**Event Driven Programming**

Event driven programming is a programming model in which flow of the program is determined by the events that occur in the system. There are few ways that events can happen some of them may be event listeners which would be mouse hover over something or keyboard inputs but events can be created not only by the user and this is very important part in event driven programming. Events can also be created without user interaction, they can be created by other things for example software in our personal computer or even other hardware. So basic thing in event driven programming are events and the way how the program will respond for them. This programming model is very popular because it can provide very high responsiveness for users inputs which directly translates to better user experience. Event driven programs could be written in any language but there are few languages that are specially designed for this type of programming, those would be languages like Visual Basic for example.

Below we can see few of event groups that can occur:

* Component events
  + This type of event is triggered when the instance of component is created. Components events are usually used in the system to communicate between multiple components.
  + User interaction with the web page can be used with component events. This would be clicking links on the web page or submitting forms.
* Hardware
  + Hardware events occur when user clicks a button on the keyboard or uses his mouse.
  + This can be printer connected to the computer and sending information that the document was successfully printed.
  + External speakers can have volume control know that controls the volume and some application can use this to respond to change the value and respectively would lower the volume or make it louder.
* Keyboard
  + Keyboard events are really easy to visualise and there are three types off keyboard events:
    - “KeyDown” event, which tells us when the key is pressed, and it can occur once.
    - “KeyPress” event, which would tell us that the button is pressed and this event could occur multiple times as the user, holds down the button.
    - “KeyUp” event tells us that the user released the button and this event is occurring only once like in key down event.
* Mouse
  + Mouse events occur when user interacts with mouse and there is few events triggering:
    - It could be mouse down, mouse up or simple click event
    - This could be double click
    - Mouse move is event as well
    - Mouse over and mouse out
* Sensor
  + Those events are very popular in mobile technology, there are many examples of sensor events and the basic events could be:
    - Using gyroscope in our smart phone
    - Or maybe map application constantly updating our GPS position
    - Recent updates on android let user choose that phone response to level of light that is read by light sensor and then can adjust brightness of our screen, which means at night our phone is not so bright and in a sunny day it would use 100% of brightness.
* System
  + System events are events coming from activities in our operating system and they help us understand of current state of our system. This could be:
    - System restarting
    - System shutting down
    - Or maybe system updating
* Touchscreen
  + Touchscreen events are relatively new type of events and mostly used in tablets and smart phones to provide response for touch based user interfaces. It can be touch with a finger or special “stylus” pen recognized by the device. In this type of events are also few states:
    - Touch start
    - Touch end
    - Touch move
    - Or touch cancel
* Windows
  + Windows events are mostly events triggered by the errors or warnings occurred in the system. Most common events are:
    - Error
    - Warning
    - Information

Discuss the following:

* Hardware Interrupts
  + Hardware interrupts is method that devices are using when they would need to tell the operating system about occurred event. Interrupts can be used to make asynchronous events. For example when writing to hard drive we could see notification that the device memory is full, that would be hard drive sending notification to the system. Referencing the interrupt is done by interrupt number which can be tracked back to the hardware that created the interruption.
* Polling
  + Event polling could be compared to gathering all the users inputs into a list called “event queue”. Polling is also continuously checking for change in state of programs or devices. For example multipoint communication where one main device controls other devices and sends a message to each one of them asking about any change.
* Event Dispatcher
  + The dispatcher is and object that holds a registry of event listeners and when some event is dispatched it also notifies all listeners registered to that event.
* Event listener
  + Event listener is an object that handles the event dispatched by event dispatcher. Listeners are programmed to respond to an input by calling event handler.
* Event handler (callback)
  + Event handler is a routine responsible for dealing with events and programmers can write code to handle the event when it occurs.

There are of course advantages of using event driven programming as well as disadvantages:

* Pros:
  + Event driven programming is very flexible and it follows logical order from start to finish. Using language like visual basic.net developers get help while writing the code and can see suggestions as the write. Its strictly drive on events so it is really easy to extend the functionality by adding new events to the system.
* Cons:
  + Event driven program constantly runs in background so it is taking up resources of the machine on which it is running, that mean we would need higher spec machine for all those tasks end events running in a loop asynchronously. EDP language is difficult to translate to different programming languages and in some cases even impossible.

Word Count: 950