Types of Languages

# Satisfactory

**Imperative**: In computer science, imperative programming is a programming paradigm that describes computation in terms of statements that change a program state. In much the same way that imperative mood in natural languages expresses commands to take action, imperative programs define sequences of commands for the computer to perform.

**Declarative**: In computer science, declarative programming is a programming paradigm, a style of building the structure and elements of computer programs, that expresses the logic of a computation without describing its control flow.

**Functional**: In computer science, functional programming is a programming paradigm, a style of building the structure and elements of computer programs, that treats computation as the evaluation of mathematical functions and avoids state and mutable data. Functional programming emphasizes functions that produce results that depend only on their inputs and not on the program state—i.e. pure mathematical functions. It is a declarative programming paradigm, which means programming is done with expressions.

**Logic**: Logic programming is a programming paradigm based on formal logic. Programs written in a logical programming language are sets of logical sentences, expressing facts and rules about some problem domain. Together with an inference algorithm, they form a program. Major logic programming languages include Prolog and Datalog.

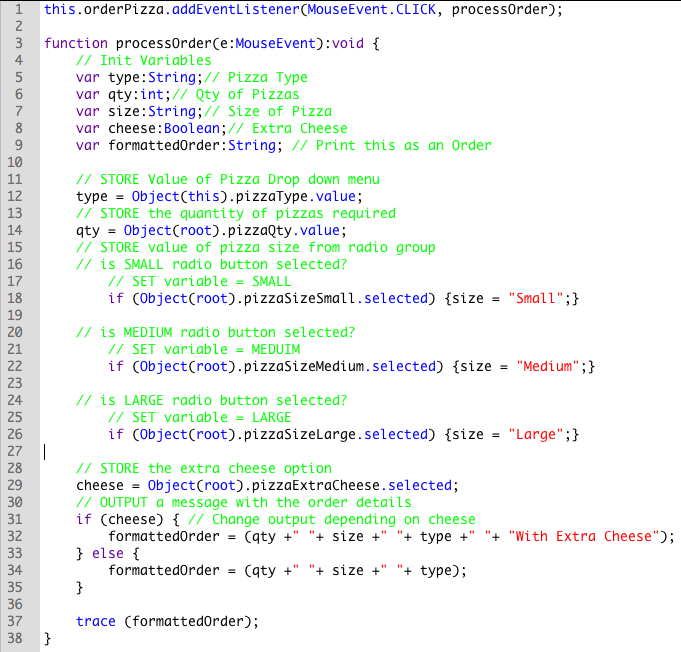
**Procedural**: Procedural programming can sometimes be used as a synonym for imperative programming (specifying the steps the program must take to reach the desired state), but can also refer (as in this article) to a programming paradigm, derived from structured programming, based upon the concept of the procedure call. Procedures, also known as routines, subroutines, methods, or functions (not to be confused with mathematical functions, but similar to those used in functional programming), simply contain a series of computational steps to be carried out. Any given procedure might be called at any point during a program's execution, including by other procedures or itself. Procedural programming is a list or set of instructions telling a computer what to do step by step and how to perform from the first code to the second code.

**Event Driven**: In computer programming, event-driven programming is a programming paradigm in which the flow of the program is determined by events such as user actions (mouse clicks, key presses), sensor outputs, or messages from other programs/threads. Event-driven programming is the dominant paradigm used ingraphical user interfaces and other applications (e.g. Javascript web applications) that are centered around performing certain actions in response to user input.

In an event-driven application, there is generally a main loop that listens for events, and then triggers a callback function when one of those events is detected. Inembedded systems the same may be achieved using hardware interrupts instead of a constantly running main loop. Event-driven programs can be written in any programming language, although the task is easier in languages that provide high-level abstractions, such as closures.

**Object-Orientated**: Object-oriented programming (OOP) is a programming paradigm that represents concepts as "objects" that have data fields (attributes that describe the object) and associated procedures known as methods. Objects, which are usually instances of classes, are used to interact with one another to design applications and computer programs. C++, Objective-C, Smalltalk, Java, C#, Perl, Python, Ruby and PHP are examples of object-oriented programming languages.

# Commendable



Event Driven Programming: This is event driven programming because it makes use of user events and event listeners to make the program.

Object Orientated Programming: This is partially object orientated because it takes advantage of UI objects such as dropdown menus, radio buttons, checkboxes and a button. It accesses data off of the objects classes.

Procedural Programming: This program uses procedural programming because it goes through a set order of steps to get the job done.

# Exceptional

this.orderPizza.addEventListener(MouseEvent.CLICK, processOrder);

function processOrder(e:MouseEvent):void {

// Init Variables

var type:String;// Pizza Type

var qty:int;// Qty of Pizzas

var size:String;// Size of Pizza

var cheese:Boolean;// Extra Cheese

var formattedOrder:String; // Print this as an Order

// STORE Value of Pizza Drop down menu

type = Object(this).pizzaType.value;

// STORE the quantity of pizzas required

qty = Object(root).pizzaQty.value;

// STORE value of pizza size from radio group

// is SMALL radio button selected?

// SET variable = SMALL

if (Object(root).pizzaSizeSmall.selected) {size = "Small";}

// is MEDIUM radio button selected?

// SET variable = MEDUIM

if (Object(root).pizzaSizeMedium.selected) {size = "Medium";}

// is LARGE radio button selected?

// SET variable = LARGE

if (Object(root).pizzaSizeLarge.selected) {size = "Large";}

// STORE the extra cheese option

cheese = Object(root).pizzaExtraCheese.selected;

// OUTPUT a message with the order details

if (cheese) { // Change output depending on cheese

formattedOrder = (qty +" "+ size +" "+ type +" "+ "With Extra Cheese");

} else {

formattedOrder = (qty +" "+ size +" "+ type);

}

trace (formattedOrder);

}

The Code is Imperative because it has a workflow.

Firstly: store all the data inputted by the user.

Secondly: format a string to be easily human readable.

Finally: print the string so the store attendant can read it or process the price of each item etc…