TP15: Design patterns

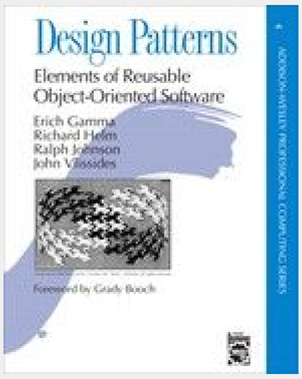
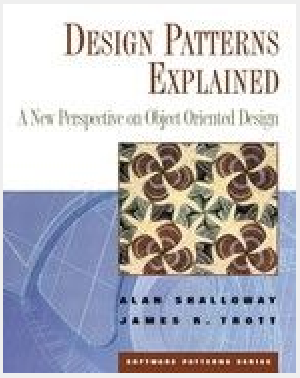
In software engineering, a **design pattern** is a general, reusable solution to a commonly occurring software problem in software design. A design pattern isn't a finished design that can be transformed directly into code. It is a description (or template) for how to solve a problem that can be used in different situations.
Design **patterns can speed up the development** process by providing proven approaches. In addition, **patterns allow developers to share common vocabulary** using well known, well understood names for software interactions.

Learning objectives for this class

- Describe the elements of a software pattern.
- Explain the 3 categories of patterns (in GoF) and give examples for each.
- Explain the role of patterns in class and method design
- For a given problem, suggest the appropriate pattern (considering the GoF patterns discussed in class)

Resources

- Slides deck, by J. Aldrich (@CMU web)
- Easy to read articles on software design patterns at Sourcemaking.com
- Select books on "patterns"

 Design Patterns Elements of Reusable Object-Oriented Software Erich Gamma Richard Helm Ralph Johnson John Vlissides Foreword by Grady Booch ADDISON-WESLEY PROFESSIONAL COMPUTING SERIES The original GoF (available from O'Reilly)	 DESIGN PATTERNS EXPLAINED A New Perspective on Object Oriented Design ALAN SHALLOWAY JAMES R. TROTT SOFTWARE PATTERNS SERIES (available from O'Reilly)

Selected patters

Selected patterns (discussed in class):

- Façade (17ss),
- Adaptor (19ss),
- Strategy (23ss),
- Abstract Factory (28ss),
- Singleton (30ss) and Object Pool
- Observer (32ss).

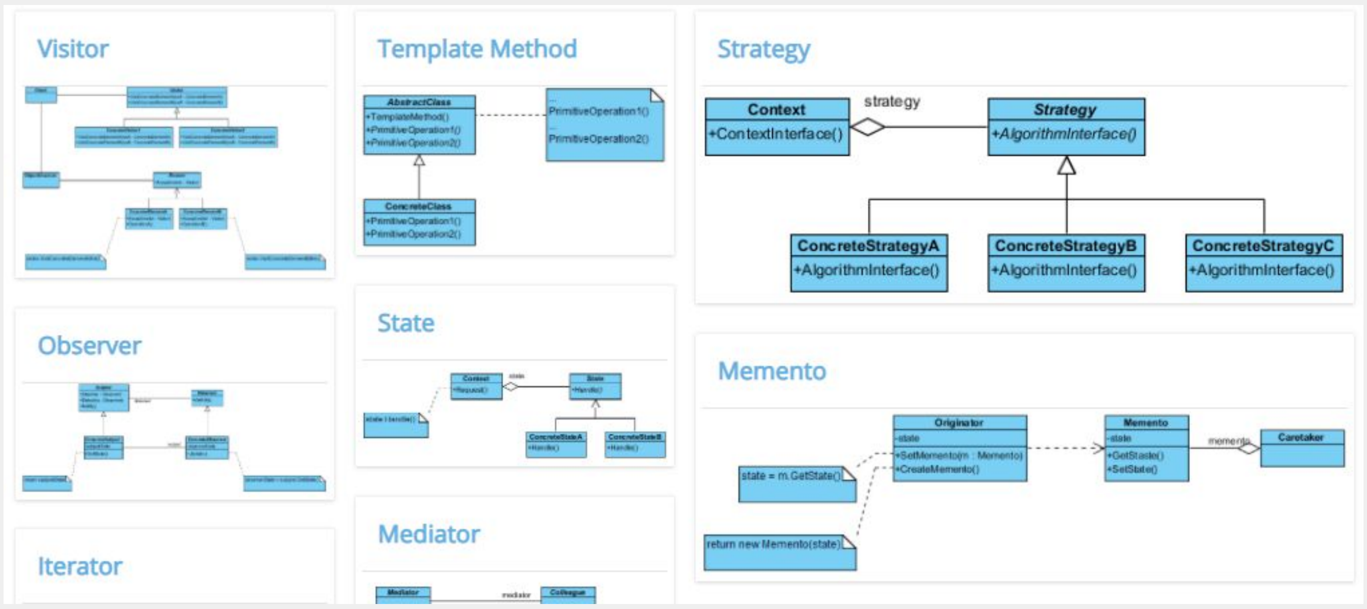
Supporting examples (used in class):

- Java EntityManager;

    - Using the EntityManager to save a House entity *vs* using a HouseFacade with a subset of the methods.
- ArrayAdapter in android
    - Make heterogeneous collections compatible with “list” views
- EntityManagerFactory in Java EE
    - Abstract the “family of classes” for the specific application server and database
- Observing changes in the text box in Android with TextChangedListener
    - “Listeners” act as observers

Design patterns and Visual Paradigm

Visual Paradigm



- Quick-reference (and importable models)
- VP Tutorials on Patterns (create your own patterns with UML models), e.g., Observer pattern