-92 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_fdu92_text_w(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_size

Запит передбачуваного розміру файлу, який вийде при збереженні документа у поточній кодуванні (`GohubFdu92`).

Об'явлення:

```
int gohub_fdu92_size(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`), текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_signer_info

Запит інформації про підписанта накопичувальної картки ФДУ-92.

Об'явлення:

```
const char* gohub_fdu92_signer_info(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_signer_info_w

Запит інформації про підписанта накопичувальної картки ФДУ-92 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_fdu92_signer_info_w(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_sign_time

Запит інформації про підписанта накопичувальної картки ФДУ-92.

Об'явлення:

```
const char* gohub_fdu92_sign_time(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_sign_time_w

Запит інформації про підписанта накопичувальної картки ФДУ-92 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_fdu92_sign_time_w(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_sign_name_w

Запит імені підписанта накопичувальної картки ФДУ-92 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_fdu92_sign_name_w(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_fdu92_has_signature

Перевірити чи підписаний документ.

Об'явлення:

```
const GohubWChar* gohub_fdu92_has_signature(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_fdu92

Збереження накопичувальної картки ФДУ-92.

Об'явлення:

```
GohubBool gohub_save_fdu92(
GohubFdu92* fdu92,
const char* path,
int codePage);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`), що зберігається на диск;

`path`

Шлях, куди зберігається файл;

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_fdu92_w

Збереження накопичувальної картки ФДУ-92.

Об'явлення:

```
GohubBool gohub_save_fdu92_w(
GohubFdu92* fdu92,
const GohubWChar* path,
int codePage);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16);

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reject_fdu92

Відкликати накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubBool gohub_reject_fdu92(
GohubConnection* connection,
const char* fdu92Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки .

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reject_fdu92_w

Відкликати накопичувальну картку ФДУ-92.

Об'явлення:

```
GohubBool gohub_reject_fdu92_w(
GohubConnection* connection,
const GohubWChar* fdu92Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_fdu92

Закриття документа (`GohubFdu92`).

Об'явлення:

```
GohubBool gohub_close_fdu92(
GohubFdu92* fdu92);
```

Параметри:

`fdu92`

Об'єкт документа (`GohubFdu92`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_fdu92_printable_form

Запитати друковану форму накопичувальної картки ФДУ-92 за його ID та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_fdu92_printable_form(
      GohubConnection* connection,
      const char* fdu92Id,
      const char* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки.

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_fdu92_printable_form_w

Запитати друковану форму накопичувальної картки ФДУ-92 за його ID (у кодуванні UTF-16) та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_fdu92_printable_form_w(
      GohubConnection* connection,
      const GohubWChar* fdu92Id,
      const GohubWChar* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92Id`

Унікальний ідентифікатор накопичувальної картки (у кодуванні UTF-16).

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.7. Робота з відомостями про користування вагонами / контейнерами ГУ-46

gohub_query_gu46	Запитати відомість про користування вагонами/контейнерами
gohub_query_gu46_w	Запитати відомість про користування вагонами/контейнерами (в UTF-16)
gohub_query_next_gu46	Запитати наступну відомість про користування вагонами/контейнерами
gohub_create_gu46	Створити відомість про користування вагонами/контейнерами
gohub_create_gu46_w	Створити відомість про користування вагонами/контейнерами
gohub_load_ gu46	Завантажити відомість про користування вагонами/контейнерами з файлу xml
gohub_load_ gu46_w	Завантажити відомість про користування вагонами/контейнерами з файлу xml (UTF-16)
gohub_send_gu46	Відправити відомість про користування вагонами/контейнерами
gohub_gu46_id	Запитати ID відомості про користування вагонами/контейнерами
gohub_gu46_id_w	Запитати ID відомості про користування вагонами/контейнерами (в UTF-16)
gohub_gu46_status	Запитати статус відомості про користування вагонами/контейнерами
gohub_gu46_revision	Запитати номер ревізії відомості про користування вагонами/контейнерами
gohub_gu46_text	Запитати текст відомості про користування вагонами/контейнерами
gohub_gu46_text_w	Запитати текст відомості про користування вагонами/контейнерами (в UTF-16)
gohub_gu46_size	Запитати передбачуваний розмір відомості про користування вагонами/контейнерами представлений в

	поточній кодової сторінці
gohub_gu46_signer_info	Запитати інформацію про підписантів відомості про користування вагонами/контейнерами
gohub_gu46_signer_info_w	Запитати інформацію про підписантів відомості про користування вагонами/контейнерами (в UTF-16)
gohub_gu46_sign_time	Запитати час підписання відомості про користування вагонами/контейнерами
gohub_gu46_sign_time_w	Запитати час підписання відомості про користування вагонами/контейнерами (в UTF-16)
gohub_gu46_signer_name_w	Запитати ім'я підписанта (в UTF-16)
gohub_gu46_has_signature	Перевірити чи підписаний документ
gohub_save_gu46	Зберегти відомість про користування вагонами/контейнерами
gohub_save_gu46_w	Зберегти відомість про користування вагонами/контейнерами (в UTF-16)
gohub_reject_gu46	Відкликати відомість про користування вагонами/контейнерами
gohub_reject_gu46_w	Відкликати відомість про користування вагонами/контейнерами (в UTF-16)
gohub_close_gu46	Закрити відомість про користування вагонами/контейнерами
gohub_query_gu46_by_number	Запитати відомість про користування вагонами/контейнерами за номером станції та номером відомості
gohub_query_gu46_by_number_w	Запитати відомість про користування вагонами/контейнерами за номером станції та номером відомості (в UTF-16)
gohub_query_and_save_gu46_printable_form	Запитати друковану форму відомості про користування вагонами/контейнерами
gohub_query_and_save_gu46_printable_form_w	Запитати друковану форму відомості про користування вагонами/контейнерами (в UTF-16)

gohub_query_gu46

Запитати відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46* gohub_query_gu46(
GohubConnection* connection,
const char* gu46Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu46_w

Запитати відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46* gohub_query_gu46_w(
GohubConnection* connection,
const GohubWChar* gu46Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_gu46

Запитати наступну відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46* gohub_query_next_gu46(
GohubConnection* connection,
int lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія документа, від якої починається пошук наступної відомості про користування вагонами/контейнерами.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_gu46

Створити відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46* gohub_create_gu46(
```

```
const char* gu46Id,
const char* content);
```

Параметри:

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами;

`content`

Рядок з xml-структурою з вмістом документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_create_gu46_w

Створити відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46* gohub_create_gu46_w(
const GohubWChar* gu46Id,
const GohubWChar* content);
```

Параметри:

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами (у кодуванні UTF-16);

`content`

Рядок з xml-структурою з вмістом документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_gu46

Завантажити відомість про користування вагонами/контейнерами ГУ-46 з файлу xml.

Об'явлення:

```
GohubGu46* gohub_load_gu46(
const char* id,
const char* path);
```

Параметри:

`id`

Ідентифікатор відомості ГУ-46 на сервері.

`path`

Шлях до файлу.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_load_gu46_w

Завантажити відомість про користування вагонами/контейнерами ГУ-46 з файлу xml.

Об'явлення:

```
GohubGu46* gohub_load_gu46_w(
const GohubWChar* id,
const GohubWChar* path);
```

Параметри:

`id`

Ідентифікатор відомості ГУ-46 на сервері.

`path`

Шлях до файлу у кодуванні UTF-16.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_send_gu46

Відправити відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubBool gohub_send_gu46(
GohubConnection* connection,
GohubGu46* gu46);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_id

Запит ID відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const char* gohub_gu46_id(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_id_w

Запит ID відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const GohubWChar* gohub_gu46_id_w(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_status

Запит статусу відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubGu46Status gohub_gu46_status(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`), статус якого запитується;

Результат:

У разі успіху - значення з перерахування GohubGu46Status . У разі помилки - «-1». Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_revision

Запит номера ревізії відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
int gohub_gu46_revision(
GohubGu46* gu46);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_text

Запит тексту відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const char* gohub_gu46_text(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_text_w

Запит тексту відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const GohubWChar* gohub_gu46_text_w(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_size

Запит передбачуваного розміру файлу, який вийде при збереженні документа у поточній кодуванні (`GohubGu46`).

Об'явлення:

```
int gohub_gu46_size(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`), текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_signer_info

Запит інформації про підписанта відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const char* gohub_gu46_signer_info(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_signer_info_w

Запит інформації про підписанта відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const GohubWChar* gohub_gu46_signer_info_w(
GohubGu46* gu46);
```

Параметри:

gu46

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_sign_time

Запит часу підписання відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const char* gohub_gu46_sign_time(
GohubGu46* gu46);
```

Параметри:

gu46

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_sign_time_w

Запит часу підписання відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
const GohubWChar* gohub_gu46_sign_time_w(
GohubGu46* gu46);
```

Параметри:

gu46

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_sign_name_w

Запит імені підписанта відомості про користування вагонами/контейнерами ГУ-46 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_gu46_sign_name_w(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu46_has_signature

Перевірити чи підписаний документ.

Об'явлення:

```
const GohubWChar* gohub_gu46_has_signature(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_gu46

Збереження відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubBool gohub_save_gu46(
GohubGu46* gu46,
const char* path,
int codePage);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`), що зберігається на диск;

`path`

шлях, куди зберігається файл;

`codePage`

числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_gu46_w

Збереження відомості про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubBool gohub_save_gu46_w(
GohubGu46* gu46,
const GohubWChar* path,
int codePage);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16);

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reject_gu46

Відкликати відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubBool gohub_reject_gu46(
GohubConnection* connection,
const char* gu46Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_reject_gu46_w

Відкликати відомість про користування вагонами/контейнерами ГУ-46.

Об'явлення:

```
GohubBool gohub_reject_gu46_w(
GohubConnection* connection,
const GohubWChar* gu46Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_gu46

Закриття документа (`GohubGu46`).

Об'явлення:

```
GohubBool gohub_close_gu46(
GohubGu46* gu46);
```

Параметри:

`gu46`

Об'єкт документа (`GohubGu46`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu46_by_number

Запитати відомість про користування вагонами/контейнерами ГУ-46 за номером станції та номером відомості.

Об'явлення:

```
GohubGu46* gohub_query_gu46_by_number(GohubConnection* connection,
const char* registration_esr, const char* registration_num);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

registration_esr

Станція формування відомості.

registration_num

Номер відомості.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu46_by_number_w

Запитати відомість про користування вагонами/контейнерами ГУ-46 за номером станції та номером відомості.

Об'явлення:

```
GohubGu46* gohub_query_gu46_by_number_w(GohubConnection* connection,
const GohubWChar* registration_esr, const GohubWChar*
registration_num);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

registration_esr

Станція формування відомості (у кодуванні UTF-16).

registration_num

Номер відомості (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_gu46_printable_form

Запитати друковану форму відомості про користування вагонами/контейнерами ГУ-46 за його ID та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_gu46_printable_form(
      GohubConnection* connection,
      const char* gu46Id,
      const char* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами.

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_gu46_printable_form_w

Запитати друковану форму відомості про користування вагонами/контейнерами ГУ-46 за його ID (у кодуванні UTF-16) та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_gu46_printable_form_w(
      GohubConnection* connection,
      const GohubWChar* gu46Id,
```

```
        const GohubWChar* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46Id`

Унікальний ідентифікатор відомості про користування вагонами/контейнерами (у кодуванні UTF-16).

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін .

4.4.8. Робота з пам'ятками про подачу / прибирання вагонів та видачу / прийом контейнерів ГУ-45

gohub_query_gu45	Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_query_gu45_w	Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів `(в UTF-16)`
gohub_query_next_gu45	Запитати наступну пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_id	Запитати ID пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_id_w	Запитати ID пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів `(в UTF-16)`
gohub_gu45_status	Запитати статус пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gu45_revision	Запитати номер ревізії пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_text	Запитати текст пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_text_w	Запитати текст пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів `(в UTF-16)`
gohub_gu45_signer_info	Запитати інформацію про підписантів пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_signer_info_w	Запитати інформацію про підписантів пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів `(в UTF-16)`
gohub_gu45_sign_time	Запитати час підписання пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_gu45_sign_time_w	Запитати час підписання пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів `(в UTF-16)`
gohub_gu45_signer_name_w	Запитати ім'я підписанта (в UTF-16)
gohub_gu45_has_signature	Перевірити чи підписаний документ
gu45_size	Запитати передбачуваний розмір пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів

	представлений в поточній кодової сторінці
gohub_save_gu45	Зберегти пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_save_gu45_w	Зберегти пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів (в UTF-16)
gohub_close_gu45	Закрити пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_query_gu45_by_number	Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів за номером станції, номер пам'ятки та дата її формування
gohub_query_gu45_by_number_w	Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів за номером станції, номер пам'ятки та дата її формування (в UTF-16)
gohub_query_and_save_gu45_printable_form	Запитати друковану форму пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів
gohub_query_and_save_gu45_printable_form_w	Запитати друковану форму пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів (в UTF-16)

gohub_query_gu45

Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubGu45* gohub_query_gu45(
GohubConnection* connection,
const char* gu45Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu45Id`

Унікальний ідентифікатор пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu45_w

Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubGu45* gohub_query_gu45_w(
GohubConnection* connection,
const GohubWChar* gu45Id);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu45Id`

Унікальний ідентифікатор пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_gu45

Запитати наступну пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubGu45* gohub_query_next_gu45(
GohubConnection* connection,
int lastRevision);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія документа, від якої починається пошук наступної пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_id

Запит ID пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const char* gohub_gu45_id(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_id_w

Запит ID пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const GohubWChar* gohub_gu45_id_w(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_status

Запит статусу пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubGu45Status gohub_gu45_status(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`), статус якого запитується;

Результат:

У разі успіху - значення з перерахування GohubGu45Status . У разі помилки - «-1». Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_revision

Запит номера ревізії пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
int gohub_gu45_revision(
GohubGu45* gu45);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_text

Запит тексту пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const char* gohub_gu45_text(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_text_w

Запит тексту пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const GohubWChar* gohub_gu45_text_w(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_size

Запит передбачуваного розміру файлу, який вийде при збереженні документа у поточній кодуванні (`GohubGu45`).

Об'явлення:

```
int gohub_gu45_size(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`), текст якого запитується;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_signer_info

Запит інформації про пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const char* gohub_gu45_signer_info(
GohubGu46* gu45);
```

Параметри:

`gu45`

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_signer_info_w

Запит інформації про пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const GohubWChar* gohub_gu45_signer_info_w(
GohubGu46* gu45);
```

Параметри:

gu45

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_sign_time

Запит часу підписання пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const char* gohub_gu45_sign_time(
GohubGu45* gu45);
```

Параметри:

gu45

Об'єкт документа (GohubGu46).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_sign_time_w

Запит часу підписання пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
const GohubWChar* gohub_gu45_sign_time_w(
GohubGu45* gu45);
```

Параметри:

gu45

Об'єкт документа (GohubGu45).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_sign_name_w

Запит імені підписанта пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45 (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_gu45_sign_name_w(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_gu45_has_signature

Перевірити чи підписаний документ.

Об'явлення:

```
const GohubWChar* gohub_gu45_has_signature(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_gu45

Збереження пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubBool gohub_save_gu45(
GohubGu45* gu45,
const char* path,
int codePage);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`), що зберігається на диск;

`path`

Шлях, куди зберігається файл;

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_gu45_w

Збереження пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45.

Об'явлення:

```
GohubBool gohub_save_gu45_w(
GohubGu45* gu45,
const GohubWChar* path,
int codePage);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`), що зберігається на диск;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16);

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_gu45

Закриття документа (`GohubGu45`).

Об'явлення:

```
GohubBool gohub_close_gu45(
GohubGu45* gu45);
```

Параметри:

`gu45`

Об'єкт документа (`GohubGu45`).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu45_by_number

Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45 за номером станції, номер пам'ятки та дата її формування.

Об'явлення:

```
GohubGu45* gohub_query_gu45_by_number(GohubConnection* connection,
const char* registration_esr, const char* registration_num, const
char* registration_date);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

registration_esr

Станція формування пам'ятки.

registration_num

Номер пам'ятки.

registration_date

Номер пам'ятки.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_gu45_by_number_w

Запитати пам'ятку про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45 за номером станції, номер пам'ятки та дата її формування (в UTF-16).

Об'явлення:

```
GohubGu45* gohub_query_gu45_by_number_w(GohubConnection* connection,
const GohubWChar* registration_esr, const GohubWChar*
registration_num , const GohubWChar* registration_date);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

registration_esr

Станція формування пам'ятки (у кодуванні UTF-16).

registration_num

Номер пам'ятки (у кодуванні UTF-16).

registration_date

Номер пам'ятки (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_gu45_printable_form

Запитати друковану форму пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45 за його ID та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_gu45_printable_form(
      GohubConnection* connection,
```

```
        const char* gu45Id,
        const char* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu45Id`

Унікальний ідентифікатор пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів.

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_gu45_printable_form_w

Запитати друковану форму пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів ГУ-45 за його ID (у кодуванні UTF-16) та зберегти її в файл.

Об'явлення:

```
GohubBool gohub_query_and_save_gu45_printable_form_w(
      GohubConnection* connection,
      const GohubWChar* gu45Id,
      const GohubWChar* path);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu45Id`

Унікальний ідентифікатор пам'ятки про подачу/прибирання вагонів та видачу/прийом контейнерів (у кодуванні UTF-16).

`path`

Шлях, за яким зберегти запитану друковану форму.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.9. Робота з фільтрами для запитуваних документів

Нижче перераховані функції встановлюють фільтри, що впливають на функції запиту документів з п. 4.4.3.

`gohub_clear_all_filters`	Скинути всі фільтри
`gohub_set_filter_by_document_status`	Встановити фільтр за статусом документа
`gohub_set_filter_by_document_number`	Встановити фільтр за номером документа

gohub_set_filter_by_document_number_w	Встановити фільтр за номером документа (в UTF-16)
gohub_set_filter_by_wagon_number	Встановити фільтр за номером вагона
gohub_set_filter_by_wagon_number_w	Встановити фільтр за номером вагона (в UTF-16)
gohub_set_filter_by_departure_client	Встановити фільтр за кодом відправника
gohub_set_filter_by_departure_client_w	Встановити фільтр за кодом відправника (в UTF-16)
gohub_set_filter_by_departure_payer	Встановити фільтр за кодом платника по відправленню
gohub_set_filter_by_departure_payer_w	Встановити фільтр за кодом платника по відправленню (в UTF-16)
gohub_set_filter_by_departure_station	Встановити фільтр за кодом станції відправлення
gohub_set_filter_by_departure_station_w	Встановити фільтр за кодом станції відправлення (в UTF-16)
gohub_set_filter_by_arrival_client	Встановити фільтр за кодом одержувача
gohub_set_filter_by_arrival_client_w	Встановити фільтр за кодом одержувача (в UTF-16)
gohub_set_filter_by_arrival_payer	Встановити фільтр за кодом платника по прибуттю
gohub_set_filter_by_arrival_payer_w	Встановити фільтр за кодом платника по прибуттю (в UTF-16)
gohub_set_filter_by_arrival_station	Встановити фільтр за кодом станції призначення
gohub_set_filter_by_arrival_station_w	Встановити фільтр за кодом станції призначення (в UTF-16)
gohub_get_filter_by_document_status	Запитати значення фільтру за статусом документа
gohub_get_filter_by_document_number	Запитати значення фільтру за номером документа
gohub_get_filter_by_document_number_w	Запитати значення фільтру за номером документа (в UTF-16)
gohub_get_filter_by_wagon_number	Запитати значення фільтру за номером вагона
gohub_get_filter_by_wagon_number_w	Запитати значення фільтру за номером вагона (в UTF-16)
gohub_get_filter_by_departure_client	Запитати значення фільтру за кодом відправника
gohub_get_filter_by_departure_client_w	Запитати значення фільтру за кодом відправника (в UTF-16)
gohub_get_filter_by_departure_payer	Запитати значення фільтру за кодом платника по відправленню
gohub_get_filter_by_departure_payer_w	Запитати значення фільтру за кодом платника по відправленню (в UTF-16)
gohub_get_filter_by_departure_station	Запитати значення фільтру за кодом станції відправлення
gohub_get_filter_by_departure_station_w	Запитати значення фільтру за кодом станції відправлення (в UTF-16)
gohub_get_filter_by_arrival_client	Запитати значення фільтру за кодом одержувача
gohub_get_filter_by_arrival_client_w	Запитати значення фільтру за кодом одержувача (в UTF-16)
gohub_get_filter_by_arrival_payer	Запитати значення фільтру за кодом платника по прибуттю
gohub_get_filter_by_arrival_payer_w	Запитати значення фільтру за кодом платника по прибуттю (в UTF-16)
gohub_get_filter_by_arrival_station	Запитати значення фільтру за кодом станції призначення
gohub_get_filter_by_arrival_station_w	Запитати значення фільтру за кодом станції призначення (в UTF-16)

gohub_clear_all_filters

Скинути всі налаштування фільтрів.

Об'явлення:

```
GohubBool gohub_clear_all_filters(
      GohubConnection* connection);
```

Параметри:

```
connection
```

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_document_status

Встановити фільтр за статусом документа.

Об'явлення:

```
GohubBool gohub_set_filter_by_document_status(
      GohubConnection* connection,
      int documentStatusCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentStatusCode`

Код статусу документа, якщо вказати 0 – фільтр буде скинуто на «всі»;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_document_number

Встановити фільтр за номером документа.

Об'явлення:

```
GohubBool gohub_set_filter_by_document_number(
      GohubConnection* connection,
      const char* documentNumber);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentNumber`

Номер документа, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_document_number_w

Встановити фільтр за номером документа (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_document_number_w(
      GohubConnection* connection,
```

```
        const GohubWChar* documentNumber);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`documentNumber`

Номер документа (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_wagon_number

Встановити фільтр за номером вагона.

Об'явлення:

```
GohubBool gohub_set_filter_by_wagon_number(
        GohubConnection* connection,
        const char* wagonNumber);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`wagonNumber`

Номер вагона, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_wagon_number_w

Встановити фільтр за номером вагона (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_wagon_number_w(
        GohubConnection* connection,
        const GohubWChar* wagonNumber);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`wagonNumber`

Номер вагона (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_client

Встановити фільтр за кодом відправника.

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_client(
      GohubConnection* connection,
      const char* clientCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`clientCode`

Код відправника, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_client_w

Встановити фільтр за кодом відправника (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_client_w(
      GohubConnection* connection,
      const GohubWChar* clientCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`clientCode`

Код відправника (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_payer

Встановити фільтр за кодом платника по відправленню.

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_payer(
      GohubConnection* connection,
      const char* payerCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`payerCode`

Код платника по відправленню, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_payer_w

Встановити фільтр за кодом платника по відправленню (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_payer_w(
      GohubConnection* connection,
      const GohubWChar* payerCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`payerCode`

Код платника по відправленню (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_station

Встановити фільтр за кодом станції відправлення.

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_station(
      GohubConnection* connection,
      const char* stationCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`stationCode`

Код станції відправлення, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_departure_station_w

Встановити фільтр за кодом станції відправлення (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_departure_station_w(
      GohubConnection* connection,
      const GohubWChar* stationCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`stationCode`

Код станції відправлення (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_client

Встановити фільтр за кодом одержувача.

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_client(
      GohubConnection* connection,
      const char* clientCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`clientCode`

Код одержувача, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_client_w

Встановити фільтр за кодом одержувача (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_client_w(
      GohubConnection* connection,
      const GohubWChar* clientCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`clientCode`

Код одержувача (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_payer

Встановити фільтр за кодом платника по прибуттю.

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_payer(
      GohubConnection* connection,
      const char* payerCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`payerCode`

Код платника по прибуттю, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_payer_w

Встановити фільтр за кодом платника по прибуттю (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_payer_w(
      GohubConnection* connection,
      const GohubWChar* payerCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`payerCode`

Код платника по прибуттю (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_station

Встановити фільтр за кодом станції призначення.

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_station(
      GohubConnection* connection,
      const char* stationCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`stationCode`

Код станції призначення, якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_set_filter_by_arrival_station_w

Встановити фільтр за кодом станції призначення (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_set_filter_by_arrival_station_w(
      GohubConnection* connection,
      const GohubWChar* stationCode);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`stationCode`

Код станції призначення (у кодуванні UTF-16), якщо передати порожній покажчик або порожній рядок – фільтр буде скинуто на «всі».

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_document_status

Запитати значення фільтру за статусом документа. Повертає значення цілого типу, що описано переліком `GohubDocumentStatus`.

Об'явлення:

```
GohubDocumentStatus gohub_get_filter_by_document_status(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_document_number

Запитати значення фільтру за номером документа.

Об'явлення:

```
const char* gohub_get_filter_by_document_number(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_document_number_w

Запитати значення фільтру за номером документа (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_document_number_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_wagon_number

Запитати значення фільтру за номером вагона.

Об'явлення:

```
const char* gohub_get_filter_by_wagon_number(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_wagon_number_w

Запитати значення фільтру за номером вагона (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_wagon_number_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_client

Запитати значення фільтру за кодом відправника.

Об'явлення:

```
const char* gohub_get_filter_by_departure_client(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_client_w

Запитати значення фільтру за кодом відправника (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_departure_client_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_payer

Запитати значення фільтру за кодом платника по відправленню.

Об'явлення:

```
const char* gohub_get_filter_by_departure_payer(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_payer_w

Запитати значення фільтру за кодом платника по відправленню (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_departure_payer_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_station

Запитати значення фільтру за кодом станції відправлення.

Об'явлення:

```
const char* gohub_get_filter_by_departure_station(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_departure_station_w

Запитати значення фільтру за кодом станції відправлення (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_departure_station_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_client

Запитати значення фільтру за кодом одержувача.

Об'явлення:

```
const char* gohub_get_filter_by_arrival_client(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_client_w

Запитати значення фільтру за кодом одержувача (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_arrival_client_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_payer

Запитати значення фільтру за кодом платника по прибуттю.

Об'явлення:

```
const char* gohub_get_filter_by_arrival_payer(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_payer_w

Запитати значення фільтру за кодом платника по прибуттю (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_arrival_payer_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_station

Запитати значення фільтру за кодом станції прибытия.

Об'явлення:

```
const char* gohub_get_filter_by_arrival_station(
      GohubConnection* connection);
const char* gohub_get_filter_by_arrival_payer(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_filter_by_arrival_station_w

Запитати значення фільтру за кодом станції прибытия (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_filter_by_arrival_station_w(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect.

Результат:

У разі успіху - не порожній рядок. Якщо результат порожній рядок - значить фільтр не накладений або з'єднання не дійсне. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.10. Перевірка електронно-цифрового підпису

gohub_document_has_signature	Дізнатися, чи містить документ електронно-цифровий підпис
gohub_document_check_signature	Виконати перевірку електронно-цифрового підпису документа
gohub_document_signer_name	Отримати ім'я особи, яка підписала електронний документ
gohub_document_signer_name_w	Отримати ім'я особи, яка підписала електронний документ (в UTF-16)
gohub_document_signer_info	Отримати детальну інформацію про особу, що підписала електронний документ
gohub_document_signer_info_w	Отримати детальну інформацію про особу, що підписала електронний документ (в UTF-16)
gohub_document_sign_time	Отримати позначку часу електронно-цифрового підпису
gohub_document_sign_time_w	Отримати позначку часу електронно-цифрового підпису (в UTF-16)
gohub_document_signer_info_by_index	Отримати детальну інформацію про особу, що підписала електронний документ за індексом
gohub_document_signer_info_by_index_w	Отримати детальну інформацію про особу, що підписала електронний документ за індексом (в UTF-16)
gohub_document_signer_name_by_index	Отримати ім'я особи, яка підписала електронний документ за індексом
gohub_document_signer_name_by_index_w	Отримати ім'я особи, яка підписала електронний документ за індексом (в UTF-16)
gohub_document_sign_time_by_index	Отримати позначку часу електронно-цифрового підпису за індексом
gohub_document_sign_time_by_index_w	Отримати позначку часу електронно-цифрового підпису за індексом (в UTF-16)
gohub_document_signature_count	Отримати кількість підписів в документі
gohub_document_sign_status_by_index	Отримати статус документу в момент підпису за індексом
gohub_document_sign_error_by_index_w	Отримати помилку за її наявності в підписі за індексом (в UTF-16)
gohub_document_sign_error_by_index	Отримати помилку за її наявності в підписі за індексом

gohub_document_has_signature

Перевірка, чи є у документа електронний підпис (GohubDocument).

Об'явлення:

```
GohubBool gohub_document_has_signature(

    GohubDocument* document);
```

Параметри:

document

Об'єкт документа (GohubDocument);

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_check_signature

Перевірка відповідності документа його електронному підпису.

Об'явлення:

```
GohubBool gohub_document_check_signature(

      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signature_name

Запит імені особи, яка підписала документ.

Об'явлення:

```
const char* gohub_document_signer_name(

      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signature_name_w

Запит імені особи, яка підписала документ, результат буде у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_signer_name_w(

      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signer_info

Запит інформації про підписанта документу – повертається форматований рядок з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const char* gohub_document_signer_info(

      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signer_info_w

Запит інформації про підписанта документу – повертається форматований рядок (у кодуванні UTF-16) з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const GohubWChar* gohub_document_signer_info_w(

      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_time

Запит інформації про час підписання документа.

Об'явлення:

```
const char* gohub_document_sign_time(
      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_time_w

Запит інформації про час підписання документа, повертає значення у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_sign_time_w(
      GohubDocument* document);
```

Параметри:

`document`

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signer_info_by_index

Запит інформації про підписанта документу за індексом – повертається форматований рядок з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const char* gohub_document_signer_info_by_index(

      GohubDocument* document, int index);
```

Параметри:

`document`

Документ;

`index`

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signer_info_by_index _w

Запит інформації про підписанта документу за індексом – повертається форматований рядок (у кодуванні UTF-16) з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const GohubWChar* gohub_document_signer_info_by_inde_w(

      GohubDocument* document, int index);
```

Параметри:

`document`

Документ;

`index`

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signature_name_by_index

Запит імені особи, яка підписала документ за індексом.

Об'явлення:

```
const char* gohub_document_signer_name_by_index(
    GohubDocument* document, int index);
```

Параметри:

`document`

Документ;

`index`

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_signature_name_by_index_w

Запит імені особи, яка підписала документ за індексом, результат буде у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_signer_name_by_index_w(
    GohubDocument* document, int index);
```

Параметри:

`document`

Документ;

`index`

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_time_by_index

Запит інформації про час підписання документа за індексом.

Об'явлення:

```
const char* gohub_document_sign_time_by_index(
    GohubDocument* document, int index);
```

Параметри:

`document`

Документ;

index

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_time_by_index_w

Запит інформації про час підписання документа за індексом, повертає значення у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_sign_time_by_index_w(
      GohubDocument* document, int index);
```

Параметри:

document

Документ;

index

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_count

Запит кількості підписів в документі.

Об'явлення:

```
const int gohub_document_sign_count(
      GohubDocument* document);
```

Параметри:

document

Документ;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_status_by_index

Запит інформації про статус документу на момент підпису за індексом.

Об'явлення:

```
UzRwcDocStatus gohub_document_sign_time_by_index_w(
      GohubDocument* document, int index);
```

Параметри:

document

Документ;

index

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_error_by_index

Запит інформації про помилку підписання документа за індексом.

Об'явлення:

```
const char* gohub_document_sign_error_by_index(
      GohubDocument* document, int index);
```

Параметри:

document

Документ;

index

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_sign_error_by_index_w

Запит інформації про помилку підписання документа за індексом, повертає значення у кодуванні UTF-16.

Об'явлення:

```
const GohubWChar* gohub_document_sign_error_by_index_w(
      GohubDocument* document, int index);
```

Параметри:

document

Документ;

index

індекс підпису;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.11. Накладення електронно-цифрового підпису

gohub_open_private_key	Відкрити електронний ключ за паролем
gohub_open_private_key_w	Відкрити електронний ключ за паролем (в UTF-16)
gohub_open_private_key_by_bytes	Відкрити електронний ключ у вигляді масиву байт за паролем
gohub_open_private_key_by_bytes_w	Відкрити електронний ключ у вигляді масиву байт за паролем (в UTF-16)
gohub_open_private_key_from_path	Відкрити електронний ключ за шляхом та паролем

gohub_open_private_key_from_path_w	Відкрити електронний ключ за шляхом та паролем (в UTF-16)
gohub_private_key_owner_name	Отримати ім'я власника електронного ключа
gohub_private_key_owner_name_w	Отримати ім'я власника електронного ключа (в UTF-16)
gohub_private_key_owner_info	Отримати детальну інформацію про власника електронного ключа
gohub_private_key_owner_info_w	Отримати детальну інформацію про власника електронного ключа (в UTF-16)
gohub_sign_document	Підписати документ електронно-цифровим підписом
gohub_close_private_key	Завершити роботу з електронним ключем
gohub_sign_fdu92	Підписати ФДУ-92 електронно-цифровим підписом
gohub_sign_gu46	Підписати ГУ-46 електронно-цифровим підписом
gohub_delete_old_certs_csk_uz	Перезапит сертифікатів АЦСК УЗ
gohub_delete_old_certs_csk_uz_w	Перезапит сертифікатів АЦСК УЗ (в UTF-16)
gohub_keys_list	Запит списку носіїв на яких може бути розташований електронний ключ
gohub_keys_list_w	Запит списку носіїв на яких може бути розташований електронний ключ (в UTF-16)
gohub_open_private_key_by_index	Відкрити електронний ключ за паролем , індексом типу носія, індексом назвою носія
gohub_open_private_key_by_index_w	Відкрити електронний ключ за паролем , індексом типу носія, індексом назвою носія (в UTF-16)

gohub_open_private_key

Відкриття сесії роботи електронно-цифрового ключа.

Об'явлення:

```
GohubBool gohub_open_private_key(
      GohubConnection* connection,
      const char* passwordToKey);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_w

Відкриття сесії роботи електронно-цифрового ключа.

Об'явлення:

```
GohubBool gohub_open_private_key_w(
      GohubConnection* connection,
      const GohubWChar* passwordToKey);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа у кодуванні UTF-16;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_by_bytes

Відкриття сесії роботи електронно-цифрового ключа, що надано масивом байт.

Об'явлення:

```
GohubBool gohub_open_private_key(
      GohubConnection* connection,
      const char* passwordToKey,
      unsigned char* keyBinary,
      unsigned int length);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа;

`keyBinary`

Електронний ключ у вигляді масиву байт;

`length`

Довжина масиву байт;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_by_bytes_w

Відкриття сесії роботи електронно-цифрового ключа, що надано масивом байт.

Об'явлення:

```
GohubBool gohub_open_private_key_w(
      GohubConnection* connection,
      const GohubWChar* passwordToKey,
      unsigned char* keyBinary,
      unsigned int length);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа у кодуванні UTF-16;

`keyBinary`

Електронний ключ у вигляді масиву байт;

`length`

Довжина масиву байт;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_from_path

Відкриття сесії роботи електронно-цифрового ключа, що розташований за вказаним шляхом.

Об'явлення:

```
GohubBool gohub_open_private_key(
      GohubConnection* connection,
      const char* passwordToKey,
      const char* keyFileName);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа;

`keyFileName`

Шлях до електронного ключу;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_from_path_w

Відкриття сесії роботи електронно-цифрового ключа, що розташований за вказаним шляхом.

Об'явлення:

```
GohubBool gohub_open_private_key_w(
      GohubConnection* connection,
      const GohubWChar* passwordToKey,
      const GohubWChar* keyFileName);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

passwordToKey

Рядок з паролем від електронного ключа у кодуванні UTF-16;

keyFileName

Шлях до електронного ключу у кодуванні UTF-16;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_private_key_owner_name

Повертає ім'я власника електронно-цифрового ключа (у кодуванні UTF-16), робоча сесія якого зараз відкрита.

Об'явлення:

```
const char* gohub_private_key_owner_name(
      GohubConnection* connection);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_private_key_owner_name_w

Повертає ім'я власника електронно-цифрового ключа (у кодуванні UTF-16), робоча сесія якого зараз відкрита.

Об'явлення:

```
const GohubWChar* gohub_private_key_owner_name_w(
      GohubConnection* connection);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_private_key_owner_info

Запит інформації про власника електронно-цифрового ключа, робоча сесія якого зараз відкрита – повертається форматований рядок з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const char* gohub_private_key_owner_info(

      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_private_key_owner_info_w

Запит інформації про власника електронно-цифрового ключа, робоча сесія якого зараз відкрита – повертається форматований рядок (у кодуванні UTF-16) з ключами полів та їх значеннями, через роздільник.

Об'явлення:

```
const GohubWChar* gohub_private_key_owner_info_w(

      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_sign_document

Підписання документа електронно-цифровим ключем, сесія якого зараз відкрита.

Об'явлення:

```
GohubBool gohub_sign_document(
      GohubConnection* connection,
      GohubDocument* document);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`document`

Документ;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_private_key

Закриття робочої сесії електронно-цифрового ключа.

Об'явлення:

```
GohubBool gohub_close_private_key(
      GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_sign_fdu92

Підписання ФДУ-92 електронно-цифровим ключем, сесія якого зараз відкрита.

Об'явлення:

```
GohubBool gohub_sign_fdu92(
GohubConnection* connection, GohubFdu92* fdu92);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`fdu92`

Накопичувальна картка ФДУ-92 ;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_sign_gu46

Підписання ГУ-46 електронно-цифровим ключем, сесія якого зараз відкрита.

Об'явлення:

```
GohubBool gohub_sign_gu46(
GohubConnection* connection, GohubGu46* gu46);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`gu46`

Відомість про користування вагонами/контейнерами;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_old_certs_csk_uz

Перезапит сертифікатів АЦСК УЗ.

Об'явлення:

```
GohubBool gohub_delete_old_certs_csk_uz(
GohubConnection* connection, const char* passwordToKey);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа ГУ-46;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_delete_old_certs_csk_uz_w

Перезапит сертифікатів АЦСК УЗ (у кодуванні UTF-16).

Об'явлення:

```
GohubBool gohub_delete_old_certs_csk_uz_w(
GohubConnection* connection, const GohubWChar* passwordToKey);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа у кодуванні UTF-16;

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_keys_list

Запит списку носіїв на яких може бути розташований електронний ключ.

Об'явлення:

```
const char* gohub_keys_list(GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_keys_list_w

Запит списку носіїв на яких може бути розташований електронний ключ (в UTF-16).

Об'явлення:

```
const GohubWChar* gohub_keys_list_w(GohubConnection* connection);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_open_private_key_by_index

Відкрити електронний ключ за паролем , індексом типу носія, індексом назвою носія.

Об'явлення:

```
GohubBool gohub_open_private_key_by_index(GohubConnection*
connection, const char* passwordToKey, int mediaIndex , int
deviceIndex);
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`passwordToKey`

Рядок з паролем від електронного ключа;

`mediaIndex`

Індекс типу носія з переліку списку носіїв.

deviceIndex

Індекс назви носія з переліку списку носіїв.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_ open_private_key_by_index_w

Відкрити електронний ключ за паролем , індексом типу носія, індексом назвою носія (в UTF-16).

Об'явлення:

```
GohubBool gohub_open_private_key_by_index_w(GohubConnection*
connection, const GohubWChar* passwordToKey, int mediaIndex , int
deviceIndex);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

passwordToKey

Рядок з паролем від електронного ключа у кодуванні UTF-16;

mediaIndex

Індекс типу носія з переліку списку носіїв.

deviceIndex

Індекс назви носія з переліку списку носіїв.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

отримати за допомогою функції gohub_last_error і ін.

4.4.12. Операції з файлами електронних ключів

Цей блок функцій дозволяє зробити доступним для використання файли електронних ключів, розміщених в довільному місці файлової системи. Монтування файлу ключа по своєму ефекту аналогічно підключенню зовнішнього носія (наприклад, модуля флеш-пам'яті) з електронним ключем.

ВАЖЛИВО! Підключення електронного ключа розповсюджується тільки на поточного користувача.

gohub_mount_file_key	Монтувати електронний ключ з вказаного файлу
gohub_mount_file_key_w	Монтувати електронний ключ з вказаного файлу (в UTF-16)
gohub_unmount_file_key	Демонтувати електронний ключ з файлу
gohub_unmount_file_key_w	Демонтувати електронний ключ з файлу (в UTF-16)
gohub_query_mounted_file_keys	Запитати кількість монтованих з файлів електронних ключів
gohub_mounted_file_key_id	Запитати ID електронного ключа за його порядковим номером у переліку
gohub_mounted_file_key_id_w	Запитати ID електронного ключа за його порядковим номером у переліку (в UTF-16)
gohub_mounted_file_key_dir	Запитати шлях до електронного ключу за його порядковим номером у переліку
gohub_mounted_file_key_dir_w	Запитати шлях до електронного ключу за його порядковим номером у переліку (в UTF-16)

gohub_mount_file_key

Монтувати електронний ключ з вказаного файлу.

Об'явлення:

```
GohubBool gohub_mount_file_key(
      const char* keyId,
      const char* path);
```

Параметри:

keyId

Ідентифікатор ключа;

`path`

Шлях до файлу ключа.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_mount_file_key_w

Монтувати електронний ключ з вказаного файлу.

Об'явлення:

```
GohubBool gohub_mount_file_key_w(
      const GohubWChar* keyId,
      const GohubWChar* path);
```

Параметри:

`keyId`

Ідентифікатор ключа у кодуванні UTF-16;

`path`

Шлях до файлу ключа у кодуванні UTF-16.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_unmount_file_key

Демонтувати електронний ключ за його ID.

Об'явлення:

```
GohubBool gohub_unmount_file_key(
      const char* keyId);
```

Параметри:

`keyId`

Ідентифікатор ключа.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_unmount_file_key_w

Демонтувати електронний ключ за його ID.

Об'явлення:

```
GohubBool gohub_unmount_file_key_w(
      const GohubWChar* keyId);
```

Параметри:

`keyId`

Ідентифікатор ключа у кодуванні UTF-16.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_mounted_file_keys

Запитати кількість монтованих з файлів електронних ключів.

Об'явлення:

```
int gohub_query_mounted_file_keys();
```

Результат:

У разі успіху - не негативна значення. У разі помилки або порожнього списку - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_mounted_file_key_id

Отримати ID електронного ключа, за його порядковим номером у переліку монтованих електронних ключів.

Об'явлення:

```
const char* gohub_mounted_file_key_id(
      int index);
```

Параметри:

`index`

Порядковий номер.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_mounted_file_key_id_w

Отримати ID електронного ключа (у кодуванні UTF-16), за його порядковим номером у переліку монтованих електронних ключів.

Об'явлення:

```
const GohubWChar* gohub_mounted_file_key_id_w(
      int index);
```

Параметри:

`index`

Порядковий номер.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_mounted_file_key_dir

Отримати шлях до файлу електронного ключа за його порядковим номером.

Об'явлення:

```
const char* gohub_mounted_file_key_dir(
      int index);
```

Параметри:

`index`

Порядковий номер.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_mounted_file_key_dir_w

Отримати шлях (у кодуванні UTF-16) до файлу електронного ключа за його порядковим номером.

Об'явлення:

```
const GohubWChar* gohub_mounted_file_key_dir_w(
      int index);
```

Параметри:

`index`

Порядковий номер.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.13. Робота з АС «Месплан»

Цей блок функцій дозволяє отримати заявки за вказаним місяцем з АС «Месплан» .

`gohub_get_mp_months`	Запитати перелік ідентифікаторів місяців доступних в АС «Месплан» у вигляді рядку
`gohub_get_mp_months_w`	Запитати перелік ідентифікаторів місяців доступних в АС «Месплан» у вигляді рядку `(в UTF-16)`
`gohub_query_and_save_orders_for_month`	Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml
`gohub_query_and_save_orders_for_month_w`	Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml `(з параметрами в UTF-16)`
`gohub_query_and_save_orders_for_month_with_relogin`	Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml. З можливістю вказати логін та пароль.
`gohub_query_and_save_orders_for_month_with_relogin_w`	Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml. З можливістю вказати логін та пароль `(з параметрами в UTF-16)`

gohub_get_mp_months

Запитати перелік ідентифікаторів місяців доступних в АС «Месплан» у вигляді рядку.

Об'явлення:

```
const char* gohub_get_mp_months(
      GohubConnection* connection,
```

```
        int codePage)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`codePage`

Числове позначення сторінки кодування, в якій зберігається файл.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_get_mp_months_w

Запитати перелік ідентифікаторів місяців доступних в АС «Месплан» у вигляді рядку (в UTF-16).

Об'явлення:

```
const GohubWChar* gohub_get_mp_months_w(
      GohubConnection* connection)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_orders_for_month

Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml.

Об'явлення:

```
GohubBool gohub_query_and_save_orders_for_month(
      GohubConnection* connection,
      const char* month,
      const char* path)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`month`

Рядок з номером місяцю за який необхідно запросити заявки.

`Path`

Шлях до файлу в який необхідно зберегти заявки.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_orders_for_month_w

Запитати та зберегти заявки за вказаним місяцем з АС «Месплан» в xml.

Об'явлення:

```
GohubBool gohub_query_and_save_orders_for_month_w(
      GohubConnection* connection,
      const GohubWChar* month,
      const GohubWChar* path)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`month`

Рядок з номером місяцю за який необхідно запросити заявки (у кодуванні UTF-16).

`Path`

Шлях до файлу в який необхідно зберегти заявки (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_orders_for_month_with_relogin

Запитати та зберегти заявки з АС «Месплан» в xml за вказаним місяцем. З можливістю вказати логін та пароль.

Об'явлення:

```
GohubBool gohub_query_and_save_orders_for_month(
      GohubConnection* connection,
      const char* month,
      const char* login,
      const char* password,
      const char* path)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`month`

Рядок з номером місяцю за який необхідно запросити заявки.

`login`

Логін до АС «Месплан».

`password`

Пароль до АС «Месплан».

`Path`

Шлях до файлу в який необхідно зберегти заявки.

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_and_save_orders_for_month _with_relogin_w

Запитати та зберегти заявки з АС «Месплан» в xml за вказаним місяцем. З можливістю вказати логін та пароль.

Об'явлення:

```
GohubBool gohub_query_and_save_orders_for_month_w(
      GohubConnection* connection,
      const GohubWChar* month,
      const GohubWChar* login,
      const GohubWChar* password,
      const GohubWChar* path)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`month`

Рядок з номером місяцю за який необхідно запросити заявки (у кодуванні UTF-16).

`login`

Логін до АС «Месплан» (у кодуванні UTF-16).

`password`

Пароль до АС «Месплан» (у кодуванні UTF-16).

`Path`

шлях до файлу в який необхідно зберегти заявки (у кодуванні UTF-16).

Результат:

У разі успіху - значення True. У разі помилки - False. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.14. Робота з документами інформаційних послуг

Цей блок функцій дозволяє отримати документи інформаційних послуг.

`gohub_query_inform_services_document`	Запросити документ
`gohub_query_next_inform_services_document`	Запросити наступний документ
`gohub_save_inform_services_document`	Зберегти документ
`gohub_save_inform_services_document_w`	Зберегти документ (з параметрами в UTF-16)
`gohub_saveXml_inform_services_document`	Зберегти документ в форматі xml
`gohub_saveXml_inform_services_document_w`	Зберегти документ в форматі xml (з

	параметрами в UTF-16)
gohub_close_inform_services_document	Закрити документ
gohub_inform_services_document_id	Запросити ID документа
gohub_inform_services_document_revision	Запросити ревізію документа
gohub_inform_services_document_filename	Запитати ім'я документа
gohub_inform_services_document_filename_w	Запитати ім'я документа (в UTF-16)
gohub_inform_services_document_comment	Запитати коментар документа
gohub_inform_services_document_comment_w	Запитати коментар документа (в UTF-16)
gohub_inform_services_document_created_date	Запитати дату створення документа
gohub_inform_services_document_created_date_w	Запитати дату створення документа
gohub_inform_services_document_doc_date	Запитати дату документа
gohub_inform_services_document_doc_date_w	Запитати дату документа (в UTF-16)
gohub_inform_services_document_doc_is_empty	Запитати стан чи порожній документ

gohub_query_inform_services_document

Запросити документ інформаційних послуг за ідентифікатором.

Об'явлення:

```
GohubInformServicesDoc* gohub_query_inform_services_document(
      GohubConnection* connection,
      unsigned __int64 docId)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`docId`

Унікальний ідентифікатор документа, що запитується.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_next_inform_services_document

Запит документа інформаційних послуг наступного за переліком ревізій, починаючи від ревізії що передано параметром.

Об'явлення:

```
GohubInformServicesDoc* gohub_query_next_inform_services_document(
      GohubConnection* connection,
      unsigned __int64 lastRevision)
```

Параметри:

`connection`

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

`lastRevision`

Ревізія документа, від якої починається пошук наступного документа.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_inform_services_document

Збереження документа інформаційних послуг (GohubInformServicesDoc) на диск, за вказаним шляхом.

Об'явлення:

```
GohubBool gohub_save_inform_services_document(
      GohubInformServicesDoc* doc,
      const char* path)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc), що відправляється на сервер;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_save_inform_services_document_w

Збереження документа інформаційних послуг (GohubInformServicesDoc) на диск, за вказаним шляхом (з параметром в UTF-16).

Об'явлення:

```
GohubBool gohub_save_inform_services_document_w(
      GohubInformServicesDoc* doc,
      const GohubWChar* path)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc), що відправляється на сервер;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_saveXml_inform_services_document

Збереження документа інформаційних послуг в форматі xml (GohubInformServicesDoc) на диск, за вказаним шляхом.

Об'явлення:

```
GohubBool gohub_saveXml_inform_services_document(
      GohubInformServicesDoc* doc,
```

```
        const char* path)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc), що відправляється на сервер;

`path`

Шлях, куди зберігається файл.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_saveXml_inform_services_document_w

Збереження документа інформаційних послуг в форматі xml (GohubInformServicesDoc) на диск, за вказаним шляхом (з параметром в UTF-16).

Об'явлення:

```
GohubBool gohub_saveXml_inform_services_document_w(
      GohubInformServicesDoc* doc,
      const GohubWChar* path)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc), що відправляється на сервер;

`path`

Шлях, куди зберігається файл (у кодуванні UTF-16).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_inform_services_document

Закриття документа інформаційних послуг (GohubInformServicesDoc) на диск, за вказаним шляхом (з параметром вUTF-16).

Об'явлення:

```
GohubBool gohub_close_inform_services_document(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_id

Запит ідентифікатору документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
unsigned __int64 gohub_inform_services_document_id(
            GohubInformServicesDoc* doc)
```

Параметри:

`doc`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_revision

Запит ревізії документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
unsigned __int64 gohub_inform_services_document_revision(
            GohubInformServicesDoc* doc)
```

Параметри:

`doc`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_filename

Запит імені документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
const char* gohub_inform_services_document_filename(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_filename_w

Запит імені документа інформаційних послуг (GohubInformServicesDoc) в UTF-16.

Об'явлення:

```
const GohubWChar* gohub_inform_services_document_filename_w(
      GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_comment

Запит коментаря документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
const char* gohub_inform_services_document_comment(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_comment_w

Запит коментаря документа інформаційних послуг (GohubInformServicesDoc) в UTF-16.

Об'явлення:

```
const GohubWChar* gohub_inform_services_document_comment_w(
      GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_created_date

Запит дати створення документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
const char* gohub_inform_services_document_created_date(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_created_date_w

Запит дати створення документа інформаційних послуг (GohubInformServicesDoc) в UTF-16.

Об'явлення:

```
const GohubWChar* gohub_inform_services_document_created_date_w(
      GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_doc_date

Запит дати документа інформаційних послуг (GohubInformServicesDoc).

Об'явлення:

```
const char* gohub_inform_services_document_doc_date(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_doc_date_w

Запит дати документа інформаційних послуг (GohubInformServicesDoc) в UTF-16.

Об'явлення:

```
const GohubWChar* gohub_inform_services_document_doc_date_w(
      GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_inform_services_document_doc_is_empty

Запит стану чи порожній документ (GohubInformServicesDoc).

Об'явлення:

```
const bool gohub_inform_services_document_doc_is_empty(
            GohubInformServicesDoc* document)
```

Параметри:

`document`

Об'єкт документа (GohubInformServicesDoc).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін .

4.4.15. Обробка помилок

gohub_last_error_code	Отримати код помилки останньої операції
gohub_last_error_title	Отримати назва помилки останньої операції
gohub_last_error_title_w	Отримати назва помилки останньої операції (в UTF-16)
gohub_last_error_text	Отримати текст помилки останньої операції
gohub_last_error_text_w	Отримати текст помилки останньої операції (в UTF-16)

gohub_last_error_code

Запит коду помилки виконання останньої операції.

Об'явлення:

```
GohubErrcode gohub_last_error_code();
```

gohub_last_error_title

Запит заголовка помилки виконання останньої операції.

Об'явлення:

```
const char* gohub_last_error_title();
```

gohub_last_error_title_w

Запит заголовка помилки виконання останньої операції (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_last_error_title_w();
```

gohub_last_error_text

Запит тексту помилки виконання останньої операції.

Об'явлення:

```
const char* gohub_last_error_text();
```

gohub_last_error_text_w

Запит тексту помилки виконання останньої операції (у кодуванні UTF-16).

Об'явлення:

```
const GohubWChar* gohub_last_error_text_w();
```

4.4.16. Робота з інформацію про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів (за останні 5 днів від поточної дати)

gohub_query_dispatch_info	Запросити об’єкт з переліком за кодом станцій
gohub_query_dispatch_info_w	Запросити об’єкт з переліком за кодом станцій (в UTF-16)
gohub_document_info_description	Отримати текст опису за індексом
gohub_document_info_description_w	Отримати текст опису за індексом (в UTF-16)
gohub_document_info_count	Отримати кількість значень в переліку
gohub_document_info_is_empty	Отримати текст заповнення вагону
gohub_document_info_is_empty_w	Отримати текст заповнення вагону (в UTF-16)

gohub_document_info_wag_owner	Отримати власника вагону
gohub_document_info_wag_owner_w	Отримати власника вагону(в UTF-16)
gohub_document_info_date	Отримати дату
gohub_document_info_date_w	Отримати дату(в UTF-16)
gohub_document_info_number	Отримати номер
gohub_document_info_number_w	Отримати номер(в UTF-16)
gohub_document_info_type	Отримати тип
gohub_document_info_type_w	Отримати тип(в UTF-16)
gohub_close_dispatch_info	Закриття об’єкту

gohub_query_dispatch_info

Запросити об’єкт з переліком (GohubDispatchInfo).

Об'явлення:

```
GohubDispatchInfo* gohub_query_dispatch_info(GohubConnection*
connection, const char* start_esr, const char* end_esr);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

start_esr

Код ЄМР станції відправлення

end_esr

Код ЄМР станції призначення

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_dispatch_info_w

Запросити об’єкт з переліком (GohubDispatchInfo).

Об'явлення:

```
GohubDispatchInfo* gohub_query_dispatch_info_w(GohubConnection*
connection, const GohubWChar* start_esr, const GohubWChar* end_esr);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

start_esr

Код ЄМР станції відправлення(в UTF-16)

end_esr

Код ЄМР станції призначення(в UTF-16)

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_description

Отримати текст опису за індексом

Об'явлення:

```
const char*  gohub_document_info_description(GohubDispatchInfo* document, int index);
```

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_description_w

Отримати текст опису за індексом (в UTF-16)

Об'явлення:

```
const GohubWChar*  gohub_document_info_description_w(GohubDispatchInfo* document, int index);
```

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_count

Отримати кількість значень в переліку

Об'явлення:

```
int gohub_document_info_count(GohubDispatchInfo* document);
```

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_is_empty

Отримати текст заповнення вагону

Об'явлення:

const char* gohub_document_info_is_empty(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_is_empty _w

Отримати текст заповнення вагону (в UTF-16)

Об'явлення:

const GohubWChar* gohub_document_info_is_empty_w(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_wag_owner

Отримати власника вагону

Об'явлення:

const char* gohub_document_info_wag_owner(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_wag_owner_w

Отримати власника вагону(в UTF-16)

Об'явлення:

const GohubWChar* gohub_document_info_wag_owner_w(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

`Індекс за яким отримуємо значення з переліку.`

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_date

Отримати дату

Об'явлення:

const char* gohub_document_info_date(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

`Індекс за яким отримуємо значення з переліку.`

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_date_w

Отримати дату(в UTF-16)

Об'явлення:

const GohubWChar* gohub_document_ info_date_w(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

`Індекс за яким отримуємо значення з переліку.`

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_number

Отримати номер

Об'явлення:

const char* gohub_document_info_number(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_number_w

Отримати номер(в UTF-16)

Об'явлення:

const GohubWChar* gohub_document_ info_number_w(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_type

Отримати тип

Об'явлення:

const char* gohub_document_info_type(GohubDispatchInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubDispatchInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_document_info_type_w

Отримати тип(в UTF-16)

Об'явлення:

```
const GohubWChar*  gohub_document_ info_type_w(GohubDispatchInfo* document, int index);
```

Параметри:

document

Об’єкт з переліком (GohubDispatchInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_ dispatch_info

Закриття об’єкту

Об'явлення:

```
GohubBool gohub_close_dispatch_info(GohubDispatchInfo* document);
```

Параметри:

document

Об’єкт з переліком (GohubDispatchInfo).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

4.4.17. Робота з переліком Номер замовлення ПСТД

gohub_query_pstd_info	Запросити об’єкт з переліком за кодом станцій та видом роботи станції
gohub_query_pstd_info_w	Запросити об’єкт з переліком за кодом станцій та видом роботи станції (в UTF-16)
gohub_pstd_info_arrivaldate	Отримати дату прибуття за індексом
gohub_pstd_info_arrivaldate_w	Отримати дату прибуття за індексом (в UTF-16)
gohub_pstd_info_count	Отримати кількість значень в переліку
gohub_pstd_info_departuredate	Отримати дату відправлення за індексом
gohub_pstd_info_departuredate_w	Отримати дату відправлення за індексом (в UTF-16)
gohub_pstd_info_reqdate	Отримати дату замовлення за індексом
gohub_pstd_info_reqdate_w	Отримати дату замовлення за індексом (в UTF-16)

gohub_pstd_info_number	Отримати номер за індексом
gohub_pstd_info_number_w	Отримати номер за індексом (в UTF-16)
gohub_close_pstd_info	Закриття об’єкту

gohub_query_pstd_info

Запросити об’єкт з переліком (GohubPSTDInfo).

Об'явлення:

```
GohubPSTDInfo* gohub_query_pstd_info(GohubConnection* connection,
const char* work_kind, const char* departure_esr, const char*
arrival_esr);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

work_kind

Код виду роботи станції

departure _esr

Код ЄМР станції відправлення

arrival _esr

Код ЄМР станції призначення

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_query_pstd_info _w

Запросити об’єкт з переліком (GohubDispatchInfo).

Об'явлення:

```
GohubPSTDInfo* gohub_query_pstd_info(GohubConnection* connection,
const GohubWChar* work_kind, const GohubWChar* departure_esr, const
GohubWChar* arrival_esr);
```

Параметри:

connection

Покажчик з'єднання з Модулем Узгодження, отриманий раніше за допомогою функції gohub_connect;

work_kind

Код виду роботи станції (в UTF-16)

departure _esr

Код ЄМР станції відправлення (в UTF-16)

arrival _esr

Код ЄМР станції призначення (в UTF-16)

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_arrivaldate

Отримати дату прибуття за індексом

Об'явлення:

```
const char*  gohub_pstd_info_arrivaldate(GohubPSTDInfo* document, int index);
```

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_arrivaldate_w

Отримати дату прибуття за індексом (в UTF-16)

Об'явлення:

```
const GohubWChar*  gohub_pstd_info_arrivaldate_w(GohubPSTDInfo* document, int
index);
```

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_count

Отримати кількість значень в переліку

Об'явлення:

```
int gohub_pstd_info_count(GohubPSTDInfo * document);
```

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_departuredate

Отримати дату відправлення за індексом

Об'явлення:

const char* gohub_pstd_info_departuredate(GohubPSTDInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubPSTDInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_departuredate_w

Отримати дату відправлення за індексом (в UTF-16)

Об'явлення:

const GohubWChar* gohub_pstd_info_departuredate_w(GohubPSTDInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubPSTDInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_reqdate

Отримати дату відправлення за індексом

Об'явлення:

const char* gohub_pstd_info_reqdate(GohubPSTDInfo* document, int index);

Параметри:

`document`

Об’єкт з переліком (GohubPSTDInfo).

`index`

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_reqdate_w

Отримати дату відправлення за індексом (в UTF-16)

Об'явлення:

const GohubWChar* gohub_pstd_info_reqdate_w(GohubPSTDInfo* document, int index);

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_number

Отримати номер

Об'явлення:

const char* gohub_pstd_info_number(GohubPSTDInfo* document, int index);

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_pstd_info_number_w

Отримати номер(в UTF-16)

Об'явлення:

const GohubWChar* gohub_pstd_ info_number_w(GohubPSTDInfo* document, int index);

Параметри:

document

Об’єкт з переліком (GohubPSTDInfo).

index

Індекс за яким отримуємо значення з переліку.

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін.

gohub_close_ pstd_info

Закриття об’єкту

Об'явлення:

```
GohubBool gohub_close_pstd_info(GohubPSTDInfo* document);
```

Параметри:

`document`

Об’єкт з переліком (GohubPSTDInfo).

Результат:

У разі успіху - значення відмінне від нуля. У разі помилки - 0. Інформацію про помилку можна отримати за допомогою функції gohub_last_error і ін .

4.5. Коди помилок

Коди помилок у *Бібліотеці* представлені переліком (*enum*) `GohubErrcode`, що містить набір констант, семантично відповідних можливим типам помилок, що виникають при роботі з *Бібліотекою*.

Звідна таблиця кодів помилок

`gohub_success`	Операція виконана успішно
`gohub_server_connection_could_not_be_established`	Помилка при відкритті з'єднання з *Модулем Узгодження*
`gohub_server_inaccessible`	Служба *Модуля Узгодження* недоступна
`gohub_document_creation_failed`	Помилка відкриття файлу
`gohub_document_query_failed`	Помилка при запиті документа з АС «Клієнт УЗ» за ідентифікатором
`gohub_next_document_query_failed`	Помилка при запиті документа з АС «Клієнт УЗ» за номером ревізії
`gohub_document_sending_failed`	Помилка при відправлені документа в АС «Клієнт УЗ»
`gohub_document_saving_failed`	Помилка при збереженні документа в файл
`gohub_document_loading_failed`	Помилка при завантаженні документа з файлу
`gohub_invalid_code_page`	Недійсна кодова сторінка
`gohub_private_key_could_not_be_opened`	Помилка при відкритті електронного ключа
`gohub_private_key_path_could_not_be_opened`	Помилка при відкритті електронного ключа за вказаним шляхом
`gohub_private_key_bytes_could_not_be_opened`	Помилка при відкритті електронного ключа масивом байт
`gohub_private_key_is_inaccessible`	Електронний ключ недоступний. Можливо, пристрій від'єднали від комп'ютера
`gohub_document_could_not_be_signed`	Помилка при накладені електронно-цифрового підпису
`gohub_document_signature_verification_failed`	Електронно-цифровий підпис недійсний, можливо, дані були підтверджені
`gohub_document_has_not_signature`	Документ не містить електронно-цифровий підпис
`gohub_client_is_obsolete`	Версія клієнта застаріла, необхідно оновити
`gohub_server_is_obsolete`	Версія сервера застаріла, необхідно оновити
`gohub_document_reclamation_failed`	Відкликання документа не вдалося
`gohub_document_deletion_failed`	Видалити документ не вдалося
`gohub_attachment_creation_failed`	Створити супровідний документ не вдалося
`gohub_attachment_sending_failed`	Відправити супровідний документ не вдалося
`gohub_attachment_query_failed`	Запитати супровідний документ не вдалося

gohub_attachment_deletion_failed	Видалити супровідний документ не вдалося
gohub_attaching_to_document_failed	Додати супровідний документ до перевізного документу не вдалося
gohub_detaching_from_document_failed	Відкріпити супровідний документ від перевізного документа не вдалося
gohub_mount_of_file_key_failed	Монтування електронного ключа з файлу не вдалося.
gohub_unmount_of_file_key_failed	Демонтувати електронного ключа з файлу не вдалося.
gohub_enumerating_of_file_keys_failed	Отримати перелік монтованих з файлів електронних ключів не вдалося.
gohub_mounted_file_key_inaccessible	Монтований з файлу електронний ключ не доступний.
gohub_edata_creating_failed	Помилка при створенні ЕД попереднього інформування
gohub_edata_sending_failed	Помилка при відправлені ЕД попереднього інформування в АС «Клієнт УЗ»
gohub_edata_updating_failed	Помилка при оновленні ЕД попереднього інформування в АС «Клієнт УЗ»
gohub_edata_query_failed	Помилка при запиті ЕД попереднього інформування з АС «Клієнт УЗ» за ідентифікатором
gohub_next_edata_query_failed	Помилка при запиті ЕД попереднього інформування з АС «Клієнт УЗ» за ревізією
gohub_pipackage_query_failed	Помилка при запиті пакета ІП з АС «Клієнт УЗ» за ідентифікатором
Gohub_next_pipackage_query_failed	Помилка при запиті пакета ІП з АС «Клієнт УЗ» за ревізією
gohub_mp_months_query_failed	Помилка при запиті переліку ідентифікаторів місяців з АС «Месплан»
gohub_orders_of_months_query_failed	Помилка при запиті заявок з АС «Месплан»
gohub_inform_services_doc_saving_failed	Помилка при збереженні документа інформаційних послуг
gohub_inform_services_doc_query_failed	Помилка при запиті документа інформаційних послуг
gohub_query_changes_inform_services_failed	Помилка при запиті змін за документами інформаційних послуг
gohub_query_next_inform_services_document_failed	Помилка при запиті наступного документа інформаційних послуг
gohub_programm_error = 0x1000	Невідома помилка в програмі
gohub_invalid_operation	Програма користувача виконала неприпустиму операцію

Більш детальний опис програмного інтерфейсу бібліотеки можна отримати з файлу gohub.client.errors.h (Додаток Б).

4.6. Приклади використання

а) Вивід на екран інформації про помилку

Визначимо функцію `print_error`, що виводить на екран детальну інформацію про помилку, включно з інформацією про вкладені помилки, якщо вони існують. Ця функція стане в нагоді у наступних прикладах.

```
#include <stdio.h>
#include "gohub.client.h"

void print_last_error()
{
  printf("Error [%04x] -- %s -- %s\n",
    gohub_last_error_code(),
    gohub_last_error_title(),
    gohub_last_error_text());
}
```

б) Завантаження документа з файлу та передача в АС «Клієнт УЗ»

Зараз продемонструємо завантаження документа з файлу та передачу його в АС «Клієнт УЗ». При цьому використовуємо електронний ключ для накладення електронно-цифрового підпису. Для виводу інформації про можливі помилки використовуємо функцію `print_error`, визначену раніше в п.4.5.а.

```
#include "gohub.client.h"
#include <stdio.h>
#include <string.h>

void print_last_error();

void load_and_send_document(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  const char* path,           // шлях до файлу документа
  const char* password) // Пароль до ключу. Якщо не задано, документ не буде
підписано
{
  GohubConnection* connection;
  GohubDocument* document;
  char document_id[256];

  // Створити підключення до Модулю Узгодження
  printf("Connectiong to %s:%d\n", host, port);
  connection = gohub_connect(host, port);

  if(connection)
  { // Підключення створити успішно
    puts("Connection opened");

    // Завантажити документ з файлу
    document = gohub_load_document(path);

    if(document)
    { // Документ завантажено успішно
      printf("Document is loaded: size=%d bytes\n",
gohub_document_size(document));
      if(password)
      { //Підписати документ
        puts("Signing document . . .");
        if(gohub_open_private_key(connection, password))
```

```
        {
          printf("Key owner name: %s\n",
gohub_private_key_owner_name(connection));
          printf("Private key info: %s\n",
gohub_private_key_owner_info(connection));
          if(gohub_sign_document(connection, document))
            puts("Document signed");
          else
          {
            print_last_error();
            return;
          }
        }
        else
        { //Не вдалося отримати доступ до закритого ключу
          print_last_error();
          return;
        }
      }
      // Передати документ в АС "Клиєнт УЗ" через Модуль Узгодження
      if(gohub_send_document(connection, document))
      { // Документ передано успішно
        printf("Document is sended: id=%s revision=%d\n",
          gohub_document_id(document),
          gohub_document_revision(document));
      // Перевірка наявності попереджень після відправки документу
        const char* str = gohub_document_warning(doc);
            if (str != NULL)
                printf("Warning in document: %s\n", str);
        // Зберігаємо ідентифікатор документа
        // для подальшого використання
        strcpy(document_id, gohub_document_id(document));
      }
      else
      { // Помилка при передачі документа
        print_last_error();
      }
      // Завершити роботу з документом
      gohub_close_document(document);
            // Спробуємо запросити цей документ за ідентифікатором
            document = gohub_query_document(connection, document_id);
            if(document)
            { // Запит документа виконано успішно
              // Виводимо документ на екран у кодуванні windows-1251
              puts(gohub_document_text(document));
              // Завершити роботу з документом
              gohub_close_document(document);
            }
            else
            { // Помилка при отриманні документа
              print_last_error();
            }
    }
    else
    { // Помилка при завантаженні документа з файлу
      print_last_error();
    }
    // Закрити підключення до Модуля Узгодження
    gohub_disconnect(connection);
  }
  else
  { // Помилка при підключенні до Модулю Узгодження
    print_last_error();
  }
}
```

в) Запит документів з АС «Клієнт УЗ»

Цей приклад демонструє техніку поступового запиту документів їх АС «Клієнт УЗ» за номерами ревізій. Окрім того показано зміни кодової сторінки документа та Збереження запрошенных документів у файли. Окрім того, виконується перевірка електронно-цифрового підпису. Для виводу інформації про помилки по-прежнему використовується функція `print_error`, визначену раніше в п.4.5.а

```
#include "gohub.client.h"
#include <stdio.h>
#include <stdlib.h>

void print_last_error();

int query_and_save_documents(
  const char* host,                 // IP адреса або ім'я комп'ютера Модуля
Узгодження
  int port,                         // номер TCP порту для підключення
  int startRevision,          // номер ревізії з якої отримувати перевізні
документи
  int maxCount,                     // Максимальна кількість документів , що буде
запитано
  const char* targetFolder)   // тека куди зберігати отримані документи
{
  int lastRevision = startRevision;
  GohubConnection* connection;
  GohubDocument* document;
  int count;

  // Створити підключення до Модулю Узгодження
  printf("Connectiong to %s:%d", host, port);
  connection = gohub_connect(host, port);

  if(connection)
  { // Підключення створити успішно
    puts("Connection opened");

    for (count = 0; count < maxCount; ++count)
    { // Запит наступного документа за ревізією
      document = gohub_query_next_document(connection, lastRevision);

      if(document)
      { // Запит документа виконано успішно
        printf("Document received: id=%s revision=%d\n",
          gohub_document_id(document),
          gohub_document_revision(document));

        // Запам'ятати ревізію останнього документа
        lastRevision = gohub_document_revision(document);

        // Вивести документ на екран
        wprintf(L"%s\n", gohub_document_text_w(document));
        { // Зберегти документ в другой кодировке.
          char path[_MAX_PATH];
          sprintf(path, "%s\\%s.xml", targetFolder, gohub_document_id(document));
          if(gohub_save_document(document, path, 866))
          { // Документ сохранен успішно
            printf("Document saved to file: path='%s'\n", path);
          }
          else
          { // Помилка при збереженні документа
            print_last_error();
```

```
            gohub_close_document(document);
                  continue;
            }
          }

          if(gohub_document_has_signature(document))
          { //Перевірка ЭЦП
            if(gohub_document_check_signature(document))
            {
              wprintf(L"%s\n", gohub_document_signer_name_w(document));
              wprintf(L"%s\n", gohub_document_signer_info_w(document));
               wprintf(L"%s\n", gohub_document_sign_time_w(document));
            }
            else
            {
              print_last_error();
            }
          }
          // Завершення роботи з документом
          gohub_close_document(document);
        }
        else
        { // Не вдалося отримати наступний за ревізією документ.
          // Це може бути помилка, або нових документів більше немає.
          if(gohub_last_error_code() != gohub_success)
            print_last_error();
          break;
        }
      }
      // Закрити підключення до Модулю Узгодження
      gohub_disconnect(connection);
      puts("Connection closed");
    }
    else
    { // Помилка при підключенні до Модулю Узгодження
      print_last_error();
    }
    // Повернення функцією номера останньої обробленої ревізії
    return lastRevision;
}
```

г) Інші приклади використання

В інсталяційному пакеті Клієнта Модуля Узгодження представлені також інші приклади використання, з якими більш детально можна ознайомитися, заглянув безпосередньо в програмний код (тека Клієнт Модуля Узгодження\samples\ gohub.client.test.c). Тут ми коротко опишемо інші функції, надані у прикладах:

- Продемонстрована техніка поступового запиту документів ФДУ-92 з АС «Клієнт УЗ» за номерами ревізій

```
int query_and_save_fdu92s(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  int startRevision,          // номер ревізії з якої отримувати перевізні
документи
  int maxCount,               // Максимальна кількість документів, що буде
запитано
  const char* targetFolder);// тека куди зберігати отримані документи
```

- Продемонстрована техніка поступового запиту документів ЕД попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
__int64 query_and_save_edatas(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  __int64 startRevision,      // номер ревізії з якої отримувати ЕД (електронні
дані) попереднього інформування
  int maxCount,               // Максимальна кількість документів ЕД, що буде
запитано
  const char* targetFolder);// тека куди зберігати отримані документи ЕД
```

- Продемонстрована техніка поступового запиту пакетів попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
__int64 query_and_save_pi_packages(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  __int64 startRevision,      // номер ревізії з якої отримувати пакети ПІ
(попереднього інформування)
  int maxCount,               // Максимальна кількість пакети ПІ, що буде
запитано
  const char* targetFolder);// тека куди зберігати отримані пакети ПІ
```

- Продемонстрована техніка додання документа ЕД в пакет попереднього інформування

```
void add_edata_to_pi_package(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  const char* piPackageId,    // ID пакета ПІ (попереднього інформування), в який
будуть додаватися ЕД (електронні дані)
  const char* edataPath,      // шлях до xml-файлу з ЕД
  int edataType,              // код типу ЕД: 190 рахунок-фактура, 320
пакувальний лист
  char *newEdataId);          // сюди буде отримано ID створеного документа ЕД
```

- Продемонстрована техніка оновлення документа ЕД, що міститься на сервері АС «Клієнт УЗ»

```
__int64 update_edata(
  const char* host,           // IP адреса або ім'я комп'ютера Модуля Узгодження
  int port,                   // номер TCP порту для підключення
  const char* edataId,        // ID документа ЕД що буде змінюватись
        const char* edataPath);     // шлях до xml-файлу з ЕД
```

- Продемонстрована техніка отримання переліку з інформацією про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати)

```
try
   {
         printf(" Test Query Dispatch Info(C++)\n");
         printf(" Please enter ESR number of start station: ");
         char start_esr[101];
         scanf("%s", start_esr);
         printf(" Please enter ESR number of end station: ");
         char end_esr[101];
         scanf("%s", end_esr);
         GohubDispatchInfo* str = gohub_query_dispatch_info(connection,
         start_esr, end_esr);
         check_errors();
         printf("\nlist loaded.\n");
         for(int i = 0; i < gohub_document_info_count(str); i++)
         {
               printf("%s\n", gohub_document_info_description(str, i));
               printf("%s\n", gohub_document_info_number(str, i));
```

```
            printf("%s\n", gohub_document_info_date(str, i));
            printf("%s\n", gohub_document_info_type(str, i));
            printf("%s\n", gohub_document_info_wag_owner(str, i));
            printf("%s\n", gohub_document_info_is_empty(str, i));
        }
        gohub_close_dispatch_info(str);
        printf("----------------\n");
    }
    catch (...)
    {
        printf("Exception");
    }
```

- Продемонстрована техніка отримання переліку підписів з ПД

```
    GohubDocument* document = gohub_query_document(connection, id):
    for (int i = 0; i < gohub_document_signature_count(document); i++)
    {
        printf("\n %d) Document received: id=%d\n %s\n", i + 1, id,
        gohub_document_signer_info_by_index(document, i));
        printf("%s\n", gohub_document_signer_name_by_index(document, i));
        printf("%s\n", gohub_document_sign_time_by_index(document, i));
        printf("%d\n", gohub_document_sign_status_by_index(document, i));
        printf("%s\n", gohub_document_sign_error_by_index(document, i));
    }
    gohub_close_document(document);
```

- Продемонстрована техніка отримання переліку доступних носіїв та відкриття електронного ключа за індексом зі списку

```
try
    {
        printf(" CryptoMedia Name :%s", gohub_keys_list(connection));
        check_errors();
        printf(" Please enter Media index: ");
        int mediaIndex;
        scanf("%d", &mediaIndex);
        printf(" Please enter Device index: ");
        int deviceIndex;
        scanf("%d", &deviceIndex);
        printf(" Please enter password: ");
        char password[101];
        scanf("%s", password);
        if(gohub_open_private_key_by_index(connection, password, mediaIndex,
    deviceIndex))
          {
            printf("Key owner name: %s\n",
gohub_private_key_owner_name(connection));
            printf("Private key info: %s\n",
gohub_private_key_owner_info(connection));
          }

        printf("%s\n", gohub_last_error_text());
    }
    catch (...)
    {
        printf("Exception");
    }
```

– Продемонстрована техніка архіву з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД

```
printf("\n*** Testing save archive (C++) ***\n");
   try
   {
         printf(" Please enter id: ");
         char Index[1000];
         scanf("%s", Index);
         printf(" Please enter path to save: ");
         char path[1000];
         scanf("%s", path);
         if(gohub_query_and_save_document_archive(connection, Index,
"D:\\1.zip"))
        {
               printf("\n Archive saved D:\\1.zip \n");
         }

         printf("%s\n", gohub_last_error_text());
   }
   catch (...)
   {
         printf("Exception");
   }
```

5. .NET бібліотека - TMSoft.Gohub.Client.Net.dll

Програмний інтерфейс *Бібліотеки* надано набором типів, перелік яких показано нижче. Усі типи бібліотеки визначені у просторі імен *`TMSoft.Gohub.Client`*.

5.1. Перелік типів

`GohubConnection`	Клас з'єднань з Сервером Модуля Узгодження
`GohubDocument`	Клас перевізних документів
`GohubAttachment`	Клас супровідних документів
`GohubEData`	Клас супровідних документів
`GohubInformServicesDoc`	Клас документів інформаційних послуг
`GohubSigner`	Клас інформації про власників електронних ключів та авторів електронно-цифрових підписів
`GohubDocumentFilter`	Клас інформації про стани фільтру запиту документів
`GohubException`	Клас виключень
`GohubErrCode`	Перелік кодів помилок
`DispatchInfo`	Клас № Замовлення для перевізного документу
`GohubDocumentStatus`	Стани перевізного документа
`GohubFdu92`	Клас накопичувальних карток ФДУ-92
`GohubGu46`	Клас відомості користування вагонами/контейнерами ГУ-46
`GohubGu45`	Клас пам'ятки подання/прибирання вагонів та видачу/прийом контейнерів ГУ-45
`GohubClient`	Клас для інформування про помилки часу виконання
`GohubPiPackage`	Клас використовується для представлення інформації пакету ПІ
`GohubPiPackageToEData`	Клас використовується для представлення інформації про електронних даних що містяться у пакеті ПІ
`PSTDInfo`	Клас номер замовлення ПСТД
`CryptoMedia`	Клас переліку типу носіїв з ЕЦП
`CryptoDevices`	Клас назви носіїв з ЕЦП

5.2. Стани перевізного документа

Коди перевізного документа в *Бібліотеці* представлені переліком (*enum*) `DocumentStatus`, що містить набір констант.

Звідна таблиця кодів станів перевізного документа

`Unknown = 0`	Статус невідомий
`Draft = 1`	Чернетка
`Sending = 2`	Документ передається товарному касиру
`Registered = 3`	Документ переданий товарному касиру
`Reclaiming = 4`	Документ відкликається від товарного касира
`Accepted = 5`	Вантаж прийнято до перевезення
`Delivered = 6`	Вантаж прибув
`Recieved = 7`	Вантаж отримано одержувачем
`Uncredited = 8`	Документ розкредитовано товарним касиром
`RecDraft = 9`	Вантаж отримано одержувачем і редагується
`RecSending = 10`	Вантаж отримано одержувачем і переданий товарному касиру

RecReclaiming = 11	Вантаж отримано одержувачем і відкликається від товарного касира
Canceled = 12	Документ зіпсований товарним касиром

5.3. GohubConnection

Клас GohubConnection використовується для представлення з'єднань з Сервером Модуля Узгодження та містить наступні члени:

GohubConnection (конструктор)	Створити з'єднання з Сервером Модуля Узгодження. Для закриття з'єднання необхідно використати метод Dispose.
DocumentFilter (властивість)	Екземпляр класу GohubDocumentFilter, що дозволяє встановлювати фільтри для запиту документів.
SignerInfo (властивість)	Інформація про власника електронного, відкритого за допомогою метода OpenPrivateKey. У випадку, якщо пристрій електронного ключа було відключено, генерується виключення GohubException з кодом помилки GohubErrCode.gohub_private_key_is_inaccessible. У випадку, якщо електронний ключ не було відкрито за допомогою метода OpenPrivateKey, генерується виключення GohubException з кодом помилки GohubErrCode.gohub_invalid_operation
ClosePrivateKey (метод)	Закрити електронний ключ, відкритий раніше за допомогою метода OpenPrivateKey. Електронний ключ автоматично закривається також при закритті з'єднання. з Сервером Модуля Узгодження.
Dispose (метод)	Закрити з'єднання з Сервером Модуля Узгодження. Автоматично також закривається електронний ключ, якщо його було відкрито.
OpenPrivateKey (метод)	Відкрити електронний ключ. Якщо електронний ключ було відкрито уже раніше, він автоматично закривається. Електронний ключ автоматично закривається також при закритті з'єднання з Сервером Модуля Узгодження. Для закриття електронного ключа без розриву з'єднання з Сервером Модуля Узгодження можна використати метод ClosePrivateKey. Електронний ключ потрібен для підписання електронних документів. Для перевірки електронно-цифрового підпису електронний ключ не використовується та його можна не відкривати. Можливі наступні варіанти відкриття електронного ключа: • За паролем. Ключ повинен бути монтований. • За вказаним шляхом та паролю. • Ключ надано масивом байт. Необхідно вказати пароль та довжину масиву байт.
QueryDocument (метод)	Запросити документ за унікальним ідентифікатором.
QueryDocuments (метод)	Отримати впорядковану послідовність документів, з номерами ревізії більше заданої
QueryDocuments2 (метод)	Отримати впорядковану послідовність документів, з номерами ревізії більше заданої
QueryDocumentPrintableForm (метод)	Запитати друковану форму документа за його ID. Повертає масив байт.
QueryAndSaveDocumentPrintableForm (метод)	Запитати друковану форму документа за його ID. Результат запиту зберігається в файл за шляхом, що вказано в параметрі path.

SendDocument (метод)	Надіслати документ в АС «Клієнт УЗ»
SignDocument (метод)	Підписати документ електронно-цифровим підписом
ReclaimDocument (метод)	Відкликання документа з сервера СГР
DeleteDocument (метод)	Видалення документа на АС «Клієнт УЗ» за його ID
SendAttachment (метод)	Відправити супровідний документ в АС «Клієнт УЗ»
QueryAttachment (метод)	Запитати супровідний документ з системи АС «Клієнт УЗ»
QueryAttachmentWithUserData (метод)	Запитати супровідний документ з електронними даними користувача з системи АС «Клієнт УЗ»
DeleteAttachment (метод)	Видалити супровідний документ з АС «Клієнт УЗ»
QueryEData (метод)	Запитати електронні дані ПІ з системи АС «Клієнт УЗ»
QueryEDatas (метод)	Запитати електронні дані ПІ з системи АС «Клієнт УЗ» за ревізією
SendEData (метод)	Відправити електронні дані ПІ в АС «Клієнт УЗ»
AddEDataToPiPackage (метод)	Додати електронні дані ПІ в АС «Клієнт УЗ»
UpdateEData (метод)	Оновити наявні електронні дані ПІ в АС «Клієнт УЗ»
QueryPiPackage (метод)	Запитати пакет ПІ з системи АС «Клієнт УЗ»
QueryPiPackages (метод)	Запитати пакети ПІ з системи АС «Клієнт УЗ» за ревізією
SendReceivedDocument (метод)	Надіслати документ по прибуттю
QueryMPMonths (метод)	Запитати перелік ідентифікаторів місяців з АС «Месплан» у вигляді переліку
QueryMPMonthsString (метод)	Запитати перелік ідентифікаторів місяців з АС «Месплан» у вигляді рядку (номера місяців відокремлені пробілом)
QueryOrdersForMonth (метод)	Запитати заявки за номером місяцю з АС «Месплан». Можливий запит з вказанням логону та пароля.
QueryAndSaveOrdersForMonth (метод)	Запитати та зберегти заявки за номером місяцю в xml. Можливий запит з вказанням логону та пароля.
QueryEDataForAttachment (метод)	Запитати електронні дані ПІ з системи АС «Клієнт УЗ» за ідентифікатором супровідного документа
QueryAndSaveFdu92PrintableForm (метод)	Запитати друковану форму ФДУ-92 за його ID. Результат запиту зберігається в файл за шляхом, що вказано в параметрі path.
QueryFdu92PrintableForm (метод)	Запитати друковану форму ФДУ-92 за його ID. Повертає масив байт.
QueryAndSaveGu46PrintableForm (метод)	Запитати друковану форму ГУ-46 за його ID. Результат запиту зберігається в файл за шляхом, що вказано в параметрі path.
QueryGu46PrintableForm (метод)	Запитати друковану форму ГУ-46 за його ID. Повертає масив байт.
QueryAndSaveGu45PrintableForm (метод)	Запитати друковану форму ГУ-45 за його ID. Результат запиту зберігається в файл за шляхом, що вказано в параметрі path.
QueryGu45PrintableForm (метод)	Запитати друковану форму ГУ-45 за його ID. Повертає масив байт.
QueryFdu92 (метод)	Запитати ФДУ-92 за унікальним ідентифікатором.
QueryFdu92_ByNumber (метод)	Запитати ФДУ-92 за номером станції та номером накопичувальної картки
QueryFdu92s	Отримати впорядковану послідовність ФДУ-92, з номерами

(метод)	ревізії більше заданої
SignFdu92 (метод)	Підписати накопичувальну картку електронно-цифровим підписом ФДУ-92
SendFdu92 (метод)	Відправити ФДУ-92 в АС «Клієнт УЗ»
QueryGu46 (метод)	Запитати ГУ-46 за унікальним ідентифікатором.
QueryGu46s (метод)	Отримати впорядковану послідовність ГУ-46, з номерами ревізії більше заданої
SignGu46 (метод)	Підписати накопичувальну картку електронно-цифровим підписом ГУ-46
SendGu46 (метод)	Відправити ГУ-46 АС «Клієнт УЗ»
QueryGu45 (метод)	Запитати ГУ-45 за унікальним ідентифікатором.
QueryGu45s (метод)	Отримати впорядковану послідовність ГУ-45, з номерами ревізії більше заданої
QueryInformServicesDoc (метод)	Запросити документ інформаційних послуг за унікальним ідентифікатором.
QueryChangesInformServices (метод)	Отримати впорядковану послідовність документів інформаційних послуг, з номерами ревізії більше заданої
QueryDispatchInfo (метод)	Отримати переліз з інформацією про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати)
QueryGu45_ByNumber (метод)	Запитати ГУ-45 за номером станції, номером пам'ятки та датою формування
QueryGu46_ByNumber (метод)	Запитати ГУ-46 за номером станції та номером картки
GetKeysList (властивість)	Запитати перелік носіїв та ключів для відкриття електронного ключа за індексом
QueryAndSaveDocumentArchive (метод)	Запитати архів з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД

5.4. GohubDocument

Клас GohubDocument використовується для представлення електронних перевізних документів та містить наступні члени:

HasSignature (властивість)	Ознака наявності у документа електронно-цифрового підпису
Id (властивість)	Унікальний ідентифікатор документа
Revision (властивість)	Номер ревізії документа
Status (властивість)	Статус документа
SignerInfo (властивість)	Інформація про особу, що підписала документ електронно-цифровим підписом. Якщо перевірка електронно-цифрового підпису не підтверджує достовірність даних, результатом буде null. Якщо документ не містить електронно-цифрового підпису, генерується виключення GohubException з кодом помилки GohubErrCode.gohub_document_has_not_signature.
TimeStamp (властивість)	Позначка часу, отримана при підписанні документа електронно-цифровим підписом. Якщо перевірка електронно-цифрового підпису не підтверджує достовірність даних, генерується виключення GohubException з кодом помилки

	GohubErrCode.gohub_document_signature_verification_failed Якщо документ не містить електронно-цифрового підпису, генерується виключення GohubException з кодом помилки GohubErrCode.gohub_document_has_not_signature.
Attachments (властивість)	Перелік ID супровідних документів, доданих до цього документу.
MeasureEquipNum (властивість get; set;)	Відомості вагонівимірювальної техніки.
BusinessUnitNum (властивість get; set;)	Номер філіалу ПрАТ УЗ.
SetVerifiedEmptyWeightForWagon (метод)	Встановити уточнену вагу тари вагона *int wagonIndex* – індекс вагона уточнену вагу якого необхідно встановити. Індексація вагонів починається з нуля. *int weight* – уточнену вагу вагона.
GetVerifiedEmptyWeightForWagon (метод)	Отримати уточнену вагу тари вагона *int wagonIndex* – індекс вагона уточнену вагу якого необхідно отримати. Індексація вагонів починається з нуля.
ForeignNotAccept (властивість get;)	Отримати позначку повернення неприйнятих прикордонними станціями іноземних залізниць вагонів на територію України
WarrantType (властивість get; set;)	Тип підстав для отримання вантажу (0 - довіреність, 1 - наказ)
Deserialize (методы)	Десеріалізувати документ з масиву байт, потоку вводу, TextReader або XmlReader. У випадку десеріалізації з масиву байт або потоку вводу Xml-документ, повинен починатися з xml-заголовка з вказанням кодування xml-документа. Фактичне кодування символів xml-документа повинне відповідати задекларованій в заголовку. У випадку відсутності заголовка xml-документа, по замовченню вважається, що xml-документ надано у кодуванні utf-8. У випадку десеріалізації документа за допомогою TextReader або XmlReader заголовок не є обов'язковим та ігнорується як надлишкова інформація.
FromXml (методы)	Створити документ з XML-документа (XmlDocument) або його елементу (XmlElement)
FromXmlText (метод)	Створити документ з тексту XML-документа
GetXmlText (метод)	Отримати XML-текст документа
GetDataXmlText (метод)	Отримати XML-текст електронних даних документа в заданій версії ЕПД *int epdVersion* – версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4)
Load (метод)	Завантажити документ з файлу. Xml-документ, що міститься у файлі повинен починатися з xml-заголовка з вказанням кодування xml-документа. Фактичне кодування символів xml-документа повинна відповідати задекларованій в заголовку. У випадку відсутності заголовка xml-документа, по замовченню вважається, що xml-документ надано у кодуванні utf-8.
Save (метод)	Зберегти документ в файл у заданому кодуванні
SaveData (метод)	Зберегти електронні дані документа в файл у заданому кодуванні та в заданій версії ЕПД *int epdVersion* – версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4)

`Serialize` `(методы)`	Серіалізувати документ в масив байт, потоку виводу, TextWriter або XmlWriter
`SerializeData` `(методы)`	Серіалізувати електронні дані документа в масив байт, потоку виводу, TextWriter або XmlWriter в заданій версії ЕПД *`int epdVersion`* – версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4)
`ToXml` `(метод)`	Перетворити електронний перевізний документ в XML-документ (XmlDocument)
`DataToXml` `(метод)`	Перетворити електронні дані документа в XML-документ (XmlDocument) в заданій версії ЕПД *`int epdVersion`* – версія ЕПД з якою можна отримати електронні дані документа (прийняті значення 10, 11, 12, 13, 14, що відповідає версії ЕПД 1.0, 1.1, 1.2, 1.3, 1.4)
`GetOTPR` `(метод)`	Збереження актуального тексту перевізного документа у форматі xml в файл.
`GetOTPRString` `(метод)`	Отримати актуальний текст перевізного документа у кодуванні utf-8.
`Warning` `(властивість)`	Текст попередження після відправки документу
`SignerInfos` `(властивість)`	Список інформації про особу, що підписала документ електронно-цифровим підписом, яку можна отримати за індексом. Якщо перевірка електронно-цифрового підпису не підтверджує достовірність даних, результатом буде null та в значенні Error буде текст помилки. Якщо документ не містить електронно-цифрового підпису, генерується виключення GohubException з кодом помилки GohubErrCode.gohub_document_has_not_signature.

5.5. GohubAttachment

Клас `GohubAttachment` використовується для представлення інформації про супровідних документах та містить наступні члени:

`Id` `(властивість)`	Унікальний ідентифікатор супровідного документа
`Description` `(властивість)`	Текстовий опис супровідного документа, що формується з назви типу документа, його реєстраційного номера та дати.
`Name` `(властивість)`	Назва супровідного документа.
`TypeCode` `(властивість)`	Код типу супровідного документа для бланків ГУ та ЦІМ.
`SmgsTypeCode` `(властивість)`	Код типу супровідного документа для бланків СМГС та ЦІМ/СМГС згідно інформаційного керівництва СМГС.
`RegistrationNumber` `(властивість)`	Реєстраційний номер супровідного документа.
`RegistrationDate` `(властивість)`	Дата реєстрації супровідного документа.
`ValidFrom` `(властивість)`	Дата початку дії супровідного документа.
`ValidTo` `(властивість)`	Дата припинення дії супровідного документа.
`Load` `(метод)`	Створює про'єкт `GohubAttachment` по параметрам: *string typeCode* – код типу документа для бланків ГУ та ЦІМ; *string smgsTypeCode* – код типу документа для бланків СМГС та ЦІМ/СМГС;

	string name – назва документа; *string registrationNumber* – реєстраційний номер; *System.DateTime registrationDate* – дата реєстрації; *System.DateTime validFrom* – дата, з якої супровідний документ дійсний; *System.DateTime validTo* – дата, до якої супровідний документ дійсний; *string path* – шлях до відсканованого тексту документа в pdf форматі.
`Load` `(метод)`	Створює про'єкт `GohubAttachment` по параметрам: *string typeCode* – код типу документа для бланків ГУ та ЦІМ; *string smgsTypeCode* – код типу документа для бланків СМГС та ЦІМ/СМГС; *string name* – назва документа; *string registrationNumber* – реєстраційний номер; *System.DateTime registrationDate* – дата реєстрації; *System.DateTime validFrom* – дата, з якої супровідний документ дійсний; *System.DateTime validTo* – дата, до якої супровідний документ дійсний; *string path* – шлях до відсканованого тексту документа в pdf форматі. *string pathUserData* – шлях до електронних даних супровідного документа в xml форматі.
`Save` `(метод)`	Зберігає відсканований текст супровідного документа в pdf форматі. *string path* – шлях до відсканованого тексту документа в pdf форматі.
`SaveUserData` `(метод)`	Зберігає дані користувача супровідного документа в xml форматі. *string path* – шлях до електронних даних супровідного документа в xml форматі.

5.6. GohubEData

Клас `GohubEData` використовується для представлення інформації про електронних даних ПІ та містить наступні члени:

`Id` `(властивість)`	Унікальний ідентифікатор електронних даних ПІ
`Revision` `(властивість)`	Номер ревізії електронних даних ПІ.
`RevisionDate` `(властивість)`	Дата ревізії електронних даних ПІ.
`Version` `(властивість)`	Версія електронних даних ПІ.
`DocType` `(властивість)`	Код типу електронних даних ПІ: 190 рахунок-фактура, 320 пакувальний лист.
`Status` `(властивість)`	Код статусу електронних даних ПІ.
`AttachmentId` `(властивість)`	ID супровідного документа електронних даних ПІ.
`Data` `(властивість)`	Масив даних що міститься в електронних даних ПІ.
`Load` `(метод)`	Створює про'єкт `GohubEData` по параметрам: *ulong attachmentSmgsTypeCode* – код супровідного документа: 325 Рахунок-проформа (Інвойс), 380 (Інвойс) рахунок-фактура, 935 Рахунок-фактура; *string xmlPath* – шлях до xml-файлу що містять електронні дані ПІ; *string name* – назва супровідного документа; *string registrationNumber* – реєстраційний номер;

	System.DateTime registrationDate – дата реєстрації; *System.DateTime validFrom* – дата, з якої супровідний документ дійсний; *System.DateTime validTo* – дата, до якої супровідний документ дійсний; *string pdfPath* – шлях до відсканованого тексту супровідного документа в pdf форматі.
`Load` (метод)	Створює про'єкт `GohubEData` по параметрам: *ulong attachmentSmgsTypeCode* – код супровідного документа: 325 Рахунок-проформа (Інвойс), 380 (Інвойс) рахунок-фактура, 935 Рахунок-фактура; *string xmlPath* – шлях до xml-файлу що містять електронні дані
`Save` (метод)	Зберігає текст електронних даних ПІ в xml форматі. *string path* – шлях до тексту електронних даних ПІ в xml форматі.
`LoadData` (метод)	Завантажити дані з файлу у наявні електронні дані ПІ. *string path* – шлях до тексту електронних даних ПІ в xml форматі.

5.7. GohubPiPackage

Клас `GohubPiPackage` використовується для представлення інформації пакету ПІ та містить наступні члени:

`Id` (властивість)	Унікальний ідентифікатор пакета ПІ
`Revision` (властивість)	Номер ревізії пакета ПІ.
`RevisionDate` (властивість)	Дата ревізії пакета ПІ.
`ConsignmentId` (властивість)	ID перевізного документа на який посилається пакет ПІ.
`Status` (властивість)	Код статусу пакета ПІ.
`PiPackageToEDataList` (властивість)	Перелік `GohubPiPackageToEData` у пакеті ПІ.
`Data` (властивість)	Масив даних що міститься в повній інформації про пакету ПІ.
`Save` (метод)	Зберігає текст дані пакета ПІ в xml форматі. *string path* – шлях до тексту даних пакета ПІ в xml форматі.

5.8. GohubPiPackageToEData

Клас `GohubPiPackageToEData` використовується для представлення інформації про електронних даних що містяться у пакеті ПІ та містить наступні члени:

`Id` (властивість)	ID про'єкту GohubPiPackageToEData
`EDataId` (властивість)	ID електронних даних в про'єкті GohubPiPackageToEData
`EDataVersion` (властивість)	Версія електронних даних в про'єкті GohubPiPackageToEData
`PiPackageId` (властивість)	ID пакета ПІ в про'єкті GohubPiPackageToEData
`Status` (властивість)	Статус про'єкту GohubPiPackageToEData
`Note` (властивість)	Примітки про'єкту GohubPiPackageToEData

5.9. GohubFdu92

Клас `GohubFdu92` використовується для представлення інформації про накопичувальні картки ФДУ-92 та містить наступні члени:

`Id` (властивість)	Унікальний ідентифікатор накопичувальної картки ФДУ-92
`Revision` (властивість)	Номер ревізії накопичувальної картки ФДУ-92.
`Status` (властивість)	Статус накопичувальної картки ФДУ-92.
`HasSignature` (властивість)	Властивість, що перевіряє чи підписано накопичувальну картку ФДУ-92 електронно-цифровим підписом.
`SignerInfo` (властивість)	Інформація про особу, що підписала накопичувальну картку ФДУ-92 електронно-цифровим підписом.
`TimeStamp` (властивість)	Позначка часу, отримана при підписанні накопичувальної картки ФДУ-92 електронно-цифровим підписом.
`Load` (метод)	Завантажити накопичувальну картку ФДУ-92 з файлу Xml.
`Save` (метод)	Зберігає текст накопичувальної картки ФДУ-92 в Xml форматі.
`GetXmlText` (властивість)	Отримати ФДУ-92 у вигляді рядку
`ToXml` (властивість)	Перетворити ФДУ-92 в Xml-документ

5.10. GohubGu46

Клас `GohubGu46` використовується для представлення інформації про відомості користування вагонами/контейнерами ГУ-46 та містить наступні члени:

`Id` (властивість)	Унікальний ідентифікатор відомості ГУ-46
`Revision` (властивість)	Номер ревізії відомості ГУ-46
`Status` (властивість)	Статус відомості ГУ-46
`HasSignature` (властивість)	Властивість , що перевіряє, чи підписано відомість ГУ-46 електронно-цифровим підписом.
`SignerInfo` (властивість)	Інформація про особу, що підписала відомість ГУ-46 електронно-цифровим підписом.
`TimeStamp` (властивість)	Позначка часу, отримана при підписанні відомості ГУ-46 Електронно-цифровим підписом.
`Load` (метод)	Завантажити відомість ГУ-46 з файлу Xml
`Save` (метод)	Зберігає текст відомості ГУ-46 в Xml форматі
`GetXmlText` (властивість)	Отримати ГУ-46 у вигляді рядку
`ToXml` (властивість)	Перетворити ГУ-46 в Xml-документ

5.11. GohubGu45

Клас `GohubGu45` використовується для представлення інформації про пам'ятку подання/прибирання вагонів та видачу/прийом контейнерів ГУ-45 та містить наступні члени:

`Id` (властивість)	Унікальний ідентифікатор пам'ятки ГУ-45
`Revision` (властивість)	Номер ревізії пам'ятки ГУ-45
`Status` (властивість)	Статус пам'ятки ГУ-45
`HasSignature` (властивість)	Властивість, що перевіряє, чи підписано пам'ятку ГУ-45 електронно-цифровим підписом.
`SignerInfo` (властивість)	Інформація про особу, що підписала пам'ятку ГУ-45 електронно-цифровим підписом.
`TimeStamp` (властивість)	Позначка часу, отримана при підписанні пам'ятки ГУ-45 електронно-цифровим підписом.
`Save` (метод)	Зберігає текст пам'ятки ГУ-45 в Xml форматі
`GetXmlText` (властивість)	Пам'ятка про подачу / прибирання вагонів у вигляді рядку
`ToXml` (властивість)	Перетворити ГУ-45 в Xml-документ

5.12. GohubDocumentFilter

Клас `GohubDocumentFilter` використовується для керування фільтрами для запитів документів з сервера та містить наступні члени:

`WagonNumber` (властивість)	Фільтрація за номером вагона
`DocumentNumber` (властивість)	Фільтрація за номером документа
`DocumentStatus` (властивість)	Фільтрація за статусом документа
`DepartureClientCode` (властивість)	Фільтрація за кодом відправника
`DeparturePayerCode` (властивість)	Фільтрація за кодом платника по відправленню
`DepartureStationCode` (властивість)	Фільтрація за кодом станції відправлення
`ArrivalClientCode` (властивість)	Фільтрація за кодом одержувача
`ArrivalPayerCode` (властивість)	Фільтрація за кодом платника по прибуттю
`ArrivalStationCode` (властивість)	Фільтрація за кодом станції призначення
`Clear` (метод)	Очистить всі фільтри.

5.13. GohubSigner

Клас `GohubSigner` використовується для представлення інформації про власників електронних ключів, авторів електронно-цифрових підписів та містить наступні члени:

`Address` (властивість)	Адрес

Department (властивість)	Відділ
Dns (властивість)	DNS
DrfoCode (властивість)	Код ДРФО
Edrpou_code (властивість)	Код ЕДРПОУ
Email (властивість)	Електронний адреса
Establishment (властивість)	Установа
FullName (властивість)	Повне ім'я (прізвище, ім'я, по батькові)
Issuer (властивість)	Організація, що видала сертифікат справжності
IssuerSummary (властивість)	Повна інформація про організацію, що видала сертифікат справжності
Locality (властивість)	Місто
Name (властивість)	Ім'я
Phone (властивість)	Телефон
Region (властивість)	Область
SerialNumber (властивість)	Серійний номер сертифікату
StaffPost (властивість)	Поштовий індекс
Summary (властивість)	Підсумкова інформація
ToString (властивість)	Отримати рядкове представлення інформації про власника електронного ключа
Status (властивість)	Отримати статус документу на момент накладання електронно-цифрового підпису при отриманні переліку підписів Значення статусу відповідає опису статусу документу в ЕПД
Erroe (властивість)	Отримати текст помилки за індексом підпису при отриманні переліку підписів

5.14. GohubClient

Клас GohubClient використовується для підключення файлів електронних ключів та містить наступні члени:

MountFileKey (метод)	Монтувати файл електронного ключа, указав шлях до директорії та указав ідентифікатор для ключа
UnmountFileKey (метод)	Демонтувати файл електронного ключа, за його ідентифікатором
GetMountedKeys (метод)	Отримати масив ідентифікаторів монтованих файлів електронних ключів
GetMountedKeyDir (метод)	Отримати шлях до директорії в якій знаходиться файл електронного ключа за його ідентифікатором

5.15. GohubException

Клас `GohubException` використовується для інформування про помилки часу виконання з використанням механізму виключень платформи .NET Framework та містить наступні члени:

`ErrCode` (властивість)	Код помилки. Можливі значення див. в п.5.6 GohubErrCode

5.16. GohubErrCode

Перелік `GohubErrCode` уявляє собою набір кодів помилок та містить наступні члени:

`gohub_success = 0`	Операція виконана успішно
`gohub_server_connection_could_not_be_established = 1`	Неможливо встановити з'єднання з Сервером Модуля Узгодження
`gohub_server_inaccessible = 2`	Сервер Модуля Узгодження недоступний
`gohub_document_creation_failed = 3`	Неможливо створити документ
`gohub_document_query_failed = 4`	Запитати документ за ідентифікатором не вдалося
`gohub_next_document_query_failed = 5`	Запитати документ за ревізією не вдалося
`gohub_document_sending_failed = 6`	Відправити документ в АС «Клієнт УЗ» не вдалося
`gohub_document_saving_failed = 7`	Зберегти документ не вдалося
`gohub_document_loading_failed = 8`	Завантажити документ не вдалося
`gohub_private_key_could_not_be_opened = 10`	Не вдалося відкрити електронний ключ
`gohub_private_key_is_inaccessible = 11`	Електронний ключ недоступний
`gohub_document_could_not_be_signed = 12`	Не вдалося підписати документ
`gohub_document_signature_verification_failed = 13`	Перевірка електронно-цифрового підпису не підтвердила достовірність даних
`gohub_document_has_not_signature = 14`	Неприпустима операція. Документ не містить електронно-цифрового підпису
`gohub_client_is_obsolete = 15`	Версія клієнта застаріла, необхідно оновити
`gohub_server_is_obsolete = 16`	Версія сервера застаріла, необхідно оновити
`gohub_document_reclamation_failed = 17`	Відкликати документ не вдалося
`gohub_document_deletion_failed = 18`	Видалити документ не вдалося
`gohub_attachment_creation_failed = 19`	Створити супровідний документ не вдалося
`gohub_attachment_sending_failed = 20`	Відправити супровідний документ не вдалося
`gohub_attachment_query_failed = 21`	Запитати супровідний документ не вдалося
`gohub_attachment_deletion_failed = 22`	Видалити супровідний документ не вдалося
`gohub_attaching_to_document_failed = 23`	Додати супровідний документ до перевізного документу не вдалося
`gohub_detaching_from_document_failed = 24`	Відкріпити супровідний документ від перевізного документа не вдалося
`gohub_mount_of_file_key_failed = 25`	Монтування електронного ключа з файлу не вдалося.
`gohub_unmount_of_file_key_failed = 26`	Демонтувати електронного ключа з файлу не вдалося.
`gohub_enumerating_of_file_keys_failed = 27`	Отримати перелік монтованих з файлів електронних ключів не вдалося.

gohub_mounted_file_key_inaccessible = 28	Монтований з файлу електронний ключ не доступний.
gohub_edata_creation_failed = 29	Неможливо створити ЕД ПІ
gohub_edata_sending_failed = 30	Оправить ЕД ПІ не вдалося
gohub_edata_updating_failed = 31	Оновити ЕД ПІ не вдалося
gohub_edata_query_failed = 32	Запит ЕД ПІ за ідентифікатором не вдалося
gohub_next_edata_query_failed = 33	Запит ЕД ПІ за ревізією не вдалося
gohub_add_edata_to_pipackage_failed = 34	Додати ЕД в пакет ПІ не вдалося
gohub_pipackage_query_failed = 35	Запит пакета ПІ за ідентифікатором не вдалося
gohub_next_pipackage_query_failed = 36	Запит пакета ПІ за ревізією не вдалося
gohub_mp_months_query_failed = 37	Помилка при запиті переліку ідентифікаторів місяців з АС «Месплан»
gohub_orders_of_months_query_failed = 38	Помилка при запиті заявок з АС «Месплан»
gohub_programm_error = 4096	Невідома помилка в програмі
gohub_invalid_operation = 4097	Клієнтський застосунок виконав неприпустиму операцію
gohub_inform_services_doc_saving_failed = 4098	Збереження документа інформаційних послуг не вдалося
gohub_inform_services_doc_query_failed = 4099	Запит документа інформаційних послуг за ідентифікатором не вдалося
gohub_query_changes_inform_services_failed = 4100	Запит ідентифікаторів документів інформаційних послуг з ревізіями вище зазначеної не вдалося
gohub_query_next_inform_services_document_failed = 4101	Запит наступного згідно ревізії документа інформаційних послуг не вдалося
gohub_gu45_get_xml_text_failed	Отримати ГУ-45 у вигляді рядку не вдалося
gohub_gu45_to_xml_failed	Отримати ГУ-45 у вигляді XmlDocument не вдалося
gohub_gu46_get_xml_text_failed	Отримати ГУ-46 у вигляді рядку не вдалося
gohub_gu46_to_xml_failed	Отримати ГУ-46 у вигляді XmlDocument не вдалося
gohub_fdu92_get_xml_text_failed	Отримати ФДУ-92 у вигляді рядку не вдалося
gohub_fdu92_to_xml_failed	Отримати ФДУ-92 у вигляді XmlDocument не вдалося

5.17. GohubInformServicesDoc

Клас GohubInformServicesDoc використовується для представлення документів інформаційних послуг та містить наступні члени:

Id (властивість get;)	Унікальний ідентифікатор документа
Revision (властивість get;)	Номер ревізії документа
Comment (властивість get;)	Коментар документа
CreatedDate (властивість get;)	Дата створення документа
DocDate (властивість get;)	Дата документа
FileBody (властивість get;)	Тіло документа у вигляді масиву байт
FileName	Найменування документа

(властивість get;)	
Save (метод)	Зберегти документ в файл
SaveXml (метод)	Зберегти документ в файл формату xml
IsEmpty (властивість get;)	Значення чи порожній документ

5.18. GohubDispatchInfo

Клас GohubDispatchInfo с використовується для представлення переліку інформації про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати):

Date (поле)	Дата
Description (поле)	Повний опис
Number (поле)	Номер замовлення
Type (поле)	Тип відправки
WagOwner (поле)	Власник вагону
IsEmpty (поле)	Стан вагону порожній чи завантажений

5.19. PSDTInfo

Клас PSDTInfo с використовується для представлення переліку інформації про номер замовлення ПСТД.

ArrivaDate (поле)	Дата прибуття
DepartureDate (поле)	Дата відправлення
Number (поле)	Номер замовлення
ReqDate (поле)	Дата замовлення

5.20. CryptoMedia

Клас CryptoMedia використовується для представлення переліку типу носіїв для вибору елетронного ключа з переліку.

devices (перелік)	Перелік носіїв CryptoDevices
index (поле)	Індекс типу носія
name (поле)	Назва типу носія

5.21. CryptoDevices

Клас CryptoDevices використовується для представлення переліку носіїв в переліку типів носіїв.

index (поле)	Індекс носія в переліку типів
name (поле)	Назва носія в переліку типів

5.22. Приклади використання

а) Завантаження документа з файлу та передача в АС «Клієнт УЗ»

Продемонструємо завантаження документа з файлу та передачу його в АС «Клієнт УЗ». При цьому використовуємо електронний ключ для накладення електронно-цифрового підпису.

```
        static string LoadAndSendDocument(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            string path,            // шлях до файлу документа
            string password)    //Пароль до ключу. Якщо не задано, документ не
буде підписано
        {
            try
            {
                // Завантажити документ з файлу
                GohubDocument document = GohubDocument.Load(path);
                connection.OpenPrivateKey(password);
                Console.WriteLine("Private key info: {0}",
connection.SignerInfo);
                try
                {
                    connection.SignDocument(document);
                }
                catch
                {
                    throw;
                }
                finally
                {
                    connection.ClosePrivateKey();
                }

                connection.SendDocument(document);
                // Документ передано успішно

                Console.WriteLine("Document is sended: id={0} revision={1}",
document.Id, document.Revision);
                // Перевірка попереджень після відправки документу
                string warning = document.Warning;
                if (!string.IsNullOrEmpty(warning))
                    Console.WriteLine("Warning in document: {0}", warning);

                // Спробуємо запросити цей документ за ідентифікатором
                    document = connection.QueryDocument(document.Id);
                Console.WriteLine(document.GetXmlText());
                return document.Id;
            }
            catch (Exception e)
            {
```

```
                Console.WriteLine(e.ToString());
            }
            return null;
        }
```

б) Запит документів з АС «Клієнт УЗ»

Цей приклад демонструє техніку поступового запиту документів з АС «Клієнт УЗ» за номерами ревізій.

```
        static int QueryAndSaveDocuments(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            int startRevision,  //номер ревізії з якої отримувати перевізні
документи
            int maxCount,       //Максимальна кількість документів , що буде
запитано
            string targetFolder)//тека куди зберігати отримані документи
        {
            int lastRevision = startRevision;
            int i = 0;
            var encoding = Encoding.GetEncoding(1251);
            foreach (var document in connection.QueryDocuments(lastRevision))
            {
                // Запит документа виконано успішно
                Console.WriteLine("Document received: id={0} revision={1}",
document.Id, document.Revision);
                // Вивести документ на екран
                Console.WriteLine(document.GetXmlText());
                try
                {
                    string path = string.Format("{0}\\{1}.xml", targetFolder,
document.Id);
                    document.Save(path, encoding);
                    Console.WriteLine("Document saved to file: path='{0}'",
path);
                }
                catch (Exception e)
                { // Помилка при збереженні документа
                    Console.WriteLine(e.ToString());
                }

                // Перевірка підпису
                if (document.HasSignature)
                {
                    Console.WriteLine("SignerInfo={0}\nTimeStamp={1}",
document.SignerInfo, document.TimeStamp);
                }

                // Запам'ятати ревізію останнього документа
                lastRevision = document.Revision;

                //збільшуємо лічильник отриманих документів
                i++;
                if (i == maxCount)
                    break;
            }
            // Повернення функцією номера останньої обробленої ревізії
            return lastRevision;
        }
```

в) Інші приклади використання

В інсталяційному пакеті Клієнта Модуля Узгодження представлені також інші приклади використання, з якими більш детально можна ознайомитися, заглянув безпосередньо

в програмний код (тека Клієнт Модуля Узгодження\samples\ gohub.client.Test.cs). Тут ми коротко опишемо інші функції, надані у прикладах:

- Продемонстрована техніка поступового запиту документів ФДУ-92 з АС «Клієнт УЗ» за номерами ревізій

```
        static ulong QueryAndSaveFdu92s(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            ulong startRevision,  //номер ревізії з якої отримувати перевізні
документи
            int maxCount,       //Максимальна кількість документів , що буде
запитано
            string targetFolder)//тека куди зберігати отримані документи
```

- Продемонстрована техніка поступового запиту документів ЕД попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
        static ulong QueryAndSavePiPackages(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            ulong startRevision,  //номер ревізії з якої отримувати перевізні
документи
            int maxCount,       //Максимальна кількість документів , що буде
запитано
            string targetFolder)//тека куди зберігати отримані документи
```

- Продемонстрована техніка поступового запиту пакетів попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
        static ulong QueryAndSavePiPackages(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            ulong startRevision,  //номер ревізії з якої отримувати перевізні
документи
            int maxCount,       //Максимальна кількість документів , що буде
запитано
            string targetFolder)//тека куди зберігати отримані документи
```

- Продемонстрована техніка додання документа ЕД в пакет попереднього інформування

```
        static string AddEDataToPiPackage(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            string piPackageId, // ID пакета ПІ (попереднього інформування) в що
будуть додаватися ЕД (електронні дані)
            string edataPath,   // шлях до xml-файлу з ЕД
            uint edataType)     // код ЕД: 190 рахунок-фактура, 320 пакувальний
лист
```

- Продемонстрована техніка оновлення документа ЕД, що міститься на сервері АС «Клієнт УЗ»

```
        static ulong UpdateEData(
            GohubConnection connection, // з'єднання з Сервером Модуля Узгодження
            string edataId,     // ID документа ЕД що буде змінюватись
            string edataPath)   // шлях до xml-файлу з ЕД
```

- Продемонстрована техніка отримання переліку з інформацією про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати)

```
        try
            {
                Console.WriteLine("Please enter ESR code station from:");
```

```
            var from_esr = Console.ReadLine();
            Console.WriteLine("Please enter ESR code station to:");
            var to_esr = Console.ReadLine();
            var list = connection.QueryDispatchInfo(from_esr, to_esr);
            if (list != null && list.Count > 0)
            {
                Console.WriteLine("list loaded.\n");
                foreach (var item in list)
                {
                    Console.Write(item.Description + "\n");
                    Console.Write(item.Number + "\n");
                    Console.Write(item.Date + "\n");
                    Console.Write(item.Type + "\n");
                    Console.Write(item.WagOwner + "\n");
                    Console.Write(item.IsEmpty + "\n");
                }
            }
            else
                Console.WriteLine("list is empty");
        }
        catch (Exception ex)
        {
            Console.WriteLine(ex.Message);
        }
```

- Продемонстрована техніка отримання переліку з інформацією про підписи ПД

```
        try
         {
             Console.WriteLine("Please enter document Id");
             string id = Console.ReadLine();

             Console.WriteLine("Test is starting...");
             GohubDocument doc = connection.QueryDocument(id);

             foreach (var item in doc.SignatureInfos)
             {
                    if (item.Error == null)
                          Console.WriteLine(item.Summary + "\n" +
                          item.Status.ToString());
                    else
                          Console.WriteLine(item.Error);
              }
          }
          catch (Exception ex)
          {
              Console.WriteLine(ex.Message);
          }
```

- Продемонстрована техніка отримання переліку та відкриття електронного ключа

```
        Console.WriteLine(" Starting get keys list list (.Net)");

        var list = connection.GetKeysList;
             if (list != null && list.Count > 0)
             {
                 Console.WriteLine("list loaded.\n");
                 foreach (var item in list)
                 {
                     Console.Write("NameMedia :" + item.name + "\t");
                     Console.Write("IndexMedia :" + item.index + "\t\n");
                     foreach (var i in item.devices)
                     {
                         Console.Write("\tNameDevice :" + i.name + "");
                         Console.Write("\tIndexDevice :" + i.index + "\n");
                     }
                 }
             }
```

```
            else
                Console.WriteLine("list is empty");
            Console.WriteLine("Enter Media index");
            int mediaIndex = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter Device index");
            int deviceIndex = int.Parse(Console.ReadLine());
            Console.WriteLine("Enter password");
            string password = Console.ReadLine();
            connection.OpenPrivateKey(password, mediaIndex, deviceIndex);
            Console.WriteLine(connection.SignerInfo);
```

- Продемонстрована техніка отримання переліку та відкриття електронного ключа

```
try
  {
      Console.WriteLine(" Testing query archive byte (.Net)");
      Console.WriteLine("Please enter id:");
      var id = Console.ReadLine();
      connection.QueryAndSaveDocumentArchive(id,
string.Format("D:\\{0}.zip", id))

     Console.WriteLine(string.Format(" Done, file saved
D:\\{0}.zip", id));

      Console.WriteLine(" ----------------------------- ");

  }
  catch (Exception ex)
  {
       Console.WriteLine(ex.Message);
   }
```

6. COM/OLE бібліотека

6.1. Стани перевізного документа

Коди перевізного документа в *Бібліотеці* представлені переліком (*enum*) `GohubDocumentStatus`, що містить набір констант.

Звідна таблиця кодів станів перевізного документа

`gohub_document_status_unknown = 0`	Статус невідомий
`gohub_document_status_draft = 1`	Чернетка
`gohub_document_status_sending = 2`	Документ передається товарному касиру
`gohub_document_status_registered = 3`	Документ переданий товарному касиру
`gohub_document_status_reclaiming = 4`	Документ відкликається від товарного касира
`gohub_document_status_accepted = 5`	Вантаж прийнято до перевезення
`gohub_document_status_delivered = 6`	Вантаж прибув
`gohub_document_status_recieved = 7`	Вантаж отримано одержувачем
`gohub_document_status_uncredited = 8`	Документ розкредитовано товарним касиром
`gohub_document_status_recieved_draft = 9`	Вантаж отримано одержувачем і редагується
`gohub_document_status_recieved_sending = 10`	Вантаж отримано одержувачем і переданий товарному касиру
`gohub_document_status_recieved_reclaiming = 11`	Вантаж отримано одержувачем і відкликається від товарного касира
`gohub_document_status_canceled = 12`	Документ зіпсований товарним касиром
`gohub_document_status_locked = 13`	Документ заблокований

Більш детальний опис програмного інтерфейсу бібліотеки можна отримати з файлу gohub.client.errors.h (Додаток Б).

6.2. Ідентифікатори інтерфейсів

Звертатися до функцій бібліотеки можна з використанням технології COM/OLE. ім'я COM-компонента (ProgID): *"TMSoft.GohubClient"*, Ідентифікатор (CLSID): `CF908D67-DAF6-43B0-9621-1DD417CFF3D7`. Компонент надає наступні інтерфейси:

ім'я	Ідентифікатор
IGohubClient	ABDA6C07-5320-4F28-B995-FADE037D0A82
IGohubDocument	0D5D6225-8E07-46E2-8B8D-9C0966481994
IGohubAttachment	044603D2-574E-463C-87EC-DCD98C30F319
IGohubEData	685B21FA-43A7-4ACC-9A57-AB7AC332942C
IGohubPiPackage	62C21013-07D6-4d9d-83C4-9FA6E770B2EA
IGohubPiPackageToEData	88AAFC89-0426-4551-BD1C-F57F63D0C335
IGohubConnection	E9967B9D-1141-47BA-A5C4-573FB02DB396
IGohubSignerInfo	B0E1F579-5836-4E97-9340-58B092418947
IGohubError	F1077417-21D9-4871-84A2-9F89525E7214
IGohubInformServicesDocument	0af49642-b12d-4fdb-b363-a7497cc78462
IGohubDispatchInfo	0669AB3B-8E1B-4161-8A2E-244D5A62029A
IGohubFdu92	EB03BE7D-48B7-4AFD-8A94-A5012D844A17
IGohubGu46	78DCD121-84B7-4D41-B15C-F9F7677A3519
IGohubGu45	C2F92D3C-295A-4453-B4CF-B33D2C8966E3
IGohubPSTDInfo	B8FFED27-656B-48C0-AD6F-8EBE09F8F8E8

Детальний опис продемонстровано в додатку В

6.3. Інтерфейс IGohubClient

Його ідентифікатор: ABDA6C07-5320-4F28-B995-FADE037D0A82

Детальний опис продемонстровано в додатку В

Методи:

Назва	Параметри	Значення, що повертається
GetLastError		IgohubError
Connect	host як String, port як Long	IGohubConnection
CreateDocument	xmlText як String	IGohubDocument
LoadDocument	path як String	IGohubDocument
LoadAttachment	typeCode як String, name як String, regNumber як String, regDate як String, validFrom як String, validTo як String, path як String	IGohubAttachment
LoadAttachmentWithUserData	typeCode як String, name як String, regNumber як String, regDate як String, validFrom як String, validTo як String, path як String, pathUserData як String	IGohubAttachment
LoadSmgsAttachment	smgsTypeCode як String, name як String, regNumber як String, regDate як String, validFrom як String, validTo як String, path як String	IGohubAttachment
LoadSmgsAttachmentWithUserData	smgsTypeCode як String, name як String, regNumber як String, regDate як String, validFrom як String, validTo як String, path як String, pathUserData як String	IGohubAttachment
MountFileKey	keyId як String, keyDir як String	Boolean
UnmountFileKey	keyId як String	Boolean
QueryMountedKeys		Long
GetMountedKeyId	index як Long	String
GetMountedKeyDir	index як Long	String
LoadEData	codeType як UInt, xmlPath як String, name як String, regNumber як String, regDate як String, validFrom як String, validTo як String, pdfPath як String	IGohubEData
LoadEDataSimple	codeType як UInt, xmlPath як String	IGohubEData
LoadFdu92	ident як String,	IGohubFdu92

	path як String	
LoadGu46	ident як String, path як String	IGohubGu46

6.4. Інтерфейс IGohubDocument

Його ідентифікатор: 0D5D6225-8E07-46E2-8B8D-9C0966481994

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Text	String
Revision	Long
Status	Long
HasSignature	Boolean
Signer	IGohubSignerInfo
SignTime	String
AttachmentsCount	Long
MeasureEquipNum	String [get, put]
BusinessUnitNum	String [get, put]
ForeignNotAccept	Boolean
WarrantType	LONG[get, put]
OTPRString	String [get]
Warning	String [get]
SignerCount	Long

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String, codePage як Long	Boolean
SaveData	Path як String, codePage як Long, epdVersion як Long	Boolean
DataText	epdVersion як Long	String
Sign	connection як IGohubConnection	Boolean
Send	connection як IGohubConnection	Boolean
GetAttachmentIdByIndex	index як Long	String
SendReceived	connection як IgohubConnection docId як String	Boolean
set_VerifiedEmptyWeightForWagon, SetVerifiedEmptyWeightForWagon	wagonIndex як LONG weight як LONG	
get_VerifiedEmptyWeightForWagon, GetVerifiedEmptyWeightForWagon	wagonIndex як LONG	LONG
GetOTPR	Path як String	Boolean
SignerByIndex	index як Long	String
SignStatusByIndex	index як Long	String
SignTimeByIndex	index як Long	String
SignErrorByIndex	index як Long	String

6.5. Інтерфейс IGohubAttachment

Його ідентифікатор: 044603D2-574E-463C-87EC-DCD98C30F319

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
TypeCode	String
SmgsTypeCode	String
Name	String
Description	String
RegNumber	String
RegDate	String
ValidFrom	String
ValidTo	String

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
Send	connection як IGohubConnection	Boolean
SaveUserData	Path як String	Boolean

6.6. Інтерфейс IGohubEData

Його ідентифікатор: 685B21FA-43A7-4ACC-9A57-AB7AC332942C

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Revision	UInt64
RevisionDate	String
Version	String
DocType	UInt
Status	Int
AttachmentId	String

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
LoadData	Path як String	Boolean
Send	connection як IGohubConnection	Boolean
Update	connection як IGohubConnection	Boolean

6.7. Інтерфейс IGohubPiPackage

Його ідентифікатор: 62C21013-07D6-4d9d-83C4-9FA6E770B2EA

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Revision	UInt64
RevisionDate	String
ConsignmentId	String
Status	Int
PiPackageToEDataCount	Int

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
PiPackageToEData	Index як Int	IGohubPiPackageToEData

6.8. Інтерфейс IGohubPiPackageToEData

Його ідентифікатор: 88AAFC89-0426-4551-BD1C-F57F63D0C335

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
EDataId	String
PiPackageId	String
Note	String
Status	Int
EDataVersion	String

6.9. Інтерфейс IGohubConnection

Його ідентифікатор: E9967B9D-1141-47BA-A5C4-573FB02DB396

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Signer	IGohubSignerInfo

Методи:

Назва	Параметри	Значення, що повертається
Close		
QueryDocument	docId як String	IGohubDocument
QueryNextDocument	revision як Long	IGohubDocument
QueryNextDocument2	revision як Long	IGohubDocument
OpenPrivateKey	password як String	Boolean
OpenPrivateKeyFromPath	password як String, path як String	Boolean
ReclaimDocument	docId як String	Boolean
DeleteDocument	docId як String	Boolean
QueryAttachment	attachmentId як String	IgohubAttachment
QueryAttachmentWithUserData	attachmentId як String	IgohubAttachment
DeleteAttachment	attachmentId як String	Boolean
FilterByDocumentStatus	newVal як Long	Властивості – повертає або встановлює параметр (в залежності від застосування)
FilterByDocumentNumber	newVal як String	
FilterByWagonNumber	newVal як String	
FilterByDepartureClientCode	newVal як String	
FilterByDeparturePayerCode	newVal як String	
FilterByDepartureStationCode	newVal як String	
FilterByArrivalClientCode	newVal як String	
FilterByArrivalPayerCode	newVal як String	
FilterByArrivalStationCode	newVal як String	
ClearAllFilters		Boolean
QueryAndSaveDocumentPrintableForm, SaveDocumentPrintableForm	docId як String, path як String	Boolean
QueryEData	eDataId як String	IGohubEData
QueryNextEData	revision як Long	IGohubEData
QueryPiPackage	piPackageId як String	IGohubPiPackage
QueryNextPiPackage	revision як Long	IGohubPiPackage
AddEDataToPiPackage	eData як IGohubEData piPackageId як String	IGohubPiPackage
GetMPMonths		String
QueryAndSaveOrdersForMonth	month як String, path	Boolean

	як String	
QueryAndSaveOrdersForMonthWithRelogin, SaveOrdersForMonthWithRelogin	month як String, login як String, password як String, path як String	Boolean
QueryEDataForAttachment	attachmentId як String	IGohubEData
QueryInfServsDoc	docId як Long	IGohubInformServicesDocument
QueryNextInfServsDoc	revision як Long	IGohubInformServicesDocument
QueryFdu92ByNumber	registration_esr як String, registration_num як String	GohubFdu92
QueryDispatchInfo	dispatch_esr як String, arrival_esr як String	IGohubDispatchInfo
QueryGu45ByNumber	registration_esr як String, registration_num як String, registration_date як String	GohubGu45
QueryGu46ByNumber	registration_esr як String, registration_num як String	GohubGu46
QueryAndSaveFdu92PrintableForm	docId як String, path як String	Boolean
QueryAndSaveGu45PrintableForm	docId як String, path як String	Boolean
QueryAndSaveGu46PrintableForm	docId як String, path як String	Boolean
RejectFdu92	id як String	Boolean
RejectGu46	id як String	Boolean
QueryAndSaveDocumentArchive	docId як String, path як String	Boolean

6.10. Інтерфейс IGohubSignerInfo

Його ідентифікатор: B0E1F579-5836-4E97-9340-58B092418947

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Name	String
Subject	String

6.11. Інтерфейс IGohubError

Його ідентифікатор: F1077417-21D9-4871-84A2-9F89525E7214

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Code	Long
Title	String
Text	String

6.12. Інтерфейс IGohubFdu92

Його ідентифікатор: EB03BE7D-48B7-4AFD-8A94-A5012D844A17

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Revision	Long
Status	Long
Text	String
HasSignature	Boolean
Signer	IGohubSignerInfo
SignTime	String

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
Send	connection як IGohubConnection	Boolean
Sign	connection як IGohubConnection	Boolean

6.13. Інтерфейс IGohubGu46

Його ідентифікатор: 78DCD121-84B7-4D41-B15C-F9F7677A3519

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Revision	Long
Status	Long
Text	String
HasSignature	Boolean
Signer	IGohubSignerInfo
SignTime	String

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
Send	connection як IGohubConnection	Boolean
Sign	connection як IGohubConnection	Boolean

6.14. Інтерфейс IGohubGu45

Його ідентифікатор: 78DCD121-84B7-4D41-B15C-F9F7677A3519

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	String
Revision	Long
Status	Long
Text	String
HasSignature	Boolean
Signer	IGohubSignerInfo
SignTime	String

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean

6.15. Інтерфейс IGohubInformServicesDocument

Його ідентифікатор: 0af49642-b12d-4fdb-b363-a7497cc78462

Детальний опис продемонстровано в додатку В

Властивості:

Назва	Значення, що повертається
Id	UInt64

Revision	UInt64
FileName	String
Comment	String
CreatedDate	String
DocDate	String
IsEmpty	Boolean

Методи:

Назва	Параметри	Значення, що повертається
Close		
Save	Path як String	Boolean
SaveXml	Path як String	Boolean

6.16. Інтерфейс IDispatchInfo

Його ідентифікатор: 0669AB3B-8E1B-4161-8A2E-244D5A62029A

Детальний опис продемонстровано в додатку В

Методи:

Назва	Параметри	Значення, що повертається
Close		
WagOwnerByIndex	index як Long	String
TypeByIndex	index як Long	String
DateByIndex	index як Long	String
DescriptionByIndex	index як Long	String
NumberByIndex	index як Long	String
IsEmptyByIndex	index як Long	String
Count		UInt64

6.17. Інтерфейс IPSTDInfo

Його ідентифікатор: B8FFED27-656B-48C0-AD6F-8EBE09F8F8E8

Детальний опис продемонстровано в додатку В

Методи:

Назва	Параметри	Значення, що повертається
Close		
ArrivelDateByIndex	index як Long	String
DepartureByIndex	index як Long	String
NumberByIndex	index як Long	String
ReqDateByIndex	index як Long	String
Count		UInt64

6.18. Приклади використання

Усі наведені приклади використання Модуля Узгодження для COM/OLE бібліотеки продемонстровані мовою Visual Basic.

а) Вивід на екран інформації про помилку

Наводимо приклад отримання інформації про помилку. Визначимо функцію `PrintError`, що виводить на екран детальну інформацію про помилку, включно з інформацією про вкладені помилки, якщо вони наявні. Ця функція стане в нагоді у наступних прикладах.

```
    Sub PrintError(ByVal client As Object)
        Dim Err As Object = client.GetLastError()
        Console.WriteLine(Err.Code)
        Console.WriteLine(Err.Title)
        Console.WriteLine(Err.Text)
        Marshal.ReleaseComObject(Err)
    End Sub
```

б) Завантаження документа з файлу та передача в АС «Клієнт УЗ»

Продемонструємо завантаження документа з файлу та передачу його в АС «Клієнт УЗ». При цьому використовуємо електронний ключ для накладення електронно-цифрового підпису. Для виводу інформації про можливі помилки використовуємо функцію `PrintError`, визначену раніше в п.6.15.а.

```
    Sub LoadAndSendDocument(ByRef client As Object, ByRef connection As Object, _
                            ByVal path As String, _
                            ByVal password As String)
        ' Завантажити документ з файлу
        Dim document As Object = client.LoadDocument(path)
        If (document Is Nothing) Then
            ' Помилка при завантаженні документа з файлу
            PrintError(client)
            Return
        End If
        ' Документ завантажено успішно
        Console.WriteLine("Document is loaded")

        ' Підписати документ
        If Not connection.OpenPrivateKey(password) Then
            ' Не вдалося отримати доступ до закритого ключу
            PrintError(client)
            CloseAndRelease(document)
            Return
        End If
        Console.WriteLine("Signing document . . .")
        Console.WriteLine("Private key info: name={0}, Subject={1}",
connection.Signer.Name, connection.Signer.Subject)

        If Not document.Sign(connection) Then
            ' Не вдалося отримати доступ до закритого ключу
            PrintError(client)
            CloseAndRelease(document)
            Return
        End If

        ' Передати документ в АС "Клієнт УЗ" засобами Модуля Узгодження
        If Not document.Send(connection) Then
            ' Помилка при передачі документа
            PrintError(client)
            CloseAndRelease(document)
            Return
        End If

        ' Документ передано успішно
        Console.WriteLine("Document is sended: id={0} revision={1}", document.Id,
document.Revision)
```

```
        'Перевірка попередження після відправки документа
        Dim warning As String
        warning = documnet.Warning
        If Not String.IsNullOrEmpty(warning) Then
            Console.WriteLine("Warning: {0}", warning)
        End If

        ' Зберігаємо ідентифікатор документа для подальшого використання
        Dim id As String = document.Id
        CloseAndRelease(document)

        ' Спробуємо запросити цей документ за ідентифікатором
        document = connection.QueryDocument(id)
        If (document Is Nothing) Then
            ' Помилка при отриманні документа
            PrintError(client)
            Return
        End If

        Console.WriteLine("Document is queried.")
        Console.WriteLine(document.Text)

        CloseAndRelease(document)
    End Sub
```

в) Запит документів з АС «Клієнт УЗ»

Цей приклад демонструє техніку поступового запиту документів їх АС «Клієнт УЗ» за номерами ревізій. Окрім того показано зміни кодової сторінки документа та збереження запитаних документів у файли. Окрім того, виконується перевірка електронно-цифрового підпису. Для виводу інформації про помилки використовується функція `PrintError`, визначена раніше в п.6.15.а

```
    Function QueryAndSaveDocuments(ByRef client As Object, ByRef connection As
Object, _
                                   ByVal startRevision As Integer, _
                                   ByVal maxCount As Integer, _
                                   ByVal targetFolder As String _
                                   ) As Integer
        Dim lastRevision As Integer = startRevision
        Dim document As Object
        For count As Integer = 0 To maxCount
            ' Запит наступного документа за ревізією
            document = connection.QueryNextDocument(lastRevision)
            If (document Is Nothing) Then
                PrintError(client)
                Exit For
            End If
            ' Запит документа виконано успішно
            ' Вивести документ на екран
            Console.WriteLine("Document queried: Id={0}, Rev={1}", document.Id,
document.Revision)

            Dim path As String = targetFolder + "\" + document.Id + ".xml"
            If (document.Save(path, 866)) Then
                Console.WriteLine("Document saved to file: path={0}", path)
            Else
                'Помилка при збереженні документа
                PrintError(client)
            End If
```

```
            ' Перевірка підпису
            If (document.HasSignature) Then
                Console.WriteLine("Signer: Name={0}, Subject={1}. TimeStamp={2}",
document.Signer.Name, document.Signer.Subject, document.SignTime)
            End If
            ' Запам'ятати ревізію останнього документа
            lastRevision = document.Revision
            CloseAndRelease(document)
        Next
        Return lastRevision
    End Function
```

г) Інші приклади використання

В інсталяційному пакеті Клієнта Модуля Узгодження представлені також інші приклади використання, з якими більш детально можна ознайомитися, заглянув безпосередньо в програмний код (тека Клієнт Модуля Узгодження\samples\ Gohub.Client.COM.Test.VB). Тут ми коротко опишемо інші функції, надані у прикладах:

- Продемонстрована техніка поступового запиту документів ФДУ-92 з АС «Клієнт УЗ» за номерами ревізій

```
    Function QueryAndSaveFdu92s(ByRef client As Object, ByRef connection As
Object, _
                                ByVal startRevision As Integer, _
                                ByVal maxCount As Integer, _
                                ByVal targetFolder As String _
                                ) As Integer
```

- Продемонстрована техніка поступового запиту документів ЕД попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
    Function QueryAndSaveEDatas(ByRef client As Object, ByRef connection As
Object, _
                                ByVal startRevision As ULong, _
                                ByVal maxCount As Integer, _
                                ByVal targetFolder As String _
                                ) As ULong
```

- Продемонстрована техніка поступового запиту пакетів попереднього інформування з АС «Клієнт УЗ» за номерами ревізій

```
    Function QueryAndSavePiPackages(ByRef client As Object, ByRef connection As
Object, _
                                    ByVal startRevision As ULong, _
                                    ByVal maxCount As Integer, _
                                    ByVal targetFolder As String _
                                    ) As ULong
```

- Продемонстрована техніка додання документа ЕД в пакет попереднього інформування

```
    Function AddEDataToPiPackage(ByRef client As Object, ByRef connection As
Object, _
                                 ByVal piPackageId As String, _
                                 ByVal edataPath As String, _
                                 ByVal edataType As Integer _
                                 ) As String
```

- Продемонстрована техніка оновлення документа ЕД, що міститься на сервері АС «Клієнт УЗ»

```
Function UpdateEData(ByRef client As Object, ByRef connection As Object, _
                     ByVal edataId As String, _
                     ByVal edataPath As String _
                     ) As ULong
```

– Продемонстрована техніка отримання переліку з інформацією про замовлення на погодження перевезення за даними календаря планування перевезень зернових вантажів(за останні 5 днів від поточної дати)

```
Try
      Console.Write(" Please enter ESR number of start station: ")
      Dim start_esr = Console.ReadLine()
      Console.Write(" Please enter ESR number of end station: ")
      Dim end_ser = Console.ReadLine()
      Dim doc = connection.QueryDispatchInfo(start_esr, end_ser)
      PrintError(client.GetLastError())
      If doc IsNot Nothing Then
          Dim count = doc.Count()
          If count > 0 Then
              Dim i = 0
              While i < count
                  Console.WriteLine(doc.DescriptionByIndex(i))
                  Console.WriteLine(doc.NumberByIndex(i))
                  Console.WriteLine(doc.DateByIndex(i))
                  Console.WriteLine(doc.TypeByIndex(i))
                  Console.WriteLine(doc.WagOwnerByIndex(i))
                  Console.WriteLine(doc.IsEmptyByIndex(i))
                  i += 1
              End While
          End If
          doc.Close()
      End If
  Catch ex As Exception
      Console.WriteLine(ex.Message)
      Console.ReadKey()
```

– Продемонстрована техніка отримання переліку з інформацією про підписи в перевізному документі

```
Try
      Dim id As Object
      Dim sign As Object
      Dim count As Integer
      Dim time As Object
      Dim status As Integer
      Dim err As Object
      Dim id = Console.ReadLine()
      Dim doc = connection.QueryDocument(id)
      PrintError(client.GetLastError())
      If (doc Is Nothing) Then
          Exit Sub
      End If
      id = doc.Id
      count = doc.SignerCount
      sign = doc.SignerByIndex(0)
      time = doc.SignTimeByIndex(0)
      status = doc.SignStatusByIndex(0)
      err = doc.SignErrorByIndex(0)
      If (sign Is Nothing) Then
            Console.WriteLine(err.ToString())
            Exit Sub
      End If
```

```
            Console.WriteLine("Document queried: Id={0} SignName={1}  Count={2}
            SignTime={3}  SignStatus{4}", id, sign.Name, count, time, status)
            doc.Close()
        Catch ex As Exception
                Console.WriteLine(ex.Message)
                Console.ReadKey()
```

– Продемонстрована техніка отримання переліку носіїв з ключами та відкриття електронного ключа за індексом

```
Try
                Dim keysList = connection.get_KeysList()
                PrintError(client.GetLastError())
                If keysList IsNot Nothing Then
                    Console.WriteLine(keysList)
                End If
                Dim mediaIndex As Integer
                Dim deviceIndex As Integer
                Dim password As String
                Console.WriteLine("Enter Media index")
                mediaIndex = Console.ReadLine()
                Console.WriteLine("Enter device index")
                deviceIndex = Console.ReadLine()
                Console.WriteLine("Enter password")
                password = Console.ReadLine()
                connection.OpenPrivateKeyByIndex(password, mediaIndex,
   deviceIndex)
                Console.WriteLine("Private key info: name={0}, Subject={1}",
   connection.Signer.Name, connection.Signer.Subject)
            Catch ex As Exception
                Console.WriteLine(ex.Message)
                Console.ReadKey()
            End Try
```

– Продемонстрована техніка архіву з файлами для перевірки накладених КЕП на кожному з етапів оформлення ПД

```
   Console.WriteLine("*** Testing save archive ***")
            Try

                Dim Index As Integer
                Dim path As String
                Console.WriteLine("Enter id of document")
                Index = Console.ReadLine()
                Console.WriteLine("Enter path")
                path = Console.ReadLine()
                connection.QueryAndSaveDocumentArchive(Index, path)
                Console.WriteLine("Archive saved to {0}", path)
            Catch ex As Exception
                Console.WriteLine(ex.Message)
                Console.ReadKey()
            End Try
```

ДОДАТОК А. Файл *gohub.client.h*

```
#ifndef GOHUB_CLIENT_H_
#define GOHUB_CLIENT_H_

#include "gohub.client.errors.h"

#ifdef __cplusplus
extern "C" {
#endif

////////////////////
// Загальні типи //
////////////////////
typedef int GohubBool;
typedef unsigned short GohubWChar;
typedef struct GohubConnection GohubConnection;
typedef struct GohubDocument GohubDocument;
typedef struct GohubAttachment GohubAttachment;
typedef struct GohubEData GohubEData;
typedef struct GohubPiPackageToEData GohubPiPackageToEData;
typedef struct GohubPiPackage GohubPiPackage;
typedef struct GohubFdu92 GohubFdu92;
typedef struct GohubGu46 GohubGu46;
typedef struct GohubGu45 GohubGu45;
typedef struct GohubGu27 GohubGu27;
typedef struct GohubInformServicesDoc GohubInformServicesDoc;
typedef struct GohubDispatchInfo GohubDispatchInfo;
typedef struct GohubPSTDInfo GohubPSTDInfo;
////////////////////////////////////
// Статуси перевізного документа //
////////////////////////////////////
typedef enum GohubDocumentStatus
{
  gohub_document_status_unknown,
  gohub_document_status_draft,
  gohub_document_status_sending,
  gohub_document_status_registered,
  gohub_document_status_reclaiming,
  gohub_document_status_accepted,
  gohub_document_status_delivered,
  gohub_document_status_recieved,
  gohub_document_status_uncredited,
  gohub_document_status_recieved_draft,
  gohub_document_status_recieved_sending,
  gohub_document_status_recieved_reclaiming,
  gohub_document_status_canceled,
  gohub_document_status_locked,
} GohubDocumentStatus;
////////////////////////////////////
// Операції перевізного документа   //
////////////////////////////////////
typedef enum UzRwcDocStatus
{
  None = 0,
  Project = 1,
  Accept = 2,
  Resend = 3,
  Arrive = 4,
  Review = 5,
  Uncredit = 6,
  Foreign = 7,
  Entered = 8,
  Exited = 9,
```

```
} UzRwcDocStatus;

//////////////////
// Статуси ГУ-27 //
//////////////////
typedef enum GohubGu27Status
{
  gohub_gu27_status_unknown,
  gohub_gu27_status_draft,
  gohub_gu27_status_sending,
  gohub_gu27_status_registered,
  gohub_gu27_status_reclaiming,
  gohub_gu27_status_accepted,
  gohub_gu27_status_delivered,
  gohub_gu27_status_recieved,
  gohub_gu27_status_uncredited,
  gohub_gu27_status_recieved_draft,
  gohub_gu27_status_recieved_sending,
  gohub_gu27_status_recieved_reclaiming,
  gohub_gu27_status_canceled,
  gohub_gu27_status_locked,
} GohubGu27Status;

////////////////////////////////
// Робота з кодовими сторінками //
////////////////////////////////
int gohub_codepage();
int gohub_set_codepage(int codepage);
int gohub_encoding_codepage(const char* encodingName);
int gohub_encoding_codepage_w(const GohubWChar* encodingName);

////////////////////////////////////
// Підключення до Модулю Узгодження //
////////////////////////////////////
GohubConnection* gohub_connect(const char* host, int port);
GohubConnection* gohub_connect_w(const GohubWChar* host, int port);
GohubBool gohub_disconnect(GohubConnection* connection);

//////////////////////////
// Робота з документами //
//////////////////////////
GohubDocument* gohub_load_document(const char* path);
GohubDocument* gohub_load_document_w(const GohubWChar* path);
GohubDocument* gohub_create_document(const char* content);
GohubDocument* gohub_create_document_w(const GohubWChar* content);
GohubDocument* gohub_query_document(GohubConnection* connection, const char*
documentId);
GohubDocument* gohub_query_document_w(GohubConnection* connection, const
GohubWChar* documentId);
GohubDocument* gohub_query_next_document(GohubConnection* connection, int
lastRevision);
GohubDocument* gohub_query_next_document2(GohubConnection* connection, int
lastRevision);
const char* gohub_document_id(GohubDocument* document);
const GohubWChar* gohub_document_id_w(GohubDocument* document);
int gohub_document_revision(GohubDocument* document);
const char* gohub_document_text(GohubDocument* document);
const GohubWChar* gohub_document_text_w(GohubDocument* document);
const char* gohub_document_data_text(GohubDocument* document, int epdVersion);
const GohubWChar* gohub_document_data_text_w(GohubDocument* document, int
epdVersion);
int gohub_document_size(GohubDocument* document);
const char* gohub_document_measure_equip_num(GohubDocument* document);
const GohubWChar* gohub_document_measure_equip_num_w(GohubDocument* document);
const char* gohub_document_business_unit_num(GohubDocument* document);
```

```
const GohubWChar* gohub_document_business_unit_num_w(GohubDocument* document);
int gohub_document_get_verified_empty_weight_for_wagon(GohubDocument* document,
int wagonIndex);
bool gohub_document_get_foreign_not_accept(GohubDocument* document);
int gohub_document_warrant_type(GohubDocument* document);
GohubBool gohub_document_set_measure_equip_num(GohubDocument* document, const
char* val);
GohubBool gohub_document_set_measure_equip_num_w(GohubDocument* document, const
GohubWChar* val);
GohubBool gohub_document_set_business_unit_num(GohubDocument* document, const
char* val);
GohubBool gohub_document_set_business_unit_num_w(GohubDocument* document, const
GohubWChar* val);
GohubBool gohub_document_set_warrant_type(GohubDocument* document, int val);
GohubBool gohub_document_set_verified_empty_weight_for_wagon(GohubDocument*
document, int wagonIndex, int val);
GohubBool gohub_send_document(GohubConnection* connection, GohubDocument*
document);
GohubBool gohub_send_received_document(GohubConnection* connection,
GohubDocument* document, const char* documentId);
GohubBool gohub_send_received_document_w(GohubConnection* connection,
GohubDocument* document, const GohubWChar* documentId);
GohubBool gohub_save_document(GohubDocument* document, const char* path, int
codePage);
GohubBool gohub_save_document_w(GohubDocument* document, const GohubWChar* path,
int codePage);
GohubBool gohub_save_document_data(GohubDocument* document, const char* path, int
codePage, int epdVersion);
GohubBool gohub_save_document_data_w(GohubDocument* document, const GohubWChar*
path, int codePage, int epdVersion);
GohubBool gohub_close_document(GohubDocument* document);
GohubBool gohub_reclaim_document(GohubConnection* connection, const char*
documentId);
GohubBool gohub_reclaim_document_w(GohubConnection* connection, const GohubWChar*
documentId);
GohubBool gohub_delete_document(GohubConnection* connection, const char*
documentId);
GohubBool gohub_delete_document_w(GohubConnection* connection, const GohubWChar*
documentId);
GohubDocumentStatus gohub_document_status(GohubDocument* document);
GohubBool gohub_document_get_otpr(GohubDocument* document, const char* path);
GohubBool gohub_document_get_otpr_w(GohubDocument* document, const GohubWChar*
path);
const char* gohub_document_get_otpr_string(GohubDocument* document);
const GohubWChar* gohub_document_get_otpr_string_w(GohubDocument* document);
const char* gohub_document_warning(GohubDocument* document);
const GohubWChar* gohub_document_warning_w(GohubDocument* document);

//////////////////////////////////////////
// Робота з фільтрами запиту документів //
//////////////////////////////////////////
// Накладення фільтрів впливає на результат роботи функції
gohub_query_next_document
GohubBool gohub_clear_all_filters(GohubConnection* connection);
GohubBool gohub_set_filter_by_document_status(GohubConnection* connection, int
documentStatusCode);
GohubBool gohub_set_filter_by_document_number(GohubConnection* connection, const
char* documentNumber);
GohubBool gohub_set_filter_by_document_number_w(GohubConnection* connection,
const GohubWChar* documentNumber);
GohubBool gohub_set_filter_by_wagon_number(GohubConnection* connection, const
char* wagonNumber);
GohubBool gohub_set_filter_by_wagon_number_w(GohubConnection* connection, const
GohubWChar* wagonNumber);
```

```
GohubBool gohub_set_filter_by_departure_client(GohubConnection* connection, const
char* clientCode);
GohubBool gohub_set_filter_by_departure_client_w(GohubConnection* connection,
const GohubWChar* clientCode);
GohubBool gohub_set_filter_by_departure_payer(GohubConnection* connection, const
char* payerCode);
GohubBool gohub_set_filter_by_departure_payer_w(GohubConnection* connection,
const GohubWChar* payerCode);
GohubBool gohub_set_filter_by_departure_station(GohubConnection* connection,
const char* stationCode);
GohubBool gohub_set_filter_by_departure_station_w(GohubConnection* connection,
const GohubWChar* stationCode);
GohubBool gohub_set_filter_by_arrival_client(GohubConnection* connection, const
char* clientCode);
GohubBool gohub_set_filter_by_arrival_client_w(GohubConnection* connection, const
GohubWChar* clientCode);
GohubBool gohub_set_filter_by_arrival_payer(GohubConnection* connection, const
char* payerCode);
GohubBool gohub_set_filter_by_arrival_payer_w(GohubConnection* connection, const
GohubWChar* payerCode);
GohubBool gohub_set_filter_by_arrival_station(GohubConnection* connection, const
char* stationCode);
GohubBool gohub_set_filter_by_arrival_station_w(GohubConnection* connection,
const GohubWChar* stationCode);

GohubDocumentStatus gohub_get_filter_by_document_status(GohubConnection*
connection);
const char* gohub_get_filter_by_document_number(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_document_number_w(GohubConnection*
connection);
const char* gohub_get_filter_by_wagon_number(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_wagon_number_w(GohubConnection*
connection);
const char* gohub_get_filter_by_departure_client(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_departure_client_w(GohubConnection*
connection);
const char* gohub_get_filter_by_departure_payer(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_departure_payer_w(GohubConnection*
connection);
const char* gohub_get_filter_by_departure_station(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_departure_station_w(GohubConnection*
connection);
const char* gohub_get_filter_by_arrival_client(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_arrival_client_w(GohubConnection*
connection);
const char* gohub_get_filter_by_arrival_payer(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_arrival_payer_w(GohubConnection*
connection);
const char* gohub_get_filter_by_arrival_station(GohubConnection* connection);
const GohubWChar* gohub_get_filter_by_arrival_station_w(GohubConnection*
connection);

////////////////////////////////////
// Запит друкованих форм документів //
////////////////////////////////////
GohubBool gohub_query_and_save_document_printable_form(
  GohubConnection* connection,
  const char* documentId,
  const char* path);

GohubBool gohub_query_and_save_document_printable_form_w(
  GohubConnection* connection,
  const GohubWChar* documentId,
  const GohubWChar* path);
```

```
GohubBool gohub_query_and_save_document_archive(
  GohubConnection* connection,
  const char* documentId,
  const char* path);

GohubBool gohub_query_and_save_document_archive_w(
  GohubConnection* connection,
  const GohubWChar* documentId,
  const GohubWChar* path);

GohubBool gohub_query_and_save_fdu92_printable_form(
  GohubConnection* connection,
  const char* fdu92Id,
  const char* path);

GohubBool gohub_query_and_save_fdu92_printable_form_w(
  GohubConnection* connection,
  const GohubWChar* fdu92Id,
  const GohubWChar* path);

GohubBool gohub_query_and_save_gu46_printable_form(
  GohubConnection* connection,
  const char* gu46Id,
  const char* path);

GohubBool gohub_query_and_save_gu46_printable_form_w(
  GohubConnection* connection,
  const GohubWChar* gu46Id,
  const GohubWChar* path);

GohubBool gohub_query_and_save_gu45_printable_form(
  GohubConnection* connection,
  const char* gu45Id,
  const char* path);

GohubBool gohub_query_and_save_gu45_printable_form_w(
  GohubConnection* connection,
  const GohubWChar* gu45Id,
  const GohubWChar* path);

GohubBool gohub_query_and_save_gu27_printable_form(
  GohubConnection* connection,
  const char* gu27Id,
  const char* path);

GohubBool gohub_query_and_save_gu27_printable_form_w(
  GohubConnection* connection,
  const GohubWChar* gu27Id,
  const GohubWChar* path);

//////////////////////////////////////////
// Робота зі супровідними документами //
//////////////////////////////////////////
GohubAttachment* gohub_load_attachment(
  const char* typeCode,
  const char* name,
  const char* regNumber,
  const char* regDate,
  const char* validFrom,
  const char* validTo,
  const char* path);
GohubAttachment* gohub_load_attachment_w(
  const GohubWChar* typeCode,
  const GohubWChar* name,
  const GohubWChar* regNumber,
```

```
  const GohubWChar* regDate,
  const GohubWChar* validFrom,
  const GohubWChar* validTo,
  const GohubWChar* path);
GohubAttachment* gohub_load_smgs_attachment(
  const char* smgsTypeCode,
  const char* name,
  const char* regNumber,
  const char* regDate,
  const char* validFrom,
  const char* validTo,
  const char* path);
GohubAttachment* gohub_load_smgs_attachment_w(
  const GohubWChar* smgsTypeCode,
  const GohubWChar* name,
  const GohubWChar* regNumber,
  const GohubWChar* regDate,
  const GohubWChar* validFrom,
  const GohubWChar* validTo,
  const GohubWChar* path);
GohubAttachment* gohub_load_attachment_with_user_data(
  const char* typeCode,
  const char* name,
  const char* regNumber,
  const char* regDate,
  const char* validFrom,
  const char* validTo,
  const char* pathPdf,
  const char* pathUserData);
GohubAttachment* gohub_load_attachment_with_user_data_w(
  const GohubWChar* typeCode,
  const GohubWChar* name,
  const GohubWChar* regNumber,
  const GohubWChar* regDate,
  const GohubWChar* validFrom,
  const GohubWChar* validTo,
  const GohubWChar* pathPdf,
  const GohubWChar* pathUserData);
GohubAttachment* gohub_load_smgs_attachment_with_user_data(
  const char* smgsTypeCode,
  const char* name,
  const char* regNumber,
  const char* regDate,
  const char* validFrom,
  const char* validTo,
  const char* pathPdf,
  const char* pathUserData);
GohubAttachment* gohub_load_smgs_attachment_with_user_data_w(
  const GohubWChar* smgsTypeCode,
  const GohubWChar* name,
  const GohubWChar* regNumber,
  const GohubWChar* regDate,
  const GohubWChar* validFrom,
  const GohubWChar* validTo,
  const GohubWChar* pathPdf,
  const GohubWChar* pathUserData);
GohubBool gohub_send_attachment(GohubConnection* connection, GohubAttachment*
attachment);
GohubAttachment* gohub_query_attachment(GohubConnection* connection, const char*
attachmentId);
GohubAttachment* gohub_query_attachment_w(GohubConnection* connection, const
GohubWChar* attachmentId);
GohubAttachment* gohub_query_attachment_with_user_data(GohubConnection*
connection, const char* attachmentId);
```

```
GohubAttachment* gohub_query_attachment_with_user_data_w(GohubConnection*
connection, const GohubWChar* attachmentId);
GohubBool gohub_save_attachment(GohubAttachment* attachment, const char* path);
GohubBool gohub_save_attachment_w(GohubAttachment* attachment, const GohubWChar*
path);
GohubBool gohub_save_attachment_with_user_data(GohubAttachment* attachment, const
char* path);
GohubBool gohub_save_attachment_with_user_data_w(GohubAttachment* attachment,
const GohubWChar* path);
GohubBool gohub_delete_attachment(GohubConnection* connection, const char*
attachmentId);
GohubBool gohub_delete_attachment_w(GohubConnection* connection, const
GohubWChar* attachmentId);
GohubBool gohub_close_attachment(GohubAttachment* attachment);
const char* gohub_attachment_id(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_id_w(GohubAttachment* attachment);
const char* gohub_attachment_description(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_description_w(GohubAttachment* attachment);
const char* gohub_attachment_type_code(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_type_code_w(GohubAttachment* attachment);
const char* gohub_attachment_smgs_type_code(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_smgs_type_code_w(GohubAttachment* attachment);
const char* gohub_attachment_name(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_name_w(GohubAttachment* attachment);
const char* gohub_attachment_reg_number(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_reg_number_w(GohubAttachment* attachment);
const char* gohub_attachment_reg_date(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_reg_date_w(GohubAttachment* attachment);
const char* gohub_attachment_valid_from(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_valid_from_w(GohubAttachment* attachment);
const char* gohub_attachment_valid_to(GohubAttachment* attachment);
const GohubWChar* gohub_attachment_valid_to_w(GohubAttachment* attachment);
int gohub_attachment_count(GohubDocument* document);
const char* gohub_attachment_id_by_index(GohubDocument* document, int
attachmentIndex);
const GohubWChar* gohub_attachment_id_by_index_w(GohubDocument* document, int
attachmentIndex);

////////////////////////////////////////////////////////////////
// Робота з електронними даними попереднього інформування //
////////////////////////////////////////////////////////////////
GohubEData* gohub_load_edata(
  unsigned int attachmentSmgsTypeCode,
  const char* xmlPath,
  const char* name,
  const char* regNumber,
  const char* regDate,
  const char* validFrom,
  const char* validTo,
  const char* pdfPath);
GohubEData* gohub_load_edata_w(
  unsigned int attachmentSmgsTypeCode,
  const GohubWChar* xmlPath,
  const GohubWChar* name,
  const GohubWChar* regNumber,
  const GohubWChar* regDate,
  const GohubWChar* validFrom,
  const GohubWChar* validTo,
  const GohubWChar* pdfPath);
GohubEData* gohub_load_edata_simple(
  unsigned int attachmentSmgsTypeCode,
  const char* xmlPath);
GohubEData* gohub_load_edata_simple_w(
  unsigned int attachmentSmgsTypeCode,
  const GohubWChar* xmlPath);
```

```
GohubBool gohub_edata_load_data(GohubEData* edata, const char* xmlPath);
GohubBool gohub_edata_load_data_w(GohubEData* edata, const GohubWChar* xmlPath);
GohubBool gohub_update_edata(GohubConnection* connection, GohubEData* eData);
GohubBool gohub_send_edata(GohubConnection* connection, GohubEData* eData);
GohubEData* gohub_query_edata(GohubConnection* connection, const char* eDataId);
GohubEData* gohub_query_edata_w(GohubConnection* connection, const GohubWChar*
eDataId);
GohubEData* gohub_query_edata_for_attachment(GohubConnection* connection, const
char* attachmentId);
GohubEData* gohub_query_edata_for_attachment_w(GohubConnection* connection, const
GohubWChar* attachmentId);
GohubEData* gohub_query_next_edata(GohubConnection* connection, unsigned __int64
lastRevision);
GohubBool gohub_save_edata(GohubEData* eData, const char* path);
GohubBool gohub_save_edata_w(GohubEData* eData, const GohubWChar* path);
GohubBool gohub_close_edata(GohubEData* eData);
const char* gohub_edata_id(GohubEData* eData);
const GohubWChar* gohub_edata_id_w(GohubEData* eData);
unsigned __int64 gohub_edata_revision(GohubEData* eData);
const char* gohub_edata_revision_date(GohubEData* eData);
const GohubWChar* gohub_edata_revision_date_w(GohubEData* eData);
unsigned int gohub_edata_doc_type(GohubEData* eData);
int gohub_edata_status(GohubEData* eData);
const char* gohub_edata_version(GohubEData* eData);
const GohubWChar* gohub_edata_version_w(GohubEData* eData);
const char* gohub_edata_attachment_id(GohubEData* eData);
const GohubWChar* gohub_edata_attachment_id_w(GohubEData* eData);

//////////////////////////////////////////////////////
// Робота з пакетами попереднього інформування //
//////////////////////////////////////////////////////
GohubPiPackage* gohub_query_pi_package(GohubConnection* connection, const char*
piPackageId);
GohubPiPackage* gohub_query_pi_package_w(GohubConnection* connection, const
GohubWChar* piPackageId);
GohubPiPackage* gohub_query_next_pi_package(GohubConnection* connection, unsigned
__int64 lastRevision);
GohubBool gohub_save_pi_package(GohubPiPackage* piPackage, const char* path);
GohubBool gohub_save_pi_package_w(GohubPiPackage* piPackage, const GohubWChar*
path);
GohubBool gohub_close_pi_package(GohubPiPackage* piPackage);
const char* gohub_pi_package_id(GohubPiPackage* piPackage);
const GohubWChar* gohub_pi_package_id_w(GohubPiPackage* piPackage);
unsigned __int64 gohub_pi_package_revision(GohubPiPackage* piPackage);
const char* gohub_pi_package_revision_date(GohubPiPackage* piPackage);
const GohubWChar* gohub_pi_package_revision_date_w(GohubPiPackage* piPackage);
const char* gohub_pi_package_consignment_id(GohubPiPackage* piPackage);
const GohubWChar* gohub_pi_package_consignment_id_w(GohubPiPackage* piPackage);
int gohub_pi_package_status(GohubPiPackage* piPackage);
int gohub_pi_package_pipacktoed_count(GohubPiPackage* piPackage);
GohubPiPackageToEData* gohub_pi_package_pipacktoed(GohubPiPackage* piPackage, int
index);

const char* gohub_pipacktoed_id(GohubPiPackageToEData* piPackageToEData);
const GohubWChar* gohub_pipacktoed_id_w(GohubPiPackageToEData* piPackageToEData);
const char* gohub_pipacktoed_edata_id(GohubPiPackageToEData* piPackageToEData);
const GohubWChar* gohub_pipacktoed_edata_id_w(GohubPiPackageToEData*
piPackageToEData);
const char* gohub_pipacktoed_pi_package_id(GohubPiPackageToEData*
piPackageToEData);
const GohubWChar* gohub_pipacktoed_pi_package_id_w(GohubPiPackageToEData*
piPackageToEData);
const char* gohub_pipacktoed_note(GohubPiPackageToEData* piPackageToEData);
const GohubWChar* gohub_pipacktoed_note_w(GohubPiPackageToEData*
piPackageToEData);
```

```
int gohub_pipacktoed_status(GohubPiPackageToEData* piPackageToEData);
const char* gohub_pipacktoed_edata_version(GohubPiPackageToEData*
piPackageToEData);
const GohubWChar* gohub_pipacktoed_edata_version_w(GohubPiPackageToEData*
piPackageToEData);

GohubPiPackage* gohub_add_edata_to_pi_package(GohubConnection* connection,
GohubEData* edata, const char* piPackageId);
GohubPiPackage* gohub_add_edata_to_pi_package_w(GohubConnection* connection,
GohubEData* edata, const GohubWChar* piPackageId);

////////////////////////////////
// Статуси електронних ФДУ-92 //
////////////////////////////////
typedef enum GohubFdu92Status
{
      gohub_fdu92_status_unknown = -1,
      gohub_fdu92_status_approving = 0,
      gohub_fdu92_status_approving_modified = 1,
      gohub_fdu92_status_confirmed = 2,
      gohub_fdu92_status_canceled = 3,
      gohub_fdu92_status_agreed_noted_sending = 4,
      gohub_fdu92_status_agreed = 5,
      gohub_fdu92_status_agreed_noted = 6,
      gohub_fdu92_status_expired = 7,
      gohub_fdu92_status_agreed_sending = 8,
      gohub_fdu92_status_confirmed_paper = 9,
      gohub_fdu92_status_rejecting = 10,
      gohub_fdu92_status_rejected = 11,
      gohub_fdu92_status_paper = 60,
} GohubFdu92Status;

////////////////////////////////
// Робота з електронними ФДУ-92 //
////////////////////////////////
GohubFdu92* gohub_load_fdu92(const char* id, const char* path);
GohubFdu92* gohub_load_fdu92_w(const GohubWChar* id, const GohubWChar* path);
GohubFdu92* gohub_create_fdu92(const char* id, const char* content);
GohubFdu92* gohub_create_fdu92_w(const char* id, const GohubWChar* content);
GohubFdu92* gohub_query_fdu92(GohubConnection* connection, const char* fdu92Id);
GohubFdu92* gohub_query_fdu92_w(GohubConnection* connection, const GohubWChar*
fdu92Id);
GohubFdu92* gohub_query_next_fdu92(GohubConnection* connection, int
lastRevision);
GohubFdu92* gohub_query_fdu92_by_number(GohubConnection* connection, const char*
registration_esr, const char* registration_num);
GohubFdu92* gohub_query_fdu92_by_number_w(GohubConnection* connection, const
GohubWChar* registration_esr, const GohubWChar* registration_num);
GohubBool gohub_save_fdu92(GohubFdu92* fdu92, const char* path, int codePage);
GohubBool gohub_save_fdu92_w(GohubFdu92* fdu92, const GohubWChar* path, int
codePage);
GohubBool gohub_close_fdu92(GohubFdu92* fdu92);
GohubBool gohub_send_fdu92(GohubConnection* connection, GohubFdu92* document);
GohubBool gohub_reject_fdu92(GohubConnection* connection, const char* id);
GohubBool gohub_reject_fdu92_w(GohubConnection* connection, const GohubWChar*
id);

const char* gohub_fdu92_id(GohubFdu92* fdu92);
const GohubWChar* gohub_fdu92_id_w(GohubFdu92* fdu92);
int gohub_fdu92_revision(GohubFdu92* fdu92);
GohubFdu92Status gohub_fdu92_status(GohubFdu92* fdu92);
const char* gohub_fdu92_text(GohubFdu92* fdu92);
const GohubWChar* gohub_fdu92_text_w(GohubFdu92* fdu92);
int gohub_fdu92_size(GohubFdu92* fdu92);
const char* gohub_fdu92_signer_info(GohubFdu92* fdu92);
```

```
const GohubWChar* gohub_fdu92_signer_info_w(GohubFdu92* fdu92);
const char* gohub_fdu92_sign_time(GohubFdu92* fdu92);
const GohubWChar* gohub_fdu92_sign_time_w(GohubFdu92* fdu92);
GohubBool gohub_fdu92_has_signature(GohubFdu92* fdu92);
const GohubWChar* gohub_fdu92_signer_name_w(GohubFdu92* fdu92);

////////////////////////////////
// Статуси електронних ГУ-46 //
////////////////////////////////
typedef enum GohubGu46Status
{
      gohub_gu46_status_unknown = -1,
      gohub_gu46_status_approving = 0,
      gohub_gu46_status_approving_modified = 1,
      gohub_gu46_status_confirmed = 2,
      gohub_gu46_status_canceled = 3,
      gohub_gu46_status_agreed_noted_sending = 4,
      gohub_gu46_status_agreed = 5,
      gohub_gu46_status_agreed_noted = 6,
      gohub_gu46_status_expired = 7,
      gohub_gu46_status_agreed_sending = 8,
      gohub_gu46_status_confirmed_paper = 9,
      gohub_gu46_status_rejecting = 10,
      gohub_gu46_status_rejected = 11,
      gohub_gu46_status_paper = 60,
} GohubGu46Status;
////////////////////////////////
// Робота з електронними ГУ-46 //
////////////////////////////////
GohubGu46* gohub_load_gu46(const char* id, const char* path);
GohubGu46* gohub_load_gu46_w(const GohubWChar* id, const GohubWChar* path);
GohubGu46* gohub_create_gu46(const char* id, const char* content);
GohubGu46* gohub_create_gu46_w(const GohubWChar* id, const GohubWChar* content);
GohubGu46* gohub_query_gu46(GohubConnection* connection, const char* gu46Id);
GohubGu46* gohub_query_gu46_w(GohubConnection* connection, const GohubWChar*
gu46Id);
GohubGu46* gohub_query_next_gu46(GohubConnection* connection, int lastRevision);
GohubBool gohub_save_gu46(GohubGu46* gu46, const char* path, int codePage);
GohubBool gohub_save_gu46_w(GohubGu46* gu46, const GohubWChar* path, int
codePage);
GohubBool gohub_close_gu46(GohubGu46* gu46);
GohubBool gohub_send_gu46(GohubConnection* connection, GohubGu46* document);
GohubBool gohub_reject_gu46(GohubConnection* connection, const char* id);
GohubBool gohub_reject_gu46_w(GohubConnection* connection, const GohubWChar* id);
GohubGu46* gohub_query_gu46_by_number(GohubConnection* connection, const char*
registration_esr, const char* registration_num);
GohubGu46* gohub_query_gu46_by_number_w(GohubConnection* connection, const
GohubWChar* registration_esr, const GohubWChar* registration_num);

const char* gohub_gu46_id(GohubGu46* gu46);
const GohubWChar* gohub_gu46_id_w(GohubGu46* gu46);
int gohub_gu46_revision(GohubGu46* gu46);
GohubGu46Status gohub_gu46_status(GohubGu46* gu46);
const char* gohub_gu46_text(GohubGu46* gu46);
const GohubWChar* gohub_gu46_text_w(GohubGu46* gu46);
int gohub_gu46_size(GohubGu46* gu46);
const char* gohub_gu46_signer_info(GohubGu46* gu46);
const GohubWChar* gohub_gu46_signer_info_w(GohubGu46* gu46);
const char* gohub_gu46_sign_time(GohubGu46* gu46);
const GohubWChar* gohub_gu46_sign_time_w(GohubGu46* gu46);
GohubBool gohub_gu46_has_signature(GohubGu46* gu46);
const GohubWChar* gohub_gu46_signer_name_w(GohubGu46* gu46);

////////////////////////////////
// Статуси електронних ГУ-45 //
```

```
////////////////////////////////
typedef enum GohubGu45Status
{
      gohub_gu45_status_unknown = -1,
      gohub_gu45_status_confirmed = 2,
      gohub_gu45_status_canceled = 3,
      gohub_gu45_status_confirmed_paper = 9,
      gohub_gu45_status_paper = 60,
} GohubGu45Status;
////////////////////////////////
// Робота з електронними ГУ-45 //
////////////////////////////////
GohubGu45* gohub_query_gu45(GohubConnection* connection, const char* gu45Id);
GohubGu45* gohub_query_gu45_w(GohubConnection* connection, const GohubWChar*
gu45Id);
GohubGu45* gohub_query_next_gu45(GohubConnection* connection, int lastRevision);
GohubBool gohub_save_gu45(GohubGu45* gu45, const char* path, int codePage);
GohubBool gohub_save_gu45_w(GohubGu45* gu45, const GohubWChar* path, int
codePage);
GohubBool gohub_close_gu45(GohubGu45* gu45);
GohubGu45* gohub_query_gu45_by_number(GohubConnection* connection, const char*
registration_esr, const char* registration_num, const char* registration_date);
GohubGu45* gohub_query_gu45_by_number_w(GohubConnection* connection, const
GohubWChar* registration_esr, const GohubWChar* registration_num, const
GohubWChar* registration_date);


const char* gohub_gu45_id(GohubGu45* gu45);
const GohubWChar* gohub_gu45_id_w(GohubGu45* gu45);
int gohub_gu45_revision(GohubGu45* gu45);
GohubGu45Status gohub_gu45_status(GohubGu45* gu45);
const char* gohub_gu45_text(GohubGu45* gu45);
const GohubWChar* gohub_gu45_text_w(GohubGu45* gu45);
int gohub_gu45_size(GohubGu45* gu45);
const char* gohub_gu45_signer_info(GohubGu45* gu45);
const GohubWChar* gohub_gu45_signer_info_w(GohubGu45* gu45);
const char* gohub_gu45_sign_time(GohubGu45* gu45);
const GohubWChar* gohub_gu45_sign_time_w(GohubGu45* gu45);
GohubBool gohub_gu45_has_signature(GohubGu45* gu45);
const GohubWChar* gohub_gu45_signer_name_w(GohubGu45* gu45);

////////////////////////////////////////
// Перевірка електронно-цифрового підпису //
////////////////////////////////////////
GohubBool gohub_document_has_signature(GohubDocument* document);
GohubBool gohub_document_check_signature(GohubDocument* document);
const char* gohub_document_signer_name(GohubDocument* document);
const GohubWChar* gohub_document_signer_name_w(GohubDocument* document);
const char* gohub_document_signer_info(GohubDocument* document);
const GohubWChar* gohub_document_signer_info_w(GohubDocument* document);
const char* gohub_document_signer_info_by_index(GohubDocument* document, int
index);
const GohubWChar* gohub_document_signer_info_by_index_w(GohubDocument* document,
int index);
const char* gohub_document_signer_name_by_index(GohubDocument* document, int
index);
const GohubWChar* gohub_document_signer_name_by_index_w(GohubDocument* document,
int index);
const char* gohub_document_sign_time_by_index(GohubDocument* document, int
index);
const GohubWChar* gohub_document_sign_time_by_index_w(GohubDocument* document,
int index);
const char* gohub_document_sign_time(GohubDocument* document);
const GohubWChar* gohub_document_sign_time_w(GohubDocument* document);
const int gohub_document_signature_count(GohubDocument* document);
```

```
UzRwcDocStatus gohub_document_sign_status_by_index(GohubDocument* document, int
index);
const GohubWChar* gohub_document_sign_error_by_index_w(GohubDocument* document,
int index);
const char* gohub_document_sign_error_by_index(GohubDocument* document, int
index);
const GohubWChar* gohub_document_sign_error_by_index_w(GohubDocument* document,
int index);

//////////////////////////////////////////
// Накладення електронно-цифрового підпису //
//////////////////////////////////////////
GohubBool gohub_open_private_key(GohubConnection* connection, const char*
passwordToKey);
GohubBool gohub_open_private_key_w(GohubConnection* connection, const GohubWChar*
passwordToKey);
GohubBool gohub_open_private_key_from_path(GohubConnection* connection, const
char* passwordToKey, const char* keyFileName);
GohubBool gohub_open_private_key_from_path_w(GohubConnection* connection, const
GohubWChar* passwordToKey, const GohubWChar* keyFileName);
GohubBool gohub_open_private_key_by_bytes(GohubConnection* connection, const
char* passwordToKey, unsigned char* keyBinary, unsigned int length);
GohubBool gohub_open_private_key_by_bytes_w(GohubConnection* connection, const
GohubWChar* passwordToKey, unsigned char* keyBinary, unsigned int length);
const char* gohub_private_key_owner_name(GohubConnection* connection);
const GohubWChar* gohub_private_key_owner_name_w(GohubConnection* connection);
const char* gohub_private_key_owner_info(GohubConnection* connection);
const GohubWChar* gohub_private_key_owner_info_w(GohubConnection* connection);
GohubBool gohub_sign_document(GohubConnection* connection, GohubDocument*
document);
GohubBool gohub_sign_fdu92(GohubConnection* connection, GohubFdu92* fdu92);
GohubBool gohub_sign_gu46(GohubConnection* connection, GohubGu46* gu46);
GohubBool gohub_close_private_key(GohubConnection* connection);
GohubBool gohub_delete_old_certs_csk_uz(GohubConnection* connection, const char*
passwordToKey);
GohubBool gohub_delete_old_certs_csk_uz_w(GohubConnection* connection, const
GohubWChar* passwordToKey);

//////////////////////////////////////////
// Операції з файлами електронних ключів //
//////////////////////////////////////////
GohubBool gohub_mount_file_key(const char* keyId, const char* path);
GohubBool gohub_mount_file_key_w(const GohubWChar* keyId, const GohubWChar*
path);
GohubBool gohub_unmount_file_key(const char* keyId);
GohubBool gohub_unmount_file_key_w(const GohubWChar* keyId);
int gohub_query_mounted_file_keys();
const char* gohub_mounted_file_key_id(int index);
const GohubWChar* gohub_mounted_file_key_id_w(int index);
const char* gohub_mounted_file_key_dir(int index);
const GohubWChar* gohub_mounted_file_key_dir_w(int index);

//////////////////////
// Обробка помилок //
//////////////////////
GohubErrcode gohub_last_error_code();
const char* gohub_last_error_title();
const GohubWChar* gohub_last_error_title_w();
const char* gohub_last_error_text();
const GohubWChar* gohub_last_error_text_w();

//////////////////
// Робота з Гу27 //
//////////////////
GohubGu27* gohub_load_gu27(const char* path);
```

```
GohubGu27* gohub_load_gu27_w(const GohubWChar* path);
GohubGu27* gohub_create_gu27(const char* content);
GohubGu27* gohub_create_gu27_w(const GohubWChar* content);
GohubGu27* gohub_query_gu27(GohubConnection* connection, const char* gu27Id);
GohubGu27* gohub_query_gu27_w(GohubConnection* connection, const GohubWChar*
gu27Id);
GohubGu27* gohub_query_next_gu27(GohubConnection* connection, int lastRevision);
const char* gohub_gu27_id(GohubGu27* gu27);
const GohubWChar* gohub_gu27_id_w(GohubGu27* gu27);
int gohub_gu27_revision(GohubGu27* gu27);
const char* gohub_gu27_text(GohubGu27* gu27);
const GohubWChar* gohub_gu27_text_w(GohubGu27* gu27);
const char* gohub_gu27_data_text(GohubGu27* gu27, int gu27Version);
const GohubWChar* gohub_gu27_data_text_w(GohubGu27* gu27, int gu27Version);
int gohub_gu27_size(GohubGu27* gu27);
GohubBool gohub_send_gu27(GohubConnection* connection, GohubGu27* gu27);
GohubBool gohub_send_received_gu27(GohubConnection* connection, GohubGu27* gu27,
const char* gu27Id);
GohubBool gohub_send_received_gu27_w(GohubConnection* connection, GohubGu27*
gu27, const GohubWChar* gu27Id);
GohubBool gohub_save_gu27(GohubGu27* gu27, const char* path, int codePage);
GohubBool gohub_save_gu27_w(GohubGu27* gu27, const GohubWChar* path, int
codePage);
GohubBool gohub_save_gu27_data(GohubGu27* gu27, const char* path, int codePage,
int gu27Version);
GohubBool gohub_save_gu27_data_w(GohubGu27* gu27, const GohubWChar* path, int
codePage, int gu27Version);
GohubBool gohub_close_gu27(GohubGu27* gu27);
GohubBool gohub_reclaim_gu27(GohubConnection* connection, const char* gu27Id);
GohubBool gohub_reclaim_gu27_w(GohubConnection* connection, const GohubWChar*
gu27Id);
GohubBool gohub_delete_gu27(GohubConnection* connection, const char* gu27Id);
GohubBool gohub_delete_gu27_w(GohubConnection* connection, const GohubWChar*
gu27Id);
GohubGu27Status gohub_gu27_status(GohubGu27* gu27);

GohubBool gohub_sign_gu27(GohubConnection* connection, GohubGu27* gu27);

////////////////////////////////////////////
// Перевірка електронно-цифрового підпису GU27 //
////////////////////////////////////////////
GohubBool gohub_gu27_has_signature(GohubGu27* gu27);
GohubBool gohub_gu27_check_signature(GohubGu27* gu27);
const char* gohub_gu27_signer_name(GohubGu27* gu27);
const GohubWChar* gohub_gu27_signer_name_w(GohubGu27* gu27);
const char* gohub_gu27_signer_info(GohubGu27* gu27);
const GohubWChar* gohub_gu27_signer_info_w(GohubGu27* gu27);
const char* gohub_Gu27_sign_time(GohubGu27* gu27);
const GohubWChar* gohub_gu27_sign_time_w(GohubGu27* gu27);

////////////////////////////
// Робота з АС "Месплан" //
////////////////////////////
const char* gohub_get_mp_months(GohubConnection* connection, int codePage);
const GohubWChar* gohub_get_mp_months_w(GohubConnection* connection);
GohubBool gohub_query_and_save_orders_for_month(GohubConnection* connection,
const char* month, const char* path);
GohubBool gohub_query_and_save_orders_for_month_w(GohubConnection* connection,
const GohubWChar* month, const GohubWChar* path);
GohubBool gohub_query_and_save_orders_for_month_with_relogin(GohubConnection*
connection, const char* month, const char* login, const char* password, const
char* path);
GohubBool gohub_query_and_save_orders_for_month_with_relogin_w(GohubConnection*
connection, const GohubWChar* month, const GohubWChar* login, const GohubWChar*
password, const GohubWChar* path);
```

```
////////////////////////////////////////////
// Робота з документами інформаційних послуг //
////////////////////////////////////////////
GohubInformServicesDoc* gohub_query_inform_services_document(GohubConnection*
connection, unsigned __int64 docId);
GohubInformServicesDoc*
gohub_query_next_inform_services_document(GohubConnection* connection, unsigned
__int64 lastRevision);
GohubBool gohub_save_inform_services_document(GohubInformServicesDoc* document,
const char* path);
GohubBool gohub_save_inform_services_document_w(GohubInformServicesDoc* document,
const GohubWChar* path);
GohubBool gohub_saveXml_inform_services_document(GohubInformServicesDoc*
document, const char* path);
GohubBool gohub_saveXml_inform_services_document_w(GohubInformServicesDoc*
document, const GohubWChar* path);
GohubBool gohub_close_inform_services_document(GohubInformServicesDoc* document);
unsigned __int64 gohub_inform_services_document_id(GohubInformServicesDoc*
document);
unsigned __int64 gohub_inform_services_document_revision(GohubInformServicesDoc*
document);
const char* gohub_inform_services_document_filename(GohubInformServicesDoc*
document);
const GohubWChar*
gohub_inform_services_document_filename_w(GohubInformServicesDoc* document);
const char* gohub_inform_services_document_comment(GohubInformServicesDoc*
document);
const GohubWChar*
gohub_inform_services_document_comment_w(GohubInformServicesDoc* document);
const char* gohub_inform_services_document_created_date(GohubInformServicesDoc*
document);
const GohubWChar*
gohub_inform_services_document_created_date_w(GohubInformServicesDoc* document);
const char* gohub_inform_services_document_doc_date(GohubInformServicesDoc*
document);
const GohubWChar*
gohub_inform_services_document_doc_date_w(GohubInformServicesDoc* document);

////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
// Робота з переліком про замовлення на погодження перевезення за даними
календаря планування перевезень зернових вантажів(за останні 5 днів від поточної
дати) //
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
GohubDispatchInfo* gohub_query_dispatch_info(GohubConnection* connection, const
char* start_esr, const char* end_esr);
GohubDispatchInfo* gohub_query_dispatch_info_w(GohubConnection* connection, const
GohubWChar* start_esr, const GohubWChar* end_esr);
const char*  gohub_document_info_description(GohubDispatchInfo* document, int
index);
int gohub_document_info_count(GohubDispatchInfo* document);
GohubBool gohub_close_dispatch_info(GohubDispatchInfo* document);
const char*  gohub_document_info_number(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_type(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_date(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_wag_owner(GohubDispatchInfo* document, int
index);
const char*  gohub_document_info_is_empty(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_number_w(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_type_w(GohubDispatchInfo* document, int
index);
```

```
const GohubWChar*  gohub_document_info_date_w(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_wag_owner_w(GohubDispatchInfo* document,
int index);
const GohubWChar*  gohub_document_info_is_empty_w(GohubDispatchInfo* document,
int index);
const GohubWChar*  gohub_document_info_description_w(GohubDispatchInfo* document,
int index);GohubDispatchInfo* gohub_query_dispatch_info(GohubConnection*
connection, const char* start_esr, const char* end_esr);
GohubDispatchInfo* gohub_query_dispatch_info_w(GohubConnection* connection, const
GohubWChar* start_esr, const GohubWChar* end_esr);
const char*  gohub_document_info_description(GohubDispatchInfo* document, int
index);
int gohub_document_info_count(GohubDispatchInfo* document);
GohubBool gohub_close_dispatch_info(GohubDispatchInfo* document);
const char*  gohub_document_info_number(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_type(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_date(GohubDispatchInfo* document, int index);
const char*  gohub_document_info_wag_owner(GohubDispatchInfo* document, int
index);
const char*  gohub_document_info_is_empty(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_number_w(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_type_w(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_date_w(GohubDispatchInfo* document, int
index);
const GohubWChar*  gohub_document_info_wag_owner_w(GohubDispatchInfo* document,
int index);
const GohubWChar*  gohub_document_info_is_empty_w(GohubDispatchInfo* document,
int index);
const GohubWChar*  gohub_document_info_description_w(GohubDispatchInfo* document,
int index);

////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
// Робота з переліком про Номер замовлення ПСТД ////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
////////////////////////////////////////////////////////////////////////////////
GohubBool gohub_close_pstd_info(GohubPSTDInfo* document);
GohubPSTDInfo* gohub_query_pstd_info_w(GohubConnection* connection, const
GohubWChar* work_kind, const GohubWChar* departure_esr, const GohubWChar*
arrival_esr);
GohubPSTDInfo* gohub_query_pstd_info(GohubConnection* connection, const char*
work_kind, const char* departure_esr, const char* arrival_esr);
const char*  gohub_pstd_info_number(GohubPSTDInfo* document, int index);
const char*  gohub_pstd_info_arrivaldate(GohubPSTDInfo* document, int index);
const char*  gohub_pstd_info_departuredate(GohubPSTDInfo* document, int index);
const char*  gohub_pstd_info_reqdate(GohubPSTDInfo* document, int index);
const GohubWChar*  gohub_pstd_info_number_w(GohubPSTDInfo* document, int index);
const GohubWChar*  gohub_pstd_info_arrivaldate_w(GohubPSTDInfo* document, int
index);
const GohubWChar*  gohub_pstd_info_departuredate_w(GohubPSTDInfo* document, int
index);
const GohubWChar*  gohub_pstd_info_reqdate_w(GohubPSTDInfo* document, int index);
int gohub_pstd_info_count(GohubPSTDInfo* document);
#ifdef __cplusplus
} //extern "C"
#endif

#endif //GOHUB_CLIENT_H_
```

ДОДАТОК Б. Файл *gohub.client.errors.h*

```
#ifndef GOHUB_CLIENT_ERRORS_H_
#define GOHUB_CLIENT_ERRORS_H_

#ifdef __cplusplus
extern "C" {
#endif

//////////////////
// Коди помилок //
//////////////////

typedef enum GohubErrcode
{
  // Operation is success
  gohub_success,

  // Gohub server connection could not be established
  gohub_server_connection_could_not_be_established,

  // Gohub server inaccessible
  gohub_server_inaccessible,

  // Document creation failed
  gohub_document_creation_failed,

  // Document query failed
  gohub_document_query_failed,

  // Next document query failed
  gohub_next_document_query_failed,

  // Document sending failed
  gohub_document_sending_failed,

  // Document saving failed
  gohub_document_saving_failed,

  // Document loading failed
  gohub_document_loading_failed,

  // Invalid code page
  gohub_invalid_code_page,

  // Private key could not be opened
  gohub_private_key_could_not_be_opened,

  // Private key path could not be opened
  gohub_private_key_path_could_not_be_opened,

  // Private key bytes could not be opened
  gohub_private_key_bytes_could_not_be_opened,

  // Private key is inaccessible
  gohub_private_key_is_inaccessible,

  // Document could not be signed
  gohub_document_could_not_be_signed,

  // Document signature verification failed
  gohub_document_signature_verification_failed,

  // Document has not signature
  gohub_document_has_not_signature,
```

```
// Gohub Client is obsolete. Upgrade your Gohub Client version
gohub_client_is_obsolete,

// Gohub Serveris obsolete. Upgrade your Gohub Server version
gohub_server_is_obsolete,

//Document reclamation failed
gohub_document_reclamation_failed,

//Document deletion failed
gohub_document_deletion_failed,

// Attachment creation failed
gohub_attachment_creation_failed,

// Attachment sending failed
gohub_attachment_sending_failed,

// Attachment query failed
gohub_attachment_query_failed,

// Attachment deletion failed
gohub_attachment_deletion_failed,

// Mount of file key failed
gohub_mount_of_file_key_failed,

// Unmount of file key failed
gohub_unmount_of_file_key_failed,

// Enumerating of file keys failed
gohub_enumerating_of_file_keys_failed,

// Mounted file key inaccessible
gohub_mounted_file_key_inaccessible,

// EData creation failed
gohub_edata_creation_failed,

// EData sending failed
gohub_edata_sending_failed,

// EData updating failed
gohub_edata_updating_failed,

// EData query failed
gohub_edata_query_failed,

// Next EData query failed
gohub_next_edata_query_failed,

// PiPackage query failed
gohub_pipackage_query_failed,

// Next PiPackage query failed
gohub_next_pipackage_query_failed,

// MP months query failed
gohub_mp_months_query_failed,

// Orders of month query failed
gohub_orders_of_months_query_failed,

// Internal program error
```

```
  gohub_programm_error = 0x1000,

  // Client application performed invalid operation
  gohub_invalid_operation,

  // "Inform services doc saving failed"
  gohub_inform_services_doc_saving_failed,

  // "Inform services doc query failed"
  gohub_inform_services_doc_query_failed,

  // "Query changes inform services failed")]
  gohub_query_changes_inform_services_failed,

  // "Query next inform services document")]
  gohub_query_next_inform_services_document_failed,

} GohubErrcode;

#ifdef __cplusplus
} //extern "C"
#endif

#endif //GOHUB_CLIENT_ERRORS_H_
```

ДОДАТОК В. Файл опису COM-інтерфейсів GohubClientCOM.idl

```
import "oaidl.idl";
import "ocidl.idl";

interface IGohubClient;
interface IGohubDocument;
interface IGohubAttachment;
interface IGohubEData;
interface IGohubPiPackage;
interface IGohubPiPackageToEData;
interface IGohubConnection;
interface IGohubSignerInfo;
interface IGohubError;
interface IGohubFdu92;
interface IGohubGu46;
interface IGohubGu45;
interface IGohubGu27;
interface IGohubInformServicesDocument;
interface IGohubPSTDInfo;
[
	object,
	uuid(B0E1F579-5836-4E97-9340-58B092418947),
	dual,
	nonextensible,
	helpstring("IGohubSignerInfo Interface"),
	pointer_default(unique)
]
interface IGohubSignerInfo : IDispatch{
  [propget, id(1), helpstring("property Name")] HRESULT Name([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property Subject")] HRESULT Subject([out, retval]
BSTR* pVal);
};
[
	object,
	uuid(0D5D6225-8E07-46E2-8B8D-9C0966481994),
	dual,
	nonextensible,
	helpstring("IGohubDocument Interface"),
	pointer_default(unique)
]
interface IGohubDocument : IDispatch{
  [propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property Text")] HRESULT Text([out, retval] BSTR*
pVal);
  [propget, id(3), helpstring("property Revision")] HRESULT Revision([out,
retval] LONG* pVal);
  [propget, id(4), helpstring("property HasSignature")] HRESULT
HasSignature([out, retval] VARIANT_BOOL* pVal);
  [propget, id(5), helpstring("property Signer")] HRESULT Signer([out, retval]
IGohubSignerInfo** pVal);
  [propget, id(6), helpstring("property SignTime")] HRESULT SignTime([out,
retval] BSTR* pVal);
  [id(7), helpstring("method Save")] HRESULT Save([in] BSTR path, [in] LONG
codePage, [out,retval] VARIANT_BOOL* result);
  [id(8), helpstring("method Sign")] HRESULT Sign([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
  [id(9), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
  [id(10), helpstring("method Close")] HRESULT Close(void);
  [propget, id(11), helpstring("property AttachmentsCount")] HRESULT
AttachmentsCount([out, retval] LONG* pVal);
```

```
  [id(12), helpstring("method GetAttachmentIdByIndex")] HRESULT
GetAttachmentIdByIndex([in] LONG index, [out,retval] BSTR* result);
  [propget, id(15), helpstring("property Status")] HRESULT Status([out, retval]
LONG* pVal);
  [id(16), helpstring("method SendReceived")] HRESULT SendReceived([in]
IGohubConnection* connection, [in] BSTR docId, [out,retval] VARIANT_BOOL*
result);
  [id(17), helpstring("method SaveData")] HRESULT SaveData([in] BSTR path, [in]
LONG codePage, [in] LONG epdVersion, [out,retval] VARIANT_BOOL* result);
  [id(18), helpstring("method DataText")] HRESULT DataText([in] LONG epdVersion,
[out, retval] BSTR* pVal);
  [propget, id(19), helpstring("property MeasureEquipNum")] HRESULT
MeasureEquipNum([out, retval] BSTR* pVal);
  [propput, id(19), helpstring("property MeasureEquipNum")] HRESULT
MeasureEquipNum([in] BSTR pVal);
  [id(20), helpstring("method set_VerifiedEmptyWeightForWagon")] HRESULT
set_VerifiedEmptyWeightForWagon([in] LONG wagonIndex, [in] LONG weight);
  [id(21), helpstring("method get_VerifiedEmptyWeightForWagon")] HRESULT
get_VerifiedEmptyWeightForWagon([in] LONG wagonIndex, [out,retval] LONG* weight);
  [propget, id(22), helpstring("property BusinessUnitNum")] HRESULT
BusinessUnitNum([out, retval] BSTR* pVal);
  [propput, id(22), helpstring("property BusinessUnitNum")] HRESULT
BusinessUnitNum([in] BSTR pVal);
  [propget, id(23), helpstring("property ForeignNotAccept")] HRESULT
ForeignNotAccept([out, retval] VARIANT_BOOL* pVal);
  [propget, id(24), helpstring("property WarrantType")] HRESULT WarrantType([out,
retval] LONG* pVal);
  [propput, id(24), helpstring("property WarrantType")] HRESULT WarrantType([in]
LONG newVal);
  [id(25), helpstring("method SetVerifiedEmptyWeightForWagon")] HRESULT
SetVerifiedEmptyWeightForWagon([in] LONG wagonIndex, [in] LONG weight);
  [id(26), helpstring("method GetVerifiedEmptyWeightForWagon")] HRESULT
GetVerifiedEmptyWeightForWagon([in] LONG wagonIndex, [out,retval] LONG* weight);
  [id(27), helpstring("method GetOTPR")] HRESULT GetOTPR([in] BSTR path,
[out,retval] VARIANT_BOOL* result);
  [propget, id(28), helpstring("property OTPRString")] HRESULT OTPRString([out,
retval] BSTR* pVal);
  [propget, id(29), helpstring("property Warning")] HRESULT Warning([out, retval]
BSTR* pVal);
  [id(30), helpstring("property SignerByIndex")] HRESULT SignerByIndex([in] LONG
index, [out, retval] IGohubSignerInfo** pVal);
  [propget, id(31), helpstring("property SignerCount")] HRESULT SignerCount([out,
retval] LONG* length);
  [id(32), helpstring("property SignTimeByIndex")] HRESULT SignTimeByIndex([in]
LONG index, [out, retval] BSTR* pVal);
  [id(33), helpstring("property SignStatusByIndex")] HRESULT
SignStatusByIndex([in] LONG index, [out, retval] LONG* pVal);
  [id(34), helpstring("property SignErrorByIndex")] HRESULT SignErrorByIndex([in]
LONG index, [out, retval] BSTR* pVal);
};
[
      object,
      uuid(E9967B9D-1141-47BA-A5C4-573FB02DB396),
      dual,
      nonextensible,
      helpstring("IGohubConnection Interface"),
      pointer_default(unique)
]
interface IGohubConnection : IDispatch{
  [propget, id(1), helpstring("property Signer")] HRESULT Signer([out, retval]
IGohubSignerInfo** pVal);
  [id(2), helpstring("method Close")] HRESULT Close(void);
  [id(3), helpstring("method QueryDocument")] HRESULT QueryDocument([in] BSTR
docId, [out,retval] IGohubDocument** document);
```

```
  [id(4), helpstring("method QueryNextDocument")] HRESULT QueryNextDocument([in]
LONG revision, [out,retval] IGohubDocument** document);
  [id(5), helpstring("method OpenPrivateKey")] HRESULT OpenPrivateKey([in] BSTR
password, [out,retval] VARIANT_BOOL* result);
  [id(6), helpstring("method ReclaimDocument")] HRESULT ReclaimDocument([in] BSTR
docId, [out,retval] VARIANT_BOOL* result);
  [id(7), helpstring("method DeleteDocument")] HRESULT DeleteDocument([in] BSTR
docId, [out,retval] VARIANT_BOOL* result);
  [id(8), helpstring("method QueryAttachment")] HRESULT QueryAttachment([in] BSTR
attachmentId, [out,retval] IGohubAttachment** attachment);
  [id(9), helpstring("method DeleteAttachment")] HRESULT DeleteAttachment([in]
BSTR attachmentId, [out,retval] VARIANT_BOOL* result);
  [propget, id(10), helpstring("property FilterByDocumentStatus")] HRESULT
FilterByDocumentStatus([out, retval] LONG* pVal);
  [propput, id(10), helpstring("property FilterByDocumentStatus")] HRESULT
FilterByDocumentStatus([in] LONG newVal);
  [propget, id(11), helpstring("property FilterByDocumentNumber")] HRESULT
FilterByDocumentNumber([out, retval] BSTR* pVal);
  [propput, id(11), helpstring("property FilterByDocumentNumber")] HRESULT
FilterByDocumentNumber([in] BSTR newVal);
  [propget, id(12), helpstring("property FilterByWagonNumber")] HRESULT
FilterByWagonNumber([out, retval] BSTR* pVal);
  [propput, id(12), helpstring("property FilterByWagonNumber")] HRESULT
FilterByWagonNumber([in] BSTR newVal);
  [propget, id(13), helpstring("property FilterByDepartureClientCode")] HRESULT
FilterByDepartureClientCode([out, retval] BSTR* pVal);
  [propput, id(13), helpstring("property FilterByDepartureClientCode")] HRESULT
FilterByDepartureClientCode([in] BSTR newVal);
  [propget, id(14), helpstring("property FilterByDeparturePayerCode")] HRESULT
FilterByDeparturePayerCode([out, retval] BSTR* pVal);
  [propput, id(14), helpstring("property FilterByDeparturePayerCode")] HRESULT
FilterByDeparturePayerCode([in] BSTR newVal);
  [propget, id(15), helpstring("property FilterByDepartureStationCode")] HRESULT
FilterByDepartureStationCode([out, retval] BSTR* pVal);
  [propput, id(15), helpstring("property FilterByDepartureStationCode")] HRESULT
FilterByDepartureStationCode([in] BSTR newVal);
  [propget, id(16), helpstring("property FilterByArrivalClientCode")] HRESULT
FilterByArrivalClientCode([out, retval] BSTR* pVal);
  [propput, id(16), helpstring("property FilterByArrivalClientCode")] HRESULT
FilterByArrivalClientCode([in] BSTR newVal);
  [propget, id(17), helpstring("property FilterByArrivalPayerCode")] HRESULT
FilterByArrivalPayerCode([out, retval] BSTR* pVal);
  [propput, id(17), helpstring("property FilterByArrivalPayerCode")] HRESULT
FilterByArrivalPayerCode([in] BSTR newVal);
  [propget, id(18), helpstring("property FilterByArrivalStationCode")] HRESULT
FilterByArrivalStationCode([out, retval] BSTR* pVal);
  [propput, id(18), helpstring("property FilterByArrivalStationCode")] HRESULT
FilterByArrivalStationCode([in] BSTR newVal);
  [id(19), helpstring("method ClearAllFilters")] HRESULT
ClearAllFilters([out,retval] VARIANT_BOOL* result);
  [id(20), helpstring("method QueryAndSaveDocumentPrintableForm")] HRESULT
QueryAndSaveDocumentPrintableForm([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(21), helpstring("method QueryAndSaveFdu92PrintableForm")] HRESULT
QueryAndSaveFdu92PrintableForm([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(22), helpstring("method QueryAndSaveGu45PrintableForm")] HRESULT
QueryAndSaveGu45PrintableForm([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(23), helpstring("method QueryAndSaveGu46PrintableForm")] HRESULT
QueryAndSaveGu46PrintableForm([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(24), helpstring("method QueryFdu92")] HRESULT QueryFdu92([in] BSTR id,
[out,retval] IGohubFdu92** fdu92);
```

```
  [id(25), helpstring("method QueryNextFdu92")] HRESULT QueryNextFdu92([in] LONG
revision, [out,retval] IGohubFdu92** fdu92);
  [id(26), helpstring("method QueryGu46")] HRESULT QueryGu46([in] BSTR id,
[out,retval] IGohubGu46** gu46);
  [id(27), helpstring("method QueryNextGu46")] HRESULT QueryNextGu46([in] LONG
revision, [out,retval] IGohubGu46** gu46);
  [id(28), helpstring("method QueryGu45")] HRESULT QueryGu45([in] BSTR id,
[out,retval] IGohubGu45** gu45);
  [id(29), helpstring("method QueryNextGu45")] HRESULT QueryNextGu45([in] LONG
revision, [out,retval] IGohubGu45** gu45);
  [id(30), helpstring("method RejectFdu92")] HRESULT RejectFdu92(BSTR id);
  [id(31), helpstring("method RejectGu46")] HRESULT RejectGu46(BSTR id);
  [id(32), helpstring("method QueryEData")] HRESULT QueryEData([in] BSTR eDataId,
[out,retval] IGohubEData** eData);
  [id(33), helpstring("method QueryNextEData")] HRESULT QueryNextEData([in]
ULONGLONG revision, [out,retval] IGohubEData** eData);
  [id(34), helpstring("method QueryPiPackage")] HRESULT QueryPiPackage([in] BSTR
eDataId, [out,retval] IGohubPiPackage** eData);
  [id(35), helpstring("method QueryNextPiPackage")] HRESULT
QueryNextPiPackage([in] ULONGLONG revision, [out,retval] IGohubPiPackage**
eData);
  [id(36), helpstring("method AddEDataToPiPackage")] HRESULT
AddEDataToPiPackage([in] IGohubEData* eData, [in] BSTR piPackageId, [out,retval]
IGohubPiPackage** piPackage);
  [id(37), helpstring("method QueryGu27")] HRESULT QueryGu27([in] BSTR gu27Id,
[out,retval] IGohubGu27** gu27);
  [id(38), helpstring("method QueryNextGu27")] HRESULT QueryNextGu27([in] LONG
revision, [out,retval] IGohubGu27** gu27);
  [id(39), helpstring("method ReclaimGu27")] HRESULT ReclaimGu27([in] BSTR
gu27Id, [out,retval] VARIANT_BOOL* result);
  [id(40), helpstring("method DeleteGu27")] HRESULT DeleteGu27([in] BSTR gu27Id,
[out,retval] VARIANT_BOOL* result);
  [id(41), helpstring("method QueryAndSaveGu27PrintableForm")] HRESULT
QueryAndSaveGu27PrintableForm([in] BSTR gu27Id, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(42), helpstring("method QueryAttachmentWithUserData")] HRESULT
QueryAttachmentWithUserData([in] BSTR attachmentId, [out,retval]
IGohubAttachment** attachment);
  [id(43), helpstring("method GetMPMonths")] HRESULT GetMPMonths([out,retval]
BSTR* result);
  [id(44), helpstring("method QueryAndSaveOrdersForMonth")] HRESULT
QueryAndSaveOrdersForMonth([in] BSTR month, [in] BSTR path);
  [id(45), helpstring("method QueryAndSaveOrdersForMonthWithRelogin")] HRESULT
QueryAndSaveOrdersForMonthWithRelogin([in] BSTR month, [in] BSTR login, [in] BSTR
password, [in] BSTR path);
  [id(46), helpstring("method SaveDocumentPrintableForm")] HRESULT
SaveDocumentPrintableForm([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(47), helpstring("method SaveOrdersForMonthWithRelogin")] HRESULT
SaveOrdersForMonthWithRelogin([in] BSTR month, [in] BSTR login, [in] BSTR
password, [in] BSTR path);
  [id(48), helpstring("method QueryEDataForAttachment")] HRESULT
QueryEDataForAttachment([in] BSTR attachmentId, [out,retval] IGohubEData**
eData);
  [id(49), helpstring("method QueryInfServsDoc")] HRESULT QueryInfServsDoc([in]
ULONGLONG docId, [out,retval] IGohubInformServicesDocument** document);
  [id(50), helpstring("method QueryNextInfServsDoc")] HRESULT
QueryNextInfServsDoc([in] ULONGLONG revision, [out,retval]
IGohubInformServicesDocument** document);
  [id(51), helpstring("method OpenPrivateKeyFromPath")] HRESULT
OpenPrivateKeyFromPath([in] BSTR password, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
  [id(52), helpstring("method QueryFdu92ByNumber")] HRESULT
QueryFdu92ByNumber([in] BSTR registration_esr, [in]BSTR registration_num,
[out,retval] IGohubFdu92** fdu92);
```

```
  [id(53), helpstring("method QueryDispatchInfo")] HRESULT QueryDispatchInfo([in]
BSTR dispatch_esr, [in]BSTR arrival_num, [out,retval] IGohubDispatchInfo** disp);
  [id(54), helpstring("method QueryGu45ByNumber")] HRESULT QueryGu45ByNumber([in]
BSTR registration_esr, [in]BSTR registration_num, [in]BSTR registration_date,
[out,retval] IGohubGu45** gu45);
  [id(55), helpstring("method QueryGu46ByNumber")] HRESULT QueryGu46ByNumber([in]
BSTR registration_esr, [in]BSTR registration_num, [out,retval] IGohubGu46**
gu46);
  [id(56), helpstring("method QueryNextDocument2")] HRESULT
QueryNextDocument2([in] LONG revision, [out,retval] IGohubDocument** document);
  [id(57), helpstring("method QueryPSTDInfo")] HRESULT QueryPSTDInfo([in]BSTR
work_kind, [in]BSTR dispatch_esr, [in]BSTR arrival_esr,
[out,retval]IGohubPSTDInfo** document);
  [id(58), helpstring("method get_KeysList")] HRESULT get_KeysList([out, retval]
BSTR* pVal);
  [id(59), helpstring("method OpenPrivateKeyByIndex")] HRESULT
OpenPrivateKeyByIndex([in] BSTR password, [in] LONG MediaIndex, [in] LONG
DeviceIndex, [out,retval] VARIANT_BOOL* result);
[id(60), helpstring("method QueryAndSaveDocumentArchive")] HRESULT
QueryAndSaveDocumentArchive([in] BSTR docId, [in] BSTR path, [out,retval]
VARIANT_BOOL* result);
};
[
      object,
      uuid(F1077417-21D9-4871-84A2-9F89525E7214),
      dual,
      nonextensible,
      helpstring("IGohubError Interface"),
      pointer_default(unique)
]
interface IGohubError : IDispatch{
  [propget, id(1), helpstring("property Code")] HRESULT Code([out, retval] LONG*
pVal);
  [propget, id(2), helpstring("property Title")] HRESULT Title([out, retval]
BSTR* pVal);
  [propget, id(3), helpstring("property Text")] HRESULT Text([out, retval] BSTR*
pVal);
};
[
      object,
      uuid(ABDA6C07-5320-4F28-B995-FADE037D0A82),
      dual,
      nonextensible,
      helpstring("IGohubClient Interface"),
      pointer_default(unique)
]
interface IGohubClient : IDispatch{
  [id(1), helpstring("method GetLastError")] HRESULT GetLastError([out,retval]
IGohubError** result);
  [id(2), helpstring("method Connect")] HRESULT Connect([in] BSTR host, [in] LONG
port, [out,retval] IGohubConnection** connection);
  [id(3), helpstring("method CreateDocument")] HRESULT CreateDocument([in] BSTR
xmlText, [out,retval] IGohubDocument** document);
  [id(4), helpstring("method LoadDocument")] HRESULT LoadDocument([in] BSTR path,
[out,retval] IGohubDocument** result);
  [id(5), helpstring("method LoadAttachment")] HRESULT LoadAttachment([in] BSTR
typeCode, [in] BSTR name, [in] BSTR regNumber, [in] BSTR regDate, [in] BSTR
validFrom, [in] BSTR validTo, [in] BSTR path, [out,retval] IGohubAttachment**
result);
  [id(6), helpstring("method MountFileKey")] HRESULT MountFileKey([in] BSTR
keyId, [in] BSTR keyDir, [out,retval] VARIANT_BOOL* result);
  [id(7), helpstring("method UnmountFileKey")] HRESULT UnmountFileKey([in] BSTR
keyId, [out,retval] VARIANT_BOOL* result);
  [id(8), helpstring("method QueryMountedKeys")] HRESULT
QueryMountedKeys([out,retval] LONG* result);
```

```
  [id(9), helpstring("method GetMountedKeyId")] HRESULT GetMountedKeyId([in] LONG
index, [out,retval] BSTR* keyId);
  [id(10), helpstring("method GetMountedKeyDir")] HRESULT GetMountedKeyDir([in]
LONG index, [out,retval] BSTR* keyDir);
  [id(11), helpstring("method LoadEData")] HRESULT LoadEData([in] UINT codeType,
[in] BSTR xmlPath, [in] BSTR name, [in] BSTR regNumber, [in] BSTR regDate, [in]
BSTR validFrom, [in] BSTR validTo, [in] BSTR pdfPath, [out,retval] IGohubEData**
result);
  [id(12), helpstring("method LoadEDataSimple")] HRESULT LoadEDataSimple([in]
UINT codeType, [in] BSTR xmlPath, [out,retval] IGohubEData** result);
  [id(13), helpstring("method CreateGu27")] HRESULT CreateGu27([in] BSTR xmlText,
[out,retval] IGohubGu27** gu27);
  [id(14), helpstring("method LoadGu27")] HRESULT LoadGu27([in] BSTR path,
[out,retval] IGohubGu27** result);
  [id(15), helpstring("method LoadSmgsAttachment")] HRESULT
LoadSmgsAttachment([in] BSTR smgsTypeCode, [in] BSTR name, [in] BSTR regNumber,
[in] BSTR regDate, [in] BSTR validFrom, [in] BSTR validTo, [in] BSTR path,
[out,retval] IGohubAttachment** result);
  [id(16), helpstring("method LoadAttachmentWithUserData")] HRESULT
LoadAttachmentWithUserData([in] BSTR typeCode, [in] BSTR name, [in] BSTR
regNumber, [in] BSTR regDate, [in] BSTR validFrom, [in] BSTR validTo, [in] BSTR
path, [in] BSTR pathUserData, [out,retval] IGohubAttachment** attachment);
  [id(17), helpstring("method LoadSmgsAttachmentWithUserData")] HRESULT
LoadSmgsAttachmentWithUserData([in] BSTR smgsTypeCode, [in] BSTR name, [in] BSTR
regNumber, [in] BSTR regDate, [in] BSTR validFrom, [in] BSTR validTo, [in] BSTR
path, [in] BSTR pathUserData, [out,retval] IGohubAttachment** attachment);
  [id(18), helpstring("method LoadFdu92")] HRESULT LoadFdu92([in] BSTR ident,
[in] BSTR path, [out,retval] IGohubFdu92** result);
  [id(19), helpstring("method LoadGu46")] HRESULT LoadGu46([in] BSTR ident, [in]
BSTR path, [out,retval] IGohubGu46** result);
};
[
      object,
      uuid(044603D2-574E-463C-87EC-DCD98C30F319),
      dual,
      nonextensible,
      helpstring("IGohubAttachment Interface"),
      pointer_default(unique)
]
interface IGohubAttachment : IDispatch{
  [propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property TypeCode")] HRESULT TypeCode([out,
retval] BSTR* pVal);
  [propget, id(3), helpstring("property Name")] HRESULT Name([out, retval] BSTR*
pVal);
  [propget, id(4), helpstring("property Description")] HRESULT Description([out,
retval] BSTR* pVal);
  [propget, id(5), helpstring("property RegNumber")] HRESULT RegNumber([out,
retval] BSTR* pVal);
  [propget, id(6), helpstring("property RegDate")] HRESULT RegDate([out, retval]
BSTR* pVal);
  [propget, id(7), helpstring("property ValidFrom")] HRESULT ValidFrom([out,
retval] BSTR* pVal);
  [propget, id(8), helpstring("property ValidTo")] HRESULT ValidTo([out, retval]
BSTR* pVal);
  [id(9), helpstring("method Save")] HRESULT Save([in] BSTR path, [out, retval]
VARIANT_BOOL* result);
  [id(10), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
  [id(11), helpstring("method Close")] HRESULT Close(void);
  [propget, id(12), helpstring("property SmgsTypeCode")] HRESULT
SmgsTypeCode([out, retval] BSTR* pVal);
  [id(13), helpstring("method SaveUserData")] HRESULT SaveUserData([in] BSTR
path, [out,retval] VARIANT_BOOL* result);
```

```
};
[
	object,
	uuid(685B21FA-43A7-4ACC-9A57-AB7AC332942C),
	dual,
	nonextensible,
	helpstring("IGohubEData Interface"),
	pointer_default(unique)
]
interface IGohubEData : IDispatch{
  [propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property Revision")] HRESULT Revision([out,
retval] ULONGLONG* pVal);
  [propget, id(3), helpstring("property RevisionDate")] HRESULT
RevisionDate([out, retval] BSTR* pVal);
  [propget, id(4), helpstring("property Version")] HRESULT Version([out, retval]
BSTR* pVal);
  [propget, id(5), helpstring("property DocType")] HRESULT DocType([out, retval]
UINT* pVal);
  [propget, id(6), helpstring("property Status")] HRESULT Status([out, retval]
INT* pVal);
  [propget, id(7), helpstring("property AttachmentId")] HRESULT
AttachmentId([out, retval] BSTR* pVal);
  [id(8), helpstring("method Save")] HRESULT Save([in] BSTR path, [out, retval]
VARIANT_BOOL* result);
  [id(9), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
  [id(10), helpstring("method LoadData")] HRESULT LoadData([in] BSTR path, [out,
retval] VARIANT_BOOL* result);
  [id(11), helpstring("method Update")] HRESULT Update([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
  [id(12), helpstring("method Close")] HRESULT Close(void);
};
[
	object,
	uuid(62C21013-07D6-4d9d-83C4-9FA6E770B2EA),
	dual,
	nonextensible,
	helpstring("IGohubPiPackage Interface"),
	pointer_default(unique)
]
interface IGohubPiPackage : IDispatch{
  [propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property Revision")] HRESULT Revision([out,
retval] ULONGLONG* pVal);
  [propget, id(3), helpstring("property RevisionDate")] HRESULT
RevisionDate([out, retval] BSTR* pVal);
  [propget, id(4), helpstring("property ConsignmentId")] HRESULT
ConsignmentId([out, retval] BSTR* pVal);
  [propget, id(5), helpstring("property Status")] HRESULT Status([out, retval]
INT* pVal);
  [propget, id(6), helpstring("property PiPackageToEDataCount")] HRESULT
PiPackageToEDataCount([out, retval] INT* pVal);
  [id(7), helpstring("method PiPackageToEData")] HRESULT PiPackageToEData([in]
INT index, [out, retval] IGohubPiPackageToEData** result);
  [id(8), helpstring("method Save")] HRESULT Save([in] BSTR path, [out, retval]
VARIANT_BOOL* result);
  [id(10), helpstring("method Close")] HRESULT Close(void);
};
[
	object,
	uuid(88AAFC89-0426-4551-BD1C-F57F63D0C335),
	dual,
```

```
	nonextensible,
	helpstring("IGohubPiPackageToEData Interface"),
	pointer_default(unique)
]
interface IGohubPiPackageToEData : IDispatch{
  [propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
  [propget, id(2), helpstring("property EDataId")] HRESULT EDataId([out, retval]
BSTR* pVal);
  [propget, id(3), helpstring("property PiPackageId")] HRESULT PiPackageId([out,
retval] BSTR* pVal);
  [propget, id(4), helpstring("property Note")] HRESULT Note([out, retval] BSTR*
pVal);
  [propget, id(5), helpstring("property Status")] HRESULT Status([out, retval]
INT* pVal);
  [propget, id(6), helpstring("property EDataVersion")] HRESULT
EDataVersion([out, retval] BSTR* pVal);
};
[
	object,
	uuid(EB03BE7D-48B7-4AFD-8A94-A5012D844A17),
	dual,
	nonextensible,
	helpstring("IGohubFdu92 Interface"),
	pointer_default(unique)
]
interface IGohubFdu92 : IDispatch{
	[propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
	[propget, id(2), helpstring("property Status")] HRESULT Status([out,
retval] LONG* pVal);
	[propget, id(3), helpstring("property Revision")] HRESULT Revision([out,
retval] LONG* pVal);
	[propget, id(4), helpstring("property Text")] HRESULT Text([out, retval]
BSTR* pVal);
	[id(5), helpstring("method Save")] HRESULT Save([in] BSTR path, [in] LONG
codePage, [out,retval] VARIANT_BOOL* result);
	[id(6), helpstring("method Close")] HRESULT Close(void);
	[id(7), helpstring("method Sign")] HRESULT Sign([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[id(8), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[propget, id(9), helpstring("property HasSignature")] HRESULT
HasSignature([out, retval] VARIANT_BOOL* result);
	[propget, id(10), helpstring("property Signer")] HRESULT Signer([out,
retval] IGohubSignerInfo** pVal);
	[propget, id(11), helpstring("property SignTime")] HRESULT SignTime([out,
retval] BSTR* pVal);
};
[
	object,
	uuid(78DCD121-84B7-4D41-B15C-F9F7677A3519),
	dual,
	nonextensible,
	helpstring("IGohubGu46 Interface"),
	pointer_default(unique)
]
interface IGohubGu46 : IDispatch{
	[propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
	[propget, id(2), helpstring("property Status")] HRESULT Status([out,
retval] LONG* pVal);
	[propget, id(3), helpstring("property Revision")] HRESULT Revision([out,
retval] LONG* pVal);
```

```
	[propget, id(4), helpstring("property Text")] HRESULT Text([out, retval]
BSTR* pVal);
	[id(5), helpstring("method Save")] HRESULT Save([in] BSTR path, [in] LONG
codePage, [out,retval] VARIANT_BOOL* result);
	[id(6), helpstring("method Close")] HRESULT Close(void);
	[id(7), helpstring("method Sign")] HRESULT Sign([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[id(8), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[propget, id(9), helpstring("property HasSignature")] HRESULT
HasSignature([out, retval] VARIANT_BOOL* result);
	[propget, id(10), helpstring("property Signer")] HRESULT Signer([out,
retval] IGohubSignerInfo** pVal);
	[propget, id(11), helpstring("property SignTime")] HRESULT SignTime([out,
retval] BSTR* pVal);
};
[
	object,
	uuid(C2F92D3C-295A-4453-B4CF-B33D2C8966E3),
	dual,
	nonextensible,
	helpstring("IGohubGu45 Interface"),
	pointer_default(unique)
]
interface IGohubGu45 : IDispatch{
	[propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
	[propget, id(2), helpstring("property Status")] HRESULT Status([out,
retval] LONG* pVal);
	[propget, id(3), helpstring("property Revision")] HRESULT Revision([out,
retval] LONG* pVal);
	[propget, id(4), helpstring("property Text")] HRESULT Text([out, retval]
BSTR* pVal);
	[id(5), helpstring("method Save")] HRESULT Save([in] BSTR path, [in] LONG
codePage, [out,retval] VARIANT_BOOL* result);
	[id(6), helpstring("method Close")] HRESULT Close(void);
	[propget, id(7), helpstring("property HasSignature")] HRESULT
HasSignature([out, retval] VARIANT_BOOL* result);
	[propget, id(8), helpstring("property Signer")] HRESULT Signer([out,
retval] IGohubSignerInfo** pVal);
	[propget, id(9), helpstring("property SignTime")] HRESULT SignTime([out,
retval] BSTR* pVal);
};
[
	object,
	uuid(4B840AC2-7128-4989-A56C-2570E8435C48),
	dual,
	nonextensible,
	helpstring("IGohubGu27 Interface"),
	pointer_default(unique)
]
interface IGohubGu27 : IDispatch{
	[propget, id(1), helpstring("property Id")] HRESULT Id([out, retval] BSTR*
pVal);
	[propget, id(2), helpstring("property Text")] HRESULT Text([out, retval]
BSTR* pVal);
	[propget, id(3), helpstring("property Revision")] HRESULT Revision([out,
retval] LONG* pVal);
	[propget, id(4), helpstring("property HasSignature")] HRESULT
HasSignature([out, retval] VARIANT_BOOL* pVal);
	[propget, id(5), helpstring("property Signer")] HRESULT Signer([out,
retval] IGohubSignerInfo** pVal);
	[propget, id(6), helpstring("property SignTime")] HRESULT SignTime([out,
retval] BSTR* pVal);
```

```
	[id(7), helpstring("method Save")] HRESULT Save([in] BSTR path, [in] LONG
codePage, [out,retval] VARIANT_BOOL* result);
	[id(8), helpstring("method Sign")] HRESULT Sign([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[id(9), helpstring("method Send")] HRESULT Send([in] IGohubConnection*
connection, [out,retval] VARIANT_BOOL* result);
	[id(10), helpstring("method Close")] HRESULT Close(void);
	[propget, id(11), helpstring("property Status")] HRESULT Status([out,
retval] LONG* pVal);
	[id(12), helpstring("method SendReceived")] HRESULT SendReceived([in]
IGohubConnection* connection, [in] BSTR gu27Id, [out,retval] VARIANT_BOOL*
result);
	[id(13), helpstring("method SaveData")] HRESULT SaveData([in] BSTR path,
[in] LONG codePage, [in] LONG gu27Version, [out,retval] VARIANT_BOOL* result);
	[id(14), helpstring("method DataText")] HRESULT DataText([in] LONG
gu27Version, [out,retval] BSTR* pVal);
};
[
	object,
	uuid(0af49642-b12d-4fdb-b363-a7497cc78462),
	dual,
	nonextensible,
	helpstring("IInformServicesDocument Interface"),
	pointer_default(unique)
]
interface IGohubInformServicesDocument : IDispatch{
	[id(1), helpstring("method Save")] HRESULT Save([in] BSTR path,
[out,retval] VARIANT_BOOL* result);
	[id(2), helpstring("method Close")] HRESULT Close(void);
	[propget, id(3), helpstring("property Id")] HRESULT Id([out, retval]
ULONGLONG* pVal);
	[propget, id(4), helpstring("property Revision")] HRESULT Revision([out,
retval] ULONGLONG* pVal);
	[propget, id(5), helpstring("property FileName")] HRESULT FileName([out,
retval] BSTR* pVal);
	[propget, id(6), helpstring("property Comment")] HRESULT Comment([out,
retval] BSTR* pVal);
	[propget, id(7), helpstring("property CreatedDate")] HRESULT
CreatedDate([out, retval] BSTR* pVal);
	[propget, id(8), helpstring("property DocDate")] HRESULT DocDate([out,
retval] BSTR* pVal);
	[id(9), helpstring("method SaveXml")] HRESULT SaveXml([in] BSTR path,
[out,retval] VARIANT_BOOL* result);
[propget, id(10), helpstring("property IsEmpty")] HRESULT IsEmpty([out, retval]
VARIANT_BOOL* result);
};
[
	object,
	uuid(0669AB3B-8E1B-4161-8A2E-244D5A62029A),
	dual,
	nonextensible,
	helpstring("IDispatchInfo Interface"),
	pointer_default(unique)
]
interface IGohubDispatchInfo : IDispatch{
	[id(1), helpstring("method Close")] HRESULT Close(void);
	[id(2), helpstring("method WagOwnerByIndex")] HRESULT WagOwnerByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(3), helpstring("method TypeByIndex")] HRESULT TypeByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(4), helpstring("method DateByIndex")] HRESULT DateByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(5), helpstring("method DescriptionByIndex")] HRESULT
DescriptionByIndex([in] ULONGLONG index, [out,retval] BSTR* pVal);
```

```
	[id(6), helpstring("method NumberByIndex")] HRESULT NumberByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(7), helpstring("method IsEmptyByIndex")] HRESULT IsEmptyByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(8), helpstring("method Count")] HRESULT Count([out,retval] LONG*
count);
};
[
	object,
	uuid(B8FFED27-656B-48C0-AD6F-8EBE09F8F8E8),
	dual,
	nonextensible,
	helpstring("IPSTDInfo Interface"),
	pointer_default(unique)
]
interface IGohubPSTDInfo : IDispatch{
	[id(1), helpstring("method Close")] HRESULT Close(void);
	[id(2), helpstring("method ArrivalDateByIndex")] HRESULT
ArrivalDateByIndex([in] ULONGLONG index, [out,retval] BSTR* pVal);
	[id(3), helpstring("method DepartureByIndex")] HRESULT
DepartureByIndex([in] ULONGLONG index, [out,retval] BSTR* pVal);
	[id(4), helpstring("method NumberByIndex")] HRESULT NumberByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(5), helpstring("method ReqDateByIndex")] HRESULT ReqDateByIndex([in]
ULONGLONG index, [out,retval] BSTR* pVal);
	[id(6), helpstring("method Count")] HRESULT Count([out,retval] LONG*
count);
};
[
	uuid(EB746ACA-C81E-42EE-B1C4-C435A8CD0082),
	version(1.0),
	helpstring("gohubclientcom 1.0 Type Library")
]
library gohubclientcomLib
{
	importlib("stdole2.tlb");
	[
		uuid(B1166D0A-F0FB-4526-803C-1D37F8EB5100),
		helpstring("GohubDocument Class")
	]
	coclass GohubComDocument
	{
		[default] interface IGohubDocument;
	};
	[
		uuid(77E0C042-770C-4EA1-9593-1E370E5E8EE3),
		helpstring("GohubConnection Class")
	]
	coclass GohubComConnection
	{
		[default] interface IGohubConnection;
	};
	[
		uuid(CF908D67-DAF6-43B0-9621-1DD417CFF3D7),
		helpstring("GohubClient Class")
	]
	coclass GohubComClient
	{
		[default] interface IGohubClient;
	};
	[
		uuid(D3160E70-00DC-4502-A84B-840D6929D99A),
		helpstring("GohubError Class")
	]
	coclass GohubComError
```

```
{
	[default] interface IGohubError;
};
[
	uuid(A86B2DEA-CFC4-48F5-B904-73D638E73F95),
	helpstring("GohubSignerInfo Class")
]
coclass GohubComSignerInfo
{
	[default] interface IGohubSignerInfo;
};
[
	uuid(C1B3AEDC-6673-470D-AFF0-48A0659A9BCD),
	helpstring("GohubAttachment Class")
]
coclass GohubComAttachment
{
	[default] interface IGohubAttachment;
};
[
	uuid(934AD530-23EA-481E-A365-CD37EC536474),
	helpstring("GohubComEData Class")
]
coclass GohubComEData
{
	[default] interface IGohubEData;
};
[
	uuid(72819B78-CC37-4bc5-A80F-E466E1D19AEB),
	helpstring("GohubComPiPackage Class")
]
coclass GohubComPiPackage
{
	[default] interface IGohubPiPackage;
};
[
	uuid(B5A1FD2F-3F4E-4acb-B884-A9AC65850DA2),
	helpstring("GohubComPiPackageToEData Class")
]
coclass GohubComPiPackageToEData
{
	[default] interface IGohubPiPackageToEData;
};
[
	uuid(5848DA64-79D3-41C1-B60C-C88D92508B11),
	helpstring("GohubComFdu92 Class")
]
coclass GohubComFdu92
{
	[default] interface IGohubFdu92;
};
[
	uuid(C2878FA7-4528-4C73-9FF2-D0AA15A91895),
	helpstring("GohubComGu46 Class")
]
coclass GohubComGu46
{
	[default] interface IGohubGu46;
};
[
	uuid(33BBBEE8-8FC6-42BD-9213-A0071F0791EC),
	helpstring("GohubComGu45 Class")
]
coclass GohubComGu45
{
```

```
		[default] interface IGohubGu45;
	};
	[
		uuid(BF5FCE75-E9A7-4282-BFB3-7B8AAB339098),
		helpstring("GohubComGu27 Class")
	]
	coclass GohubComGu27
	{
		[default] interface IGohubGu27;
	};
	[
		uuid(f7da867b-d7fa-4831-8c81-b8d450d34229),
		helpstring("GohubComInfServsDoc Class")
	]
	coclass GohubComInfServsDoc
	{
		[default] interface IGohubInformServicesDocument;
	};
	[
		uuid(C5C644CF-8353-470f-AFA9-A12F8396EF87),
		helpstring("GohubComDispatchInfo Class")
	]
	coclass GohubComDispatchInfo
	{
		[default] interface IGohubDispatchInfo;
	};
	[
		uuid(5f323e6c-3d6f-42b7-84d1-60fb25fe2dde),
		helpstring("GohubComPSTDInfo Class")
	]
	coclass GohubComPSTDInfo
	{
		[default] interface IGohubPSTDInfo;
	};
};
```