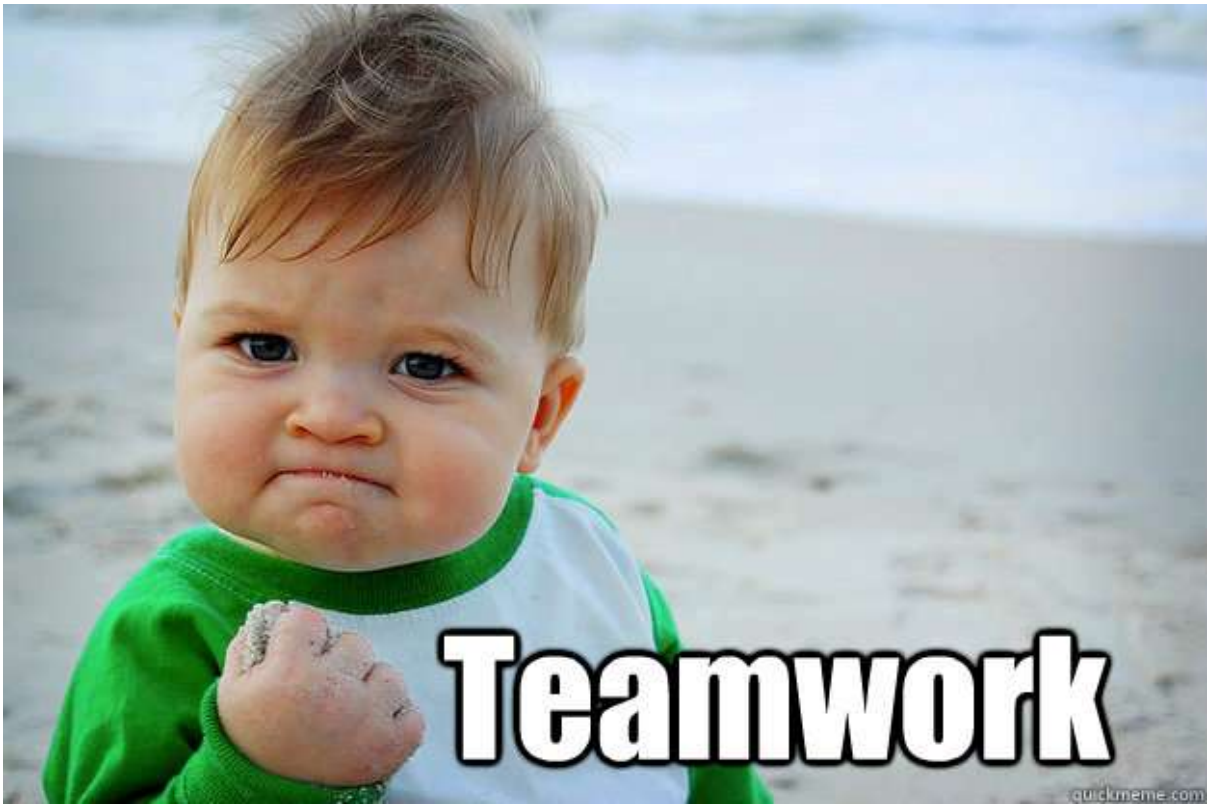
Notes before begin coding in a teamwork

Common issues associated to programming in a teamwork



Ahmad Ahmadi
Ahmadi.dt@gmail.com

A member of Best programmer group
August 22, 2014

Table of Contents

Preface 3
Introduction 4
Common issues in programming teams 5
1- Naming conventions and code styles 5
2- Code sharing and versioning 7
3- Online compilers 9
Conclusion 10

Preface

This document aimed to tell you about common issues and problems related to teamwork in programming and solutions to solve them. Most teams around the world either small or big have some rules and tools they use for best collaboration in their group and to avoid relevant problems. At first I try to tell you those common problems and make you feel that situation. Then I will let you know about solutions.

Understanding the fundamental ideals, principles, and techniques is the essence of a good programmer. Only well-designed code has a chance of becoming part of a correct, reliable, and maintainable system.

Bjarne Stroustrup - Programming principles and practice using C++ - Page xxvii

Introduction

We are going to do actual work, coding! To have a straight progress and a nice collaboration we need to know something about coding rules, conventions and the value of code sharing in a teamwork. They are common issues and you will face related problems soon. I prefer my teammates not to experience them as I did and use their time and power to focus on our main aim; make the world a better place!

Please read this document carefully and try to imagine described situations and problems. When you don't believe, you will face those problems, you won't pay attention and you won't use given solutions.

Common issues in programming teams

1- Naming conventions and code styles

Consider a team with two members (smallest group). In each program there are almost more than 100 functions, variables, classes, etc. They don't use descriptive names for those parts. So can member A understand member B's codes? Will even member B remember what those parts s/he wrote means and what they do?

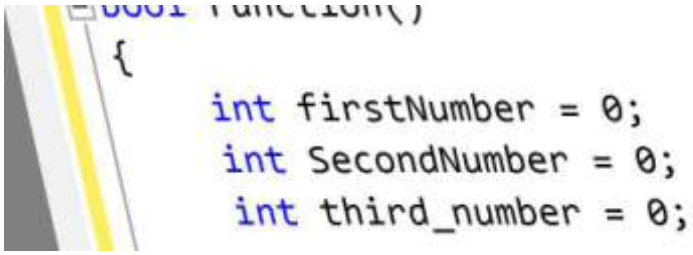
This can be even worse if they are from different countries (as we are!). Member A used localized names like Hazf (which means Delete in Persian). Will member B understand what that means?
Remember that programming is a repetitive process, so your code will change (by you or others) over the time. So when your code could not be understood by others, you just wrote a shit.

Maintaining smelly legacy code...

Like when you write a document and you bold headers, italic notes, etc. you have to do so in your codes. It helps you and other members to easily understand. (Strict people like me have serious problems with non well-formed codes and affect their efficiency). For instance take a look at following code snippets. Which one you feel good with?

```
//Ugh!
for(int i=0;i<10;i++){
for(int j=0;j<10;j++){
if(i<j)
cout<<j;
else
cout<<I;
}}
```

```
//Hmmm. I like it!
for(int i=0;i<10;i++)
{
   for(int j=0;j<10;j++)
   {
      if(i<j)
         cout<<j;
      else
         cout<<I;
   }
}
```



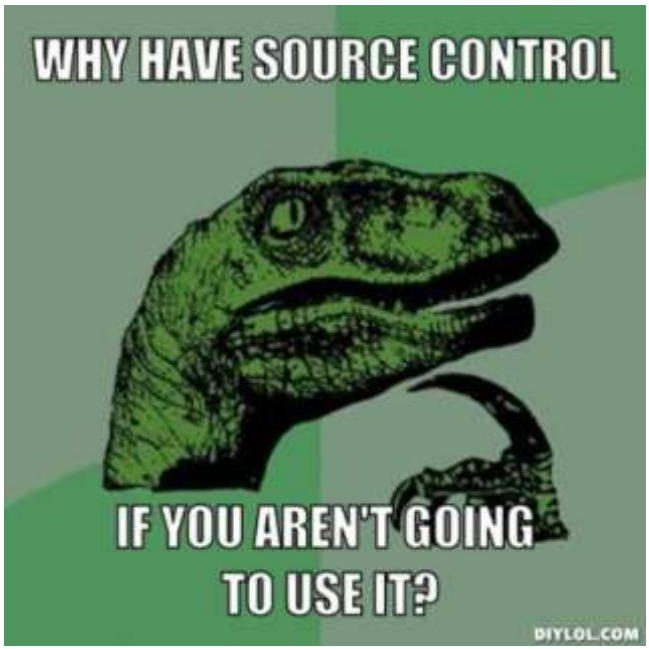
You have to read more about how to name code parts, write comments, name files, etc. to become a better programmer and be able to become part of a successful application development.

Please read the following page and try to apply those rules in your codes.
http://geosoft.no/development/cppstyle.html

2- Code sharing and versioning

Think about other teams or companies. How they work? How big teams work on a project? Do they write their codes in a text document and give that to other teammates?
If a bug has found in program, how they will solve them? If several members change a file, how they can find that who, when, where changed the code? How they can find which change broke the program?

They all use Source control or Version control systems. That enables teams to work on a project together without the fear of losing codes. They also have other benefits. Keep track of file changes which enables you to know how much code has changed, which parts has changed and who changed each part. There are also lots of other benefits that we don't need to know them right now.



How we can use source control in our group?
With source controls, we have a public place that all members have access to, and they can put their codes there. And when someone has a question about his/her codes, s/he can put code in repository (where all files will place in a source control) and ask other teammates to help. Then other members can fix/improve that code and write notes for their edits. In over of time, a file will change several times. And everyone can visit file change history and see how this code changed and what happened in each version and who made that change.

Don’t worry, that’s not too scary that you think! We will use that together step by step. For now, think about those problems and read a little about source controls systems.

I will write a complete guide on use source control for our needs in group. But you don’t have to wait for me, you can read practical tutorials, and do some tests. It’s better to do so.

3- Online compilers

We are novice programmers and our first programs are small and has no very special parts, just simple input-process-output. We will write some codes to solve our main book's exercises. How do you prefer to test other's sample code? Do you download the code, create a new project and compile the code? I prefer not to do so. There are several online compilers that we can copy-paste other's code and get the results. This can help us to easily test simple codes and save lots of time.

I tried a few of them and I found the following services. Each has its features. I prefer first.

http://code.hackerearth.com
Supports:
C++11, custom input (stdin), Code video (a video of your coding!), Sharing, Statistics

http://www.compileonline.com/compile_cpp11_online.php
Supports:
C++11, custom input (stdin), Command line arguments, multi file

Does not support sharing.

http://ideone.com
Supports:
C++11, custom input (stdin), Sharing

Conclusion

You introduced with common issues and their solutions. As we progress we will become better in related issues and we will gain more ability to use necessary tools.

Please try to use and get familiar with source control because of its value in our team and don’t limit yourself to other member’s notes and guides.

To have a good progress together, please do your best with those issues.

Make the world a better place…