



# Arbeitsauftrag Informatik

Name:

Vorname:

Klasse:

## PicoIO-Modell – Methoden der Klasse Hardware

Pico-IO ist eine Hardwarekomponente mit Sensoren und Aktoren. Sie besteht aus einem Raspberry Pico, an dem sieben LED (Nummer 16 bis 22) und ein Minilautsprecher angeschlossen sind. Drei Sensoren dienen der Erfassung von Helligkeit, Tastendruck und Magnetfeldstärke. Das Modell wird über USB mit dem PC verbunden und kann in BlueJ über das unsichtbare Objekt `hardware` verwendet werden.



### Klassenkarte der Klasse Hardware

Hardware
<pre>+ledOn(int) +ledOff(int) +ledSwitch(int) +ledSet(int,boolean) +ledDim(int,int) +ledsOff() +getLEDValue(int): int +getLEDState(int): boolean +playBeep(int) +stopBeep() +isPressed(): boolean +wasPressed(): boolean +getHall(): int +isContacted(): boolean +getLight(): int +getLightNormalized(): double +startClock() +stopClock(): long +getClock(int): long +pause(int) +getStatus(): boolean +reconnect() +close() +startLoop(Object,String): int +startLoopWithDelay(Object,String,int): int +stopLoop(int) +stopLoops()</pre>



# Arbeitsauftrag Informatik

Name:

Vorname:

Klasse:

---

## Methoden für Aktoren

`ledOn(int), ledOff(int), ledSwitch(int)`

LED schalten, Parameter: LED-Nummer

`ledSet(int, boolean), ledDim(int, int)`

LED schalten/dimmen, Parameter: LED-Nummer und Wert

`getLEDValue(int), getLEDState(int)`

Rückgabe Wert/Status, Parameter: LED-Nummer

`ledsOff()`

alle LED ausschalten

`playBeep(int)`

Tonausgabe, Parameter: Zeit in Millisekunden

`stopBeep()`

Tonausgabe stoppen

`getLight(), getLightNormalized()`

Rückgabe [mittlerer] Messwert [0, 1023] am Helligkeitssensor

`isPressed(), wasPressed()`

Rückgabe des aktuellen/bisherigen Tastenzustands

`isContacted(), getHall()`

Rückgabe Hallsensorzustands [true,false] bzw. Wert [0, 1023]

## Methoden für Sensoren

`getLight(), getLightNormalized()`

Rückgabe [mittlerer] Messwert [0, 1023] am Helligkeitssensor

`isPressed(), wasPressed()`

Rückgabe des aktuellen/bisherigen Tastenzustands

`isContacted(), getHall()`

Rückgabe Hallsensorzustands [true,false] bzw. Wert [0, 1023]

## Methode zur Hardwaresteuerung

`reconnect(), getStatus(), close()`

Hardwareverbindung wiederaufnehmen/prüfen/beenden

## Methoden für zeitliche Aspekte

`pause(int)`

Programmablauf pausieren, Parameter: Zeit in Millisekunden

`startClock()`

Stoppuhr mit Null starten

`getClock()`

Rückgabe der Laufzeit in Millisekunden

`stopClock()`

Uhr stoppen und Rückgabe der Laufzeit in Millisekunden

## Methoden zur Nebenläufigkeit

Bei der Loop-Nebenläufigkeit wird eine Methode eines Objekts als Endlosschleife quasiparallel zu anderen Prozessen ausgeführt.

`startLoop(Object, String)`

Methode des Objekts nebenläufig ausführen,  
Parameter: Objekt, Methodenname; Rückgabe Loop-ID

`startLoopWithDelay(Object, String, int)`

analog zu `startLoop`, 3. Parameter: Pause in Millisekunden

`stopLoop(int)`

Nebenläufigkeit der Methode beenden, Parameter: LoopID

`stopLoops()`

alle nebenläufigen Methoden stoppen