

OOSE200 Report

Company Training Simulation

Connor Beardsmore - 15504319

The Sacred Elements of the Faith

the holy
origins

the holy
structures

107 FM Factory Method	the holy behaviors					139 A Adapter		
117 PT Prototype	127 S Singleton					223 CR Chain of Responsibility	163 CP Composite	175 D Decorator
87 AF Abstract Factory	325 TM Template Method	233 CD Command	273 MD Mediator	293 O Observer	243 IN Interpreter	207 PX Proxy	185 FA Façade	
97 BU Builder	315 SR Strategy	283 MM Memento	305 ST State	257 IT Iterator	331 V Vistor	195 FL Flyweight	151 BR Bridge	

“Company Training Simulation”

Polymorphism

- Property - kept in map, polymorphically call calcProfit() via strategy
- Events + Plan - both use strategy so can call run() on parent class
- WageObserver list allows ANY class to become an observer if it implements

Design Pattern Implemented

- Factory - for Events and Plans
- Dependency Injection - import objects, new in main, no statics
- MVC - overall layout, multiple controllers
- Observer - wage observer for global updates
- Composite - company owns other companies, tree structure for profit
- Template Method - file reading, common code for opening/closing files
- Strategy - plans.run(), events.run() and property.calcProfit()
- Iterator - for loops are sick, even if super inefficient

Testability

- Test cases!! sample outputs to clear up order ambiguity
- Factory + Dependency Injection allow for easy mocking of objects, low coupling
- Mad toString() and debug output methods
- clear and consise exception handling
- tested on heaps of invalid file types for all 3 input files

Alternative Design Choices

- design is mad extensible and shit due to the patterns used
- Iterators instead of for loops
- Controllers are easy to switch in and out, could have used one bunta controller
- could have used scanner instead of BufferedReader