1. CommunicationHandler Interface

Purpose: This interface defines the communication methods that any communication handler (like UDP) should implement.

* sendMessage(String message): Sends a message to the specified destination.
* listenForMessages(): Listens for incoming messages.

1. JSONProcessor Class:

Purpose: This class handles JSON message encoding and decoding.

* encodeMessage(...): Creates a JSON string containing client type, message type, client ID, and timestamp. This will be used to format messages sent to the MCP.
* decodeMessage(String message): Parses a JSON string and formats it for pretty printing. This is useful for logging or debugging received messages.
* getTimestamp(): Generates a timestamp in a specified format to include in the JSON message.

1. SimpleCCP Class:

Purpose: This class represents the Carriage Control Program (CCP) that manages communication with the Master Control Program (MCP).

* Attributes:
  + bladeRunnerId: Unique ID for the Blade Runner.
  + stateManager: Manages the state of the CCP.
  + commHandler: Handles communication (e.g., sending and receiving messages).
* Constructor: Initializes the bladeRunnerId, creates a new StateManager, and sets up the communication handler (using UDP).
  + Sets the initial state of the CCP to STARTED.
* connect():
  + Checks if the current state is STARTED.
  + Sends a connection initialization message (CCIN) to the MCP.
  + Updates the state to CONNECTED.
* onMessageReceived(String message):
  + A callback method that gets triggered when a message is received.
  + It prints the received message in pretty JSON format.
* main(): Entry point for the program. It creates an instance of SimpleCCP and calls connect() to establish communication with the MCP.

1. StateManager Class

Purpose: This class manages the state of the CCP.

* CCPState: An enum representing possible states (STARTED, CONNECTED).
* currentState: Keeps track of the current state.
* Constructor: Initializes the current state to STARTED.
* updateState(CCPState newState): Updates the current state and prints a message indicating the change.
* getCurrentState(): Returns the current state.

1. UDPcommunicationHandler.java

Purpose: Implements the CommunicationHandler interface using UDP for sending and receiving messages.

Attributes:

* socket: The UDP socket used for communication.
* mcpAddress and mcpPort: The address and port of the MCP for sending messages.
* messageListener: A listener that processes incoming messages.

Constructor: Initializes the socket and starts a thread to listen for incoming messages.

* sendMessage(String message): Converts the message to bytes and sends it to the MCP.
* listenForMessages(): Continuously listens for incoming messages in a loop. When a message is received, it notifies the messageListener via the callback.