# Глава 3

## 3.1. Предполагаемый функционал.

Итак, исходя из требований, сформулированных в главе 2 наше приложений должно отвечать следующим требованиям:

* Алгоритм шифрования - AES.
* Все операции должны осуществляться исключительно на устройстве, без передачи данных по сети.
* Приложение должно обладать User-Friendly интерфейсом.
* Приложение должно иметь встроенный способ работы с файловой системой устройства, независимый от наличия других приложений.
* Целевая операционная система для приложения - Android.
* Разрабатываемое Приложение, по возможности, должно иметь возможность переноса на другие ОС с максимальным повторным использованием кода.

Также, для удобства конечного пользователя предполагается внедрить следующие функции:

* Иметь возможность генерации паролей заданной длина.
* Иметь возможность хранить пароли в хранилище паролей защищённым мастер-паролем.

## 3.2 Выбор средств реализации.

С учётом сформулированных в части 3.1 требований, мною были выбраны следующие средства реализации – Microsoft Visual Studio Community Edition включающую в себя средства для разработки на ООП языке C#, с установленным дополнением Xamarin и пакетом разработки Xamarin.Android.(рисунок)

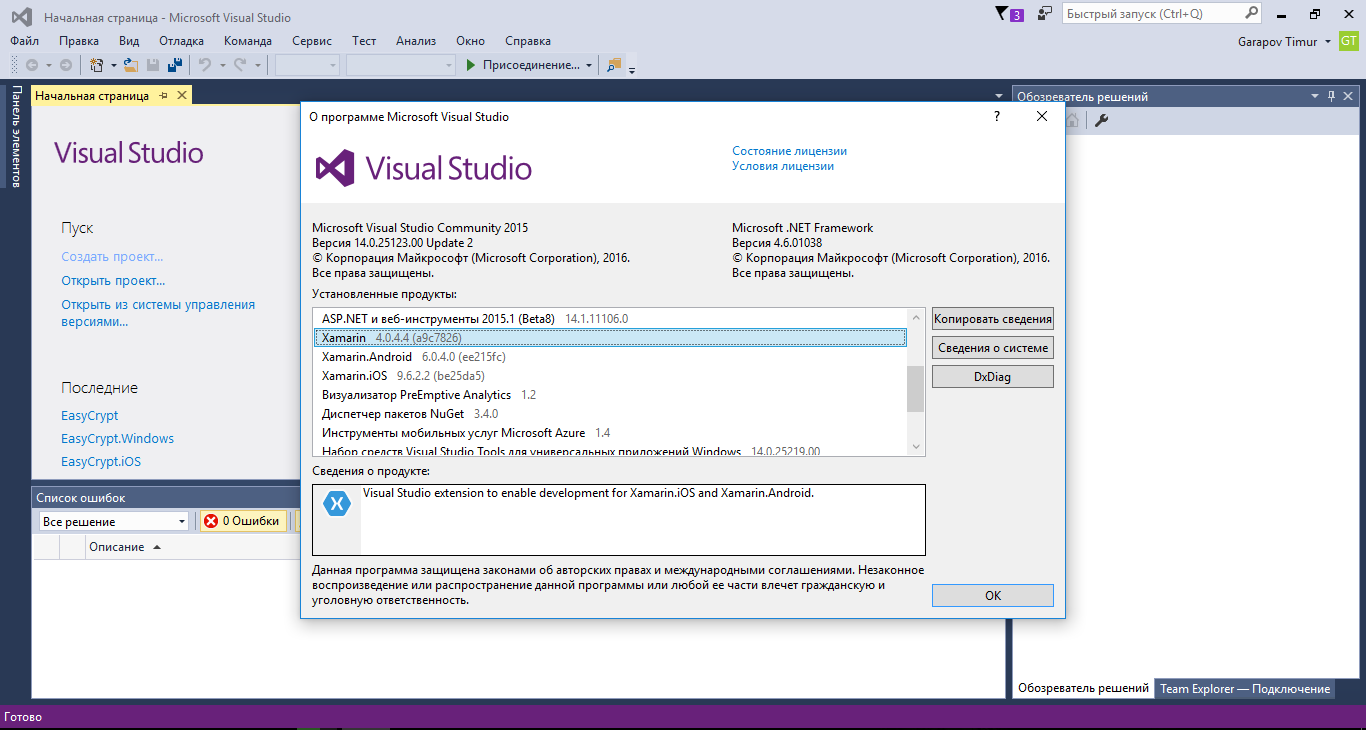


Рисунок 1

Microsoft Visual Studio – это IDE разработанная компанией Microsoft, и поддерживающая множество языков программирования. Однако в данной работе потребуется лишь язык C#, а большую важность для осуществления работы представляет пакет технологий Xamarin, в частности – технология под названием Xamarin Forms.

Следующие факторы послужили причиной выбора связки из языка C# и технологии Xamarin для разработки приложения :

1. Кроссплатформенность.

Xamarin Forms – это технология, позволяющая писать программный код, который возможно скомпилировать сразу под множество платформ. Используя данную технологию можно достичь высокого процента повторного использования кода, так как лишь некоторые специфичные функции, требуют индивидуального написания под конкретную платформу. Пример такого функционала - взаимодействие с файловой системой, которое реализуется разными методами на различных ОС.

1. Скорость разработки.

Использование высокоуровнего языка С# в сочетании с готовыми библиотеками функций, предоставляемых технологиями .NET и Xamarin позволяют быстро реализовывать сложные приложения, используя обширную базу готовых решений.

1. Скорость работы.

Использование функций, предоставляемых используемыми библиотеками, позволяет взаимодействовать с мобильными ОС оптимальными методами.

1. Строгая Типизация.

Язык C# является языком со строгой типизацией переменных. Строгая типизация переменных упрощается процесс отладки кода, так как в отличии от динамической типизации исключаются ошибки, которые могут возникнуть при некорректном обращении к переменным.

1. Инструменты тестирования.

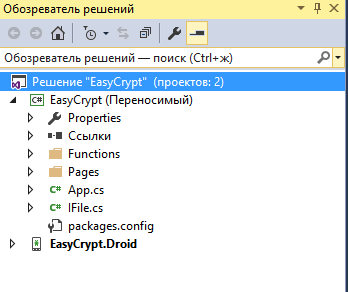
Среда разработки Microsoft Visual Studio имеет встроенные средства тестирования приложений, и кроме прочего, для языка C# существует библиотека NUnit, предназначенная для произведения тестирования приложений.

## 3.3 Описание программной реализации. Интерфейс.

Технология Xamarin.Forms позволяет реализовывать общий интерфейс для всех платформ. Для описания интерфейса используется расширяемый язык разметки для приложений XAML. XAML – это основанный на XML язык разметки разработанный Microsoft.

Любой проект, использующий технологию Xamarin.Forms имеет в своём составе два и более под-проекта. Один проект – основной, и в нём находится весь программный код, который будет являться общим для всех производных проектов. Именно в этом проекте находится логика приложения, реализация которой не зависит от конкретной платформы. На рисунке № можно увидеть разбиение проекта на под-проекты в текущем решении:

Рисунок 2



Основной проект имеет метку “(Переносимый)”. Основой этого проекта является файл App.cs, именно он является входной точкой приложения.

Весь код проекта, описывающий интерфейс страниц приложения находится в папке Pages. Каждая страница состоит из связки файлов вида [Название страницы].xaml, описывающего внешний вид страницы, и сопутствующего файла [Название страницы].xaml.cs, описывающего логику страницы на языке C#.

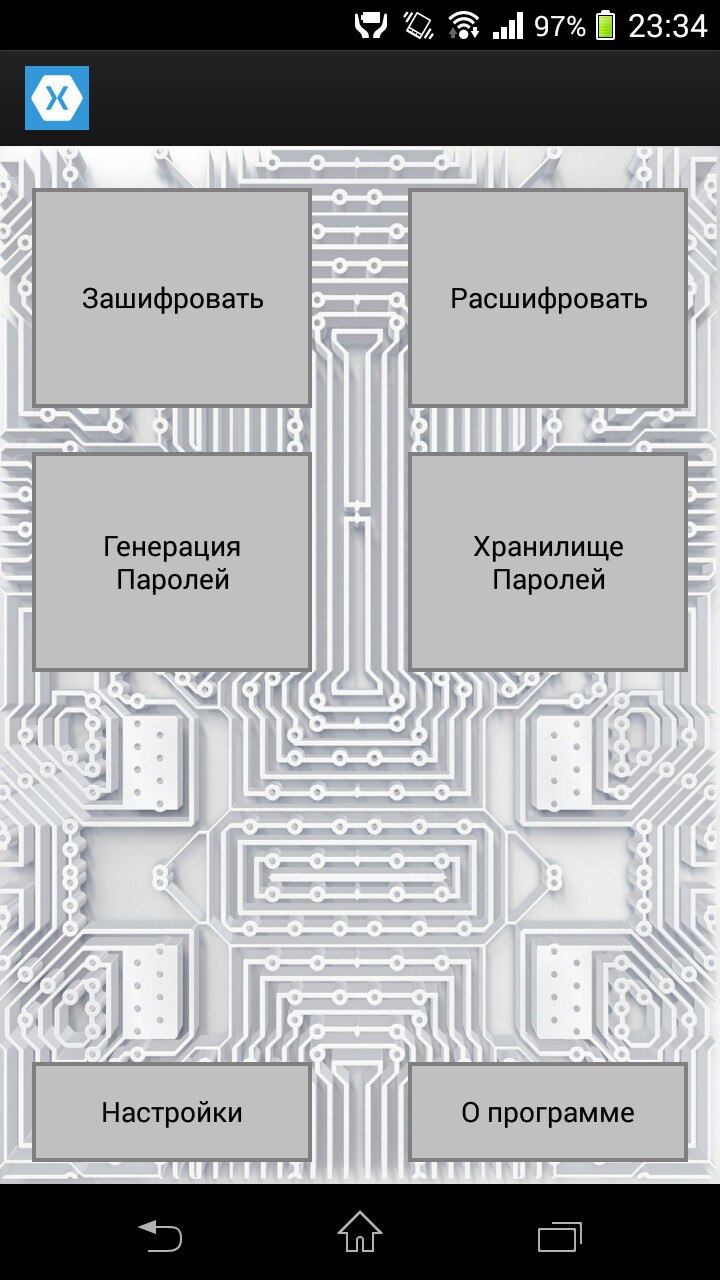
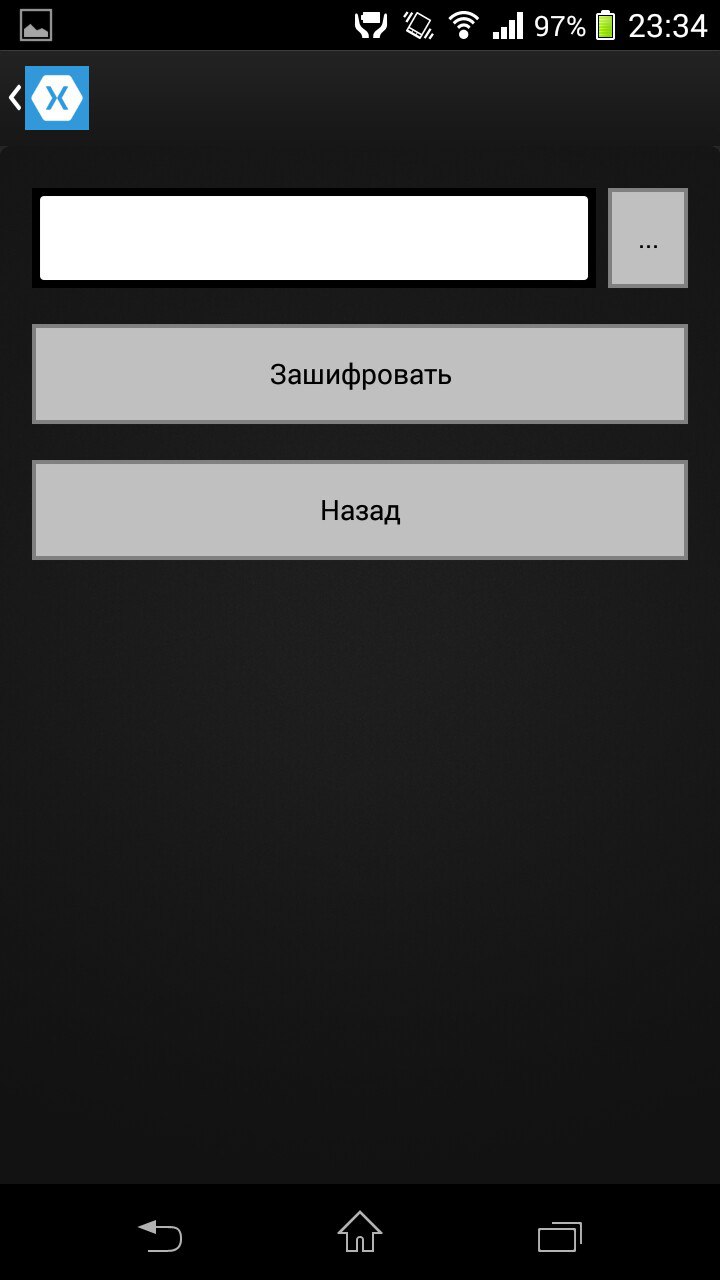
Страница может состоять из различных элементов, расположенных на ней. В приложении для упорядочивания создаваемых элементов на странице используется инструмент с названием Grid – сетка. Сетка позволяет быстро описывать интерфейс, не теряя при этом в качестве.

Пример использования языка Xaml для описания элементов на главной странице, под названием MainPage:

|  |
| --- |
| <Button x:Name="button01" Text="Зашифровать" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Clicked="Button\_Click01" Grid.Column="1" Grid.Row="1" />  <Button x:Name="button02" Text="Генерация Паролей" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Clicked="Button\_Click02" Grid.Column="1" Grid.Row="3" />  <Button x:Name="button11" Text="Расшифровать" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Clicked="Button\_Click11" Grid.Column="3" Grid.Row="1" />  <Button x:Name="button12" Text="Хранилище Паролей" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Clicked="Button\_Click12" Grid.Column="3" Grid.Row="3" />  <Button x:Name="button21" Text="Настройки" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Grid.Column="1" Grid.Row="6" />  <Button x:Name="button22" Text="О программе" BorderWidth="2" BorderRadius="1" BorderColor="Gray" BackgroundColor="Silver" TextColor="Black" Grid.Column="3" Grid.Row="6" /> |

Данная часть кода описывает инициализацию кнопок на главной странице, связывает их нажатие с соответствующими функциями.

Пример интерфейса продемонстрирован на рисунках № и №.

На рисунке № изображена главная страница приложения, а на рисунке № показан внешний вид страницы, на которой происходит выбор файла и осуществляется его шифрование.

Полный код интерфейса будет представлен в рамках кода всего созданного программного обеспечения в Приложении А.

## 3.4 Описание программной реализации. Функционал.

Функционал приложений описывается в соответствующих файлах, являющихся либо описанием классов объектов, используемых в приложении, либо эти функции описаны в файле Functions.cs, описывающих класс, предоставляющий объектам доступ к статическим методам, обеспечивающим функциональность приложения.

Список некоторых методов, используемых в приложении:

* DoAes – функция, принимающая на вход адрес файла, путь к выходной папке, куда следует записать результирующий файл, и пароль, используемый для генерации ключа шифрования.
* UndoAes – функция, обратная предыдущей. Получает на вход адрес зашифрованного файла, путь к результирующей папке, и пароль, используемый для генерации ключа.
* GetPath – функция, вызывающая окно выбора файла и возвращающая упорядоченный адрес результирующего файла.
* GetPassword – функция, принимающая на вход длину требуемого пароля и осуществляющая его генерацию. Возвращает строку, представляющую собой сгенерированный пароль.
* GetMainPage – Функция, при вызове возвращающая новый объект типа Page, используется при старте приложения для получения стартовой страницы.

Для навигации и перемещения между страницами была создана группа функций, вызываемая при нажатии соответствующих кнопок. Название функций имеет вид Goto[название страницы]. Данные функции работают используя функции навигации:

* Navigation.PushAsync() ;
* Navigation.PopAsync() ;

Данные функции предоставляют доступ к работе с навигационным стеком, и позволяют записывать или изымать страницы из этого стека.

## 3.5 Руководство пользователя.

### Осуществление шифрования.

1. Запускаем приложение
2. Выбираем пункт меню – шифрование
3. Нажимаем на кнопку “[…]” и выбираем файл, который требуется зашифровать
4. Нажимаем кнопку “Зашифровать”
5. Вводим пароль
6. Ждём сообщения об успешном окончании операции или сообщения об ошибке.

### Осуществление расшифровывания.

1. Запускаем приложение
2. Выбираем пункт меню – расшифровать
3. Нажимаем кнопку “[…]”
4. Нажимаем кнопку “Расшифровать”
5. Вводим пароль, использованный для зашифровывания
6. Ждём сообщения об успешном окончании операции или сообщения об ошибке.

### Генерация Пароля.

1. Запускаем приложение
2. Выбираем пункт меню – Генерация Паролей
3. Вводим требуемое количество символов, по умолчанию – 8
4. Нажимаем кнопку “Генерировать”

Получаем пароль.ЗАКЛЮЧЕНИЕ

За последние годы распространение мобильных устройств происходило лавинообразными темпами. На территории РФ у каждого третьего жителя страны на руках имеется хотя-бы одно мобильное устройство. Количество мобильных устройств в мире и активность их использования говорит об актуальности выбранной темы.

В ходе данной работы были рассмотрены особенности практического применения шифрования. Был произведён анализ популярных алгоритмов шифрования, изучены исследования экспертов, посвящённые рассмотренным алгоритмам.