Ce este un OpMode 🧠

- Un OpMode (Operation Mode) este un program care controlează robotul FTC.
- Toate programele pe care le vezi în aplicația Driver Station (ex. „TeleOp - SoloDrive”, „Auto - Blue Left”) sunt OpMode-uri.

- Fiecare OpMode spune cum trebuie să se comporte robotul:
  - în timpul meciului (TeleOp)
  - sau înaintea acestuia, într-o secvență automată (Autonom).

Tipuri principale de OpMode-uri⚙️

TeleOp (Teleoperated Mode)

- Robotul este controlat manual de driveri, folosind gamepadurile.
- Se folosește de obicei pentru controlul mecanismelor (braț, intake, claw etc.).
- Rămâne activ tot timpul cât meciul este în desfășurare (2 minute).

Autonomous (Auto)

- Robotul se mișcă singur, fără control de la driver.
- Codul rulează secvențial: merge, ridică, lasă, se întoarce, etc.
- Se folosește la începutul meciului (primele 30 secunde).

🧩 Clasele de bază pentru OpMode-uri

- **În SDK-ul FTC există două tipuri principale de clase care definesc comportamentul unui OpMode:**

- OpMode (Ne-liniar)
  - Codul nu se execută de sus în jos, ci în cicluri de ~50Hz.
  - Este potrivit pentru TeleOp, pentru că permite actualizarea constantă a inputurilor.
- LinearOpMode (Liniar)
  - Codul se execută pas cu pas, într-o singură funcție
  - Este folosit mai ales în Autonom, unde robotul face acțiuni în ordine.

OpMode (Ne-liniar)

```java
public class ConcreteTeleOp extends OpMode {

    // Declaram motoarele
    3 usages
    private DcMotor leftMotor;
    3 usages
    private DcMotor rightMotor;

    @Override
    public void init() {
        // Conectam motoarele cu numele din configuration
        leftMotor = hardwareMap.get(DcMotor.class, "left_motor");
        rightMotor = hardwareMap.get(DcMotor.class, "right_motor");

        telemetry.addData("Status", "Init complete");
        telemetry.update();
    }

    @Override
    public void loop() {
        // Citim joystick-ul pentru miscare
        double leftPower = -gamepad1.left_stick_y;
        double rightPower = -gamepad1.right_stick_y;

        leftMotor.setPower(leftPower);
        rightMotor.setPower(rightPower);

        // Telemetrie pentru debug
        telemetry.addData("Left Power", leftPower);
        telemetry.addData("Right Power", rightPower);
        telemetry.update();
    }

    @Override
    public void stop() {
        // Oprire motoare
        leftMotor.setPower(0);
        rightMotor.setPower(0);
    }
}
```

LinearOpMode (Liniar)

```java
@Autonomous
public class ConcreteAuto extends LinearOpMode {

    3 usages
    private DcMotor leftMotor;
    3 usages
    private DcMotor rightMotor;

    @Override
    public void runOpMode() throws InterruptedException {
        // Conectam motoarele
        leftMotor = hardwareMap.get(DcMotor.class, "left_motor");
        rightMotor = hardwareMap.get(DcMotor.class, "right_motor");

        telemetry.addData("Status", "Init complete");
        telemetry.update();

        // Asteptam startul
        waitForStart();

        // Secventa automata: mergem inainte 2 secunde
        leftMotor.setPower(0.5);
        rightMotor.setPower(0.5);
        sleep(2000);

        // Oprim motoarele
        leftMotor.setPower(0);
        rightMotor.setPower(0);

        telemetry.addData("Status", "Auto complete");
        telemetry.update();
    }
}
```