Template Method Pattern

Design Patterns

Motivating Example

Template Method Pattern

- Model a process or algorithm of several steps
- Allow variation of the details of each step, while enforcing the structure and order of the steps themselves
- For instance, a game engine might dictate certain steps making up a variety of games:
  - SetUpGame()
  - TakeTurn(Player p)
  - IsGameOver()
  - DisplayWinner()
- Given this structure, specific games like Tic-Tac-Toe, Chess, Monopoly, etc could be implemented as subclasses

Intent

Template Method Pattern

- Encapsulate and enforce the workflow or process that is not variable
- Allow subclasses to alter specific behavior via concrete implementation
- Redefine one or more steps of an algorithm without altering its structure

Learn more about the **Don't Repeat Yourself** Principle in the **Principles of Object Oriented Design** course at Pluralsight On Demand

Applicability

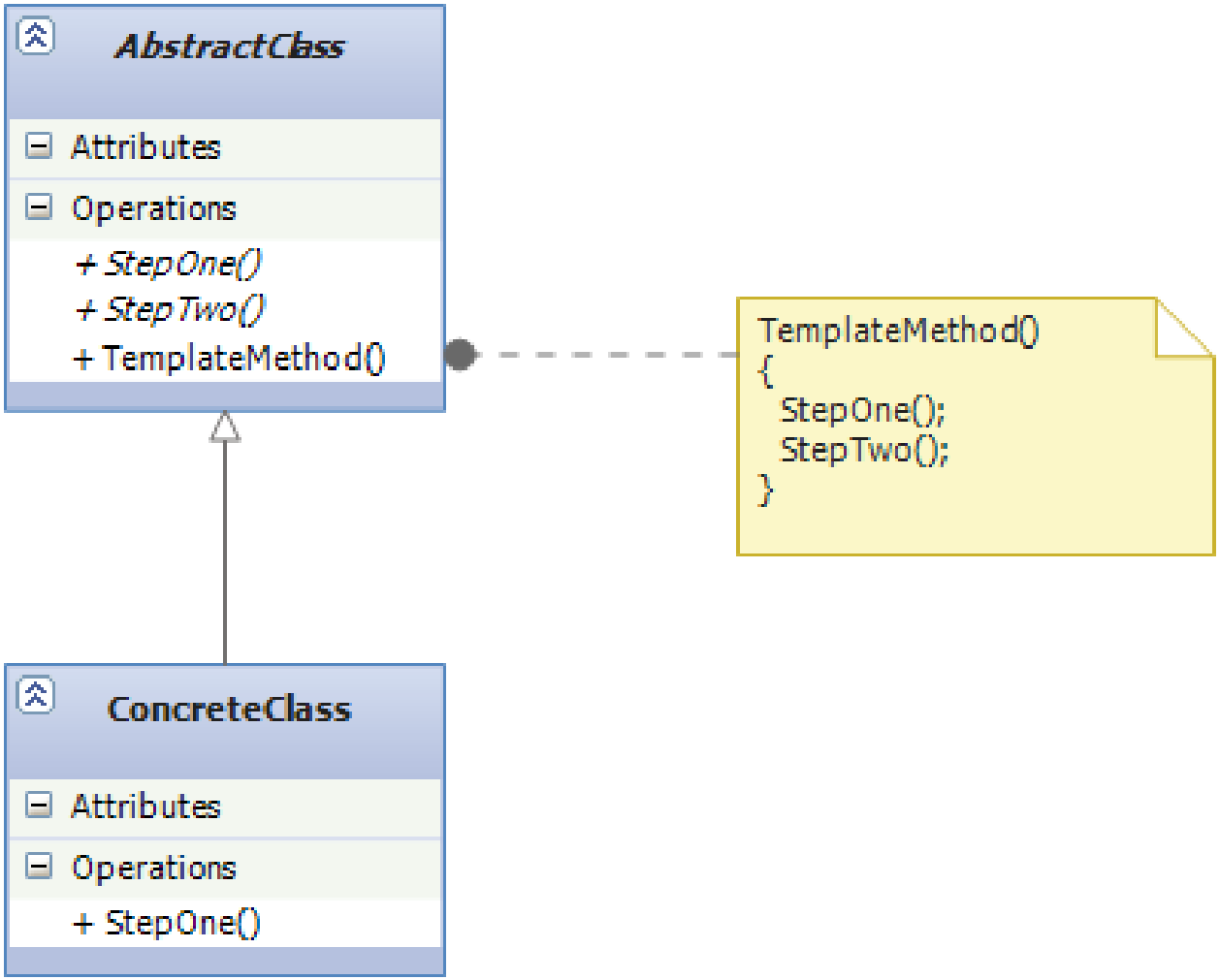
Template Method Pattern

Use the Template Method Pattern when:

- Two or more classes should follow the same common algorithm or workflow
- The workflow is invariant. Subclasses may redefine certain steps, but may not change the algorithm's structure
- Some workflow steps may be encapsulated in the base class with a default implementation, and only overridden if necessary, allowing code reuse

Structure

Template Method Pattern



How It Gets Used

Template Method Pattern

- **Clients Call Children of Base Implementation**
- **Child Types Customize Individual Step Behavior**
  - Might change all, many, or just one step
- **Effective way to achieve Open/Closed Principle**

Learn more about the **Open/Closed Principle** Principle in the **Principles of Object Oriented Design** course at Pluralsight On Demand

Consequences

Template Method Pattern

- **Algorithm steps must be known and relatively inflexible at the time the pattern is applied**
- **Relies on inheritance, rather than composition, which can be a limitation**
  - See the Strategy pattern for a composition-based solution
- **Single inheritance makes it difficult to merge two child algorithms into one**
  - See the Decorator pattern for a possible solution to this problem

Hooks

Template Method Pattern

- Hooks are methods declared in the abstract class that have no implementation

- Allow sub-classes to “hook into” the behavior of the algorithm at various points (or to ignore the hooks entirely)

The Hollywood Principle

Template Method Pattern

- "Don't call us, we'll call you."
- High level components should not depend on low-level components
- Base class with template method is high level component – clients should depend on this class
- The subclasses are the low-level implementation – they don't call anything themselves, and are only called by the high-level template method

http://www.flickr.com/photos/storm-crypt/2896456099

Implementation Example

Template Method Pattern

ASP.NET Web Forms

- Page Life Cycle

Order Processing

- Customizing steps in the order

Related Patterns

Template Method Pattern

- **Strategy**
  - Inject a complete algorithm implementation into another module
- **Decorator**
  - Compose an algorithm or behavior from several sub-parts
- **Factory Method**
  - Define a common interface for creating new instances of types, with many implementations

References

Template Method Pattern

- **Books**
  - Design Patterns, http://amzn.to/95q9ux
  - Design Patterns Explained, http://amzn.to/cr8Vxb
  - Design Patterns in C#, http://amzn.to/bqJgdU
  - Head First Design Patterns, http://amzn.to/aA4RS6
- **Online**
  - http://en.wikipedia.org/wiki/Template_method_pattern
- **Related Patterns**
  - Strategy
  - Decorator
  - Factory Method

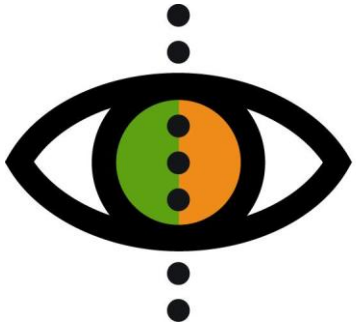
Summary

Template Method Pattern

- Template Method uses inheritance to define an algorithm in a superclass while delegating responsibility for detailed implementations to child classes
- Provides greater reuse but less flexibility than Strategy pattern
- Remember the Hollywood Principle:

  Don't Call Us, We'll Call You

For more in-depth online developer training visit



pluralsight

see what you can learn

on-demand content from authors you trust