Информатика и системы управления

УДК 004.514

Герман Анна Сергеевна
Амурский государственный университет
г. Благовещенск, Россия
E-mail: veselovablg@gmail.ru
German Anna Sergeevna
Amur State University
Blagoveshchensk, Russia
E-mail: veselovablg@gmail.ru
Веселова Елена Михайловна
Амурский государственный университет
г. Благовещенск, Россия
E-mail: veselovablg@gmail.ru
Veselova Elena Mikhailovna
Amur State University
Blagoveshchensk, Russia
E-mail: veselovablg@gmail.ru

РАЗРАБОТКА ФРЕЙМВОРКА ДЛЯ СОЗДАНИЯ ПОЛЬЗОВАТЕЛЬСКОГО ИНТЕРФЕЙСА МАКРОСОВ VBA

DEVELOPING A FRAMEWORK FOR CREATING USER INTERFACE VBA MACROS

Аннотация. Статья посвящена проблеме создания графических пользовательских интерфейсов в среде VBA. Сформулированы основные требования к разработке пользовательских интерфейсов с учетом специфики использования интерфейсов в программах VBA. Предложен прототип подключаемой библиотеки фреймворка для создания пользовательских интерфейсов, который должен позволить экономить время при разработке VBA-приложений.

Abstract. The article is devoted to the problem of creating graphical user interfaces in the VBA development environment. The main requirements for the development of user interfaces are formulated, taking into account the specifics of using interfaces in VBA programs. The prototype of an application framework for creating user interfaces is made, which should save developers time.

Ключевые слова: VBA, графический пользовательский интерфейс, фреймворк, MVC.
Key words: VBA, GUI, framework, MVC.

DOI: 10/22250/jasu.6

Введение

Вероятно, каждый разработчик программ в интегрированной среде программирования VBA (Visual Basic for Application) сталкивался с задачей создания пользовательского интерфейса с целью облегчить взаимодействие пользователя с программой. Для этих целей в среде VBA имеется встроенный инструмент для создания пользовательских форм, представляющий собой полноценный *конструктор графического пользовательского интерфейса* (GUI-редактор), позволяющий создавать диалоговые окна на основе технологии WinForms [1]. Другим важным аспектом является возмож-

ность полноценно использовать COM-технологию в VBA-программах, а следовательно, и все имеющиеся на машине ActiveX компоненты [2]. Это позволяет расширять возможности как самих программ, так и пользовательских интерфейсов. Однако достоинство одновременно является и недостатком: использование сторонних компонентов ограничивает переносимость программы на другие устройства ввиду необходимости установки и регистрации этих компонентов в операционной системе, что не всегда возможно в случае ограниченного доступа пользователя на целевом устройстве.

Альтернативой может быть применение сторонних фреймворков, позволяющих быстро и эффективно создавать сложные интерфейсы. Существует множество готовых решений, в том числе кроссплатформенных [3, 4], но в силу особенностей среды VBA использование данных библиотек либо крайне затруднительно, либо невозможно в принципе [2].

Формулировка требований

В результате анализа предметной области были сформулированы следующие требования к продукту, предъявляемые разработчиком:

1. Быстрый старт. Программный продукт должен быстро подключаться к среде программирования VBA и иметь минимум обязательных преднастроек.

2. Простота использования. Минимум базовых команд, формирующих основной функционал продукта.

3. Гибкость. Возможность необязательной подстроки, изменения формы, цвета и иных «косметических» и функциональных особенностей интерфейса.

Разработчик, используя фреймворк, создает интерфейс, целевым конечным пользователем которого являются люди без профессиональных знаний в области программирования и информационных систем, что порождает специфические требования к разрабатываемым интерфейсам:

1. Интуитивная понятность. Пользователю не требуются специальные навыки для работы с интерфейсом.

2. Интерактивная справка. Пользователь должен иметь возможность получить исчерпывающую информацию о том, что от него требуется в данный момент, без необходимости изучения инструкции.

3. «Наставничество». Пользователь должен своевременно быть уведомлен о совершении ошибки и получить информацию о том, что именно было сделано неверно.

Стоит отметить, что такие требования продиктованы спецификой использования интерфейсов в рамках макросов. В подавляющем большинстве случаев интерфейс используется для ввода входных параметров с целью исполнения программы макроса и в меньшей степени – отображения полученной информации. Пользователь использует макрос с четким пониманием, какой результат он хочет получить, поэтому от разработчика требуется с помощью доступных средств ясно и однозначно показать, какую именно информацию пользователю нужно передать программе посредством графического интерфейса.

Удовлетворение потребностей конечного пользователя ложится на разработчика, поэтому необходимо обеспечить его как инструментами, так и готовыми решениями, позволяющими при минимальных затратах сил и времени наиболее полно обеспечить пользователя всем необходимым.

Разработка фрейворка

Как уже упоминалось, не все современные решения возможно использовать в среде VBA. Наиболее универсальным вариантом с точки зрения функциональности, мобильности и автономности являются динамически подключаемые библиотеки (DLL) [5]. А как показано в работе [6], использование динамических библиотек поможет сильно выиграть в производительности – больше, чем в случае попытки разработки фреймворка только средствами VBA.

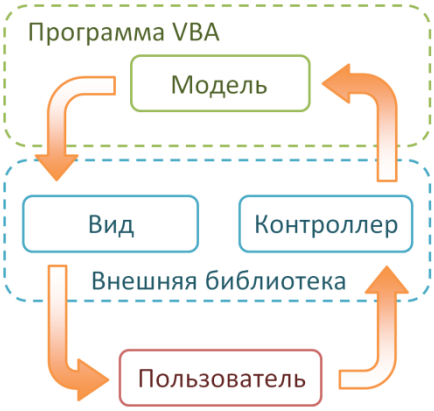

Рис. 1. Схематичное представление внутренней структуры и взаимодействия VBA-программы и фреймворка.

Реализацию поставленной задачи было решено осуществлять на основе архитектурной модели MVC (англ. *Model-View-Controlle*, «Модель-Представление-Контроллер»). Принцип MVC предполагает разделение управляющей логики и пользовательского интерфейса на три независимых модуля: *модель, представление (вид), контроллер*. В рамках задачи модуль *модели*, который отвечает за доступ к данным и реализует управляющую логику, расположен непосредственно в VBA-программе, а библиотека фреймворка содержит модуль *контроллера*, обрабатывающий команды пользователя, и модуль *представления (вида)*, отвечающий за графическую часть интерфейса пользователя. Схематичное устройство и взаимодействие описанных блоков изображено на рис. 1.

За основу были взяты списочные интерфейсы, когда каждый последующий элемент управления расположен под предыдущим. С этой моделью хорошо знаком каждый, кто имеет сенсорный мобильный телефон, так как именно с приходом сенсорного управления данный тип интерфейсов получил наибольшую популярность. Тем не менее это обстоятельство не исключает возможности использовать его для ПК-приложений. Основной особенностью такого интерфейса является его последовательность: пользователь идет сверху вниз, исключая возможность пропустить что-то или запутаться в большой концентрации активных элементов управления. Именно такой интерфейс, по нашему мнению, хорошо подходит для малых и средних приложений VBA, когда необходимо ввести несколько входных параметров для выполнения программы.

Второй важной особенностью списочных интерфейсов является их относительно быстрое создание: разработчику нет необходимости располагать каждый элемент по координатной сетке, достаточно указать, какие элементы необходимо отобразить на форме.

За основу цветового представления HSB предлагается взять цветовую модель (англ. *Hue, Saturation, Brightness – тон*, *насыщенность*, *яркость*). Данная цветовая модель более удобна для восприятия человеком, чем принятая RGB-модель. Построение модуля *представления* на HSB-модели позволило бы сделать визуальное оформление гораздо более гибким.

Приведем простой пример использования HSB цветовой модели: элементу управления «кнопка» зададим относительно нормального состояния при наведении курсора бóльшую яркость, при нажатии – меньшую яркость, а в неактивном состоянии установим значение насыщенности, равное нулю. Тогда, чтобы изменить цвет кнопки, необходимо определить только одно значение – *тон*. Таким образом, можно настроить весь пользовательский интерфейс, взяв за основу один или несколько базовых цветов, что позволит разработчику менять цветовое оформление любого элемента управления или всего графического интерфейса одной командой.

Использование технологии внутристрочного форматирования текста (например, *BB-кодов*, широко распространенных в Web-практике) может дать разработчику широкие возможности визуального представления текста без усложнения программного кода.

Последний важный момент – система подсказок и проверки, вводимых пользователем значений, возможность их своевременной проверки и вывод уведомления о неправильных действиях. Обычно принято сообщать пользователю об ошибках в заполненных им полях только после нажатия кнопки подтверждения. Наличие возможности контроля правильности в реальном времени поможет пользователю сразу исправить свои ошибки, без необходимости возвращения к уже заполненным полям, что поможет сэкономить время пользователя и уменьшить дискомфорт при взаимодействии с интерфейсом.

На рис. 2 показаны примеры всплывающих подсказок, информирующих пользователя о действиях, которые от него требуется совершить, и о допущенной ошибке.

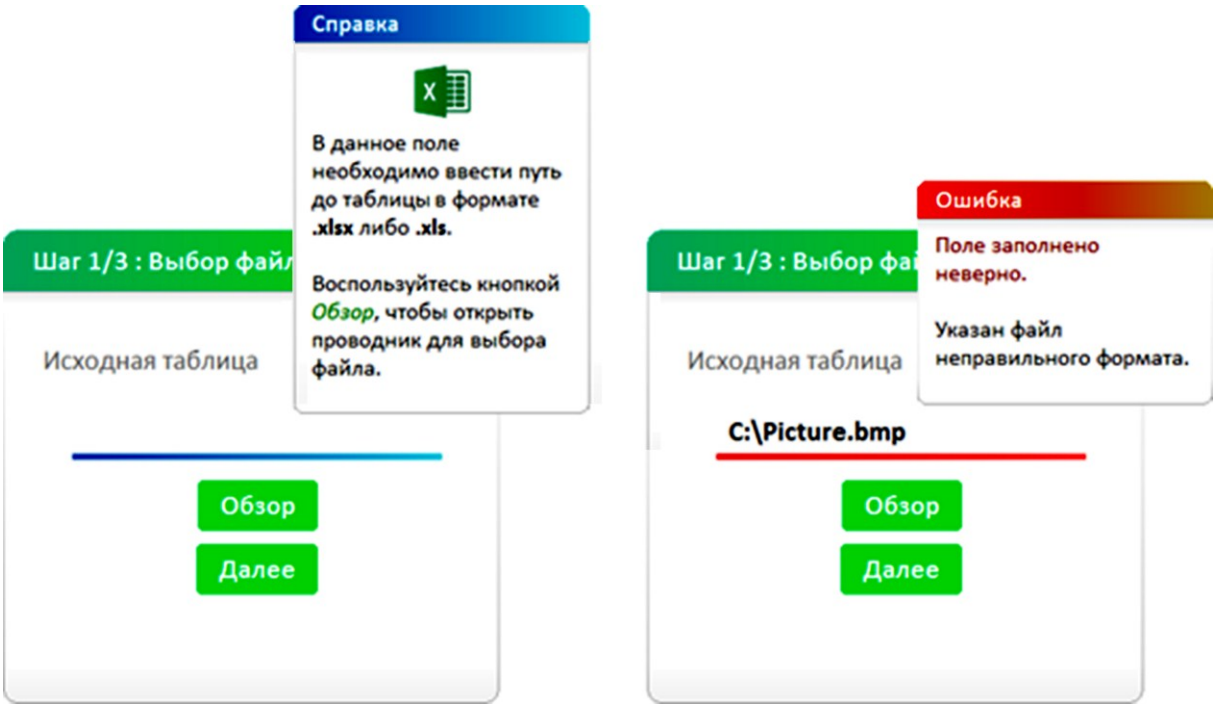

Рис. 2. Иллюстрация интерактивной всплывающей подсказки и сообщения об ошибке.

Функционал проверки правильности ввода закладывается в реализацию модуля *контроллера*, разработчик должен иметь возможность настройки шаблона ввода. В качестве шаблона предлагается использовать регулярные выражения. С целью сокращения трудозатрат должны быть предусмотрены уже настроенные шаблоны для типовых полей ввода.

Важной особенностью интерактивности при сообщении об ошибках является указание на совершенную ошибку, а не только на поле, в котором она была обнаружена. На рис. 3 рассмотрен пример, когда пользователь, заполняя поле «Имя», к которому предъявляется требование использовать только символы кириллицы, ввел первую букву в другой раскладке.

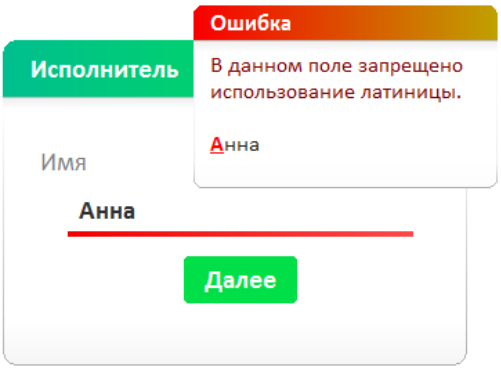

Рис. 3. Иллюстрация сообщения об ошибке: в поле содержится символ в неверной раскладке.

Заключение

В данной работе были сформулированы требования к графическому интерфейсу пользователя с учетом специфики использования интерфейсов в рамках макросов на VBA. Был разработан прототип библиотеки фреймворка для быстрого создания списочных интерфейсов, реализующий описанные принципы.

Планируется продолжить разработку фреймворка с последующим распространяем как ПО с открытым кодом.

1. Абрамова, И.А. Расширения стандартного функционала и интерфейса приложений MS Office на основе разработки пользовательских надстроек / И.А. Абрамова, В.В. Сыркин, А.П. Степанов // Наука и военная безопасность. – 2020. – № 2 (21). – С. 192-199.

2. Макапов, А.А. Основные возможности VBA и MS EXCEL: преимущества и недостатки // Информация и образование: границы коммуникаций. – 2019. – № 11 (19). – С. 124-125.

3. Искра, Н.А. Изучение и оценка подходов к разработке графического интерфейса пользователя / Н.А. Искра, В.Н. Макоед, Е.Ю. Куница // Объектные системы. – 2015. – №10. – С. 63-68.

4. Грузин, Н.А. Обзор и сравнение библиотек пользовательских интерфейсов: GTK, QT, WXWIDGETS / Н.А. Грузин, А.А. Голубничий // E-Scio. – 2020. – № 2 (41). – С. 1-8.

5. Лебедев, В.М. Программирование на VBA в MS EXCEL. – М.: Юрайт, 2019. – 272 с.

6. Герман, А.С. Параллельные вычисления в среде VBA Excel / А.С. Герман, Е.М. Веселова // Молодежь и наука: актуальные проблемы фундаментальных и прикладных исследований. Материалы III Всероссийской национальной научной конференции студентов, аспирантов и молодых ученых: В 3-х частях. – 2020. – С. 245-247.