Complete OO Support

- gen-class, proxy, reify, definterface, etc. make creation, design, and instantiation of Java classes easy
- defrecords allow the creation of value classes
- Clojure is more polymorphic than Java
  - interfaces work as Java
  - protocols support early- or late-binding single dispatch
  - multimethods support arbitrary dispatch
- Calling Java classes and APIs in Clojure is very simple
- In practice, you usually just code as data
- Calling Clojure from Java is straightforward
  - Caveat: Special care must be taken to prevent type erasure of generic types

Separated Concerns

- Clojure's design separates fundamental concerns:
  - Value: Modeled as data
  - Transition: Pure functions
  - State: Atoms, agents, refs (STM)
  - Shape: spec
- Clojure applications have well-defined boundaries between the model and view layers. It is easy to create a domain model as pure Clojure and write desktop and web front ends.
- Functional languages with strong Object/Type systems complect concerns
  - E.g. Scala