

## Требования к разработке

### *Название приложения*

Используйте соответствующие названия для ваших приложений и файлов. Так, например, наименование настольного приложения должно обязательно включать название компании- заказчика.

### *Файловая структура*

Файловая структура проекта должна отражать логику, заложенную в приложение. Например, все формы содержатся в одной директории, пользовательские визуальные компоненты – в другой, классы сущностей – в третьей.

### *Структура проекта*

Каждая сущность должна быть представлена в программе как минимум одним отдельным классом. Классы должны быть небольшими, понятными и выполнять одну единственную функцию.

### *Макет и технические характеристики*

Все компоненты системы должны иметь единый согласованный внешний вид, а также следующим требованиям:

- разметка и дизайн (предпочтение отдается масштабируемой компоновке);
- должно присутствовать ограничение на минимальный размер окна;
- должна присутствовать возможность изменения размеров окна, где это необходимо;
- увеличение размеров окна должно увеличивать размер контентной части, например, таблицы с данными из БД);
- группировка элементов (в логические категории);
- использование соответствующих элементов управления (например, выпадающих списков для отображения подстановочных значений из базы данных);
- расположение и выравнивание элементов (метки, поля для ввода и т.д.);
- последовательный переход фокуса по элементам интерфейса (по нажатию клавиши TAB);
- общая компоновка логична, понятна и проста в использовании;
- соответствующий заголовок на каждом окне приложения (не должно быть значений по умолчанию типа MainWindow, Документ1 и т. п.).

### *Обратная связь с пользователем*

Уведомляйте пользователя о совершаемых им ошибках или о запрещенных в рамках задания действиях, информируйте об отсутствии результатов поиска и т.п. Окна сообщений соответствующих типов (например, ошибка, предупреждение, информация) должны отображаться с соответствующим заголовком и пиктограммой. Текст сообщения должен быть полезным и информативным, содержать полную информацию о совершенных

ошибках пользователя и порядок действий для их исправления. Также можно использовать визуальные подсказки для пользователя при вводе данных.

#### *Обработка ошибок*

Не позволяйте пользователю вводить некорректные значения в текстовые поля сущностей. Например, в случае несоответствия типа данных или размера поля введенному значению. Оповестите пользователя о совершенной им ошибке.

При возникновении непредвиденной ошибки приложение не должно аварийно завершать работу.

#### *Оформление кода*

Идентификаторы переменных, методов и классов должны отражать суть и/или цель их использования, в том числе и наименования элементов управления (например, не должно быть значений по умолчанию типа Form1, button3).

Идентификаторы должны соответствовать соглашению об именовании среды разработки.

Допустимо использование не более одной команды в строке.

#### *Комментарии*

Используйте комментарии для пояснения неочевидных фрагментов кода. Запрещено комментирование кода. Хороший код воспринимается как обычный текст. Не используйте комментарии для пояснения очевидных действий. Комментарии должны присутствовать только в местах, которые требуют дополнительного пояснения.