**Projetos**

**Introdução**

Atualmente não se concebe um processo de desenvolvimento de software sério sem a utilização da orientação a objetos, pois esta permite agregar qualidades importantes aos sistemas desenvolvidos sob seus paradigmas, como a extensibilidade. Contudo, somente por estar utilizando-a, não é garantia de obter essas qualidades. Para criar as melhores soluções é preciso seguir um processo detalhado para obter uma análise dos requisitos, funcionais ou não funcionais, e desenvolver um projeto que os satisfaça e que possibilite submetê-los a teste para constatar eventuais falhas, se deseja que o projeto tenha uma arquitetura flexível para acomodar futuros problemas e requisitos sem a necessidade da realização do re-projeto.

Analisando o cotidiano do desenvolvimento de software é possível identificar que a procura por uma solução de um problema específico possui características idênticas, senão igual a encontrada em um projeto anteriormente desenvolvido, mas que devido à deficiência do processo, a solução e o problema não foram documentados e, às vezes, tão pouco compreendido em sua totalidade, impossibilitando o reaproveitamento das ideias e soluções. Desta forma, problemas idênticos que se repetem em outros contextos não são reconhecidos como tal, consumindo tempo e recursos em busca de soluções que em tese, já haviam sido encontradas.

**Tipo de projeto**

### **Singleton**

Garantir que um objeto terá apenas uma única instância, isto é, que uma classe irá gerar apenas um objeto e que este estará disponível de forma única para todo o escopo de uma aplicação.

Algumas aplicações têm a necessidade de controlar o número de instâncias criadas de algumas classes, seja pela necessidade da própria lógica ou por motivos de performance e economia de recursos.

Imagine o momento em que uma aplicação que exista simultaneamente num dispositivo móvel (Pocket PC, Celular, Palm) e num ambiente corporativo, necessite de um processo de sincronização entre as informações processadas no dispositivo móvel e na base corporativa. Ambas as aplicações deveriam se comunicar com um objeto que deveria ser único para processar este sincronismo, a fim de evitar a possibilidade de criar dados na base.

O uso do padrão Singleton está condicionado a:

* Quando for necessário manter num sistema, seja ele distribuído ou não, apenas uma instância de objeto e que o ponto de acesso para este seja bem conhecido (Ex. objeto responsável por um pool de impressão numa rede, gerenciador de janelas);
* Quando a única instância tiver de ser extensível através de subclasses, possibilitando aos clientes usarem uma instância estendida sem alterar o seu código (visões polimórficas).

Os desenvolvedores Delphi já utilizam o comportamento do padrão Singleton em suas aplicações, quando declaram variáveis globais na área de inicialização do projeto e depois reutilizam a instância dos objetos.TApplication, TCiipBoard são exemplos de objetos que não teriam sentido existir mais de uma instância na aplicação e por isso assumem o comportamento do padrão Singleton.