# Principles, characteristics and features of Event driven Programming

### Principles of Event Driven Programming

Event driven programming is a standard in which the operation and general functioning of the program is determined by direct interaction by outside elements e.g. user input, output from other programs/threads and even third-party interactions. Event driven programming is the dominant standard used in graphical user interfaces (GUIs) and other applications (e.g. web applications) that are focused on activating specific actions in response to inputs.

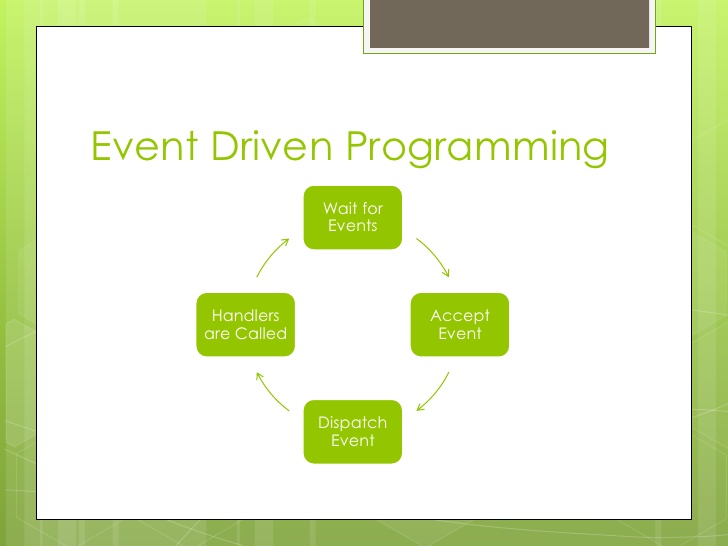
### Characteristics

The defining characteristic of an event driven program is ultimately how the program handles user input.

An Event driven program spends the majority of its time idle while waiting for user interaction to control program execution. The user triggers an event by clicking on graphical interface controls; each event is tied to an event handler that is associated with a block of code. Needless to say, an event driven program is perpetually responding to events rather than making things happen.

A sequential program differs from an event driven program in that it performs the intended actions and then closes. There may be conditional statements, or events that require the program to pause while waiting for user input; but ultimately execution of the program can be followed from beginning to end. This type of program will typically display a menu and then wait until the user selects an option, then the program will display another menu and wait again until the user selects an option etc.

### Features of Event Driven Programming



##### Source = [Examples of Event driven programming](http://gallerygogopix.net/examples+event+driven+programming+languages)

The main features of event driven Programming are: service orientated, time driven event handlers, trigger functions, events utilising mouse and keyboard, as well as flexibility and suitability for graphical interfaces. In addition this style of programming offers simplicity and ease of development which procedural programming cannot match.

### Advantages

Event driven programming offers several advantages over alternative styles of programming. It allows for more interactive programs and as a result almost all modern GUI based programming languages use event driven programming. In addition it can be implemented in a manner that can facilitate hardware interrupts thus reducing CPU time as well as aid in hardware peripherals to more easily interact with software.

### Disadvantages

Event driven programming offers several disadvantages over procedural programming languages. This includes overcomplicating simple programs by being more complex and cumbersome than more streamlined procedural alternatives. In addition to this, the flow of a program created using event driven programming can be less logical and understandable for inexperienced programmers. Finally, event driven programming is generally only useful when designing GUI programs.

### Examples

While the style of event driven programming can be completed in any language, with enough dedication and forethought; it is primarily associated with the following languages:  
VB.Net  
C#

A real world example of event driven programming is the microwave. It is possible to see this as when the microwave has no inputs it simply sits there waiting for user interaction. However, once the microwave has been given an input: in this case a time duration or a cooking program; it gets to work. In addition, the microwave contains many different buttons with many different options making it a particularly suitable example of event driven programming.