Kazuru Takase git@git

Prof. Keith William

IS 217 002

2/28/13

Exam 1

1. Design pattern is a rewritable solution to a problem in any programs. It is a template to answer problems. In object-oriented programs, like Java script, design pattern illustrates the interaction between classes and objects. Design patterns are reusable, and it is effective in problem solving.
2. It gives the people the same vocabulary. Because everyone codes differently, it is difficult to read and explain its jobs. Using the same term and names can make communication easier, and get your thoughts across faster. It is also organized in a format so it if easy to follow along.
3. The main purpose of a design pattern is to solve common problem. In a design pattern, certain methods will always be used. For example, the Observer will always have a method to add and remove objects.
4. I think design pattern is a guideline, not a law. Because it is a language, it is not strict. In programming, it doesn’t matter how long a code is. As long as the code work, no matter the variation, it will be fine. However, the longer the code is, the more vulnerable it is to error. The design pattern is a guideline to keep the code organized, show other programmers the works done.
5. I personally don’t think I became a better programmer. It could be because I didn’t take the time to fully understand the concept and how to properly write it. However, I can imagine how beneficial the design pattern will be in the future. I will be able to write effective codes that are organized and easy to read. It will show how fluent I am with the programming language. I can also appeal to the people who are hiring by being knowledgeable about the template.
6. Factory pattern is a creational pattern. It gives the writer the ability to create multiple instances. There will have implementation and sub classes in it.
7. Observer is a behavioral pattern. The Observer watches over the code, and will respond when it detects changes in the code. The Observer needs to contact the object to get information of their interest. The relation between the publisher and subscribe is also dependant in this design pattern.
8. Decorator is a structural pattern. It gives the programmer the benefit of reusing the codes. Using it can give you to add additional characteristics or feature to an object. It does not use subclass.
9. Singleton is a pattern to prevent instances of an object. It is a creational pattern. It can only make one class. The initialization can be delayed, because it might not pass through it during runtime. The Singleton can also be considered as an object literal.
10. Constructor is a creational pattern. Once there is available memory, the Constructor can make a special type of object. It is used to prepare an object to be used and to take in the argument in which the constructor can use.
11. Prototype is a creational pattern. Its main job is to create duplicate of the original class. Prototype can also improve performance, because there are created by using reference.
12. Pub Sub a behavioral pattern. Like the Observer, it watches over the runtime and makes changes. However, unlike the former, the information will be broadcast by the publisher. That way, multiple subscribers can get the data of interest. It is mostly used in forums, since contact between the subscriber and publisher is important. Independent
13. Flyweight is a structural pattern. The purpose of a flyweight is for to minimize the uses of data. The data is shred in two ways: data-layer and DOM layer. It is shared with similar objects in the application
14. Singliton:

Var Singleton() {

Singleton.getInstance()

}

1. Factory:
2. Prototype:
3. Constructor:
4. Decorator:

19. The difference between a classical inheritance language and prototypical inheritance language is that classical inheritance is strict with objects and classes. This means that objects can only share data within the same class. Prototypical inheritance does not make a distinction between objects and classes. It can also be reused.

Inheritance: Prototype

Classical: Singleton

20. Two design patterns that can work well are constructor and decorator.

You can use the constructor to create a base, such as a table. You can use the decorator to make various types. This can be ranged from the material, color, additional accessory (tablecloth, vase) or height of the legs.