

A decorative graphic on the left side of the slide, consisting of a network of blue dots of varying sizes connected by thin blue lines, forming a complex web-like structure.

Creating safety.
With passion.

NewTec

Software-Engineering

Projekt-Kodierstandard

**Die folgenden Regeln sind im Rahmen
des praktischen Teils einzuhalten!**

1. Code Style

1.1 Englische Sprache


- Quellcode und Kommentierung *müssen* in Englisch gehalten werden.

1.2 Namensgebung

- Namen für Bezeichner (Module, Includes, Konstanten, Makros, Typen Variablen und Funktionen) *sollen* aussagekräftig und damit leicht zu merken sein.
 - Bei der Benennung *sollte* die natürlichen Sprache Vorbild sein.
- Schleifenvariablen *dürfen nicht* nur mit „i“, „j“, „ii“ oder ähnlichen Kurzbezeichnungen benannt werden.

Module:

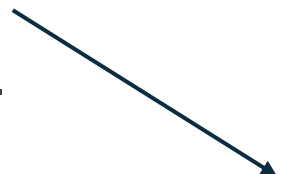
- Selbsterklärend & spezifisch, erster Buchstabe groß, Rest klein.
- Ganze Worte, Zusammensetzung durch Großschreibung („CamelCase“).



```
LineSensor.c  
LineSensor.h
```

Includes:

- Bei der Reihenfolge der Includes werden zuerst Standardbibliotheken aufgeführt, gefolgt von einer Leerzeile.
- Danach die Header-Datei des eigenen Moduls.
- Dann alle weiteren Header-Dateien.



```
#include <avr/io.h>  
  
#include "LineSensor.h"  
#include "Gpio.h"
```



- Konstanten :


- Nur Großbuchstaben werden. Zur Abgrenzung von Wörtern werden Unterstriche verwendet.
- Standardpräfixe: MIN_, MAX_, DEFAULT_, ...



```
#define DEFAULT_DURATION (200)
```


- Makros:

- Nur Großbuchstaben.
- Zusammengesetzte Worte werden mit „_“ getrennt.
- Makros sollen immer geklammert werden.
- Makros sollen immer kommentiert werden.




```
/** Determines the minimum of two comparative values */  
#define MIN(a,b) ((a)<(b)?(a):(b))
```

- Typen :
 - Selbsterklärend & spezifisch, erster Buchstabe groß, Rest klein.
 - Ganze Worte, Zusammensetzung durch Großschreibung („CamelCase“).



```
typedef enum
{
    GPIO_RET_OK = 0,    /**< Ok. */
    GPIO_RET_ERROR      /**< Error. */
} GpioRet;
```

- Aufzählungstypen: nur Großbuchstaben, Wort Trennung mit „_“.
- Strukturkomponenten beginnen mit einem Kleinbuchstaben.



```
typedef struct
{
    UInt8 options;    /**< Options. */
    UInt8 index       /**< Index. */
} DemoStruct;
```


- Variablen :

- Selbsterklärend & spezifisch, beginnen mit einem Kleinbuchstaben.
- Ganze Worte, Zusammensetzung durch Großschreibung („camelCase“).
- Bildet die Variable eine physikalische Einheit ab, soll die Variable mit der Einheit abschließen.
- Modulvariablen beginnen mit einem kleinen „g“ mit lokaler Gültigkeit für das Modul → Deklaration als „static“.

UInt32 sensorValues;



UInt32 gTickCountMs;



- Funktionen :

- Selbsterklärend & spezifisch, beginnen mit einem Kleinbuchstaben.
- Ganze Worte, Zusammensetzung durch Großschreibung („CamelCase“).
- Externe Funktionen beginnen mit dem Modulnamen gefolgt von einem “_”.



```
void initDemoStruct(void)
```

1.8 Erlaubte Datentypen

- Es *müssen* folgende Datentypen, anstelle der intrinsischen Datentypen, verwendet werden:
 - Int8
 - Int16
 - Int32
 - UInt8
 - UInt16
 - UInt32
 - Bool
 - Float32
 - Float64

Hinweis: Entsprechende Definitionen bekommen Sie in der Beistellung zur Verfügung gestellt