## Design patterns

- Evolved from practices in OO design, but concepts are more widely applicable
- General idea: there are some common styles of problem or system in software design, so we can establish a common “pattern” for a solution
- Provides a common starting point for explaining (some) problems and solutions
- Many projects and tools support standardized frameworks or templates for patterns they encounter frequently

## Common use

- Frequently it is possible to express a particular problem or solution in different ways, possibly start by suggesting a couple of design patterns as starting point
- Many problems require a customization or combination of patterns for different parts of the problem
- Be wary of trying too hard to shoe-horn a problem or solution into any particular pattern

## A few broad groups of patterns

- Creational patterns: typically deal with creating an object or instantiating a class (see Singleton)
- Structural patterns: typically deal with combining or organizing other classes and objects into more complex structures that provide more functionality (see Adapter)
- Behavioural patterns: typically deal with the way objects communicate or interact (see Command, Iterator)

## A few pattern examples

- Singleton: patterns where there can only be one instance of a specific class
- Adapter: converts interface of one class to be compatible with that of another
- Iterator: supports traversals of all the elements of a data structure