# **Documentazione codice Client: Interfaccia**

## **Descrizione generale**

Classe Java che implementa un'interfaccia grafica Swing per controllare un robot EV3 via rete. Permette di inviare comandi di movimento al robot e visualizzare informazioni di stato ricevute dal server (velocità e posizione).

## **Funzionalità principali**

* Connessione TCP al server del robot su IP 10.0.1.1 e porta 12345.
* Invio comandi di movimento (avanti, indietro, sinistra, destra, stop) e gestione di più marce con velocità diverse.
* Ricezione e visualizzazione in tempo reale dei messaggi di stato dal server (velocità e posizione).
* Interfaccia utente con pulsanti e visualizzazione log.
* Controllo via tastiera (W, A, S, D per direzioni, tasti freccia per cambiare marcia, ESC per uscire).

## **Componenti principali**

### **Variabili e costanti**

* SERVER\_ADDRESS e PORT: indirizzo e porta del server robot.
* PrintWriter out e BufferedReader in: stream di output/input per comunicazione socket.
* JTextArea logArea: area per mostrare log e messaggi.
* JLabel marciaLabel, velocitaLabel, posizioneLabel: etichette per informazioni di marcia, velocità e posizione.
* Variabili di stato per marcia (marcia), velocità, e tasti premuti (avantiPremuto, ecc.).

### **Costruttore Interfaccia()**

* Apre la connessione TCP con il server robot.
* Avvia un thread per ricevere messaggi dal server e aggiorna GUI.
* Costruisce l'interfaccia grafica (pulsanti, pannelli, log).
* Imposta listener per pulsanti e tastiera.

### **Metodi principali**

* aggiungiBottone(JPanel panel, String testo): aggiunge un pulsante con azione al pannello.
* log(String message): stampa log su console e area testuale GUI.
* inviaComando(String comando): invia un comando testuale al server tramite socket; chiude il programma se il comando è "esci".
* aggiornaComando(): invia il comando corrente basato sui tasti premuti.
* aggiornaPosizione(String line): aggiorna l'etichetta posizione parsando il messaggio ricevuto.
* actionPerformed(ActionEvent e): gestisce i click sui pulsanti, inviando comandi.
* keyPressed(KeyEvent e): gestisce la pressione dei tasti per aggiornare lo stato di movimento o altre azioni.
* keyReleased(KeyEvent e): gestisce il rilascio dei tasti per aggiornare lo stato.

## **Uso**

* Avviare il client, si connette automaticamente al server.
* Usare i pulsanti o i tasti W/A/S/D per muovere il robot.
* Usare i tasti freccia su/giù per cambiare marcia/velocità.
* Premere SPAZIO per fermare.
* Premere ESC per uscire (chiude l’applicazione).

# **Documentazione codice Server: RiceviComandi**

## **Descrizione generale**

Programma Java da eseguire su un robot EV3 che funge da server TCP, riceve comandi da un client e li traduce in movimenti controllando i motori. Invia inoltre dati di stato (velocità, posizione) periodicamente al client. Gestisce anche un sensore ultrasonico per rilevare ostacoli.

## **Funzionalità principali**

* Apre un server socket sulla porta 12345 e accetta connessioni dai client.
* Controlla due motori regolati (leftMotor, rightMotor) per muovere il robot.
* Riceve comandi testuali dal client per muovere il robot (avanti, indietro, sinistra, destra, stop, esci).
* Aggiorna la posizione stimata del robot e invia periodicamente velocità e posizione al client.
* Monitora un sensore ultrasonico; emette un beep in caso di ostacoli vicini (< 5 cm).
* Gestisce la terminazione dell’applicazione su comando "esci".

## **Componenti principali**

### **Variabili e costanti**

* port: porta TCP per il server.
* leftMotor e rightMotor: motori controllati (Motor.B e Motor.C).
* running: flag per controllo del ciclo principale.
* ultrasonicSensor: sensore ultrasonico collegato a porta S3.
* position: array per la posizione stimata (x, y).
* lastUpdate: timestamp ultimo aggiornamento per calcolo posizione.
* currentSpeed: velocità motori sinistro e destro correnti.

### **Flusso principale**

1. Creazione di ServerSocket e attesa di connessioni.
2. Avvio di un thread per leggere continuamente il sensore ultrasonico e segnalare ostacoli.
3. Per ogni client connesso:  
   * Avvio di un thread per inviare periodicamente messaggi di velocità e posizione.
   * Ciclo di lettura comandi dal client:  
     + avanti:, indietro:, sinistra:, destra:: controlla motori con velocità limitata.
     + stop: ferma motori.
     + esci: chiude connessione e ferma server.
   * Chiude la connessione client al termine.

## **Uso**

* Eseguire il programma sul robot EV3 con motori collegati a porte B e C e sensore ultrasonico a porta S3.
* Il server aspetta la connessione da un client sulla porta 12345.
* Riceve comandi per muovere il robot e invia dati di stato.
* Il comando "esci" chiude la connessione e termina il server.