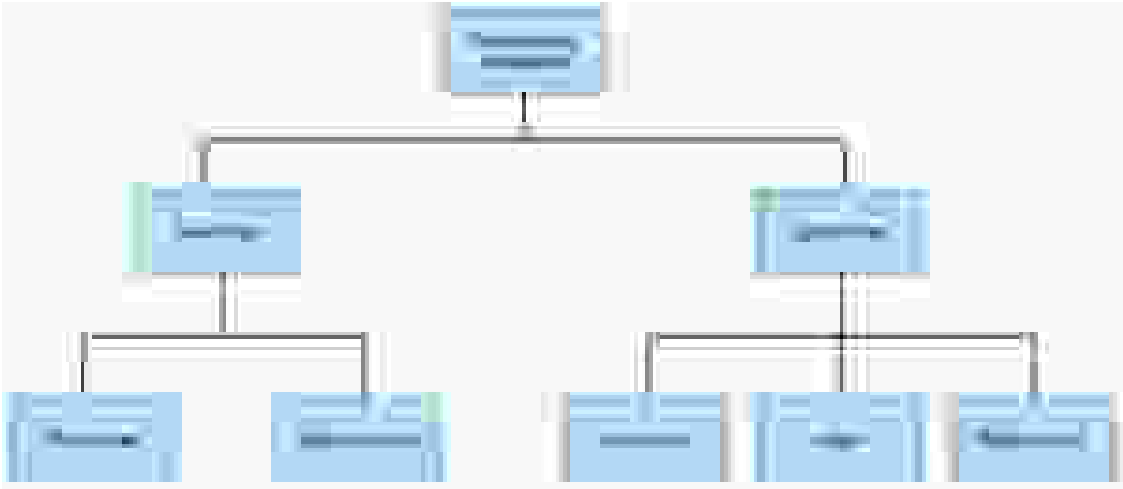
VihangaGunathilake · 16 minutes ago · 8 min read

Full Blog Report(pdf.)

Programming Paradigm...

Programming Paradigm is a ***style*** or ***way*** of programming. It is a way to classify programming languages based on their features and the class of problem they are trying to solve.
ex: Object Oriented Programming, Functional programming
Programming Paradigm can classify two main things.



What is an Imperative language..?

Imperative programming is a paradigm of computer programming in which the program describes a sequence of steps that change the state of the computer.

What are the states of computer.....?

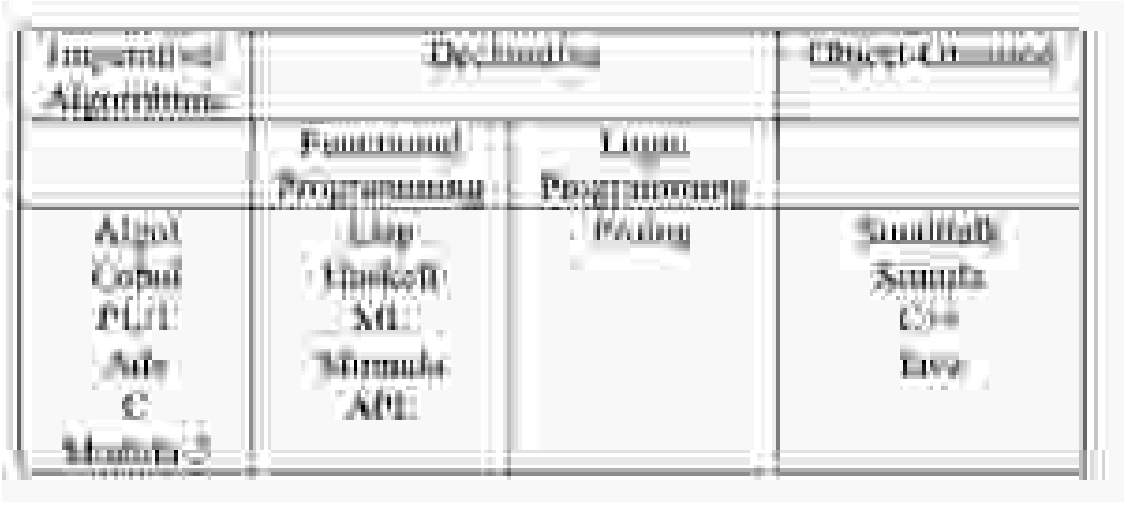
In information technology and computer science, a program is described as stateful if it is designed to remember preceding events or user interactions; the remembered information is called the **state** of the system.

What is a Declarative language

Declarative programming is a programming paradigm … that expresses the logic of a computation without describing its control flow. ***(when you write your code in such a way that it describes what you want to do, and not how you want to do it. It is left up to the compiler to figure out the how.*)**

Imperative Programming
Describe how to solve the problem.
A sequence of commands.
Fast, Specialized program.
Declarative Programming
Describe what the problem is.
A set of statements.
General, readable and correct program.

Examples for Imperative and Declarative



Non-Structured programming - Non-structured programming is the historically earliest programming paradigm capable of creating Turing-complete algorithms.

A series of code.
Flow control with GOTO statements.
Becomes complex as the number of lines increases .

E.g. BASIC, Fortran, COBOL

Structured Programming - Structured programming (sometimes known as *modular programming*) is a subset of procedural programming that enforces a logical structure on the program being written to make it more efficient and easier to understand and modify.

What is the Functional Programming...?

Functional programming is a ***programming paradigm in which we try to bind everything in pure mathematical functions style***. It is a declarative type of programming style. Its main focus is on “what to solve” in contrast to an imperative style where the main focus is “how to solve”. It uses expressions instead of statements. An expression is evaluated to produce a value whereas a statement is executed to assign variables. Execution of a function does not affect the global state of the system. and also functionalprogramming does not have any changers. In here we have to use declarative approach. It's describe what the program must accomplished. And does not describe how to accomplish the task as a sequence of the programming language primitives .So that there is no side effects.

Referential Transparency

The output of a function only depends on the inputs. And similar to constant. So referential transparency means that equals can be replaced by equals.

Functional Programming is **based on Lambda Calculus.**
what is a Lambda Calculus..?

Lambda calculus is framework developed by Alonzo Church to study computations with functions.
It can be called as the smallest programming language of the world. It gives the definition of what is computable.
Anything that can be computed by lambda calculus is computable. It is equivalent to Turing machine in its ability to compute.
It provides a theoretical framework for describing functions and their evaluation. It forms the basis of almost all current functional programming languages.
Lambda Expressions

A lambda expression with no parameters

```
1() -> System.out.println("Hi!Lambda Expressions is great.");
```

A lambda expression with one parameter

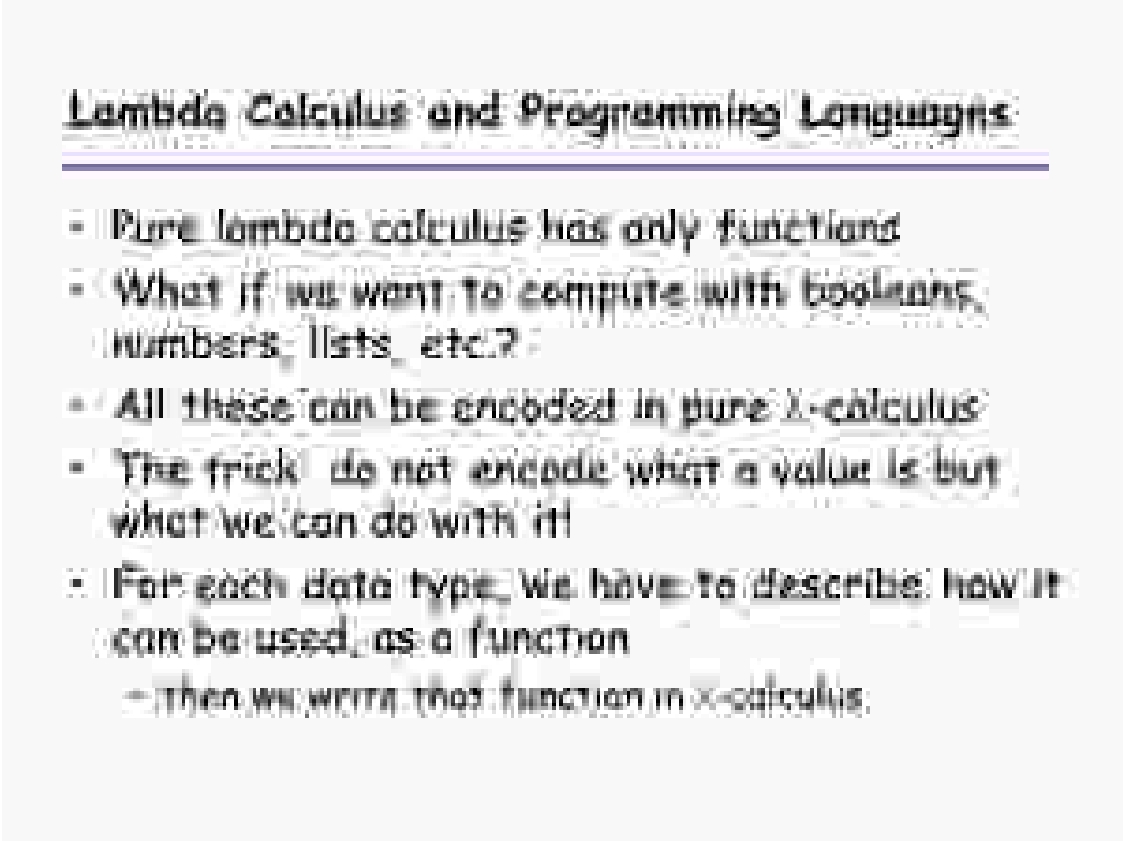
```
1(String s) -> System.out.println("Hi!Lambda Expressions is       great."+s);
```

Another lambda expression with one parameter

```
1s -> System.out.println("Hi!Lambda Expressions is great."+s);
```

A lambda expression with 2 parameters (Note parantesis is required for 2 or more params)

```
1(s1, s2) -> System.out.println("Hi!Lambda Expressions is great." + s1       + s2);
```



What is the Procedural Programming..?

Procedural programming is a programming paradigm that uses a linear or top-down approach. It relies on procedures or subroutines to perform computations.
Procedural programming paradigm helps to structure the code using blocks called **procedures,routines,sub-routines,functions, methods**. A procedure can implement a single algorithm using the control structures(Sequential, Selection, Repetition).
Procedural programming is also known as **imperative programming**.
In procedural programming, a program consists of data and modules/procedures that operate on the data. The two are treated as separate entities. In the object-oriented programming (OOP) paradigm, however, a program is built from objects. An object is an instance of a class, which is an encapsulation of data (called fields) and the procedures (called methods) that manipulate them. In most, but not all, cases, the fields can only be accessed or modified through the methods. An object therefore is like a miniature program or a self-contained component, which makes the OOP approach more modularized and thus easier to maintain and extend.
Another type of programming paradigm that procedural programming can be contrasted with is event-driven programming. In this approach, procedures are called/executed only in response to events, which may include mouse clicks, keyboard press, attaching or removing a device, arrival of data from an external source, etc. As these events are unpredictable, the procedures that handle them cannot be executed linearly as is the case with procedural programming

Imperative approach

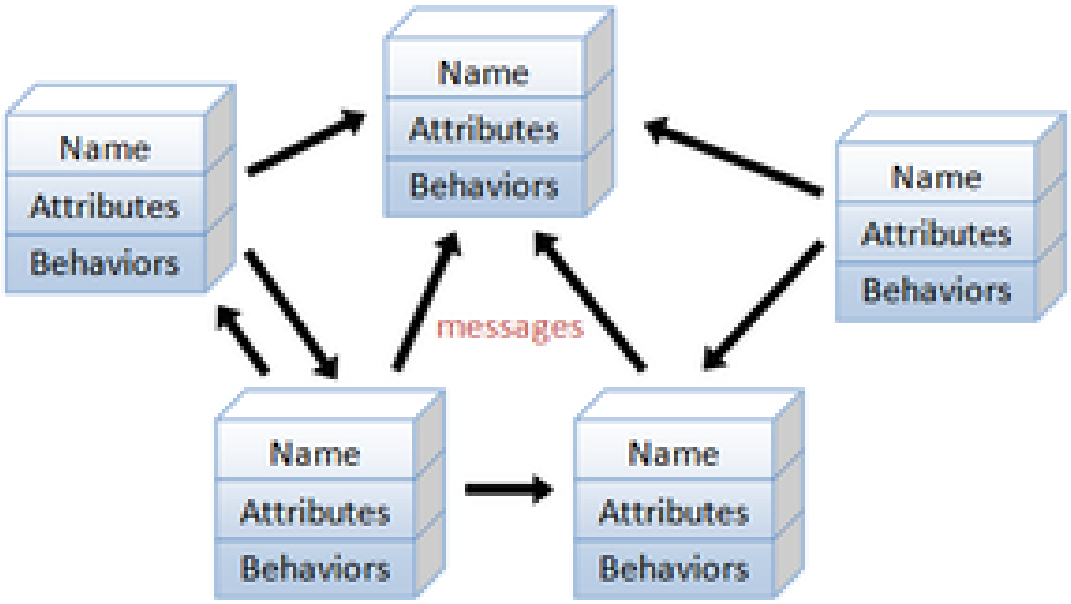
Using a sequence of code to explain the flow of execution. It has side effects. Because execution of a code block, changes the state of the system.

Event Driven Programming

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 or threads.
Use event-listeners (event-handlers) to act when the events are triggered/fired. An internal **event loop** (main loop) is used to identify the events and then call the necessary handlers.
event loop - In computer science, the event loop, message dispatcher, message loop, message pump, or run loop is a programming construct that waits for and dispatches events or messages in a program.

Key features of Object Oriented Programming paradigm

Object-oriented programming is a programming language model organized around objects rather than "actions" and data rather than logic.



An object-oriented program consists of many well-encapsulated objects and interacting with each other by sending messages

Instantiation and Constructors

There are two structuring concepts called **classes** and **objects** in OOP. Class is containing attributes and methods. Objects are objects are the things you think about first in designing a program and they are also the units of code that are eventually derived from the process. When we create new objects using class as a templates called **Instantiation.**
A Constructor is a kind of Method. It has the extra feature that it creates (instantiates) new Objects.

A Method is a kind of Procedure. It has the extra rule that it must be part of a Class and/or Object.
A Procedure in OOP is the same as a Procedure in non-OOP languages.
Example for Instantiation

```
class Employee {

 double payRate;
 double hoursWorked;

   public Employee(double payRate, double hoursWorked) {

       this.hoursWorked = hoursWorked;
       this.payRate = payRate;
 }
}
```

Polymorphism

Polymorphism means to **process objects differently based on their data type**. In other words it means, one method with multiple implementation, for a certain class of action. And which implementation to be used is decided at runtime depending upon the situation (i.e., data type of the object) This can be implemented by designing a generic interface, which provides generic methods for a certain class of action and there can be multiple classes, which provides the implementation of these generic methods.

```
public class Dog extends Animal {

    public void makeSound() {
        System.out.println("Woof!");
    }

    public void makeSound(boolean injured) {
        if (injured) {
            System.out.println("Whimper");
        }
    }
```

```
}

public class Cat extends Animal {

}
```

Polymorphism could be static and dynamic both. Method Overloading is static polymorphism while, Method Overriding is dynamic polymorphism.

Overloading in simple words means more than one method having the same method name that behaves differently based on the arguments passed while calling the method. This called static because, which method to be invoked is decided at the time of compilation.
Overriding means a derived class is implementing a method of its super class. The call to overriden method is resolved at runtime, thus called runtime polymorphism.

Ad-hoc Polymorphism

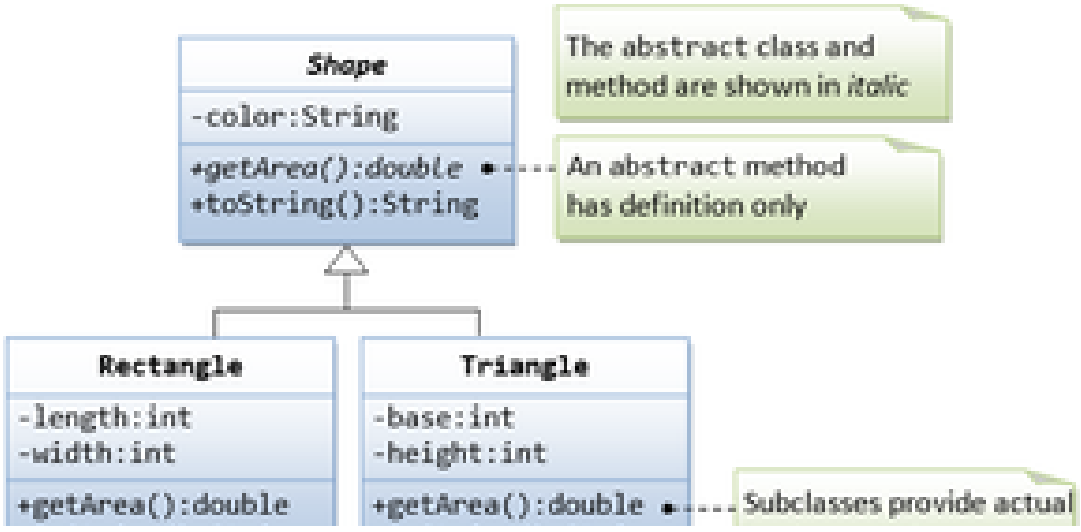
The simplest form of Polymorphism is Ad-hoc Polymorphism when the programmer writes multiple different versions of the Procedure: e.g. one that accepts Objects of type A, and one that accepts Objects of type B. Both versions have the same name, and the OOP language knows to treat them as if they are the same, but to intelligently use one or the other depending on how the Procedure is invoked at runtime.

subtype polymorphism

The more powerful form, used heavily in OOP, is Subtype. With subtype polymorphism, the programmer links different datatypes to each other, promising the computer that - in some way - those datatypes can be used interchangeably. The mechanism for this is Inheritance.

Inheritance

Inheritance is the mechanism by which an object acquires the some/all properties of another object. It supports the concept of hierarchical classification.
Example for inheritance

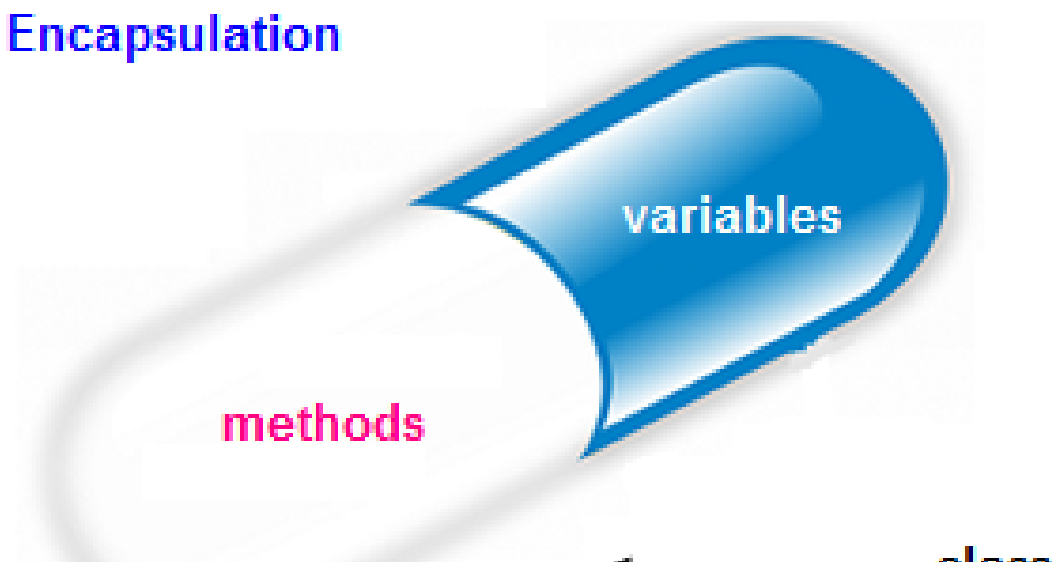



```
class Teacher {
   String designation = "Teacher";
   String collegeName = "Beginnersbook";
   void does(){
        System.out.println("Teaching");
   }
}

public class PhysicsTeacher extends Teacher{
   String mainSubject = "Physics";
   public static void main(String args[]){
        PhysicsTeacher obj = new PhysicsTeacher();
        System.out.println(obj.collegeName);
        System.out.println(obj.designation);
        System.out.println(obj.mainSubject);
        obj.does();
   }
}
```

Encapsulation

Binding the data with the code that manipulates it. It keeps the data and the code safe from external interference.
Similarly, same concept of encapsulation can be applied to code. Encapsulated code should have following characteristics:

Everyone knows how to access it.
Can be easily used regardless of implementation details.
There shouldn't any side effects of the code, to the rest of the application.
The idea of encapsulation is to keep classes separated and prevent them from having tightly coupled with each other.

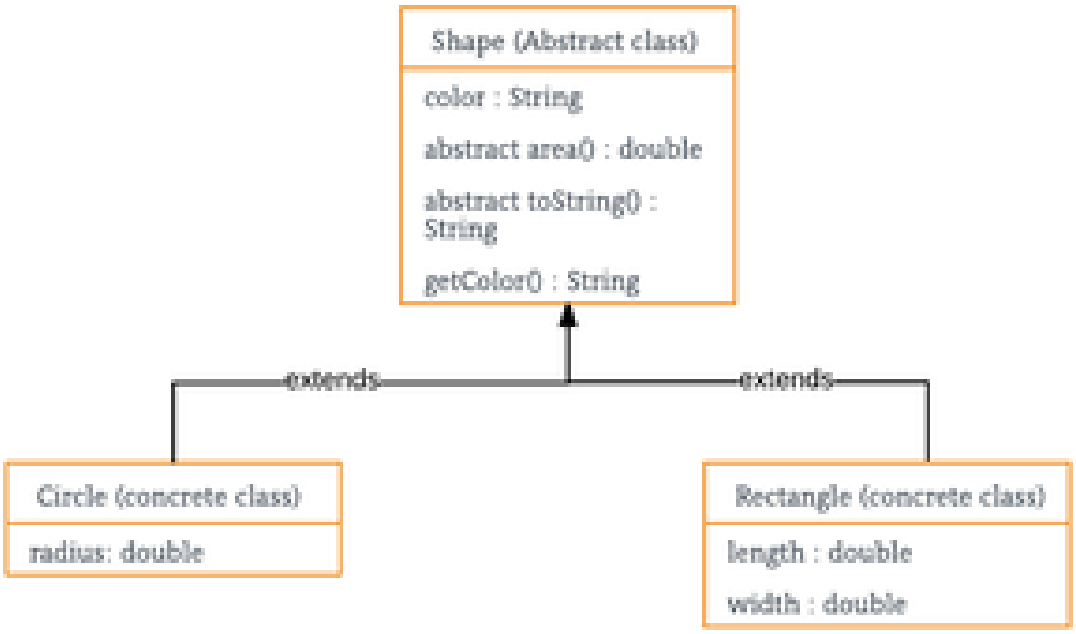



```
class EncapsulationDemo{
    private int ssn;
    private String empName;
    private int empAge;

    //Getter and Setter methods
    public int getEmpSSN(){
        return ssn;
    }

    public String getEmpName(){
        return empName;
    }

    public int getEmpAge(){
        return empAge;
    }

    public void setEmpAge(int newValue){
        empAge = newValue;
    }

    public void setEmpName(String newValue){
        empName = newValue;
    }

    public void setEmpSSN(int newValue){
        ssn = newValue;
    }
}
public class EncapsTest{
    public static void main(String args[]){
         EncapsulationDemo obj = new EncapsulationDemo();
         obj.setEmpName("Mario");
         obj.setEmpAge(32);
         obj.setEmpSSN(112233);
         System.out.println("Employee Name: " + obj.getEmpName());
         System.out.println("Employee SSN: " + obj.getEmpSSN());
         System.out.println("Employee Age: " + obj.getEmpAge());
    }
}
```

Abstraction

Abstraction is a process where you show only “relevant” data and “hide” unnecessary details of an object from the user. For example, when you login to your Google account online, you enter your username and password and press login, what happens when you press login

enter your username and password and press login, what happens when you press login, how the input data sent to Google server, how it gets verified is all abstracted away from the you.



Compiled Languages

A compiled language is a programming language whose implementations are typically compilers, and not interpreters.(Normally if we **compile** the code to get an output, it's called compiled language.)This convert the language which you use to code a programming ,to the Machine language.ex: JAVA, C++

Scripted Languages

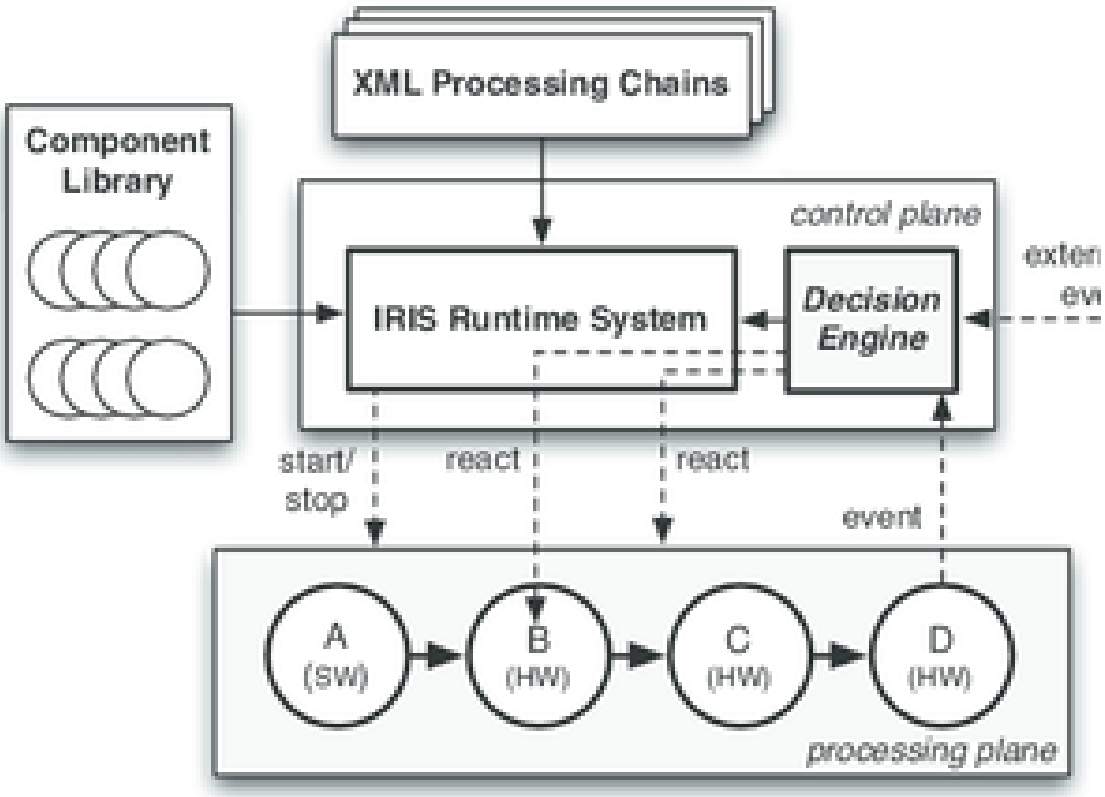
A scripting or script language is a programming language that supports scripts programs written for a special run-time environment that automates the execution of tasks that could alternatively be executed one-by-one by a human operator. In here the source code can't compile. only can execute.
ex: JAVASCRIPT, PHP

Markup Languages

A markup language is a system for annotating a document in a way that is syntactically distinguishable from the text. The idea and terminology evolved from the "marking up" of paper manuscripts, i.e., the revision instructions by editors, traditionally written with a blue pencil on authors' manuscripts. Can not compile and execute program. Only render the program and get output.
ex:HTML . XML

Software runtime Architecture

A **software architectural** model describes the design of the **software** system in terms of components and connectors.



What is the virtual machine and Containers

virtual machine (**VM**) is an emulation of a computer system. Virtual machines are based on computer architectures and provide functionality of a physical computer. Their implementations may involve specialized hardware, software, or a combination.
There are two type of virtual machines :-
System virtual machines
Process virtual machines

Subscribe

Email Address

Sign Up