Лекция № 8.
Окна как элементы графического интерфейса.
Модели интерфейса.

Модели построения интерфейса.

В соответствии с концепциями, положенными в основу графического интерфейса, объекты приложения могут быть визуально представлены на Рабочем столе либо в виде пиктограмм, либо в виде окон, отображающих содержимое объекта.

Во многих случаях для реализации взаимодействия пользователя с объектами приложения или приложением в целом оказывается достаточным единственного первичного окна, возможно, дополненного набором вторичных окон.

Однооконная модель приложения облегчает пользователям ассоциативную связь между объектами и их визуальным представлением, существенно упрощает пользователю работу с окнами.

Объекты некоторых типов (например, устройства) могут не требовать создания первичного окна и использовать только вторичное окно для просмотра и редактирования их свойств. Иногда объект может быть представлен в приложении лишь своей пиктограммой.

При выполнении некоторых заданий однооконная модель не обеспечивает достаточно эффективного управления приложением или отдельными его объектами; такая ситуация может иметь место в тех случаях, когда пользователю требуется работать одновременно с несколькими различными форматами представления одних и тех же данных или с несколькими видами взаимосвязанных данных в пределах одного окна. В таких случаях следует использовать другие модели приложения: на основе *многодокументного интерфейса* (MDI) или *Проекта*.

Техника взаимодействия пользователя с приложением существенно зависит от выбранной модели его построения.

Пиктограммы.

Все пиктограммы, используемые в приложении, следует разрабатывать как единый набор. При этом должна обеспечиваться их согласованность и друг с другом, и с заданиями пользователя.

Каждая пиктограмма должна быть реализована в трех стандартных форматах:

- 16x16 пикселов (для 16 цветов);
- 32x32 пиксела (для 16 цветов);

- 48x48 пикселов (для 256 цветов).

Для меньших размеров может быть использована большая глубина цвета, но это требует увеличения памяти для хранения пиктограмм и более жестких требований к конфигурации компьютера.

Система автоматически формирует цветовую схему пиктограммы для монохромных конфигураций. Тем не менее, следует заранее оценить качество зрительного восприятия разработанных пиктограмм в монохромном режиме. Если результат окажется неудовлетворительным, необходимо разработать собственные монохромные варианты пиктограмм.

Пиктограммы разрабатываются не только для исполнимого файла приложения, но и для всех типов файлов данных, поддерживаемых вашим приложением. При этом пиктограммы для файлов данных (или документов) должны отличаться от пиктограмм приложения, но иметь с ними общий элемент. Пиктограммы должны отображать сущность хранимой информации.

Все созданные пиктограммы должны быть зарегистрированы в системном реестре, иначе система будет автоматически использовать вместо них системные пиктограммы.

В основу рисунка, отображаемого на пиктограмме, должен быть положен образ объекта реального мира, точнее, те его детали, которые действительно необходимы для однозначного восприятия объекта пользователем. Где это возможно, лучше использовать трехмерное изображение и светотень.

Выводимое на пиктограммах изображение должно вызывать у пользователя вполне определенную предсказуемую ассоциацию с объектами реального мира.

Окна. Их виды и структура.

Окна предоставляют доступ к различным видам информации и классифицируются согласно своему предназначению.

Первичное окно.

Взаимодействие с объектами реализуются средствами первичного окна, в котором происходит первоначальный просмотр и редактирование данных.

Типовая структура первичного окна:

- рамка – определяет размеры окна;
- заголовок окна – идентифицирует информацию, представленную в окне, может содержать кнопки управления первичным окном (Закрыть, Развернуть/Восстановить, Свернуть);

- полосы прокрутки – используются, если объем выводимой информации превышает текущий размер окна;
- другие элементы интерфейса (меню, панель инструментов, строка состояния).

Внешний вид рамки окна определяется типом окна. Изменяемое окно имеет четкую границу, которая обеспечивает управление размерами на основе прямого манипулирования. Если окно не может изменять размеры, граница сливается с краем окна.

Первичное окно содержит уменьшенную копию пиктограммы объекта или приложения, к которому оно относится. Она выводится в левом верхнем углу окна – в полосе заголовка и выбирается по следующим правилам:

- если окно относится к компоненту приложения, не создающему свои файлы данных, то используется пиктограмма самого приложения;
- если приложение обеспечивает работу с документами (файлами) различных форматов, то используйте пиктограмму, соответствующую формату отображаемого в окне документа;
- если приложение использует многодокументный интерфейс, пометите пиктограмму приложения в заголовке родительского окна, а в заголовке дочернего окна – пиктограмму конкретного типа файла данных.

Как отмечалось выше, поле заголовка содержит кнопки управления первичным окном. Для первичных окон в число этих кнопок не включается кнопка для вызова контекстно-зависимой справочной информации. Если наличие справки необходимо, то соответствующая кнопка включается в панель инструментов.

Для кнопок управления первичным окном используются следующие правила:

- если команда не поддерживается окном – не отображайте соответствующую кнопку;
- кнопка закрытия окна всегда должна быть самой правой кнопкой. Оставляйте промежуток между ней и другими кнопками;
- кнопка Свернуть должна предшествовать кнопке развернуть;
- Кнопка восстановить всегда заменяет кнопку Развернуть или кнопку Свернуть после выполнения соответствующей команды.

Использование подокон.

Окно может разделяться на две и более относительно независимых областей, которые называются *подокнами*. Разделение окна позволяет, например, просматривать одновременно две части одного документа или отображать одну и ту же информацию в различной форме.

Если необходимо одновременно получить доступ к нескольким файлам при выполнении одного задания, следует использовать технологию MDI.

Разбиение окна на подокна может быть установлено либо разработчиком приложения как основная форма окна, либо пользователем, посредством задания соответствующего параметра.

Для того чтобы поддерживать разбиение окна, которое не определено заранее, включите в состав создаваемой программы *блок разделения*. Блок разделения - специальный элемент управления, который отображается в конце полосы прокрутки окна и обозначает регулируемую границу между подокнами.

Пользователь может изменять размеры подокон, перемещая блок разделения в нужную позицию.

При использовании подокон каждое из них должно иметь собственные значения атрибутов. При этом область выбора следует отображать только в активном подокне.

Когда основное окно закрывается пользователем, следует запоминать состояние подокон (количество, расположение, отображаемая информация, состояние выбора) как часть информации о состоянии этого окна. Это необходимо для восстановления окна.

Вторичные окна.

Вторичные окна предназначены для приема от пользователя или отображения дополнительной информации об объектах, представленных в первичном окне. Они позволяют устанавливать дополнительные параметры обработки или обеспечивают доступ к более специфическим деталям взаимодействия с объектами первичного окна.

Вторичные окна обладают некоторыми свойствами первичных окон, тем не менее отличаются от первичных во многих аспектах поведения и использования.

Для вторичных окон не создаются кнопки на панели задач!

Стандартное вторичное окно содержит:

- полосу заголовка окна;
- поле, ограниченное рамкой.

Пользователь может перемещать его с помощью мыши.

Нежелательно изменять размеры вторичного окна, кроме окна палитры, поскольку любое вторичное окно предназначено для отображения конкретной предопределенной информации.

В некоторых случаях может возникнуть необходимость последовательного уточнения или дополнения отображаемой в окне информации; в таком окне может использоваться специальная кнопка – Дополнить.

Вторичное окно не имеет кнопок управления Развернуть и Свернуть. Для закрытия окна используется кнопка Закрыть.

Заголовок вторичного окна является его меткой и поясняет назначение окна; полоса заголовка вторичного окна не содержит пиктограммы.

Разрешается включать во вторичные окна строку состояния, но не рекомендуется дублировать в ней элементы, используемые в строке состояния первичного окна.

Вторичное окно может содержать в полосе заголовка окна кнопку вызова контекстной помощи (Что это?). Эта кнопка позволяет пользователю получать контекстно-зависимую справочную информацию о компонентах, отображенных в окне.

Вторичное окно может быть независимым или модальным.

Независимое вторичное окно позволяет пользователю взаимодействовать с другими вторичными или первичными окнами, а также переключаться между первичными окнами. Независимое вторичное окно целесообразно использовать в тех ситуациях, где пользователю может потребоваться повторить действие, связанное с этим окном (например, при поиске слова в тексте или при форматировании текста).

Модальное вторичное окно требует от пользователя завершить ввод данных в пределах данного окна и закрыть его, прежде чем продолжить работу за пределами окна. Вторичное окно может быть модальным по отношению к своему первичному окну или по отношению к системе. В последнем случае пользователь должен выполнить требующиеся действия и закрыть окно прежде, чем взаимодействовать с любыми другими объектами или окнами.

Модальные вторичные окна используются только в ситуациях определенного типа:

- когда для выполнения команды требуется ввести дополнительную информацию;
- когда необходимо приостановить работу пользователя, пока не будет выполнено некоторое условие.

Избегайте использования системных модальных вторичных окон, если ваше приложение не исполняется в качестве системной утилиты; но даже и в этом случае применяйте их только в наиболее серьезных ситуациях, игнорирование которых может привести к фатальной системной ошибке.

При выборе расположения вторичного окна на экране следует учитывать большое число факторов: назначение окна, причину его появления, размеры экрана и т.д.

Вторичное окно следует отображать в той позиции, где оно появлялось в последний раз.

При первом открытии окна установите его в позиции, удобной для работы пользователя (окно должно отображаться полностью!).

Удобно располагать вторичное окно таким образом, чтобы оно находилось в центре первичного окна по горизонтали и ниже заголовка окна, меню и всех панелей инструментов.

Виды вторичных окон.

1. Панель свойств – наиболее универсальное средство представления свойств объекта. Представляет собой независимое вторичное окно, которое отображает доступные пользователю свойства объекта, причем необязательно пользователю должно быть предоставлено право изменять их. Панель свойств отображается на экране по команде Свойства для конкретного (выбранного) объекта. Стандартные кнопки панели свойств – ОК, Отменить, Применить.
2. Панель контроля параметров. Реализуется в виде модального вторичного окна, связанного с тем объектом, свойства которого она отображает. Панель контроля параметров всегда относится к выбранному в данный момент объекту. При изменении параметров свойства применяются динамически – не требуется кнопок для подтверждения или отказа.
3. Диалоговые панели обеспечивают обмен информацией или ведение диалога между пользователем и приложением. Используются для получения от пользователей дополнительной информации, необходимой для выполнения команды или задания. Название диалоговой панели должно отражать название команды.
4. Окно Палитра – является независимым вторичным окном, которое содержит набор взаимосвязанных элементов управления. В виде такого окна может быть представлена панель инструментов или ее часть. Каждое окно Палитра может иметь собственное название, отображаемое в полосе заголовка, и собственный формат. Полоса заголовка содержит только одну кнопку Закрыть.
5. Окно Сообщение – предназначено для вывода на экран сообщений пользователю; обычно это информация о конкретной ситуации или условиях выполнения операций. Окна сообщений содержат графический символ, который указывает на тип выводимого сообщения, и собственно текст сообщения.
6. Всплывающие окна – используются для отображения дополнительной информации в тех случаях, когда в основном окне она представлена в

сокращенной форме; для вывода контекстно-зависимой справочной информации. Всплывающая подсказка – разновидность всплывающего окна – используется для пояснений к элементам управления панели инструментов.

Многодокументный интерфейс (MDI).

В процессе работы с одним и тем же приложением пользователю может потребоваться иметь на экране несколько открытых окон, содержащих информацию различных типов, либо представляющих собой разное изображение одних и тех же данных. Для создания таких окон и управления ими существует специальная технология – многодокументный интерфейс.

Техника MDI заключается в использовании одного первичного окна, называемого *родительским* окном, которое может содержать набор взаимосвязанных с ним *дочерних* окон. Каждое дочернее окно – это также первичное окно, единственным ограничением для которого является то, что оно может появиться только в пределах родительского окна. Родительское окно обеспечивает как визуальное, так и операционное пространство для своих дочерних окон. Например, на дочернее окно обычно распространяется область действия меню родительского окна и, возможно, других элементов его интерфейса (панели инструментов, строки состояния и т.д.). Их вид может изменяться, если необходимо отразить команды и атрибуты активного дочернего окна.

Вторичные окна, такие как диалоговые панели, окна сообщений или панели свойств, появляются на экране как результат тех или иных действий пользователя в родительском или дочернем окне. Эти окна должны активизироваться и отображаться в соответствии с общими соглашениями для вторичных окон, связанных с первичным окном, даже если они относятся к дочернему окну.

Заголовок родительского окна обычно содержит пиктограмму и имя приложения или объекта, который он представляет. Заголовок дочернего окна содержит пиктограмму, представляющую тип документа или файла данных, и имя файла. Как для родительского окна, так и для всех его дочерних окон должны поддерживаться всплывающие меню; перечень команд в таком меню соответствует первичному окну.

Пользователь может активизировать MDI-приложение, либо непосредственно открыв его, либо открыв документ того типа, который поддерживается этим приложением. Если MDI-приложение активизировано посредством открытия документа, сначала открывается родительское окно, а затем внутри его рабочей области – дочернее окно, отображающее выбранный документ. Для того, чтобы упростить пользователю

открытие других документов, связанных с этим приложением, включите в его интерфейс диалоговую панель Открыть.

В том случае, когда пользователь непосредственно открывает документ за пределами родительского окна приложения, то если родительское окно уже открыто, следует создать второй экземпляр приложения (еще одно родительское окно), а не окно документа в существующем родительском окне. Хотя открытие нового дочернего окна может быть более эффективным, его появление может нарушить среду задания, уже установленную в этом родительском окне.

Поскольку дочерние окна являются разновидностью первичных окон, при их закрытии следуют соглашениям, принятым для первичных окон. Когда пользователь закрывает дочернее окно, любые несохраненные изменения должны быть обработаны в соответствии с общими соглашениями для всех первичных окон.

Приложение не должно разрешать пользователю закрыть дочернее окно, если это не позволит ему продолжить работу с приложением.

Когда пользователь закрывает родительское окно, закройте все его дочерние окна. Где возможно, сохраняйте состояние дочернего окна (размер и положение внутри родительского окна) и восстанавливайте это состояние, когда пользователь вновь открывает окно.

Технология MDI имеет свои ограничения:

- Хотя пользователь может запустить приложение, открыв документ, для того, чтобы работать с несколькими документами одновременно требуется использовать интерфейс приложения.
- Если открыто несколько файлов в пределах одного родительского окна, может быть нарушена согласованность связи между дочерними окнами и отображаемыми в них объектами (файлы не связаны между собой).
- Родительское окно в действительности не содержит объекты, представленные в дочерних окнах. Это не позволяет обеспечить эффективную непрерывную работу пользователя (После закрытия родительского окна созданная ранее рабочая среда не восстанавливается).

Перечисленные недостатки MDI могут быть преодолены за счет применения альтернативных средств, таких как Рабочие области, Рабочие книги и Проекты. Хотя эти средства реализуют однооконную модель интерфейса, тем не менее они обнаруживают целый ряд достоинств, присущих технологии MDI.

Рабочая область – контейнер.

Основное отличие Рабочей области от MDI заключается в использовании концепции объединения отображаемых объектов. Это означает, что объекты, отображаемые в Рабочей области, могут соответствовать файлам, содержащимся в одном и том же контейнере. Внешне же соответствующие им окна выглядят как дочерние окна, расположенные в пределах родительского окна.

Таким образом, концепция использования Рабочей области подобна концепции использования Рабочего стола, за исключением того, что она сама является объектом, который может быть представлен в виде пиктограммы и отображен в виде открытого окна. Чтобы окно объекта могло появиться в Рабочей области, сам объект должен входить в состав соответствующего контейнера.

Для рабочей области должна быть предусмотрена команда Сохранить все – для сохранения содержимого всех объектов области.

В настоящее время реализация механизма хранения объектов зависит от типа используемого контейнера.

Рабочая книга.

Рабочая книга – это еще один альтернативный вариант управления формой представления отображаемых данных, в основе которого лежит метафора книги или записной книжки. В Рабочей книге различные формы данных представляются как отдельные разделы в пределах одного первичного окна.

В качестве средств навигации между разделами Рабочей книги могут использоваться этикетки вкладок. Каждый раздел представляет данные, которые могли бы быть отдельным документом. Рабочая книга лучше подходит для представления таких данных, которые могут быть определенным образом упорядочены и этот порядок имеет существенное значение.

Для рабочей книги действительны те же соглашения, что и обеспечивающие связь между родительскими и дочерними окнами.

Как и для Рабочей области, должна быть предусмотрена команда Сохранить все.

Проект.

Проект реализует еще один альтернативный способ управления окном, который предусматривает возможность отображения в одном окне взаимосвязанных объектов, представленных их пиктограммами. В этом смысле Проект подобен каталогу: для выбранного объекта открывается окно, которое относится к тому же уровню, что и родительское окно. В результате, каждое дочернее окно проекта может также иметь собственную кнопку входа на панели задач.

В отличие от каталога, проект обеспечивает управление из родительского окна окнами входящих в него объектов (если открыт документ, пользователь может закрыть окно каталога; при закрытии проекта – закрываются все окна проекта).

Для различных окон проекта возможен различный набор элементов управления.

По аналогии с Рабочей областью и Рабочей книгой, Проект должен иметь команды для создания новых объектов, для их пересылки в Проект и из него, а также для сохранения любых изменений объектов, входящих в проект. Окно проекта должно содержать средства для работы с самим проектом как с объектом (в т.ч. для изменения его свойств).