Features of event driven programming

Service orientated

Service-orientated techniques are used so that different applications are able to reuse components. Different types of software applications can benefit from these services, for example so they can request data from a database.

Time driven

Time driven programming is a type of computer programming that is frequently used in real-time computing, where code execution is managed by the computer clock. This programming concept is mainly used for safety-critical programs, because the performance of the program is highly deterministic.

Event handlers

Event handlers are made in code form and they determine what is to happen if a user is to (for example) make an action with the mouse by clicking a specific button. One the button is clicked by the mouse the event handles the action by carrying it out and actually making the event happen.

Trigger functions

When using event driven programming, trigger functions are used to select which type of event handler should be run according to the event occurring.

Events

An event is the result of an action. There are loads of events that trigger the event handlers to run. For example:

* A mouse clicks events include left click, double left click, right click and hover
* Keyboard events include key press, key down and key up
* HTML object events happens when a user clicks on an object that may be connected to a linked web page
* Form events have to do with the load event when the form shows up for the first time and the activate event used when a form regains focus. Form events also triggers event handlers intended for objects on the form for instance, defining the accept button code for when the user uses the enter key on a form.
* A user interface event happens with anything that the user does with a GUI operating system or to the form of a running program.

Event loops

Event loops are needed so that they can continuously test the user interface to detect whether anything has happened, for instance clicking on a button or typing into a text box. If events are detected, the event is then passed to the trigger functions which would then call the best event handler that is suitable to run any code in the program that’s designed and written for the event.

Forms

Forms are a major feature and are used to hold all the controls the programmer used to create programs. Forms can also be a type of control that has a collection of events that may be used by the programmer. Using these forms can be a strong way to control a program mostly when a program loads and ends. One example of a form event is **Load**, which occurs when a form is first used. The load event is useful for code in order to set variables, to be able to set default values and other matters that is needed to be done just once. Another example of a form event is **activated** which occurs whenever the form has received focus (is brought up). An activated event is used as an advantage for updating a form whenever a user returns to it from another form.

Flexibility

Flexibility is an advantage for a programmer as they have control over codes and how they can start them. Each object has choices of events that the program can respond to. The events allow you to have good control over what the program will respond to when the user does something. For example when using a text box, there is usually a choice of events that is given so that fine programming has good control over anything typed into it.

Suitability for graphical interfaces

Event driven programs are suitable for graphical user interfaces when paired as they work well with each other. A GUI allows the users to have a good variety of graphical choices and menus that are used with the mouse or keyboard or any other input device, with no set sequence that the user has to follow. Controls used will react to events with code that is suitable for each event and for how the program is likely to be used.

Simplicity of programming

Event driven programming language is able to make programming seem straightforward compared to traditional flow driven languages, because the programming language is very visual.

Ease of development

Program development can be easy with an event driven language as the programmer would only need to work on one event of one control at a time. Flow driven languages must be complete before they can run and be tested, whereas an event driven language will run with as little as only the code for a single event completed.

Disadvantages

Some event driven languages have long and complicated commands that programmers would have to use, and as a downside, sometimes programmers would need to type them all up. Testing would be essential in order to make sure the program works as normal. Also event driven languages can be quite slow due to the fact that processing power gets soaked up by event loops, and more processing power is taken up by the trigger functions.