Activity 18-2: The Factory and Singleton Patterns

Why?

The Factory and Singleton patterns are among the most widely used and important design patterns.

Learning Objectives

- Understand the Factory and Singleton patterns and how to use them

Success Criteria

- Be able to explain the Factory Method and Abstract Factory patterns and the differences between them
- Be able to explain the Singleton pattern and how it is implemented in object-oriented systems
- Be able to use the Factory and Singleton patterns appropriately and correctly

Resources

ISED sections 18.2 and 18.3

Vocabulary

Factory Method pattern, Abstract Factory pattern, factory class, Singleton pattern, singleton class

Plan

1. Review *ISED* sections 18.2 and 18.3 individually.
2. Answer the Key Questions individually, and then evaluate the answers as a team.
3. Do the Exercises as a team, and check your answers with the instructor.
4. Do the Problems and Assessment as a team.
5. Turn in the Problems and Assessment as a team deliverable.

Key Questions

1. Why are interfaces important for the two Factory patterns?
2. How does the Factory Method pattern differ from the Abstract Factory pattern?
3. Why does the `instance()` operation in the Singleton pattern need to be static?

Exercise

1. What would a Factory Method pattern look like without any interfaces? Would it still be the Factory Method pattern?
2. Does the Singleton pattern have a factory method?

Problems (Deliverable)

1. Please do *ISED* exercises 18.8 and 18.13.
2. A program that interprets two language, Alpha and Beta, builds parse trees with two kinds of nodes: non-terminal nodes (which are large), and terminal nodes (which are small). The terminal and non-terminal nodes for Alpha and Beta are slightly different, but the interpreter runs code from only one language at a time, which it knows when its starts up, so the right kinds of nodes can be allocated. For efficiency, the program has a node manager that allocates and deallocates nodes, which are objects, from a pool of nodes that allows it to reuse node objects rather than creating and destroying them repeatedly. The node manager must contain operations for allocating terminal and non-terminal nodes and for deallocating nodes (returning them to the node pool). Space will be wasted and page faults may slow down the interpreter if there is more than one node pool, because then node objects will not be reused efficiently. Use your knowledge of the Factory and Singleton patterns to design the node manager.

Assessment (Deliverable)

1. Did this activity help you achieve the learning objectives?
2. How could the instructor improve this activity?