

Programmiersprachen für LEGO Mindstorms

EV3 – Vergleich & Entscheidung

Einleitung und Dokumentstruktur

Dieses Dokument bietet einen umfassenden Vergleich verschiedener Programmiersprachen für den LEGO Mindstorms EV3 Roboter. Unser Ziel ist es, die Stärken und Schwächen jeder Sprache zu analysieren, um eine fundierte Entscheidung für unser Projekt treffen zu können.

Aufbau des Dokuments

Das Dokument ist wie folgt strukturiert:

1. **Einleitung und Überblick**
2. **Analyse der Programmiersprachen:**
 - Python für EV3
 - Java für EV3
 - C für EV3
 - Rust für EV3
 - C++ für EV3
 - MicroPython für EV3
3. **Vergleichende Zusammenfassung**
4. **Unsere Entscheidung und Begründung**

Jede Programmiersprache wird nach den gleichen Kriterien analysiert: Beschreibung, Merkmale, Plattformen, Vorteile und Nachteile. Dies ermöglicht einen direkten Vergleich und erleichtert die Entscheidungsfindung.

Python für EV3

Beschreibung

Python ist eine der beliebtesten Programmiersprachen für den LEGO Mindstorms EV3. Sie wird hauptsächlich über die **ev3dev-Plattform** und das speziell für LEGO entwickelte **Pybricks-Framework** unterstützt.

Merkmale

- **Interpretiert:** Kein vorheriges Kompilieren notwendig
- **Einfach zu erlernen:** Gute Lesbarkeit und strukturierte Syntax
- **Große Community:** Zahlreiche Bibliotheken & Tutorials verfügbar

Plattformen

- **ev3dev (Linux-basiert)** – Erfordert eine MicroSD-Karte mit ev3dev
- **Pybricks** – Lässt sich direkt auf den EV3 spielen, keine SD-Karte nötig

Vorteile

- + Einsteigerfreundlich
- + Große Community & zahlreiche Bibliotheken
- + Unterstützt Multithreading (ev3dev)
- + Flexibel & für verschiedene Projekte einsetzbar

Nachteile

- Langsamer als kompilierte Sprachen (Java/C++)
- Höherer Speicherverbrauch auf dem EV3
- Direktes Hardware-Tuning ist schwieriger als in C/C++

Java für EV3 (LeJOS & ev3dev-lang-java)

Beschreibung

Java ist eine **objektorientierte, kompilierte Sprache**, die durch **LeJOS** und **ev3dev-lang-java** unterstützt wird. Sie wird oft für **strukturierte, komplexe Projekte** genutzt.

Merkmale

- **Kompilierte Sprache:** Muss in Bytecode übersetzt werden
- **Läuft auf der JVM:** Java Virtual Machine ist erforderlich
- **Objektorientierung:** Unterstützt Klassen, Vererbung & Bibliotheken

Plattformen

- **LeJOS** – Kompletter Ersatz der LEGO-Firmware
- **ev3dev-lang-java** – Läuft unter ev3dev (Linux)

Vorteile

- + Gute Code-Strukturierung & Wiederverwendbarkeit
- + Starke Standardbibliotheken (z. B. für KI)
- + Gute Dokumentation & breite Entwickler-Unterstützung

Nachteile

- JVM führt zu erhöhtem Speicherverbrauch
- Langsamer als C oder Rust aufgrund der JVM
- Komplexere Einrichtung als Python

C für EV3

Beschreibung

C ist eine **niedrigere, kompilierte Sprache**, die für **Maximalperformance & Echtzeitprogrammierung** genutzt wird. Da sie direkten Zugriff auf die Hardware bietet, wird sie für **leistungsoptimierte Anwendungen** eingesetzt.

Merkmale

- **Kompiliert:** Direkt in Maschinencode übersetzt
- **Sehr effizient:** Hohe Geschwindigkeit, geringer Speicherverbrauch
- **Manuelle Speicherverwaltung:** Erfordert präzise Kontrolle

Plattformen

- **ev3dev (Linux-basiert)** – Direkte Steuerung der EV3-Hardware

Vorteile

- + Höchste Performance & Geschwindigkeit
- + Sehr geringe Latenz
- + Maximale Kontrolle über Speicher & Sensoren

Nachteile

- Erfordert tiefes Verständnis von Speicherverwaltung
- Komplexe Programmierung (kein einfacher Einstieg)
- Weniger Community-Support für EV3

Rust für EV3

Beschreibung

Rust ist eine **moderne Systemprogrammiersprache**, die für **höchste Performance & Sicherheit** entwickelt wurde. Dank **ev3dev** kann Rust auf dem LEGO EV3 genutzt werden.

Merkmale

- **Kompiliert:** Sehr schnell & effizient
- **Sicherheit durch Ownership-Modell:** Keine typischen Speicherfehler
- **Unterstützt Nebenläufigkeit (Concurrency):** Perfekt für parallele Steuerung

Plattformen

- **ev3dev (Linux-basiert)**

Vorteile

- + Sehr hohe Performance (vergleichbar mit C++)
- + Sicherer als C/C++ (kein "Memory Leak")
- + Unterstützt parallele Prozesse für Sensoren/Motoren

Nachteile

- Sehr steile Lernkurve
- Weniger Ressourcen & Dokumentation für EV3

C++ für EV3

Beschreibung

C++ ist eine **erweiterte Version von C**, die **objektorientierte Programmierung** bietet und dennoch sehr effizient ist. Durch ev3dev kann es genutzt werden, um **direkt mit der EV3-Hardware zu kommunizieren**.

Merkmale

- **Kompiliert:** Direkt in Maschinencode übersetzt
- **Bietet Multithreading:** Ideal für parallele Prozesse
- **Starke Kontrolle über Speicher & Hardware**

Plattformen

- **ev3dev (Linux-basiert)**

Vorteile

- + Sehr leistungsfähig & schnell
- + Direkter Zugriff auf Sensoren & Motoren
- + Unterstützt objektorientierte Programmierung

Nachteile

- Komplexere Syntax als Python oder Java
- Erfordert tiefes Hardwareverständnis
- Speicherverwaltung kann herausfordernd sein

MicroPython für EV3

Beschreibung

MicroPython ist eine **optimierte Version von Python**, die speziell für **Mikrocontroller & eingebettete Systeme** entwickelt wurde. Durch **Pybricks** ist es nativ mit LEGO EV3 kompatibel.

Merkmale

- **Leichtgewichtig:** Entwickelt für Embedded-Systeme
- **Ähnlich wie Python:** Einfach zu lernen & lesen
- **Kein vorheriges Kompilieren erforderlich**

Plattformen

- **Pybricks (EV3 ohne ev3dev)**

Vorteile

- + Extrem einfache Einrichtung
- + Effizienter als normales Python
- + Perfekt für Einsteiger & schnelle Prototypen

Nachteile

- Eingeschränkte Leistung für komplexe Projekte
- Weniger Community-Support als Python

Zusammenfassung: Welche Sprache für welchen Zweck?

Sprache	Schwierigkeit	Performance	Hauptanwendung	Plattform
Python	Einfach	Mittel	Einsteigerfreundliche Projekte	ev3dev, Pybricks
Java	Mittel	Mittel	Strukturierte, objektorientierte Projekte	LeJOS, ev3dev-lang-java
C	Schwer	Sehr hoch	Echtzeitsteuerung & Systemprogrammierung	ev3dev
Rust	Sehr schwer	Sehr hoch	Sichere, parallele Steuerung	ev3dev
C++	Schwer	Sehr hoch	High-Performance-Anwendungen	ev3dev
MicroPython	Sehr einfach	Mittel	Einfache Steuerung, schnelle Entwicklung	Pybricks

Unsere Entscheidung

Für unser Projekt haben wir uns für **MicroPython auf PyBricks** entschieden. Diese Wahl basiert auf mehreren Faktoren:

- MicroPython ist einfach, intuitiv und verständlich, auch für Personen ohne umfangreiche Programmiererfahrung
- Die Einrichtung ist unkompliziert und erfordert keine komplexe Konfiguration
- Die Syntax ist leicht zu erlernen und ermöglicht schnelle Entwicklungszyklen
- PyBricks bietet eine gute Balance zwischen Einfachheit und Funktionalität
- Die Performance ist für unsere Anforderungen völlig ausreichend

Andere Sprachen können unter bestimmten Umständen besser geeignet sein. Nichtsdestotrotz ist MicroPython auf PyBricks die ideale Lösung für unser Team und unsere Projektziele.