20CYS312 - Principles of Programming Languages
Exploring Programming Paradigms

Assignment-01

Presented by «Your Name»

«Your Roll Number»

TIFAC-CORE in Cyber Security

Amrita Vishwa Vidyapeetham, Coimbatore Campus

Feb 2024

AMRITA
VISHWA VIDYAPEETHAM

Outline

1. Reactive Programming Paradigm
2. Reactive - Angular
3. Logic Programming Paradigm
4. Logic - Datalog
5. Comparison and Discussions
6. Bibliography

Reactive Programming Paradigm

- **Introduction to Reactive Programming:**
  - Reactive programming is a declarative programming paradigm that focuses on managing and responding to changes in application state.
  - It provides an alternative approach to traditional imperative programming by emphasizing the propagation of data changes and events.
- **Key Concepts:**
  - **Observables:** Represent asynchronous data streams, allowing components to react to changes over time.
  - **Observers:** Subscribe to observables to handle and respond to emitted values or events.
  - **Operators:** Transform, filter, and manipulate data emitted by observables, providing a powerful toolset for handling asynchronous operations.
  - **Subjects:** Special type of observable that allows both the emission and subscription of values, facilitating communication between different parts of the application.
- **Common Use Cases:**
  - GUI applications responding to user interactions.
  - Network communication, such as handling HTTP requests.
  - Real-time data processing and event-driven architectures.

Reactive - Angular

Angular:

- Angular is a web framework application developed and managed by Google.
- It is built using the TypeScript programming language.
- Developers use TypeScript to write code for Angular applications, which is then compiled into standard JavaScript for web browsers.

Reactive Programming in Angular: Reactive programming in Angular revolves around the use of reactive extensions for JavaScript (RxJS), facilitating the handling of asynchronous operations and event-based programming. Key concepts include:

- **Observables:** Observables are extensively used in Angular for handling asynchronous tasks such as user interactions and HTTP requests.
- **Operators:** Operators are functions applied to observables to transform, filter, and combine data streams, providing a clean and declarative approach to handling complex asynchronous scenarios.
- **Observers:** Components, services, and other parts of an Angular application act as observers, responding to changes in the data stream.

Reactive - Angular

- **Subjects:** Subjects are both observers and observables. They can emit values and be subscribed to, making them suitable for scenarios where data needs to be multicast to multiple observers.
- **Reactive Forms:** Angular's reactive forms module leverages reactive programming concepts to manage and react to changes in form controls. It provides a declarative and reactive way to handle form input.
- **HTTP Requests:** Angular's HTTP client returns observables when making asynchronous HTTP requests. This enables developers to handle responses in a reactive way, applying operators for data transformation.

Logic Programming Paradigm

- A logic programming paradigm is a set of principles and techniques that guide the design and implementation of logic programs.
- A logic program consists of a collection of facts and rules that describe the relationships and properties of entities, and a query language that allows asking questions and obtaining answers from the program.
- A logic programming paradigm defines the syntax and semantics of the facts, rules, and queries, as well as the inference mechanism that derives new facts and rules from the existing ones.

Logic - Datalog

- Datalog is a declarative query language, derived from Prolog.
- Its logic is based on first order logic and more specifically logical clauses expressed as Horn clauses.
- Its origins date back to the beginning of logic programming, and is often being used to describe systems, or to build domain models.
- It can be used in many different ways, including to:
  - query data from one system to another
  - model data in terms of relations between entities
  - create new data models with minimal code
- Example of a datalog query

```
[:find <entity>
 :where
 [<entity> <attribute> <value>]]
```

Comparison

- **P**urpose:
  - **Reactive Programming:** Focuses on building systems that react to changes in data or events. It is particularly useful for handling asynchronous and event-driven scenarios.
  - **Logic Programming:** Primarily concerned with expressing and solving problems in terms of relations and rules. It is often used for declarative problem-solving and is associated with symbolic reasoning.
- **D**ata Flow:
  - **Reactive Programming:** Emphasizes the flow of data and the propagation of changes. Reactive systems respond to events and changes in a continuous manner.
  - **Logic Programming:** Concerned with relationships and rules. It emphasizes the logical relationships between entities and the use of logical inference to derive new information.
- **E**xecution Model:
  - **Reactive Programming:** Typically employs an event-driven or data-flow-driven execution model, reacting to changes as they occur.
  - **Logic Programming:** Often relies on a backtracking search strategy, exploring possible solutions and considering alternative paths until a satisfactory solution is found.

Comparison

- **E**rror Handling:
  - **Reactive Programming:** Typically involves handling errors through the observable stream or event pipeline.
  - **Logic Programming:** Errors are often handled through backtracking or exception mechanisms.
- **U**se Cases:
  - **Reactive Programming:** Reactive programming is well-suited for developing responsive user interfaces, applications that require real-time updates, such as stock market dashboards and live sports scores, systems that rely heavily on events, like IoT applications, sensor data processing, or gaming engines
  - **Logic Programming:** Logic programming, especially with languages like Prolog, is often used in building expert systems and AI applications, applied in NLP tasks, such as language parsing, understanding, and question-answering systems,are employed in querying and manipulating databases using a declarative approach[Datalog].

References

- https://www.baeldung.com/cs/reactive-programming: :text=Reactive
- https://medium.com/front-end-weekly/why-should-we-use-reactive-programming-in-angular-2e2913297054
- https://angular.io/guide/rx-library
- https://www.linkedin.com/advice/0/what-main-characteristics-examples-logic-programming
- https://en.wikipedia.org/wiki/Logicprogramming
- https://www.virtusa.com/digital-themes/logic-programming