



Кафедра информатики  
и информационных технологий

**Б1.В.ДВ.06.02 ИНТЕГРАЦИЯ ДАННЫХ В КОРПОРАТИВНЫХ  
ИНФОРМАЦИОННЫХ СИСТЕМАХ**

**Лабораторные работы. Интеграция данных в системе «1С:Предприятие»**

**МЕТОДИЧЕСКИЕ УКАЗАНИЯ**

к лабораторным работам и самостоятельной работе

Направление подготовки  
09.03.03 ПРИКЛАДНАЯ ИНФОРМАТИКА

Профиль подготовки:  
*Прикладная информатика цифровой экономики*

Квалификация (степень) выпускника:  
бакалавр

Уфа 2021

Рекомендовано к изданию методической комиссией экономического факультета (протокол № 8 от 25.03.2021 г.)

Составитель: доцент, к.ф.-м.н. Шамсутдинова Т.М.

Рецензент: ст. преподаватель Прокофьева С.В.

Ответственный за выпуск: зав. кафедрой ИИТ, д.т.н.,  
Беляева А.С.

г.Уфа, БГАУ, Кафедра информатики и информационных технологий

## ОГЛАВЛЕНИЕ

Лабораторная работа №1  
Использование текстовых файлов для переноса данных

Лабораторная работа №2  
Интернет-протоколы HTTP и FTP

Лабораторная работа №3  
Технологии OLE и COM

Лабораторная работа №4  
Интернет-протоколы HTTP и FTP

Лабораторная работа №5  
Механизм Web-сервисов

Лабораторная работа №6  
Работа с локальной файловой системой

Лабораторная работа №7  
Передача файлов между клиентом и сервером

Библиографический список

Приложение А  
Задания для самостоятельного выполнения

## **Лабораторная работа № 1**

### **Использование текстовых файлов для переноса данных**

**Цель лабораторной работы:** ознакомиться с принципами использования текстовых файлов для переноса данных в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### **1 Общие положения**

На практике очень часто возникает необходимость в организации взаимодействия с внешним приложением из программы 1С. Пример - операции импорта или экспорта данных или внедрение одного документа в другой через технологию OLE-Automation. Система 1С:Предприятие предоставляет мощные механизмы для выполнения таких операций.

Использование текстовых файлов для переноса данных.

Для работы с файлами в системе используется специальный агрегатный тип данных – «ФС». По умолчанию в системе всегда доступен уже существующий объект с именем «ФС», к которому можно применять методы объекта типа «ФС». Кроме того, можно создать произвольное число объектов типа «ФС» при помощи функции «СоздатьОбъект(«ФС»)».

У объекта типа «ФС» имеются стандартные функции для работы с файлами: «ВыбратьФайл()» – открывает диалог выбора файла, «ВыбратьКаталог()» – открывает диалог выбора каталога, «СуществуетФайл()» – проверяет, существует ли файл с указанным именем, «КопироватьФайл()», «УдалитьФайл()», «ПереименоватьФайл()» и др.

Для работы с текстами в системе используется специальный тип данных «Текст». Средства языка позволяют выводить строку в текстовые файлы и считывать из имеющихся файлов текст с последующим разбором его по строкам.

Для чтения данных из файла используют следующие свойства этого объекта:

- Открыть(<ИмяФайла>) – открывает файл.
- КодоваяСтраница(<Режим>) – получить/установить режим кодировки. <Режим> = 0 – Windows-кодировка, 1 – DOS-кодировка
- Показать(<Заголовок>, <ИмяФайла>) – открыть окно редактирования текста.
- Записать(<ИмяФайла>) – записывает текст в файл.
- КоличествоСтрок() – количество строк в тексте.
- ПолучитьСтроку(<НомерСтроки>) – получить строку текста по номеру

#### **2 Содержание работы**

2.1 Добавим в обработку реквизит ТД типа ТекстовыйДокумент и реквизит СтрокаРазделителя, в котором будет храниться разделитель между значениями текстового файла. Перетащим их в окно элементов формы.

Затем добавим команду ЗаписатьДанныеСотрудников. Обработчик команды заполним следующим образом (пример 1).

Пример 1. Обработчик команды «ЗаписатьДанныеСотрудников»  
&НаКлиенте

*Процедура ЗаписатьДанныеСотрудников(Команда)*

*НовыйТД = Новый ТекстовыйДокумент;*

*// Установить кодировку документа UTF8. Это кодировка по умолчанию.*

*НовыйТД.УстановитьТипФайла(КодировкаТекста.UTF8);*

*Если ПустаяСтрока(СтрокаРазделителя) Тогда*

*СтрокаРазделителя = ",";*

*КонецЕсли;*

*Текст = ПолучитьСтрокиСотрудников(СтрокаРазделителя);*  
*Текст = СформироватьЗаголовок("ООО Быстрее, выше, сильнее") + Символы.ПС + Текст;*  
*НовыйТД.УстановитьТекст(Текст);*  
*НовыйТД.НачатьЗапись(, "c:\temp\doc\_text.txt");*  
*КонецПроцедуры*

В этом обработчике мы создаем объект ТекстовыйДокумент и устанавливаем тип кодировки этого документа.

Затем вызываем функцию ПолучитьСтрокиСотрудников(), в которую передаем строку разделителя значений (см. пример 2). Эта функция выполняется на сервере. В ней в цикле обхода выборки элементов справочника Сотрудники мы формируем строки с данными сотрудников и в виде одной большой строки возвращаем в процедуру ЗаписатьДанныеСотрудников(). Для объединения данных мы используем функцию глобального контекста СтрСоединить().

После этого в полученную строку мы добавляем заголовок файла обмена. Для получения заголовка используется функция СформироватьЗаголовок(). Затем методом УстановитьТекст() объекта ТекстовыйДокумент мы устанавливаем эту строку в виде текста в текстовый документ и записываем документ в текстовый файл методом НачатьЗапись().

Функции СформироватьЗаголовок() и ПолучитьСтрокиСотрудников() реализуют достигнутые при переговорах «договоренности» о формате файла обмена.

Пример 2. Функция «ПолучитьСтрокиСотрудников()»  
&НаСервере

*Функция ПолучитьСтрокиСотрудников(Разделитель)*  
*СтрокаСотрудника = Новый Массив();*  
*МассивСтрокСотрудников = Новый Массив();*  
*Выборка = Справочники.Сотрудники.Выбрать();*  
*Пока Выборка.Следующий() Цикл*  
*СтрокаСотрудника.Очистить();*  
*СтрокаСотрудника.Добавить(Выборка.Код);*  
*СтрокаСотрудника.Добавить(Выборка.Наименование);*  
*СтрокаСотрудника.Добавить(Формат(Выборка.ДатаРождения, "ДЛФ=D"));*  
*СтрокаСотрудника.Добавить(Выборка.КоличествоДетей);*  
*Данные = СтрСоединить(СтрокаСотрудника, Разделитель);*  
*МассивСтрокСотрудников.Добавить(Данные);*  
*КонецЦикла;*  
*Возврат СтрСоединить(МассивСтрокСотрудников, Символы.ПС);*  
*КонецФункции*

#### **Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

#### **Контрольные вопросы:**

1. Общие принципы работы с файлами.
2. Специфика работы с файлами в управляемом режиме «1С:Предприятие».
3. Работа с текстовым документом.
4. Элемент управления «ПолеТекстовогоДокумента».
5. Организация последовательного доступа к тексту.

## **Лабораторная работа № 2**

### **Интернет-протоколы HTTP и FTP**

**Цель лабораторной работы:** ознакомиться с принципами использования интернет-протоколов в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### **1 Общие положения**

Работа с электронной почтой возможна непосредственно из встроенного языка. Разработчик может выполнять отправку и прием писем электронной почты.

Система 1С:Предприятие 8 предоставляет две возможности работы с электронной почтой: используя почтового клиента, ранее установленного на компьютере, или без использования внешнего почтового клиента.

Механизм интернет-почты позволяет организовать обмен электронной корреспонденцией между адресатами, не имея отдельного установленного почтового клиента.

Универсальность набора объектов этого механизма позволяет решать различные задачи коммерческой деятельности предприятия, связанные с информационным обменом. К таким задачам относятся обмен информацией с клиентами предприятия, рассылка пресс-релизов в медиа-издания, обмен коммерческой информацией с контрагентами и т.д.

Решение этих задач упрощается благодаря возможности передавать и получать данные в различных форматах (текст, HTML, графика, двоичные данные и т.д.).

В основе механизма электронной почты лежат общепринятые почтовые Интернет-протоколы SMTP и POP3, что позволяет использовать уже существующие и широко распространенные в Интернете почтовые инфраструктуры, а для пользователей такая опора на известные стандарты сокращает время освоения прикладных решений.

Механизм интернет-почты позволяет:

- подключаться к почтовому серверу, указывая адреса и параметры протоколов IMAP, POP3 и SMTP;

- использовать защищённые соединения SSL/TLS и STARTTLS;

- отключаться от почтового сервера;

- отправлять почтовые сообщения, в том числе с предварительной обработкой текста сообщения перед отправкой;

- выбирать сообщения с почтового сервера, в том числе с удалением;

- удалять сообщения с почтового сервера;

- создавать новые почтовые сообщения;

- задавать отправителя сообщения, получателей, копии, тему, текст письма и перечень вложенных файлов;

- использовать различные типы текста почтового сообщения: HTML, простой текст и размеченный текст (Rich Text);

- использовать в качестве вложений двоичные данные или другие почтовые сообщения;

- задавать кодировку как всего сообщения в целом, так и отдельных его элементов;

- принимать только заголовки сообщений;

- получать исходные тексты почтовых сообщений, что позволяет строить полноценные email клиенты.

В системе поддерживается коллекция объектов различных типов для организации взаимодействия по электронной почте.

С ее помощью можно создавать, посылать и принимать сообщения, причем сообщения могут иметь несколько адресатов и присоединенных файлов, добавлять

(менять, удалять) адреса как пары (пользователь – сервер), так и полные почтовые адреса, управлять почтовыми вложениями и т.д.

Встроенный язык содержит набор объектов, которые позволяют осуществлять обмен данными по протоколам **HTTP (HTTPS)** и **FTP (FTPS)**.

#### **HTTP (HTTPS)**

Встроенный язык поддерживает обмен данными по протоколам **HTTP (HTTPS)**.

При этом разработчик может:

- создать http-соединение;
- записать, получить и удалить файл;
- отправить ресурс на указанный адрес для обработки;
- определить параметры установленного соединения
- вызвать произвольный HTTP метод;
- передать на сервер заголовки и получить заголовки с сервера;
- в качестве тела запроса отправить не только файл, но и строку (с автоматическим перекодированием в нужную кодировку), и двоичные данные;
- использовать Basic-авторизацию;
- проверить сертификат сервера с использованием стандартных механизмов ОС или файла сертификата;
- отправить на сервер сертификат клиента;
- использовать протокол TLS 1.0 для установления защищенного соединения.

#### **FTP (FTPS)**

Средствами встроенного языка доступна организация обмена данными по протоколу **FTP (FTPS)**, при этом разработчику доступны следующие возможности:

- определить активный или пассивный режим работы ftp-соединения;
- установить пользователя, от имени которого установлено соединение, порт сервера, с которым произведено соединение, сервер (прокси), через который установлено соединение;
- записать (найти) файлы объектов типа FTPФайл;
- переименовать файлы и каталоги на сервере;
- получить ресурс из указанного адреса;
- создать (удалить) каталог на сервере;
- установить текущий каталог на сервере;
- определить размер файла (в байтах);
- определить параметры файлов и каталогов;
- использовать протокол TLS 1.0 для установления защищенного соединения;
- проверить сертификат сервера с использованием стандартных механизмов ОС или файла сертификата.

На встроенном языке можно описать параметры прокси-серверов для различных протоколов. Допустимые протоколы для использования в объекте ИнтернетПрокси задаются строками http, https, ftp, ftps. Разработчик имеет возможность:

- установить пароль пользователя и имя пользователя для авторизации на прокси-сервере;
- получить порт прокси-сервера по имени протокола;
- получить прокси-сервер по имени протокола;
- установить параметры прокси-серверов для различных протоколов.

## **2 Содержание работы**

2.1 Предположим, нам нужно получить файлы определенного типа с FTP-сервера через FTP-соединение. Для этого нужно создать FTP-соединение с сервером, используя имя FTP-сервера, а также имя и пароль пользователя на этом сервере. Затем найти нужные файлы по заданной маске (например, "\*.tif") методом НайтиФайлы() в каталоге на сервере и

скопировать найденные файлы (объекты FTPФайл) в локальный каталог на компьютере пользователя методом Получить() объекта FTPСоединение.

Обработчик команды для получения файлов с FTP-сервера может выглядеть следующим образом (пример 1).

Пример 1.. Процедура «ПолучитьФайлыССервера»

*&НаКлиенте*

*Процедура ПолучитьФайлыССервера(Команда)*

*Сообщение = Новый СообщениеПользователю;*

*Попытка*

*// Установить соединение с FTP-сервером.*

*FTPSервер = Новый FTPСоединение(Сервер, , ИмяПользователя, ПарольПользователя);*

*Исключение*

*// Вывести сообщение об ошибке соединения с сервером.*

*Сообщение.Текст = "Не удалось соединиться с сервером: " + Сервер;*

*Сообщение.Сообщить();*

*Сообщение.Текст = ОписаниеОшибки();*

*Сообщение.Сообщить();*

*Возврат;*

*КонецПопытки;*

*// Найти файлы в нужном каталоге по указанной маске.*

*МаскаФайлов = "\*.tif";*

*МассивФайлов = FTPСервер.НайтиФайлы(КаталогНаСервере, МаскаФайлов);*

*Для Каждого Файл Из МассивФайлов Цикл*

*// Проверить, что это не каталог.*

*Если Файл.ЭтоФайл() Тогда*

*// Копировать файл в локальный каталог.*

*Сообщение.Текст = "Считывается файл - " + Файл.Имя;*

*Сообщение.Сообщить();*

*FTPSервер.Получить(Файл.ПолноеИмя, ЛокальныйКаталог + Файл.Имя);*

*КонецЕсли;*

*КонецЦикла;*

*КонецПроцедуры*

В первом параметре метода Получить() содержится относительный адрес ресурса на FTP-сервере (полное имя FTP-файла), из которого будут прочитаны данные. Во втором – полное имя файла, в который будут помещены данные полученного ресурса.

**Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

**Контрольные вопросы:**

1. Организация Интернет соединения.
2. Работа с электронной почтой.
3. Объекты «Почта» и «ИнтернетПочта».
4. Использование протоколов HTTP, FTP, организации соединений.



## **Лабораторная работа № 3** **Технологии OLE и COM**

**Цель лабораторной работы:** ознакомиться с принципами использования технологий OLE и COM в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

### **1 Общие положения**

Механизм OLE был задуман как технология интеграции программных продуктов Microsoft Office. Технология OLE предоставляет нам возможность работать с объектами, созданными в других приложениях (документы, рисунки, таблицы). Основные термины, с которыми оперирует данная технология, — это OLE-объект, сервер приложения и контейнер приложения.

*OLE-объектом* называют объект, созданный в другом приложении и сохранивший связь с этим приложением. Документ, редакции Word, или электронные таблицы в формате Excel — все они могут быть OLE-объектами, если будут вставлены в документ соответствующим образом. Если не вставлять их как OLE-объект, то связь с оригинальным приложением будет отсутствовать.

*Контейнером* приложения OLE называют приложение, в котором создается составной документ, позволяя обрабатывать его в исходном приложении (например, таком как Word или Excel), которое использовалось для создания этого объекта.

*Сервером* приложения OLE (OLE Server Application) называют приложение, создающее объекты, которые можно поместить в документ-контейнер.

Программы — "1С:Предприятие", Microsoft Word и Excel являются приложениями, которые могут выступать и как OLE-сервер, и как OLE-контейнер. Другими словами, эти приложения могут создавать новые OLE-объекты, а также хранить OLE-объекты, созданные в других приложениях.

С точки зрения пользователя, составной документ выглядит единым набором информации, но фактически содержит элементы, созданные двумя или несколькими разными приложениями.

Система "1С:Предприятие 8.2" может использоваться внешними приложениями в качестве OLE Automation сервера.

Обычно в этих целях "1С:Предприятие" используют для управления конфигурациями системы программ "1С:Предприятие" из других приложений и выполнения действий аналогичным интерактивным действиям пользователя (например, построение отчетов).

Для запуска системы "1С:Предприятие" в качестве OLE Automation сервера из внешнего приложения выполняется следующая последовательность действий:

1. Создается OLE Объект с идентификатором "V82.Application".
2. Выполняется инициализация системы "1С:Предприятие" методом Connect.
3. Вызываются свойства и методы системы "1С:Предприятие" как OLE Automation сервера.

### **2 Содержание работы**

2.1 Реализуем автоматический обмен данными через COMСоединение.

Автоматический обмен может быть реализован при помощи внешней программы, использующей возможности объекта COMСоединение платформы «1С:Предприятие». Данный метод может быть использован, когда изменение конфигурации (для внедрения кода

поддержки автоматического обмена) по каким-либо причинам невозможно или нежелательно.

Для примера напишем программу на языке Visual Basic.

Пример 1. Пример процедуры обмена

```
Dim connector = CreateObject("V82.COMConnector")
Dim connection = connector.connect("file=d:\DemoExchange")
Dim nodeRef = connection.ПланыОбмена.УдаленныеСклады.НайтиПоКоду("Onm")
If (Not nodeRef.Пустая()) Then
Dim node = nodeRef.ПолучитьОбъект()
node.ПрочитатьСообщениеСИзменениями()
node.ЗаписатьСообщениеСИзменениями()
End If
```

В данном примере используются те же процедуры узлов плана обмена УдаленныеСклады, что и в реализации обмена с использованием командной строки.

Полученный исполняемый модуль может быть поставлен в очередь планировщика. Пример на языке JavaScript приведен в примере 2.

Пример 2. Пример постановки задания в очередь

```
<%@ Language=javascript %>
<%
entConn = new ActiveXObject("v82.ComConnector");
conn = entConn.connect("file=d:\DemoExchange");
nodeRef = conn.ПланыОбмена.УдаленныеСклады.НайтиПоКоду("Onm");
if (nodeRef.Пустая() == false)
{
node = nodeRef.ПолучитьОбъект();
node.ПрочитатьСообщениеСИзменениями();
node.ЗаписатьСообщениеСИзменениями();
}
%>
```

Данный код можно размещать в документах \*.asp, \*.aspx.

#### **Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

#### **Контрольные вопросы:**

1. Основы технологий OLE и COM.
2. Работа с Microsoft Excel.
3. Назначение обработчиков событий на COM-объекты.
4. «1С:Предприятие 82 как OLE и COM сервер.
5. Внешние источники данных.
6. Подключение к базе данных Access, таблицам (книгам) Excel.
7. Организация связи web-приложения с информационной базой «1С:Предприятие».

## **Лабораторная работа № 4**

### **Обмен данными на базе XML**

**Цель лабораторной работы:** ознакомиться с принципами использования XML для обмена данными в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### **1 Общие положения**

Работа с XML-документами доступна непосредственно из встроенного языка системы 1С:Предприятие 8.

Имеется возможность:

- последовательно читать и записывать xml-документы:
  - o преобразовывать из строки, полученной из текста элемента или значения атрибута XML, в значение в соответствии с указанным типом;
  - o получать строковое представление значения для помещения в текст элемента или значение атрибута XML;
  - o получить тип данных XML, соответствующий переданному в качестве параметра типу;
  - o производить проверку возможности чтения из XML значения указанного типа;
  - o производить проверку соответствия схеме XML при чтении XML
  - o производить запись значения в формате XML;
  - o возвращать тип, соответствующий типу данных XML.
- использовать модель объектного доступа к данным xml-документов (ДокументDOM), соответствующую следующим стандартам:
  - o DOM Level 2;
  - o XPath (DOM Level 3);
  - o DOM Load and Save (DOM Level 3).
- использовать объектную модель схемы XML (СхемаXML).

Используя внешнее соединение и механизмы работы с XML можно организовывать интеграцию с прикладными системами по принятым в этих системах форматам. Для этого применяются механизмы XSL-преобразования. Например, для такой интеграции можно использовать BizTalk сервер компании Microsoft.

Fast Infoset.

Платформа предоставляет средства для работы с XML-документами в бинарном формате Fast Infoset. Технология Fast Infoset использует альтернативный синтаксис отображения XML-данных. Это обеспечивает меньший объем файлов и более высокую скорость обработки, чем скорость обработки данных, записанных в обычном XML-формате. Файл, записанный в формате fast infoset, имеет расширение .fi или .finf.

**Механизм XDTO** - это один из механизмов интеграции с другими системами. Аббревиатура XDTO расшифровывается как XML Data Transfer Objects. XDTO является механизмом объектного моделирования данных, описываемых с помощью схемы XML.

Основные возможности использования XDTO

- описание типов параметров и возвращаемых значений Web-сервисов;
- обмен данными между конфигурациями 1С:Предприятия 8 с существенно разными структурами данных;
- обмен данными на основе схем XML, не привязанных к той или иной конфигурации (например, обмен с информационными системами, построенными не на основе 1С:Предприятия 8);

- создание собственной системы типов и значений для обработки произвольных данных.

XDTO пакет

Механизм XDTO реализован с помощью набора объектов встроенного языка и объекта конфигурации XDTO-пакет. Они позволяют описать систему типов и значений, которая будет использоваться для взаимодействия с другими программными системами.

## 2 Содержание работы

2.1 Рассмотрим задачу обмена данными об элементах справочника Сотрудники. Рассмотрим пример использования объекта ЗаписьXML (пример 1).

Пример 1. Пример записи XML-документа

*&НаКлиенте*

*Процедура ЗаписьДанных(Команда)*

*СтрокаРазделителя = "\*";*

*ЗаписьXML = Новый ЗаписьXML;*

*ЗаписьXML.ОткрытьФайл("c:\temp\document.xml");*

*// Записать директиву.*

*ЗаписьXML.ЗаписатьОбъявлениеXML();*

*// Записать начало корневого элемента.*

*ЗаписьXML.ЗаписатьНачалоЭлемента("Корневой");*

*// Записать атрибут корневого элемента.*

*ЗаписьXML.ЗаписатьАтрибут("ИмяСправочника", "Сотрудники");*

*ЗаписьXML.ЗаписатьКомментарий("Выгрузка элементов справочника");*

*// Получить данные сотрудников в виде одной большой строки.*

*СтрокаСотрудников = ПолучитьСтрокиСотрудников(СтрокаРазделителя);*

*// Получить массив строк для каждого сотрудника.*

*СтрокиСотрудников = СтрРазделить(СтрокаСотрудников, Символы.ПС);*

*Для ТекущаяСтрока = 0 По СтрокиСотрудников.Количество() - 1 Цикл*

*// Получить данные каждого сотрудника.*

*Данные = СтрРазделить(СтрокиСотрудников[ТекущаяСтрока], СтрокаРазделителя);*

*ЗаписьXML.ЗаписатьНачалоЭлемента("ЭлементСправочника");*

*ЗаписьXML.ЗаписатьНачалоЭлемента("Код");*

*ЗаписьXML.ЗаписатьТекст(Данные[0]);*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*ЗаписьXML.ЗаписатьНачалоЭлемента("Наименование");*

*ЗаписьXML.ЗаписатьТекст(Данные[1]);*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*ЗаписьXML.ЗаписатьНачалоЭлемента("ДатаРождения");*

*ЗаписьXML.ЗаписатьТекст(Данные[2]);*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*ЗаписьXML.ЗаписатьНачалоЭлемента("КоличествоДетей");*

*ЗаписьXML.ЗаписатьТекст(Данные[3]);*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*КонецЦикла;*

*ЗаписьXML.ЗаписатьКонецЭлемента();*

*ЗаписьXML.Закрыть();*

*КонецПроцедуры*

Здесь, так же как и при записи текстового файла, вызывается серверная функция ПолучитьСтрокиСотрудников(), в которой в цикле обхода выборки элементов справочника Сотрудники формируются строки с данными сотрудников и в виде одной большой строки возвращаются на клиент. Затем мы производим обратную операцию, получаем массив строк с данными для каждого сотрудника и разбираем строки с данными с помощью функции глобального контекста СтрРазделить(). Полученные элементы массива с данными каждого сотрудника мы записываем в виде элементов XML-файла.

В результате выполнения представленного кода будет получен следующий XML-документ (пример 2).

Пример 2. Полученный XML-документ

```
<?xml version="1.0" encoding="UTF-8"?>
<Корневой ИмяСправочника="Сотрудники">
<!--Выгрузка элементов справочника-->
<ЭлементСправочника>
<Код>000000001</Код>
<Наименование>Алексеев Сергей Иванович</Наименование>
<ДатаРождения>10.12.1980</ДатаРождения>
<КоличествоДетей>1</КоличествоДетей>
</ЭлементСправочника>
<ЭлементСправочника>
<Код>REST-0003</Код>
<Наименование>Артемов Игорь Владимирович</Наименование>
<ДатаРождения>17.05.2019</ДатаРождения>
<КоличествоДетей>2</КоличествоДетей>
</ЭлементСправочника>
<ЭлементСправочника>
<Код>000000002</Код>
<Наименование>Смирнова Светлана Ивановна</Наименование>
<ДатаРождения>22.02.1990</ДатаРождения>
<КоличествоДетей>0</КоличествоДетей>
</ЭлементСправочника>
</Корневой>
```

Сначала создается объект ЗаписьXML. Через созданный объект в модели последовательного доступа будет производиться запись данных в XML-документ. С помощью метода ОткрытьФайл() указывается имя файла, куда будет производиться запись данных. Метод ЗаписатьОбъявлениеXML() определяет директиву, что создаваемый документ будет являться XML-документом (директива размещается в первой строке созданного документа).

Для записи элемента XML-документа используются методы ЗаписатьНачалоЭлемента(), ЗаписатьАтрибут(), ЗаписатьТекст(), ЗаписатьКонецЭлемента(), которые производят запись соответствующего узла элемента XML в файл. При этом важно соблюдать порядок (иерархию) вызова этих методов.

В заключение методом Закреть() объекта ЗаписьXML завершается запись, и файл с данными закрывается.

**Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

**Контрольные вопросы:**

1. Базовые средства работы с XML.
2. XML сериализация.
3. Простые и сложные типы данных.
4. Выгрузка и загрузка объектов с различающейся структурой.
5. Объектная модель XML-документа.
6. Работа с XML-парсером

## **Лабораторная работа № 5**

### **Механизм Web-сервисов**

**Цель лабораторной работы:** ознакомиться с принципами использования Web-сервисов в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### **1 Общие положения**

Web-сервисы - это один из механизмов платформы, используемых для интеграции с другими информационными системами. Он является средством поддержки SOA (Service-Oriented Architecture) - сервис-ориентированной архитектуры, которая является современным стандартом интеграции приложений и информационных систем.

Значительным преимуществом сервис - ориентированной архитектуры является то, что она позволяет развивать инфраструктуру предприятия однородным образом, без разрушения уже существующих решений. Ее использование позволяет минимизировать издержки за счет интеграции разнородных и унаследованных систем в современный ландшафт предприятия. Она позволяет реализовывать слабо связанные программные компоненты с тем, чтобы максимально повысить их повторную используемость.

Сервис-ориентированная архитектура интенсивно развивается и поддерживается крупными вендорами. Она строится на базе сервисов, автономных или управляемых извне. Предпочтительным способом их реализации являются веб-сервисы. Они независимы от платформы, автономны и поддерживаются повсеместно.

Прикладное решение 1С:Предприятия 8 может являться как поставщиком веб-сервисов, так и потребителем веб-сервисов, опубликованных другими поставщиками.

1С:Предприятие - поставщик веб-сервисов:

В конфигурацию можно добавить специальный объект, - Web-сервис,- с помощью которого описать некоторую функциональность прикладного решения, например, получение списка имеющихся на некотором складе товаров, их количества и цен. После публикации на веб-сервере такой сервис будет доступен сторонним потребителям.

1С:Предприятие - потребитель веб-сервисов:

В прикладном решении можно описать ссылку на веб-сервис, опубликованный сторонним поставщиком. После этого прикладное решение сможет использовать данные, получаемые с помощью такого веб-сервиса, в своих внутренних прикладных алгоритмах.

Техническая реализация web-сервисов

Если прикладное решение является поставщиком веб-сервиса то и в файловом, и в клиент-серверном варианте работы взаимодействие между прикладным решением и потребителями веб-сервиса осуществляется через веб-сервер, с помощью модуля расширения веб-сервера.

При этом, когда потребитель обращается к web-сервису прикладного решения, выполняется модуль web-сервиса. Этот модуль содержится в конфигурации и в нем располагаются процедуры, выполняемые при вызове тех или иных операций web-сервиса.

В случае клиент-серверного варианта работы этот модуль будет исполняться в кластере. В случае файлового варианта работы - в модуле расширения веб-сервера.

Если прикладное решение является потребителем веб-сервиса стороннего поставщика, то в этом случае взаимодействие между прикладным решением и поставщиком веб-сервиса осуществляет клиентское приложение. Оно вызывает те или иные операции веб-сервиса и обрабатывает полученные данные.

Система «1С:Предприятие» может использовать Web-сервисы, предоставляемые другими поставщиками, следующими способами:

- с помощью статических ссылок, создаваемых в дереве объектов конфигурации;
- с помощью динамических ссылок, создаваемых средствами встроенного языка;
- комбинацией предыдущих способов.

При использовании статической ссылки система «1С:Предприятие» получает описание Web-сервиса поставщика только один раз, при создании ссылки. За счет этого достигается большая скорость работы.

При использовании динамической ссылки описание Web-сервиса получается каждый раз при вызове Web-сервиса. Скорость работы при этом уменьшается, но зато такой подход обеспечивает актуальность описания Web-сервиса поставщика. В случае же использования статических ссылок для получения актуального описания Web-сервиса требуется выполнить повторный импорт WSDL-описания средствами конфигуратора и затем сохранить измененную конфигурацию.

## 2 Содержание работы

### 2.1 Создадим WS-прокси на основании WS-ссылки (пример 1).

Пример 1. Использование статической WS-ссылки

*// Создать WS-прокси на основании WS-ссылки.*

*Прокси =*

*WSCсылки.ДанныеРасходнойНакладной.СоздатьWSПрокси("http://www.MyCompany.ru/shipment", "ДанныеРасходнойНакладной", "ДанныеРасходнойНакладнойSoap");*

При использовании динамической ссылки на этот Web-сервис WS-прокси создается на основании WS-определения (пример 2).

Пример 2. Использование динамической WS-ссылки

*// Создать WS-прокси на основании WS-определения.*

*Определение = Новый*

*WSОпределения("http://www.MyCompany.ru/shipment/ws/Shipment.1cws?wsdl");*

*Прокси = Новый WSПрокси(Определение, "http://www.MyCompany.ru/shipment", "ДанныеРасходнойНакладной", "ДанныеРасходнойНакладнойSoap");*

На практике бывают ситуации, когда один и тот же Web-сервис предоставляется по разным адресам (URL), однако имеет абсолютно одинаковое описание (WSDL). Например, когда используется тиражируемый сервис, выполняющий некоторую функцию. При этом адрес сервиса может быть различным. В этом случае можно использовать комбинированный способ. То есть сначала загрузить в конфигурацию описание Web-сервиса (добавить в дерево объектов конфигурации статическую ссылку на Web-сервис), а затем во время использования указать конкретный адрес, по которому расположен Web-сервис.

Например, если адрес реального расположения сервиса отличается от адреса, который использовался во время загрузки описания Web-сервиса в конфигурацию, то можно предусмотреть ввод адреса конкретного экземпляра сервиса в настройках прикладного решения, и затем этот новый адрес явно указать при создании объекта WSПрокси (пример 3).

Пример 3. Комбинированный способ на основе статической ссылки

*// Создать WS прокси на основании ссылки.*

*Прокси =*

*WSCсылки.ДанныеРасходнойНакладной.СоздатьWSПрокси("http://www.MyCompany.ru/shipment", "ДанныеРасходнойНакладной", "ДанныеРасходнойНакладнойSoap");*



*ent", "ДанныеРасходнойНакладной", "ДанныеРасходнойНакладнойSoap", , , ,  
"http://www.realURL/realPath");*

Или же можно использовать динамическую ссылку, но адрес расположения Web-сервиса получать не из файла описания (WSDL), а непосредственно указывать при создании объекта (пример 4).

Пример 4. Комбинированный способ на основе динамической ссылки

*// Создать WS-прокси на основании WS-определения.*

*Определение = Новый*

*WSОпределения("http://www.MyCompany.ru/shipment/ws/Shipment.1cws?wsdl");*

*Прокси = Новый WSПрокси(Определение, "http://www.MyCompany.ru/shipment",*

*"ДанныеРасходнойНакладной", "ДанныеРасходнойНакладнойSoap", , , ,*

*"http://www.realURL/realPath");*

При попытке загрузить описание Web-сервиса в конфигураторе (создание статической ссылки) или при использовании динамической ссылки (при помощи объекта WSOпределения) система выполняет проверку загружаемого описания Web-сервиса (WSDL). Если в описании Web-сервиса присутствует ошибка (с точки зрения системы «1С:Предприятие»), то описание не будет загружено и будет сгенерировано исключение. В тексте исключения будет находиться подробная диагностика причин отказа в загрузке.

2.2 Выполнить задание для самостоятельной работы из приложения А (по указанию преподавателя)

**Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

**Контрольные вопросы:**

- 1 Основы сервисно-ориентированной архитектуры (SOA).
- 2 Язык описания сервисов WSDL.
- 3 Реализация протокола SOAP.
- 4 Сериализация сообщений и вызов сервисов.
- 5 Создание WEB-сервисов (SOAP) в «1С:Предприятие».
- 6 Использование WEB-сервисов, опубликованных сторонними поставщиками.
- 7 Использование динамических и статических ссылок.
- 8 REST web-сервисы

## Лабораторная работа № 6

### Работа с локальной файловой системой

**Цель лабораторной работы:** ознакомиться с принципами работы с локальной файловой системой в «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### 1 Общие положения

Для работы с файлами на локальном клиентском компьютере используются свойства и методы объекта Файл, а также различные процедуры и функции работы с файлами, которые представлены глобальным контекстом платформы.

С помощью объекта Файл можно получить короткое имя файла, расширение, имя без расширения, каталог, в котором файл находится, и т. п.

Среди часто встречающихся задач – поиска и удаления файлов, создания и удаления каталога и т. д.

В форме обработки используются следующие параметры, которые хранятся в соответствующих строковых реквизитах формы:

- МаскаФайлов – маска, по которой требуется найти или удалить файлы в каталоге. Например, «\*.xml»;
- ИмяНовогоФайла – имя для копирования выбранного файла;
- ИмяНовогоКаталога – имя для создания нового каталога.

#### 2 Содержание работы

2.1 Предположим, нам нужно скопировать файл, находящийся в определенном каталоге, в тот же каталог, но с новым именем, которое задано в поле ввода демонстрационной обработки (реквизит ИмяНовогоФайла).

Для решения этой задачи добавим команду КопироватьФайлСНовымИменем. Обработчик команды заполним следующим образом.

Пример 1. Обработчик команды «КопироватьФайлСНовымИменем»

*&НаКлиенте*

*Процедура КопироватьФайлСНовымИменем(Команда)*

*Диалог = Новый ДиалогВыбораФайла(РежимДиалогаВыбораФайла.Открытие);*

*Диалог.Заголовок = "Выберите файл";*

*Диалог.Показать(Новый ОписаниеОповещения("КопироватьФайлЗавершение",  
ЭтотОбъект));*

*КонецПроцедуры*

В этом обработчике мы выбираем файл, который нам нужно скопировать. Для этого мы показываем пользователю диалог выбора файла с помощью немодального метода Показать(). В этот метод мы передаем описание оповещения, указывающее на экспортную процедуру КопироватьФайлЗавершение(), которая будет выполнена после того, как пользователь выберет файл для копирования.

Пример 2. Процедура «КопироватьФайлЗавершение()»

*&НаКлиенте*

*Процедура КопироватьФайлЗавершение(ВыбранныеФайлы, Параметры) Экспорт*

*Если ВыбранныеФайлы = Неопределено Тогда*

```

Возврат;
КонецЕсли;
ИмяФайлаИсточника = ВыбранныеФайлы[0];
ФайлИсточник = Новый Файл(ИмяФайлаИсточника);
ИмяФайлаПриемника = ФайлИсточник.Путь + СокрЛП(ИмяНовогоФайла) +
ФайлИсточник.Расширение;
НачатьКопированиеФайла(Новый
ОписаниеОповещения("КопироватьФайлСНовымИменемЗавершение", ЭтотОбъект),
ИмяФайлаИсточника, ИмяФайлаПриемника);
КонецПроцедуры

```

В этом обработчике оповещения, в случае если файл выбран, мы получаем полный путь к файлу-источнику как первый элемент массива, содержащийся в параметре ВыбранныеФайлы (для простоты примера будем выбирать только один файл).

Затем создаем на основе пути к файлу-источнику объект Файл. Используя свойства этого объекта, получаем отдельно путь (ФайлИсточник.Путь) и расширение (ФайлИсточник.Расширение) файла-источника. Вставив имя нового файла, мы получаем полный путь к файлу-приемнику. После этого мы вызываем процедуру НачатьКопированиеФайла(), в которую передаем полные пути к файлу-источнику и файлу-приемнику.

Первым параметром в немодальный метод НачатьКопированиеФайла() мы передаем описание оповещения, указывающее на экспортную процедуру КопироватьФайлСНовымИменемЗавершение(), которая будет вызвана по окончании копирования файла.

Пример 3. Процедура «КопироватьФайлСНовымИменемЗавершение()»

*&НаКлиенте*

```

Процедура КопироватьФайлСНовымИменемЗавершение(СкопированныйФайл,
Дополнительно) Экспорт
РезультирующийФайл = Новый Файл(СкопированныйФайл);
Сообщение = Новый СообщениеПользователю;
Сообщение.Текст = "Выбранный файл скопирован в: " + РезультирующийФайл.Имя;
Сообщение.Сообщить();
КонецПроцедуры

```

В этой процедуре в параметре СкопированныйФайл содержится полный путь к новому файлу. На его основе мы создаем объект Файл. И затем показываем короткое имя нового файла (имя с расширением – РезультирующийФайл.Имя) в сообщении пользователю.

#### **Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

#### **Контрольные вопросы:**

- 1 Приведите свойства и методы объекта Файл.
- 2 Охарактеризуйте процедуры и функции работы с файлами.
- 3 Охарактеризуйте обработчик копирования файла.

## Лабораторная работа № 7

### Передача файлов между клиентом и сервером

**Цель лабораторной работы:** ознакомиться с принципами передачи файлов между клиентом и сервером в системе «1С: Предприятие».

**Требования к организации рабочего места:** лабораторная работа должна проводиться в компьютерном классе с установленной системой «1С: Предприятие».

#### 1 Общие положения

Для передачи файлов между клиентом и сервером через временное хранилище в платформе «1С:Предприятия» реализованы новые методы глобального контекста `НачатьПомещениеФайлаНаСервер()`, `НачатьПомещениеФайловНаСервер()`, `НачатьПолучениеФайлаССервера()`, `НачатьПолучениеФайловССервера()`. Существенным преимуществом этих методов является возможность их работы в интерактивном режиме (например, при отображении диалога выбора помещаемых/получаемых файлов). В этом режиме работы в веб-клиенте методы не требуют установки расширения работы с файлами.

При помещении файлов на сервер есть возможность отображения прогресса помещения файлов. Можно проверить характеристики помещаемых файлов и отказаться от загрузки перед началом или в процессе помещения файлов на сервер. Кроме того, можно несколько раз вызывать методы для помещения файлов на сервер, не дожидаясь окончания предыдущего вызова.

При получении файлов с сервера есть очень удобная возможность получить несколько файлов в виде архива.

При выполнении некоторых операций в веб-клиенте может потребоваться получить разрешение на операции по работе с файлами. Чтобы не подтверждать каждую такую операцию по отдельности, можно воспользоваться методом `НачатьЗапросРазрешенияПользователя()`. При использовании этого метода пользователю отображается список всех операций, которые планируется выполнить, и предлагается разрешить выполнение группы операций. Если пользователь разрешил выполнение, то запрошенные операции будут выполняться без дополнительных запросов пользователю. Если разрешение не предоставлено, операции будут выполняться в обычном режиме: один запрос на каждую операцию.

В тонком и толстом клиентских приложениях метод `НачатьЗапросРазрешенияПользователя()` всегда возвращает значение `Истина`, без взаимодействия с пользователем.

#### 2 Содержание работы

2.1 Предположим, у нас есть сформированный ранее JSON-файл (пример 1) и нам нужно прочитать из него информацию, изменить ее и записать в новый файл. Хотя сделать все это можно и на клиенте, более правильно – передать файл на сервер (поместить его на сервере во временное хранилище), обработать там содержимое файла, вернуть адрес измененных данных на клиент и затем по этому адресу получить файл.

Пример 1. Исходный JSON-файл

```
{
  "Контрагент": "ОАО Тоназ",
  "ОбъемПродаж": 5000000,
  "Адрес": "Страна: Россия, Индекс: 112233, Город: Москва",
  "Поставщик": false
}
```

Например, в показанном выше файле с информацией о контрагенте нам нужно заменить вывод значений true/false поля Поставщик (типа Булево) на более привычные Да/Нет.

Для решения этой задачи добавим команду ОбработатьФайлНаСервере. Обработчик команды заполним следующим образом (пример 2).

Пример 2. Обработчик команды «ОбработатьФайлНаСервере»

*&НаКлиенте*

*Процедура ОбработатьФайлНаСервере(Команда)*

*ОповещениеОЗавершении = Новый*

*ОписаниеОповещения("ОбработатьФайлНаСервереЗавершение", ЭтотОбъект);*

*НачатьПомещениеФайлаНаСервер(ОповещениеОЗавершении, , , "", ,*

*УникальныйИдентификатор);*

*//НачатьПомещениеФайлаНаСервер(ОповещениеОЗавершении, , , "", "c:\temp\example.json",*

*УникальныйИдентификатор);*

*КонецПроцедуры*

В этом обработчике с помощью метода глобального контекста НачатьПомещениеФайлаНаСервер() мы начинаем помещение файла с данными из локальной файловой системы во временное хранилище.

При этом пользователю предлагается диалог для выбора помещаемого файла. Параметры этого диалога мы можем описать и передать в пятом параметре метода. Но это будет сделано в следующем примере. В данном случае мы опускаем этот параметр, но диалог все равно будет показан.

Можно также использовать вариант вызова этого метода с указанием в пятом параметре имени помещаемого файла (см. закомментированный фрагмент кода), но следует иметь в виду, что для работы этого метода в веб-клиенте потребуется установить расширение для работы с файлами. В то время как в первом случае это расширение не требуется.

В метод НачатьПомещениеФайлаНаСервер() первым параметром мы передаем описание оповещения, указывающее на экспортную процедуру ОбработатьФайлНаСервереЗавершение(), которая будет выполнена, после того как файл с данными будет помещен во временное хранилище.

Пример 3. Процедура «ОбработатьФайлНаСервереЗавершение()»

*&НаКлиенте*

*Процедура ОбработатьФайлНаСервереЗавершение(ОписаниеПомещенногоФайла, Дополнительно) Экспорт*

*ПомещенныйФайл = ОписаниеПомещенногоФайла.СсылкаНаФайл.Файл;*

*ПолучаемыйФайл = ПомещенныйФайл.Путь + ПомещенныйФайл.ИмяБезРасширения + "\_2" + ПомещенныйФайл.Расширение;*

*АдресДокументаВХранилище = ОбработкаФайла(ОписаниеПомещенногоФайла.Адрес);*

*//НачатьПолучениеФайлаССервера(АдресДокументаВХранилище);*

*НачатьПолучениеФайлаССервера(АдресДокументаВХранилище, ПолучаемыйФайл);*

*КонецПроцедуры*

После того как файл будет помещен, в параметре ОписаниеПомещенногоФайла будет содержаться описание помещенного файла (адрес данных в хранилище и ссылка на файл). Адрес помещенного файла (ОписаниеПомещенногоФайла.Адрес) мы передаем в серверную функцию ОбработкаФайла(), в которой данные по этому адресу получают из временного хранилища, записываются во временный файл и производится чтение файла.

Пример 4. Функция «ОбработкаФайла()»  
*&НаСервереБезКонтекста*  
*Функция ОбработкаФайла(АдресДокументаВХранилище)*  
*ДанныеДокумента = ПолучитьИзВременногоХранилища(АдресДокументаВХранилище);*  
*ИмяФайла = КаталогВременныхФайлов() + "temp.xml";*  
*ДанныеДокумента.Записать(ИмяФайла);*  
*Чтение = Новый ЧтениеJSON;*  
*Чтение.ОткрытьФайл(ИмяФайла);*  
*Запись = Новый ЗаписьJSON;*  
*Запись.УстановитьСтроку();*  
*Запись.ЗаписатьНачалоОбъекта();*  
*Пока Чтение.Прочитать() Цикл*  
*Если Чтение.ТипТекущегоЗначения = ТипЗначенияJSON.ИмяСвойства Тогда*  
*Если Чтение.ТекущееЗначение = "Поставщик" Тогда*  
*Чтение.Прочитать();*  
*Запись.ЗаписатьИмяСвойства("Поставщик");*  
*Запись.ЗаписатьЗначение(Строка(Чтение.ТекущееЗначение));*  
*Иначе*  
*Запись.ЗаписатьИмяСвойства(Чтение.ТекущееЗначение);*  
*КонецЕсли;*  
*КонецЕсли;*  
*Если Чтение.ТипТекущегоЗначения = ТипЗначенияJSON.Число Или*  
*Чтение.ТипТекущегоЗначения = ТипЗначенияJSON.Строка Тогда*  
*Запись.ЗаписатьЗначение(Чтение.ТекущееЗначение);*  
*КонецЕсли;*  
*КонецЦикла;*  
*Чтение.Закрыть();*  
*Запись.ЗаписатьКонецОбъекта();*  
*СтрокаJSON = Запись.Закрыть();*  
*Возврат ПоместитьВоВременноеХранилище(СтрокаJSON);*  
*КонецФункции*

В данной функции в режиме потокового чтения и записи производятся чтение, изменение и запись данных в формате JSON. Отметим, что для поля Поставщик типа Булево мы записываем строковое представление значения, в результате значения true/false будут заменены на Да/Нет.

В заключение в функции ОбработкаФайла() измененные данные записываются в строку JSON, которая помещается во временное хранилище, и адрес этих данных в хранилище возвращается в процедуру ОбработатьФайлНаСервереЗавершение(), приведенную выше.

Затем в этой процедуре с помощью метода глобального контекста НачатьПолучениеФайлаССервера() мы начинаем получение файла по адресу во временном хранилище и сохраняем файл в локальную файловую систему пользователя. При этом мы явно указываем имя получаемого файла.

Отметим, что для работы этого варианта метода (с указанием имени получаемого файла) в веб-клиенте требуется установить расширение работы с файлами.

Для формирования имени получаемого файла мы на основе описания помещенного файла (ОписаниеПомещенногоФайла.СсылкаНаФайл.Файл) создаем объект Файл. Используя свойства этого объекта: Путь, ИмяБезРасширения и Расширение, мы формируем имя сохраняемого файла с постфиксом «\_2».

**Отчет по лабораторной работе должен содержать:**

- результаты выполнения заданий;
- разработанный программный код;
- ответы на контрольные вопросы.

**Контрольные вопросы:**

- 1 Как реализовать передачу файлов между клиентом и сервером через временное хранилище.
- 2 Охарактеризуйте функцию ОбработкаФайла().
- 3 Охарактеризуйте метод НачатьПомещениеФайлаНаСервер().

**Библиографический список**

- 1) Хрусталева Е. Ю. Технологии интеграции 1С:Предприятия 8.3 [Электронный ресурс]. - М. : 1С-Паблишинг, 2020. - 503 с. – Режим доступа: <https://its.1c.eu/db/intgr83>
- 2) Дадян Э. Г. 1С: Предприятие. Проектирование приложений [Электронный ресурс]: учебное пособие / Э.Г. Дадян. - М.: Вузовский учебник: НИЦ ИНФРА-М, 2015. - 288 с. – Режим доступа: <http://znanium.com/bookread2.php?book=480629>
- 3) Дадян Э. Г. Проектирование бизнес-приложений в системе "1С: Предприятие 8": Учебное пособие / Э.Г. Дадян. - М.: Вузовский учебник: НИЦ ИНФРА-М, 2014. - 283 с. – Режим доступа: <http://znanium.com/bookread2.php?book=416778>
- 4) Радченко М. Г. 1С: Предприятие 8.2. Практическое пособие разработчика. Примеры и типовые приемы [Текст] / М. Г. Радченко, Е. Ю. Хрусталева. - М. : 1С-Паблишинг, 2009. - 872 с. + 1 эл. опт. диск (CD-ROM)

## **Приложение А**

### **Задания для самостоятельного выполнения**

Реализовать следующие примеры HTTP-сервисов.

Создать HTTP-сервисы, с помощью которых можно получать информацию из соответствующей базы:

- 1 Информация о сотрудниках:
  - 1.1 Получить список сотрудников
  - 1.2 Получить данные о конкретном сотруднике
  - 1.3 Удалить данные о конкретном сотруднике
- 2 Информация о ценах на товары:
  - 2.1 Получить цены товаров на определенную дату
  - 2.2 Получить данные о ценах на конкретный товар
  - 2.3 Проиндексировать цены на товары