artkeller@gmx.de
2018-02-01

Wichtige Kriterien bei der Selektion durch #include <INCLUDE.h>

Mehrere Bibliotheken mit identischen Namen in der Arduino-IDE

Wie lautet die Strategie, Namenskonkurrenz mehrerer gleichnamiger Standard-Bibliotheken aufzulösen?

Beispiel:

```
#include   <SD>
```

Arduino Bibliotheken enthalten in der Datei "library.properties" Angaben über die durch diese Bibliothek unterstützte(n) Architektur(en).

Beispiel:

```
architectures=esp32
```

oder

```
architectures=*
```

Diese Angaben sind offenbar aber nicht allein für die Auswahl der richtigen Bibliothek verantwortlich, wenn Bibliotheksnamen mehrfach vorkommen.

Die Angabe

```
#include   <SD>
```

veranlasst die Suche nach "SD.h" und dabei können unterschiedliche SD Bibliotheken in unterschiedlichen Architekturen gefunden werden.

Aber warum werden die Angaben aus "library.properties" nicht verwendet?

Weil die Arduino IDE zuerst im **Hardware**-Bibliothekspfad der zugehörigen Architektur nach dem Namen sucht.

Beispiel:

Bei der "avr"-Architektur im Pfad

```
"hardware\arduino\avr\libraries\SD"
```

Bei der "esp32"-Architektur im Pfad

```
"hardware\espressif\esp32\libraries\SD"
```

So wird im Falle der Konkurrenz von gleichnamigen Bibliotheken die Reihenfolge aufgelöst.

Findet Arduino bei der Kompilierung von

```
#include   <SD>
```

nun mehrere Headerdateien "SD.h", wird diejenige gewählt, die im Pfadnamen der Architektur vorkommt.
Fehlt diese Bibliothek allerdings, wird als nächstes im Arduino "Default Verzeichnis" der avr-Achitektur gesucht und danach in den allgemeinen "Arduino\libraries" unter Verwendung der oben genannten Architektur Filter.

Das bedeutet, dass im Falle von "SD.h" die entsprechende avr-Bibliothek verwendet wird, falls in der eigentlichen Zielarchitektur, z. B. "esp32" diese fehlt.

Daher müssen nach Konvention alle sog. Standard-Bibliotheken in allen Architekturen existieren, um keine schwer verständlichen Seiteneffekte und Linker-Fehler zu erhalten. Üblicherweise werden Standard-Bibliotheken an die jeweilige Zielarchitektur angepasst.

Das Fehler von Standard-Bibliothek in einer Architektur kann allerdings in der Arduino-Welt bewusst erfolgen. Dadurch liegt Effekt aus der Optimierung der avr-Welt vor, weil die meisten Architekturen dort kompatibel sind und sich der Zugriff auf das "Default Verzeichnis" geradezu anbietet.

Nach der Öffnung der Arduino-IDE für andere Architekturen ist dieser Seiteneffekt unbedingt zu vermeiden.

Fehler bei der Benennung neuer Architekturen

Beliebtes Problem ist die Verwendung falscher "SD.h" und "WiFi.h" in den ESP Architekturen, weil die Architekturnamen in den hardware-Pfaden nicht stimmen.

Beispiel:

Falsch

"hardware\espressif**arduino-esp32**"

Richtig

"hardware\espressif**esp32**"