JAVA

**Create and destroy objects**

1. consider static factory methods instead of constructors
2. consider builder when faced with many constructors params
3. enforce the singleton property with an enum type or (if context is applicable) Spring singleton bean
4. enforce non instantiable with a private constructor
5. prefer DI to hardwriting resources
6. avoid creating unnecessary objects
7. eliminate obsolete objects references
8. avoid finalizers and cleaners *Sometimes they are necessary, but you should be aware of their pitfalls implement them right.*
9. pre-try-with -resources to try-finally

**Methods common to all objects:**

1. obey the general contract when overriding equals
2. always override hashcode when you override equals
3. always override toString *Of course, don't do it blindly. Consider, if the method would be called thousands of times per second, how it will impact performance and if so, is the toString() essential for that particular type then?*
4. override clone judiciously
5. consider implementing Comparable

**Classes and interfaces**

1. minimize the accessibility of classes and members
2. in public classes, use accessors method, not public fields
3. minimized mutability
4. favour composition over inheritance
5. design and document for inheritance or else prohibit it
6. prefer interface to abstract classes
7. design interface for posterity
8. use interface only to define types
9. prefer class hierarchies to tagged classes
10. favour static member classes over non-static
11. limit source files to a single top-level class

**Generics**

1. 26: don't use raw types
2. 27: eliminate unchecked warnings
3. 28 : prefer list to arrays
4. 29 : favour generic types
5. 30 : favour generics method
6. 31: use bounded wildcards to increase API flexibility
7. 32: combine generics and varanrgs judiciously
8. 33: consider type-safe heterogeneous containers

--------------------------------------------------------

* all non-static imports in a single block
* code not longer than 100 characters
* whitespaces spaces variable declaration one var per declaration
* comments each public and protected class and have . at the end /\*\* This is okey. / /  
  This is  
  okey. \*\*/  
  /\*\*  
  This is  
  okey.
* @param @return @throws @deprecated \*\*/
* todo(username) : comment or todo(jira number) : comment
* modificators public protected private abstract default final transient volatile synchronized native strictfp
* Naming no prefixes kName, m\_Name, name\_\_ Use class UpperCamelCase use method lowerCamelCase constant UPPER\_SNAKE\_CASE param lowerCamelCase local var lowerCamelCase
* More about naming convention:
* Naming convention about fields, kafka topics etc please check this link
* use @overide
* caught most specific Exception
* static method run on Class Foo aFoo = ... Foo.staticMethod(); //good aFoo.staticMethod(); //not good
* Try to make everything immutable as posible

--------------------------------------------------------

Use findbugs, pmd and checkstyle for code automatic code review