# **№ 8 Ресурсы, стили, триггеры, шаблоны**

## Задание

К предыдущей лабораторной №6-7 добавьте:

1. *Словарь ресурсов* (определите в нем объекты). Установите ресурс динамически и статически.
2. Создайте **MultiTrigger** *триггер*, триггер данных **DataTrigger** и триггер события **EventTrigger**. Отслеживайте изменение свойства (нулевое, max, заданное) или события и напишите код реакции (оповещение, лог в файл, анимацию и т.д.).
3. Создайте несколько *тем* на основе словарей со стилями (например, оптимистичный, розовый, оттенки-серого и т.д.) и предоставьте пользователю функцию смены темы в запущенном приложении.
4. Создайте и примените *шаблон*, изменяющий форму ЭУ вашего интерфейса.
5. Разработайте один *пользовательский ЭУ*. элемент. Типа проекта - Custom Control Library (WPF).  Используйте визуальные эффекты, анимацию, стили, графику.
6. Добавьте к своему приложению функцию *undo, redo*

## Вопросы

1. Для чего в WPF используются ресурсы? Каким образом можно определить новый ресурс и управлять им? Опишите назначение класса ResourceDictionary. Зачем каждый элемент имеет собственную коллекцию ресурсов?
2. Какая разница между статическими и динамическими ресурсами?
3. Что такое триггеры ? Для чего в WPF используются триггеры? Назовите основные типы триггеров.
4. Что такое локализация и как ее обеспечить.
5. Что такое тема? Опишите процесс создания темы на основе ресурсов и стилей.
6. Что такое шаблон и как его создать?
7. Зачем нужны пользовательские элементы управления? Как создать собственный элемент. Опишите члены класса UserControl.
8. Что такое привязка данных? В чем разница между следующими режимами привязки: OneWay, TwoWay и OneTime.
9. Объясните назначение интерфейса INotifyPropertyChanged

**1) Ресурсы в WPF**

**Назначение:** Ресурсы в WPF используются для централизованного управления и повторного использования объектов, таких как цвета, шаблоны, стили, анимации и другие объекты. Это облегчает управление изменениями в приложении и повышает его масштабируемость.

**Создание и Управление:** Ресурс можно определить в XAML с помощью тега **<ResourceDictionary>** внутри тегов **<Window.Resources>** или **<Application.Resources>**. Чтобы использовать ресурс, обычно его задают с помощью ключа (например, **x:Key="myResource"**).

**Класс ResourceDictionary:** Этот класс представляет коллекцию ресурсов, где каждый ресурс идентифицируется уникальным ключом. **ResourceDictionary** может содержать другие словари, формируя таким образом иерархию ресурсов.

**Собственная Коллекция Ресурсов:** Каждый элемент может иметь свою коллекцию ресурсов для локализации управления этими ресурсами в контексте конкретного элемента, что упрощает управление зависимостями и изоляцию компонентов.

**2) Статические и Динамические Ресурсы**

**Статические Ресурсы (StaticResource):** Значение ресурса извлекается в момент загрузки страницы и не изменяется в последующем, независимо от изменений в ресурсе.

**Динамические Ресурсы (DynamicResource):** Значение ресурса извлекается и может быть обновлено в любой момент, если сам ресурс изменится. Это полезно для тем, которые меняются во время выполнения приложения.

**3) Триггеры в WPF**

**Назначение:** Триггеры в WPF используются для выполнения действий или изменения внешнего вида элемента при наступлении определённых условий без написания дополнительного кода.

**Типы Триггеров:**

* **Property Triggers:** Реагируют на изменения свойств элементов.
* **Data Triggers:** Реагируют на изменения данных, связанных с элементами.
* **Event Triggers:** Срабатывают в ответ на события, такие как наведение мыши.
* **Multi Triggers:** Комбинируют условия нескольких триггеров.

**4) Локализация**

**Назначение:** Локализация — это процесс адаптации интерфейса приложения под разные языки и региональные настройки.

**Реализация:** Обычно используют ресурсы для хранения локализованных строк, изображений и других данных. Можно использовать **ResourceDictionary** для каждого языка и переключаться между ними в зависимости от текущих настроек локализации пользователя.

**5) Темы**

**Создание Темы:** Тема в WPF обычно включает набор стилей и шаблонов, определённых в ресурсах. Создать тему можно, описав эти стили и шаблоны в XAML и применяя их ко всему приложению или отдельным его частям через **ResourceDictionary**.

**6) Шаблоны**

**Назначение:** Шаблоны определяют визуальную структуру элемента управления. Шаблон позволяет полностью переопределить вид элемента, сохраняя его функциональность.

**Создание:** Создать шаблон можно, определив **ControlTemplate** и применив его к нужному элементу. Шаблоны часто содержат разметку XAML, которая описывает новую визуальную структуру элемента.

**7) Пользовательские Элементы Управления**

**Назначение:** Создание собственных элементов управления позволяет реализовывать уникальный функционал и внешний вид, которого нельзя достичь с помощью стандартных элементов.

**Реализация:** Создать можно через наследование от **UserControl**. Класс **UserControl** обычно содержит XAML для интерфейса и код для логики.

**8) Привязка Данных**

**Назначение:** Привязка данных в WPF используется для синхронизации данных между источником (например, моделью данных) и целевым элементом (например, интерфейсом пользователя).

**Режимы Привязки:**

* **OneWay:** Данные передаются от источника к цели.
* **TwoWay:** Данные передаются от источника к цели и обратно, что позволяет реализовать интерактивные формы.
* **OneTime:** Данные передаются один раз при инициализации цели.

**9) Интерфейс INotifyPropertyChanged**

**Назначение:** Интерфейс **INotifyPropertyChanged** используется в моделях данных для уведомления интерфейса пользователя о изменениях в данных, что позволяет автоматически обновлять интерфейс при изменении данных.

Эти темы освещают основные аспекты разработки на платформе WPF и могут значительно повысить качество и удобство взаимодействия с пользовательским интерфейсом.