INTERFACE SEGREGATION PRINCIPLE

ISP

- Here, interface means the public methods of a class. (In Java, these are often specified by defining an interface, which other classes then implement.)
- Context: a class that provides a service for other "client" programmers usually requires that the clients write code that has a particular set of features. The service provider says "your code needs to have this interface".
- No client should be forced to implement irrelevant methods of an interface. Better to have lots of small, specific interfaces than fewer larger ones: easier to extend and modify the design.
- (Uh oh: "The interface keyword is harmful." [Uncle Bob, 'Interface' Considered Harmful] — we encourage you to read this and discuss with others. Does the fact that Java supports "default methods" for interfaces change anything?)

INTERFACE SEGREGATION PRINCIPLE

“Keep interfaces small so that users don’t end up depending on things they don’t need.

We still work with compiled languages. We still depend upon modification dates to determine which modules should be recompiled and redeployed. So long as this is true we will have to face the problem that when module A depends on module B at compile time, but not at run time, then changes to module B will force recompilation and redeployment of module A.” [Uncle Bob, SOLID Relevance]