# AL-JUNAID TECH INSTITUTE

## CS411 IMPORTANT TOPIC

Write Basic Touch events.  Lecture No 22

Let's Revise Grid (GRID) Lecture No 20

Dock panel Button Lecture No 19

List Box Lecture No 17

Write the Code of Button in XML Lecture No 13

Write a Name Windows Form Lecture No 8

Delegating Lecture No 8

Feature of C#  Lecture 4

## Important for MCQS Lecture

## 12, 15,1,5,4,3,2,19

graphical user interfaces and the event-driven model as applied to desktop. The pre-requisites are C++ programming and data structures

We will primarily use 2 books in this course. "Event processing in action" and "Windows resentation foundationunleashed". The first one is authored by Opher Etzion and Peter Niblett and is one of the very fewbooks on event processing. Event processing is often taught as a side-concept, but we will focus on it in this course. The other book is by Adam Nathan. It is about Windows presentation foundation or WPF in short. We use it as our gui example with C#. We will introduce C# concepts and WPF concepts. They are not a pre-requisite for this course.

Later in the course, we will touch event-driven programming in the browser, in particular made possible by techniques called AJAX. Again not a pre-requisite. I will introduce all the concepts in class. Towards the very end, we will discuss event driven programming on mobiles. Again no pre-requisite. I will introduce all the tools involved.

So what is Visual Programming and what you will learn in this course. Its primarily event driven concepts and application of these concepts on GUI programs on desktop, web, and mobile.

So what is an event? It's an occurrence within a particular system or domain. There are two meanings: something that happened and the corresponding detection in computer world. An event captures "some" things from the actual occurrence and multiple events may capture "one" occurrence. Probabilistic events may or may not relate to an actual occurrence e.g. A fraud detection event on a banking transaction. Every event is represented by an event-object. There are various types of events and information in the event describes details of the particular event type. E.g. Key press, file event etc.

Events are based on the principle of decoupling Events have already happened whereas requests are asking something to happen. r. In event-driven architecture event producer sends event to event consumer. A customer order can be represented as an event or a request: what are the benefits of each approach? Event-based programming, also called event-driven architecture (EDA) is an architectural style in which one or more components in a software system execute in response to receiving one or more event notifications. Service-oriented architecture (SOA) is built from request-response. It moves away from monolithic applications.



EVENT PRODUCER: An event producer is an entity at the edge of an event process- ing system that introduces events into the system.

EVENT CONSUMER: An event consumer is an entity at the edge of an event pro- cessing system that receives events from the system.

RAW EVENT: A raw event is an event that is introduced into an event processing system by an event producer. DERIVED EVENT: A derived event is an event that is generated as a result of event processing that takes place inside an event processing system.

STATELESS EVENT PROCESSING An event processing agent is said to be stateless if the way it processes one event does not influence the way it processes any subsequent events.

Let's take example of events in computing systems. Interrupts and exceptions on a computer e.g. Divide by zero. Patient monitored by sensors. Car sensors alerting to oil or pressure situations. Banking alerts. Road tolling. Etc. Event processing is computing that performs operations on events. Common event processing operations include reading, creating, transforming, and deleting events. The design, coding and operation of applications that use events, either directly or indirectly is called event-based programming or applications based on event-driven architecture. So what if we don't use event-driven programming. We will poll for events. Even if you make your application non-event-driven, you still wait for events. Wait for a single event is blocking operation.

Synchronous operations are completed before the next operation can be started. Asynchronous operations can be started and we can do something else before they are completed.

Why do we want our applications to be event based. They are easier to scale. Well suited to Visual programming where multiple GUI elements and many sources of events exist. It has a direct mapping to real

EVENT STREAM A stream in which all the events must be of the same type is called a homogeneous event streamt. Each attribute has a name and a data type.

Clean room design (also known as the Chinese wall technique) is the method of copying a design by reverse engineerin,. The initial name was "Cool", which stood for "C-like Object Oriented Language"

Here are some notable features of C#.

• No global variables or functions

. • Locals cannot shadow global variables

. • There is a strict boolean type.

• Memory address pointers can only be used in specifically marked "unsafe" blocks and require permissions

• No instruction to "free" memory. Only garbage collection.

• Try-finally block

The common type system of C# has value types and reference types . default value (int, float, char, System.datetime, enum, struct) Not "always" possible to create an instance of a reference type.

piler expects to have available. In practice, C# is most often used with some implementation of the Common Language Infrastructure (CLI), which is standardized as ECMA-335 Common Language Infrastructure (CLI).

# AL-JUNAID TECH INSTITUTE

Here are the integer types in C#.

| Type | Size (in bits) | Range |
|---|---|---|
| sbyte | 8 | -128 to 127 |
| byte | 8 | 0 to 255 |
| short | 16 | -32768 to 32767 |
| ushort | 16 | 0 to 65535 |
| int | 32 | -2147483648 to 2147483647 |
| uint | 32 | 0 to 4294967295 |
| long | 64 | -9223372036854775808 to 9223372036854775807 |
| ulong | 64 | 0 to 18446744073709551615 |
| char | 16 | 0 to 65535 |

The System.String type supports the following sequences.

| Char | Meaning | Value |
|---|---|---|
| \' | Single quote | 0x0027 |
| \" | Double quote | 0x0022 |
| \\ | Backslash | 0x005C |
| \0 | Null | 0x0000 |
| \a | Alert | 0x0007 |
| \b | Backspace | 0x0008 |
| \f | Form feed | 0x000C |
| \n | New line | 0x000A |
| \r | Carriage return | 0x000D |
| \t | Horizontal tab | 0x0009 |
| \v | Vertical tab | 0x000B |

jagged arrays or multi-dimensional arrays. Jagged is basically array of arrays. Array size is any integer type value. It uses a zero-based index.

The "this" pointer in methods refers to the object on which the method is called. Parameters can be "ref" parameter which are passed by reference, "out" parameters which are used for return values or the "params" argument for variable arguments.

Calling this namespace from within testnamespace we use “tutorial.” Using theexample = testnamespace.tutorial.myexample.

Multiple classes are conventionally stored in multiple files. To create a new class in IDE, use the new class wizard. Multiple .cs files can be compiled as csc a.cs b.cs c.cs. Default ctors are written with no arguments when no ctor is written. An initializer list can be used to use an alternate constructor. Public outputclass() : this("Default Constructor String") { } Multiple ctors can be written. Types of class members in C# are instance and static

Static ctor exists to initialize class static members. Its called only once. It has no parameters. Destructors (dtors) are called by garbage collector.

Cast back to base type to call an overridden method of base.

Auto-implemented properties improve the common-case we saw. Here is the same example using auotimplemented properties. Properties have the same idea as getters and setters.

An indexer enables your class to be treated like an array. However you have your internal data representation not an actual array. Its implemented using “this” keyword and square brackets syntax

Our next topic is a “struct”. Its a value-type, whereas class is a reference-type. Value types hold their value in memory where they are declared Structs cant have destructors, cant have implementation inheritance. Syntax of struct and class are very similar. They are inherited by classes which provide the real implementation. So, what are interfaces good for if they don’t implement functionality? They are great for putting together plug-n-play like architectures. Interfaces define a contract. A delegate is a reference to a method. Its like function pointers in some other languages. Methods are algorithms that operate on data. A C# event is a class member that is activated whenever the event it was designed for occurs (fires). At the time an event fires, methods registered with the event will be invoked. Events and delegates work hand in hand. Any class, including the same class that the event is declared in, may register one of its methods with the event. Form, which essen-tially makes it a Windows Form. This automatically gives you all the functionality of a Windows Form, including Title.Bar, Minimize/Maximize/Close buttons, System Menu, and Borders. It is started by calling the Run() method of the static Application object with a reference to the form object as its parameter

They have a message property, contain a stacktrace, a tostring method. Identifying the exceptions you’ll need to handle depends on the routine . Attributes add declarative information to your programs. They are declared with square brackets, “[” and “]” surrounding the attribute such as “ [obsoleteattribute]”. The ”Attribute” part of the attribute name is optional i.e. “[Obsolete]” is correct as well.

Stathread is a common attribute you will see later. It stands for Single Threaded Apartment model which is used for communicating with unmanaged COM. Attribute parameters can be either positional parameters or named parameters.

Enums (or enumerations) are strongly typed constants. They are unique types that allow you to assign symbolic names to integral values. Enum of one type may not be implicitly assigned to an enum of another type (even though the underlying value of their members is the same). All assignments between different enum types and integral types require an explicit cast. It allows you to work with integral values, but using a meaningful name. North, South, East, and West or the zet of integers 0, 1, 2, and 3. C# type, enum, inherits the Base Class Library (BCL) type, Enum.

Overloaded operators must be static.

Let's discuss Anonymous methods. It is used with delegates and handlers and events. Anonymous methods result in much less code. Anonymous method is a method without a name. You don't declare anonymous methods, Instead they get hooked up directly to events. With delegates 1) you declare the delegate, 2) write a method with a signature defined by the delegate interface, 3) declare the event based on that delegate, and 4) write code to hook the handler method up to the delegate. To delcare an anonymous method, you just use keyword "delegate" followed by method body.

Breakpoints allow stopping the program during execution. Press F5 and execution will stop at breakpoint. s XML. XML or extensible Markup Language is widely used for exchanging data. Its readable for both humans and machines. Its a stricter version of HTML. There are two methods to read XML document. Using xmldocument and xmlreader. Xmldocuments reads entire document in memory, Let's you go forward, backward, even apply xpath searches on it. Xmlreader is fast, uses less memory and provides one element at a time.

xmlnode is derived from xmlelement and contains Name, innertext, innerxml, outerxml, and Attributes. Xpath is a cross-platform Xml Query language. We will look at basic examples only. We will see xmldocument methods that take xpath queries and return xmlnode(s) in particular selectsinglenode and selectnodes. We will use the RSS feed from CNN.

## Chapter:12

We will start discussing WPF (Windows Presentation Foundations) now. It was publicly announced in 2003 (codenamed Avalon). WPF 4 was released in April 2010. It has a steep learning curve. Code has to be written in many places. There are multiple ways to do a particular task.

WPF enables polished user interfaces which are getting a lot of attention. It enables rapid iterations and major interface changes throughout the development process. It allows to keep user interface description and implementation separate. Developers can create an "ugly" application which designers can re-theme. Win32 style of programming makes such re-theming difficult. The code to re-paint the user interface is mixed up with program logic. GDI was an earlier user interface library introduced in windows 1.0 in 1985. Opengl was a leap ahead introduced in the 90's with directx coming in 95 and directx 2 in 96. GDI+ is a newer user interface library based on directx. It is also used behind Xbox graphics. Next was Windows

Application Markup Language (XAML; pronounced "Zammel").

# AL-JUNAID TECH INSTITUTE

In short, WPF aims to combine the best attributes of systems such as directx (3D and hardware acceleration), Windows Forms (developer productivity), Adobe Flash (powerful animation support) and HTML (declarative markup). The first release in November 2006 was WPF 3.0 because it shipped as part of the .NET Framework 3.0. WPF 3.5 came an year later. Next version as part of .NET 3.5 SP1 came in August 2008. WPF Toolkit released in Aug 2008 was experimental. The toolkit has quick releases. Regarding tool support, WPF extensions for Visual Studio 2005 came a few months after the first WPF release and a public release of expression blend. Now, Visual Studio 2012 is a first class WPF devevelopment environment. Its mostly re-written using WPF and expression blend is 100% WPF and is great for designing and prototyping WPF apps.

New things that came in WPF 3.5/3.5SP1 include Interactive3D with 2d elements in 3d scenes, first class interoperability with directx, Better data binding using XLINQ and better validation and debugging which reduces code, Better special effects, High performance custom drawing, Text improvements, enhancements to Partial-trust apps, improved deployment, and improved performance.

Things that came with WPF 4.0 include multi-touch support — compatible with surface API v2, win7 support like jump lists, new common dialogs etc., new controls like datagrid, calendar etc, easing animation functions (bounce, elastic), enhanced styling with visual state manager, improved layout on pixel boundaries, non-blurry text but some limitations so must opt-in, deployment improvements, and performance improvements.

Silverlight in comparison is a light-weight version for web. It chose to follow WPF appraoch. First released in 2007 and in April 2010 4th version was released near WPF 4. There is often confusion when to use one or

XAML is primarily used to describe interfaces in WPF and Silverlight. workflow foundation (WF) and windows communication foundation (WCF). Field experts are graphic designers. They can use a design tool such as expression blend. Other than co-ordinating with designers, XAML is good for a concise way to represent UI or hierarchies.

## Lecture 13

< listbox xmlns="
http://schemas.microsoft.com/winfx/2006/xaml/presentation"/>

Write the Code of Button in XML

XAML specification defines rules that map .NET namespaces, types, properties, and events into XML namespaces, elements, and attributes. Let's see XAML and equivalent C#.

```
<Button xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" Content="OK"/>
```

And the corresponding C# code is

```
System.Windows.Controls.Button b= new System.Windows.Controls.Button();
B.Content= "OK";
```

Declaring an XML element in XAML (known as an object element) is equivalent to instantiating the

XAML can no longer run standalone in the browser becuase of the butto click method — event handlers are attached before properties are set. in XAML must specify at least one XML namespace.

And in XAML

```
<Button xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
<Button.Content>
<Rectangle  Height="40" Width="40" Fill="Black"/>
</Button.Content ></Button >
```

The period distinguishes property elements from object elements. It doesn't have attributes except x:uid for localization. The syntax can be used for simple properties as well. The following two are equivalent.

```
<Button xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" Content="OK"
Backgroun
<Button xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
<Button.Content>OK</Button.Content >
<Button.Background>White</Button.Background >
</Button >
```

WPF provides type converters for many common data types: Brush, Color, fontweight, Point, and so on.

You can also write your own type converters for custom data types

```
<Button xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation" Content="OK">
<Button.Background>
<solidcolorbrush  Color="White"/>
</Button.Background >
</Button>
```

XAML enables you to add items to the two main types of collections that support indexing: lists and dictionaries.

```
<listbox xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation">
<listbox.Items>
<listboxitem Content="Item 1"/>
<listboxitem Content="Item 2"/> </listbox.Items>
</listbox>
```

Otherwise, if the child is plain text and a type converter exists to transform the child into the parent type (and no properties are set on the parent element), treat the child as the input to the type converter and use the output as the parent object instance

# AL-JUNAID TECH INSTITUTE

## Lecture 15

BAML is binary application markup language. It just a compressed representation of XAML. There is even a BAML reader available. Earlier there was CAML which stands for compiled application markup language but its not used now.

Some glue code is generated when we use x:Class. Its kind of same as loading and parsing the XAML file. We must call initializecomponent and we can refer named elements like class members.

Procedural code can even be written inside the XAML file.

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
xmlns:x="http://schemas.microsoft.com/winfx/2006/xaml" x:Class="mynamespace.mywindow">
    <Button Click="button_Click">OK</Button>
```

XAML 2006 keywords include x:asyncrecords - Controls the size of asynchronous XAML-loading chunks, x:Class, x:classmodifier - visibility, public by default, x:Code, x:connectionid - not for public use, x:fieldmodifier - field visibility (internal def), x:Key, x:Name, x:Shared - =false means same resource instance not shared, x:subclass - only needed when partial not supported, x:Synchronous mode - xaml loaded in async mode, x:typearguments - used only with root with x:Class in xaml2006, x:Uid - represents system.uri, and x:xdata - data opaque for xaml parser.

Freezable is the base class for objects that can be "frozen" into a read-only state for performance reasons. Freezables, once frozen, can be safely shared among multiple threads, unlike all other dispatcherobjects. Frozen objects can never be unfrozen, but you can clone them to create unfrozen copies. Most Freezables are graphics primitives such as brushes, pens, and geometries or animation classes.

Xaml good for UI because of hierarchical nature. Logical tree exists even if there is no xaml.

## Chapter:16

Dependency properties are represented by System.Windows.dependencyproperty. Visual Studio has a snippet called propdp.

Dependency properties support change notification. Its based on metadata at register time. Actions can be re-rendering the appropriate elements, updating the current layout, refreshing data bindings, and much property triggers - imagine you want color change on hovering.

```
<Button  mouseenter="Button_mouseenter" mouseleave="Button_mouseleave" minwidth="75"
Margin="10">
<Button  mouseenter="Button_mouseenter" mouseleave="Button_mouseleave" minwidth="75"
```

arbitrary objects. Sounds strange but there are many applications for it. There is new xaml syntax for attached props.

```
<Window xmlns="http://schemas.microsoft.com/winfx/2006/xaml/presentation"
```

## Chapter:17

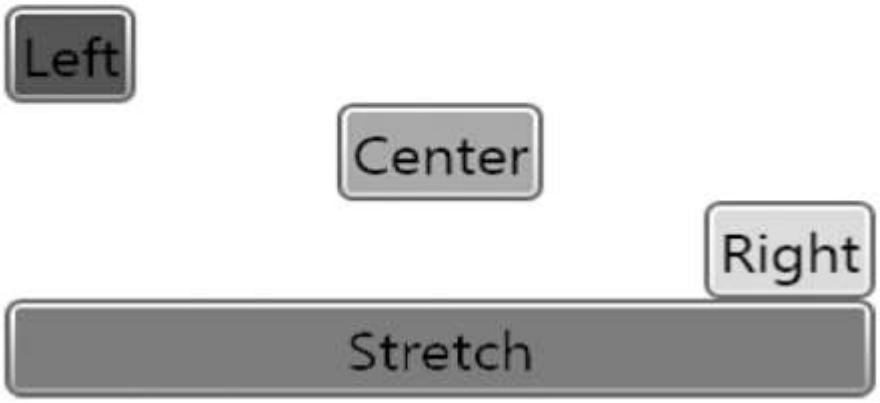
Children tell their size to parents. . Min=0 and max=Infinity by default. Nan, Auto, Double.isnan mean size to content. Read only properties desiredsize, rendersize, actualheight, actualwidth.

They are useful to act programatically on final size. Margin is extra space “outside” and padding is “inside” edges of the element.

The lengthconverter type converter associated with the various length properties supports specifying explicit units of cm, pt, in, or px (the default). Default are device independent pixels. These logical pixels are 1/96 in regardless of screen DPI setting. They are always stored as “double”. A typical display is 96 DPI.

Center, Right, and Stretch and verticalalignment can take values Top, Center, Bottom, and Stretch.

```
<stackpanel>
<Button  horizontalalignment="Left" Background="Red">Left</Button>
<Button  horizontalalignment="Center" Background="Orange">Center</Button>
<Button  horizontalalignment="Right" Background="Yellow">Right</Button>
<Button  horizontalalignment="Stretch" Background="Lime">Stretch</Button>
</stackpanel >
```




Transforms are from the base class System.Windows.Media.Transform.

**Chapter:18**

Matrix Transform has a single Matrix property (of type System.Windows.Media.Matrix)

5 main builtin panels in System.Windows.Controls: Canvas, stackpanel, wrappanel, dockpanel, and Grid.

```
<Canvas>
   <Button  Background="Red">Left=0, Top=0</Button>
   <Button Canvas.Left="18" Canvas.Top="18"
   Background="Orange"> Left=18, Top=18</Button>
   <Button Canvas.Right="18" Canvas.Bottom="18"
   Background="Yellow">Right=18, Bottom=18</Button>
```





```
   <Button Canvas.Right="0" Canvas.Bottom="0"
   Background="Lime">Right=0, Bottom=0</Button>
   <Button Canvas.Right="0" Canvas.Top="0"
   Background="Aqua">Right=0, Top=0</Button>
   <Button Canvas.Left="0" Canvas.Bottom="0"
   Background="Magenta">Left=0, Bottom=0</Button>
</Canvas ></Window >
```

## Lecture 19

Dock panel allows easy docking of elements to an entire side. Dock attached property has left, right, top, bottom values. Last child fills space unless lastchildfill=false.

```
<dockpanel>
  <Button dockpanel.Dock="Top" Background="Red">1(Top)</Button>
  <Button dockpanel.Dock="Left" Background="Orange">2 (Left)</Button>
  <Button dockpanel.Dock="Right" Background="Yellow">3 (Right)</Button>
  <Button dockpanel.Dock="Bottom" Background="Lime">4 (Bottom)</Button>
  <Button  Background="Aqua">5</Button>
</dockpanel >
```

Online article list too small and and start page label not full width. We can make spanning rows and cols. Rowspan and columnspan 1 by default. By default height n width are same. Height and Width = "Auto" for sizing to content. Showgridlines = "True" to show grid lines.

For sizing rows and columns, use rowdefinition and Col. Height and Width not double but gridlength and not default to Auto or nan. Three types of sizing. Absoulte sizing: device independent pixels means no grow shrink, Autosizing: size to content, or Proportional or star sizing: grows or shrinks. When 1 row col * all remaining space is taken by it. When more row col * they divides remaining space. It can an be 2* or

# AL-JUNAID TECH INSTITUTE

## Chapter 20

### Lecture 20

Let's revise grid.

```
<Grid>
  <Grid.columndefinitions>
    <columndefinition  Width="Auto"/>
    <columndefinition/>
    <columndefinition/>
  </Grid.columndefinitions >
  <Label Grid.Column="0" Background="Red" horizontalcontentalignment="Center"
verticalcontenta=""
```

Sharedsizegroup enables multiple row cols to remain the same width height when length changed via gridsplitter.

Content overflow can be dealt with Clipping, Scrolling, Scaling, Wrapping and Trimming. Wrapping already seen. Only way for non-text to wrap is using wrappanel. Trimming is intelligent form of clipping.

System.Windows.Controls.Viewbox, a type of class called Decorator (also Border). A panel-like thing but has only one child. Stretches to fill available space by default but also Stretch=None (like not using it at all), Fill, Uniform (aspect ratio, default), uniformtofill (cropped) strechdirection=uponly, downonly, Both (default).

Routing Strategies include Tunneling: from root down to source or until marked handled, Bubbling: from source to root or until marked handled, and Direct: only on source just like .net events but still participate in routed event specific things like event triggers.

There aer many keyboard, mouse, multi-touch, and stylus events. Most are bubbling events but many have a tunneling counterpart. By convention tunneling event names are prefixed with Preview and come just before the bubbling event comes.

Basic touch events are touchenter and touchleave, touchmove and previewtouchmove, touchdown, touchup, previewtouchdown and previewtouchu, gottouchcapture and losttouchcapture. With multiple fingers, events raised for each finger separately. For first finger mouse events are generated as well. Toucheventargs has gettouchpoint, getintermediatetouchpoints, touchdevice. Touchpoint has Position, Size, Bounds, touchdevice, Action (Down, Up, Move). Each finger has its own touchdevice identified by Id porp.

# AL-JUNAID TECH INSTITUTE