2023.03.05

Использование DataSnap-технологии на примере разработки комплекса взаимодействующих приложений (ОС Windows и ОС Android) в среде Delphi 10.2 Tokyo. Последовательность действий

В этом документе используются материалы из статьи «***Delphi (C++Builder) Android Mobile Client DataSnap Server***».

Статья размещена здесь:
https://community.embarcadero.com/write-blog-post/entry/delphi-cbuilder-android-mobile-client-datasnap-server-1840

Оглавление

Сокращения ... 3
1 Введение ... 4
1.1 Цель этого документа ... 4
1.2 Соглашения ... 4
2 Обобщенная последовательность действий ... 5
3 Постановка задачи ... 6
4 Решение задачи ... 7
4.1 Требования к разрабатываемом комплексу программ ... 7
4.2 Обобщенный алгоритм игры ... 7
4.3 Методы DataSnap-сервера ... 8
4.3.1 Метод TServerMethods1.LogString ... 9
4.3.2 Метод TServerMethods1.Color_GiveMe ... 9
4.3.3 Метод TServerMethods1.Btn_Presh ... 9
4.4 DataSnap-сервер. Формирование проекта ... 10
4.4.1 Создать новый проект DataSnap-сервера ... 11
4.4.2 Указать основные параметры (характеристики) DataSnap-сервера ... 12
4.4.3 Подготовительные действия ... 18
4.4.4 Сформировать «заглушки» методов DataSnap-сервера ... 20
4.5 DataSnap-клиент. Формирование проекта ... 23
4.5.1 Создать новый проект DataSnap-клиента ... 24
4.5.2 Добавить к мобильному приложению модуль «DataSnap Client» ... 27
4.5.3 Указать основные параметры (характеристики) DataSnap-клиента ... 29
4.5.4 Подготовительные действия ... 32
4.6 Формирование «заявленного» функционала комплекса программ ... 36
4.6.1 Проба (тест) «пустого» комплекса программ ... 36
4.6.2 «Подтягивание» проекта на заданный уровень функционала ... 44

4.7 Тестирование комплекса программ 51
4.7.1 Последовательность действий. 51
5 Наращивание функционала программного комплекса в процессе его эксплуатации 55
5.1 Доработка DataSnap-сервера 55
5.1.1 Последовательность действий 55
5.2 Доработка DataSnap-клиента 57
5.3 Тестирование 59
6 Выводы 60

https://roamer55.ru/

Сокращения

В таблице ниже приведен перечень используемых в документе сокращений.

Таблица 1 − Перечень используемых в документе сокращений

Сокращение	Обозначение
1	2
Android	Операционная система Android
Delphi	Embarcadero Delphi 10.2 Tokyo
GUI	Graphical user interface (графический интерфейс пользователя)
Windows	Операционная система MS Windows
Клиент	DataSnap-клиент
ОС	Операционная система
Сервер	DataSnap-сервер
Телефон	Android-устройство (мобильный телефон, работающий под управлением ОС Android)
Шарик	Графическое изображение (закрашенная красным или синим цветом окружность), формирующееся в компоненте TImage на форме Сервера

1 Введение

DataSnap – технология создания многозвенных приложений.

Материалов об этой технологии опубликовано очень много (в том числе и в Интернет). Повторяться не стОит.

1.1 Цель этого документа

Цель этого документа: предоставить заинтересованным читателям краткую инструкцию (последовательность действий), которую (как вариант) можно использовать для создания комплекса взаимодействующих приложений в среде Delphi 10.2 Tokyo применительно к случаю, когда одно из приложений (DataSnap-сервер) функционирует в ОС Windows, а другое/ие (DataSnap-клиент/ы) – в ОС Android.

Уточнение: упор сделан на то, что информационное взаимодействие приложений (в комплексе программ) должно производиться с использованием Wi-Fi.

1.2 Соглашения

1. «Сколько геологов, столько и мнений» или «на вкус и цвет товарищей нет».
2. Автор не ставит перед собой цель – удивить этот мир «красотой» и оптимальностью исходного кода, а также применяемой методики.
3. Автор в этом документе лишь излагает свой собственный подход при решении (программировании) подобных задач на конкретном, простом примере (с целью – помочь тем, кто в этом нуждается).
4. Предполагается, что читатель знает Delphi и умеет программировать на достаточном, прикладном уровне.
5. Прилагаемые к этому документу исходники программного комплекса хотя и рабочие, но носят чисто иллюстративный характер.

2 Обобщенная последовательность действий

Не зависимо от того, какой функционал планируется реализовать в разрабатываемом комплексе программ, автор использует указанную на рисунке ниже обобщенную последовательность действий.

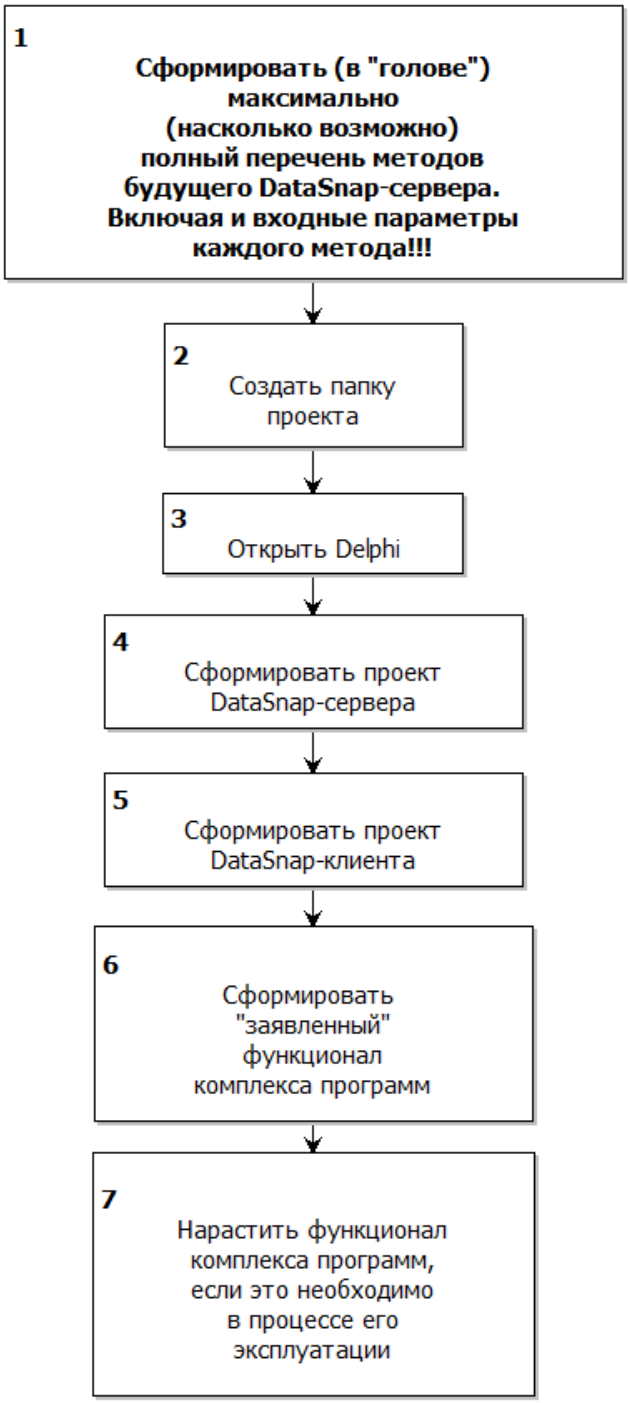

Рисунок 1 – Обобщенная последовательность действий при разработке комплекса программ

Все эти действия будут более подробно рассмотрены в разделах ниже.

3 Постановка задачи

Задача (иллюстрирующий пример):

1. Разработать простую игру «Столкновение» «шаров» на плоскости.
2. Количество игроков: 2.
3. Цель игры: обеспечить столкновение двух «шаров» (т.е., игроки, управляя «шарами», должны столкнуть их друг с другом).
4. Визуализация процесса игры должна быть реализована в программе, функционирующей в среде ОС Window.
5. Управление игрой должно быть реализовано в программе, функционирующей в среде ОС Android.
6. Информационное взаимодействие программ (в комплексе) должно производиться с использованием Wi-Fi.

https://roamer55.ru/

4 Решение задачи

4.1 Требования к разрабатываемом комплексу программ

Исходя из поставленной задачи, можно принять, что:

1. Весь функционал, связанный с визуализацией процесса игры, а также, по взаимодействию игроков, должен быть реализован на «стороне» DataSnap-сервера (ОС Windows).
2. На приложение, функционирующее в среде ОС Android (DataSnap-клиент), возлагается только роль «пульта» управления игрой для каждого игрока.

4.2 Обобщенный алгоритм игры

На рисунке ниже приведен алгоритм работы Сервера.

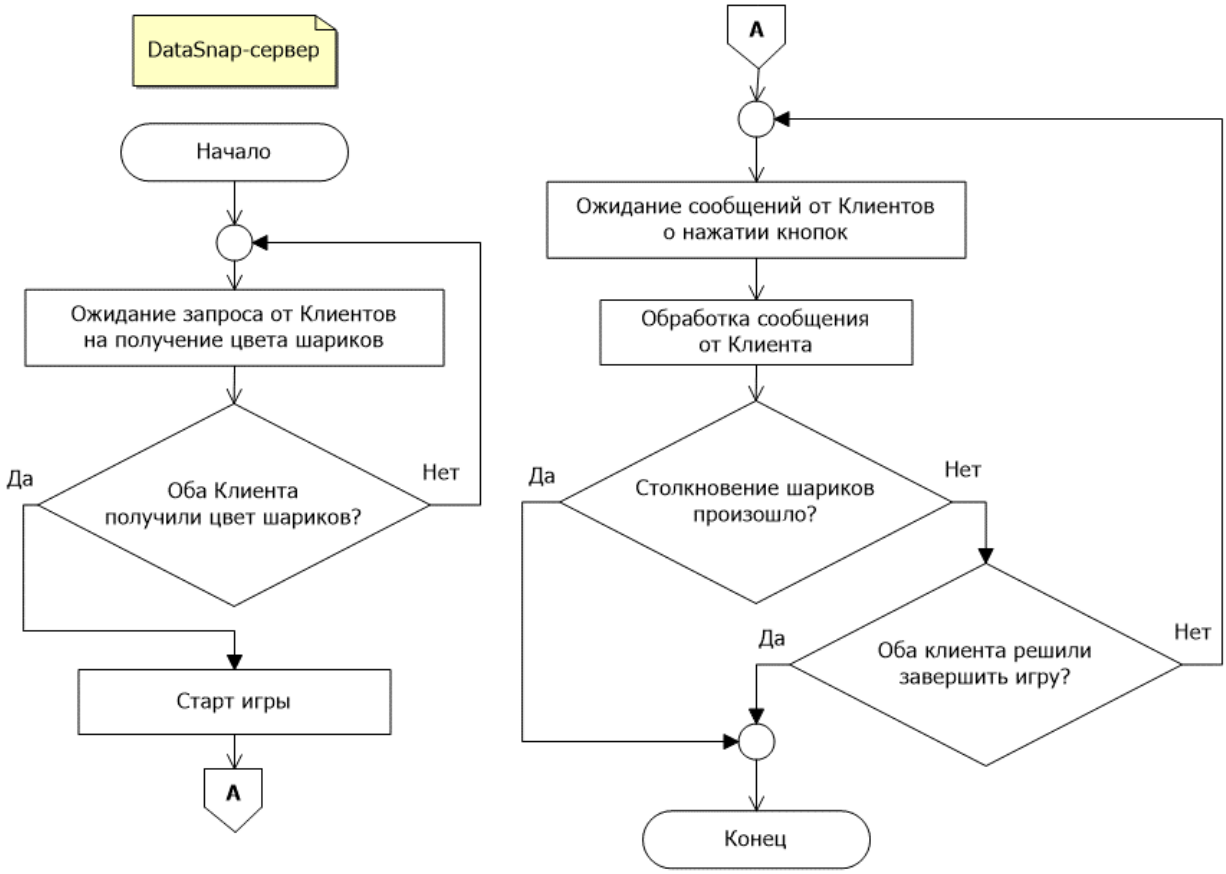

Рисунок 2 – Обобщенный алгоритм работы Сервера

На рисунке ниже приведен алгоритм работы Клиента.

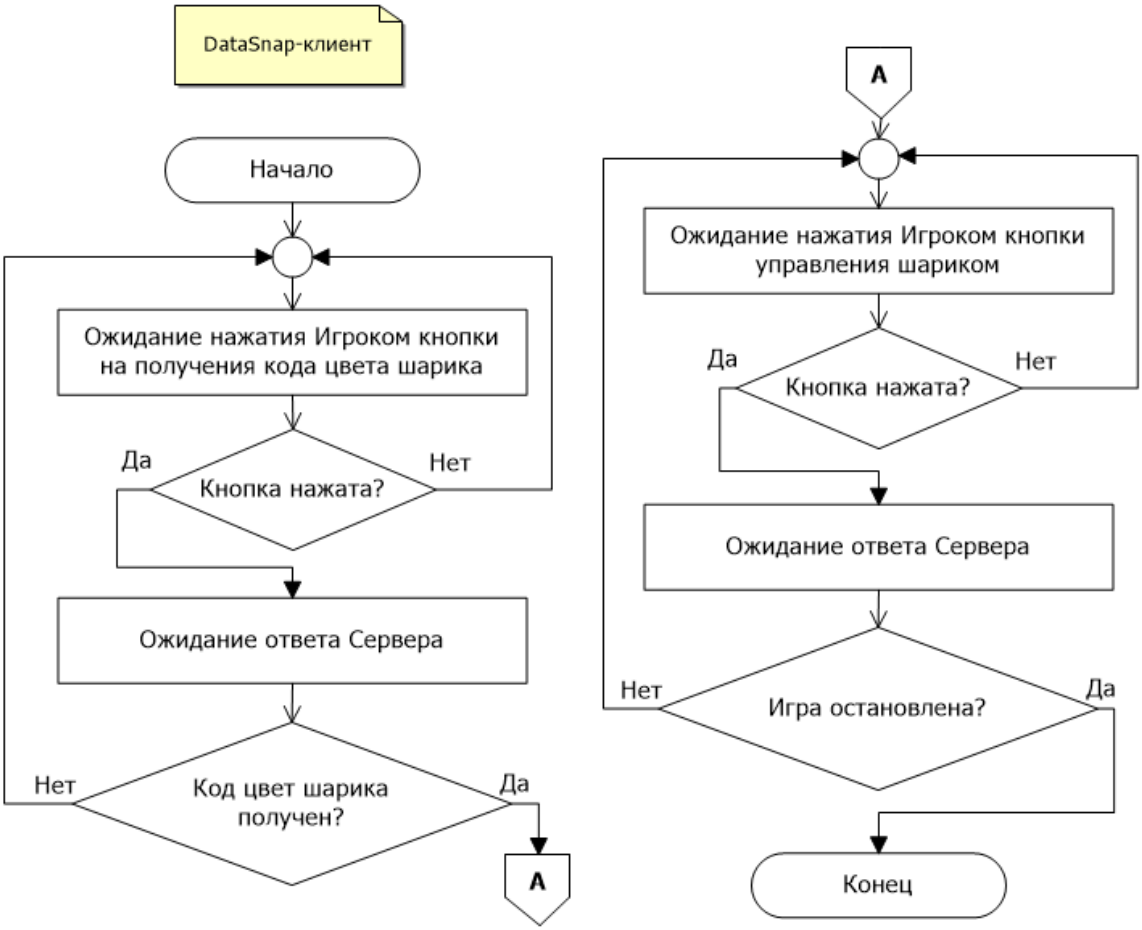

Рисунок 3 – Обобщенный алгоритм работы Клиента

4.3 Методы DataSnap-сервера

1. Delphi (при создании DataSnap-сервера) по умолчанию создает сразу 2 метода, которые для поставленной задачи особого значения не имеют (игнорируем их).
2. Необходимо реализовать 3 новых метода (см. таблицу ниже).

Таблица 2 – Методы DataSnap-сервера

Метод	Назначение
1	2
`LogString`	Добавить принятую от Клиента строку – в LOG Сервера. Этот метод необходим только для отладочных работ
`Color_GiveMe`	Обработка запроса от Клиента на получение цвета шара
`Btn_Presh`	Реакция Сервера на нажатие какой-либо функциональной кнопки на Клиенте

Важно! На этапе создания DataSnap-сервера (см. раздел «4.4 DataSnap-сервер. Формирование проекта») достаточно сформировать лишь «пустые заглушки» методов.

4.3.1 Метод TServerMethods1.LogString

Исходный текст «заглушки» метода приведен ниже.

```
function TServerMethods1.LogString(Value: string): string;
begin
  Result:='';
end;
```

Входные параметры:

`Value` – строка символов (поступившая от Клиента), которые нужно добавить в LOG Сервера.

Возвращаемое значение: всегда пустая строка.

4.3.2 Метод TServerMethods1.Color_GiveMe

Исходный текст «заглушки» метода приведен ниже.

```
function TServerMethods1.Color_GiveMe(sColorNow: string):
string;
begin
  Result:='';
end;
```

Входные параметры:

`sColorNow` – код цвета шарика, который уже назначен Клиенту (если Клиент уже получил код шарика, то запрос игнорируется).

Возвращаемое значение: код цвета шарика или пустая строка в случае отказа Сервера.

4.3.3 Метод TServerMethods1.Btn_Presh

Исходный текст «заглушки» метода приведен ниже.

```
function TServerMethods1.Btn_Presh(sColor:string; sBtnName:
string): string;
begin
  Result:='';
end;
```

Входные параметры:

`sColor` – код цвета шарика, который назначен Клиенту.

`sBtnName` – имя (идентификатор) нажатой игроком на Клиенте кнопки.

Возвращаемое значение: код «стоп-игры» или пустое значение в противном случае.

4.4 DataSnap-сервер. Формирование проекта

На рисунке ниже приведена обобщенная последовательность действий по созданию DataSnap-сервера.

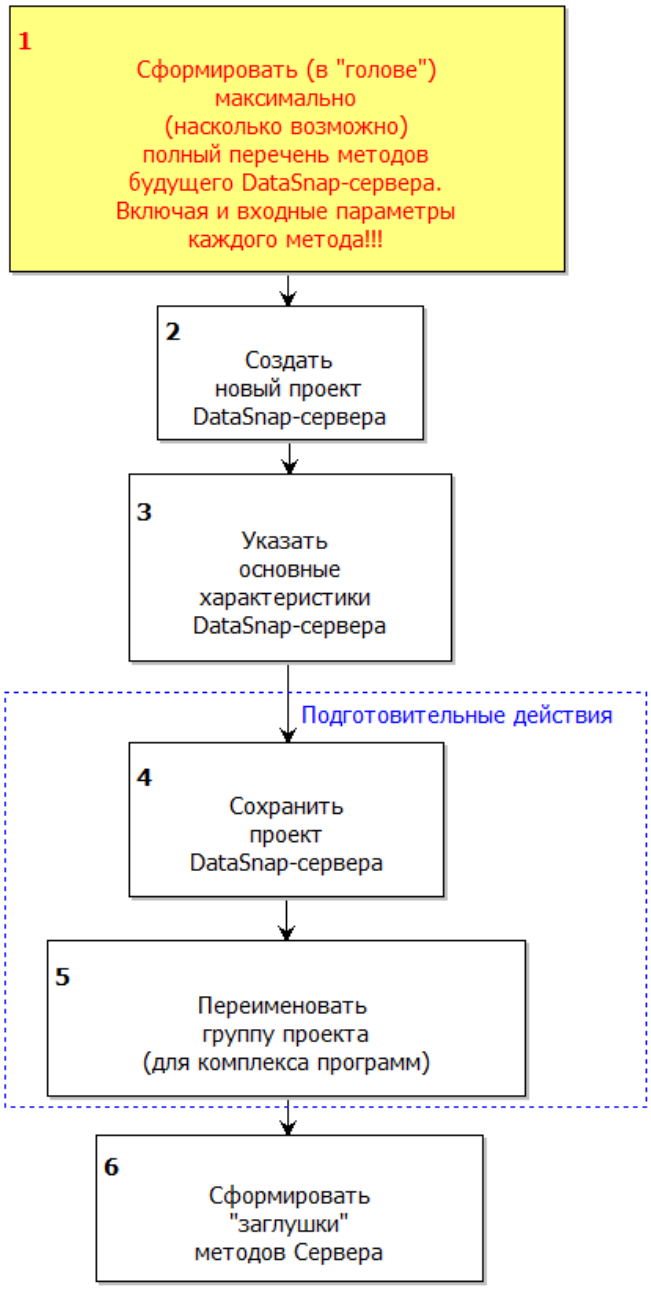

Рисунок 4 – Обобщенная последовательность действий
по созданию DataSnap-сервера

В разделах ниже приведено подробное описАние указанных выше действий (операций).

4.4.1 Создать новый проект DataSnap-сервера

Последовательность операций (действий):

1. Выбрать пункт главного меню приложения «Other…» (см. рисунок 5).

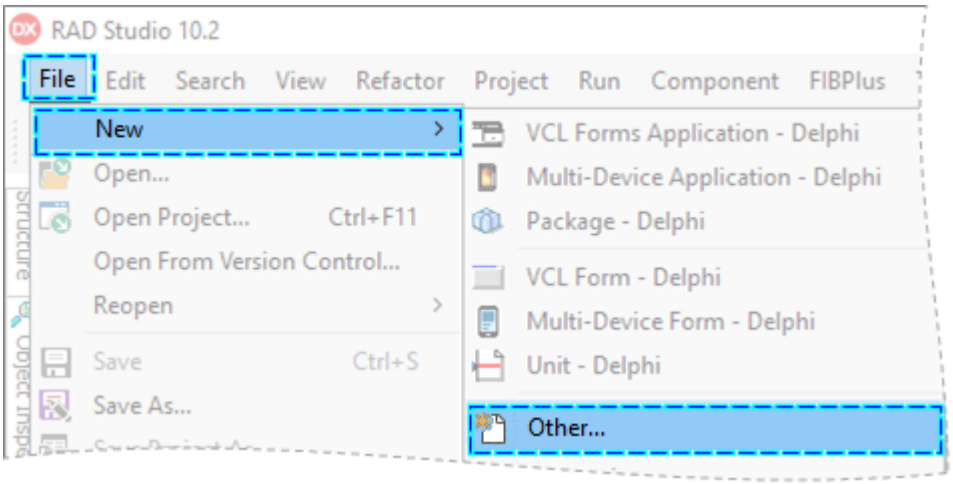

Рисунок 5 – Выбор пункта главного меню Delphi: «Other…»

2. Выбрать тип приложения «DataSnap Server» (см. рисунок 6).

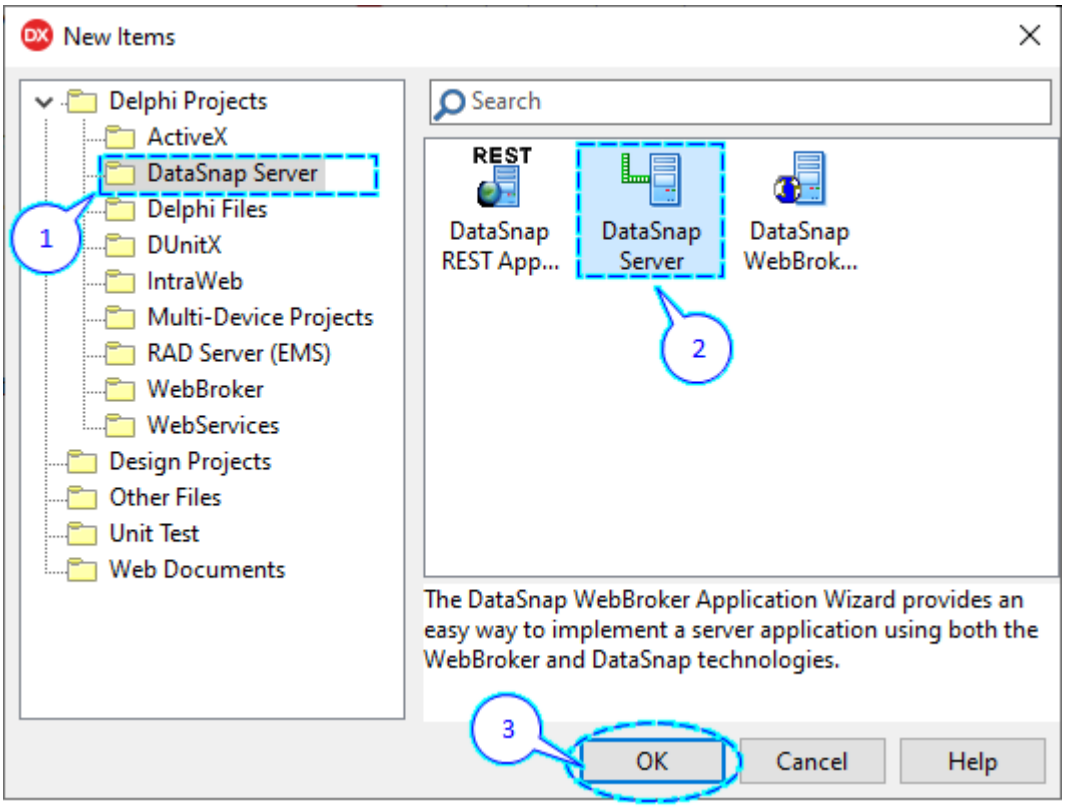

Рисунок 6 – Выбор типа приложения «DataSnap Server»

Далее, Delphi автоматически запустит «мастера» для формирования основных параметров DataSnap-сервера (см. раздел ниже).

4.4.2 Указать основные параметры (характеристики) DataSnap-сервера

Все дальнейшие действия Разработчика четко регламентируется средой Delphi.

На каждом шаге Разработчик должен выбрать (ввести) соответствующие (предлагаемые Delphi) значения параметров.

Для каждого шага (а их всего 6) Delphi выдает отдельную GUI-форму.

4.4.2.1 Шаг-1. Платформа (ОС)

На этом шаге Разработчик изменить ничего не может.

Delphi по умолчанию сразу же предлагает ОС Windows (см. выноску-1 на рисунке ниже).

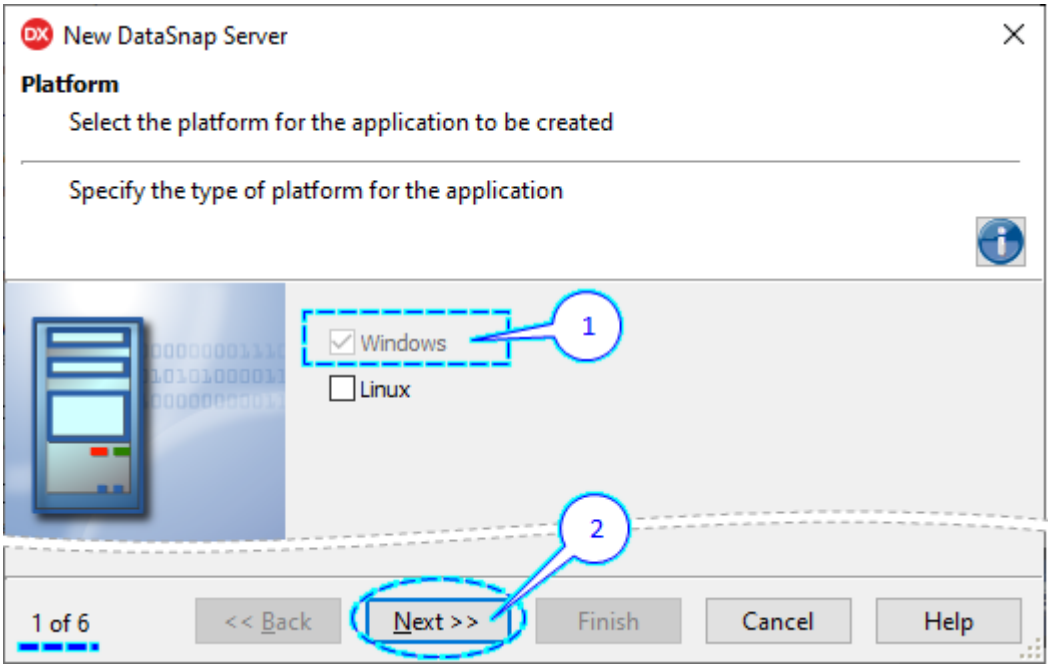

Рисунок 7 – Создание DataSnap-сервера. Шаг-1

Разработчик должен нажать на кнопку «Next» (см. выноску-2 на рисунке выше).

4.4.2.2 Шаг-2. Тип приложения

Здесь следует выбрать «Forms Application» (см. выноску-1 на рисунке ниже) и нажать на кнопку «Next» (выноска-2).

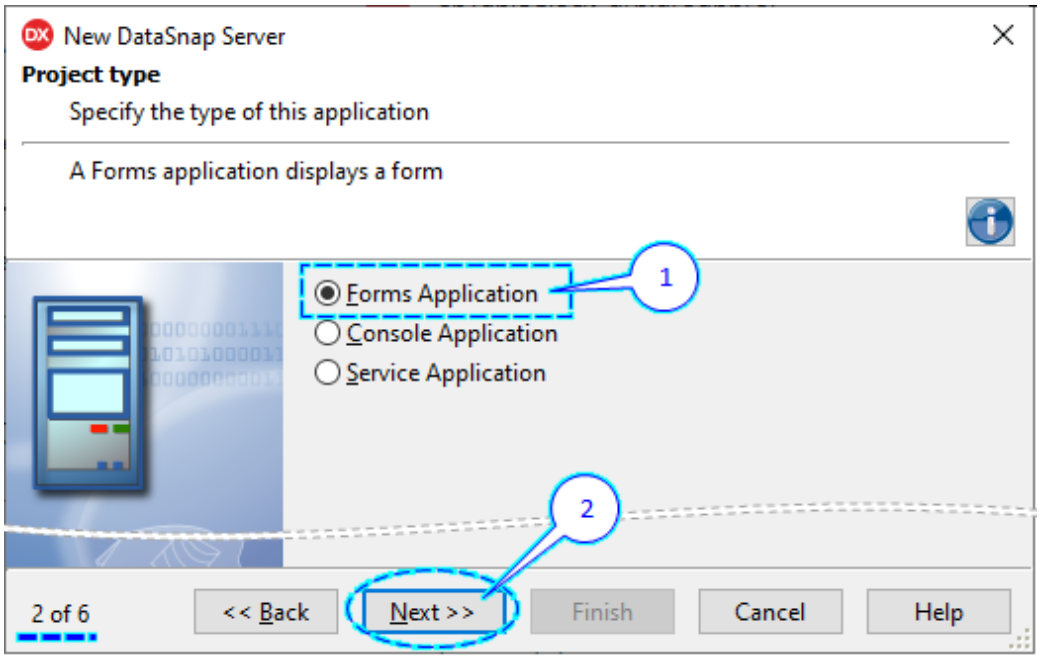

Рисунок 8 – Создание DataSnap-сервера. Шаг-2

Примечание – выбор «Forms Application» связан (в том числе) с тем, что одно из требований к проекту (см. раздел «4.1 Требования к разрабатываемом комплексу программ»): весь функционал, связанный с визуализацией процесса игры, должен быть реализован на «стороне» DataSnap-сервера.

4.4.2.3 Шаг-3. Тип приложения (продолжение)

Здесь следует выбрать «VCL application» (см. выноску-1 на рисунке ниже) и нажать на кнопку «Next» (выноска-2).

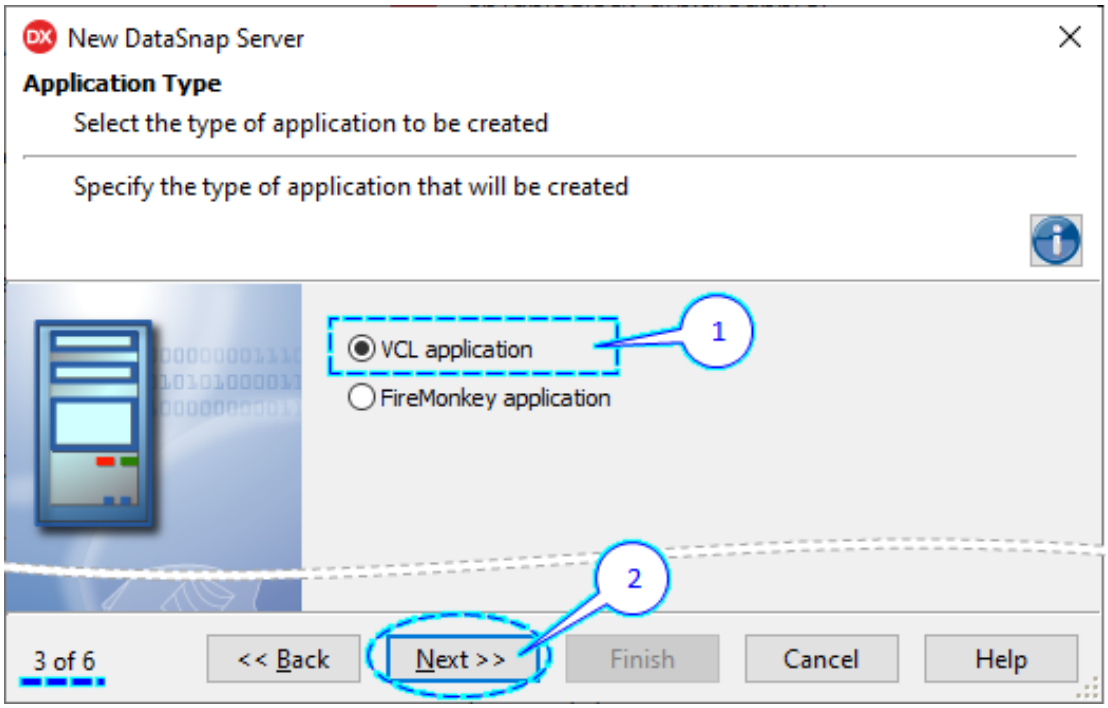

Рисунок 9 – Создание DataSnap-сервера. Шаг-3

4.4.2.4 Шаг-4. Выбор протокола информационного обмена с Клиентом

Здесь следует отметить галочкой протокол «TCP/IP» (см. выноску-1 на рисунке ниже), а также − «Sample Methods» (см. выноску-2 на рисунке ниже).

Далее − нажать на кнопку «Next» (выноска-3).

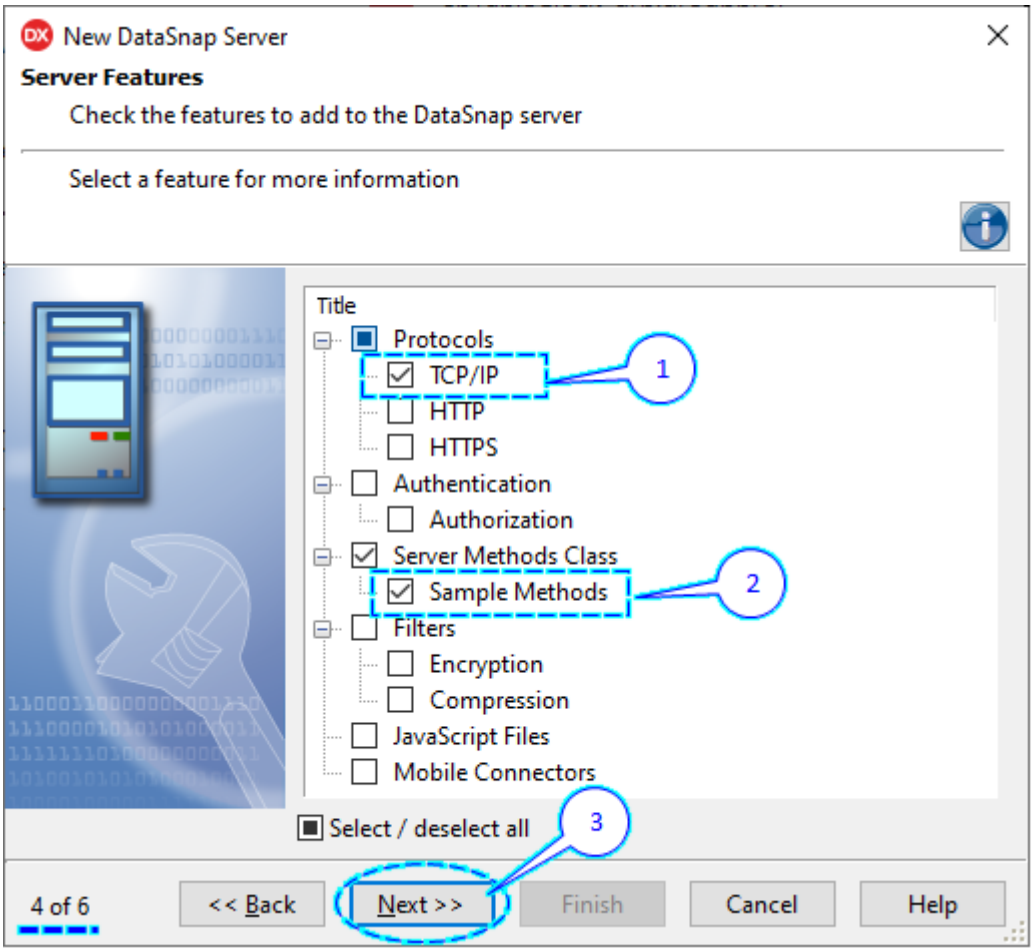

Рисунок 10 − Создание DataSnap-сервера. Шаг-4

4.4.2.5 Шаг-5. Выбор номера порта

Delphi предлагает значение по умолчанию − «211» (см. выноску-1 на рисунке ниже), но можно это значение изменить.

При нажатии на кнопку «Find Open Port» (выноска-3 на рисунке ниже) – будет автоматически найден другой «свободный» номер порта.

После того, как номер порта введен, следует его протестировать (нажав на кнопку «Test Port», см. выноску-2 на рисунке ниже).

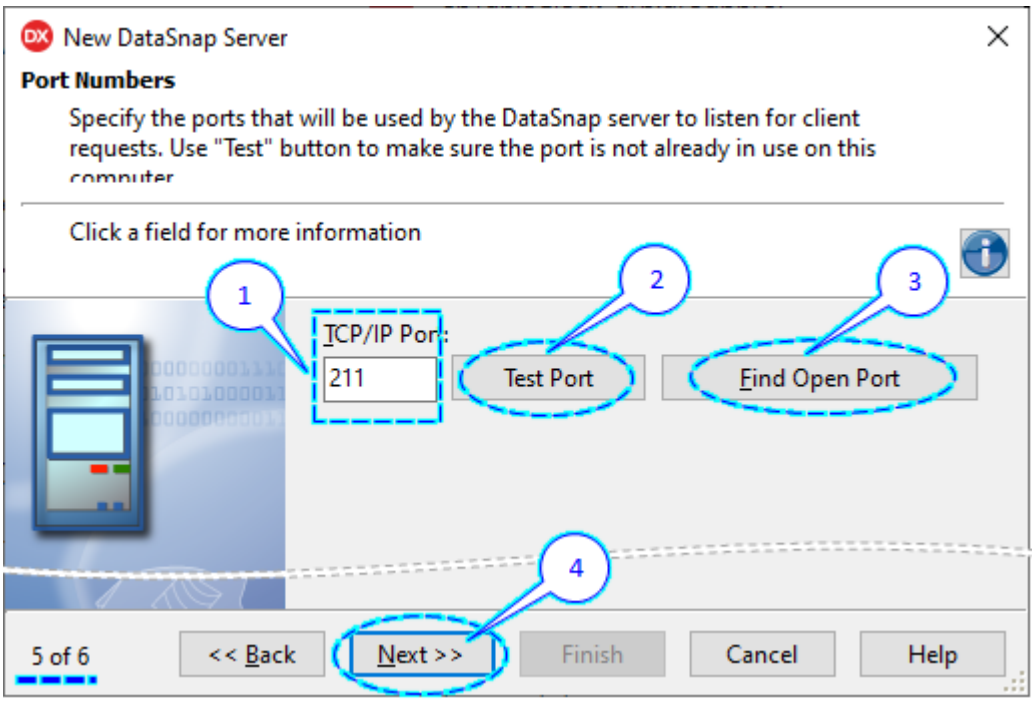

Рисунок 11 − Создание DataSnap-сервера. Шаг-5

Если тест пройден успешно, то на экране появляется соответствующее сообщение (см. рисунок ниже).

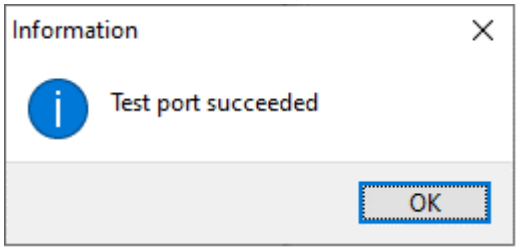

Рисунок 12 – Результат успешного теста доступного порта

Далее, необходимо нажать на кнопку «Next» (выноска-4).

4.4.2.6 Шаг-6. Выбор номера порта

Здесь следует выбрать пункт «TDSServerMosule» (см. выноску-1 на рисунке ниже).

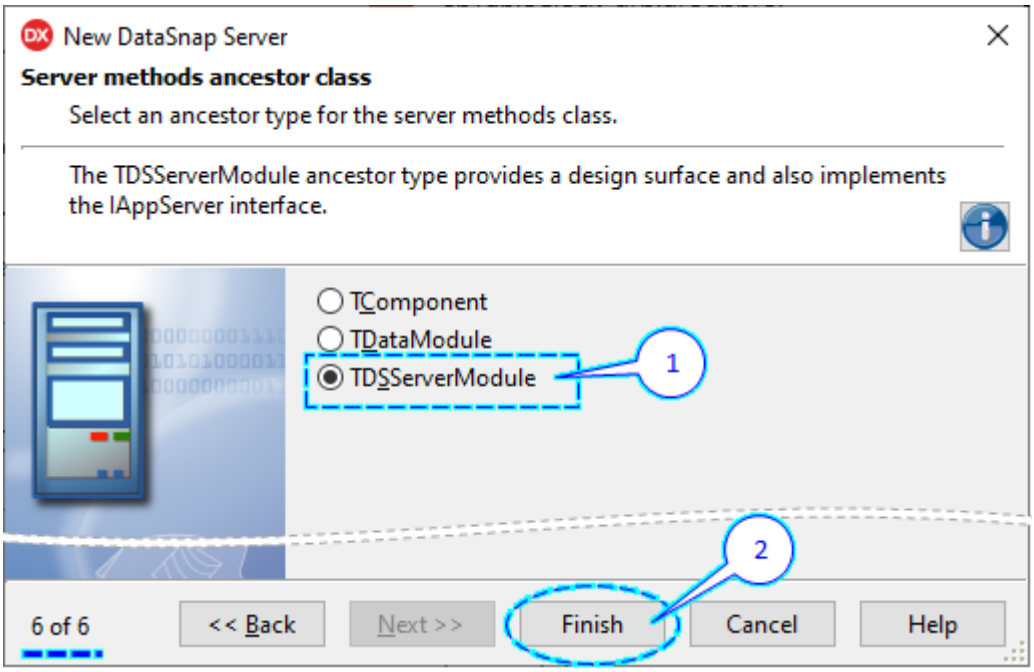

Рисунок 13 − Создание DataSnap-сервера. Шаг-6

Далее, необходимо нажать на кнопку «Finish» (выноска-2).

В результате, Delphi автоматически создаст все необходимые модули DataSnap-сервера (см. рисунок ниже).

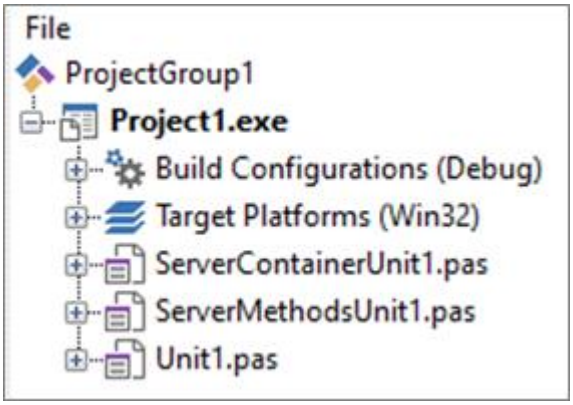

Рисунок 14 – Структура проекта

4.4.3 Подготовительные действия

Далее, необходимо сохранить все созданные Delphi модули.

Автор, в процессе сохранения проекта, задает имена файлов для каждого модуля в соответствии с принятыми (им самим же и для себя самого же) правилами.

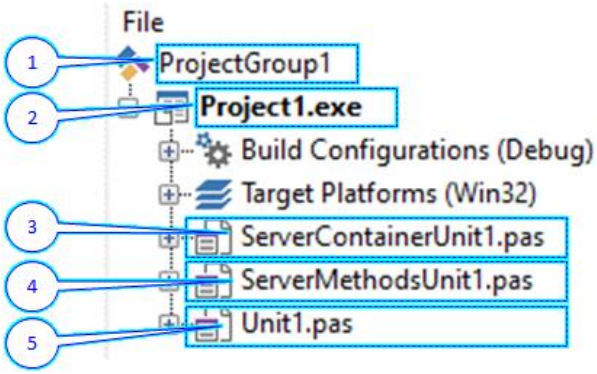

Рисунок 15 – Рекомендуется изменить имена файлов модулей группы проекта в соответствии с какими-то (принятыми) правилами

Выноской-1 на рисунке выше обозначено имя файла группы всего проекта в целом.

Выноской-2 на рисунке выше обозначено имя файла (будущего EXE-модуля) проекта DataSnap-сервера.

Выносками 3−5 обозначены имена файлов специальных модулей DataSnap-сервера.

На рисунке ниже представлен результат сохранения проекта и переименования группы файлов проекта.

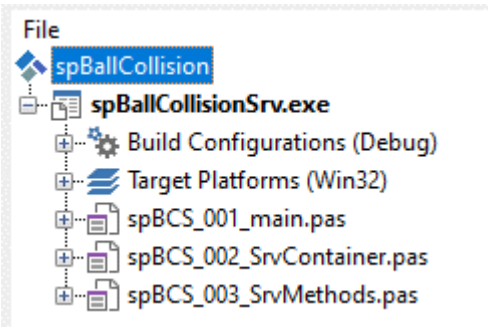

Рисунок 16 − Результат сохранения проекта и переименования группы файлов проекта

В разделах ниже приведено более подробное описАние указанных выше действий.

4.4.3.1 Сохранить Проект

Чтобы сохранить проект, следует выбрать пункт главного меню Delphi «Save Project As» (см. рисунок ниже).

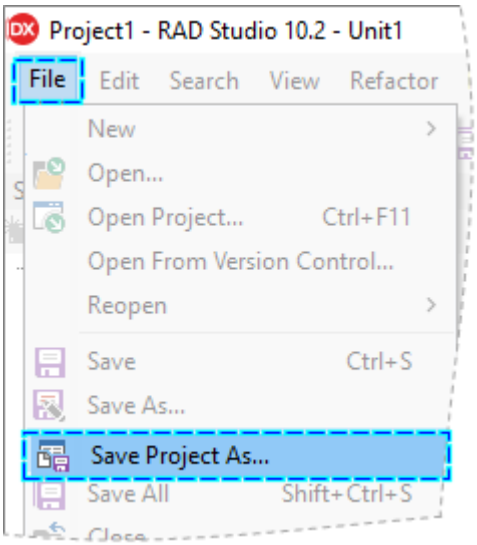

Рисунок 17 – Выбор пункта главного меню Delphi: «Save Project As»

4.4.3.2 Переименовать группу файлов проекта

Чтобы переименовать имя группы файлов Проекта, необходимо выполнить следующие действия (см., также, рисунок ниже):

1. Выбрать группу проекта «**`ProjectGroup`**»;
2. Вызвать (по правой кнопке) всплывающее (контекстное) меню «**`Save Project Group As`**»;
3. Указать соответствующее имя файла для группы.

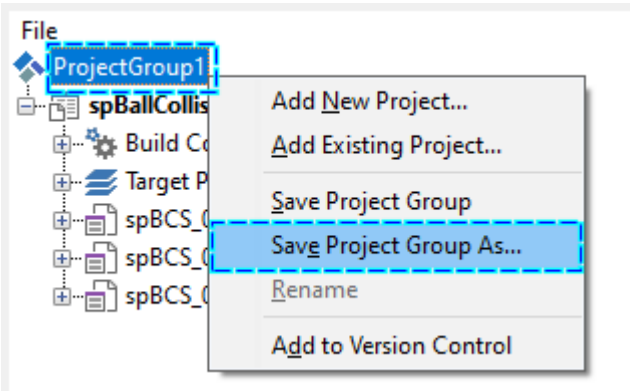

Рисунок 18 – Изменить имя группы файлов проекта

4.4.4 Сформировать «заглушки» методов DataSnap-сервера

На этом шаге необходимо сформировать «пустые заглушки» методов Сервера.

Для этого, следует:

1. Открыть модуль spBCS_003_SrvMethods.pas.
2. Сформировать «заглушки» соответствующих методов Сервера (см. раздел «4.3 Методы DataSnap-сервера»).

Важно! Этот шаг крайне важен, поскольку впоследствии (при создании DataSnap-клиента) это позволит избавить Разработчика от «многих лишних нажатий кнопок на клавиатуре» и, как следствие, потерь времени.

На рисунках ниже приведена соответствующая иллюстрация.

```
unit spBCS_003_SrvMethods;

interface

uses System.SysUtils, System.Classes, System.Json,
    DataSnap.DSProviderDataModuleAdapter,
    Datasnap.DSServer, Datasnap.DSAuth;

type
  TServerMethods1 = class(TDSServerModule)
  private
    { Private declarations }
  public
    { Public declarations }
    //-----------
    //Методы, созданные по умолчанию)
    function EchoString(Value: string): string;
    function ReverseString(Value: string): string;
    //-----------

    //-----------
    //Методы, необходимые для реализации функционала игры
    function LogString(sValue: string): string;
    function Color_GiveMe(sColorNow: string): string;
    function Btn_Presh(sColor:string; sBtnName: string): string;
    //-----------

  end;
```

Рисунок 19 – Методы DataSnap-сервера

```
implementation

{$R *.dfm}

uses System.StrUtils;

//*****************************************
//Методы, созданные по умолчанию)
function TServerMethods1.EchoString(Value: string): string;
begin
  Result := Value;
end;
function TServerMethods1.ReverseString(Value: string): string;
begin
  Result := System.StrUtils.ReverseString(Value);
end;
//*****************************************

//*****************************************
//Методы, необходимые для реализации функционала игры
function TServerMethods1.LogString(sValue: string): string;
begin
  Result:='';
end;
function TServerMethods1.Color_GiveMe(sColorNow: string): string;
begin
  Result:='';
end;
function TServerMethods1.Btn_Presh(sColor:string; sBtnName: string): string;
begin
  Result:='';
end;
//*****************************************

end.
```

Рисунок 20 – Методы DataSnap-сервера (продолжение)

Далее, необходимо выполнить следующие действия:

1. Открыть модуль `spBCS_002_SrvContainer.pas`;
2. Изменить некорректные (после сохранения проекта) ссылки на файл `ServerMethodsUnit1` (см. выноски 1 и 2 на рисунке 21) на корректное значение `spBCS_003_SrvMethods` (см. рисунок 22).

Рисунок 21 – Редактирование ссылки на модуль класса `TServerMethods1`

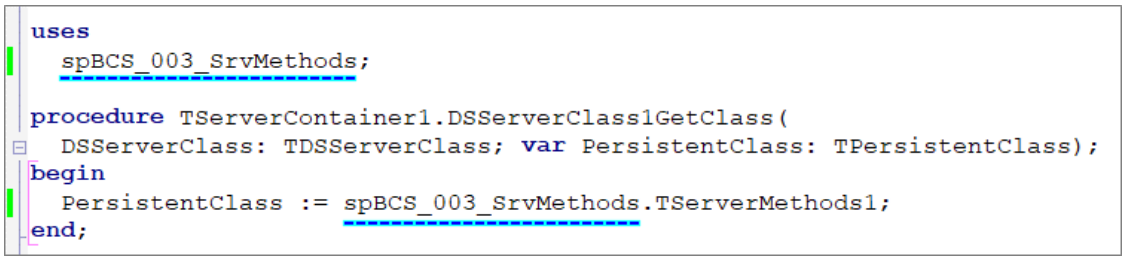

Рисунок 22 – Редактирование ссылки на модуль класса `TServerMethods1` (продолжение)

4.5 DataSnap-клиент. Формирование проекта

Предполагается, что:

1. Среда Delphi запущена и открыта группа файлов проекта `spBallCollision`.
2. Проект `spBallCollisionSrv.dpr` (DataSnap-сервер) скомпилирован.
3. Приложение `spBallCollisionSrv.exe` запущено (стандартными средствами ОС Windows или из среды Delphi при выборе пункта главного меню «**`Run Without Debugging`**», см. рисунок ниже).

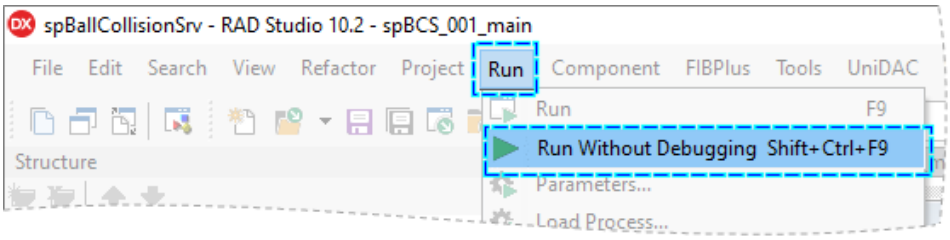

Рисунок 23 – Запуск DataSnap-сервера из среды Delphi

Важно! Приложение **`spBallCollisionSrv.exe`** должно быть запущено перед началом формирования проекта DataSnap-клиента!

На рисунке ниже приведена обобщенная последовательность действий по созданию DataSnap-клиента.

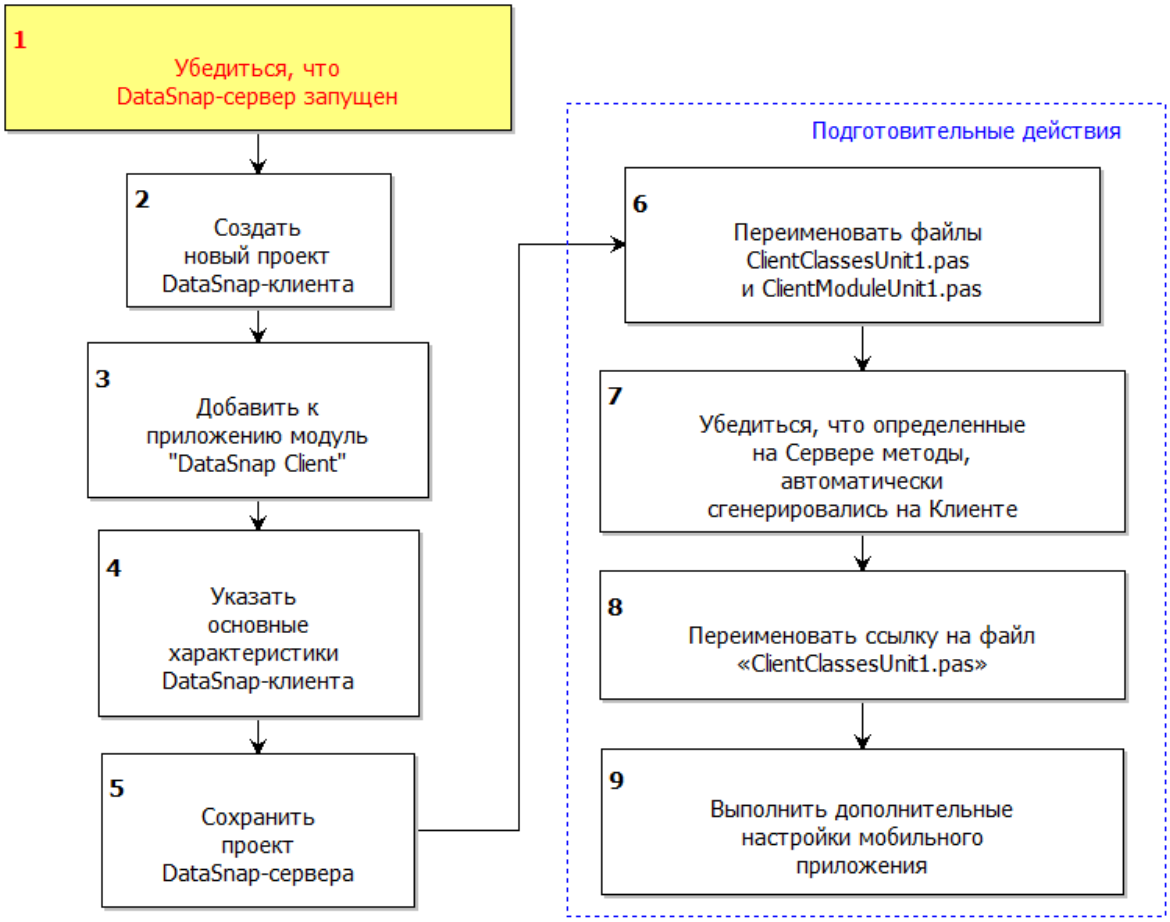

Рисунок 24 – Обобщенная последовательность действий по созданию DataSnap-клиента

4.5.1 Создать новый проект DataSnap-клиента

Последовательность операций (действий):

1. Выбрать группу проекта **`spBallCollision`** (см. выноску-1 на рисунке ниже);

2. Вызвать (по правой кнопке) всплывающее (контекстное) меню «**`Add New Project`**» (выноска-2 на рисунке ниже);

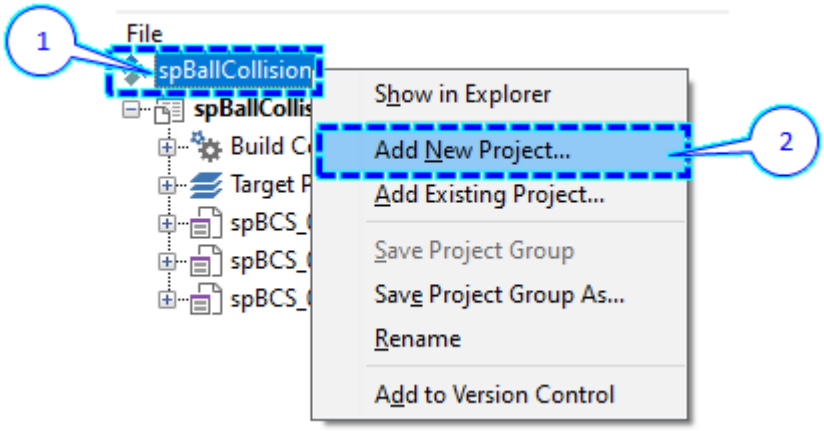

Рисунок 25 – Создание нового проекта DataSnap-Клиент

3. Далее, в группе «**`Multi-Device Projects`**» (см. выноску-1 на рисунке ниже), выбрать «**`Blank Application`**» (см. выноску-2 на рисунке ниже) и нажать на кнопку «**`OK`**» (выноска-3).

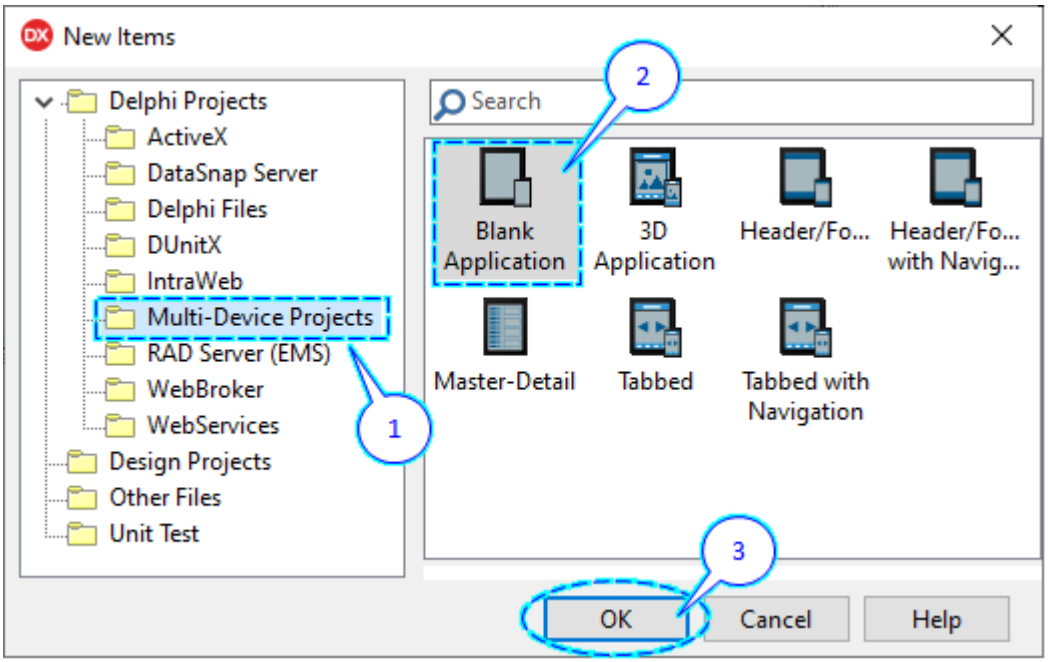

Рисунок 26 – В группе «`Multi-Device Projects`» выбрать «`Blank Application`»

4. Из выпадающего списка выбрать **платформу** «Android» (см. рисунок ниже).

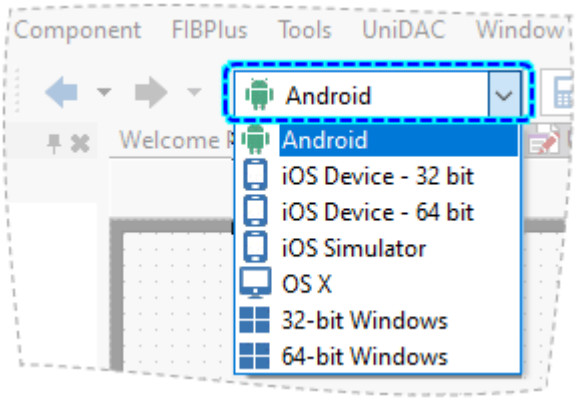

Рисунок 27 – Из выпадающего списка выбрать `платформу` «**Android**»

5. Из выпадающего списка выбрать **стиль** «Android» (см. рисунок ниже).

Рисунок 28 – Из выпадающего списка выбрать `стиль` «**`Android`**»

6. Из выпадающего списка выбрать `View` «**`Master`**» (см. рисунок ниже).

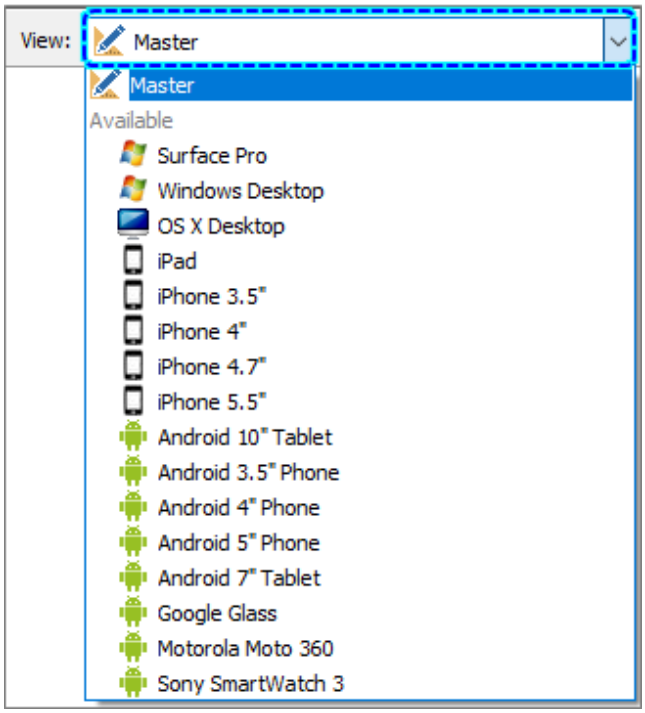

Рисунок 29 – Из выпадающего списка выбрать **`View`** «Master»

На рисунке ниже представлен результат.

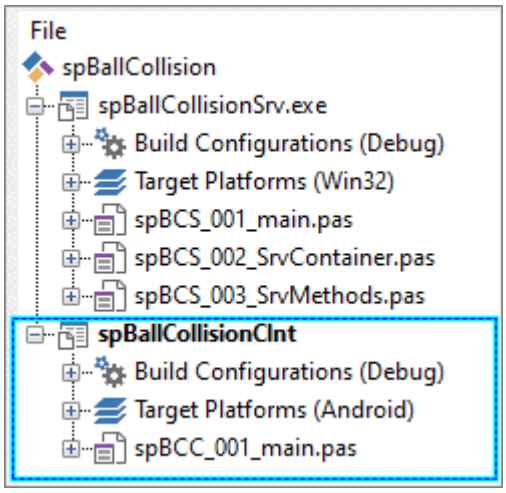

Рисунок 30 – Результат создания нового проекта для DataSnap-клиента

7. Сохранить проект

В данном случае, имя файла проекта: **`spBallCollisionClnt.`**

На этом перечень минимальных действий для создания «пустого» Android-приложения заканчивается.

4.5.2 Добавить к мобильному приложению модуль «DataSnap Client»

1. Следует выбрать проект **`spBallCollisionClnt`** (см. рисунок ниже).

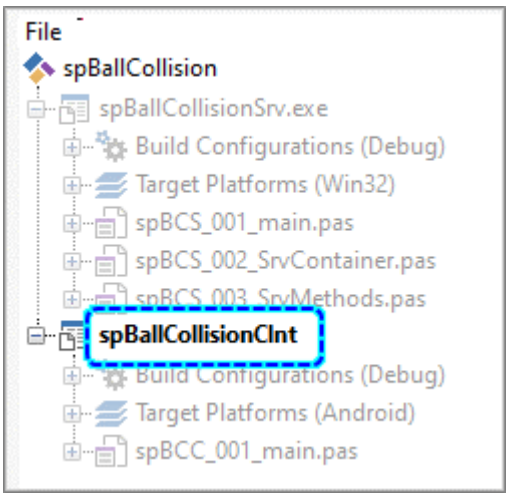

Рисунок 31 – Убедиться, что в группе `spBallCollision` выбран проект **`spBallCollisionClnt`** (DataSnap-клиент)

2. Далее, выбрать пункт главного меню Delphi: «**Other**» (см. рисунок ниже).

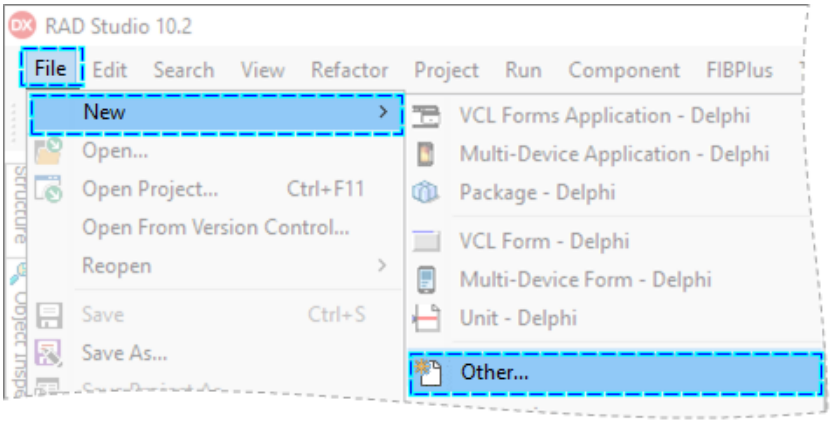

Рисунок 32 – Выбор пункта главного меню Delphi: «Other…»

3. В группе «**DataSnap Server**» (см. выноску-1 на рисунке ниже), выбрать «**DataSnap Client**» (см. выноску-2 на рисунке ниже) и нажать на кнопку **OK** (выноска-3).

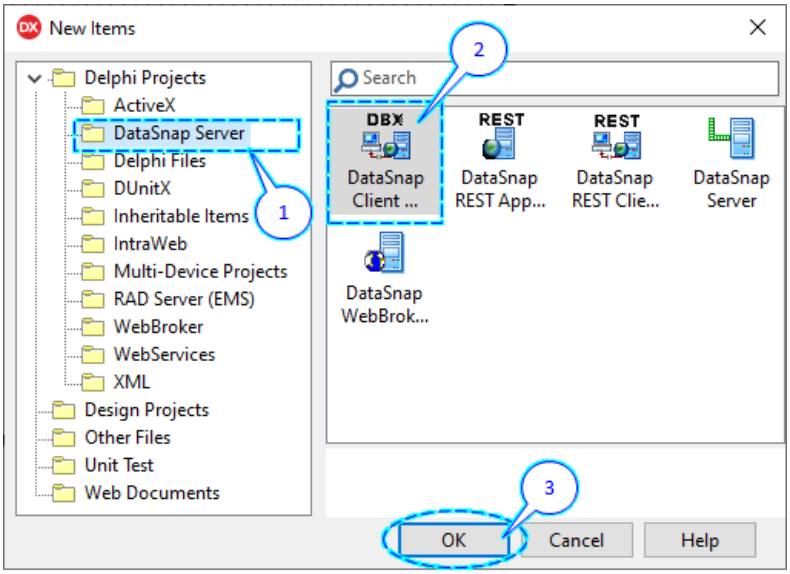

Рисунок 33 – Добавить в мобильное приложение модуль «**DataSnap Client**»

4.5.3 Указать основные параметры (характеристики) DataSnap-клиента

Все дальнейшие действия Разработчика четко регламентируется средой Delphi.

На каждом шаге Разработчик должен выбрать (ввести) соответствующие (предлагаемые Delphi) значения параметров.

Для каждого шага (а их всего 4) Delphi выдает отдельную GUI-форму.

4.5.3.1 Шаг-1. «Местоположение» DataSnap-сервера

Выбрать RadioButton «**`Remote server`**» (см. выноску-1 на рисунке ниже) и нажать на кнопку **`Next`** (выноска-2).

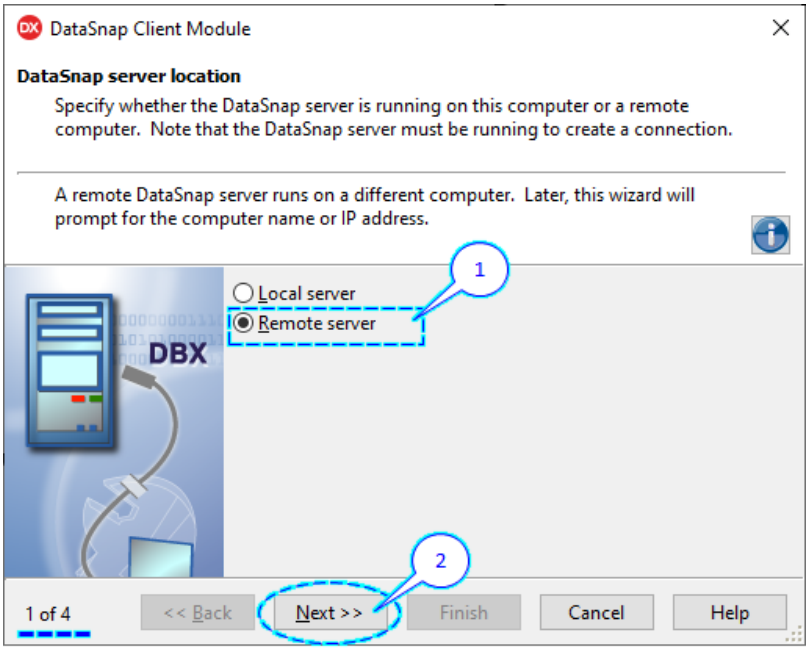

Рисунок 34 – «Местоположение» DataSnap-сервера

4.5.3.2 Шаг-2. Тип DataSnap-сервера

Выбрать «**`DataSnap stand alone server`**» (см. выноску-1 на рисунке ниже) и нажать на кнопку **`Next`** (выноска-2).

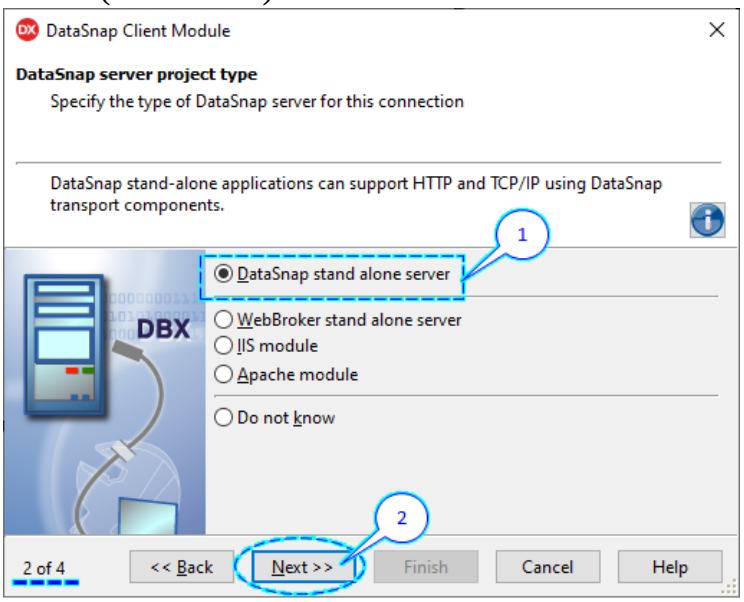

Рисунок 35 – Тип DataSnap-сервера

4.5.3.3 Шаг-3. Протокол информационного обмена с DataSnap-сервером

Выбрать «**TCP/IP**» (см. выноску-1 на рисунке ниже) и нажать на кнопку **Next** (выноска-2).

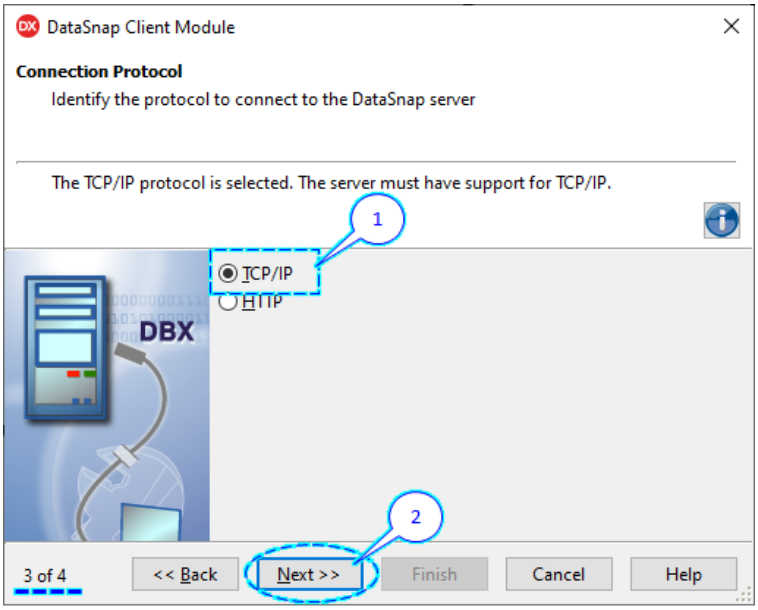

Рисунок 36 – Протокол информационного обмена с DataSnap-сервером

4.5.3.4 Шаг-4. Адрес и порт соединения с DataSnap-сервером

Важно! Предполагается, что DataSnap-сервер запущен!

1. Указать **корректное** значение параметров **Host** (см. выноску-1 на рисунке ниже) и **Port** (см. выноску-2).

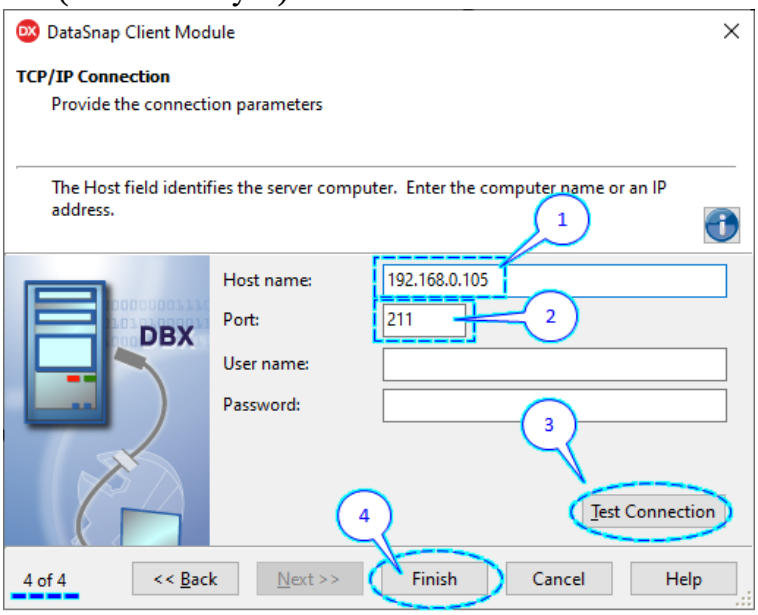

Рисунок 37 – Адрес и порт DataSnap-сервера

Важно! Для того, чтобы узнать IP-адрес Сервера, можно использовать, например:

A. Рекомендации, данные в статье «*Delphi (C++Builder) Android Mobile Client DataSnap Server*».
B. Программу **spMyIP.exe**, см. здесь:
https://roamer55.ru/my_download/progs/os_win/spMyIP/.

2. Нажать на кнопку «**`Test Connection`**» (см. выноску-3 на рисунке выше).

При успешном соединении с Сервером, на экран будет выведено сообщение (см. рисунок ниже).

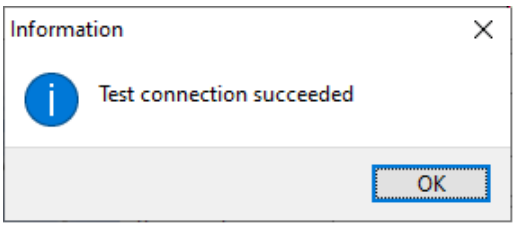

Рисунок 38 – Успешное соединение с DataSnap-Сервером

3. Нажать на кнопку **`Finish`** (выноска-4 на рисунке 37).

Результат представлен на рисунке ниже.

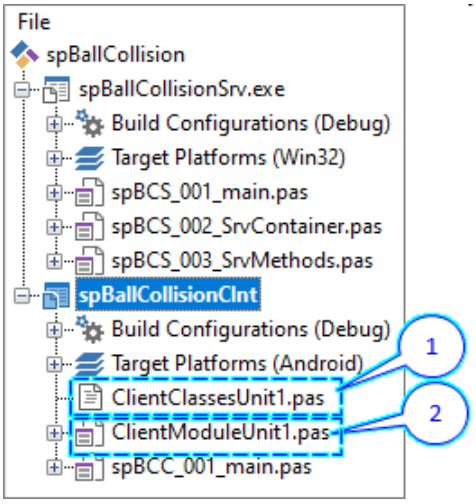

Рисунок 39 – Рекомендуется изменить имена файлов модулей группы проекта в соответствии с какими-то (принятыми) правилами

Выносками 1 и 2 на рисунке выше обозначены файлы группы проекта, которые рекомендуется переименовать.

4.5.4 Подготовительные действия

4.5.4.1 Переименовать файлы ClientClassesUnit1.pas и ClientModuleUnit1.pas

Для переименования файла (отмеченного, например, выноской-1 на рисунке ниже) можно воспользоваться пунктом всплывающего меню «**Rename**» (выноска-2).

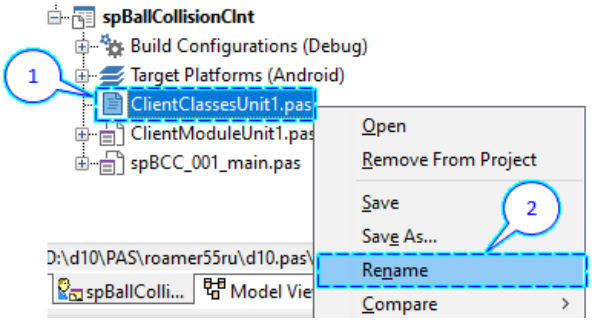

Рисунок 40 – Переименование модулей

Примечание – вообще-то, Автор предпочитает использовать банальное «**SaveAs**» для каждого из этих модулей…

Результат представлен на рисунке ниже.

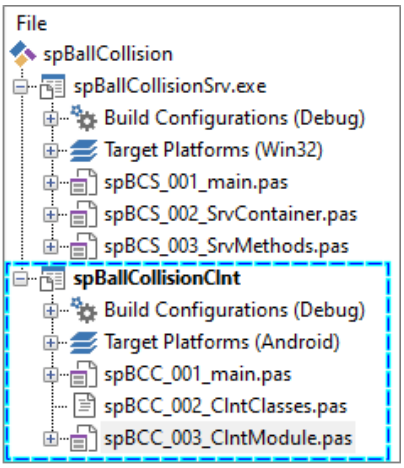

Рисунок 41 – Результат переименования модулей

4.5.4.2 Убедиться, что определенные на Сервере методы, автоматически сгенерировались на Клиенте

Необходимо открыть модуль **spBCC_002_ClntClasses** и убедиться, что методы Сервера автоматически и корректно перенесены Клиенту (см. рисунок ниже).

```
TServerMethods1Client                    TServerMethods1Client.EchoString
//
// Created by the DataSnap proxy generator.
// 20.02.2023 00:09:04
//

unit spBCC_002_ClntClasses;

interface

uses System.JSON, Data.DBXCommon, Data.DBXClient, Data.DBXDataSnap,
     Data.DBXJSON, Datasnap.DSProxy, System.Classes, System.SysUtils,
     Data.DB, Data.SqlExpr, Data.DBXDBReaders, Data.DBXCDSReaders,
     Data.DBXJSONReflect;

type
  TServerMethods1Client = class(TDSAdminClient)
  private
    FEchoStringCommand: TDBXCommand;
    FReverseStringCommand: TDBXCommand;
    FLogStringCommand: TDBXCommand;
    FColor_GiveMeCommand: TDBXCommand;
    FBtn_PreshCommand: TDBXCommand;
  public
    constructor Create(ADBXConnection: TDBXConnection); overload;
    constructor Create(ADBXConnection: TDBXConnection; AInstanceOwner: Boolean); overload;
    destructor Destroy; override;
    function EchoString(Value: string): string;
    function ReverseString(Value: string): string;
    function LogString(sValue: string): string;
    function Color_GiveMe(sColorNow: string): string;
    function Btn_Presh(sColor: string; sBtnName: string): string;
  end;

implementation
```

Рисунок 42 – Модуль **spBCC_002_ClntClasses** DataSnap-клиента

4.5.4.3 Переименовать ссылку на файл «ClientClassesUnit1.pas»

Далее, необходимо открыть модуль spBCC_003_ClntModule и заменить ссылку «**ClientClassesUnit1**» на «**spBCC_002_ClntClasses**» (см. рисунок ниже).

```
unit spBCC_003_ClntModule;

interface

uses
  System.SysUtils, System.Classes, ClientClassesUnit1, Data.DBXDataSnap,
  Data.DBXCommon, IPPeerClient, Data.DB, Data.SqlExpr;

type
  TClientModule1 = class(TDataModule)
    SQLConnection1: TSQLConnection;
  private
    FInstanceOwner: Boolean;
```

ЗАМЕНИТЬ → spBCC_002_ClntClasses

Рисунок 43 – Переименование ссылки на модуль **ClientCkassesUnit1**

4.5.4.4 Выполнить дополнительные настройки мобильного приложения

Далее, необходимо выполнить дополнительные настройки мобильного приложения. Для этого, необходимо выбрать пункт главного меню Delphi:

«**Project**» –> «**Options**»

Важно! Нижеуказанные настройки определяются спецификой разрабатываемого Android-приложения и поэтому, ест-нно, могут отличаться от приведенных в этом документе.

4.5.4.4.1 Раздел «User Permissions»

Разрешение мобильного приложения на доступ к ресурсам ОС Android

1. Открыть раздел «User Permissions» (см. рисунок ниже).

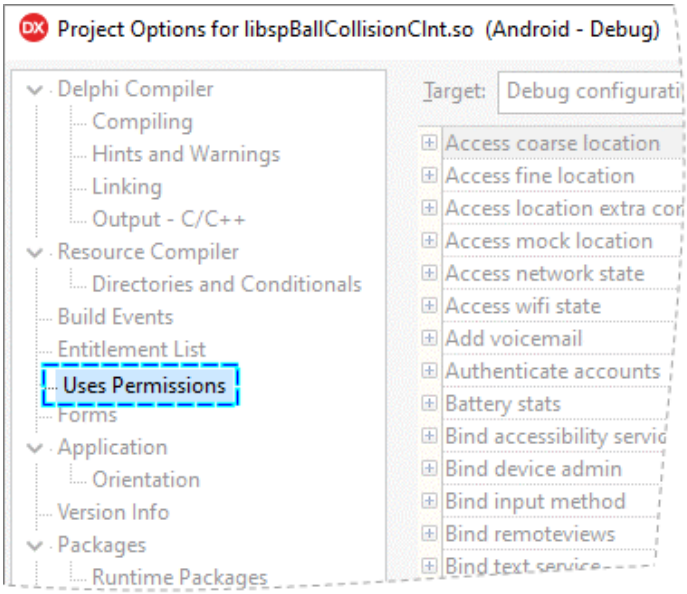

Рисунок 44 – Раздел «User Permissions»

2. Из всего множества доступных «разрешений» оставить только 3, указанные на рисунке ниже.

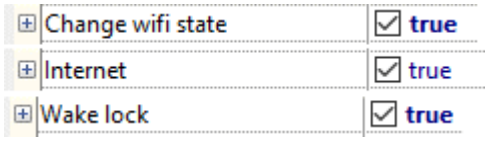

Рисунок 45 – «Разрешения» Android-приложения

4.5.4.4.2 Раздел «Orientation»

Должно быть так, как указано на рисунке ниже.

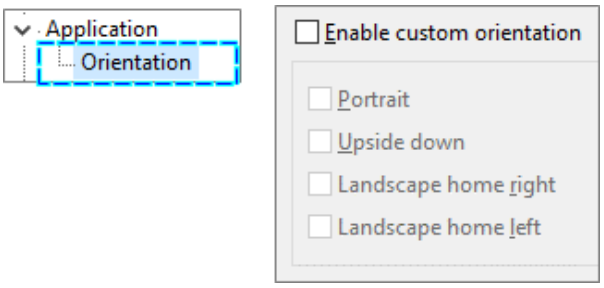

Рисунок 46 − Раздел «Orientation»

4.5.4.4.3 Раздел «Version Info»

По умолчанию, наименование пакета содержит подстроку «**`embarcadero`**» для всех создаваемых (новых) приложений.

Лучше это сразу изменить (см., например, рисунок ниже).

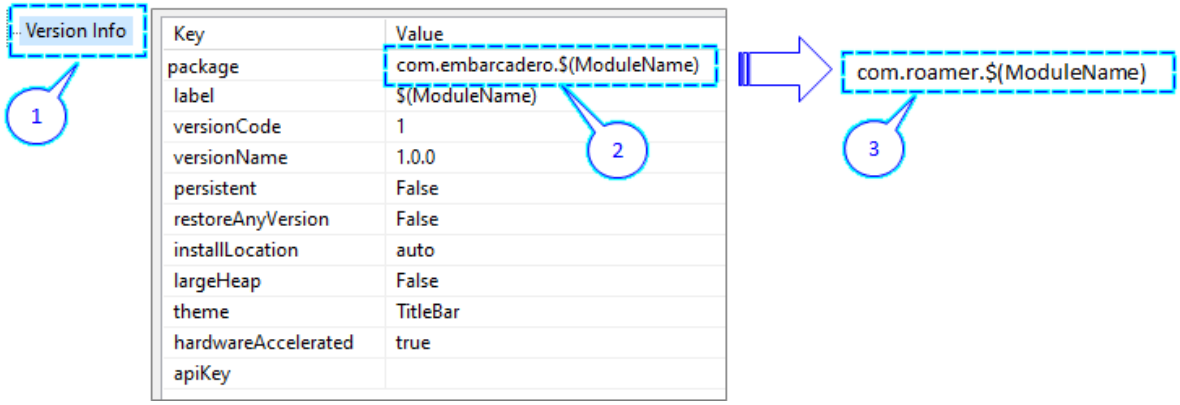

Рисунок 47 − Раздел «Version Info»

4.6 Формирование «заявленного» функционала комплекса программ

В этом разделе документа приведен процесс формирования функционала комплекса программ применительно к рассматриваемой (игровой) задаче.

Ясное дело, что в других проектах это будет совершенно по другому.

4.6.1 Проба (тест) «пустого» комплекса программ

В силу каких-то (непонятных Автору причин) – далеко не всегда даже простое сформированное Android-приложение будет работать на Android-устройстве.

«О вкусах не спорят», конечно…

Но Автор не упускает возможности тестировать Android-приложение при добавлении каждого нового функционала или изменения GUI (сохраняя, при этом, рабочие варианты в архиве).

И в данном случае рекомендуется убедиться, что «почти пустое» приложение работоспособно.

4.6.1.1 DataSnap-сервер

Выполнить приведенные ниже действия.

1. Открыть модуль **`spBCS_001_main`** и переименовать главную форму приложения на **`fMain`**.
2. Для формы fMain выбрать значение свойства **`Position`**: `poScreenCenter`.
3. Разместить на форме следующие компоненты:
   Panel_Top: TPanel (align=alTop);
   Panel_Bottom: TPanel (align=alBottom);
   pcMain: TPageControl (align=alClient).

См., также, рисунок ниже.

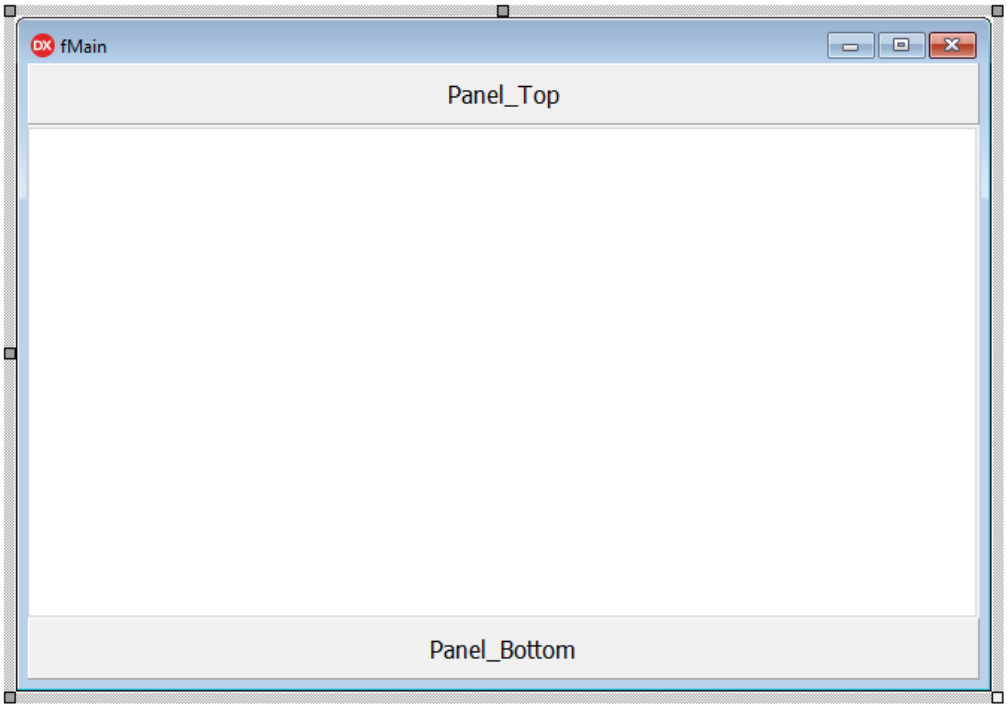

Рисунок 48 – Главная форма DataSnap-сервера

4. Для компонента **pcMain** добавить 2 вкладки:
   tsGame (caption: Игра);
   tsLog (caption: LOG).

5. На вкладке **tsLog** разместить компонент: **MemoLog**: TMemo.

См., также, рисунок ниже.

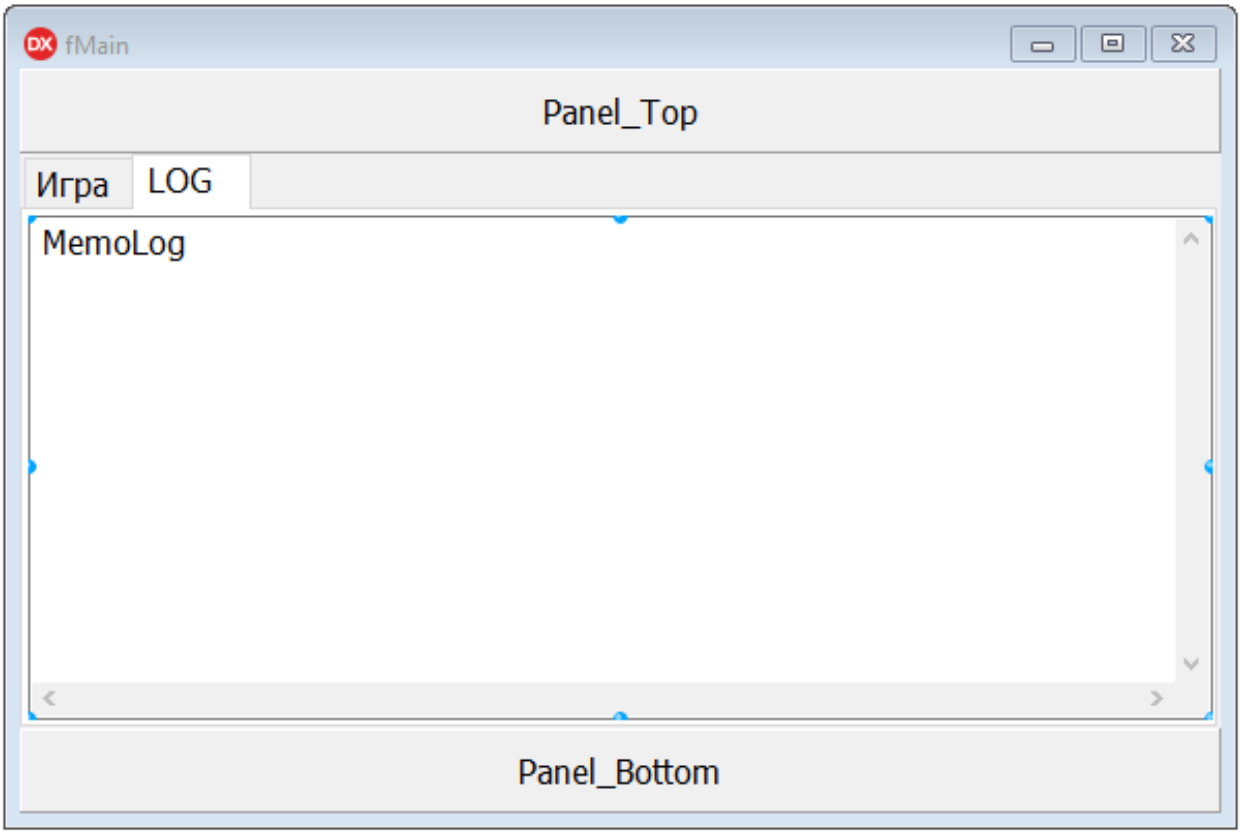

Рисунок 49 – Вкладки компонента **pcMain**

6. Добавить метод формы (fMain): AddToLog (см. рисунок ниже).

```
type
  TfMain = class(TForm)
    pcMain: TPageControl;
    Panel_Bottom: TPanel;
    Panel_Top: TPanel;
    tsGame: TTabSheet;
    tsLog: TTabSheet;
    MemoLog: TMemo;
  private
    { Private declarations }
  public
    { Public declarations }

    //Добавить сообщение в LOG
    function AddToLog(sMsg:string):integer;
  end;

var
  fMain: TfMain;

implementation

{$R *.dfm}

function TfMain.AddToLog(sMsg:string):integer;
//Добавить сообщение в LOG
begin
  Result:=-1;
  sMsg:=trim(sMsg);
  if sMsg<>'' then begin
     MemoLog.Lines.Add(TimeToStr(NOW)+'. '+sMsg);
     Result:=MemoLog.Lines.Count;
  end;
end;

end.
```

Рисунок 50 – Метод **AddToLog** формы fMain

7. Открыть модуль **spBCS_003_SrvMethods**.

8. Добавить в раздел **Uses** ссылку на модуль **spBCS_001_main** (см. рисунок ниже).

```
implementation

{$R *.dfm}

uses
    spBCS_001_main,
    System.StrUtils;
```

Рисунок 51 – Ссылка на модуль **spBCS_001_main**

9. Доработать «заглушку» метода **LogString** (см. рисунок ниже).

```
function TServerMethods1.LogString(sValue: string): string;
//Добавить в LOG сообщение от DataSnap-клиента
begin
  Result:='Err';
  sValue:=trim(sValue);
  if sValue<>'' then begin
     fMain.AddToLog(sValue);
     Result:='';
  end;
end;
```

Рисунок 52 – Метод **LogString** класса `TServerMethods1`

10. Скомпилировать и запустить DataSnap-сервер (стандартными средствами ОС Windows или из среды Delphi при выборе пункта главного меню «**Run Without Debugging**» (см. рисунок 23)

4.6.1.2 DataSnap-клиент

1. Выбрать проект **spBallCollisionClnt** (см. рисунок ниже).

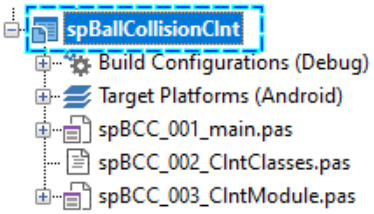

Рисунок 53 – Выбор проекта **spBallCollisionClnt**

2. Открыть модуль **spBCC_001_main**.

3. Добавить в раздел **Uses** ссылку на модуль **spBCC_003_ClntModule** (см. рисунок ниже).

```
unit spBCC_001_main;

interface

uses
  spBCC_003_ClntModule,
  System.SysUtils, System.Types, System.UITypes, System.Classes, System.Variants,
  FMX.Types, FMX.Controls, FMX.Forms, FMX.Graphics, FMX.Dialogs,
  FMX.Controls.Presentation, FMX.StdCtrls;
```

Рисунок 54 – Ссылка на модуль **spBCC_003_ClntModule**

4. Разместить на форме кнопку:
   btnToLog: TButton;

5. Сформировать обработчик «нажатия» для этой кнопки (см. рисунок ниже).

```
procedure TForm2.btnToLogClick(Sender: TObject);
begin
  ClientModule1.ServerMethods1Client.LogString((Sender as TButton).Name);
end;
```

Рисунок 55 – Обработчик «нажатия» на кнопку **btnToLog**

6. Выбрать пункт главного меню Delphi (и дождаться окончание процесса):

«Project» –> «Build spBallCollisionClnt»

7. Далее, выбрать пункт главного меню Delphi (и дождаться окончание процесса):

«Project» –> «Deploy spBallCollisionClnt.so»

8. В проводнике открыть папку **bin** (см. рисунок ниже).

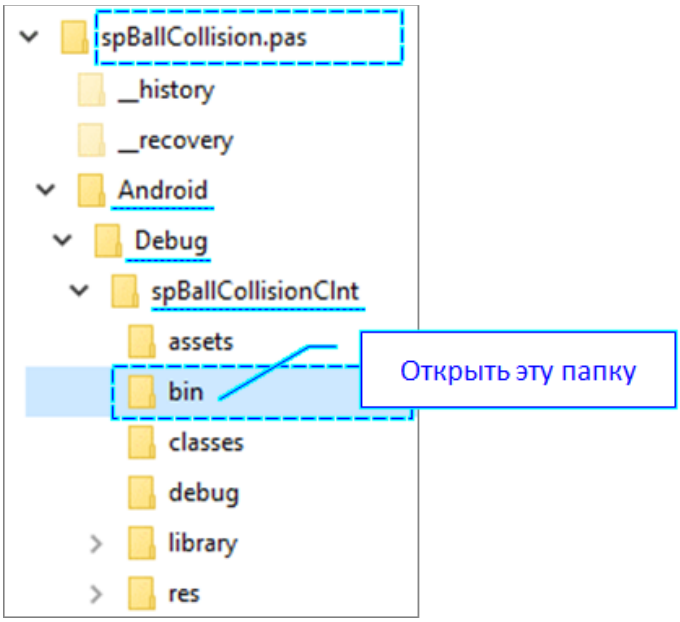

Рисунок 56 – Открыть папку **bin**

9. В папке bin найти файл **spBallCollisionClnt.apk**.

10. Установить этот файл на телефон (под Android).

4.6.1.3 Тестирование комплекса программ

Предполагается, что:

1. Сеть WI-FI доступна для информационного обмена между DataSnap-сервером и DataSnap-клиентом.

2. DataSnap-сервер (`spBallCollisionSrv.exe`) запущен.

3. Адрес и Порт Сервера в DataSnap-клиенте установлены корректно (см. рисунок 37 в разделе «4.5.3.4»).

4. DataSnap-клиент (`spBallCollisionClnt.apk`) установлен на телефон под Android.

4.6.1.3.1 Процесс тестирования

1. Запустить DataSnap-клиент на телефоне (под Android).

На рисунке ниже представлен внешний вид формы DataSnap-клиента.

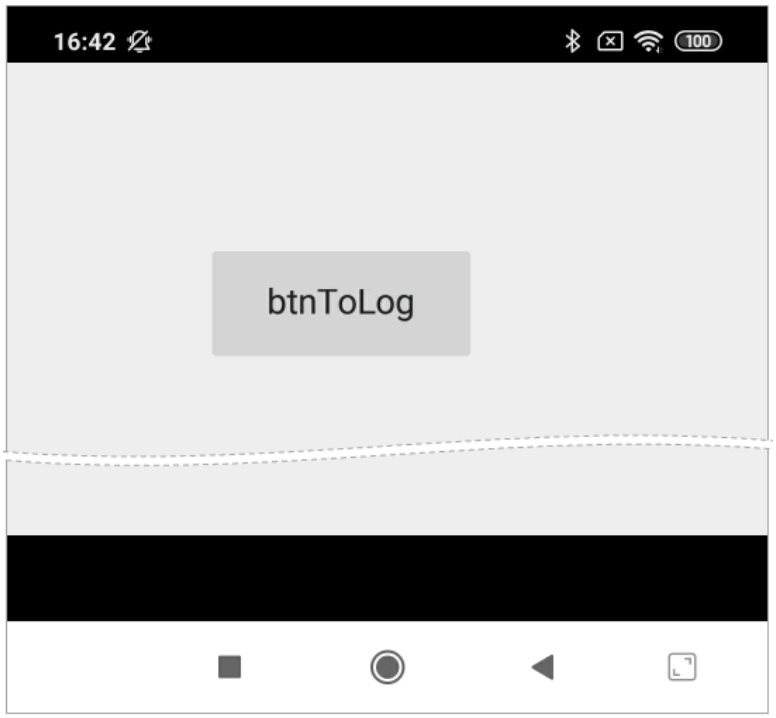

Рисунок 57 – Внешний вид формы DataSnap-клиента

2. Нажать несколько раз (с разумным интервалом времени) на кнопку **`btnToLog`** (см. рисунок выше).

3. На форме DataSnap-сервера появятся сообщения в LOG (см. рисунок ниже).

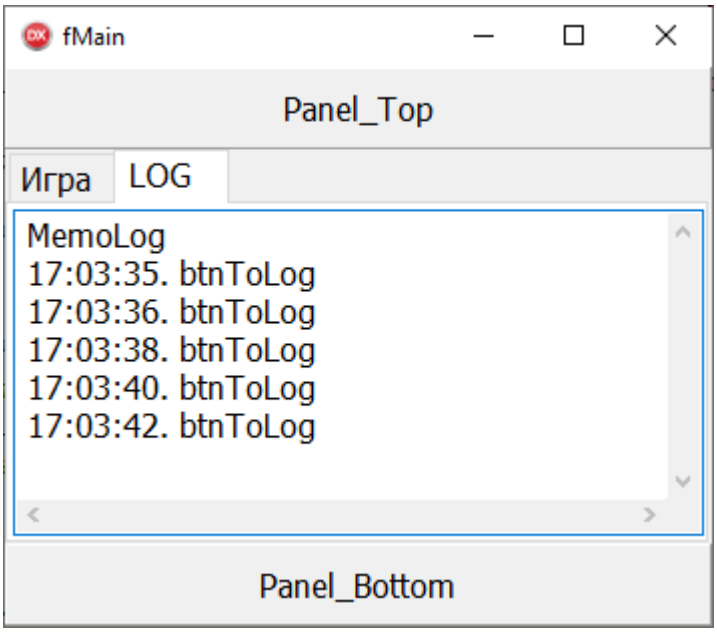

Рисунок 58 – Сообщения в LOG Сервера

В общем-то, «получили именно то, что и хотели»…

На этом этапе процесс тестирования можно считать завершенным.

4.6.2 «Подтягивание» проекта на заданный уровень функционала

4.6.2.1 DataSnap-клиент

1. Открыть файл проекта DataSnap-клиента (**spBallCollisionClnt.dpr**).

2. Внести дополнительные строки (см. исходник ниже и рисунок 59).

```
program spBallCollisionClnt;

uses
  System.StartUpCopy,
  FMX.Forms,
  //---------------------------
  //Добавить
  Androidapi.Helpers,
  FMX.Helpers.Android,
  Androidapi.JNI.App,
  Androidapi.JNI.GraphicsContentViewText,
  //---------------------------
  spBCC_001_main in 'spBCC_001_main.pas' {Form2},
  spBCC_002_ClntClasses in 'spBCC_002_ClntClasses.pas',
  spBCC_003_ClntModule   in   'spBCC_003_ClntModule.pas'   {ClientModule1:
TDataModule};

{$R *.res}

begin

  //---------------------------
  //Добавить
TAndroidHelper.Activity.setRequestedOrientation(TJActivityInfo.JavaClass.
SCREEN_ORIENTATION_SENSOR_LANDSCAPE);
  //---------------------------
  Application.Initialize;
  //---------------------------
  //Добавить
SharedActivity.getWindow.addFlags(TJWindowManager_LayoutParams.JavaClass.
FLAG_KEEP_SCREEN_ON);
  //---------------------------
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TClientModule1, ClientModule1);
  Application.Run;
end.
```

```
program spBallCollisionClnt;

uses
  System.StartUpCopy,
  FMX.Forms,
  //----------------------------
  //Добавить
  Androidapi.Helpers,
  FMX.Helpers.Android,
  Androidapi.JNI.App,
  Androidapi.JNI.GraphicsContentViewText,
  //----------------------------
  spBCC_001_main in 'spBCC_001_main.pas' {Form2},
  spBCC_002_ClntClasses in 'spBCC_002_ClntClasses.pas',
  spBCC_003_ClntModule in 'spBCC_003_ClntModule.pas' {ClientModule1: TDataModule};

{$R *.res}

begin

  //----------------------------
  //Добавить
  TAndroidHelper.Activity.setRequestedOrientation(TJActivityInfo.JavaClass.SCREEN_ORIENTATION_SENSOR_LANDSCAPE);
  //----------------------------
  Application.Initialize;
  //----------------------------
  //Добавить
  SharedActivity.getWindow.addFlags(TJWindowManager_LayoutParams.JavaClass.FLAG_KEEP_SCREEN_ON);
  //----------------------------
  Application.CreateForm(TForm2, Form2);
  Application.CreateForm(TClientModule1, ClientModule1);
  Application.Run;
end.
```

Рисунок 59 – Дополнить файл проекта DataSnap-клиента

Это нужно для того, чтобы при повороте телефона/планшета ориентация отображения на экране не менялась.

3. Доработать GUI формы следующим образом (см. рисунок ниже).

Рисунок 60 – GUI формы DataSnap-клиента

На рисунке ниже приведены идентификаторы кнопок, размещенных на форме DataSnap-клиента.

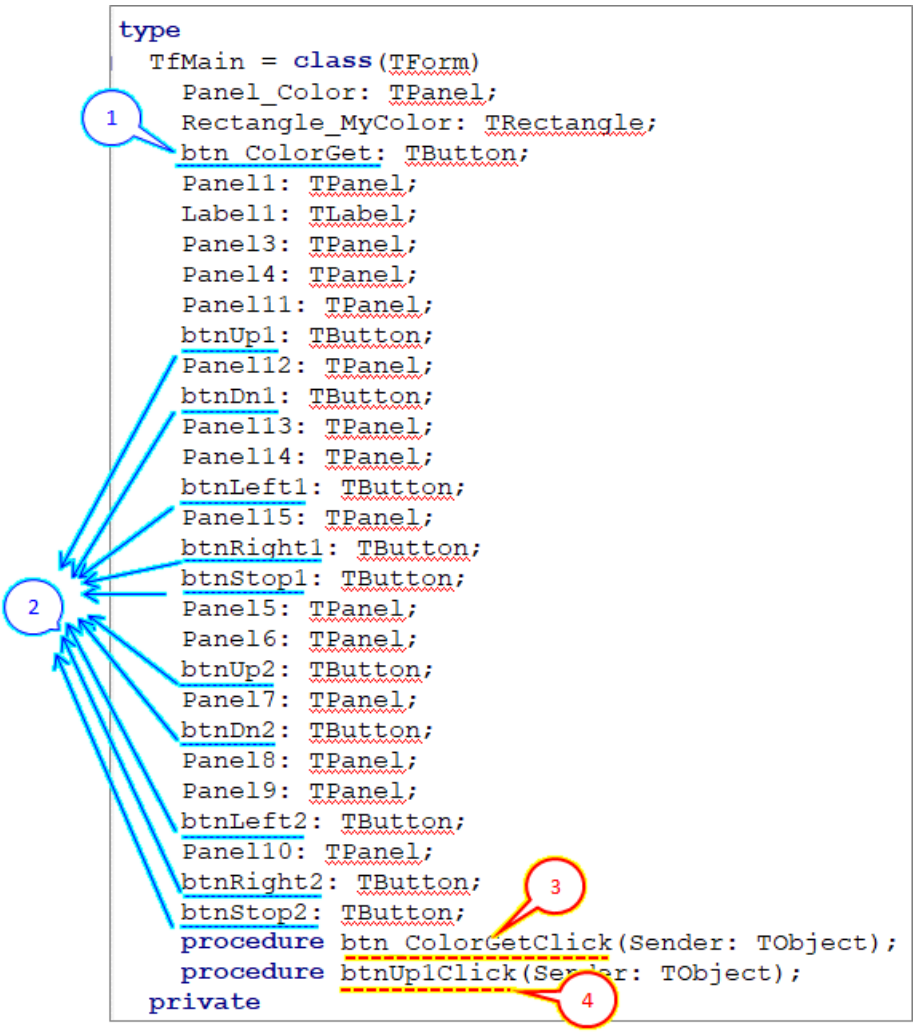

Рисунок 61 – Идентификаторы кнопок на форме DataSnap-клиента

Выноской-1 на рисунке выше обозначен идентификатор кнопки «Получи свой цвет».

Выноской-2 обозначены все остальные кнопки, предназначенные для управления шариком на форме DataSnap-сервера.

Выноской-3 обозначен обработчик события: нажатие на кнопку «Получи свой цвет».

Выноской-4 обозначен обработчик события: нажатие на любую из кнопок управления шариком (один обработчик на все эти кнопки).

4. В секцию `public` класса **`TfMain`** добавить свойство **`fMyColor`**:string (см. рисунко ниже).

```
public
  { Public declarations }

  fMyColor:string;

end;
```

Рисунок 62 – Свойство **`fMyColor`**

5. Реализация обработчика **btn_ColorGetClick** приведена на рисунке ниже.

```
procedure TfMain.btn_ColorGetClick(Sender: TObject);
Var
  Sx:string;
  ItsOk:boolean;
begin
  ItsOk:=false;
  Sx:=ClientModule1.ServerMethods1Client.Color_GiveMe(fMyColor);
  if (Sx='R') or (Sx='B') then ItsOk:=true;
  if ItsOk then begin
     fMyColor:=Sx;
     if (fMyColor='R') then begin
        Rectangle_MyColor.Fill.Color:=TAlphaColorRec.Red;
        btn_ColorGet.Text:='';
     end;
     if (fMyColor='B') then begin
        Rectangle_MyColor.Fill.Color:=TAlphaColorRec.Blue;
        btn_ColorGet.Text:='';
     end;
  end;
end;
```

Рисунок 63 – Реализация обработчика **btn_ColorGetClick**

6. Реализация обработчика **btnUp1Click** приведена на рисунке ниже.

```
procedure TfMain.btnUp1Click(Sender: TObject);
begin
  ClientModule1.ServerMethods1Client.Btn_Presh(fMyColor, (Sender as TButton).Name);
end;
```

Рисунок 64 – Реализация обработчика **btnUp1Click**

На этом разработку DataSnap-клиента можно считать законченной.

Чтобы протестировать функционал DataSnap-клиента – следует немного доработать **DataSnap-сервер**.

А именно, нужно в модуле **spBCS_003_SrvMethods** доработать заглушки методов **Color_GiveMe** и **Btn_Presh** следующим образом (см. рисунок ниже).

```
function TServerMethods1.Color_GiveMe(sColorNow: string): string;
//Обработать запрос на получение Цвета от DataSnap-клиента
begin
  Result:='';
  fMain.AddToLog('Color_GiveMe: '+sColorNow);
end;

function TServerMethods1.Btn_Presh(sColor:string; sBtnName: string): string;
//Обработать запрос на нажатие кнопку в DataSnap-клиенте
begin
  Result:='';
  fMain.AddToLog('Btn_Presh: '+sColor + ';' + sBtnName);
end;
//****************************************
```

Рисунок 65 – Доработка методов **Color_GiveMe** и **Btn_Presh**

Результаты теста представлены на рисунках ниже.

Рисунок 66 – DataSnap-клиент, запущенный на Android-устройстве

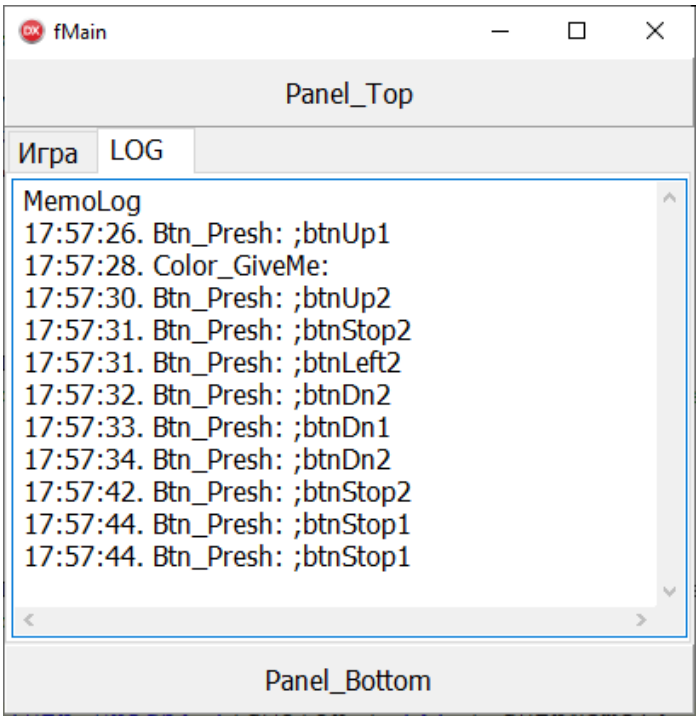

Рисунок 67 – Отображение факта нажатия кнопок DataSnap-клиента в ЛОГе DataSnap-сервера

На этом этапе процесс тестирования можно также считать завершенным.

4.6.2.2 DataSnap-сервер

На рисунке ниже приведены компоненты, которые используются в процессе реализации «заявленного» функционала на DataSnap-сервере.

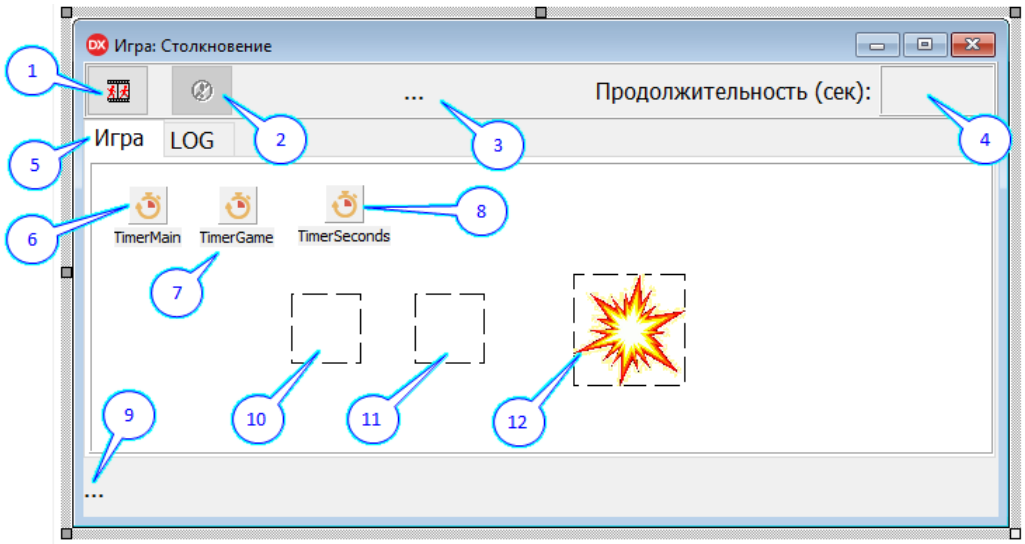

Рисунок 68 – Компоненты на главной форме приложения

Выноски, размещенные на рисунке выше, приведены в таблице ниже.

Таблица 3 – Выноски, размещенные на рисунке выше

Выноска	Идентификатор компонента	Тип (класс) компонента	Назначение
1	2	3	4
1	**sbStartGame**	TSpeedButton	Кнопка «Старт»
2	**sbStopGame**	TSpeedButton	Кнопка «СТОП-Игра»
3	**Label_Info**	TLabel	Label_Info
4	**Label_Seconds**	TLabel	Визуализация секунд игрового времени
5	**tsGame**	TTabSheet	Вкладка компонента компонента **pcMain**
6	**TimerMain**	TTimer	Таймер отчета текущего времени
7	**TimerGame**	TTimer	Движение шаров (обработка команд, поступивших от DataSnap-клиентов)
8	**TimerSeconds**	TTimer	Отсчет секунд игрового времени (от старта и до столкновения)
9	**LabelDT**	TLabel	Текущее время
10	**R_Image**	TImage	Красный шар
11	**B_Image**	TImage	Синий шар
12	**Image_Bang**	TImage	Имитация взрыва от столкновения шаров

4.6.2.2.1 Доработка метода TServerMethods1.Color_GiveMe

Исходный текст доработанного метода приведен на рисунке ниже.

```
function TServerMethods1.Color_GiveMe(sColorNow: string): string;
//Обработать запрос на получение Цвета от DataSnap-клиента
Var
  CountZ:integer;
begin
  //fMain.AddToLog('Color_GiveMe: '+sColorNow);
  Result:='Отказано';
  CountZ:=0;
  sColorNow:=trim(sColorNow);
  if sColorNow='' then begin
     if not Red_In_Use then begin
        Result:='R';
        Red_In_Use:=true;
     end
     else begin
        if not Blue_In_Use then begin
           Result:='B';
           Blue_In_Use:=true;
        end;
     end;

     if (Red_In_Use) then CountZ:=CountZ+1;
     if (Blue_In_Use) then CountZ:=CountZ+1;

     if (CountZ = 2) then begin
        fMain.Game_Start; //Старт игры
     end;

   end;
end;
```

Рисунок 69 – Доработка метода `TServerMethods1.Color_GiveMe`

4.6.2.2.2 Доработка метода TServerMethods1.Btn_Presh

Исходный текст доработанного метода приведен на рисунке ниже.

```
function TServerMethods1.Btn_Presh(sColor:string; sBtnName: string): string;
//Обработать запрос на нажатие кнопку в DataSnap-клиенте
begin
  Result:='';
  //fMain.AddToLog('Btn_Presh: '+sColor + ';' + sBtnName);
  sColor:=trim(sColor);
  if sColor<>'' then begin
     if (sColor='R') or (sColor='B') then begin
        sBtnName:=trim(sBtnName);
        if sBtnName<>'' then begin
           //Перемещение шаров
           if fMain.Circle_Move(sColor, sBtnName) then begin
              Result:='Ok';
           end;
        end;
     end;
  end;
end;
```

Рисунок 70 – Доработка метода `TServerMethods1.Btn_Presh`

4.7 Тестирование комплекса программ

Предполагается, что:

1. Оба программных модуля скомпилированы (`spBallCollisionSrv.exe` и `spBallCollisionClnt.apk`).
2. Модуль DataSnap-клиент (`spBallCollisionClnt.apk`) установлен на двух Android-устройствах (телефонах).

4.7.1 Последовательность действий.

1. Запустить DataSnap-сервер (пиктограмма приведена на рисунке ниже).

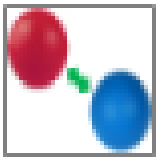

Рисунок 71 – Пиктограмма DataSnap-сервера

На рисунке ниже приведен внешний вид формы DataSnap-сервера после запуска.

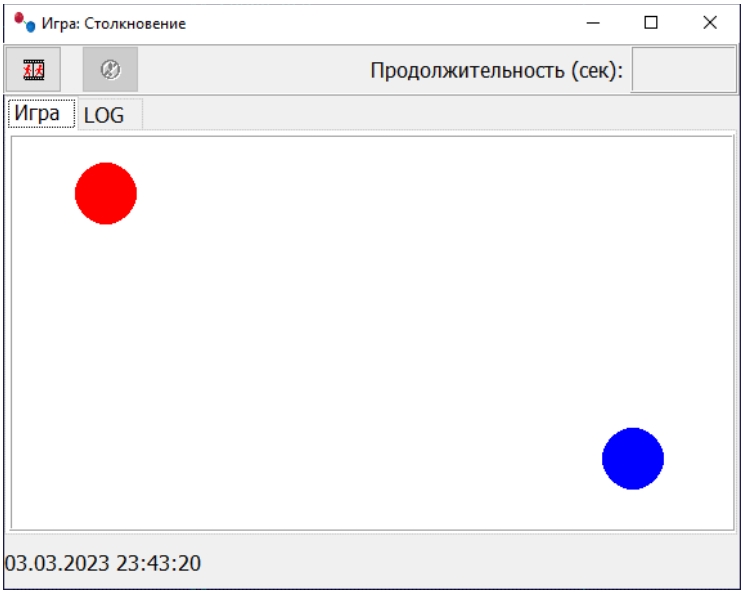

Рисунок 72 – Внешний вид GUI-формы DataSnap-сервера после запуска

2. На каждом Android-устройстве (телефоне) запустить DataSnap-клиент (пиктограмма приведена на рисунке ниже).

Рисунок 73 – Пиктограмма DataSnap-клиента

На рисунке ниже приведен внешний вид формы DataSnap-клиента после запуска.

Рисунок 74 – Вид формы DataSnap-клиента после запуска

3. На одном из телефонов нажать на кнопку «Получи свой цвет» (см. рисунок выше).
   На рисунке ниже приведен результат для телефона, на котором эта кнопка была нажата первой (по очереди).

Рисунок 75 – Вид формы DataSnap-клиента, «получившего» красный цвет

На рисунке ниже приведен результат для телефона, на котором эта кнопка была нажата второй (по очереди).

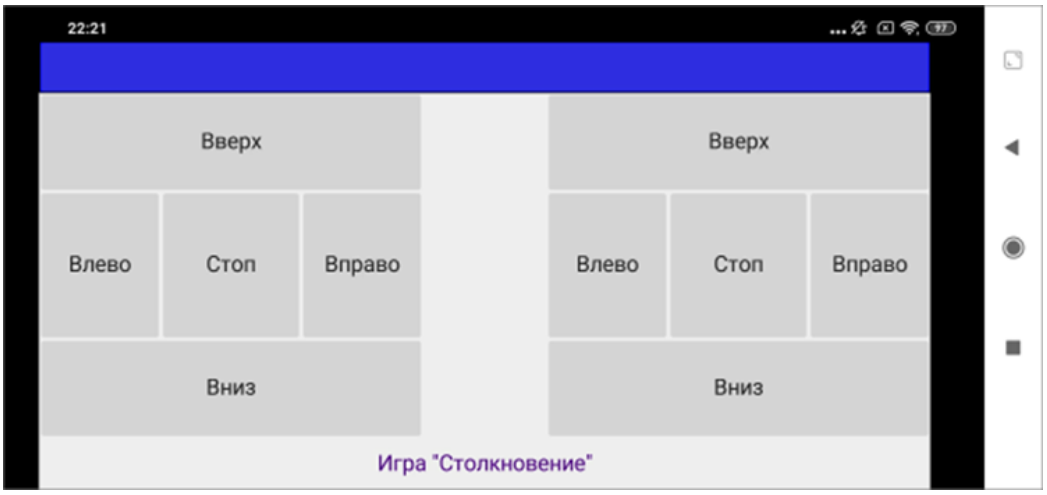

Рисунок 76 – Вид формы DataSnap-клиента, «получившего» синий цвет

Важно! Игра «стартует» автоматически сразу после того, как оба телефона получили свой цвет (эффект такой же, как при нажатии на кнопку «Старт» DataSnap-сервера, см. выноску-1 на рисунке 68).

4. Далее, протестировать управление шариками, нажимая на кнопку телефонов.

5. Затем, нажать на кнопку «Стоп» DataSnap-сервера (см. выноску-2 на рисунке 68). Результат приведен на рисунке ниже.

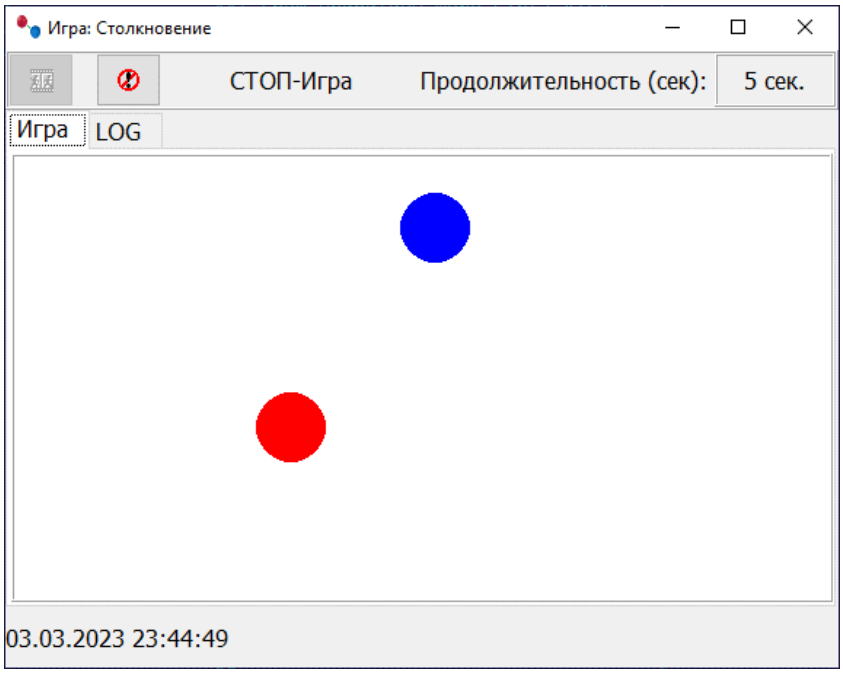

Рисунок 77 – СТОП-Игра

6. Возобновить игру, нажав на кнопку «Старт» DataSnap-сервера (см. выноску-1 на рисунке 68). Результат приведен на рисунке ниже.

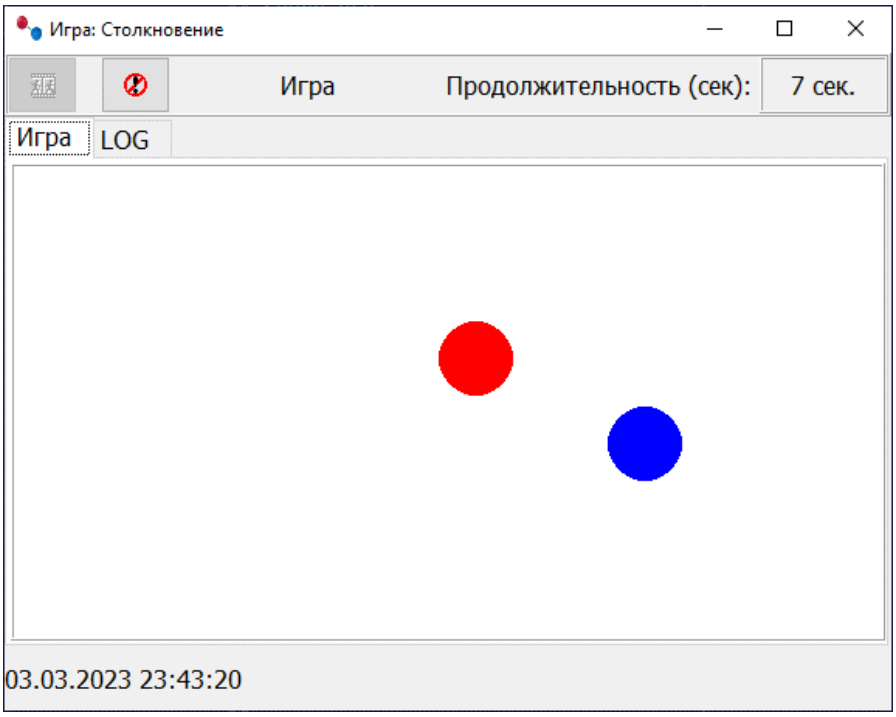

Рисунок 78 – Сближение шаров

7. Управляя шариками с телефонов (DataSnap-клиенты), добиться их столкновения на форме DataSnap-сервера. Результат приведен на рисунке ниже.

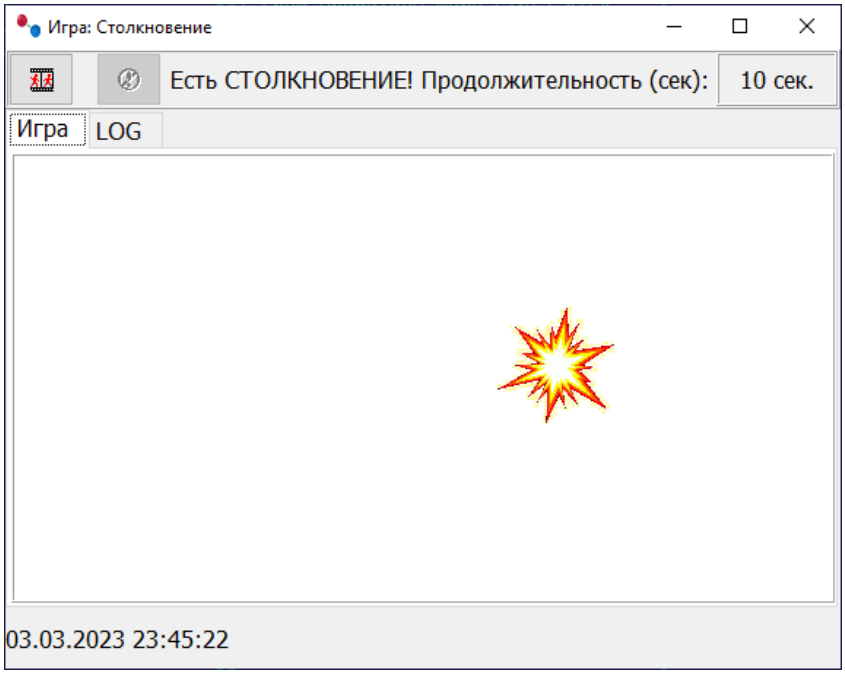

Рисунок 79 – Столкновение шаров

На этом процесс тестирования можно считать завершенным.

5 Наращивание функционала программного комплекса в процессе его эксплуатации

Предположим, что в рамках комплекса программ появилось желание реализовать несколько типовых игр.

В этом случае (для начала) необходимо добавить соответствующие новые методы как на Сервере, так и на Клиенте.

И делать это придется уже полностью «ручками», что является знАчимо более трудоемким, чем при разработке нового проекта.

Применительно к рассматриваемому примеру добавим один метод: возможность получить на Клиенте название игры, которая активирована на Сервере.

Имя метода:

`GameName_Get`.

Входные параметры:

`sColor:string;` //код цвета шарика, который назначен Клиенту.

5.1 Доработка DataSnap-сервера

Поскольку в этом проекте реализована только одна игра, то достаточно создать строковую константу, где и будет храниться название игры.

5.1.1 Последовательность действий

1. Добавить константу `NameOfThisGame` в модуль `spBCS_000_lib.pas` (см. рисунок ниже).

```
unit spBCS_000_lib;

interface

Uses
  Dialogs,
  Windows,
  SysUTILS;

Const
  NameOfThisGame = 'Игра: Столкновение шариков на плоскости';
```

Рисунок 80 – Константа `NameOfThisGame`

2. Открыть модуль `spBCS_003_SrvMethods.pas`, где добавить новый метод `GameName_Get` (см. рисунки ниже).

```
unit spBCS_003_SrvMethods;

interface

uses
    spBCS_000_lib,
    System.SysUtils, System.Classes, System.Json;

type
  TServerMethods1 = class(TDSServerModule)
  private
    { Private declarations }
  public
    { Public declarations }
    //-----------
    //Методы, созданные по умолчанию
    //Реакция сервера на нажатие кнопку в DataSnap-клиенте
    function Btn_Presh(sColor:string; sBtnName: string): string;
    //-----------

    //-----------
    //Новый метод
    //Получить название игры, которая активирована на Сервере
    function GameName_Get(sColor: string): string;
    //-----------

  end;
```

Рисунок 81 – Метод `GameName_Get` DataSnap-сервера

```
//*****************************************
//Новый метод
//Получить название игры, которая активирована на Сервере
function TServerMethods1.GameName_Get(sColor: string): string;
begin
  Result:=NameOfThisGame;
end;
//*****************************************
```

Рисунок 82 – Метод `GameName_Get` DataSnap-сервера (продолжение)

3. Скомпилировать DataSnap-сервер и запустить его (см. рисунок 23).

5.2 Доработка DataSnap-клиента

1. Открыть модуль `spBCC_002_ClntClasses.pas`.

2. В секцию private класса `TServerMethods1Client` добавить:
   `FGameName_Get: TDBXCommand;`
   (см., также, рисунок ниже)

```
type
  TServerMethods1Client = class(TDSAdminClient)
  private
    FEchoStringCommand: TDBXCommand;
    FReverseStringCommand: TDBXCommand;
    FLogStringCommand: TDBXCommand;
    FColor_GiveMeCommand: TDBXCommand;
    FBtn_PreshCommand: TDBXCommand;

    //---------------
    //Новый метод
    FGameName_GetCommand: TDBXCommand;
    //---------------
```

Рисунок 83 – Секция private класса `TServerMethods1Client`

3. В секцию public класса `TServerMethods1Client` добавить новый метод (см. рисунок ниже).

```
public
  constructor Create(ADBXConnection: TDBXConnection); overload;
  constructor Create(ADBXConnection: TDBXConnection; AInstanceOwner: Boolean); overload;
  destructor Destroy; override;
  function EchoString(Value: string): string;
  function ReverseString(Value: string): string;
  function LogString(sValue: string): string;
  function Color_GiveMe(sColorNow: string): string;
  function Btn_Presh(sColor: string; sBtnName: string): string;
  //-----------
  //Новый метод
  //Получить название игры, которая активирована на Сервере
  function GameName_Get(sColor: string): string;
  //-----------
```

Рисунок 84 – Секция public класса `TServerMethods1Client`

4. Далее, в разделе implementation сформировать реализацию нового метода (см. рисунок ниже).

```
//***************************************
//Новый метод
//Получить название игры, которая активирована на Сервере
function TServerMethods1Client.GameName_Get(sColor: string): string;
begin
  if FGameName_GetCommand = nil then
  begin
    FGameName_GetCommand := FDBXConnection.CreateCommand;
    FGameName_GetCommand.CommandType := TDBXCommandTypes.DSServerMethod;
    FGameName_GetCommand.Text := 'TServerMethods1.GameName_Get';
    FGameName_GetCommand.Prepare;
  end;

  //.........................
  //ВАЖНО! Индексы массива Parameters[] должны соответствовать
  //перечню входных параметров метода, а также, возвращаемому значению
  FGameName_GetCommand.Parameters[0].Value.SetWideString(sColor);
  FGameName_GetCommand.ExecuteUpdate;
  Result := FGameName_GetCommand.Parameters[1].Value.GetWideString;
  //.........................

end;
//***************************************
```

Рисунок 85 – Реализация метода `GameName_Get` DataSnap-клиента

5. Затем, необходимо внести соответствующие изменения в деструктор класса `TServerMethods1Client` (см. рисунок ниже).

```
destructor TServerMethods1Client.Destroy;
begin
  FEchoStringCommand.DisposeOf;
  FReverseStringCommand.DisposeOf;
  FLogStringCommand.DisposeOf;
  FColor_GiveMeCommand.DisposeOf;
  FBtn_PreshCommand.DisposeOf;
  //---------------
  //Новый метод
  FGameName_GetCommand.DisposeOf;
  //---------------
  inherited;
end;
```

Рисунок 86 – Деструктор класса `TServerMethods1Client`

6. Отрыть модуль `spBCC_001_main` (форма DataSnap-клиента).

7. На форме DataSnap-клиента заменить компонент `TLabel` на `TButton` (см. рисунок ниже).

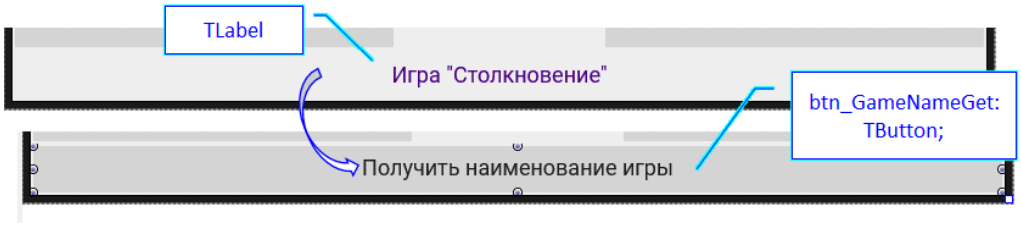

Рисунок 87 – Заменить компонент `TLabel` на `TButton`

8. Сформировать обработчик нажатия на кнопку `btn_GameNameGet` (см. рисунок ниже).

```
//*****************************
procedure TfMain.btn_GameNameGetClick(Sender: TObject);
begin
  btn_GameNameGet.Text := ClientModule1.ServerMethods1Client.GameName_Get(fMyColor);
end;
//*****************************
```

Рисунок 88 – Обработчик нажатия на кнопку `btn_GameNameGet`

9. Далее, сформировать модуль **`spBallCollisionClnt.apk`** (см. раздел «4.6.1.2» на странице 40) и установить его на телефоне.

5.3 Тестирование

ОписАние процесса тестирования приведено в разделе «4.7 Тестирование комплекса программ».

6 Выводы

Летят Петька с Василием Иванычем в самолете:
– Петька, прибор?!
– Тридцать!
– Что «тридцать»?
– А что «прибор»?

А что «выводы»?

Возможности DataSnap-технологии существенно интересны и полезны для практически применения.

Например, на основе этой технологии создан комплекс программ «Управление голосовым вводом текста/данных с телефона в MS Word, MS Excel и текстовый редактор». См. здесь: https://roamer55.ru/my_download/progs/os_win/spDSW/