Advanced Software Development

Daily Topics

Summary of Daily Topics

Contents

Advanced Software Development

Daily Topics

Day 1: Introduction and course Overview

- Overview of Advanced Software Systems
    - Value of Abstraction
    - Elements of Software Development
- Definition of Advanced Software Development & topics
    - Methods of ASD
    - Refactoring and SE^{++}
- Course Structure & Administration
    - Daily Schedule
    - Quizzes
    - Labs:
        - *Timelog*, hand-in times
    - Grading, Exams
- Levels of architecture
    - PLangs, idiom, patterns, Frameworks
- Software Patterns & Frameworks; overview
    - Patterns, and design
    - GOF & pattern formalizations
    - Frameworks
- Software Metrics ⇨ Laws ⇨ Principles
- Patterns are both language { independent, relative } (?!)
- Reading
    - Homework

Advanced Software Development

Daily Topics

Day 2: Programming Language Features & Design Patterns

- Quiz & Review
- Refactoring
- Programming Language Abstractions
    - OO; model, terminology, concepts
    - Types, Sub-type Abstraction
    - Polymorphism
- Principles ⇨ methods ⇨ patterns
    - *Variation Oriented Design* (VOD)
        - *change :: non-change*
    - decoupling, encapsulation, abstraction
- Patterns Catalog
    - GOF ⇨ standard format
    - Pattern catalog survey…
- Resources for Patterns
- Reading (GOF)
    - Homework

Advanced Software Development

Daily Topics

Day 3: Pattern Mining & the Observer Pattern

- Quiz & Review
- Pattern Mining
    - Swing
- Event driven Programming
    - event architecture
        - functor pattern
    - Java event model
        - *ActionListeners*
        - Button examples
        - *= observer pattern*
    - Java idioms for listeners → OO
- Observer Pattern ⇨ GOF
    - Standard pattern template
- Patterns in Swing:
    - MVC
    - command
    - observer
    - Composite
    - Strategy, Bridge, …
- Assignment:
    - Labs:
        - *Timelog*, hand-in times
        - *Readme*
    - Observer Lab

Advanced Software Development

Daily Topics

Day 4: Façade, Strategy & Template Patterns

- Quiz & Review
- Observer
  - examples & issues
  - Java Implementation ∈ SI ⇨ MI
  - Types, coupling (push, pull)…
  - generics
- Façade pattern
- Strategy pattern
  - uses *functor*
- Template pattern
  - ≈ refactored Strategy
- Lab:
  - Review: Observer Lab
  - Strategy Lab

Advanced Software Development

Daily Topics

Day 5: Collections & Iteration

- Quiz & Review
- Collections
  - Java collections ⇨ generics
- Iterators
  - issues: Robust, modular, stateful, …
  - polymorphic iterators
- Iterator pattern: GOF
  - general design ⇨ OO
- Iterator *factory* (pattern)
- Variations: (from *@element ⇨ wholeness*…)
  - uses *functor* pattern
  - (Iterator + functor) ⇨ …
    - internal iterator
    - selective iterator
  - ≈ Functional Programming
    - higher level of abstraction
    - Declarative, less *over constraint* (sequential)
  - Other fanciness…
- Soon, Java *closures* will make iterators more powerful, and popular!
- Lab:
  - [Simple] Iterator Lab

Advanced Software Development

Daily Topics

Day 6: More Iterators…

- Quiz & Review
- Iterators…
  - Selective iterator
  - Returning results of *doAll(); → accumulator pattern*
    - Alternative, *generic methods*
  - make ***Iterable*** → *views*
- Standard FP functions
  - Filter, map, reduce
  - Reduce ⇨ foldr, foldl, …
- Iterator Exercise
  - Add internal iterator to *Vector* ⇨ *MyVector*
  - *accumulator* pattern
    (or, *generic methods*)
- Lab:
  - Iterator Lab'

Advanced Software Development

Daily Topics

Day 7a: Functional Iterators

- Iterator Exercise Review
- Iterators…
  - make ***Iterable*** → *views*
  - add compound predicates
- FP style → **SE**$^{++}$
  - Declarative; more abstract ⇨ simpler, shorter, clearer, …
  - *Loopless* programming
- Examples…
  - Guava iterators
- Generators

Advanced Software Development

Daily Topics

Day 7a: Generics…

- Type abstractions
- Easy to use,
    - harder to define
    - This is a good tradeoff
- Generics are not classes
    - = meta-classes
    - higher level abstractions
    - instantiate ⇨ class
- Inheritance with generics
    - class extends class
    - generic extends generic
- generics are not covariant
    - but wildcards help…
    - ≈ Type variances
- Many other topics
    - type constraints
    - …

Advanced Software Development

Daily Topics

Day 7b: Composite Pattern

- Quiz & Review
- General representation of recursive data-structures
  - list, tree
- ⇨ Composite pattern
  - parts ≈ whole
  - based on SR loop ⇨ ∞ composition
- Design options
  - uniform, *safe*
- Recursive iteration?
  - internal 2
  - external ?? ☹
- Lab:
  - Composite Lab

Advanced Software Development

Daily Topics

Day 7c: Visitor Pattern

- Quiz & Review
- Goal:
  - separate Data Structure(s) & Operations
  - ≠ OO!
  - Intent: {Δ Data hard, Δ^+ Ops easy}
- Method
  - *accept(Visitor)* method @D_i
  - Overloaded *visit(D_i)* @ V_j
  - ≈ 2-D Polymorphism
  - ⇨ multi-methods
    - Some PLang directly implement
    - Thus, no need for this pattern!
- See: GOF
- [Handout]
- Lab:
  - Composite Lab

Advanced Software Development

Daily Topics

Day 8: Command & Mediator Patterns

- Quiz & Review
- Another Functional Iterator Example…
  - *Iterator transformers*
  - closures
- Mediator Pattern
- Command Pattern
  - Encapsulate request as object ≈ functor
  - Nice usage with *ActionListeners*
  - also adds additional Capabilities

    { undo, redo, …}

    history, logging, transactions, …
- Lab:
  - Command Lab

Advanced Software Development

Daily Topics

Day 9: More: Command Pattern

- Example code…
  - GUI
  - Switch …
  - Scheduler
  - ⇨ different *CmdManager* models /methods
    - Event driven (swing)
    - Message drives (switch)
    - Timer driven (scheduler)
    - …
- *CommandButtons*
  - Combine functions: GUI & Command
    (Only works with stateless commands)
- Swing undo/redo
- Lab: Φ_2
  - Review ϕ_1
  - Add Command Manager
  - Stateful commands { *undo*, *redo* }
  - *meta*-commands ?
  - GUI *mediator*

Advanced Software Development

Daily Topics

Day 10: State Pattern

- Quiz & Review
- Review command lab
- *State Pattern*
  - basic idea
  - ⇨ OO pattern
  - @State; new binding of: $msg_n \rightarrow method_m$
    Thus, new behavior(s)!
  - Who controls state transitions?
  - Context ⇨ delegation
  - Structure ≈ Strategy (!)
- Lab:
  - State pattern lab

Advanced Software Development

Daily Topics

Day 11: Proxy, Decorator, & COR Pattern

- Quiz & Review
- Review state lab
- *Proxy Pattern*
    - important concept
    - uses, impacts
    - ⇨ OO pattern
    - See: GOF
- Proxy improvements, implementations
    - Functor, generics
    - Dynamic proxy (later…)
- Examples
- Review Session for Midterm Exam
- Next;
    - *dynamic proxy*

Advanced Software Development

Daily Topics

Day 12: Dynamic Proxy

- Exam Review
- Review *Proxy Pattern*
- Proxy^{++}
  - increased generality through abstractions
  - derived proxy classes
    - override wrapper methods
    - use template pattern,
      create *pre()* & *post()* methods
      - ⇨ method abstraction
  - functor proxy
    - ⇨ functional abstraction
  - generic functor proxy
  - layered Proxies
  - proxy factory
- Results:
  - dynamic, reuse, abstraction
- Exercise:
  - write simple logging proxy
- State lab using proxy
  - and selective *DataSetter* interface
- Decorator pattern
  - ≈ proxy (in structure)
  - different(?) intents
- *dynamic proxy*
  - motivation
  - implementation
  - examples

Advanced Software Development

Daily Topics

Day 12b: COR, Bridge Patterns

- Quiz & Review
- Chain of Responsibility pattern
- Bridge Pattern
    - separate description and implementation
        - ≈ *Abstraction and implementation* [GOF]
- Next;
    - factory pattern

Advanced Software Development

Daily Topics

Day 13: Factory Patterns

- Quiz & Review
- *Factory pattern* family
    - Simple Factory
    - Factory Method
    - Abstract Factory
    - Generic Factories?
    - Reflective Factory
- No *magic*, just OO ☺
- Friends of Factory;
    - Singleton Pattern
    - Builder
    - Prototype

Advanced Software Development

Daily Topics

Day 14: Frameworks

- Exam Review
- Basic definitions and principles
    - Higher level abstraction and reuse
    -
- Classifications
    - Horizontal, Vertical
    - Granularity
- Usage
- Inheritance & composition
- Homework:
    - two designs…

Advanced Software Development

Daily Topics

Day 15: Framework Development

- Review summary of reading assignment
- Refactoring to frameworks
    - Based on existing developed applications
    - Iterative development
- Functional abstraction
    - Environmental continuity
    - Inheritance (overriding)
        - *Template Method* Pattern
    - Composition (DI)
    - Declarative DI
        - XML, Properties, …
- Value abstraction ↔ Functional abstraction
    - Functions → values
    - Functions *as* values
        - Functors, *strategy pattern*
- Setup & IOC
    - Dependency injection
    - Outcalls (IOC), upCalls (Template)
    - Out → In-calls; transfer of authority
- Factories & Frameworks
- [Viewgraphs]
    - Black-Box, White-Box, … Gray
- Current usage of Frameworks
    - Infrastructure, Web, …
- Homework:
    - two designs ⇨ Framework…