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

Одиночка (Singleton)

Паттерн Singleton относится к порождающим паттернам уровня объекта. Паттерн контролирует создание единственного экземпляра некоторого класса и предоставляет доступ к нему. Другими словами, Singleton гарантирует, что у класса будет только один экземпляр и предоставляет к нему точку доступа, через фабричный метод.

Требуется для реализации:

1. Функция GetInstance, создающая экземпляр класса Singleton только один раз. Если до этого экземпляр уже был создан, то просто возвращает этот экземпляр.

[!] 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.

```
//singleton.go
// Package singleton is an example of the Singleton Pattern.
package singleton

import (
	"sync"
)

// Singleton implementation.
type Singleton struct {
}

var (
	instance *Singleton
	once     sync.Once
)
```

```
// GetInstance returns singleton
func GetInstance() *Singleton {
    once.Do(func() {
        instance = &Singleton{}
    })
    return instance
}
```

```
//singleton_test.go
package singleton

import (
    "testing"
)

func TestSingleton(t *testing.T) {

    instance1 := GetInstance()
    instance2 := GetInstance()

    if instance1 != instance2 {
        t.Error("Objects are not equal!\n")
    }
}
```

Revision #1
Created 2022-07-03 10:26:27 UTC by gasick
Updated 2022-07-03 10:27:12 UTC by gasick