Если вы видите что-то необычное, просто сообщите мне.

Шаблонный метод (Template Method)

Паттерн Template Method относится к поведенческим паттернам уровня класса.

Паттерн Template Method формирует структуру алгоритма и позволяет в производных классах реализовать, перекрыть или переопределить определенные шаги алгоритма, не изменяя структуру алгоритма в целом.

Проектировщик решает, какие шаги алгоритма являются неизменными, а какие изменяемыми. Абстрактный базовый класс реализует стандартные неизменяемые шаги алгоритма и может предоставлять реализацию по умолчанию для изменяемых шагов. Изменяемые шаги могут предоставляться клиентом компонента в конкретных производных классах.

Требуется для реализации:

1. Абстрактный класс AbstractClass, реализующий Template Method, который описывает порядок действий;
2. Класс ConcreteClass, реализующий изменяемые действия.

[!] В описании паттерна применяются общие понятия, такие как Класс, Объект, Абстрактный класс. Применимо к языку Go, это Пользовательский Тип, Значение этого Типа и Интерфейс. Также в языке Go за место общепринятого наследования используется агрегирование и встраивание.

Т.к. в Go нет понятия "Абстрактный Класс" и знакомого нам полиморфизма на наследовании, следует использовать встравивания общего для ConcreteClass типа с реализацией Template Method.

```
//template_method.go
// Package template_method is an example of the Template Method Pattern.
// In fact, this pattern is based on Abstract Class and Polymorphism.
// But there’s nothing like that in Go, so the composition will be applied.
package template_method

// QuotesInterface provides an interface for setting different quotes.
type QuotesInterface interface {
	Open() string
	Close() string
}

// Quotes implements a Template Method.
type Quotes struct {
	QuotesInterface
}

// Quotes is the Template Method.
func (q *Quotes) Quotes(str string) string {
	return q.Open() + str + q.Close()
}

// NewQuotes is the Quotes constructor.
func NewQuotes(qt QuotesInterface) *Quotes {
	return &Quotes{qt}
}

// FrenchQuotes implements wrapping the string in French quotes.
type FrenchQuotes struct {
}

// Open sets opening quotes.
func (q *FrenchQuotes) Open() string {
	return "«"
}

// Close sets closing quotes.
func (q *FrenchQuotes) Close() string {
	return "»"
}
```

```
// GermanQuotes implements wrapping the string in German quotes.
type GermanQuotes struct {
}

// Open sets opening quotes.
func (q *GermanQuotes) Open() string {
	return "„"
}

// Close sets closing quotes.
func (q *GermanQuotes) Close() string {
	return "“"
}
```

```
//template_method_test.go
package template_method

import (
	"testing"
)

func TestTemplateMethod(t *testing.T) {

	expect := "«Test String»"

	qt := NewQuotes(&FrenchQuotes{})

	result := qt.Quotes("Test String")

	if result != expect {
		t.Errorf("Expect result to equal %s, but %s.\n", expect, result)
	}
}
```

Revision #1

Created 2022-07-03 10:19:46 UTC by gasick

Updated 2022-07-03 10:20:57 UTC by gasick