Class Design Guidelines – Instance vs. Static

An instance in java is a method that require objects of their own class to be created prior to it being called. Static methods do not need their own objects to be called. The only reference they have is their own class names or the references to the objects of that class. Statics cannot be over written but can be overloaded. What that means is within those classes there can be multiple methods with the same name. Similar to constructor overloading which is when a class has more than one constructor with different argument lists. Static methods are made with the purpose to be shared among all objects created from that same class. Within instance methods each individual object that was created in the class has its own copy of the method of that class.

When trying to determine whether a variable should be static you need to ask the following questions. Does the variable describe a specific object? If no, and if it describes all the objects in the class, make it a static variable. If it doesn’t describe all the object, make it a local variable and if necessary you can pass around as a parameter. However if it does describe a specific object, then you need to ask if it would make sense to have more than one of these objects. If yes, then make it an instance variable. If no, then you have the option to make it static, but making it an instance variable wouldn’t hurt.

As for methods, if you’re trying to determine whether it should be static you should begin by asking if it uses any fields of a specific object. If it does you can go ahead and make it an instance method. If not, and it doesn’t use any instance methods inside, then you can make it a static method. Otherwise, you can make it an instance method.

Here is an example of a class involving static methods I found on: <https://www.cis.upenn.edu/~matuszek/cit591-2006/Pages/static-vs-instance.html>

class SomeOtherClass {

void aStaticMethod() {

SomeClass.myID = 5; // illegal

SomeClass.nextID = 5;

SomeClass.someStaticMethod(5);

SomeClass.someInstanceMethod(5); //illegal

SomeClass thing = new SomeClass();

thing.myID = 5;

thing.nextID = 5;

thing.someStaticMethod(5);

thing.someInstanceMethod(5);

}

void anInstanceMethod() {

// same as in a static method

}

}