20CYS312 - Principles of Programming Languages
Exploring Programming Paradigms

Assignment-01

Presented by Harshitha Ranjith
CB.EN.U4CYS21022
TIFAC-CORE in Cyber Security
Amrita Vishwa Vidyapeetham, Coimbatore Campus

Feb 2024

Outline

Programming Paradigm

A paradigm is a way of approaching a problem or carrying out a task. A programming paradigm is a way of approaching problems with programming languages, or with the tools and techniques at our disposal while adhering to a certain approach. There are many well-known programming languages, but when they are used, they all need to adhere to a technique or strategy, and this approach is known as paradigms. In addition to several programming languages, there are numerous paradigms to meet every need. There are 2 types of programming paradigms:

1. Imperative Programming Paradigm
   1. Procedural Programming paradigm
   2. Object Oriented Programming Paradigm
   3. Parallel Processing Approach
2. Declarative Programming Paradigm
   1. Logic Programming Paradigm
   2. Functional Programming
   3. Database Processing Approach

Programming Paradigm

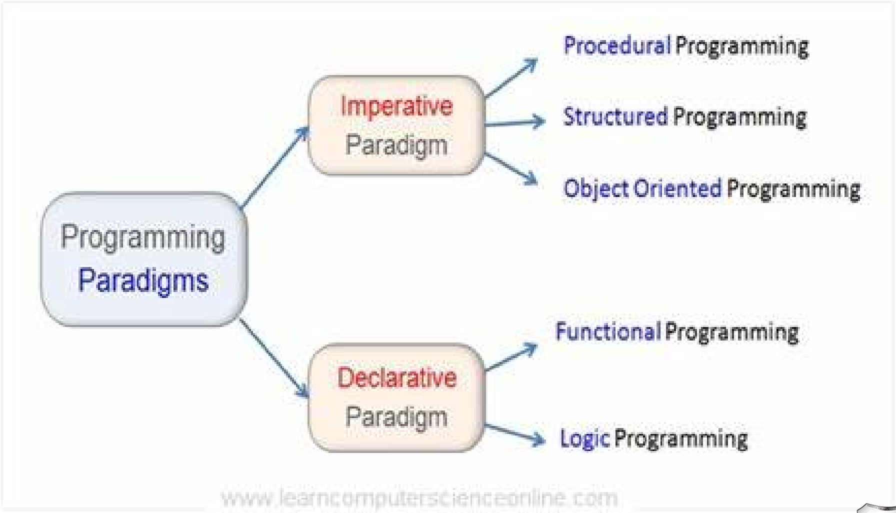

Figure: Programming paradigms

Imperative Paradigm

Definition

The imperative programming paradigm uses a sequence of statements to modify a program's state through the use of variables. The goal of the imperative paradigm is to specify how a program should execute through explicit instructions.

If this is confusing, for now consider this:

- a cookbook instructs step-by-step how to cook a recipe.
- a restaurant menu shows what you can order

and you don't need to worry how it's made.

Imperative Paradigm

Main Characteristics of Imperative Programming

- Sequence of Statements.
- Order of execution of Statements is very important.
- They contain state.
- They use both Immutable and Mutable Data.
- They can change state.
- They may have Side-effects.
- Stateful Programming Model.
- They directly change the state of Program.
- They represent state with Data Fields.

Imperative - Objective C

It is an object-oriented, general purpose language and was created with the vision of providing small talk-style messaging to the C programming language.

- This language allows the users to define a protocol by declaring the classes and the data members can be made public, private and protected.
- This language was used at Apple for iOS and OS X operating systems. Swift language was developed at Apple in 2014 to replace this language.
- The main difference in C and Objective C is that C is a procedure programming language which doesn't support the concepts of objects and classes and Objective C is Object-oriented language which contains the concept of both procedural and object-oriented programming languages.

Imperative - Objective C

Features of Objective c:

1. Objective C supports exception handling, which can be implemented using catch and try blocks.
2. Using the concept of encapsulation, security can be achieved in Objective C.
3. It supports inline functions.
4. Objective C supports function and operator overloading.
5. It is known as an object-driven language.
6. Encapsulation, data hiding, inheritance, polymorphism, and abstraction are the key features.
7. Objective C supports templates.
8. It is well-suited for networking, gaming, etc.

Imperative - Objective C

Objective-C's Imperative Design :

1. Explicit Commands: Statements in Objective-C directly instruct the computer to perform actions. Every line of code, like function calls, assignment statements, and loops, tells the program what to do next.
2. Sequencing and Control Flow: Program execution follows a specific order dictated by the sequence of statements. Control flow structures like if statements, for loops, and while loops allow the program to make decisions and execute different blocks of code based on conditions.
3. Procedural Decomposition: Complex tasks are broken down into smaller, clearly defined functions or methods. This allows for code reuse, readability, and modularity.
4. State Management: Variables and objects hold the state of the program, representing data and memory locations. Imperative programs explicitly manipulate these states through assignments and method calls.
5. Side Effects: Many operations change the state of the program outside the immediate context of the statement. This requires careful consideration of program state and the potential for unexpected behavior.

Imperative - Objective C

Figure: Objective-c

Even Driven Paradigm

Definition

A programming paradigm that structures and organizes the flow of code around responding to events originating from external sources such as user input or system changes.

It allows the creation of dynamic applications where the flow of control is determined by the sequence of events, rather than a predetermined order of execution.
There are several key components in Event Driven Programming that work together to handle and process events in an efficient way.
Understanding these components is essential for creating successful Event Driven applications.

Even Driven Paradigm

Event Handlers: Event handlers are the backbone of Event Driven Programming. These are functions or methods that are designed to be triggered when a specific event occurs.
For instance, when a user clicks on a button on a graphical user interface, an event handler associated with that button responds by executing the designated code.
Event handlers can be further categorized as:
i)Synchronous event handlers: Execute the code immediately when an event occurs.
ii)Asynchronous event handlers: Allow other tasks to continue executing while the code for handling the event is being processed. Example: A simple event handler in JavaScript for a button click event:

```
function displayMessage()
alert("Hello, World!");
```

Advantages and Disadvantages of Event-Driven

Advantages:

1. Flexibility: It is easier to alter sections of code as and when required.
2. Suitability for graphical interfaces: It allows the user to select tools (like radio buttons etc.) directly from the toolbar.
3. Allows for more interactive programs: It enables more interactive programming. Event-driven programming is used in almost all recent GUI apps.
4. Allows sensors and other hardware: It makes it simple for sensors and other hardware to communicate with software.
5. A good way to model systems: Useful method for modeling systems that must be asynchronous and reactive.

Disadvantages:

1. Complex: Simple programs become unnecessarily complex.
2. Difficult to find error: Debugging an event-driven program is difficult.
3. Less logical and obvious: The flow of the program is usually less logical and more obvious.
   Once you become used to Delphi, it gets difficult for a programmer to adapt to another programming language.

Event Driven - Node.js

Node JS is an open-source and cross-platform runtime environment built on Chrome's V8 JavaScript en- gine for executing JavaScript code outside of a browser.

It provides an event-driven, non-blocking (asyn- chronous) I/O and cross-platform runtime environment for building highly scalable server-side applications using JavaScript.

Features of Node JS :

1. Easy Scalability: Node.js is built upon Chrome V8's engine powered by Google, enabling lightning-fast JavaScript execution in a server-side runtime environment.
2. Real-time web apps: The modern web emphasizes real-time interaction, and Node.js excels in supporting features like chat, gaming, social media updates, and more.
3. Fast Suite: Node.js is highly scalable and lightweight, making it a preferred choice for microservice architectures that involve breaking down applications into isolated services.
4. Easy to learn and code: JavaScript is essential for front-end development, and Node.js allows developers to focus on a single language for both front-end and back-end development.
5. Data Streaming: Node.js efficiently handles I/O processes, facilitating tasks like transcoding media files during uploads faster than traditional methods.
6. Corporate Support: Node.js has an independent community and foundation dedicated to accelerating its development and ensuring broad adoption.

Features of Node.js

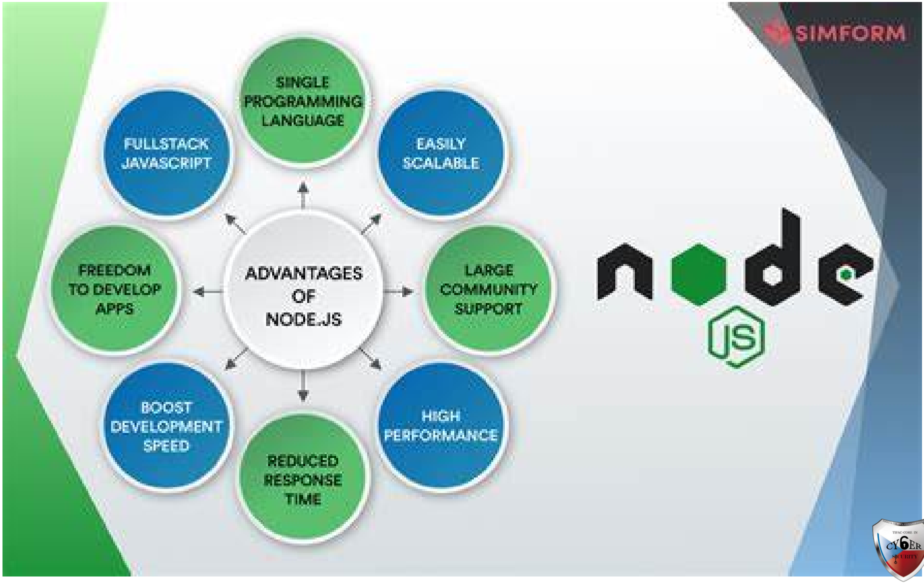

Figure: Node.js

Event-Driven Programming in Node.js:

1. Node.js leverages events for enhanced speed. When a Node.js server starts, it initializes variables and functions, listening for events.
2. Event-Driven Programming simplifies programs. Components include:
   - Callback function (event handler) triggered when an event occurs.
   - Event loop detecting triggers and calling corresponding event handlers.
   - Functions acting as 'Observers,' triggered by events.
3. EventEmitter: Core of Node's event-driven architecture, allowing objects to communicate.
4. In-built modules inherit from EventEmitter, enabling access to a range of events.
5. Emitter objects send named events, triggering pre-registered listeners.

Comparison and Discussions

Both offers contrasting ways to navigate the choreography of your program's execution. Let's compare and contrast these approaches:

Similarities:

1. Both paradigms guide programs to achieve desired outcomes.
2. Abstraction is leveraged in both, allowing focus on the bigger picture.
3. Many languages offer versatility with elements from both paradigms.

Differences:

1. Control Flow:
   - Imperative: Orchestrates flow through explicit commands.
   - Event-Driven: Reacts to events, improvising based on triggers.
2. Focus:
   - Imperative: Focuses on "how" to achieve goals.
   - Event-Driven: Focuses on "what" needs to happen in response to events.
3. State Management:
   - Imperative: Actively manages state through variables and assignments.
   - Event-Driven: State evolves dynamically with minimal manual manipulation.
4. Examples:
   - Imperative: Writing a loop for specific file checks.
   - Event-Driven: Setting up a listener for mouse clicks on a button.

References

- https://www.w3schools.com/nodejs/nodejs_intro.asp
- https://www.geeksforgeeks.org/nodejs
- https://www.geeksforgeeks.org/difference-between-c-and-objective-c
- https://www.geeksforgeeks.org/explain-event-driven-programming-in-node-js
- https://www.tutorialspoint.com/objective_c/index.htm
- https://www.devmaking.com/learn/programming-paradigms/imperative-paradigm
- https://www.geeksforgeeks.org/introduction-of-programming-paradigms
- https://www.studysmarter.co.uk/explanations/computer-science/computer-programming/event-driven-programming