ФОП Шахраманов В.В.

Квитка

Администрирование и программирование

Шахраманов В.В.(с)
04.12.2018

Оглавление

1 О программе

Программа «Квитка» построена по принципу *Клиент – Сервер* где клиентом выступает собственно сама программа, а сервером является сервер базы данных (SQLite и MySQL в данной редакции).

Программа построена как среда управления и исполнения конфигурациями. В программе предусмотрены два режима работы:

- Пользовательская система
- Конструктор

Структура состоит из набора объектов с заданными свойствами, набора описания экранных форм, набора описания печатных форм(макетов) и программного кода.

Описание классов, используемых в программе прилагается в файле ***ClassDescription.chm***

1.1 Конструктор

Режим работы Конструктор позволяет вносить изменения в существующую конфигурацию данных, создавать новые объекты, создавать экранные и печатные формы объектов, писать программный код логики поведения объектов. Также с помощью конструктора можно администрировать базу данных, выгружать и загружать данные.

1.1.1 Работа с конструктором

Дерево структуры отображает текущую конфигурацию. При изменении структуры в ее названии появляется признак измененности «*». Если при изменении объектов структуры необходимо произвести обновление базы данных, то в названии появляется признак «(!)».

Сохранение структуры происходит при нажатии кнопки на панели функций *Сохранить структуру* или же через одноименные пункты меню. Клавиатурное сокращение для сохранения структуры – Ctrl+S. Если не сохранить структуру и попытаться закрыть Конструктор, будет выведен диалог с предупреждением, что структура не записана.

Обновление базы данных при изменении структуры происходит при нажатии кнопки на панели функций *Обновить базу данных* или же через одноименные пункты меню. Клавиатурное сокращение для обновления базы данных – F6. Клавиатурное сокращение действует даже если структура не менялась.

Запуск Пользовательского режима осуществляется посредством нажатия кнопки на панели функций *Квитка* или посредством клавиатурного сокращения F5. Если программа Квитка запущена из под конструктора, то кнопка меняет свое название на *Перезапуск Квитка* и нажатие на нее приводит к перезапуску Пользовательского режима.

1.1.1.1 Функции меню Файл

Открыть - открывает внешний файл заданного типа(Внешний отчет, печатная форма)

Параметры – открывает диалог параметров программы

1.1.1.2 Функции меню Структура

Открыть структуру – открывает дерево структуры

Сохранить структуру в файл – сохраняет структуру во внешний файл с расширением *.str

Загрузить структуру из файла – загружает структуру из внешнего файла *.str полностью заменяя текущую структуру.

Объединить структуру из файла – вызывает окно пообъектного сравнения и объединения текущей структуры и структуры из внешнего файла.

Проверить обновления – запускает поиск обновления структуры через сервер обновления и установку найденного обновления.

1.1.1.3 Функции меню Сервис

Квитка/Перезапуск Квитка – запускает или перезапускает Пользовательский режим

Конструктор запроса – открывает форму конструктора запроса

Запрос к данным - открывает редактор прямых SQL запросов к базе данных

Страница разработчика – вспомогательные инструменты для разработчика конфигурации

Пользователи онлайн – работающие в текущий момент пользователи системы

Журнал действий – досье на все операции и события, происходящие в Пользовательском режиме

Сохранить данные – создание файла с выгрузкой данных. Файл это архив zip, содержит в себе структуру и xml-таблицы с данными. Имя таблиц с данными - это GUID объекта структуры.

Восстановить данные – восстанавливает данные в базе данных из внешней выгрузки. При этом структура текущая структура заменяется структурой из выгрузки.

Обслуживание БД – сервисные операции с базой данных, контроль целостности, сжатие и др.

1.1.1.4 Функции меню Помощь

Активация – открывает диалог активации программы

Информация о программе и лицензии – открывает окно с информацией о лицензии, о параметрах подключения к базе данных и др.

1.2 Файл структуры

Структура содержится в файле *structure.str* в каталоге размещения данных, задаваемых в свойствах базы данных в диалоговом окне запуска программы.

Также структура сохраняется в самой базе данных и является приоритетной. Это означает что при запуске любого режима программы имеющийся в каталоге данных файл *structure.str* будет переименован в *old_structure.str* и будет загружен файл структуры из базы данных. Если же в базе данных структуры нет, то использоваться будет файл, размещенный в каталоге данных.

В базе данных файл структуры хранится в сжатом виде (zip архив)

1.3 Ссылка

Ссылка это системное поле в справочниках, документах, истории, регистрах.

Ссылка это строка, которая состоит из двух частей, разделенных символом |:

- Guid объекта структуры
- ID записи элемента в таблице базы данных

Пример:
e4bd2004-d894-41b3-adb6-34287db9fcdf|1

2 Свойства структуры

При двойном щелчке на название структуры откроется форма свойств структуры. В этой форме можно задать следующие свойства структуры:

- Указать название структуры
- Указать авторскую информацию о структуре
- Управлять идентификатором конфигурации

Авторская информация может быть закрыта от изменения паролем, установленным автором конфигурации. Стандартные конфигурации находятся на поддержке и имеют установленный идентификатор обновления. Если необходимо снять с поддержки конфигурацию, то необходимо нажать на кнопку *Снять с поддержки* и сохранить структуру.

Идентификатор структуры – это GUID присвоенный структуре и служит для идентификации.

3 Системные объекты структуры

Системные объекты структуры включают в себя перечень объектов, предназначенных для системного функционирования Квитки в пользовательском режиме. Создание, редактирование, удаление и другие функции с системными объектами проводятся с помощью *контекстного меню* при щелчке на любую ветку, находящуюся в разделе Система

3.1 Права

Настройка прав на доступ к объектам структуры для Пользовательского режима. В структуре всегда присутствуют *Полные права*, которые дают абсолютный доступ ко всем объектам и функциям.

Права делятся на подразделы:

- Система
- Справочники
- Документы
- Отчеты
- История

Для каждого объекта структуры задается свой набор разрешений. Новосозданные объекты системы по умолчанию не имеют никаких прав и их необходимо устанавливать.

3.2 Пользователи

3.3 Меню

Конструктор набора меню позволяет составить меню, которое будет отображаться в Пользовательском режиме. Для каждого пункта меню можно выбрать/удалить изображение или же сгенерировать изображение из наименования пункта меню.

Если необходимо создать разделитель, то в наименовании пункта меню необходимо указать «-»

Если установить галочку *Показывать на стартовой панели*, то этот пункт меню будет отображен на стартовой панели в Пользовательском режиме.

3.4 Модули программы

Модуль это статический класс, область действия которых распространяется на весь код конфигурации

Один из модулей может быть назначен *Главным модулем*, для того, чтобы разместить в нем функции, вызываемые при старте и при завершении работы.

Все объявления в Модуле должны производиться как *static*

Предопределенные методы модуля, назначенного *Главным модулем*:

```
public static void OnStartProgram()
```
Вызывается при старте программы

```
public static bool OnCloseProgram()
Вызывается при закрытии программы, если вернуть false то программа не закроется
```

3.5 Задания

Задания это периодически повторяющийся программный код. Выполнение кода происходит в фоновом режиме.

При создании задания необходимо задать периодичность повторения, указать дни или месяца выполнения задания.

При указании диапазона выполнения необходимо обращать внимание на время начала выполнения задания. Это время срабатывания задания, например ежедневное задание стартует с 01.07.2018 – 12:00, это значит что ежедневно в 12:00 будет выполняться это задание. Если время выполнения задания в этот день указано раньше, чем была запущена программа в Пользовательском режиме, то задание все равно выполнится.

При выполнении задания вызывается функция из кода:

```
public static void Выполнить()
```

3.6 Запросы

Системные запросы предназначены для вызова из любого места программного кода. Системные запросы по функциональности ничем не отличаются от запросов, заданных в свойствах объектов.

Вызов системного запроса из кода осуществляется с помощью хелпера

```
Action.QRunSystemQuery(string queryName, Dictionary queryParams, Boolean isTotals);
```

Где queryName – имя системного запроса, queryParams – словарь параметров, isTotals-флаг подсчета итогов.

Словарь параметров задается следующим образом:

```
Dictionary<string, string> queryParams = new Dictionary<string, string>();
queryParams.Add("paramName1", value);
queryParams.Add("paramName2", value);
queryParams.Add("paramName3", value);
```

4 Основные объекты структуры

4.1 Поле

Объект Поле является ключевым и входит в стостав других объектов, таких как Справочник, Документ, Регистр, История, Табличная часть Документа и Справочника.

Поле может быть системным(отображается синим цветом) и пользовательское (отображается зеленым цветом)

Поле состоит из:

- GUID – уникальный идентификатор поля в структуре.
- Идентификатор – строковое имя поля. Все вызовы поля в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление поля.
- Тип поля – содержит указание типа поля из следующих типов:
    - String - строковое значение, ограниченное длиной значения
    - Number – числовое значение
    - DateTime – значение типа Дата и Время
    - Boolean – булево значение
    - Reference - ссылка на объект-справочник
    - Document - ссылка на объект-документ
    - BLOB – поле для хранения двоичных данных. *Не рекомендуется применять в текущих релизах!*
    - Image – поле для хранения изображения.
    - History - ссылка на объект-история
    - Enumeration - ссылка на объект-перечисление
- Объект - задается для типов поля Reference, Document, History, Enumeration

- Длина значения – длина значения поля в случае типов String, Number.
- Точность – количество знаков после запятой(разрядность) для типа Number
- Индексировать – признак записи этого поля в индексный массив базы данных.
- Данные регистра – признак того, что это поле является данными регистра. Устанавливается автоматически

В текущем релизе поле может быть с незаданным Объектом. Это означает, что в этом поле может быть указан любой объект данного типа, заданный программно из кода.

4.2 Справочники

Справочник предназначен для хранения массивов ссылочных данных. Физически справочник соответствует одной таблице в базе данных и таблицам табличных частей в базе данных.

Справочник содержит:

- список полей
- список табличных частей
- список экранных форм
- список печатных форм
- список запросов

4.2.1 Свойства

В свойствах справочника задаются:

- Идентификатор - строковое имя справочника. Все вызовы справочника в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление справочника.

Экранные формы:

- Экранная форма записи – указывается экранная форма записи справочника
- Экранная форма списка – указывается экранная форма списка справочника
- Экранная форма выбора – указывается экранная форма списка выбора справочника

Если экранные формы не указаны, то система автоматически сгенерирует экранные формы на основе полей и табличных частей справочника.

Поля для отображения – указываются поля для отображения записей справочника в Пользовательском режиме, всегда должно быть выбрано одно поле, например Имя. Если выбрано несколько полей, то они отображаются в Пользовательском режиме через запятую.

Поля для поиска – указываются поля, по которым будет осуществляться поиск среди полей справочника.

4.2.2 Поля

Содержит перечень системных полей и полей, заданных пользователем.

Системные поля:

- Код - поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Номер – строковое поле номера записи, автоинкрементно, доступно для изменения, задается автоматически при записи в базу данных.
- Имя – строковое поле длинной 100 символов, имя записи.
- Удален – булево поле, признак пометки удаления.
- Группа – булево поле, признак, что запись справочника является группой.
- Родитель – Reference этого же объекта, ссылка на родительскую группу справочника (п.п. 1.3).
- Ссылка - Reference этого же объекта, ссылка на запись справочника (п.п. 1.3).

4.2.3 Табличные части

Список табличных частей справочника задает табличные части, принадлежащие этому справочнику.

Табличные части содержат список системных полей и полей, заданных пользователем.

Системные поля:

- Код – поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Справочник – Reference этого же объекта справочника (п.п. 1.3).

4.2.4 Экранные формы

Содержит список экранных форм справочника. Подробнее о работе дизайнера экранных форм см. п.6

4.2.5 Печатные формы

Содержит список печатных форм справочника. Подробнее о работе дизайнера печатных форм см. п.7

4.2.6 Исполняемый код

В модуле справочника описываются методы работы с объектом справочника.

Предопределенные поля:

```
public BSRererence Context
Ссылка на объект с данными записи справочника

public Rererence StructureObject
Ссылка на объект структуры

public BSActions Action
Абстрактная прослойка (см. выше)
```

Предопределенные методы:

```
public void OnCreate()
Вызывается при создании объекта справочника

public void OnBeforeSave()
Вызывается перед записью элемента справочника
```

```
public void OnAfterSave()
Вызывается после записи элемента справочника
```

Пример создания объекта Справочник

```
using (var bsRef = new BSReference("Номенклатура"))
{
      string имяЗаписи = "Услуга слесаря";
      var finded = bsRef.FindByName(имяЗаписи); //Поиск записи по имени
      if (!finded) //Если запись не найдена
      {
        bsRef.CreateRecord(имяЗаписи); //Создает запись справочника
      }
      bsRef.SetField("Услуга", true); //Устанавливает для поля Услуга значение true
      bsRef.Save();
}
```

4.2.7 Запросы

Содержит список запросов, заданных для этого справочника. Подробнее о конструктора запроса см. п.8

4.3 Документы

Документ предназначен для проведения операций с массивами ссылочной информации. Физически документ соответствует одной таблице в базе данных и таблицам табличных частей в базе данных.

Документ содержит:

- список полей
- список табличных частей
- список регистров, по которым документ совершает движения
- список экранных форм
- список печатных форм
- список запросов

4.3.1 Свойства

В свойствах документа задаются:

- Идентификатор - строковое имя документа. Все вызовы документа в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление документа.

Виды нумерации – варианты автоматической нумерации документов:

- За весь период – номера новых документов начинаются с «1» и автоматически увеличиваются на «1» на протяжении всего периода ввода документов.

- В пределах года - номера новых документов начинаются с «1» и автоматически увеличиваются на «1» на протяжении текущего года
- В пределах месяца - номера новых документов начинаются с «1» и автоматически увеличиваются на «1» на протяжении текущего месяца

Если программно задается *префикс номера*, то автоматическая нумерация осуществляется с учетом префикса.

Экранные формы:

- Экранная форма записи – указывается экранная форма записи документа
- Экранная форма списка – указывается экранная форма списка документа
- Экранная форма выбора – указывается экранная форма списка выбора документа

Если экранные формы не указаны, то система автоматически сгенерирует экранные формы на основе полей и табличных частей документа.

Документы, создаваемые на основании - содержит перечень документов, которые могу создаваться на основании текущего документа.

4.3.2 Поля

Содержит перечень системных полей и полей, заданных пользователем.

Системные поля:

- Код - поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Удален – булево поле, признак пометки удаления.
- Активен - булево поле, признак активного документа.
- Номер – строковое поле номера записи, автоинкрементно, доступно для изменения, задается автоматически при записи в базу данных.
- Дата – DateTime поле, содержит дату и время документа.
- Ссылка - Document этого же объекта, ссылка на запись документа (п.п. 1.3).
- Основание - Document, ссылка на запись документа, являющегося основанием для этого документа (п.п. 1.3). Заполняется тогда, когда текущий документ был создан на основании другого документа.

4.3.3 Табличные части

Список табличных частей документа задает табличные части, принадлежащие этому документу.

Табличные части содержат список системных полей и полей, заданных пользователем.

Системные поля:

- Код – поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Справочник – Document этого же объекта документа (п.п. 1.3).

4.3.4 Движения

Содержит список выбранных объектов Регистр и объектов История, по которым документ может осуществлять движения. Движения по выбранным регистрам и историям описываются в Исполняемом коде документа.

4.3.5 Экранные формы

Содержит список экранных форм документа. Подробнее о работе дизайнера экранных форм см. п.6

4.3.6 Печатные формы

Содержит список печатных форм документа. Подробнее о работе дизайнера печатных форм см. п.7

4.3.7 Исполняемый код

В модуле документа описываются методы работы с объектом документа и описание формирования движений по регистрам

Предопределенные свойства модуля документа:

```
public BSDocument Context
Ссылка на объект с данными записи документа

public Document StructureObject
Ссылка на объект структуры

public BSActions Action
Абстрактная прослойка (см. выше)
```

Предопределенные методы модуля документа:

```
public void OnCreate()
Вызывается при создании объекта документа

public bool OnActivate()
Вызывается перед активацией документа. Если возвращается False то движения не записываются и документ остается неактивным

public void OnDeActivate()
Вызывается при деактивации документа

public void OnBeforeSave()
Вызывается перед записью документа

public void OnAfterSave()
Вызывается после записи документа

public void OnBaseCreate(BSDocument baseDoc)
Вызывается при создании записи документа на основании другого документа
```

Пример создания объекта Документ

```
using (var bsDoc = new BSDocument("Счет"))
{
        bsDoc.CreateRecord();    //Создание нового документа
        bsDoc.Save(true);        //Запись документа. True активирует документ
}
```

Пример работы с табличной частью документа, добавление строки

```
BSDocumentTable ТЧТаблица = Action.MGetDocumentTable("Товары");
DataRow СтрокаТЧ = ТЧТаблица.CreateNewRow();
СтрокаТЧ[“Идентификатор поля таблицы”] = “Значение”;
ТЧТаблица.AddNewRow(СтрокаТЧ);
```

Пример работы с табличной частью документа, перебор строк

```
//Получение объекта DataTable таб.части
DataTable dt = Action.MGetDocumentDataTable("Товары");
//Перебор строк
foreach(DataRow СтрокаТЧ in dt.Rows)
{
        //получение или установка значния колонки строки таб.части
        var Значение   = СтрокаТЧ["Идентификатор поля таблицы"];
        СтрокаТЧ["Идентификатор поля таблицы"] = "Значение";
}
```

4.3.8 Запросы

Содержит список запросов, заданных для этого справочника. Подробнее о конструктора запроса см. п.8

4.4 Отчеты

Отчеты предназначены для формирования отчетности, создания обработок. Объект отчет не записывается в базу данных.

Отчет содержит:

- список полей
- список экранных форм
- список печатных форм
- список запросов

4.4.1 Свойства

В свойствах отчета задаются:

- Идентификатор - строковое имя отчет. Все вызовы отчета в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление отчета.

Экранные формы:

- Экранная форма записи – указывается экранная форма отчета

Если экранные формы не указаны, то система автоматически сгенерирует экранные формы на основе полей и табличных частей документа.

Использовать стандартный тулбар – установка галочки означает, что при отображении формы отчета будет использоваться стандартные команды отчета.

Группа кнопок *Внешние отчеты*

Сохранить во внешний отчет – сохраняет отчет во внешний файл, который потом можно будет вызывать через меню *Файл – Открыть*.

Загрузить внешний отчет – загружает внешний отчет из файла, полностью заменяя текущий.

4.4.2 Поля

Содержит перечень полей, заданных пользователем. При создании нового отчета автоматически создаются два поля:

- НачДата - DateTime поле.
- КонДата - DateTime поле.

4.4.3 Экранные формы

Содержит список экранных форм отчета. Подробнее о работе дизайнера экранных форм см. п.6

4.4.4 Печатные формы

Содержит список печатных форм отчета. Подробнее о работе дизайнера печатных форм см. п.7

4.4.5 Исполняемый код

В модуле отчета описываются методы работы с объектом отчета.

Предопределенные свойства модуля отчета:

```
public BSDocument Context
Ссылка на объект с данными записи документа

public Document StructureObject
Ссылка на объект структуры

public BSActions Action
Абстрактная прослойка (см. выше)
```

Предопределенные методы модуля документа:

```
public void OnCreate()
Вызывается при создании объекта документа
```

4.4.6 Запросы

Содержит список запросов, заданных для этого справочника. Подробнее о конструктора запроса см. п.8

4.5 Регистры

Регистр предназначен для фиксации движений документов и получения итоговых и промежуточных значений в разрезах.

Регистр содержит:

- список измерений

- список полей

4.5.1 Свойства

В свойствах регистра задаются:

- Идентификатор - строковое имя регистра. Все вызовы регистра в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление регистра.

4.5.2 Измерения

Измерения – это поля, которые позволяют отбирать значения по регистру. Измерения могут быть любого типа, заданного пользователем.

Стандартные измерения:

- Код - поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Операция – поле с типом Number, содержит признак операции движения по регистру.
- Дата - DateTime поле, содержит дату и время движения.
- Активатор - Document поле без указания ссылочного типа, ссылка на запись документа (п.п. 1.3).

4.5.3 Поля

Поля – список полей имеющих только тип Number, к которые записываются значения движения по регистру.

4.6 История

История предназначена для фиксации периодических данных с точностью до секунды. Значения в историю могут быть записаны как интерактивно, так и с помощью движений документов.

История содержит:

- список полей
- список экранных форм

4.6.1 Свойства

В свойствах истории задаются:

- Идентификатор - строковое имя истории. Все вызовы истории в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление истории.

Экранные формы:

- Экранная форма записи – указывается экранная форма записи истории
- Экранная форма списка – указывается экранная форма списка истории
- Экранная форма выбора – указывается экранная форма списка выбора истории

Если экранные формы не указаны, то система автоматически сгенерирует экранные формы на основе полей истории.

Контроль уникальности – поля истории, по которым будет определяться контроль уникальности записываемых данных. Таким образом сли уникальным отмечена дата, то если при записи значения даты совпадет с ранее записанным значением, данные полностью заменяются.

4.6.2 Поля

Список полей, которые позволяют отбирать значения по истории. Поля могут быть любого типа, заданного пользователем.

Стандартные поля:

- Код - поле с типом Number содержит порядковый номер записи, уникально, автоинкрементно.
- Дата - DateTime поле, содержит дату и время движения.
- Активатор - Document поле без указания ссылочного типа, ссылка на запись документа (п.п. 1.3).

4.6.3 Экранные формы

Содержит список экранных форм отчета. Подробнее о работе дизайнера экранных форм см. п.6

4.6.4 Исполняемый код

В модуле отчета описываются методы работы с объектом отчета.

Предопределенные свойства модуля отчета:

```
public BSDocument Context
Ссылка на объект с данными записи документа

public Document StructureObject
Ссылка на объект структуры

public BSActions Action
Абстрактная прослойка (см. выше)
```

Предопределенные методы модуля документа:

```
public void OnCreate()
Вызывается при создании объекта документа
```

4.7 Перечисления

Перечисление – это объект списочных значений, заданных в структуре.

4.7.1 Свойства

В свойствах истории задаются:

- Идентификатор - строковое имя истории. Все вызовы истории в коде возможны только по идентификатору. Идентификатор не может содержать пробелы.
- Наименование – строковое представление истории.

Списочные значения перечисления записываются в текстовом виде и разделяются на отдельные строки.

4.8 Внешние обработки

Создание объекта внешней обработки

```
var exRep = new ExternalReports();
```

Загрузка внешней обработки из файла

```
exRep.LoadReport(ИмяФайла);
```

Создание объекта обработки BSReport (необходим для функционирования внешних обработок)

```
exRep.CreateBSReport();
```

Установка или чтение полей внешней обработки

```
exRep.BSReportObject.SetField("ИмяПоля", Значение);
```

Вызов метода Исполняемого кода внешней обработки

```
exRep.BSReportObject.objectCodeObject.ВызовМетода();
```

Получение и отображение формы внешней обработки

```
var fDialog = exRep.GetFormRecord();
```

Полный пример:

```
var exRep = new ExternalReports();
exRep.LoadReport(ИмяФайла);
exRep.CreateBSReport();
```

```
exRep.BSReportObject.SetField("ИмяПоля", Значение);
exRep.BSReportObject.objectCodeObject.ВызовМетода();
var fDialog = exRep.GetFormRecord();
WindowManagerSys.ShowWindow(fDialog);
```