目的・範囲:
Objective & Scope singleton

分類名:
Classification Name

著作者:
Author 津久井

実施日:
Enforcement day

バージョン:
Version

初版発行日:
Original Release

現行版発行日:
Current Release

キーワード:
Key Words

背景情報:
Background

◇目的
あるクラスのインスタンスが一つしかないことを保証したい場合があります。
注意深く設計すれば、唯一のインスタンスを使いまわすことは可能でしょうが、
このインスタンスが唯一であることを保障するものとはなりません。このような場合に、威力を発揮するのが Singleton パター

◇効果
Singleton パターンは、コンストラクタを private とすることで、
他クラスから新たにインスタンスが生成されないような構造とすることで、インスタンスの生成を制御します。

◇背景
小規模なプログラムなら、インスタンスを1つしか作らないという暗黙の規則を作り、
注意深くコーディングすれば対応することはできるでしょう。しかし、プログラムの規模が大きくなってくると、
そのような暗黙の規則だけではいずれ破たんしてしまいます。
もともと1つだけのインスタンスを扱うように作られたクラスが、いつの間にか複数のインスタンスで扱われていたりすると、
バグを引き起こす原因になります。
そういった場合で、パターンを使うということになります。

◇singletonパターンの実際のコードと考え方

```
class MyCount
    @@obj=nil
    def self.instance
        return @@obj if @@obj
        @@obj=new
    end

    def up
        @count+=1
    end
    private_class_method:new
    def initialize
        @count=0
    end
end
```

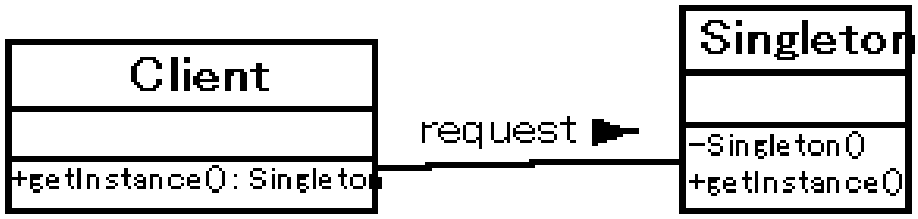
クライアント側では以下のようにして呼び出します。

```
a=MyCount.instance
b=MyCount.instance

puts a.up
puts b.up
```

◇問題点の改善
必要なインスタンスが絶対に1つだけと確信できるクラス以外では、シングルトンを使わない。

◇singletonパターンのまとめ



シングルトンクラス のコンストラクタを
private にすることで、別のクラスが シングルト
ンクラスのインスタンスを直接生成するのを 防いでいる。

◇注意
　理論的には独立しているはずのコード間に暗黙の依存関係を生んでしまう。

◇総括