Forum:Hack Forums

Thread: Why isn't my switch statement currency convertor working?

SureDawgg - *May 25, 2016, 11:00 am*
Delete this thread please, thank you

Raijin - *May 25, 2016, 11:04 am*
I haven't done any C programming in a while but are you not missing the ending return statement?
return 0;

?Opulence - *May 25, 2016, 11:17 am*
look at this example:

```
int caseSwitch = 1;
switch (caseSwitch)
{
    case 1:
        Console.WriteLine("Case 1");
        break;
    case 2:
        Console.WriteLine("Case 2");
        break;
    default:
        Console.WriteLine("Default case");
        break;
}
```

You need the default case, this handles cases where the caseSwitch isn't one of the listed cases..

Das Kartoffel - *May 25, 2016, 11:42 am*
You forgot the 'break' keywords, as follow:

PHP Code
```
<?php
   switch (currency) {
   case 'y':
      cout << yen*value << "\n";
            break;
   case 'e':
      cout << euro*value << "\n";
            break;
   case 'p':
      cout << pound*value << "\n";
            break;
   default:
      cout << "failure\n";
            break;
   }
```

```
    return 0;
}
```

?Opulence - *May 25, 2016, 11:42 am*
OH I GOT IT. You forgot to put 'break;' after each case, refer to my example I found.

6577657534735 - *May 25, 2016, 1:54 pm*
Missing a break is annoying. Compiler should give a warning.

0xDEAD10CC - *May 25, 2016, 4:09 pm*
(May 25, 2016 01:54 PM)Abraluna Wrote: *[CITE]* Missing a break is annoying. Compiler should give a warning.
NO - Definitely not! ?

Think about it... Would you want the compiler to warn you that you missed typing 'else if' instead of 'if' the second time here:
Code if (conditionA)
{

}

if (conditionB_which_is_not_related_to_A)
{

}
^ Same thing basically.

There's some cases where you want a case statement to fall through... Having compiler warnings in cases like that which clutters the view of more important warnings would be annoying. Have you ever tried enabling the most verbose warning output and included <Windows.h>?

6577657534735 - *May 26, 2016, 11:43 pm*
(May 25, 2016 04:09 PM)0xDEAD10CC Wrote: *[CITE]* NO - Definitely not! ?

Think about it... Would you want the compiler to warn you that you missed typing 'else if' instead of 'if' the second time here:
Code if (conditionA)
{

}

if (conditionB_which_is_not_related_to_A)
{

}
^ Same thing basically.

There's some cases where you want a case statement to fall through... Having compiler warnings in cases like that which clutters the view of more important warnings would be annoying. Have you ever tried enabling the most verbose warning output and included <Windows.h>?
Yeah what was I thinking. Very dumb on my part

Corios - *May 27, 2016, 2:36 pm*

(May 26, 2016 11:43 PM)Abraluna Wrote: [CITE] Yeah what was I thinking. Very dumb on my part

No, I agree with you. Falling through from one case labels code to another is pretty evil, I would prefer it give the warning. That shouldn't be intended functionality, unlike two ifs

Say you have:

```
Code Switch num
    Case 1:
    Case 2:
        DeleteAllFees()
    Case 3:
    Case 4:
        CreateNewStudents()
```

If you are working with students and not fees, it will appear that it's case 3 or 4. You may not realize you've been deleting all fees the entire time. Obviously not a good example as those should usually never be in the same switch, its just for demonstration. You should not need to check above your cases every time to make sure nothing else is happening.

0xDEAD10CC - *May 27, 2016, 3:05 pm*

(May 27, 2016 02:36 PM)Corios Wrote: [CITE] No, I agree with you. Falling through from one case labels code to another is pretty evil, I would prefer it give the warning. That shouldn't be intended functionality, unlike two ifs

Say you have:

```
Code Switch num
    Case 1:
    Case 2:
        DeleteAllFees()
    Case 3:
    Case 4:
        CreateNewStudents()
```

If you are working with students and not fees, it will appear that it's case 3 or 4. You may not realize you've been deleting all fees the entire time. Obviously not a good example as those should usually never be in the same switch, its just for demonstration. You should not need to check above your cases every time to make sure nothing else is happening.

So you're willing to adopt a compiler warning that will end up being redundant in lots of optimized code for the purpose of programmer stupidity? It should be pretty obvious in your case that things are getting deleted, and if the programmer was competent enough to breakpoint and step through the code it would be a very easy fix.

It would be silly.. How is it to be considered "evil"? It can lead to more clear code opposed to if statements that do the same thing, and end up being more optimized also. This is like the people who keep saying that goto is evil. It's only as evil as you make it but it has a place in the language and its usages still as you can see in lots of Linux kernel code even for cleanup and cleanly exiting nested scopes without introducing more locals.

And how is that semantically not like 2 if's?

```
Code if (x == 1 || x == 2)
{
    DeleteAllFees();
```

```
}
if (x == 3 || x == 4)
{
    CreateNewStudents();
}
```

In more complex situations a switch statement with fall-throughs is much more readable. If code is properly commented too it can help with re-assuring the intended logic if someone is concerned.

In C#'s case it was removed due to programmer incompetence, and people later accepted that the use of a goto was an acceptable workaround:

```
Code switch (/*...*/) {
    case 0: // shares the exact same code as case 1
    case 1:
        // do something
        goto case 2;
    case 2:
        // do something else
        goto default;
    default:
        // do something entirely different
        break;
}
```

Lastly, here is a great example: https://en.wikipedia.org/wiki/Duff%27s_device

Remix - *May 27, 2016, 3:43 pm*

(May 27, 2016 02:36 PM)Corios Wrote: *[CITE]* No, I agree with you. Falling through from one case labels code to another is pretty evil, I would prefer it give the warning. That shouldn't be intended functionality, unlike two ifs

Say you have:

```
Code Switch num
    Case 1:
    Case 2:
        DeleteAllFees()
     Case 3:
     Case 4:
         CreateNewStudents()
```

If you are working with students and not fees, it will appear that it's case 3 or 4. You may not realize you've been deleting all fees the entire time. Obviously not a good example as those should usually never be in the same switch, its just for demonstration. You should not need to check above your cases every time to make sure nothing else is happening.

What if you wanted a fall through?

Corios - *May 27, 2016, 6:24 pm*

(May 27, 2016 03:43 PM)Remix Wrote: *[CITE]* What if you wanted a fall through?

Then you shouldn't be using a switch, a switch is meant for a multi-branch flow, if your case sled is going through multiple branches, that doesn't sound right, now does it?

I don't know of another language that allows it (albeit I only really use about 5 languages so there are probably a few more that support break structures and allow this)

0xDEAD10CC - *May 27, 2016, 6:57 pm*

(May 27, 2016 06:24 PM)Corios Wrote: *[CITE]* Then you shouldn't be using a switch, a switch is meant for a multi-branch flow, if your case sled is going through multiple branches, that doesn't sound right, now does it?

I don't know of another language that allows it (albeit I only really use about 5 languages so there are probably a few more that support break structures and allow this)
Explain how it's not right. There's absolutely nothing that says it isn't.

Quote:... a switch is meant for a multi-branch flow

This does not mean that a branch can't lead to the same code path, and if you know how a switch works internally as a jump table there's no reason as to why it should be avoided, but you didn't read my previous post either: https://en.wikipedia.org/wiki/Duff%27s_device

Lots of programmers still use it, and the Eclipse Foundation uses code that does too: https://github.com/eclipse/ptp/blob/master/debug/org.eclipse.ptp.debug.sdm/libaif/aggregate.c

It's not uncommon and there's no reason to not use it where it makes sense. If statements would be a horrible alternative, less optimized, and more ugly to read... Again, it comes down to programmer stupidity. If one has the mindset that you shouldn't be able to do it because C# by default disallows this, then stick to C# or whatever other language it is that you're used to. Languages are not supposed to conform to standards from another language and using that as a reason for why it should be disallowed or avoided is ridiculous.

What you are suggesting with the warning about break statements in switch statements -- is taking something that is completely capable of doing something that may be desirable (with the added benefit of being more optimized), and putting training wheels on it so to speak to provide an obstacle that doesn't need to be there in the first place. Nobody goes mountain biking with training wheels.

Remix - *May 27, 2016, 11:03 pm*

(May 27, 2016 06:24 PM)Corios Wrote: *[CITE]* Then you shouldn't be using a switch, a switch is meant for a multi-branch flow, if your case sled is going through multiple branches, that doesn't sound right, now does it?

I don't know of another language that allows it (albeit I only really use about 5 languages so there are probably a few more that support break structures and allow this)
That means you are not allowed to execute the same stuff for multiple cases?
Imagine you had 15 cases which should execute the same code
Would you write it 15 times?

Kantos - *June 1, 2016, 10:16 am*

All the C switch statement does is simply jump to the case label that matches the control expression value if it exists and continues execution from there. Some languages give each case its own block and exit the switch once the instructions in that block have completed, and require an explicit instruction for any fallthrough.