==================================1=============

***Система ресурсов*** WPF представляет собой простой способ поддержания вместе набора полезных объектов, таких как наиболее часто используемые кисти, стили или шаблоны, что существенно упрощает работу с ними.

В WPF ресурсы можно определять в коде или в различных местах внутри разметки (вместе с отдельными элементами управления, внутри отдельных окон или во всем приложении).

Ресурсы обладают рядом важных преимуществ, которые перечислены ниже:

**Эффективность**

Ресурсы позволяют определять объект один раз и затем использовать его в нескольких местах внутри разметки. Это упрощает код и делает его намного эффективнее.

**Сопровождаемостъ**

Ресурсы позволяют переносить низкоуровневые детали форматирования (вроде размеров шрифтов) в центральное место, где их легко изменять. Это своего рода XAML-эквивалент создания констант в коде.

**Адаптируемость**

После отделения определенной информации от остальной части приложения и ее помещения в раздел ресурсов появляется возможность ее динамической модификации. Например, может понадобиться изменять детали ресурсов на основе пользовательских предпочтений или текущего языка.

Коллекция ресурсов

Каждый элемент включает свойство **Resources**, в котором хранится словарная коллекция ресурсов (представляющая собой экземпляр класса ResourceDictionary). Эта коллекция ресурсов может хранить объект любого типа с индексацией по строке.

Хотя каждый элемент имеет свойство Resources (которое определено в классе FrameworkElement), чаще всего ресурсы определяются на уровне окна. Причина в том, что каждый элемент имеет доступ к ресурсам из собственной коллекции ресурсов, а также к ресурсам из коллекции ресурсов всех своих родительских элементов.

=========================================2==========================

A [StaticResource](https://msdn.microsoft.com/en-gb/library/cc189045(v=vs.95).aspx) будет разрешен и назначен этому свойству во время загрузки XAML, который возникает до того, как приложение действительно будет запущено. Он будет назначен только один раз, и любые изменения в словаре ресурсов игнорируются.

A [DynamicResource](https://msdn.microsoft.com/en-us/library/vstudio/ms748942(v=vs.100).aspx) назначает объект Expression свойству во время загрузки, но на самом деле не ищет ресурс до выполнения, когда объект Expression запрашивается для стоимость. Это откладывает поиск ресурса, пока он не понадобится во время выполнения. Хорошим примером может служить прямая ссылка на ресурс, определенный позже в XAML. Другим примером является ресурс, который даже не будет существовать до выполнения. Он обновит цель, если будет изменен словарь исходного ресурса.

=====================================3============================================

Тригерры позволяют декларативно задать некоторые действия, которые выполняются при изменении свойств стиля. Существует три вида триггеров:

* **Триггеры свойств**: вызываются в ответ на изменения свойствами зависимостей своего значения
* **Триггеры данных**: вызываются в ответ на изменения значений любых свойств (они необязательно должны быть свойствами зависимостей)
* **Триггеры событий**: вызываются в ответ на генерацию событий
* **Мультитриггеры**: вызываются при выполнении ряда условий

==========================================4=========================================

==============================================5=====================================

========================================6===========================================

Все визуальные элементы в WPF уже имеют встроенные шаблоны, которые определяют визуальное дерево, структуру и даже поведение элементов. Однако мощь шаблонов состоит в том, что мы можем их переопределить по своему вкусу. Например, сделать круглое окно, а не квадратное, или кнопку в виде морской звезды.

===========================7========================================================8=============================

OneWay–целевое свойство обновляется при  
изменении значения источника.  
► TwoWay–при изменении источника меняется целевое  
свойство и наоборот.  
► OneTime–целевое свойство устанавливается  
изначально на основе свойства источника и с этого  
момента изменения значений в источнике  
игнорируются.  
► OneWayToSource–свойство источника обновляется  
при изменении целевого свойства.

INotifyPropertyChanged

Для реализации механизма привязки, надо реализовать интерфейс  
Когда объект класса изменяет  
значение свойства, то он через  
событие PropertyChanged извещае  
систему об изменении свойства. А  
система обновляет все привязанны  
объекты.

10

=========================1========================

**DependencyProperty  
Свойства зависимости**►Есть у ЭУ унаследованных от  
DependencyObject  
►Могут наследовать свои значения от  
родительского элемента  
►Позволяют вычислять значение на  
основе нескольких внешних значений  
►Используются при анимации, привязке  
данных и стилей

**1) ValidateValueCallback**: делегат - true и  
false – прошло или нет проверку – срабатывает  
первым  
**2) CoerceValueCallback**: делегат, который  
может подкорректировать уже существующее  
значение свойства, если оно вдруг не попадает  
в диапазон допустимых значени  
срабатывает вторым  
Могут использоваться вместе или поотдельности

* **Прямые** (direct events) - они возникают и отрабытывают на одном элементе и никуда дальше не передаются. Действуют как обычные события.
* **Поднимающиеся** (bubbling events) - возникают на одном элементе, а потом передаются дальше к родителю - элементу-контейнеру и далее, пока не достигнет наивысшего родителя в дереве элементов.
* **Опускающиеся, туннельные** (tunneling events) - начинает отрабатывать в корневом элементе окна приложения и идет далее по вложенным элементам, пока не достигнет элемента, вызвавшего это событие.

Паттерн Command  
► Суть: Представить запрос как ООП объект (не метод)  
♣ Конфигурация команды  
♣ Для определения и выполнения в разное время: создания  
очереди, stop и ….  
♣ undo, redo  
♣ Протоколирования и структурирования системы

Команда  
► **Команды** - механизм выполнения  
задачи (паттерн “Команда” Command)  
Назначение:  
- использовать одну и ту же команду для  
нескольких ЭУ  
- абстрагировать набор действий от  
конкретных событий конкретных  
элементов