|  |  |
| --- | --- |
| Департамент образования Ярославской области  Государственное профессиональное образовательное автономное учреждение  Ярославской области  «ЯРОСЛАВСКИЙ ПРОМЫШЛЕННО-ЭКОНОМИЧЕСКИЙ КОЛЛЕДЖ  им. Н.П. Пастухова» | |
| **Отчет по УЧЕБНОЙ практике**  **ПМ.02 Осуществление интеграции программных модулей**  УП.02.09.02.07.19ИП2.20 | |
|  | Студент  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_А.С. Ткачук  «\_\_\_\_» мая 2022 г. |
| Руководитель практики  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Е.О. Куксов  «\_\_\_\_» мая 2022 г. |
| 2022 | |

# **Практическая работа №1.1**

**Тема**

Создание базы данных.

**Цель работы**

По заданной схеме данных разработать базу данных, создать и сохранить скрипт с БД.

**Пояснение к работе**

**Создание схемы данных.**

Была создана база данных с помощью «Диаграмм баз данных», вариант 5.1 (магазин).

Схема бд представлена на рисунке 1

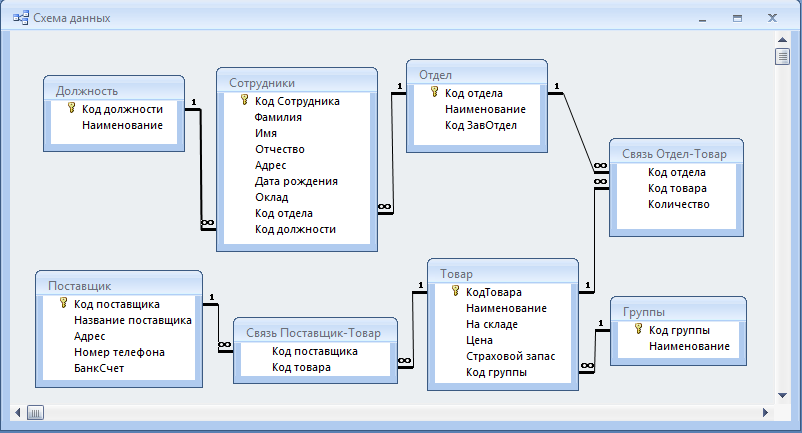
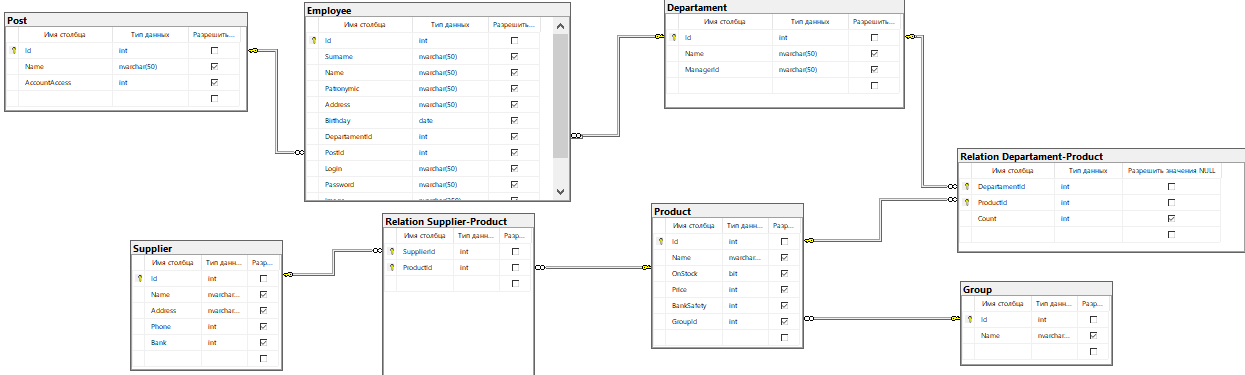


Диаграмма базы данных представлена на рисунке 1



В связях при обновлении и удалении данных из таблицы правило удаления и обновления выставлено “Каскадно”. Начальное значение первичного ключа единица, прибавляется на единицу.

Создание скрипта базы данных

Создать скрипт можно, кликнув ПКМ по базе данных, в появившемся контекстном меню выбрать Задачи > Сформировать скрипты. В новом окне жмем далее, потом выбираем нужные таблицы для скрипта, выбираем место для сохранения. Готово.

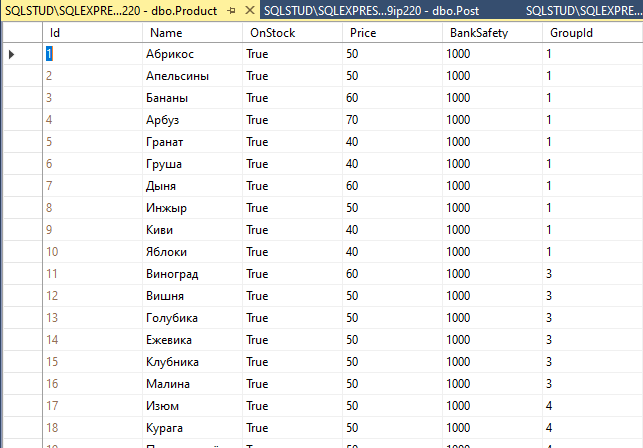
Часть скрипта представлено на рисунке 3



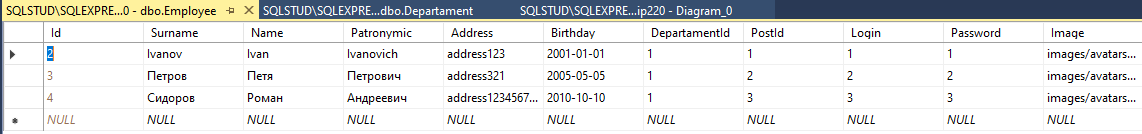
**Заполнение таблицы первичными данными**

**Для заполнения таблиц данными жмем ПКМ по нужной таблице и выбираем “изменить первые 200 строк”**

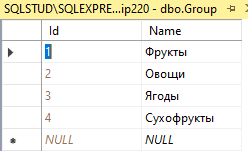
**Заполненные таблицы представлены на рисунках 4-8**



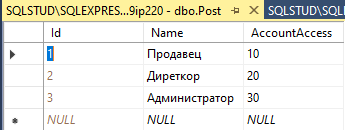
**Рисунок 4 Таблица “Продукты”**



**Рисунок 5 Таблица “Сотрудники”**



**Рисунок 6 “Виды продуктов”**



**Рисунок 7 “Должности”**

**Контрольные вопросы**

1. Что такое «база данных»?
2. Что такое первичный ключ (Primary key)?
3. Что такое внешний ключ (Foreign key)?
4. Какие связи существуют в реляционных базах данных?
5. Что подразумевается под целостностью данных?

Что такое нормализация и каковы ее преимущества?

**Практическая работа №1.2**

**Тема**

Подключение базы данных.

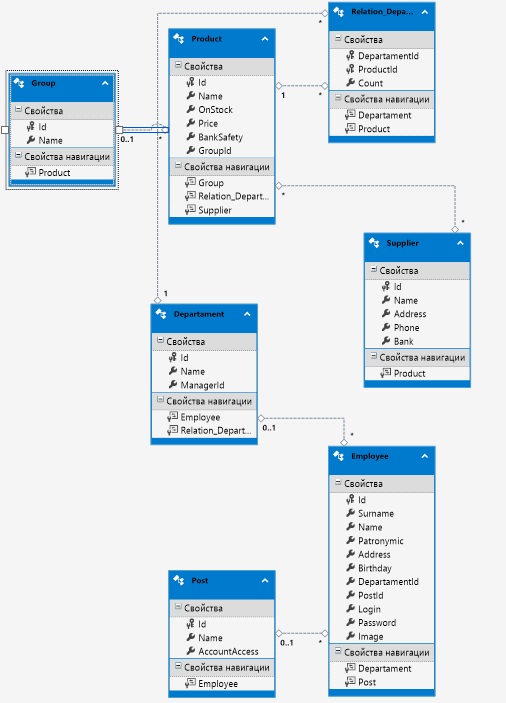
**Цель работы**

По заданной схеме данных разработать базу данных, создать и сохранить скрипт с БД, реализовать программное приложение с возможностью авторизации в системе.

**Подключение базы данных.**

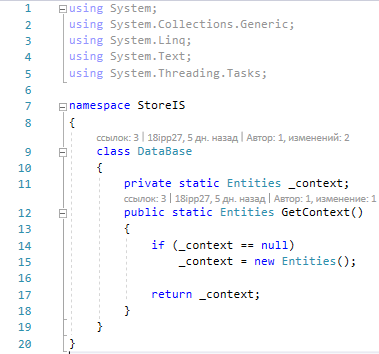
**Была подключена база данных.**

**Модель диаграммы представлена на рисунке 9**



**Был реализован класс DataBase для возможности работы с базой данных**

**Код класса представлен на рисунке 10**



**Контрольные вопросы**

1. Как создать подключение к базе данных?

**Практическая работа №1.3**

**Тема**

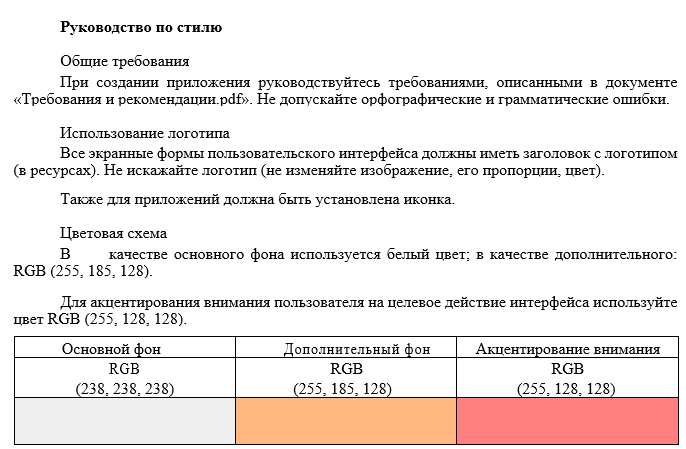
Создание руководства по стилю, логотипа, иконки. Размещение файлов в репозитории.

**Цель работы**

Разместить скрипт с базой данных, руководство по стилю, логотип и иконку в удаленный репозиторий проекта.

**Задание**

**Разработано руководство по стилю, представлено на рисунке 11**



**Разработан логотип и иконка программы, представлены на рисунке 12**



**Скрипт был создан ранее в предыдущей работе.**

**Руководство по стилю, иконка и лого программы, скрипт БД были загружены в удаленный репозиторий проекта в отдельную ветку “pr1\_3”. Для этого была использована программа Git Bash.**

**С помощью команды git branch была создана ветка “pr1\_3”.**

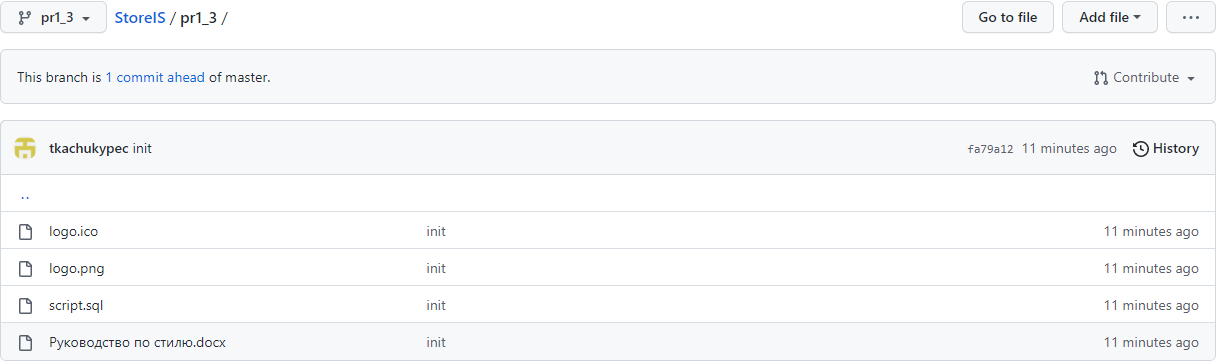
**Команда git bash checkout сменила ветку на нужную.**

**git add pr1\_3/ добавила все файлы из папки pr1\_3 в отслеживание системой Git.**

**Git commit –m “init” – Делаем коммит в новую ветку с добавлением нужных файлов.**

**Git push origin pr1\_3 – Синхронизация локального репозитория с удаленным, то есть создание ветки pr1\_3 и отправление первого коммита в нее с файлами.**

**Ветка с файлами представлена на рисунке 13**



**Контрольные вопросы**

1. В чем заключается экономия времени при использовании системы контроля версий?
2. В чем преимущества использования системы контроля версий?
3. Что такое Git?
4. Как начать использовать Git?
5. Как начать использовать GitHub?
6. Основные (наиболее часто используемые) команды Git.
7. Какие сервисы существуют для Git?
8. Как работать с локальным репозиторием?
9. Как работать с распределенным репозиторием?

**Практическая работа №1.4**

**Тема**

Создание словаря стилей. Создание Git репозитория с помощью Visual Studio.

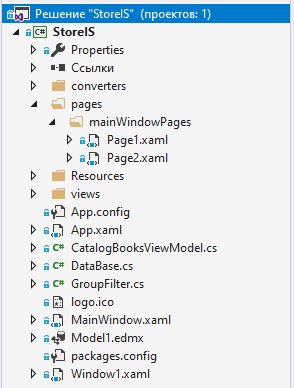
**Цель работы**

Создать и настроить проект в Visual Studio, создать репозиторий, используя средства Visual Studio.

**Настройка проекта WPF**

**Название проекта было изменено на StoreIS (информационная система Магазин). Разработана первоначальная структура файлов проекта, добавлены папки Resources (в ней будет находиться словарь ресурсов), Icons (иконка программы), Images (все необходимые изображения для программы включая лого)**

**Структура файлов проекта представлена на рисунке 14**



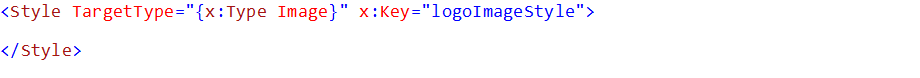
**Некоторые папки и классы, отображенные на рисунке, были добавлены в ходе выполнения последующих практических работ.**

Далее были добавлены лого и иконка программы. Лого будет находиться в папке Images, Иконка в Icons.

Далее был создан словарь ресурсов DefaultTheme в папке Themes. После создания он был прописан в App.xaml, чтобы программа тоже определяла его как словарь.

**Использование атрибута x:Key у свойства**

Бывает необходимость применить стиль к какому-то конкретному экземпляру пользовательского элемента на форме, а не ко всем элементам указанного типа. Допустим, нам нужно задать стиль для логотипа на форме. Понятное дело, что в качестве TargetType будет выступать элемент Image. Но при той форме записи, что мы использовали в словаре, данный стиль будет применен для всех элементов Image. Чтобы этого избежать, укажем для стиля изображения еще один дополнительный атрибут x:Key (рисунок 18).



*Рисунок 18. Стиль для изображения-логотипа.*

В качестве значения для атрибута x:Key выступает строка, которая будет являться «именем» данного стиля.

Пропишем Setterы для данного стиля (рисунок 19).



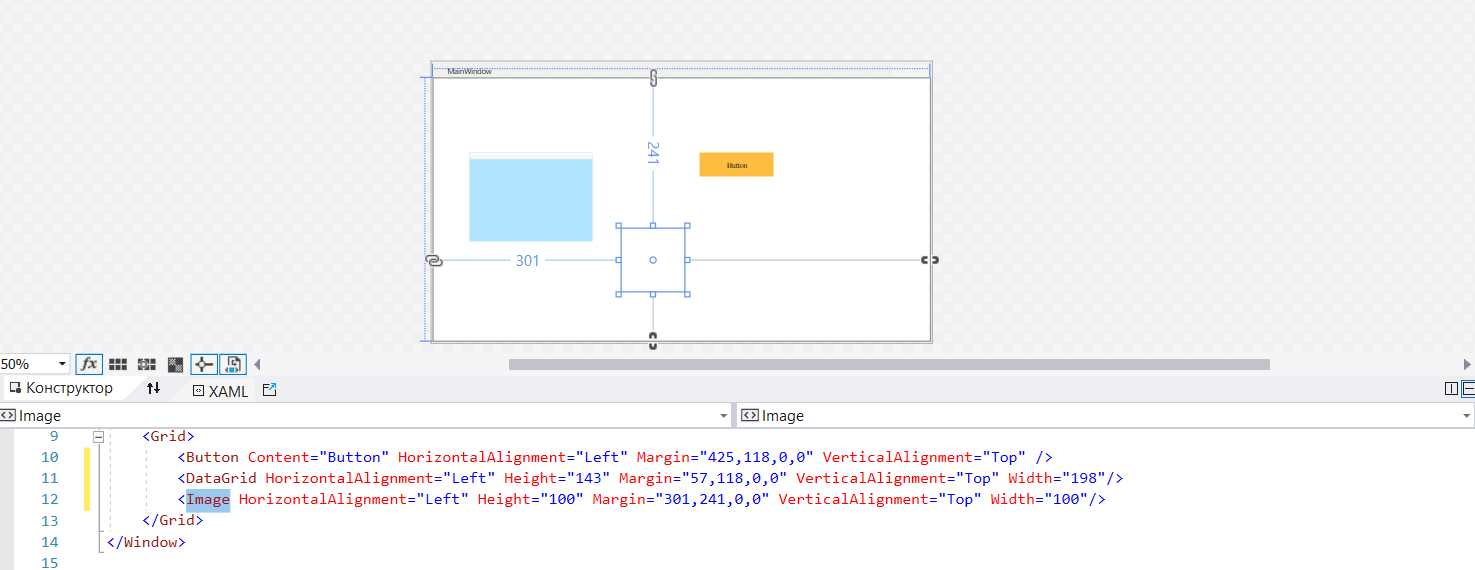
*Рисунок 19. Полное описание стиля для логотипа.*

Свойство Source отвечает за путь до изображения, которое мы хотим отобразить на форме. Стоит помнить, что словарь ресурсов располагается в каталоге Themes, а изображение – в каталоге Image. Поэтому, нам необходимо подняться в каталог Resources (это делается с помощью двух точек), затем перейти в каталог Images, а там указать название изображения.

С помощью свойства Margin указываем отступ элемента. В данном случае, отступ будет слева и сверху по 10 пикселей (так как мы ниже прописываем значения HorizontalAligment и VerticalAligment как Left и Top соответственно).

Также задаем явные размеры элемента с помощью свойств Width и Height.

Вернемся обратно к форме MainWindow. Если сейчас мы попытаемся добавить элемент Image на форму, то сможем убедиться, что описанный нами стиль не применяется к элементу Image (рисунок 20).



*Рисунок 20. Стиль не применяется((*

Все потому, что мы явно не указали стиль для данного элемента.

Для того, чтобы задать стиль для элемента, нужно использовать следующий атрибут (рисунок 21).



*Рисунок 21. Атрибут Style.*

После ключевого слова *StaticResource* мы должны указать точно такое же значение key, которое мы прописывали для стиля.

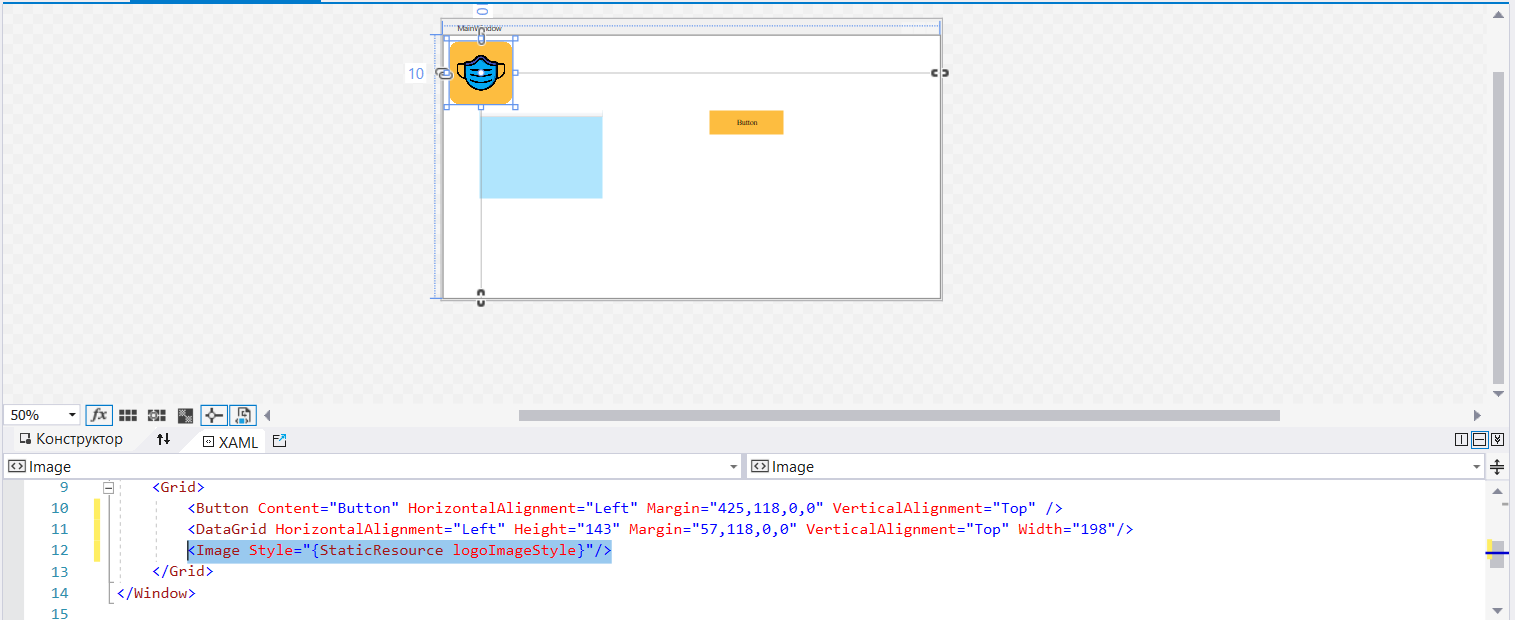
Т.е в нашем случае мы должны прописать (рисунок 22):



*Рисунок 22. Атрибут Style у Image.*

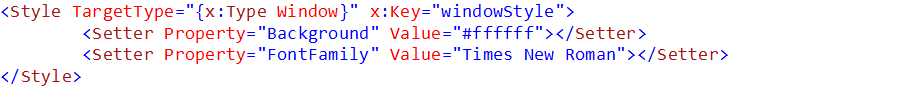
ВАЖНО: НИКАКИХ БОЛЬШЕ АТРИБУТОВ ПРОПИСЫВАТЬ ДЛЯ ДАННОГО ЭЛЕМЕНТА НЕ НУЖНО.

На форме у нас формируется следующая картина (рисунок 23):



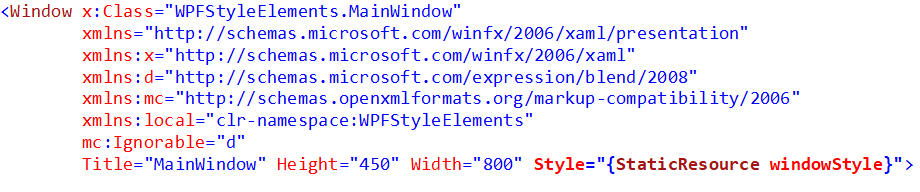
*Рисунок 23. А теперь применяется))*

Последнее, что нужно сделать, это задать шрифт и цвет фона окна. Шрифт, который мы укажем для окна, будет автоматически применен для всех элементов внутри окна. Перейдем обратно к словарю и пропишем следующий стиль (рисунок 24):



*Рисунок 24. Стиль для окна.*

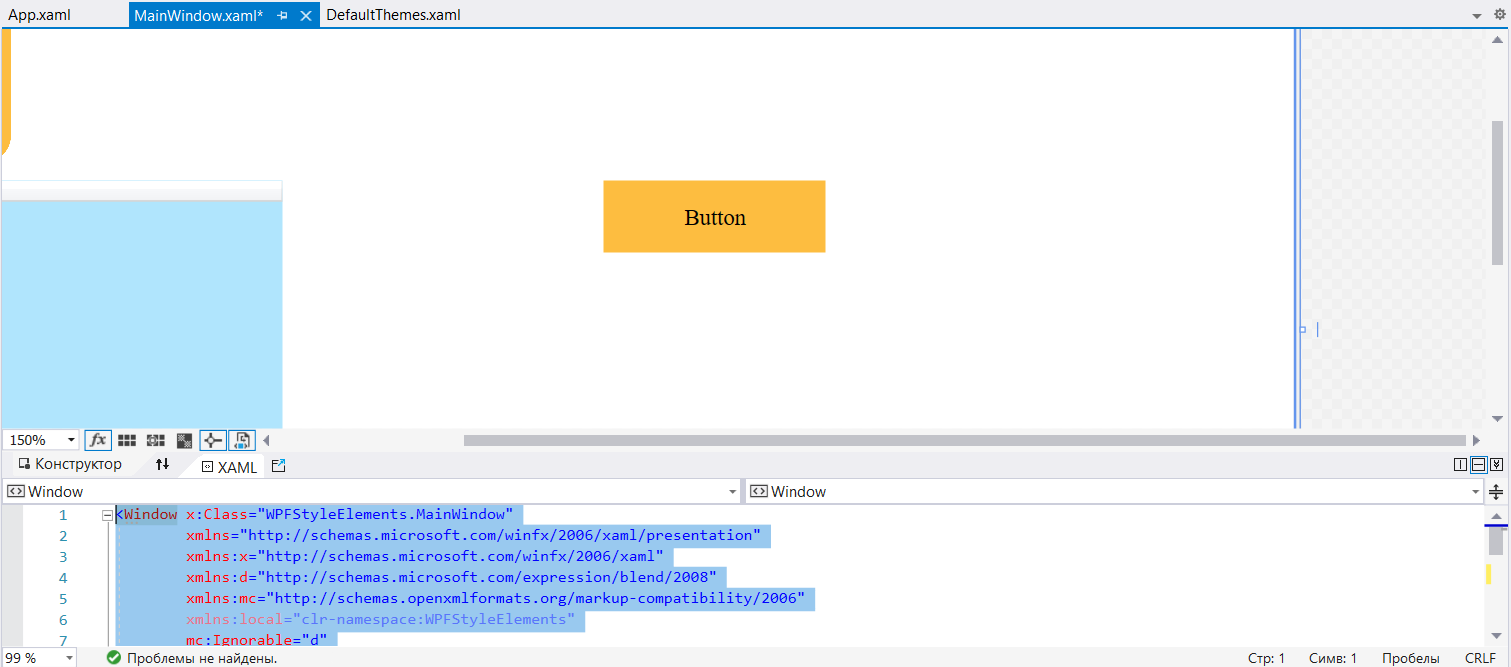
Здесь мы задаем цвет фона окна белым, а качестве шрифта (свойство FontFamily) у нас выступает шрифт Times New Roman. Обратите внимание, что здесь у стиля указан атрибут x:Key, поэтому необходимо у окна явно указать данный стиль (в файле MainWindow.xaml и во всех остальных формах приложения) (рисунок 25).



*Рисунок 25. Применяем стиль к окну.*

***ВАЖНО! НЕ ЗАБЫВАЙТЕ МЕНЯТЬ СВОЙСТВО TITLE у формы.***

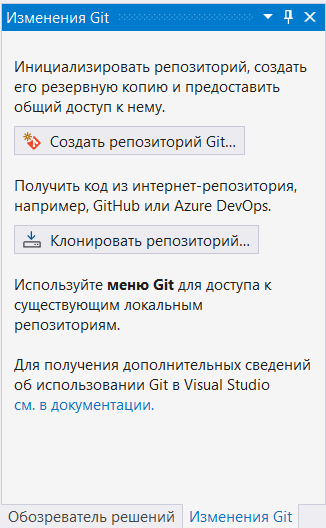
Как видно на скриншоте ниже, у кнопки поменялся шрифт на Times New Roman (рисунок 26).



*Рисунок 26. Результат.*

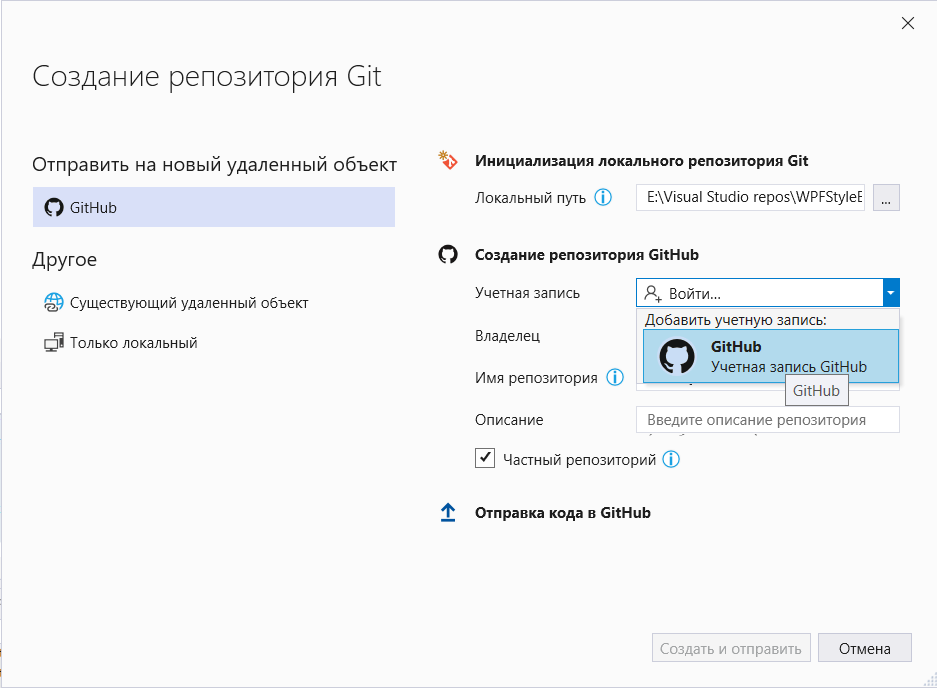
**Создание Git репозитория в Visual Studio.**

Чтобы создать репозиторий средствами Visual Studio, необходимо перейти в обозревателе решений на вкладку «Изменения Git» (рисунок 27).



*Рисунок 27. Изменение Git.*

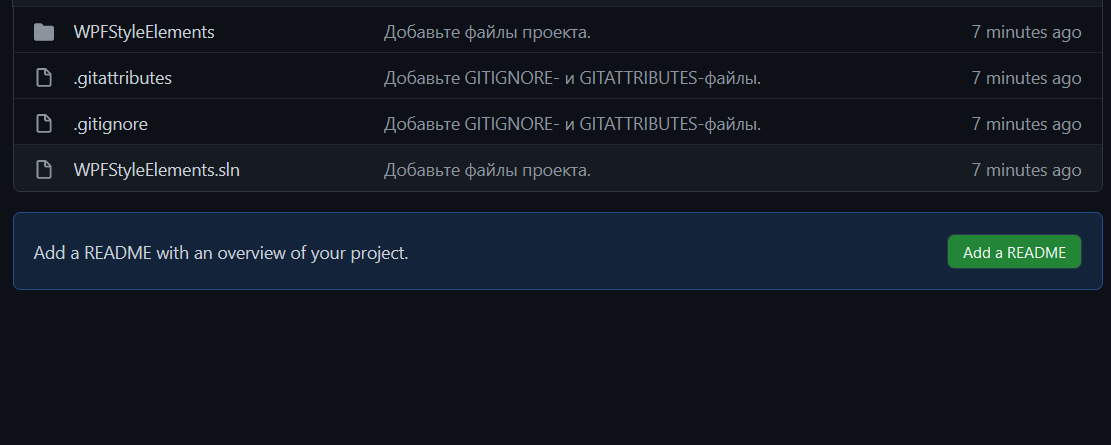
Кликаем по кнопке «Создать репозиторий Git». Откроется окно с настройкой Git. Настройку локального репозитория оставляем без изменений, а в настройках «Создание репозитория GitHub» осуществляем вход под своей учетной записью в GitHub (рисунок 28).



*Рисунок 28. Создание репозитория.*

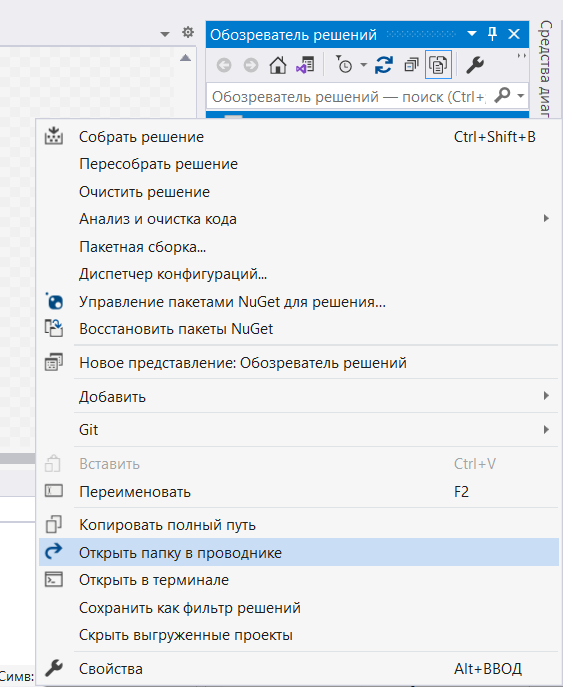
После успешной авторизации указываем название репозитория (можно оставить по умолчанию, в таком случае, репозиторий будет назван как проект), даем описание репозитория. Галочку у «частный репозиторий» не трогаем. Нажимаем на кнопку «Создать и отправить».

Если мы зайдем на GitHub, то увидим примерно следующую картину (рисунок 29).



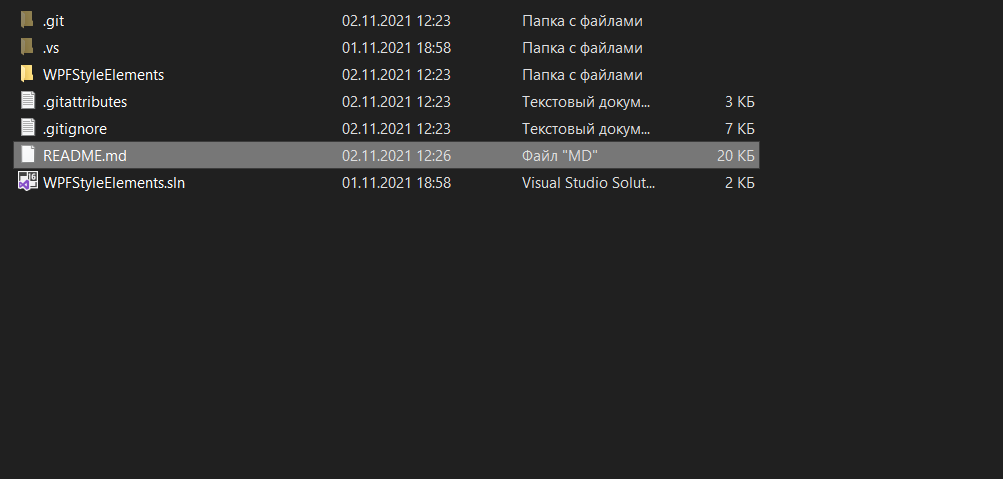
*Рисунок 29. Репозиторий в GitHub.*

В проекте не хватает Readme.md файла. Добавим его к нашему проекту. Для этого, в Visual Studio кликнем правой кнопкой мыши по решению и выберем пункт «Открыть папку в проводнике» (рисунок 30).



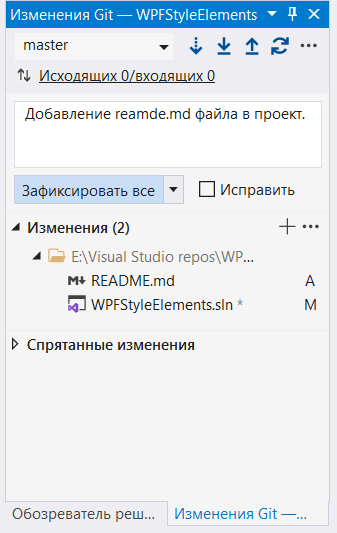
*Рисунок 30. Обозреватель решений.*

В открывшеюся папку скидываем разработанный нами файл Readme.md (рисунок 31).



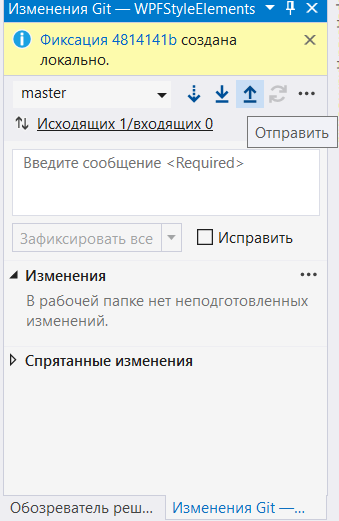
*Рисунок 31. Добавление файла Readme.*

Перейдем обратно в Visual Studio, на вкладку «Изменения Git» и создадим коммит с изменениями в проекте. Для этого в текстовое поле введём комментарий к коммиту и нажмем на кнопку «Зафиксировать все» (рисунок 32).



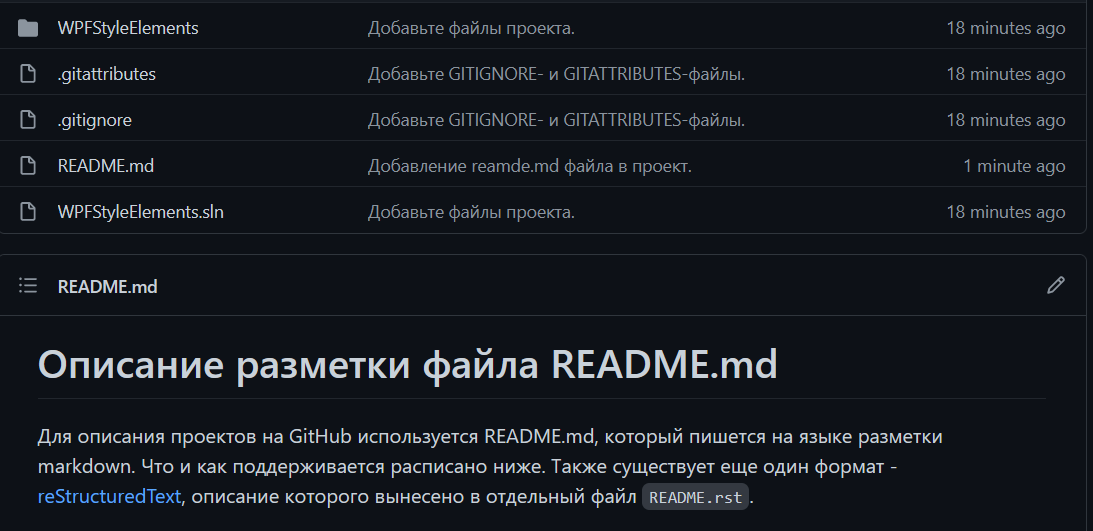
*Рисунок 32. Создание коммита.*

Далее, отправляем созданную локальную фиксацию на удаленный репозиторий с помощью кнопки «Отправить» (рисунок 33).



*Рисунок 33. Отправка изменений на удаленный репозиторий.*

Перейдем в GitHub и убедимся, что файл Reamde.md был добавлен в репозиторий (рисунок 34).



*Рисунок 34. Красота какая стала.*