Enum 类型

如果一种数据类型的取值只有少数几种可能，并且每种取值都有自己的含义，为了提高代码的可读性，可以将它们定义为 Enum 类型，中文名为枚举。

```
enum colors {RED, GREEN, BLUE};

printf("%d\n", RED); // 0
printf("%d\n", GREEN);  // 1
printf("%d\n", BLUE);  // 2
```

上面示例中，假定程序里面需要三种颜色，就可以使用 `enum` 命令，把这三种颜色定义成一种枚举类型 `colors`，它只有三种取值可能 `RED`、`GREEN`、`BLUE`。这时，这三个名字自动成为整数常量，编译器默认将它们的值设为数字 `0`、`1`、`2`。相比之下，`RED` 要比 `0` 的可读性好了许多。

注意，Enum 内部的常量名，遵守标识符的命名规范，但是通常都使用大写。

使用时，可以将变量声明为 Enum 类型。

```
enum colors color;
```

上面代码将变量 `color` 声明为 `enum colors` 类型。这个变量的值就是常量 `RED`、`GREEN`、`BLUE` 之中的一个。

```
color = BLUE;
printf("%i\n", color); // 2
```

上面代码将变量 `color` 的值设为 `BLUE`，这里 `BLUE` 就是一个常量，值等于 `2`。

typedef 命令可以为 Enum 类型起别名。

```
typedef enum {
  SHEEP,
  WHEAT,
  WOOD,
  BRICK,
  ORE
} RESOURCE;
```

```
RESOURCE r;
```

上面示例中，`RESOURCE`是 Enum 类型的别名。声明变量时，使用这个别名即可。

还有一种不常见的写法，就是声明 Enum 类型时，在同一行里面为变量赋值。

```
enum {
  SHEEP,
  WHEAT,
  WOOD,
  BRICK,
  ORE
} r = BRICK, s = WOOD;
```

上面示例中，`r`的值是`3`，`s`的值是`2`。

由于 Enum 的属性会自动声明为常量，所以有时候使用 Enum 的目的，不是为了自定义一种数据类型，而是为了声明一组常量。这时就可以使用下面这种写法，比较简单。

```
enum { ONE, TWO };

printf("%d %d", ONE, TWO);  // 0 1
```

上面示例中，`enum`是一个关键字，后面跟着一个代码块，常量就在代码内声明。`ONE`和`TWO`就是两个 Enum 常量。

常量之间使用逗号分隔。最后一个常量后面的尾逗号，可以省略，也可以保留。

```
enum { ONE, TWO, };
```

由于 Enum 会自动编号，因此可以不必为常量赋值。C 语言会自动从 0开始递增，为常量赋值。但是，C 语言也允许为 ENUM 常量指定值，不过只能指定为整数，不能是其他类型。因此，任何可以使用整数的场合，都可以使用 Enum 常量。

```
enum { ONE = 1, TWO = 2 };

printf("%d %d", ONE, TWO);  // 1 2
```

Enum 常量可以是不连续的值。

```
enum { X = 2, Y = 18, Z = -2 };
```

Enum 常量也可以是同一个值。

```
enum { X = 2, Y = 2, Z = 2 };
```

如果一组常量之中，有些指定了值，有些没有指定。那么，没有指定值的常量会从上一个指定了值的常量，开始自动递增赋值。

```
enum {
  A,    // 0
  B,    // 1
  C = 4,  // 4
  D,    // 5
  E,    // 6
  F = 3,   // 3
  G,    // 4
  H     // 5
};
```

Enum 的作用域与变量相同。如果是在顶层声明，那么在整个文件内都有效；如果是在代码块内部声明，则只对该代码块有效。如果与使用 `int` 声明的常量相比，Enum 的好处是更清晰地表示代码意图。

版本号 #1
由 壳壳虫 创建于 2021-09-04 22:34:34 UTC
由 壳壳虫 更新于 2022-06-30 13:08:38 UTC