miniFPC - minimal FPC and RTL

0.0.1

Erzeugt von Doxygen 1.12.0

Kapitel 1

Themen Index

1.1 Themen

Es folgt eine Liste der Themen mit einer Kurzbeschreibung

Kapitel 2

Themen Dokumentation

2.1 Der Free Pascal Compiler - FPC

Themen

- Schlüsselwörter

2.1.1 Ausführliche Beschreibung

2.1.2 Schlüsselwörter

Themen

- Aufzählungen
- Schleifen
- Bedingungen

2.1.2.1 Ausführliche Beschreibung

2.1.2.2 Aufzählungen

Themen

- ENUM
- SET

2.1.2.2.1 Ausführliche Beschreibung

2.1.2.2.2 ENUM

ein enum

2.1.2.2.3 SET

ein set

2.1.2.3 Schleifen

Themen

- DO
- FOR
- LOOP
- UNTIL
- WHULE

2.1.2.3.1 Ausführliche Beschreibung

2.1.2.3.2 DO

ein do

2.1.2.3.3 FOR

ein for

2.1.2.3.4 LOOP

ein loop

2.1.2.3.5 UNTIL

ein until

2.1.2.3.6 WHULE

eine while

2.1.2.4 Bedingungen

Themen

- IF
- ELSE
- CASE

2.1.2.4.1 Ausführliche Beschreibung

2.1.2.4.2 IF

ein IF

2.1.2.4.3 ELSE

ein ELSE

2.1.2.4.4 CASE

ein case

2.2 Der GNU C/C++ Compiler

Themen

- Schlüsselwörter

2.2.1 Ausführliche Beschreibung

Dies ist die zweite Hauptseite.

2.2.2 Schlüsselwörter

Themen

- typedef
- Aufzählungen
- Schleifen
- Bedingungen

2.2.2.1 Ausführliche Beschreibung

2.2.2.2 typedef

Themen

- Varianten

2.2.2.2.1 Ausführliche Beschreibung

`typedef` ist ein Schlüsselwort in C und C++, das verwendet wird, um Aliasnamen für bestehende Datentypen zu erstellen. Es ermöglicht, komplexe Typen zu vereinfachen und fördert die Lesbarkeit des Codes.

2.2.2.2.2 Verwendung

Die grundlegende Syntax für typedef ist:

```
typedef existierender_typ neuer_typ_name;
```

2.2.2.2.3 Beispiel

```
typedef unsigned long ulong;
```

In diesem Beispiel wird `ulong` als Alias für `unsigned long` definiert. Dies ist besonders nützlich, wenn ein Typ in vielen Teilen des Codes verwendet wird und ein kürzerer oder klarerer Name bevorzugt wird.

2.2.2.2.4 Vorteile

- **Kürzere Typnamen:** Typen können kürzer und prägnanter gemacht werden.
- **Lesbarkeit:** Komplexe Typen können unter einem verständlicheren Namen zusammengefasst werden.
- **Flexibilität:** Änderungen an der zugrunde liegenden Typdefinition können zentral vorgenommen werden, ohne dass der gesamte Code geändert werden muss.

2.2.2.2.5 Typische Anwendungsfälle

- Definieren von Aliasnamen für strukturierte Typen:
  ```
  typedef struct {
      int x;
      int y;
  } Point;
  ```
- Vereinfachung von Funktionszeigern:
  ```
  typedef int (*OperationFunc)(int, int);
  ```

Weitere Informationen zu `typedef` finden Sie in der offiziellen C++ Dokumentation.

2.2.2.2.6 Varianten

Themen

- size_t
- wchar_t

2.2.2.2.6.1 Ausführliche Beschreibung

2.2.2.2.6.2 size_t

definiert einen ssss 32

```
typedef unsigned int size_t;
```

2.2.2.2.6.3 wchar_t

Ein Datentyp für breite Zeichen.

wchar_t ist ein spezieller Datentyp in C und C++, der verwendet wird, um breite Zeichen zu speichern, normalerweise für die Arbeit mit Unicode-Zeichen. Die genaue Größe von wchar_t kann je nach System variieren, aber er ist in der Regel groß genug, um alle gültigen Unicode-Zeichen darzustellen.

Der wchar_t Typ wird häufig in Programmen verwendet, die internationalisiert sind und Zeichen in verschiedenen Schriftsystemen unterstützen müssen. Er kann mit Standardfunktionen wie wprintf, wscanf und wcslen verwendet werden.

```
typedef int wchar_t;
```

2.2.2.3 Aufzählungen

Themen

- enum
- enum class

2.2.2.3.1 Ausführliche Beschreibung

Aufzahl

2.2.2.3.2 enum

enum

2.2.2.3.3 enum class

enum class

2.2.2.4 Schleifen

Themen

- for
- while

2.2.2.4.1 Ausführliche Beschreibung

einige schleifen

2.2.2.4.2 for

eine for

2.2.2.4.3 while

ein while

2.2.2.5 Bedingungen

Themen

- if
- else

2.2.2.5.1 Ausführliche Beschreibung

IF Bedingte bearbeten

2.2.2.5.2 if

ifer

2.2.2.5.3 else

ELSE Bedingte bearbeten

Index