Long version explaining everything

For the last 1-2 years I've been using C#, C and friends alongside each other and I think I found the most annoying thing about C#, it's CS0031. Or at least CS0031 is a symptom of a problem that exists in C++, too. It is the problem of not being able to write **simple** code. In C I can do something like

```
FILE *f = open_file();
if(f) read_from(f);
else return error;
```

while in C# I would have to do something like

```
try
{
    FileStream f = OpenFile();
    ReadFrom(f);
}
catch
{
    return error;
}
```

And that isn't even the worst part about it, because you could still argue that exceptions are better than returning zero values. But there are many use-cases where you could produce just simpler and functionally 100% equal code, I just grep-ed through some projects of mine and the "== 0" syndrome occures a pretty often, just all the `== 0`, `!= 0` and `== 1` is only there because of CS0031. `variable == 0` can just be `!variable`, `variable != 0` can be `variable` and `variable == 1` is usually just another way for saying `variable != 0`, which is `variable`. (for classes or nullables replace all '0's by "null"s)

My conclusion from all of this is that for reducing the anger of C# developers it should be allowed to write `while(1)` and `if(!f) return error;`, at least with some kind of preprocessor instruction or compiler flag.

TLDR

It has caused many of my angry moments that in C# you can't use your non-binary values (actually all non-`bool` values) as booleans, while C allows you to check if something is 0/null by using `if(variable)`, you **must** use `if(variable == null)` or `if(variable == 0)` in C#. I think there should be a preprocessor instruction or compiler flag for that.