C2X Proposal: WG14 N2496

Title: **Free Positioning of Labels Inside Compound Statements**
Author: **Martin Uecker, University Medical Center Göttingen**
Date: **2020-02-29**

Introduction

Declarations and statements can be freely mixed inside compound statements, but labels are only allowed before statements. Directly before declarations or at the end of a compound statements labels are not allowed. This is an unnecessarily and annoying limitation.

Example:

```
void f(int x)
{
restart:          // label not allowed before declaration
  int z = 3 * x;

  switch (x) {
  case 1:         // label not allowed before declaration
    int y = z;
    if (y == 3)
      goto out;
    if (y > z)
       goto restart;
    break;

  default::       // label not allowed at end of block
     /* do nothing */
  }
out:              // label not allowed at end of block
}
```

Labels Inside Compound Statements

Programmers often work around this limitation by adding a semicolon, i.e. adding a null statement. While this is simple and shows that this is only a syntactical issue, it is annoying to have to add semicolons in such cases. Especially when refactoring code, deleting or moving lines of code can cause a label to move directly before a declaration or at the end of a block, which then causes an error that needs to be fixed by adding a semicolon. For beginners, this surprising limitation makes the language more difficult to learn. For all these reasons, we propose to lift this limitation and to allow labels

everywhere where statements and declarations are currently allowed inside compound statements. This can be achieved by decoupling labels from statements and by specifying that they are followed by an implied null statement.

Labeled Statements

With the proposed change, labels are first class citizens inside compound statements. The concept of labeled statements is then required only for labels occurring outside of compound statements, which is a rarely used feature and then usually confusing. Therefor, it is also suggested to make labeled statements, i.e. labels outside of compound statements, an obsolescent feature.

Examples:

```
switch (x)
  default: foo(x) ;

goto loop;
while (1)
  loop: {  foo(1); }

if (0)
  output: foo(3);
```

NOTE: If such a case should actually occur somewhere, it could always be rewritten in clearer way by adding an additional pair of curly brackets around it to make it part of a compound statement. The first example could be re-written like this:

```
switch (x)
{
   default:
     foo(x);
}
```

Suggested Wording Changes

6.8.1 Labeled statements

Syntax

```
label:
     identifier :
     case constant-expression :
     default :
```

labeled-statement:

```
     label ~~identifier :~~ statement
     label ~~case constant-expression :~~ statement
     label ~~default :~~ statement
```

6.8.2 Compound statement

Syntax

```
compound-statement:
     { block-item-list opt }

block-item-list:
     block-item
     block-item-list block-item

block-item:
     declaration
     statement
     label
```

Semantics

A compound statement is a block. **A label shall be translated as it were followed by a null statement.**

6.8.6.2 The continue statement

(in the examples, the now needless semicolon can be removed)

Optional Additional Change

6.11.X Labeled Statements

Labels outside of compound statements are an obsolescent feature.