ONE PAGE SUMMARY

IT & Software › Software Architecture › Event Driven Design

Introduction to Event Driven Design

Unlocking the Power of Event Driven Design in Software Architecture

DEFINITION

> Event Driven Design is a software architecture pattern where system actions are triggered by events. It allows for more responsive and scalable applications by decoupling components.

KEY CONCEPTS

Events

Events are significant changes in state or occurrences that trigger actions in a system.

Event Producers

These are components that generate events when certain actions occur, like user inputs or data updates.

Event Consumers

Components that listen for and respond to events, executing specific actions when events are received.

Message Brokers

Intermediaries that facilitate communication between event producers and consumers, ensuring messages are delivered.

Asynchronous Processing

A method where event handling occurs independently, allowing the system to remain responsive while processing events.

EXAMPLES

→ User clicks a button, triggering an event to save data.

→ An online store updates inventory levels when a purchase event occurs.

→ A weather application fetches new data when a weather update event is received.

MEMORY TIPS

★ Think of events as 'triggers' that set things in motion.

★ Remember 'producer-consumer' like a factory: producers create, consumers use.

★ Associate message brokers with mail carriers delivering messages between parties.

COMMON MISTAKES

✗ Confusing events with actions; events are triggers, not the actions themselves.

✗ Neglecting to handle errors in event processing, leading to system failures.

✗ Overcomplicating event flows instead of keeping them simple and clear.

QUICK RECAP

Event Driven Design focuses on using events to trigger system actions, promoting decoupled and scalable architecture. Key components include events, producers, consumers, and message brokers. Understanding these elements is crucial for effective implementation.