Designing classes in general

- Highly cohesive
  - Each class does one thing
  - Interface is minimally complete, avoid kitchen sink
    - What if client/user might want to hammer with an awl?

- Loose coupling (and minimize coupling)
  - Classes depend on each other minimally
  - Changes in one don’t engender changes in another
  - Subclasses are tightly coupled, aggregates are not
    - Prefer Has-a to Is-a

- Test classes independently
  - Unit testing means just that, and every class should have a unit test suite

More heuristics (some from Riel)

- Users depend on a class’s interface, but a class shouldn’t depend on its users

- Be suspicious of “God”-classes, e.g., Driver, Manager, System
  - Watch out for classes supporting method subsets

- Beware of classes with lots of get/set methods

- Support Model/View distinction
  - The model shouldn’t depend on the view, but should support multiple views

- If a class contains an object it should directly use the object by sending it messages