Does generated code need to be human readable?

Asked 16 years, 3 months ago Modified 8 years, 5 months ago

Viewed 2k times

7

I'm working on a tool that will generate the source code for an interface and a couple classes implementing that interface. My output isn't particularly complicated, so it's not going to be hard to make the output conform to our normal code formatting standards.

But this got me thinking: how human-readable does auto-generated code need to be? When should extra effort be expended to make sure the generated code is easily read and understood by a human?

In my case, the classes I'm generating are essentially just containers for some data related to another part of the build with methods to get the data. No one should ever need to look at the code for the classes themselves, they just need to call the various getters the classes provide. So, it's probably not too important if the code is "clean", well formatted and easily read by a human.

However, what happens if you're generating code that has more than a small amount of simple logic in it?

language-agnostic code-generation readability

Share

Improve this question

Follow

edited Jul 16, 2016 at 17:07

Richard Chambers
17.5k ● 4 ● 91 ● 118

asked Sep 15, 2008 at 14:13

Herms
38.7k ● 13 ● 79 ● 104

22 Answers

Sorted by: Highest score (default)

18

I think it's just as important for generated code to be readable and follow normal coding styles. At some point, someone is either going to need to debug the code or otherwise see what is happening "behind the scenes".

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:15

Scott Dorman
42.5k ● 12 ● 81 ● 112

8

Yes!, absolutely!; I can even throw in a story for you to explain why it is important that a human can easily read the auto generated code...

I once got the opportunity to work on a new project. Now, one of the first things you need to do when you start writing code is to create some sort of connection and data representation to and from the database. But instead of just writing this code by hand, we had someone who had developed his own code generator to automatically build base classes from a database schema. It was really neat, the tedious job of writing all this code was now out of our hands... The only problem was, the generated code was far from readable for a *normal* human.

Of course we didn't about that, because hey, it just saved us a lot of work. But after a while things started to go wrong, data was incorrectly read from the user input (or so we thought), corruptions occurred inside the database while we where *only* reading. Strange.. because reading doesn't change any data (again, so we thought)...

Like any good developer we started to question our own code, but after days of searching.. even rewriting code, we could not find anything... and then it dawned on us, the auto generated code was broken!

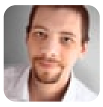
So now an even bigger task awaited us, checking auto generated code that no sane person could understand in a reasonable amount of time... I'm talking about non indented, really bad style code with unpronounceable variable and function names... It turned out that it would even be faster to rewrite the code ourselves, instead of trying to figure out how the code actually worked.

Eventually the developer who wrote the code generator remade it later on, so it now produces *readable* code, in case something went wrong like before.

Here is a link I just found about the topic at hand; I was acctually looking for a link to one of the chapters from the "pragmatic programmer" book to point out why we looked in our code first.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:35



sven
18.3k ● 10 ● 52 ● 62

Just run the autogenerated code trough a beautifier can make a lot of difference, then a lot of the indentions get right anyway. Then to have the code generator insert some \n (new line) is not that bad either ;-) – Johan Mar 17, 2009 at 5:39

3

Yes it does. Firstly, you might need to debug it -- you will be making it easy on yourself. Secondly it should adhere to any coding conventions you use in your shop because someday the code might need to be changed by hand and thus become human code. This scenario typically ensues when your code generation tool does not cover one specific thing you need and it is not deemed worthwhile modifying the tool just for that purpose.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:21

Sklivvz
31.1k ● 24 ● 118 ● 174

2

I think that depends on how the generated code will be used. If the code is not meant to be read by humans, i.e. it's regenerated whenever something changes, I don't think it has to be readable. However, if you are using code generation as an intermediate step in "normal" programming, the generated could should have the same readability as the rest of your source code.

In fact, making the generated code "unreadable" can be an advantage, because it will discourage people from "hacking" generated code, and rather implement their changes in the code-generator instead—which is very useful whenever you need to regenerate the code for whatever reason and not lose the changes your colleague did because he thought the generated code was "finished".

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:17

Anders Sandvig

21k ● 16 ● 61 ● 74

2 I'm not sure obfuscating it is a good way to discourage people from "hacking" it. I just stuck a big comment at the top of the file saying that the code is regenerated at build so changes will be lost. – Herms Sep 23, 2008 at 13:40

Good or not, I speak from personal experience. I have several times observed developers "hacking" generated code instead of changing the generator implementation because they didn't understand the code generator and were too lazy to spend time learning it. – Anders Sandvig Sep 24, 2008 at 21:07

Even people who do understand the code generator are sometimes too lazy to update, recompile and rerun it. – Anders Sandvig Sep 24, 2008 at 21:07

If you needed to debug the combination of hand-rolled code & generated code, you could always instrument the generated code with conditionally enabled logging or some other form of optional diagnostics. If you were generating code from some algorithm that produces huge amounts of code, logging would be better than having to source-level debug it. – Dafydd Rees Nov 23, 2009 at 11:42

1 If the code is automatically generated, any changes will be destroyed in the next build, assuming you don't check in generated code, so your point is moot, the "rogue" developer will only be able to make changes on his own copy of the system, which is useful many times, like when trying to debug the generated code. – Ruan Mendes Sep 23, 2010 at 17:00

2

Look up active code generation vs. passive code generation. With respect to passive code generation, **absolutely yes, always.** With regards to active code generation, *when* the code achieves the goal of being transparent, which is acting exactly like a documented API, then no.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:25

Jesse Millikan
3,135 ● 1 ● 22 ● 32

1

I would say that it is imperative that the code is human readable, unless your code-gen tool has an excellent debugger you (or unfortunate co-worker) will probably by the one waist deep in the code trying to track that oh so elusive bug in the system. My own excursion into 'code from UML' left a bitter tast in my mouth as I could not get to grips with the supposedly 'fancy' debugging process.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:18

TK.
47.8k ● 47 ● 121 ● 148

1

You will kill yourself if you have to debug your own generated code. Don't start thinking you won't. **Keep in mind that when you trust your code to generate code then you've already introduced two errors into the system - You've inserted yourself twice.**

There is absolutely NO reason NOT to make it human parseable, so why in the world would you want to do so?

-Adam

Share Improve this answer Follow

edited Sep 15, 2008 at 14:28

answered Sep 15, 2008 at 14:19

Adam Davis
93.5k ● 60 ● 271 ● 333

Sure there's a reason. It can take more time to write the generator such that it outputs readable code. Depending on the extra effort, and the usage of the output, it may not be worth spending the extra effort. – Herms Sep 23, 2008 at 13:42

But were not talking 100% beautiful code, the level should be readable code. A big difference there! – Johan Mar 17, 2009 at 5:45

1

One more aspect of the problem which was not mentioned is that the generated code should also be "version control-friendly" (as far as it is feasible).

I found it useful many times to double-check diffs in generated code vs the source code.

That way you could even occasionally find bugs in tools which generate code.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:59

squadette

8,276 ● 4 ● 30 ● 39

In this case the generated code isn't version controlled. It's generated as part of the build. But that is a good point. At one point the old version of the generator was creating things in a random order. Just making it sort alphabetically made thigns nicer. – Herms Sep 15, 2008 at 15:45

0

It's quite possible that somebody in the future will want to go through and see what your code does. So making it somewhat understandable is a good thing.

You also might want to include at the top of each generated file a comment saying how and why this file was generated and what it's purpose is.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:17

Mattias

5,189 ● 4 ● 24 ● 18

0

Generally, if you're generating code that needs to be human-modified later, it needs to be as human-readable as possible. However, even if it's code that will be generated and never touched again, it still needs to be readable enough that you (as the developer writing the code generator) can debug the generator - if your generator spits out bad code, it may be hard to track down if it's difficult to understand.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:17

Adrian
46.2k ● 6 ● 117 ● 108

0

I would think it's worth it to take the extra time to make it human readable just to make it easier to debug.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:18

Jacksonh
378 ● 1 ● 6

0

Generated code should be readable, (format etc can usually be handled by a half decent IDE). At some stage in the codes lifetime it is going to be viewed by someone and they will want to make sense of it.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:19

Ron Tuffin
54.5k ● 24 ● 67 ● 78

0

I think for data containers or objects with very straightforward workings, human readability is not very important.

However, as soon as a developer may have to read the code to understand how something happens, it needs to be readable. What if the logic has a bug? How will anybody ever discover it if no one is able to read and understand the code? I would go so far as generating comments for the more complicated logic sections, to express the intent, so it's easier to determine if there really is a bug.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:19

David Thibault
8,736 ● 3 ● 39 ● 51

0

Logic should always be readable. If someone else is going to read the code, try to put yourself in their place and see if you would fully understand the code in high (and low?) level without reading that particular piece of code.

I wouldn't spend too much time with code that never would be read, but if it's not too much time i would go through the generated code. If not, at least make comment to cover the loss of readability.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:19

Vargen
726 ● 2 ● 7 ● 20

0

If this code is likely to be debugged, then you should seriously consider to generate it in a human readable format.

Share Improve this answer

Follow

answered Sep 15, 2008 at 14:19

Romain Verdier

13k ● 7 ● 59 ● 77

0

There are different types of generated code, but the most simple types would be:

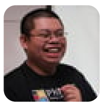
1. Generated code that is not meant to be seen by the developer. e.g., xml-ish code that defines layouts (think .frm files, or the horrible files generated by SSIS)
2. Generated code that is meant to be a basis for a class that will be later customized by your developer, e.g., code is generated to reduce typing tedium

If you're making the latter, you *definitely* want your code to be human readable.

Classes and interfaces, no matter how "off limits" to developers you think they should be, would almost certainly fall under generated code type number 2. They will be hit by the debugger at one point of another -- applying code formatting is the least you can do the ease that debugging process when the compiler hits those generated classes

Share Improve this answer Follow

answered Sep 15, 2008 at 14:21



Jon Limjap
95.3k ● 15 ● 103 ● 153

0

The whole point of generated code is to do something "complex" that is easier defined in some higher level language. Due to it being generated, the actual maintenance of this generated code should be within the subroutine that generates the code, not the generated code.

Therefor, human readability should have a lower priority; things like runtime speed or functionality are far more important. This is particularly the case when you look at tools like bison and flex, which use the generated code to pre-generate speedy lookup tables to do pattern matching, which would simply be insane to manually maintain.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:23

You forget about repetitive code that a lot of people cut and paste.... That code is not complex, it is just boring to write the same translation code 50 times with different names (I have seen system that does this), and the quality goes up if that code is auto generated instead. – Johan Mar 17, 2009 at 5:49

0

Like virtually everybody else here, I say make it readable. It costs nothing extra in your generation process and you (or your successor) will appreciate it when they go digging.

For a real world example - look at anything Visual Studio generates. Well formatted, with comments and everything.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:24

MarcE
3,731 ● 1 ● 25 ● 29

0

Generated code is code, and there's no reason any code shouldn't be readable and nicely formatted. This is cheap especially in generated code: you don't need to apply formatting yourself, the generator does it for you everytime! :)

As a secondary option in case you're really that lazy, how about piping the code through a beautifier utility of your choice before writing it to disk to ensure at least some level of consistency. Nevertheless, almost all good programmers I know format their code rather pedantically and there's a good reason for it: there's no write-only code.

Share Improve this answer Follow

answered Sep 15, 2008 at 14:26

yason

0

Absolutely yes for tons of good reasons already said above. And one more is that if your code need to be checked by an assesor (for safety and dependability issues), it is pretty better if the code is human redeable. If not, the assessor will refuse to assess it and your project will be refected by authorities. The only solution is then to assess... the code generator (that's usually much more difficult ;))

Share Improve this answer Follow

answered Sep 15, 2008 at 14:32

Didou

0

It depends on whether the code will only be read by a compiler or also by a human. In addition, it matters whether the code is supposed to be super-fast or whether readability is important. When in doubt, put in the extra effort to generate readable code.

Share Improve this answer Follow

answered Sep 15, 2008 at 15:05

Wouter Lievens

4,029 ● 5 ● 43 ● 67

0

I think the answer is: it depends.

*It depends upon whether you need to configure and store the generated code as an artefact. For example, people very rarely keep or configure the object code output from a c-compiler, because they know they can reproduce it from the source every time. I think there may be a similar analogy here. *It depends upon whether you need to certify the code to some standard, e.g. Misra-C or DO178. *It depends upon whether the source will be generated via your tool every time the code is compiled, or if it will you be stored for inclusion in a build at a later time.

Personally, if all you want to do is build the code, compile it into an executable and then throw the intermediate code away, then I can't see any point in making it too pretty.

Share Improve this answer

Follow

answered Sep 15, 2008 at 15:16

Jon Mills

1,875 ● 4 ● 20 ● 28