Writing program code course work examples

Business, Company

\n[toc title="Table of Contents"]\n

\n \t

1. Beautiful code and design \n \t
2. Open source software \n \t
3. Widgets \n \t
4. Grassroots computing \n \t
5. References \n

\n[/toc]\n \n

Beautiful code and design

It is important to have a beautiful code apart from good code alone. This is important because it serves as a guide from the programmers and developers who will work on the program in the future. Even though most codes serve the right purpose, there is needed to be able to communicate the program intention to the future users. Beautiful code should be self-describing with comments that are meant to show what the code is supposed to do. It will be easier to improve the performance of the program in the times to come. Good code without comments is useless as they may be replaced in future if they are not understood then.

Open source software

What open source is: Open source software defines a new category of software for which source code and other reserved rights are provided to the user under a public domain software license. Open source languages include Linux, UNIX, Java, and MySQL. Their intent is so that the software development is strengthened by contributions from different developers. The

open source languages are not intended to eradicate proprietary software as such but to help other developers to have a look at the source code of others. The advantages of open source software include eradication of license for the software that are obtained in open source development, availability of strong software because they have been developed by many professionals, the eradication of flaws in a simple way; with open source software, it is fast to resolve a flaw as the problems are tackled by many. The disadvantages of open source are the fact that there is no full ownership of software and the software which an organization has acquired could be found to be used somewhere else. Another disadvantage is that they are found to be difficult to use because they are normally coded hard.

API

An API, short for Application Programming Interface, is the interface that is developed by an application so that it will be able to communicate with other applications. APIs enable applications which are not the same to communicate and probably share information.

One example of an API is that of Google. Google is one company that I would like to work for. I have admired the way they program their applications. Google has developed an API that is used with their Google maps application. With this API, various programming languages are now able to get the application and integrate it in their applications. For this to be possible, it is a requirement that an API be developed to enable applications to communicate with the application.

The limitations that are associated with API are that if there is a breakdown of communication, the link between the two applications has been savored. APIs have no way of ensuring communication.

Another limitation is that there is no standard API that can be followed. This therefore means that each company comes up with their own API which follows no standard process or rules. It brings the issue of incompatibility. API is as useful as OOP. It is because they provide a basis for communication. Without APIs different applications cannot communicate.

Widgets

Widgets are applications that are developed to add up to a major application. They can be referred to as modules. There are many things and functionalities that programmers cannot add to applications that they develop. Applications like Facebook had been lacking in some functionality. This is now possible with widgets which are programmed into them by independent programmers.

There are advantages that are associated with widgets. One advantage is that they help make applications to be up to date as possible. They make applications more useful. One disadvantage is that they might overcrowd the code and make the original code fail.

Grassroots computing

This is the approach to applications development where users who non-IT professionals take initiative to develop a solution to a business problem without waiting to be developed the IT-traditional way. In organizations, software development is developed online and tested online. It is no longer

the IT professionals who work on the software. Rather, this process is undertaken from far people who have seen the problem.

With grassroots computing, there is no longer the process of system analysis because anytime a problem is encountered, a solution is developed by the people who experienced the problem.

References

Cherbakov, L., Bravery, A., Goodman, B. D., Pandya, A., & Baggett. (2007). Changing the corporate IT development model: Tapping the power of grassroots computing. IBM Systems Journal , 46 (4), 1-20.

Ellis, B. (2007). Widgets. New York: Cengage Learning.

Wirfs-Brock, R. J. (2007). Does beautiful code imply beautiful design? Design , 1 (1), 1-5.