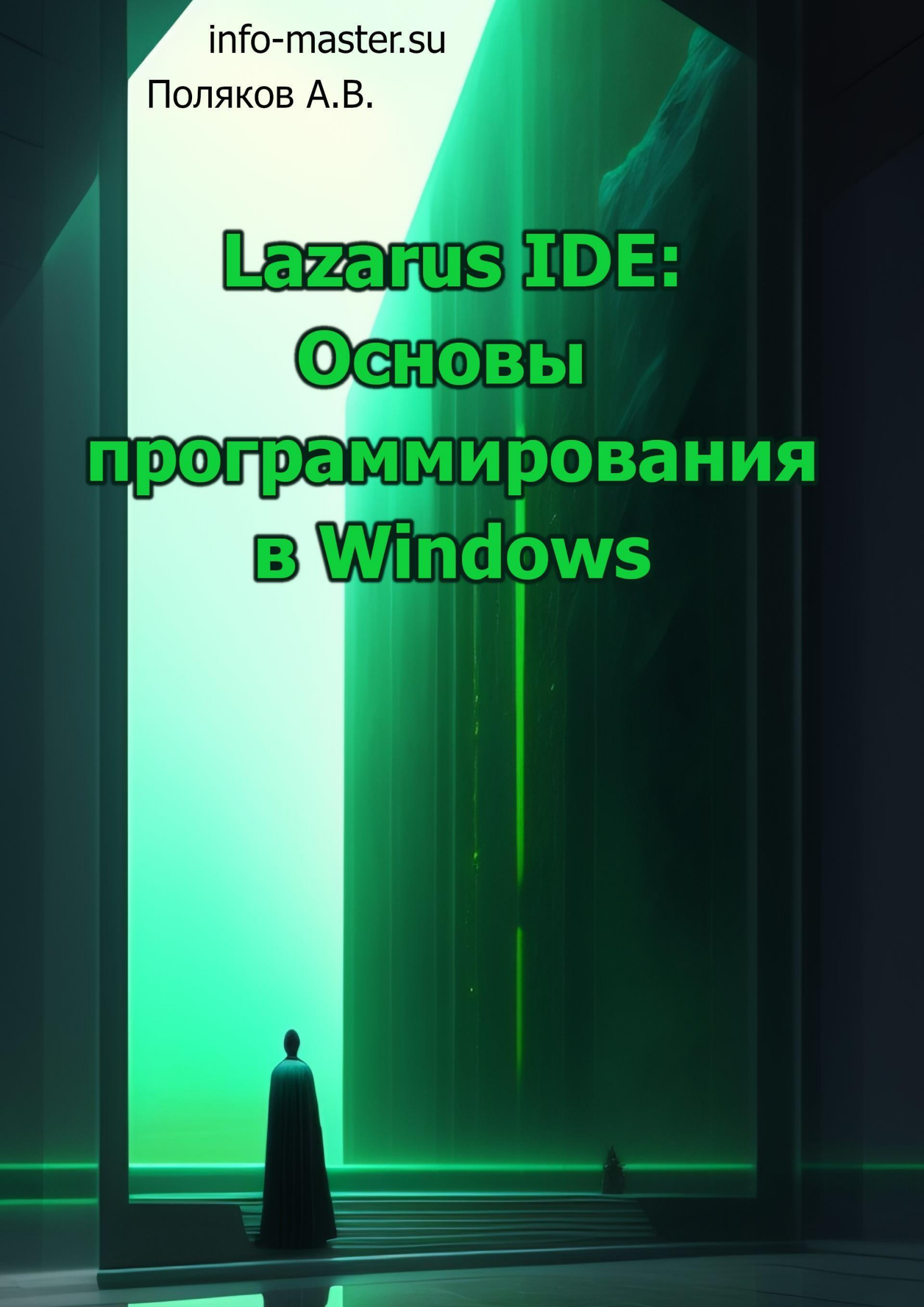
info-master.su
Поляков А.В.

Lazarus IDE: Основы программирования в Windows



СОДЕРЖАНИЕ

ПРЕДИСЛОВИЕ

Эту книгу я собирался написать очень давно. Однако с годами свободного времени становилось только меньше, поэтому постоянно откладывал. Но вот этот день настал!

Данная книга посвящена созданию приложений с графическим интерфейсом в среде разработки Lazarus. На момент начала работы над этой книгой была доступна версия 2.0.12 этой замечательной программы. Когда эта книга попала к вам, то наверняка имеются более свежие версии. Но, скорее всего, они не сильно отличаются от версии 2.0.12, поэтому можете смело применять книгу и к более новым редакциям Lazarus.

Непосредственно о Lazarus я будут рассказывать на протяжении всей книги, поэтому в предисловии только очень краткие сведения.

Lazarus – это бесплатная среда разработки, похожая на Delphi. Но я решил использовать именно Lazarus для примеров программ, потому что «бесплатность» для многих начинающих программистов является решающим фактором при выборе.

А теперь пара важных вопросов, которые я просто обязан озвучить, и советую вам прочитать следующие несколько страниц в разделе ВВЕДЕНИЕ, чтобы потом не пришлось к ним возвращаться.

ВВЕДЕНИЕ

Современное программирование – это очень, очень, очень сложное дело с огромным количеством различных направлений. И, например, программист 1С офигенно высокого уровня может оказаться полным чайником в веб-программировании.

Но даже если вы нацелены на какое-то одно направление, например, на создание прикладных программ для Windows (о чём и пойдёт речь в этой книге), то и здесь вы не сможете закрыть все вопросы за один подход. И эта книга – лишь начало пути. Это надо понимать – данная книга не сделает из вас законченного профессионала по Lazarus, а тем более по программированию для Windows вообще. Потому как эта ниша просто фантастически огромна.

Но надо с чего-то начинать. И вот для начала данная книга будет очень даже хорошим способом продвинуться в этом направлении (программировании для Windows). Здесь нет ничего лишнего, и в то же время описаны все основные приёмы, которые позволяют решать большинство наиболее часто встречающихся задач.

Основные разделы книги перечислены ниже.

ПРЕДИСЛОВИЕ И ВВЕДЕНИЕ
Общие сведения о книге и о том, как ей пользоваться.

1. БЫСТРЫЙ СТАРТ
Раздел для самых нетерпеливых. Здесь мы создадим нашу первую программу и разберёмся с основными принципами создания программ для Windows в Lazarus.

2. СРЕДА РАЗРАБОТКИ
Здесь мы подробно изучим возможности среды разработки. Разберёмся с тем, для чего нужны те или иные команды меню, как настроить среду под себя и т.п.

3. СОЗДАНИЕ ПРОГРАММ ДЛЯ WINDOWS
Здесь мы уже будем говорить о том, как использовать те или иные возможности среды разработки для создания пользовательского интерфейса и программ.

4. ПРИМЕРЫ
Это, возможно, для кого-то будет самый любопытный раздел. Здесь мы создадим несколько полезных и настоящих программ.

Об исходных файлах

Если вы получили эту книгу из официального источника, то вместе с ней вы получили все исходные файлы примеров программ, описанных в этой книге.

Исходные файлы находятся в папке `SOURCE`. Имена папок с примером для конкретной главы состоят из номеров этих глав. Например:

03010102 – в этой папке хранятся исходные файлы примера к разделу **3-1-1-2**.

В некоторых случаях последняя цифра в имени может означать не номер подраздела, а номер примера (по порядку).

О ссылках

В этой книге довольно часто будут встречаться ссылки, которые могут вести на сайт в Интернете, на какой-то раздел в самой книге или на видеоурок.

Ссылки выглядят вот так:

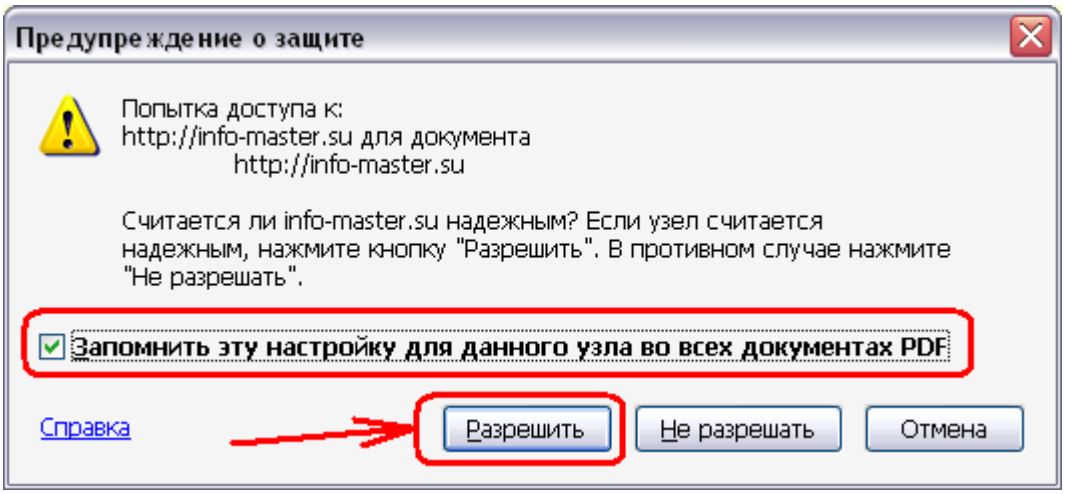
Основы программирования

или так:

http://info-master.su/programming/

Если вы щёлкните левой кнопкой мыши по такой ссылке, то вы перейдёте по соответствующему адресу.

Если ссылка ведёт на страницу в Интернете, то программа, с помощью которой вы читаете эту книгу, может запросить у вас разрешения на переход. В этом случае ответьте ДА (или YES, или OK, или Allow, или РАЗРЕШИТЬ, или что-то типа того – см. рисунок ниже).



Оглавление книги – это тоже ссылки. Если вы щёлкните по названию раздела в оглавлении, то вы перейдёте в соответствующий раздел.

Если ваша программа просмотра этой книги почему-то не даёт перейти по ссылке, то в случае, если ссылка выглядит как адрес в Интернете (например, так: http://info-master.su/programming/), вы можете просто скопировать её и вставить в адресную строку браузера или вручную переписать эту ссылку в адресной строке браузера, а затем нажать на клавишу ENTER.

Если вы плохо понимаете, о чём я говорил в предыдущем абзаце, то советую почитать бесплатную книгу «Чайникам о компьютерах» здесь: https://info-master.su/mail/comp/

О видеоуроках

К некоторым разделам книги прилагаются видеоуроки. Вы можете найти их в папке `VIDEO`.

Кроме того, в том разделе, к которому прилагается видеоурок, указан путь к этому уроку (то есть имя видеофайла).

Если на вашем компьютере не воспроизводится видео, то попробуйте установить другой видеоплеер или обновить видеокодеки.

1. БЫСТРЫЙ СТАРТ

Скоро только кошки родятся…
Остап Бендер
(Ильф и Петров, "12 стульев")

Этот раздел для самых нетерпеливых. Здесь мы быстро освоим главные принципы программирования в Lazarus и сразу напишем нашу первую программу с графическим интерфейсом для Windows.

Таким образом вы быстро сможете въехать в тему и уже начать создавать свои первые маленькие программы, а потом уже, пошагово изучая следующие разделы, вы сможете перейти к более серьёзным проектам.

Однако не забывайте бессмертное выражение Остапа Бендера, которое приведено в качестве эпиграфа к этому разделу. Быстро можно только "стартануть", но вообще программист обучается всю жизнь. И эта книга - только начало вашего пути (но, надеюсь, ещё не конец моего :-)

1-1. Что такое визуальная среда разработки

Видеоприложение к этому разделу находится здесь:
`VIDEO\01010000 Что такое визуальная среда разработки.mp4`

Если вы сразу перешли к этой книге, не изучив "Основы программирования", то на всякий случай напомню, что такое среда разработки.

Итак, **среда разработки** – это набор программ, с помощью которых выполняется разработка и отладка программного обеспечения. В обычной среде разработки программа создаётся путём написания исходных текстов (исходных кодов) на каком-либо языке программирования. Причём эти исходные коды можно писать даже в обычном текстовом редакторе, без каких-либо дополнительных программ.

IDE – это **I**ntegrated **D**evelopment **E**nvironment (интегрированная среда разработки) – это английское название среды разработки.

Визуальная среда разработки – это среда (средство) разработки, где программы создаются не только путём написания исходных текстов, но и с помощью особых графических компонентов, которые, например, с помощью мыши, размещаются на окнах разрабатываемой программы.

Визуальные средства разработки очень сильно упрощают жизнь программисту, потому что большинство стандартных элементов оконного приложения (такие как кнопки, поля ввода-вывода, графики, таблицы и т.п.) уже созданы, и программисту не надо писать большой и сложный код для работы с такими элементами.

Многие начинающие программисты думают, что оконные приложения можно создавать только с помощью визуальных средств разработки. Но на самом деле это не так. И в книге “Основы C++” я показывал пример создания оконного приложения в невизуальной среде разработки. Но в этом случае приходится весь исходный код писать вручную. И если вы с этим сталкивались, то знаете, насколько это сложно и как много отнимает времени.

Поэтому обычные средства разработки сейчас если и используются, то лишь для создания консольных приложений. Но большинство современных программистов давно уже перешли на визуальные средства, потому что с их помощью можно создавать как консольные, так и графические (оконные приложения).

Ну и напоследок, для совсем-совсем начинающих программистов покажу отличия между консольным и графическим приложением:

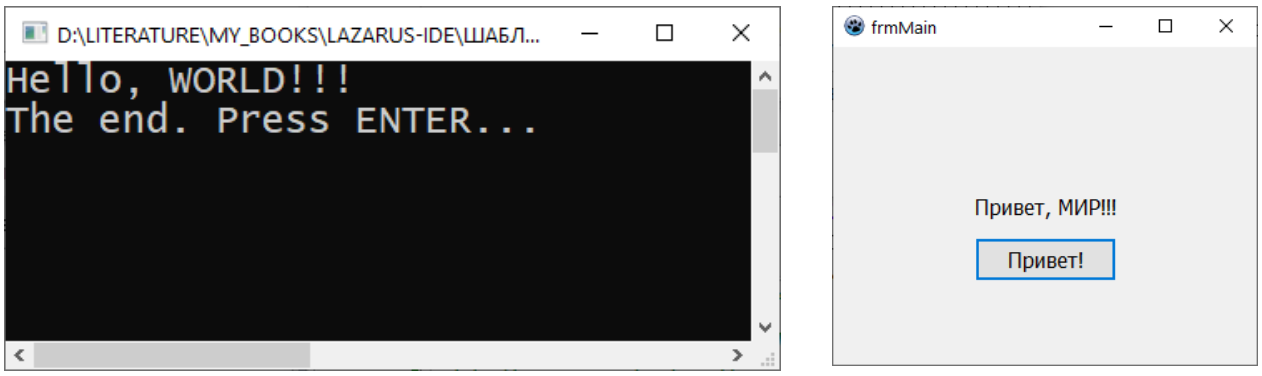


Рис. 1-1. Консольное (слева) и графическое (справа) приложения.

Как видите, консольное приложение отличается от графического. Однако главное отличие – это не внешний вид.

Главное отличие заключается в следующем:

- Консольное приложение допускает только текстовый ввод-вывод данных.
- Оконное (графическое) приложение может выводить ещё и графические данные (то есть можно рисовать на окне любые фигуры).

Полезные ссылки:

- Книга Основы программирования 2.0 (если вы не украли книгу по Lazarus, то «Основы программирования» вы получили вместе с ней в комплекте)
- Английский за 3 месяца
- Книга Основы C++

Ну а теперь переходим к практике…

1-2. Где взять и как установить среду разработки Lazarus

Видеоприложение к этому разделу находится здесь:
`VIDEO\01020000 Где взять и как установить Lazarus.mp4`

В этой книге я не буду рассказывать об основах Lazarus. Потому что я рассказывал об этом в книге "Основы программирования", которую вы получили в качестве подарка к этому курсу. В той же книге я рассказывал о том, где взять и как установить Lazarus. Однако здесь этот рассказ я повторю, поскольку с тех пор, как я написал книгу "Основы программирования", прошло уже немало времени, и появились новые версии Lazarus.

Итак, скачать Lazarus можно здесь:

https://www.lazarus-ide.org/

Когда вы перейдёте по ссылке, то в верхнем правом углу увидите кнопку с надписью DOWNLOAD NOW. Можно сразу щёлкнуть по ней, и вы перейдёте на страницу скачивания. Однако я советую вам не спешить.

Дело в том, что сайт пытается определить вашу операционную систему и, соответственно, предлагает вам версию Lazarus именно для этой операционной системы.

Однако, во-первых, он может ошибиться. А во-вторых, может оказаться, что вы скачиваете Lazarus вовсе не для того компьютера, с которого зашли на сайт и пытаетесь скачать IDE.

Так что я обычно поступаю по-другому: щёлкаю по надписи OTHER, и в выпадающем списке выбираю нужную мне операционную систему (см. рис. 1.2).

Загрузка обычно начинается автоматически. Но если этого не случилось, то следуйте инструкциям на странице загрузки.

ВАЖНО!

Если вы собираетесь создавать программы для 32-разрядных операционных систем, то советую скачивать 32-разрядную Lazarus, даже если ваша операционная система 64-разрядная.

ВНИМАНИЕ!

Некоторые антивирусы, а также браузеры, не дают открыть страницу загрузки. Если это ваш случай - отключите антивирус или защиту браузера. А вообще от антивирусов больше вреда, чем пользы.

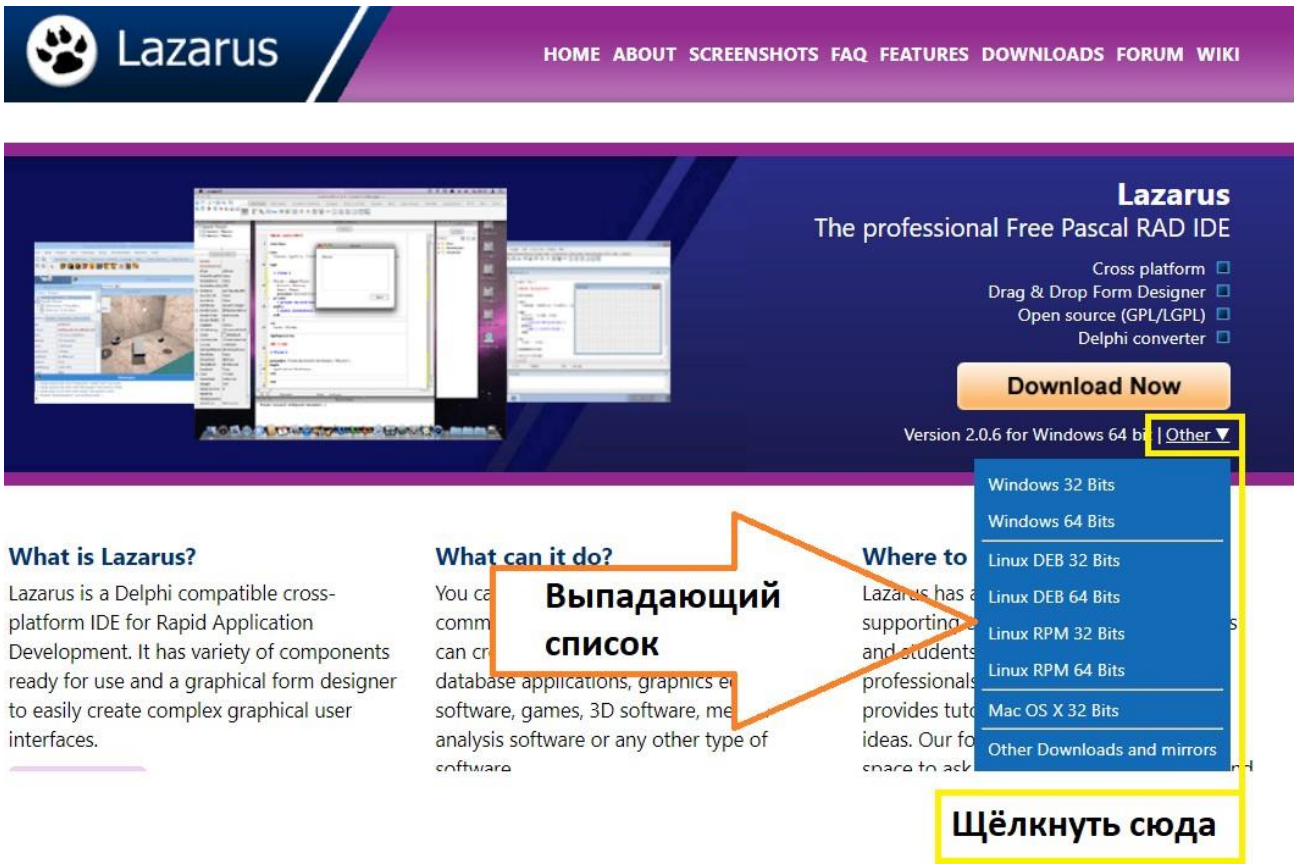


Рис. 1-2. Выбор операционной системы при загрузке Lazarus.

Если вам никак не удаётся скачать Lazarus, то версию 2.0.12 для 64-х разрядной Windows вы можете найти здесь (пароль для архива 123):

https://drive.google.com/file/d/1C0pzNWZ2YEHgf9Pb-_az7nn_eIIPI8is/view?usp=share_link

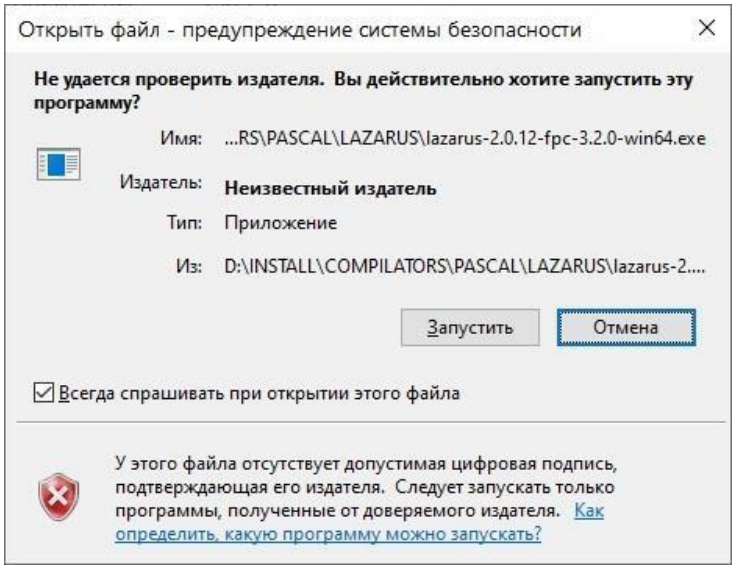
Файл установщика называется `lazarus-2.0.12-fpc-3.2.0-win64.exe`.

Теперь переходим к установке. Запускаем файл-установщик (в моём случае это `lazarus-2.0.12-fpc-3.2.0-win64.exe`, в вашем случае это может быть другой файл, если у вас другая операционная система или вы скачали более новую версию).

ПРИМЕЧАНИЕ

Если у вас другая версия Lazarus (не 2.0.12), то процесс установки может немного отличаться от описанного здесь. Но отличия не будут очень сильными, так что вы справитесь.

При запуске установщика в Windows 10 вы увидите такое сообщение:



а потом ещё и такое

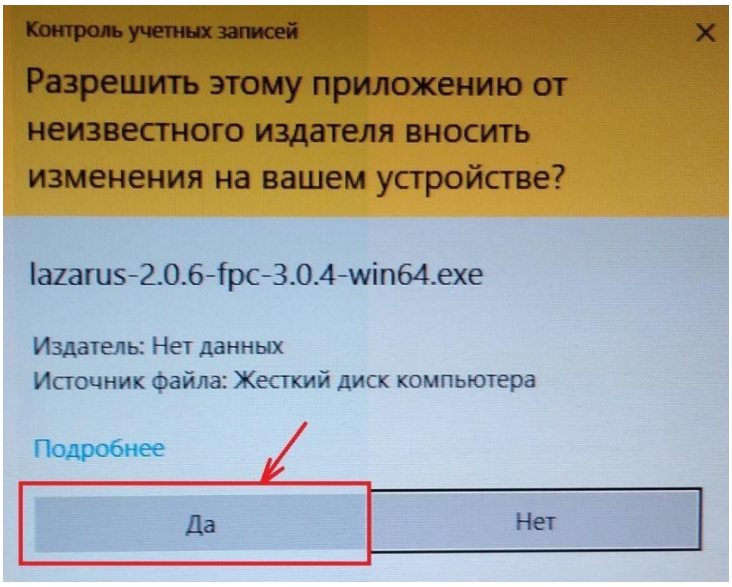


Рис. 1-3. Запрос на разрешение установки в Windows 10.

В других операционных системах вид сообщения будет другим, но суть будет такой же – вас спросят, доверять ли этому издателю (точнее, “Разрешить ли этому приложению от неизвестного издателя вносить изменения на вашем устройстве?”). Надо ответить ДА.

Появится окно, где надо выбрать язык и нажать ОК:

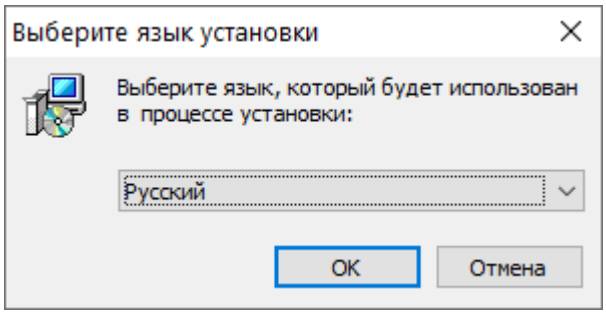


Рис. 1-4. Выбор языка.

Затем вы увидите окно мастера установки, где надо просто нажать ДАЛЕЕ:

Рис. 1-5. Мастер установки.

Затем появится окно выбора каталога установки:

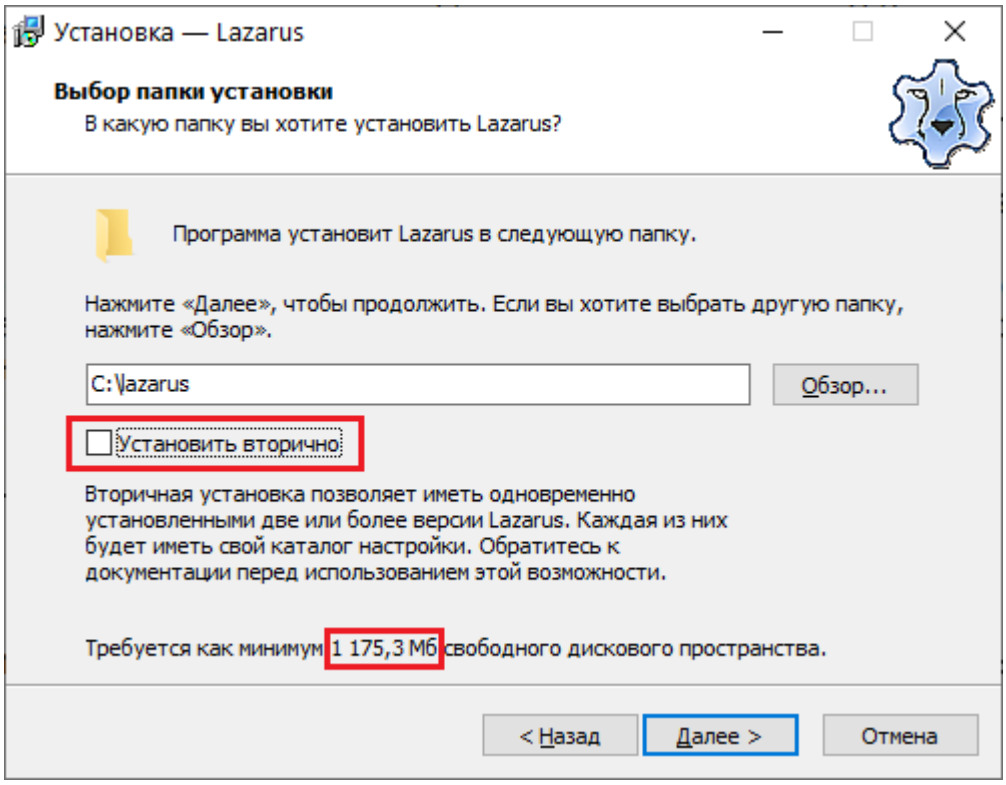


Рис. 1-6. Выбор каталога установки.

Я советую оставить каталог по умолчанию (`C:\lazarus`). Почему? Потому что если вы выберете свой каталог, то при установке более новой версии вы можете забыть про это и таким образом у вас будет две копии Lazarus. Причём в новой версии все настройки придётся делать заново.

Также обратите внимание на галочку “Установить вторично”. Я не советую выбирать эту опцию (по крайней мере сейчас, пока вы только начинаете изучать Lazarus). Но если вы её выберите, то в случае, если у вас уже установлена Lazarus, новая версия будет также установлена отдельно (то есть вы сможете пользоваться и новой, и старой версиями) и для новой версии также придётся все настройки устанавливать заново.

Впрочем, если вы устанавливаете Lazarus в первый раз, то всё это особого значения не имеет.

Когда вы определитесь с каталогом установки, нажмите кнопку ДАЛЕЕ.

Если Lazarus у вас уже была установлена в выбранном каталоге, то появится такое окно:

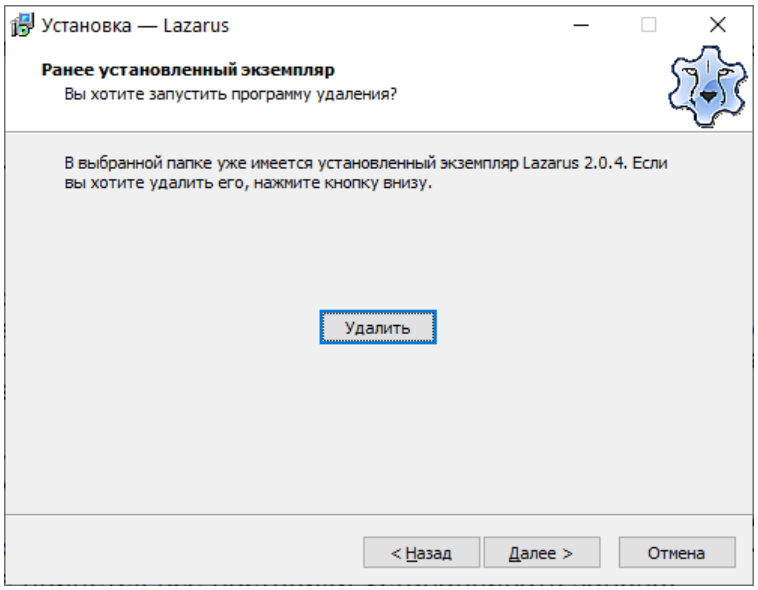


Рис. 1-7. Запрос на удаление существующей версии Lazarus.

Если вы хотите, чтобы все настройки существующей версии Lazarus у вас сохранились, то старую версию удалять не надо - просто нажмите кнопку ДАЛЕЕ. Если же вы хотите стереть все воспоминания о старой версии, то нажмите кнопку УДАЛИТЬ.

Я никогда не удаляю старые версии, чтобы сохранить настройки и не тратить время на настройку новой версии (хотя иногда всё равно приходится).

Следующее окно – выбор компонентов.

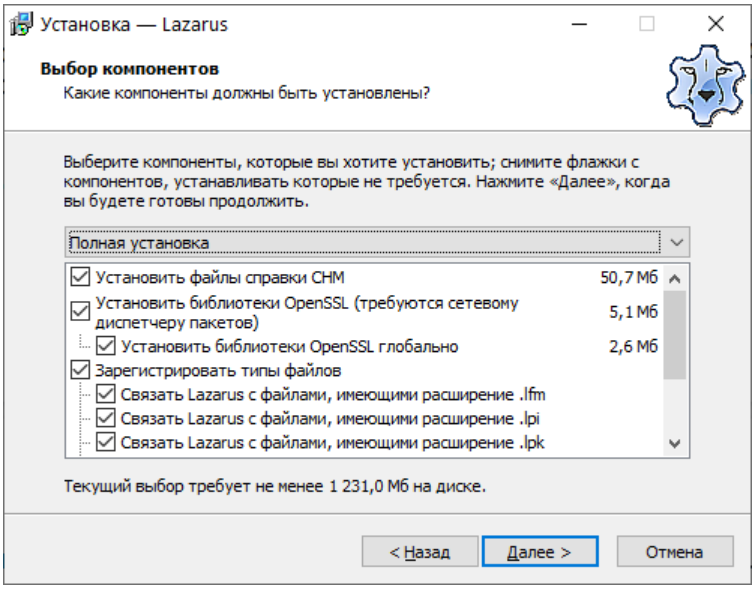


Рис. 1-8. Выбор компонентов для установки.

Я советую выбрать всё (по умолчанию так и есть). Затем опять нажимаем ДАЛЕЕ.

Появится окно, где вы можете выбрать ещё кое-какие опции:

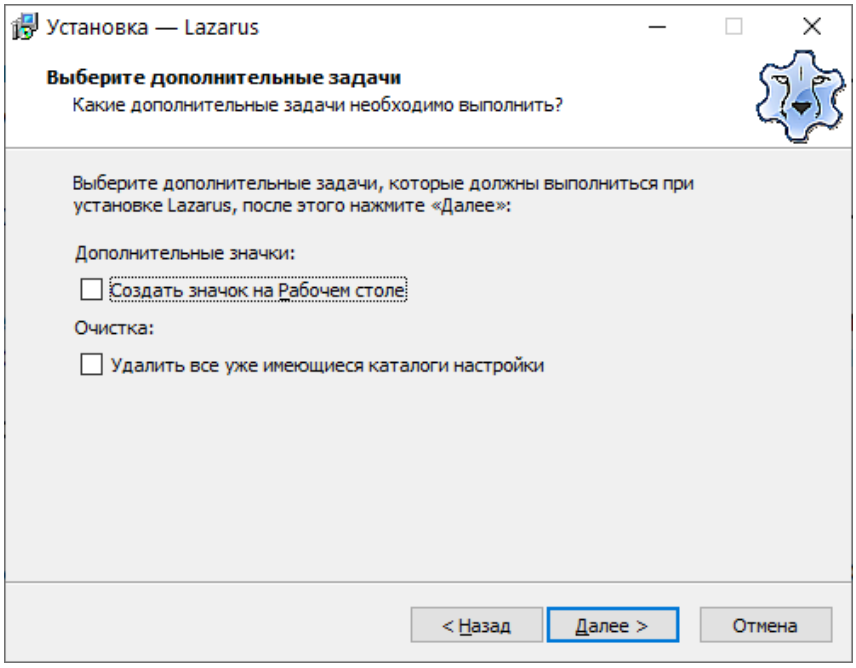


Рис. 1-9. Дополнительные возможности установки.

Я обычно не создаю ярлык на рабочем столе, и имеющиеся каталоги настройки также не удаляю. То есть оставляю всё по умолчанию – обе галочки НЕ установлены.

Ну и наконец появляется окно проверки параметров установки:

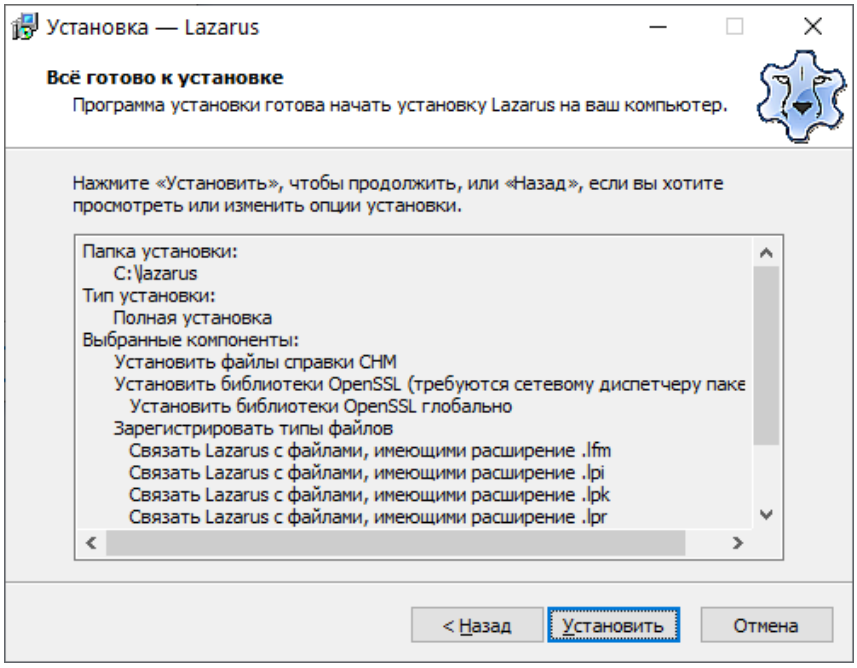


Рис. 1-10. Проверка параметров установки.

Здесь вы можете проверить все настройки установки, и если что-то не так, то вернуться назад и исправить. Если же всё правильно, то нажимаем кнопку УСТАНОВИТЬ. Начнётся установка, ход которой будет отображаться в следующем окне:

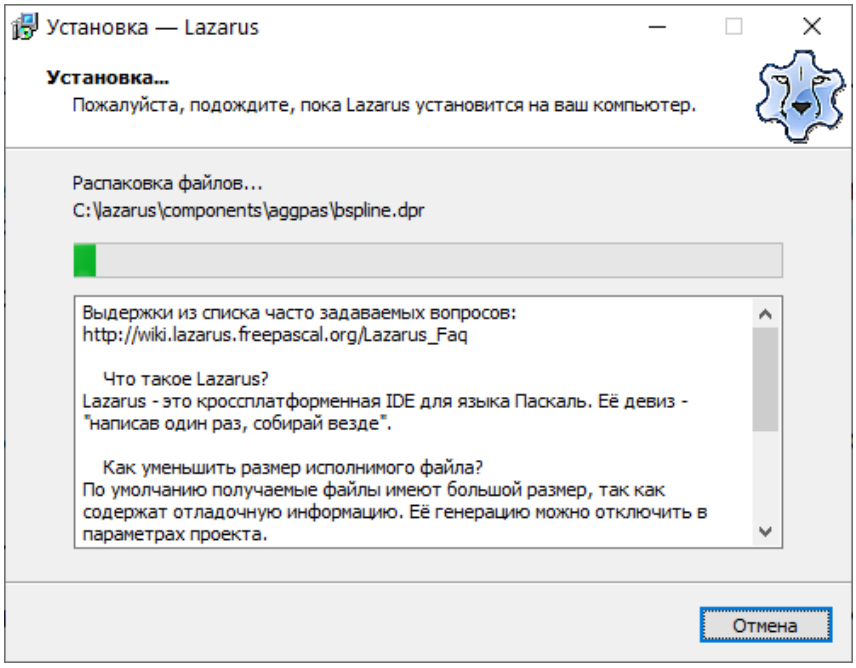


Рис. 1-11. Установка выполняется.

В зависимости от быстродействия вашего компьютера, установка может занять продолжительное время. Именно поэтому (чтобы сократить время установки) перед установкой программ советуют закрыть все другие программы.

Если вам нечем заняться в ходе установки, можете почитать краткие сведения о Lazarus в окне установки. Когда установка наконец закончится, появится последнее окно:

Рис. 1-12. Завершение установки.

Здесь просто нажимаем кнопку ЗАВЕРШИТЬ.

Ну вот и всё. Среда разработки установлена и готова к работе. Теперь запускаете её как обычную программу – если поставили галочку создания ярлыка, то ярлык найдёте на рабочем столе. Если нет, то в меню ПУСК (см. рис. 1.13).

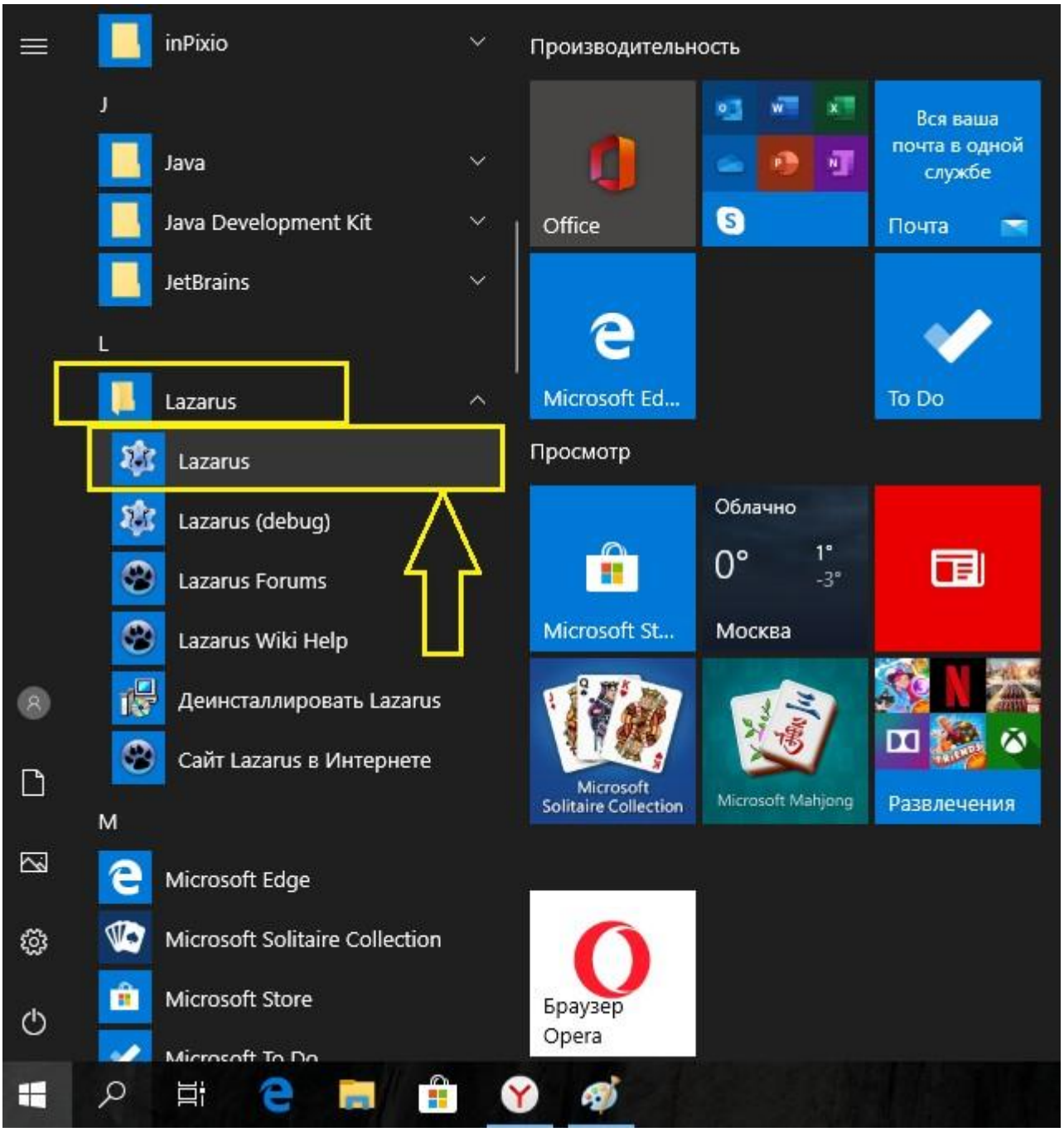


Рис. 1-13. Ярлыки в меню ПУСК.

Запускаем среду разработки и видим примерно следующее:

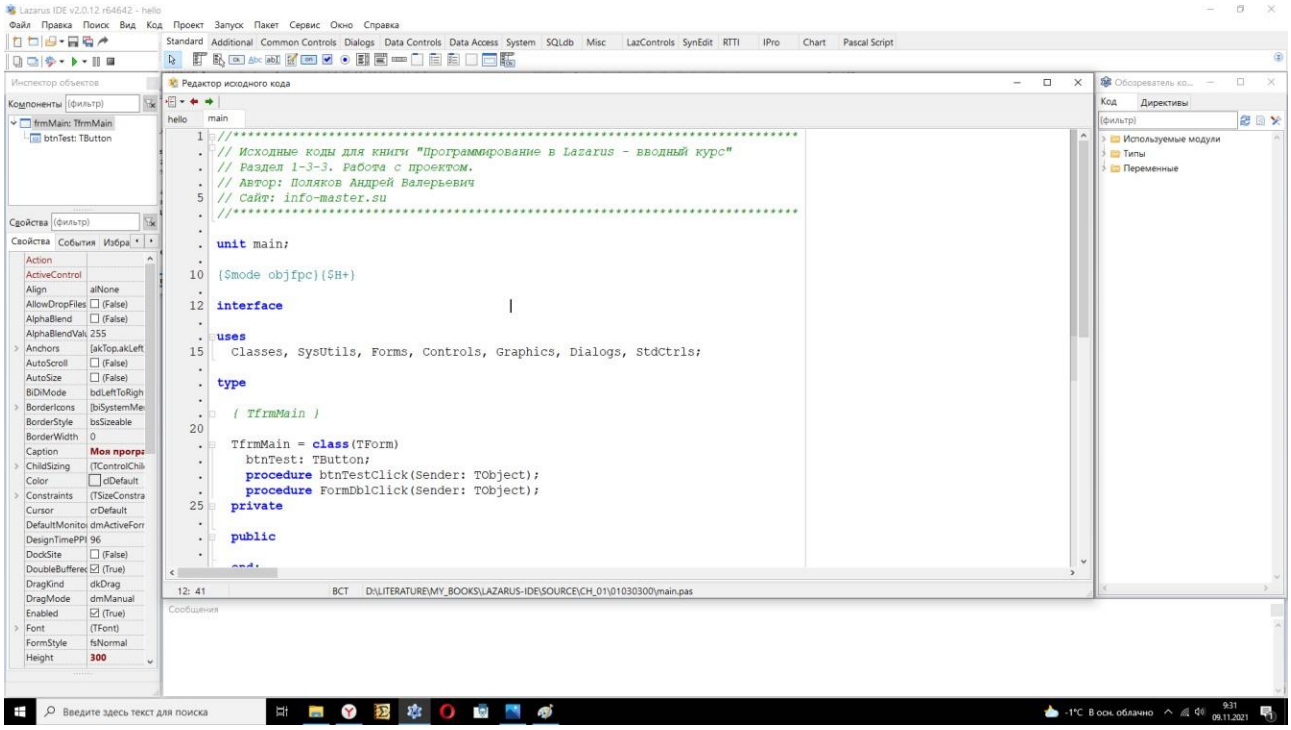


Рис. 1-14. IDE Lazarus.

Если у вас уже была установлена Lazarus, то может появиться такое окно:

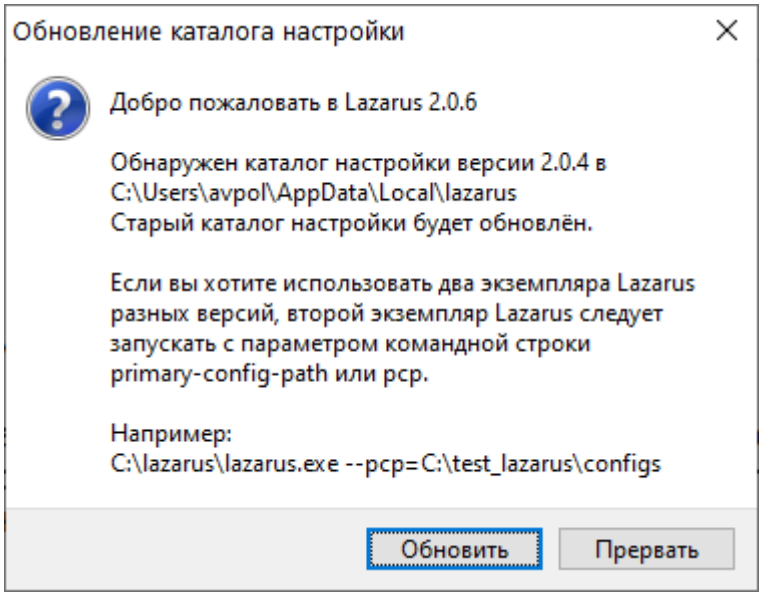


Рис. 1-15-1. Запрос на обновление каталога настроек.

Здесь нажмите ОБНОВИТЬ.

Если же это первая установка или установка в новый каталог, то может появиться такое окно:

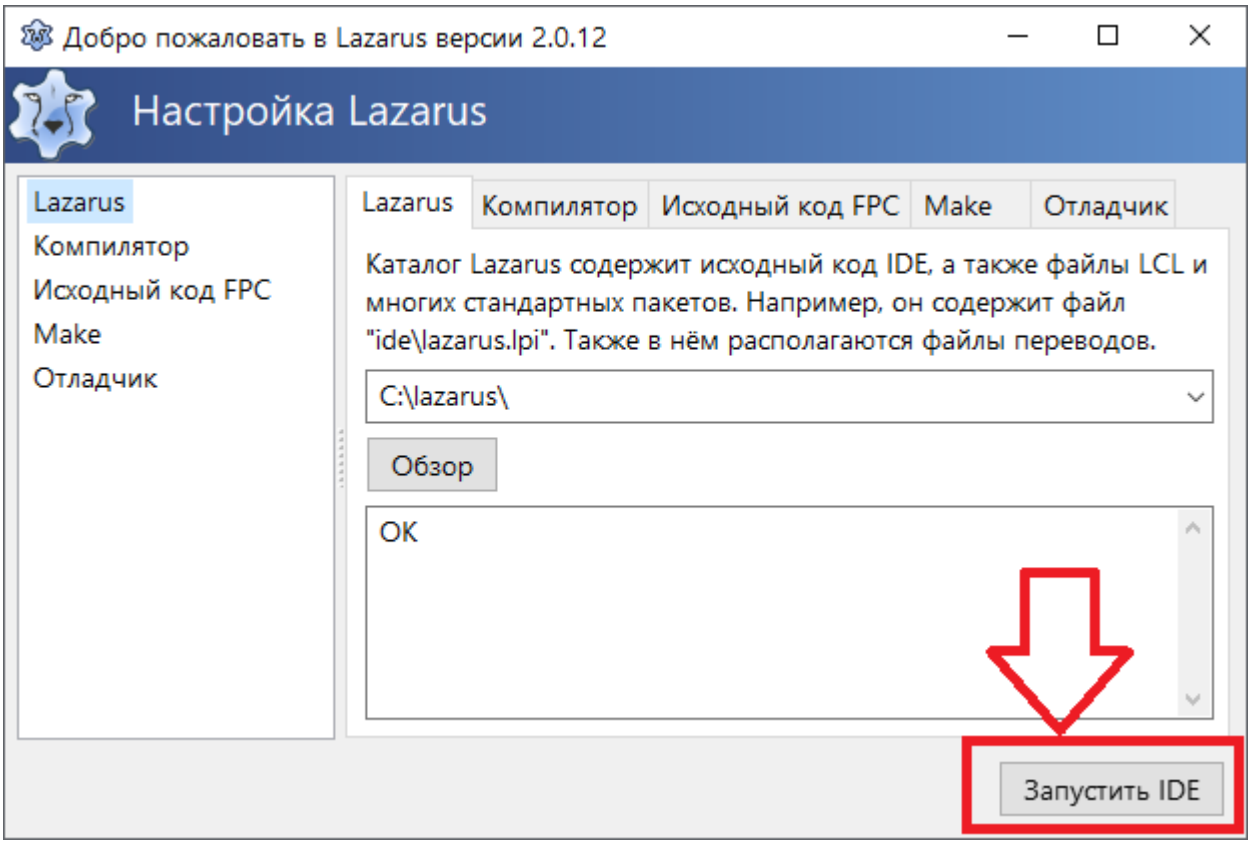


Рис. 1-15-2. Окно предварительной настройки.

Здесь пока просто нажмите ЗАПУСТИТЬ IDE, ничего не изменяя в настройках.

Полезные ссылки:

- Антивирус - бессмысленный и беспощадный

Ну а теперь создадим нашу первую программу…

1-3. Создание и компиляция проекта

Видеоприложение здесь: `VIDEO\01030000 Создание и компиляция проекта.mp4`
Исходные коды к этому разделу находятся здесь: `SOURCE\CH_01\01030200`

Прежде чем вы создадите свою первую программу, позволю себе дать вам несколько советов.

Во-первых, создайте на диске отдельную папку для ваших программ. У меня это папка `MYPROG`, вы можете придумать своё имя.

Во-вторых, для каждой программы создавайте отдельный каталог.

В-третьих, не желательно присваивать папкам имена, в которых содержатся русские символы.

Ну и напоминаю, что исходные коды всех программ, которые приводятся в этой книге, вы можете найти в каталоге `SOURCE`.

А теперь переходим к делу…

1-3-1. Создание проекта

Когда вы запускаете среду разработки, то обычно в неё автоматически загружается последняя программа, над которой вы работали, или создаётся новый проект. Но мы пройдём путь с нуля. Поэтому после запуска Lazarus закроем все открытые файлы через меню:

`ФАЙЛ - ЗАКРЫТЬ ВСЁ`

Если вас спросят о сохранении, то откажитесь (я исхожу из того, что вы впервые запускаете Lazarus).

Теперь создадим новый проект. Делается это также через меню:

`ФАЙЛ - СОЗДАТЬ`

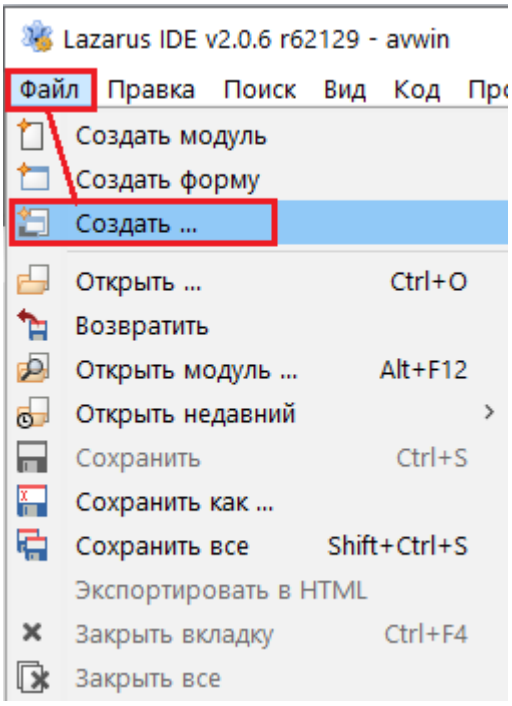


Рис. 1-16. Создание нового проекта.

Откроется окно, где надо выбрать вид программы. В нашем случае это приложение с графическим интерфейсом. Поэтому в группе ПРОЕКТ выбираем ПРИЛОЖЕНИЕ и нажимаем кнопку ОК (рис. 1-17).

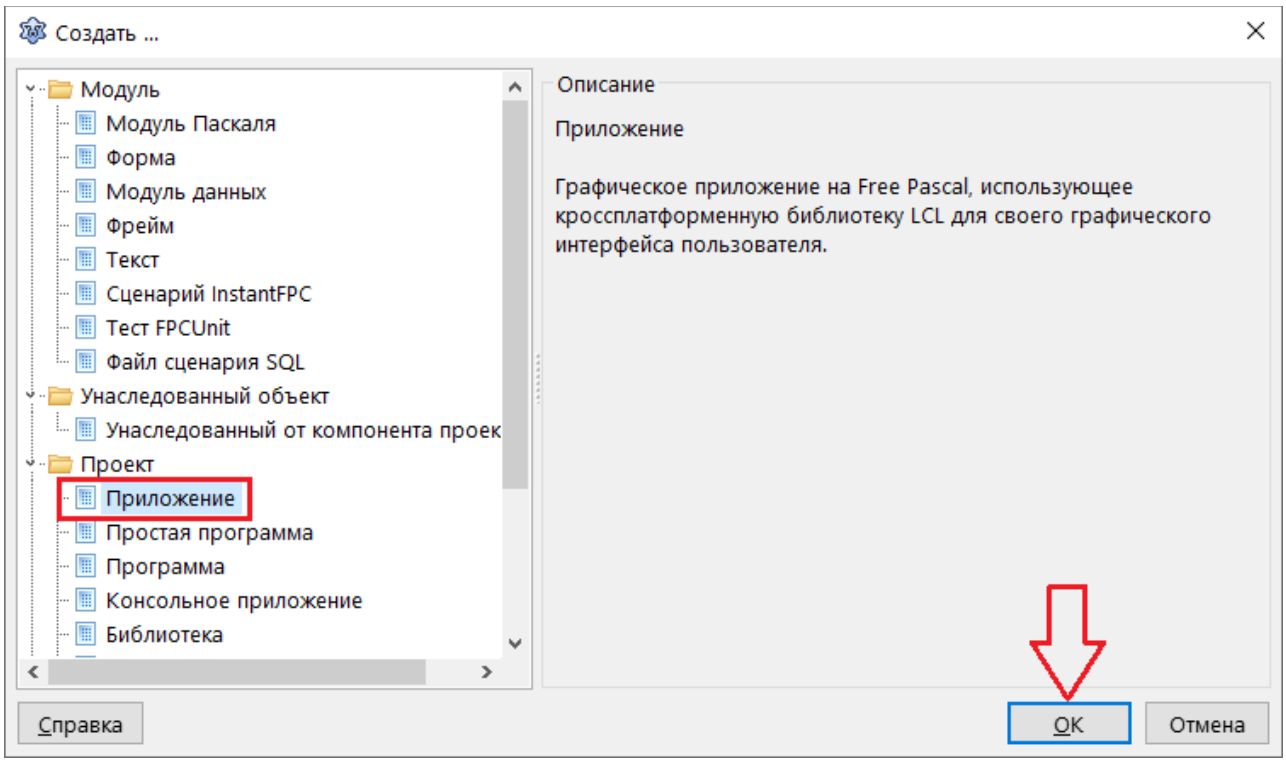


Рис. 1-17. Выбор вида приложения.

Будет создано приложение с пустой формой. Но проект пока не сохранён. Поэтому первым делом его надо сохранить. Вопрос – куда сохранять? Но, я надеюсь, вы внемли моим советам и уже создали отдельную папку для ваших программ, а в этой папке отдельную папку для вашей первой программы.

Сохраняем проект через меню

`ФАЙЛ - СОХРАНИТЬ`

или просто нажимаем `CTRL + S`.

Проект состоит из нескольких файлов. Большинство из них создаются и сохраняются автоматически. Однако для двух файлов вы должны указать имена (имена не должны содержать русских символов).

Первый файл – это файл проекта с расширением `LPR`, второй файл – файл модуля главной формы с расширением `PAS`. Вообще форм в проекте может быть много, но в любом проекте должна быть хотя бы одна форма.

Какое имя вы зададите для проекта, такое имя будет и у вашей программы. Например, если файл вашего проекта будет называться `MYPROG.LPR`, то файл вашей программы будет `MYPROG.EXE`.

Первое окно сохранения, которое появится после того, как вы выберите первый раз команду меню ФАЙЛ - СОХРАНИТЬ, это как раз и будет окно сохранения файла проекта:

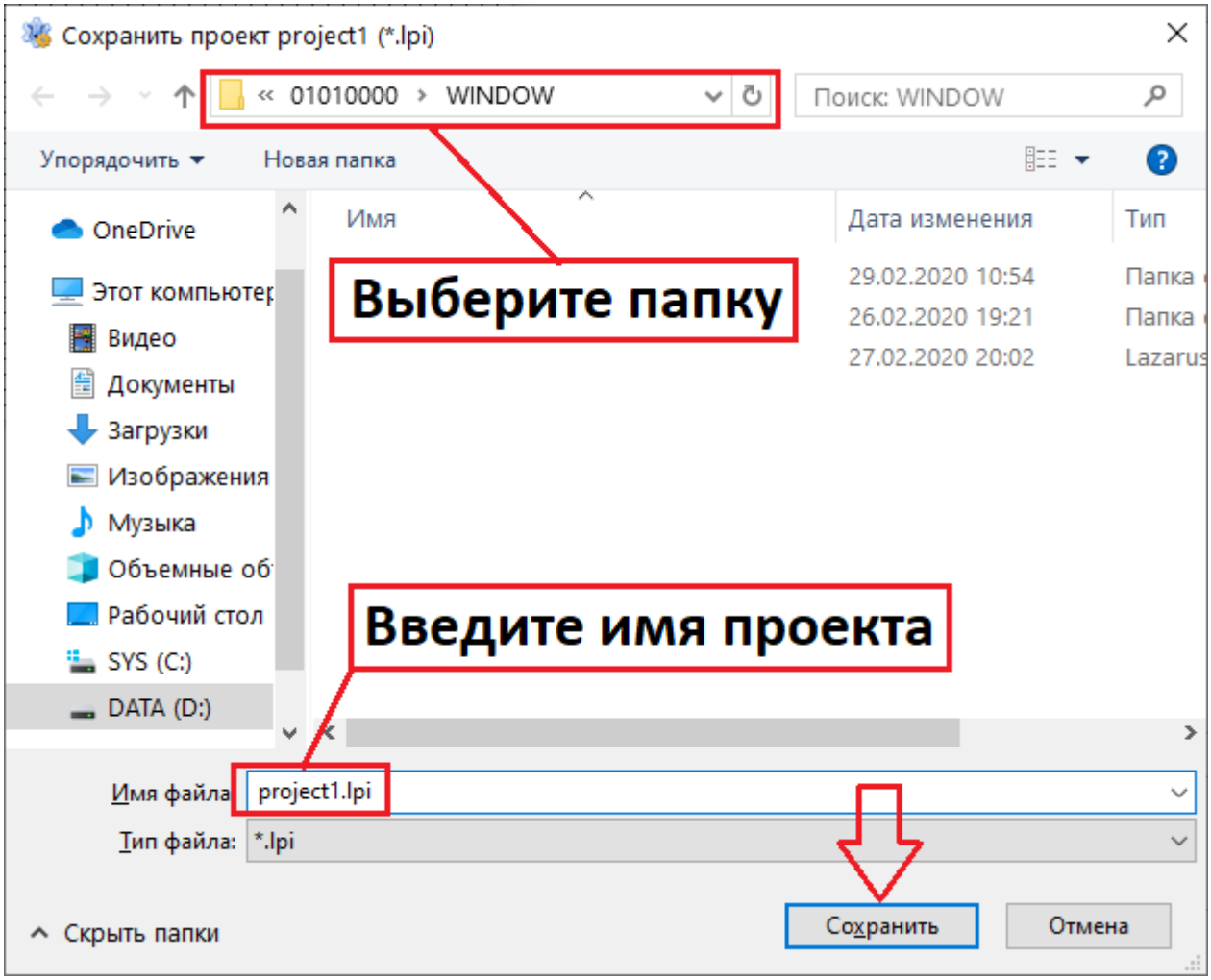


Рис. 1-18. Окно сохранения файла проекта.

В этом окне сначала выберите папку, куда надо сохранить проект. Затем введите имя проекта и нажмите СОХРАНИТЬ.

ВНИМАНИЕ!

Менять можно только имя файла проекта, но нельзя менять или удалять расширение.

Давайте первый наш проект сделаем точно так, как это сделал я. То есть назовём его

```
hello.lpr
```

А следующие ваши проекты уже будете называть так, как пожелаете.

Итак, вы выбрали папку для вашего проекта и придумали для него имя. Теперь нажимаем СОХРАНИТЬ и видим ещё одно окно сохранения файла.

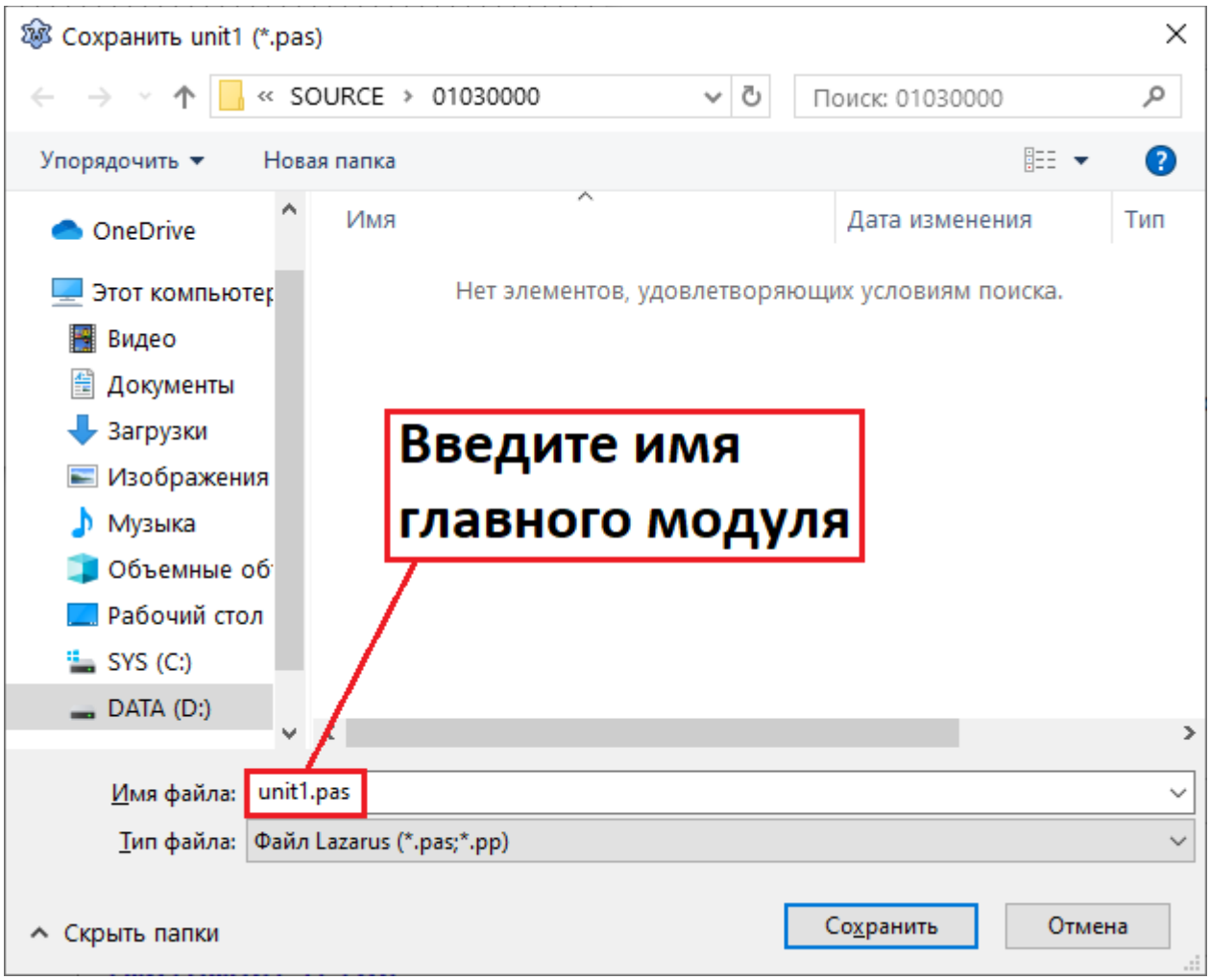


Рис. 1-19. Окно сохранения файла главной формы.

На этот раз мы сохраняем модуль главной формы. Каталог выбирать вам уже не надо, поскольку вы выбрали его при сохранении файла проекта. Вы также можете оставить имя по умолчанию, но я обычно всегда присваиваю модулям какие-то осмысленные имена. А модулю главной формы я всегда присваиваю имя

```
main.pas
```

Ну всё, наш проект сохранён. И это уже готовая программа, хотя мы с вами не написали ещё ни одной строчки кода. Но эту программу уже можно откомпилировать и запустить. Чем мы и займёмся в следующем разделе.

1-3-2. Компиляция и сборка

Исходные коды к этому разделу находятся здесь: `SOURCE\CH_01\01030200`

На тот случай, если вы уже закрыли проект и Lazarus, расскажу, как можно открыть уже созданный проект для просмотра или доработки.

Итак, есть несколько способов. Я использую обычно следующий – захожу в папку с исходными кодами проекта и дважды щёлкаю по файлу проекта (это файл с расширением `LPI` или `LPR`). При этом Lazarus автоматически запускается и загружает проект.

Правда, у Lazarus есть небольшой глюк, который переходит из версии в версию. Заключается он в том, что проект загружается не всегда. Если это произошло, и проект не загрузился, то ничего страшного. Надо просто загрузить его вручную.

Сделать это можно двумя способами. Первый через меню:

`ФАЙЛ – ОТКРЫТЬ НЕДАВНИЙ`

и выбрать нужный файл из списка (см. рис. 1.20).

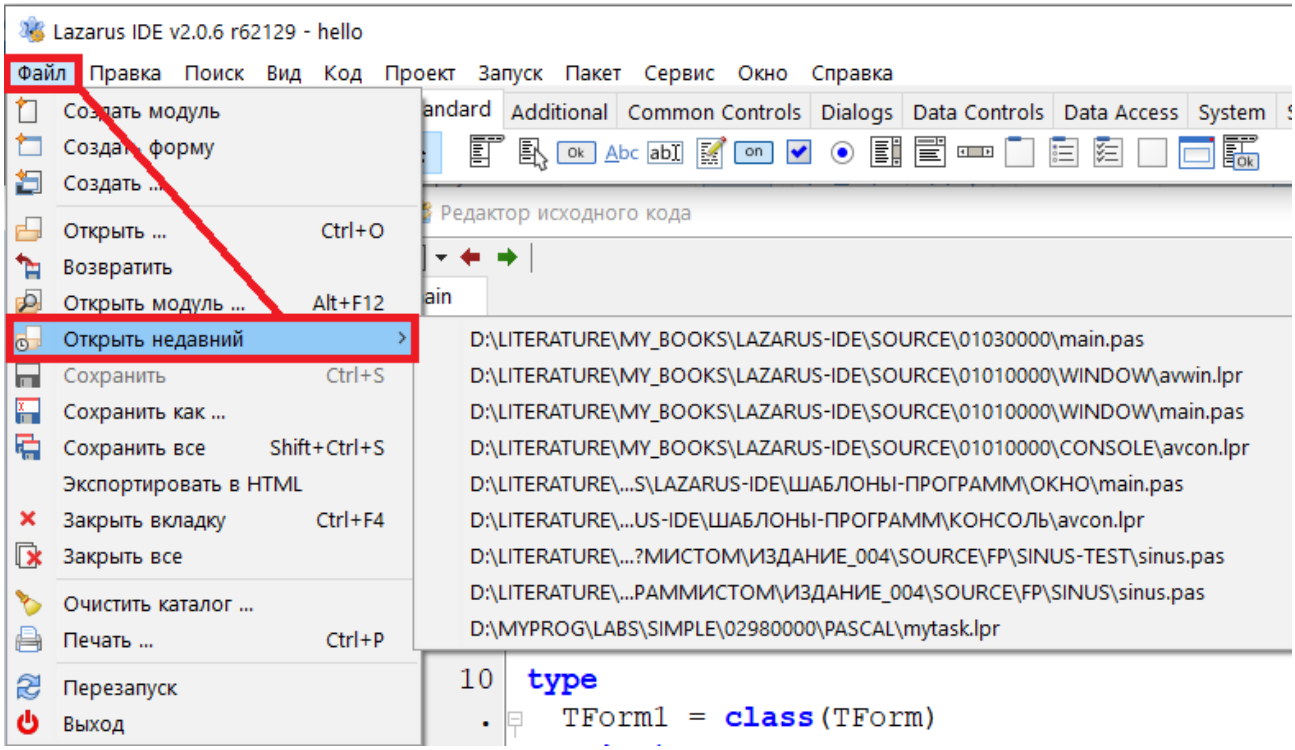


Рис. 1-20. Список недавних проектов.

Если нужного файла в списке не оказалось, тогда действуем иначе:

`ФАЙЛ – ОТКРЫТЬ`

В диалоговом окне выбора файла заходим в папку с вашим проектом, выбираем файл исходного кода проекта (это файл с расширением `LPR`) и щёлкаем по кнопке ОТКРЫТЬ.

В предыдущем разделе мы создали новый проект с именем `hello.lpr`. Вот его и откроем.

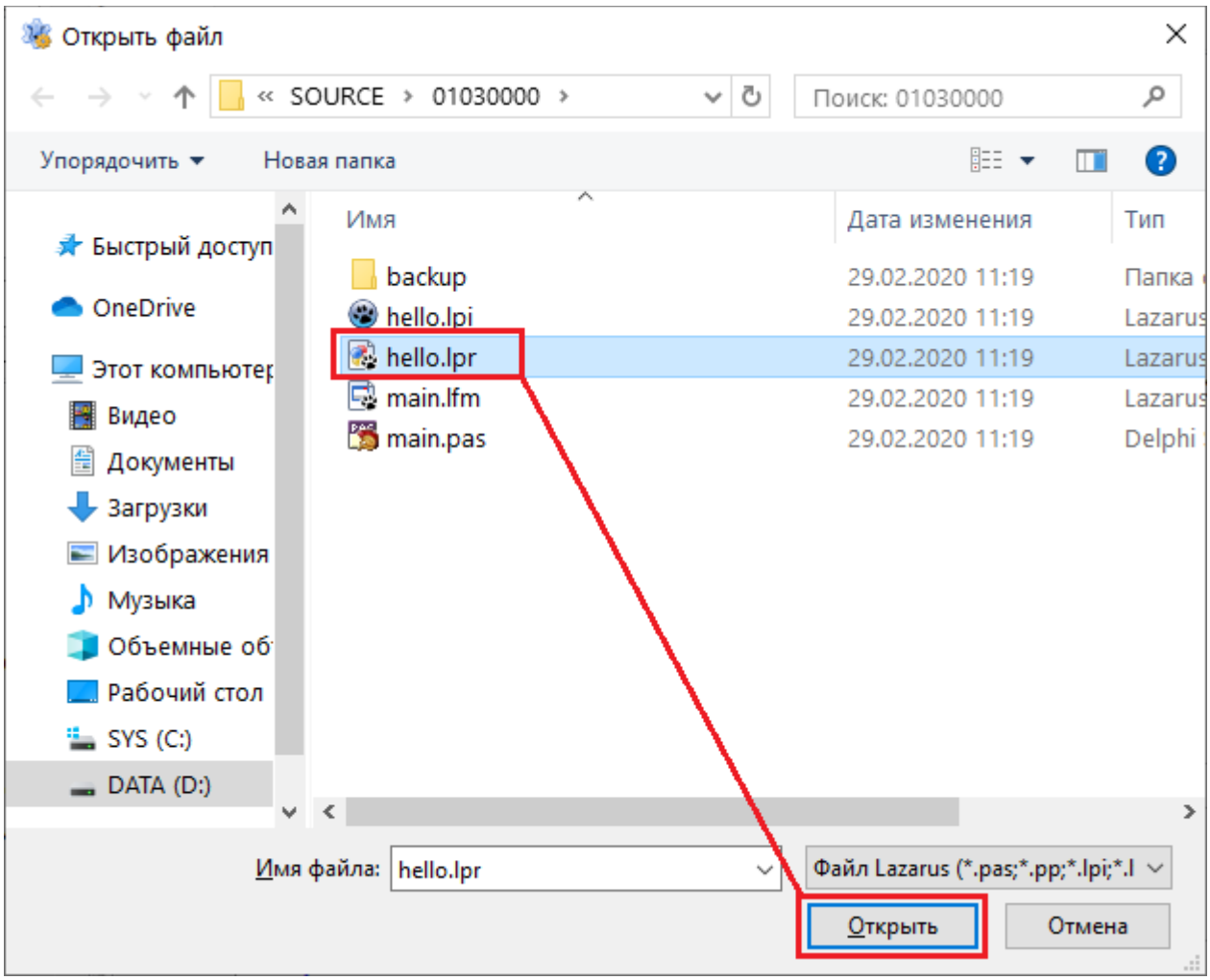


Рис. 1-21. Открытие проекта .

А теперь создадим уже наконец рабочую программу. Потому что пока у нас есть только исходные коды. Для этого надо скомпилировать и собрать проект.

Что такое компиляция и что такое сборка, надеюсь, вы знаете. Если нет, то напомню.

Компиляция – это преобразование исходных кодов программы, написанных на каком-либо языке программирования, в набор машинных кодов (набор команд) процессора. Некоторые компиляторы переводят исходные коды не в машинные коды, а в исходный файл на языке ассемблера. В итоге создаётся один или несколько файлов с машинными кодами (или ассемблерный файл).

Компиляция в Lazarus выполняется через меню:

`ЗАПУСК - КОМПИЛИРОВАТЬ`

или комбинацией клавиш `CTRL + F9`.

В итоге, если компиляция завершилась успешно (об этом вам скажут в окне сообщений), то в папке вашего проекта появится новый файл с расширением EXE – ваша первая программа готова!

А что такое сборка?

Если компиляция – это создание исполняемого файла (программы) из исходных кодов, то **сборка** – это (по замыслу) создание готового программного продукта для конечного пользователя.

Но, честно говоря, чем компиляция отличается от сборки в Lazarus, я так и не понял. Сборка выполняется через меню:

`ЗАПУСК - СОБРАТЬ`

или комбинацией клавиш `SHIFT + F9`.

Если вы хотите скомпилировать проект и сразу запустить вашу программу из IDE для проверки или отладки, то это можно сделать через меню:

`ЗАПУСК - ЗАПУСТИТЬ`

или клавишей `F9`.

СОВЕТ

- Если вы хотите скомпилировать и запустить вашу программу – используйте клавишу `F9`.
- Если же вы не хотите запускать программу, а вам нужно лишь скомпилировать её и проверить исходные коды на ошибки – используйте комбинацию клавиш `CTRL + F9`.

Ну что же, в нашем проекте мы пока не написали ни одной строчки кода, поэтому ошибок в нём нет и он прекрасно скомпилировался. И это ваша первая программа с графическим интерфейсом. Можете запустить её и проверить, что она работает – вы можете перемещать форму, свернуть её, развернуть на весь экран и закрыть.

И выглядит она примерно так:

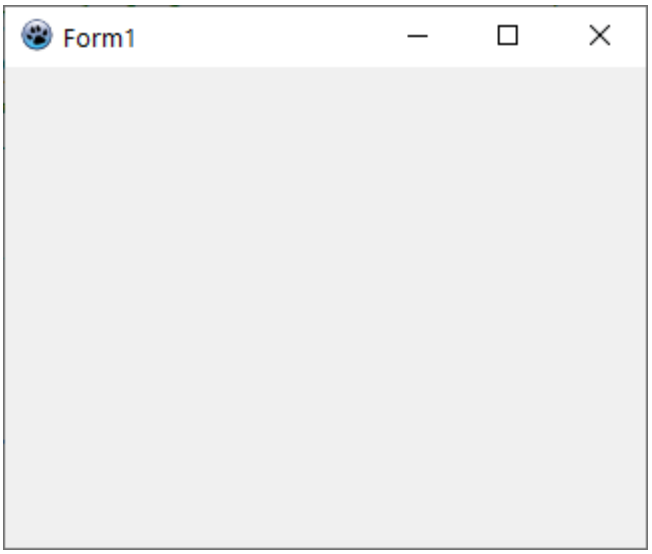


Рис. 1-22. Наша первая программа с графическим интерфейсом.

А теперь попробуем сделать что-нибудь с этой программой…

1-3-3. Работа с проектом

Видеоприложение здесь: `VIDEO\01030300 Работа с проектом.mp4`
Исходные коды к этому разделу находятся здесь: `SOURCE\CH_01\01030300`

Очень краткий ввод в основы создания программ в IDE Lazarus.

Программы создаются следующим образом:

1. На форму набрасываются в нужные места графические элементы управления, такие как кнопки, надписи, списки, таблицы и т.п.
2. Каждый компонент является объектом, у которого есть свойства и методы (надеюсь, вы уже прочитали **книгу об объектно-ориентированном программировании**). Поэтому для компонентов (при необходимости) прописываются методы и устанавливаются свойства.

Вот, в общем-то, и всё.

Например, у кнопки есть такие свойства, как ширина и высота, а также надпись на кнопке.

И у кнопки есть события, такие как нажатие (щелчок мышью по кнопке). Эти события обрабатываются автоматически – программисту ничего делать не надо (за него всё это уже сделали разработчики Lazarus – в этом и прелесть визуальных средств разработки). Программисту остаётся только написать код, который будет выполняться при наступлении этих событий. Вот этим мы сейчас и займёмся. И начнём с главной формы…

1-3-3-1. Задаём имя, заголовок и размер формы

У формы довольно много свойств, которые можно изменять как программно (то есть в исходных кодах), так и в инспекторе объектов.

В инспекторе объектов лучше задавать те свойства, которые не изменяются по ходу выполнения программы. Например, размеры (хотя в некоторых случаях изменять размеры тоже требуется именно программно).

А в исходных кодах лучше изменять свойства, значения которых меняются по ходу выполнения программы. Например, когда ваша программа выводит текущее время.

Итак, у нас есть проект `hello.lpr`. Его то мы и помучаем немного.

Чтобы изменять в инспекторе объектов свойства формы или элементов на форме, надо, чтобы форма была видна на экране.

Здесь как раз пришло время узнать, как переключаться между редактором исходного кода и графической частью проекта. Делать это можно либо через меню:

`ВИД – ПЕРЕКЛЮЧИТЬ ФОРМУ/МОДУЛЬ`

либо клавишей `F12`. То есть, если вы не видите форму на экране, то выберите модуль формы вверху (см. рис. 1.23) и нажмите клавишу `F12`.

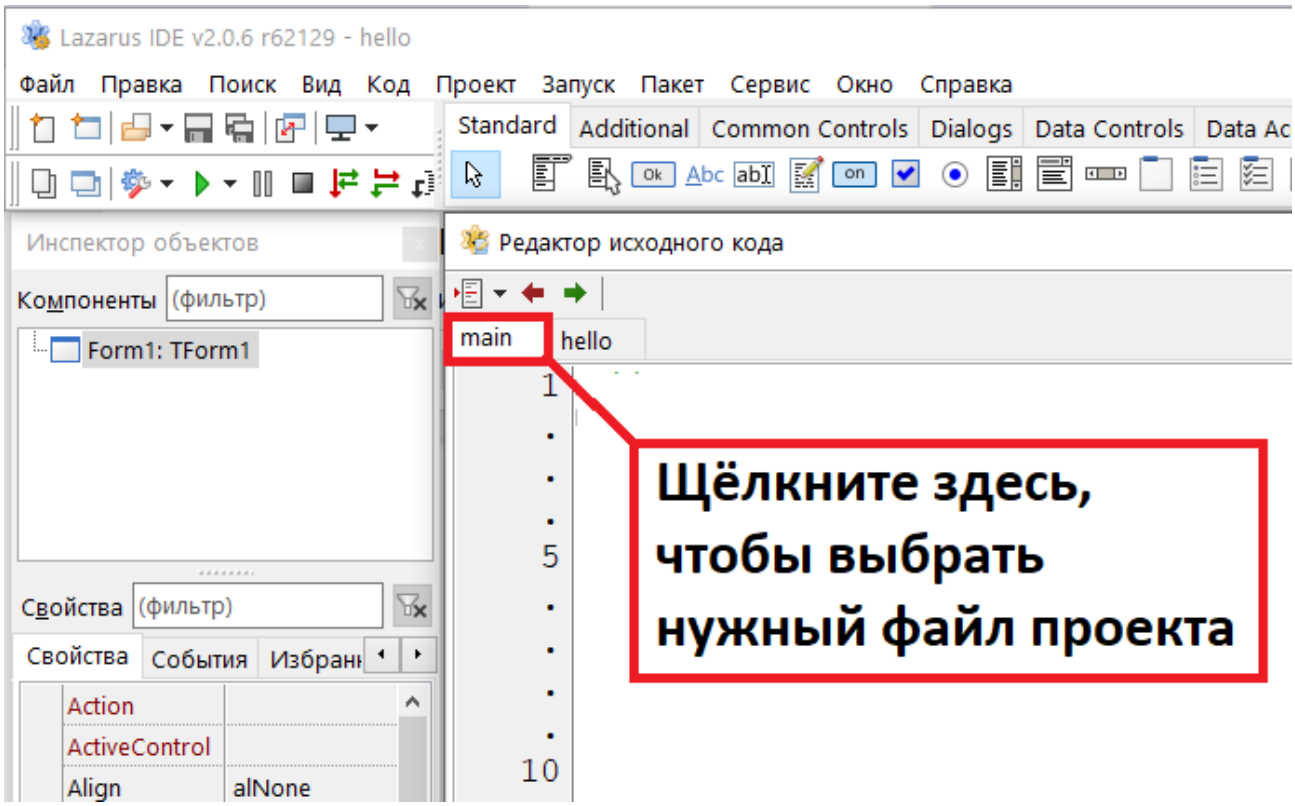


Рис. 1-23. Переключение между файлами проекта.

Теперь, когда форма появилась на экране, можно изменять её свойства. Свойства выбранного элемента (это может быть необязательно форма, но и элемент на форме) отображаются в инспекторе объекта.

Щёлкаем по форме, чтобы быть уверенными, что в инспекторе объектов будут именно свойства формы. Затем будем искать нужные нам свойства и изменять их.

Первым делом я советую изменить имя формы – имя переменной, с которой связана форма. Для этого в инспекторе объектов ищем свойство `Name`. Щёлкаем левой кнопкой мыши справа от имени свойства (где пока написано `Form1`), и пишем там имя (русские буквы не допускаются).

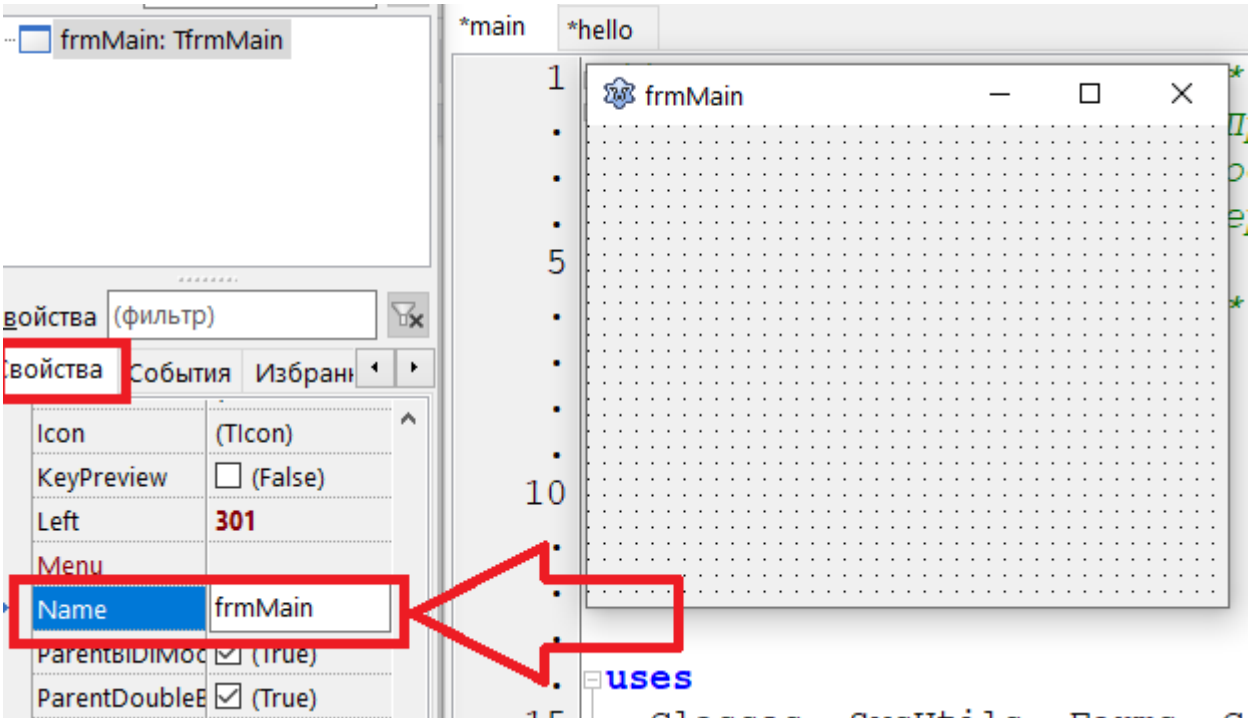


Рис. 1-24. Изменение имени формы.

Я написал здесь `frmMain`. Советую в вашем первом проекте написать то же имя, чтобы не запутаться. А вообще вы можете задавать любые имена, которые соответствуют правилам именования идентификаторов в данном языке программирования (в нашем случае это Паскаль).

Когда напишите имя – нажмите ENTER, чтобы применить изменения. Также советую сохранять проект после изменений.

Теперь с нашей главной формой связана переменная `frmMain`, которую можно использовать в программе для вызова методов формы и изменения её свойств.

Вы наверно обратили внимание на то, что заголовок формы изменился автоматически, а в исходном коде имя переменной также изменилось автоматически. Если не успели увидеть – нажмите `F12`, чтобы перейти в редактор исходного кода – там вы в разделе объявления переменных найдёте строку:

```
frmMain: TfrmMain;
```

А теперь снова вернитесь в визуальную часть (нажмите `F12`), чтобы наша форма была видна на экране. Сейчас будем менять заголовок.

Найдите в инспекторе объектов свойство `Caption` и присвойте ему какое-нибудь значение также, как мы это делали для имени формы. Здесь можно писать по-русски. Я напишу `“Моя программа”`.

И вы увидите, что заголовок формы изменился. Можете скомпилировать и запустить программу, и убедиться, что теперь у формы другой заголовок.

Вот так! Мы до сих пор не написали ни одной строчки кода, а программа наша изменяется и дорабатывается!

Ну и для закрепления успехов давайте изменим размеры формы.

Ширина – это свойство `Width`, а высота – `Height`. Предлагаю сделать нашу форму шириной 400 и высотой 300 пикселей. Забейте эти значения в соответствующие свойства, и вы увидите, что размеры формы изменились.

Мы научились изменять свойства формы в инспекторе объектов. Прекрасно! Можно переходить к обработке событий.

1-3-3-2. Обработка событий

У формы довольно много видов событий. Событиями являются, например, щелчок по форме, изменение размера, закрытие окна и т.п. Сейчас я расскажу вам, как обрабатывать такие события. Но для начала давайте перейдём в редактор исходного кода (`F12` – больше напоминать не буду), и в разделе объявления переменных добавим новую переменную:

```
var
  frmMain     : TfrmMain;
  strCaption : String;    //Это мы добавили
```

Теперь делаем следующее:

1. Переходим к работе с формой (`F12`) и для верности щёлкаем один раз по форме, чтобы быть уверенными, что в инспекторе объектов именно форма.
2. В инспекторе объектов переходим на вкладку СОБЫТИЯ.
3. Находим событие `OnDblClick`.
4. Дважды щёлкаем левой кнопкой мыши справа от имени события.

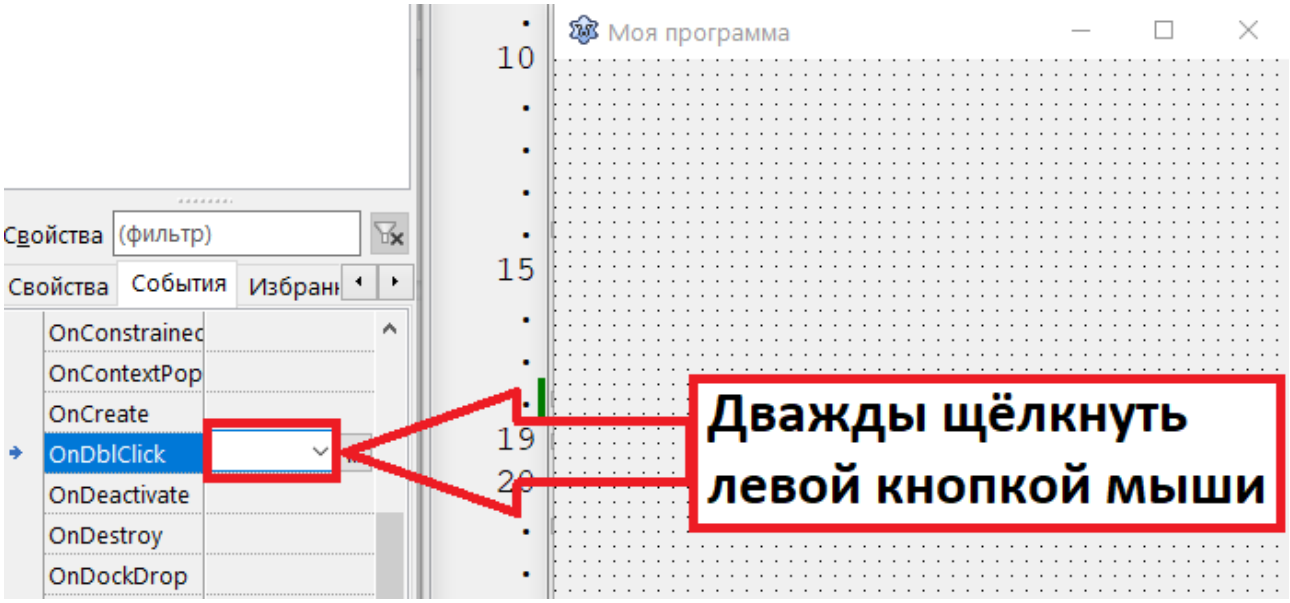


Рис. 1-25. События формы.

При этом вы автоматически перейдёте в редактор исходного кода в обработчик этого события, который также будет создан автоматически.

Это будет процедура, которая будет вызываться при наступлении выбранного события:

```
procedure TfrmMain.FormDblClick(Sender: TObject);
begin


end;
```

И в теле этой процедуры мы должны написать свой исходный код.

И теперь если пользователь дважды щёлкнет по форме (а это и будет нужное нам событие – двойной щелчок мышью – `OnDblClick`), то будет вызвана и выполнена процедура `TfrmMain.FormDblClick`.

А теперь давайте напишем уже какой-нибудь обработчик события. Например:

```
procedure TfrmMain.FormDblClick(Sender: TObject);
const STR = 'Ура!!! Заработало!!!';
begin
  if frmMain.Caption <> STR then
    strCaption := frmMain.Caption;
  if frmMain.Caption = STR then
    frmMain.Caption := strCaption
  else
    frmMain.Caption := STR;
end;
```

Здесь мы проверяем, что написано в заголовке формы. Если это первый щелчок по форме, то содержимое заголовка запоминаем в глобальную переменную `strCaption`.

И в любом случае, если в заголовке у нас `'Ура!!! Заработало!!!'`, то мы меняем заголовок на первоначальное значение (в нашем случае это `Моя программа`). И наоборот.

Но, надеюсь, что вы уже имеете представление о программировании на Паскале и такие простые вещи пояснять не надо. А вот если вы впервые сталкиваетесь с визуальной средой разработки, то обратите внимание на то, как мы изменяем в программе свойство **`Caption`** формы **`frmMain`**:

```
frmMain.Caption := 'Какое-то значение'
```

То есть сначала мы записываем имя переменной **`frmMain`**, связанной с объектом (в нашем случае это главная форма приложения), затем ставим точку, а затем пишем имя свойства (в нашем случае это **`Caption`**).

Ну а как выполняется присваивание в Паскале, думаю, вы знаете.

Ещё раз советую (если вы этого ещё не сделали) изучить книгу по объектно-ориентированному программированию (ООП). Потому что в данной книге про ООП я рассказывать не буду.

1-3-3-3. Добавляем элемент управления

Вы, конечно, знаете, что у программы с графическим интерфейсом обычно есть много разных элементов управления, таких как кнопки, списки, флажки, поля ввода данных и т.п.

В Lazarus эти элементы представлены в виде компонентов, которые можно помещать на форму, а дальше уже связывать с ними события (и обработчики событий), а также читать и изменять их свойства (по тому же принципу, как мы это делали с формой в предыдущем разделе). Таким образом и создаются программы.

Компоненты бывают видимыми (визуальными) и невидимыми (невизуальными). Но об этом мы будем говорить подробно в следующих разделах.

Пока же достаточно знать, что все доступные компоненты расположены на панели компонентов в верхней части IDE под главным меню:

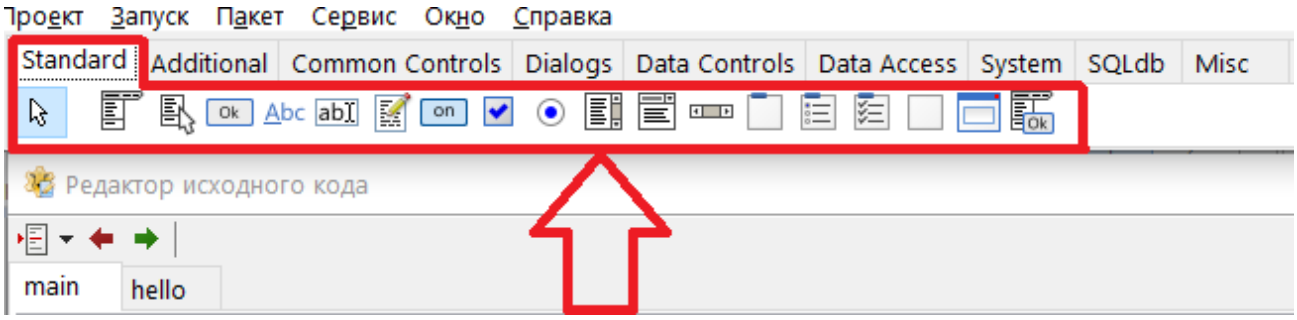


Рис. 1-26. Панель компонентов.

Компоненты разделены на несколько групп. На рисунке показаны компоненты группы STANDARD. Чтобы отобразить компоненты другой группы, надо щёлкнуть по заголовку соответствующей вкладки.

Чтобы поместить компонент на форму, надо сначала щёлкнуть левой кнопкой мыши по нужному компоненту на панели компонентов, а потом по форме в том месте, где вам нужен этот компонент.

Изменить положения компонента можно путём перетаскивания мышью, либо путём изменения его координат в инспекторе объектов (свойства `Left` и `Top`).

Изменить размер компонента можно также с помощью мыши (наведя курсор на край компонента, нажав левую кнопку и перемещая мышь), либо путём изменения его ширины и высоты в инспекторе объектов (свойства `Width` и `Height`).

Ну и понятно, что это также можно делать программно, присваивая свойствам нужные значения.

А теперь попробуем сделать всё это на примере кнопки.

1. Ищем на панели компонентов в группе STANDARD маленький прямоугольник с надписью OK.
2. Щёлкаем по нему левой кнопкой мыши.
3. Переводим курсор мыши на форму.
4. Щёлкаем ещё раз.

Теперь на форме появилась кнопка с надписью `Button1`.

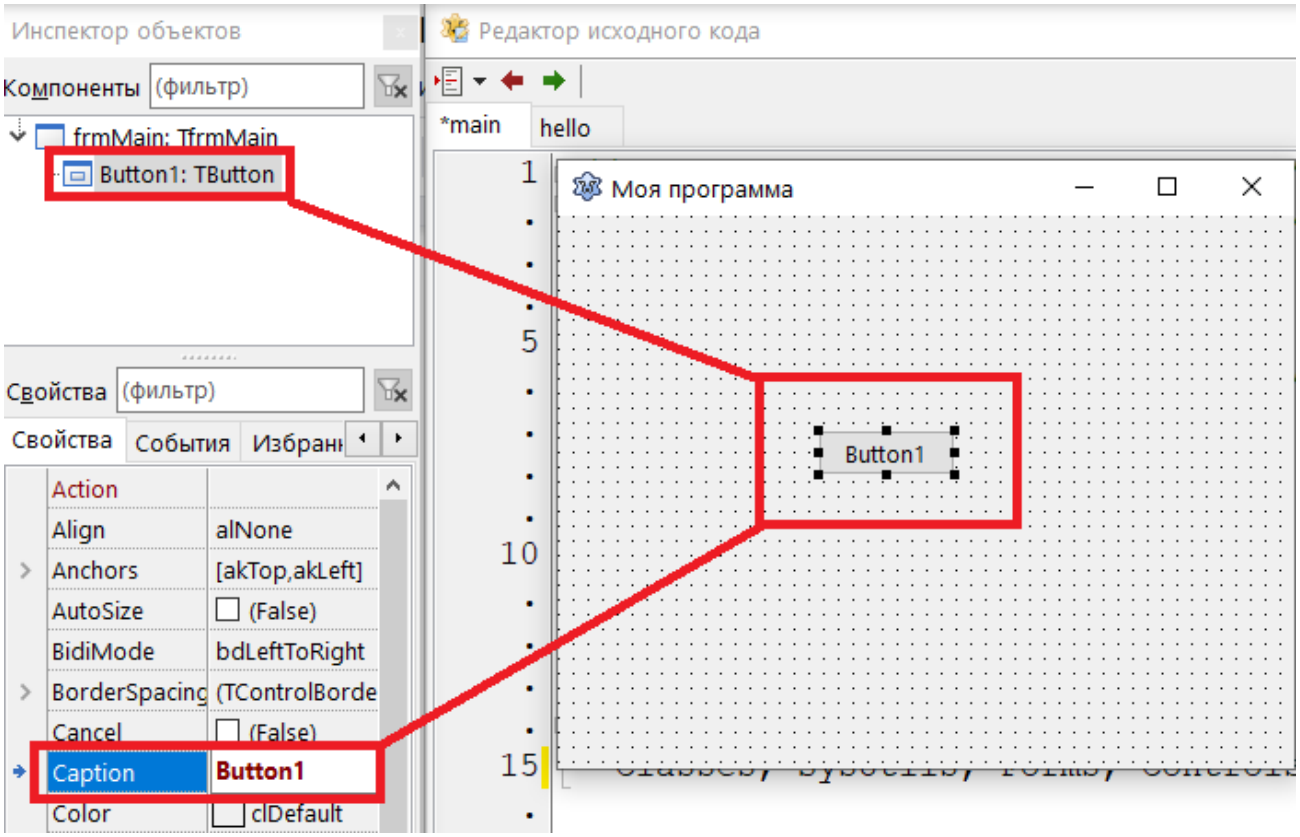


Рис. 1-27. Кнопка на форме.

Чтобы дальше выполнять какие-то действия с кнопкой, надо, чтобы она была выбрана. О том, что компонент выбран, свидетельствуют чёрные квадратики по границе компонента, а также выделение компонента в списке компонентов инспектора объектов (см. рис. 1.27).

Если кнопка не выбрана, то щёлкаем по кнопке на форме или по имени компонента в верхнем окне инспектора объектов.

Теперь изменим имя кнопки и надпись на кнопке. Это делается также, как и для формы. Свойству `Caption` присвоим значение `Нажми меня!`, а свойству `Name` - значение `btnTest`. Увеличим размеры кнопки, чтобы надпись полностью поместилась на ней.

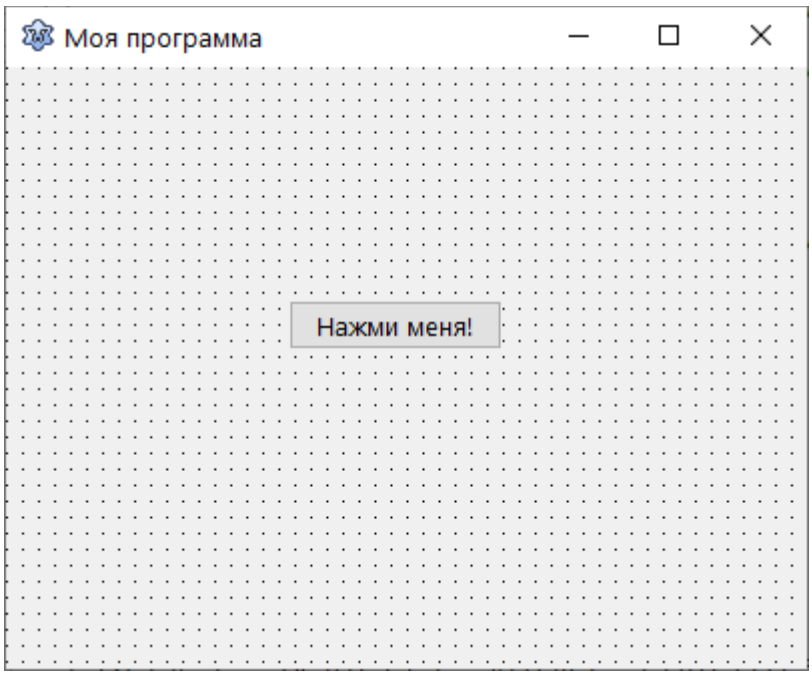


Рис. 1-28. Работаем с кнопкой.

А теперь давайте создадим обработчик события для кнопки. Пусть будет одинарный щелчок левой кнопки мыши – `OnClick`. Вы можете сделать это также, как и для формы, а можете просто дважды щёлкнуть по кнопке. Двойной щелчок по компоненту создаёт обработчик события по умолчанию для компонента. Для кнопки это событие `OnClick`. У других компонентов могут быть другие события по умолчанию.

В обработчике события напишем следующее:

```
procedure TfrmMain.btnTestClick(Sender: TObject);
begin
  ShowMessage('Кнопка работает!!!');
  btnTest.Visible := FALSE;
end;
```

Здесь мы сначала вызываем стандартное диалоговое окно с надписью `'Кнопка работает!!!'`, а затем делаем кнопку невидимой.

Причём невидимой кнопка станет только после того, как вы закроете диалоговое окно. Поскольку окно вызывается в модальном режиме, то программа (точнее, обработчик события) приостанавливает свою работу до тех пор, пока это окно открыто.

Проверьте всё это и попробуйте сами придумать что-нибудь. А мы на этом быстрый ввод в программирование в Lazarus завершим.

В следующем подразделе ещё немного об основных настройках среды разработки, а в следующей части уже будем изучать Lazarus более серьёзно…

1-3-4. Основные настройки среды разработки

Видеоприложение здесь:
`VIDEO\01030400 Основные настройки среды разработки.mp4`

Любой профессионал должен в совершенстве владеть инструментами, которые он использует в своей работе. Основной инструмент программиста – это среда разработки.

Но любой инструмент должен не просто выполнять своё предназначение – он ещё должен быть удобным и приятным в использовании.

И поэтому всегда, как только я устанавливаю новую среду разработки, я первым делом её настраиваю под себя. В этом разделе я кратко расскажу о настройках, которые я изменяю в первую очередь. А более подробно о настройках среды разработки и настройках проекта я буду говорить в основной части книги.

Итак, первым делом я всегда настраиваю подсветку синтаксиса. Потому что я привык именно к таким настройкам и любые другие цвета меня сильно бесят.

Подсветка синтаксиса – это отображение разных языковых элементов (таких как ключевые слова, комментарии и т.п.) разным цветом и стилем текста.

В Lazarus окно настроек вызывается через меню:

`СЕРВИС - ПАРАМЕТРЫ`

В появившемся окне надо найти группу РЕДАКТОР. В этой группе есть подгруппа ОТОБРАЖЕНИЕ, где вы можете выбрать шрифт редактора исходного кода и его размер. Я выбираю шрифт `Courier New`, а размер шрифта обычно устанавливаю равным 14 или даже 16, поскольку давно сижу за монитором и зрение уже ни к чёрту.

В подгруппе ОТОБРАЖЕНИЕ также есть несколько подгрупп. Нам нужна подгруппа ЦВЕТА (см. рис. 1.29).

Когда вы щёлкните по подгруппе ЦВЕТА в левой части окна, в правой части отобразятся текущие настройки подсветки синтаксиса. Вы можете изменить их по своему усмотрению. Например, если вы хотите изменить цвет зарезервированных слов, таких как `program`, `var`, `const`, `begin`, `end` и т.п., то щёлкните в списке по надписи "**abc** Зарезервированное", а затем установите нужные вам цвет и стиль.

После того, как вы установите нужные вам настройки подсветки синтаксиса для всех элементов, нажмите кнопку ОК (справа внизу), чтобы сохранить настройки.

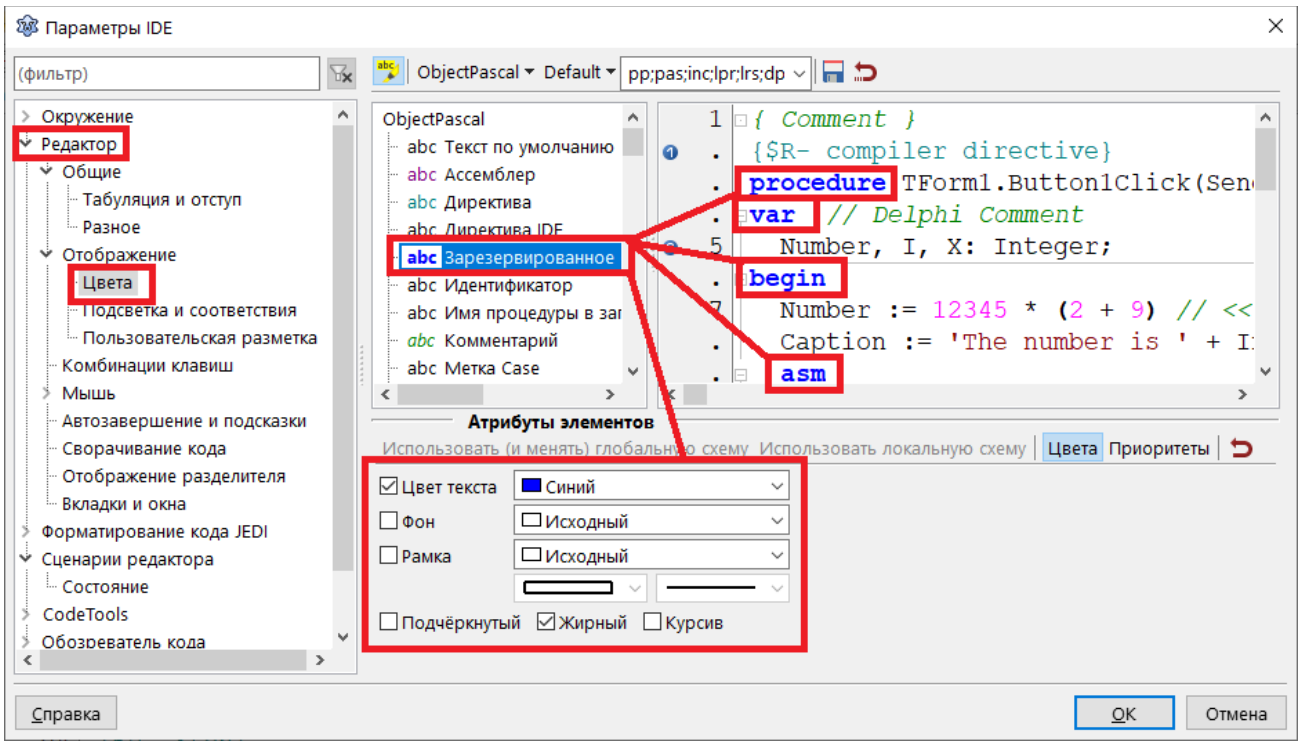


Рис. 1-29. Окно настроек подсветки синтаксиса.

1-3-4-1. Почему программа такая большая?

Видеоприложение здесь: `VIDEO\01030400 Почему программа такая большая.mp4`

Среда разработки Lazarus очень похожа на Делфи. Однако все мои знакомые программисты, которым по тем или иным причинам приходилось “пересаживаться” с Делфи на Lazarus, при компиляции первой же программы буквально охреневали от одной особенности Lazarus. А именно – от очень большого размера исполняемого файла (EXE-файла) откомпилированной программы (кстати, эта особенность в новых версиях Делфи также имеется).

Ну например, если графическое приложение, которое ничего не делает (пустое окно), после компиляции в Делфи занимает примерно 385 килобайт, то такое же приложение, созданное с помощью Lazarus, “весит” около 15 МЕГАбайт. То есть в 40 раз больше!

На самом деле всё не так страшно. Просто Lazarus по умолчанию “пихает” в EXE-файл отладочную информацию, которая и занимает так много места.

Избавиться от этого достаточно просто – надо в настройках проекта запретить включать в исполняемый файл отладочную информацию. Для этого выбираем меню:

```
ПРОЕКТ - ПАРАМЕТРЫ ПРОЕКТА
(или нажимаем SHIFT+CTRL+F11)
```

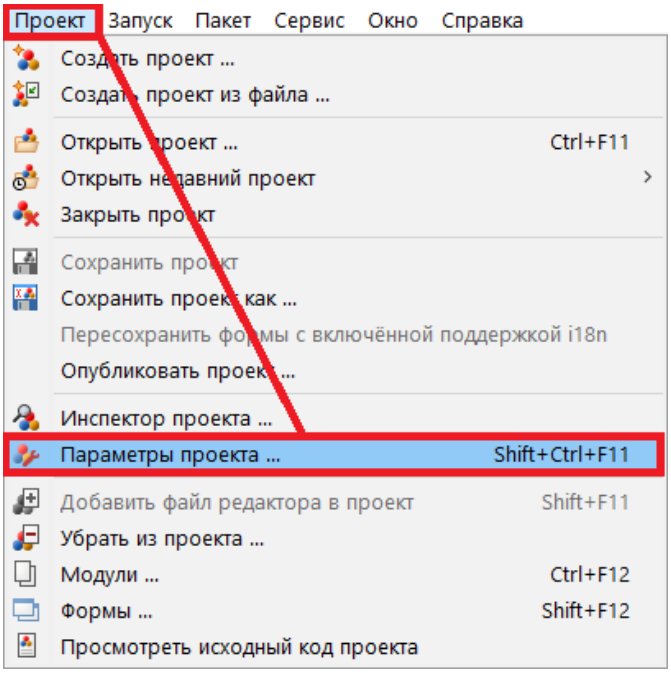


Рис. 1-30. Вызываем окно параметров проекта.

а в открывшемся окне в группе ПАРАМЕТРЫ КОМПИЛЯТОРА выбираем ОТЛАДКА. Здесь снимаем галочку у надписи "Генерировать отладочную информацию..." и нажимаем кнопку ОК (см. рис.).

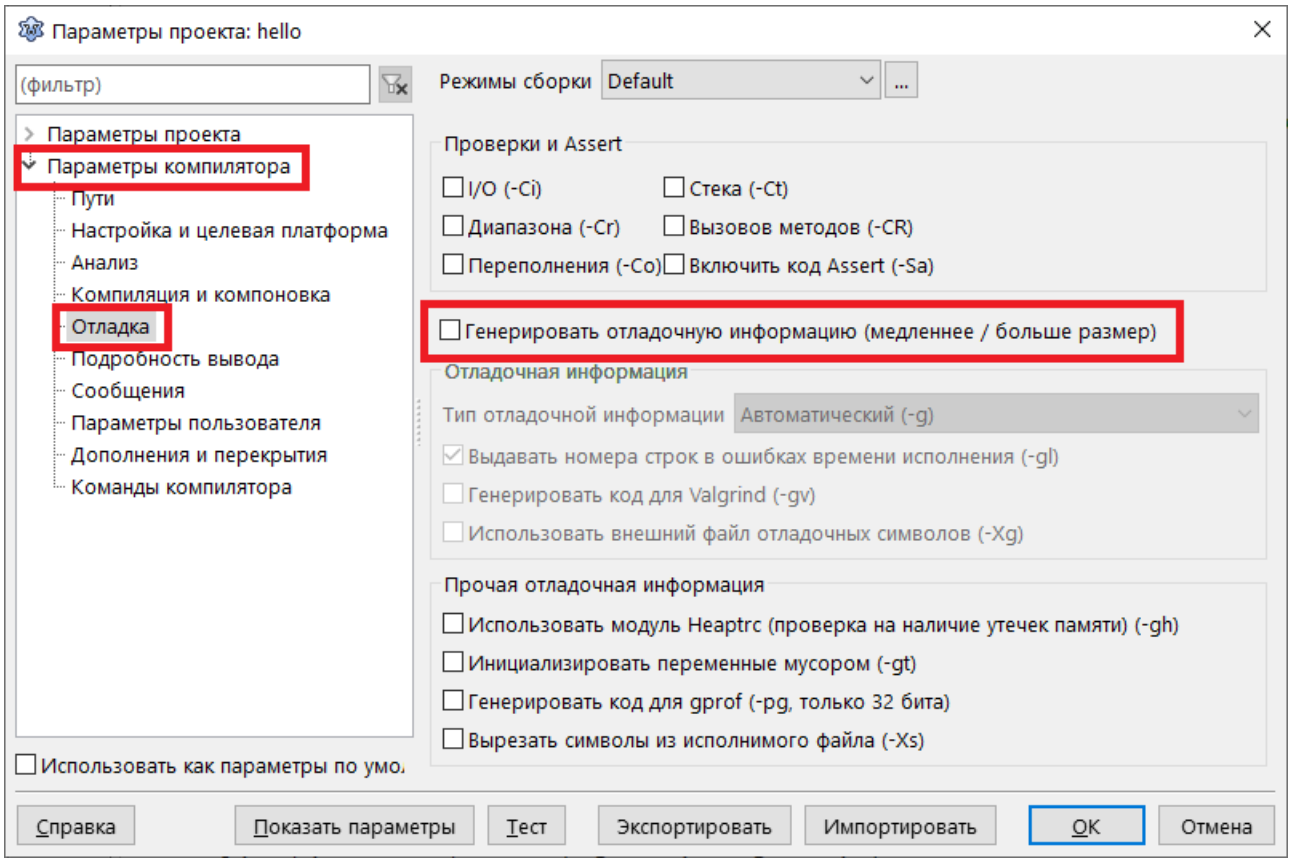


Рис. 1-31. Убираем отладочную информацию из исполняемого файла.

Ну вот и всё. Осталось только перекомпилировать вашу программу, и вы увидите, что размер исполняемого файла стал намного меньше. Правда, он всё равно будет немного больше, чем такая же программа, созданная в Делфи. Но это уже плата за бесплатность Lazarus.

ВНИМАНИЕ!

Таким образом вы отменяете генерацию отладочной информации ТОЛЬКО для данного проекта. При создании нового проекта по умолчанию отладочная информация БУДЕТ помещаться в EXE-файл. Чтобы её убрать, надо проделать все описанные выше шаги для нового проекта (и для всех остальных проектов, в которых отладочная информация вам не нужна).

1-4. Подведём итоги

Итак, мы теперь знаем, как создаются программы с графическим интерфейсом в Lazarus. На первый взгляд, всё очень просто. И для создания простых программок полученных знаний вам будет достаточно.

Однако, профессионал отличается от любителя отношением к мелочам. А “мелочей”, которые отличают любительскую программу от профессиональной, в любой среде разработки очень много.

Так что если хотите шагнуть чуть дальше, то вам всё-таки придётся продолжить обучение. Потому что раздел “Быстрый старт” – это лишь краткое знакомство.

Для профессионального программиста его будет достаточно. Потому что дальше он во всём разберётся сам с помощью справочной документации и собственного опыта.

Однако новичку разобраться самому будет очень сложно. Потому что новичок не знает и не может знать потенциальных возможностей этой среды разработки, поэтому он даже не сможет “начать думать, чтобы додуматься”. В итоге будет потрачено очень много времени, или, что ещё хуже, пропадёт запал и обучение будет вообще заброшено.

Так что если у вас запал ещё не пропал, то переходите к следующим разделам книги, где мы уже будем изучать Lazarus более тщательно и более подробно...

2. СРЕДА РАЗРАБОТКИ

В течение 6 часов мы будем прививать вам любовь к строевой подготовке
Из серии "Всепобеждающий армейский язык"

Среду разработки надо знать. Её изучение – это дело нудное, как строевая подготовка в армии, но столь же необходимое.

Те, кто служил в армии, могут возразить – какая польза в строевой подготовке? Ведь она не учит воевать!

А польза есть. Просто не все её видят. Строевая подготовка развивает выносливость, которая на войне очень даже необходима, потому что любую войну выигрывают не те, кто лучше вооружён, а те, кто более вынослив. Потому что большая часть действий в любой войне – это изнурительная работа.

Так и в любом деле: тяжело в учении – легко в бою.

Да, можно ринуться в драку с одним противником, если уверен в своих силах и если эти силы у тебя есть. Но если противников несколько, то здесь на силу уже рассчитывать не приходится. Здесь уже требуется умение вести бой.

К чему все эти лирические отступления?

Просто я хочу донести до вас простую мысль: хорошее знание среды разработки позволит вам более быстро и более легко создавать свои программы. Поэтому, потратив относительно немного времени на изучение и настройку среды разработки, в будущем сэкономите этого времени намного больше.

В этом разделе я расскажу об основных элементах и настройках среды разработки Lazarus. Советую, прежде чем перейти к разделу 3, где мы будем уже учиться создавать программы, хотя бы бегло прочитать раздел 2. А потом возвращаться к нему по мере необходимости.

Ну а сейчас начнём. И начнём с общего описания основных элементов интерфейса пользователя.

ОБ АВТОРЕ

На всякий случай представлюсь (вдруг кому интересно).

Это я - Поляков Андрей Валерьевич (можно просто Андрей))).

По образованию я инженер. Много лет отработал руководителем инженерного отдела в крупном (по меркам нашего региона) агрохолдинге. Но сейчас являюсь фрилансером.

Это основная деятельность.

Но есть у меня и другие интересы. Я пишу книги, создаю обучающие курсы. А также немного (на уровне любопытства) занимаюсь инфо-бизнесом. Все ссылки на мои контакты и проекты можно найти здесь:

http://info-master.su/contact.php