

# Forum:[Hack Forums](#)

Thread: [\[Examples\] HowTo Write Clear Source Code](#)

**My Life** - January 15, 2011, 3:53 am

## HowTo Write Clear Source Code

I am starting to write this thread in order to hope seeing in the future clear and understandable source code in the different programming sections. Also I am writing this for the new programmers who want to learn how their code can be more understandable as well as for the experience programmers, despite the fact that I am pretty sure they know these tips, who want to make perfect their code.

The following examples are in C but , I want to believe that even non-C programmers can understand the code.

So let's start:

? Write your program structrure by using empty lines

Use them when you use brankets for example: in functions/procedures , loops or if statements

```
Code int main()
{
    int i=123;

    if ( i != 123 )
        printf("Impossible");
    else
        return 0;
}
```

Ofcourse you can write it even more detailed like this:

```
Code int main()
{
    int i=123;

    if ( i != 123 )
    {
        printf("Impossible");
    }
    else
    {
        return 0;
    }
}
```

? Add comments to your source code

Believe me , your code will much more understandable when you are writing many lines of code. Also , if you code has bugs and you are trying to spot and fix them , the comments can help you and lead you to the success. Finally your code will be undestandable by third persons who may try to read your code and follow your mind.

Example:

```
Code if (key_code == 8) /* if key_code = [DEL] then ... */
{
    /* Do the job */
    ....
}
```

It is much easier to remember that this if statement checks if the key\_code variable is the Delete Button (eg: Keylogger).

Although we shouldn't overdo it like here:

```
Code /* Now we increase the ammount of the variable The_number_of_cats by 1 */
The_number_of_cats = The_number_of_cats + 1;
This will be very tiring if we start to read a bunch of these comments. We aim to comment the work
of our algorithm , even comment the command use , without useless comments.
```

? Remember to define variable names that reminds you the use/purpose of the var

In many languages you can define constand variables and use them instead of 'naked' numbers. ( In C language you can use the "#define" command in order to achive this.

```
Code change_background_color( 4 );
You must be wondering here : "What Color is the number 4 ?"
```

```
Code /* screen.h
* Constants and prototypes for the screen driver */
```

```
#define BLACK    0x00
#define BLUE     0x01
#define GREEN    0x02
#define CYAN     0x03
#define RED      0x04
#define WHITE    0x0F
```

```
int main()
{
    change_background_color( GREEN );
    return 0;
}
```

As you can see here , it is obvious that we change the color to green.

? Do not use long variable names

Just imagine if you have to use it many times in your code something like this:

Code `int THE_TOTAL_NUMBERS_OF_CATS_IN_THE_STREET = 0;`

This can be very tiring when you read/edit the source code. Instead of this , you can use something like:

Code `int total_cats = 0;`

or even better:

Code `int cCats = 0;`

? Do not be carried away to optimize your source code with the result to loose your clear code

If you found out a new way (better algorithm) to do some work it is ok to write it , but you should to write/edit it so I will look clear and before that, comment the code by giving a generic idea on that it actually does.

Code `/* Fib(int value): Finds a given value in the array */`

`int Search(int *array, int value)`

```
{  
    //...  
}
```

A great idea is to use the Hungarian Notation for the Variable Names:

[http://en.wikipedia.org/wiki/Hungarian\\_notation](http://en.wikipedia.org/wiki/Hungarian_notation)

Links

[GNU Coding Standards - Wikipedia, the free encyclopedia](#)

[Programming style - Wikipedia, the free encyclopedia](#)

[Writing clean code](#)

[Six ways to write more comprehensible code](#)

Any Feedback is Welcome . If you would like something to be fixed/modified feel free to post your suggestions :)

**Sia** - January 15, 2011, 4:02 am

Nice, in every programming book I've read, PureBasic, C#, VB, and so on, they where always talking about writing clean code.

**My Life** - January 15, 2011, 4:07 am

(Jan 15, 2011 05:02 AM)VirtualDUDE Wrote: [\[CITE\]](#) Nice, in every programming book I've read, PureBasic, C#, VB, and so on, they where always talking about writing clean code.

Yeah , because in every programming language a clear code is a must ;)

**Himself** - January 15, 2011, 3:35 pm

I guess you should mention about other notations also,and make a short description of them.

Ahh and about comments,try to write something large in Asm and look at it after one week ,you won't have an idea what is it.After that you'll learn to comment stuff ;P

**Knautschtabak** - January 15, 2011, 11:19 pm

Using many comments may also be a hint of smelling code.

You can do a lot understanding by naming your variables and functions or methods right and meaningful. If you need to comment several sections of one function/method, maybe you should

extract this section to another function/method instead.

**infamousnoone** - January 16, 2011, 12:10 am

(Jan 15, 2011 04:51 PM)Hidden Dragon Wrote: [\[CITE\]](#) Comments are vital. Whenever I write I try to comment a lot, especially if it's something I'm going to need to keep updating or be giving source code away. Especially when writing assembly. If you don't comment that you probably won't know what is going on at all.

Assembly, I always comment. If I intend for the code to be used by others, I comment, or provide documentation on how to use it and how it functions.

Otherwise, I don't comment anything because I'm the only one working with it and I know my way around everything I write. Also, @ the OP (while I realized you're tone slightly changes by the end of the post...), optimized code > readable code.

Also, when I do comment my code I don't go all out and explain in detail what everything does. I assume if I intended for my code to be reused for someone, that someone isn't a half-assed coder that can't read what's going on (in other words, I do comment; but I don't over-comment).

**My Life** - January 16, 2011, 2:38 am

(Jan 16, 2011 01:10 AM)infamousnoone Wrote: [\[CITE\]](#) Assembly, I always comment. If I intend for the code to be used by others, I comment, or provide documentation on how to use it and how it functions.

Otherwise, I don't comment anything because I'm the only one working with it and I know my way around everything I write. Also, @ the OP (while I realized you're tone slightly changes by the end of the post...), optimized code > readable code.

Also, when I do comment my code I don't go all out and explain in detail what everything does. I assume if I intended for my code to be reused for someone, that someone isn't a half-assed coder that can't read what's going on (in other words, I do comment; but I don't over-comment).

The purpose of the comments is to understand your code. Some people can understand it with a short description, and some with a large one.

It highly depends on the experience of the coder. Also always optimize code > readable code, but I just mention that we shouldn't forget to comment it afterwards ;)

**Sia** - January 16, 2011, 3:49 am

(Jan 15, 2011 04:35 PM)Himself Wrote: [\[CITE\]](#) I guess you should mention about other notations also, and make a short description of them.

Ahh and about comments, try to write something large in Asm and look at it after one week, you won't have an idea what it is. After that you'll learn to comment stuff ;P

Well in VB comments seem as little useless cause of the easy code being written, but everything serves its purpose and so do comments.

**infamousnoone** - January 16, 2011, 9:04 am

(Jan 16, 2011 03:38 AM)WesternGuy Wrote: [\[CITE\]](#) The purpose of the comments is to understand your code. Some people can understand it with a short description, and some with a large one.

It highly depends on the experience of the coder. Also always optimize code > readable code, but I just mention that we shouldn't forget to comment it afterwards ;) Like I said, if I intend on publishing something for public use, I'll comment it to a degree where a competent-level programmer should understand what is going on.

However, I've written 20k line projects before that have likely under 50 lines of comments because I intended on nobody looking at that source for me.

Likely, if I consider the code unreadable to myself, I'll heavily comment it, but that's very rare and

usually only ever pops up in assembly.

Just my 2 cents.

**Uncomplicated** - *January 16, 2011, 11:11 am*

Your tutorial makes sense to me. It makes code revision a lot more easier , if they are clean and well commented.