Ponteiros Inteligentes

Prof. Valdir Pedrinho de Tomin Junior, Eng.

Universidade Federal de Santa Catarina
Centro Tecnológico de Joinville
EMB5631 – Programação III

e-mail: valdir.pedrinho@ufsc.br

6 de novembro de 2018

Introdução

Ponteiros Inteligentes

- Ponteiro inteligente é uma classe de invólucro sobre um ponteiro com o operador como * e -> sobrecarregado;
- São ponteiros que não precisam ser excluídos explicitamente;
- A liberação de memória é feita de forma automática, quando o ponteiro sai do seu escopo;

Exemplo

Ponteiro *template*

Tipos de Ponteiros Inteligentes

Bibliotecas C ++ fornecem implementações de ponteiros inteligentes nos seguintes tipos:

- **std::auto_ptr**
- **std::unique_ptr**
- **std::shared_ptr**
- **std::weak_ptr**

Ponteiros Inteligentes

std::auto_ptr

- Este modelo de classe está obsoleto a partir do C ++ 11;
- **auto_ptr** é um ponteiro inteligente que gerencia um objeto obtido via comando **new** e exclui esse objeto quando o próprio **auto_ptr** é destruído;
- Um objeto quando descrito usando a classe **auto_ptr** armazena um ponteiro para um único objeto alocado que garante que quando ele sair do escopo, o objeto para o qual ele aponta deve ser automaticamente destruído;
- É baseado no modelo de propriedade exclusiva, ou seja, dois ponteiros do mesmo tipo não podem apontar para o mesmo recurso ao mesmo tempo.

Ponteiros Inteligentes

std::unique_ptr

- **std::unique_ptr** foi desenvolvido em C ++ 11 como um substituto para **std::auto_ptr**;
- O **std::unique_ptr** é um novo recurso com uma funcionalidade semelhante, mas com segurança aprimorada (sem atribuições falsas de cópia), recursos adicionais e suporte para *arrays*;
- É um contêiner para ponteiros brutos. Impede explicitamente a cópia de seu ponteiro contido como aconteceria com a atribuição normal, ou seja, permite exatamente um dono do ponteiro subjacente.

Quando usar o std::unique_ptr?

Use **std::unique_ptr** quando quiser ter propriedade única (exclusiva) do recurso. Apenas um **unique_ptr** pode apontar para um recurso.

Ponteiros Inteligentes

std::shared_ptr

- Um destaque **std::shared_ptr** é um contêiner para ponteiros brutos;
- Possui contagem por referência;
- Contagem de Referência: É uma técnica de armazenar o número de referências, ponteiros ou alças para um recurso, como um objeto, bloco de memória, espaço em disco ou outros recursos;
- Um objeto referenciado pelo ponteiro bruto contido não será destruído enquanto a contagem de referência seja maior que zero, ou seja, até que todas as cópias de **std::shared_ptr** tenham sido excluídas.

Quando usar o std::shared_ptr?

Devemos usar **std::shared_ptr** quando queremos atribuir um ponteiro bruto a vários proprietários.

std::weak_ptr

- Um **std::weak_ptr** é criado como uma cópia do **std::shared_ptr**;
- Fornece acesso a um objeto que pertence a uma ou mais instâncias **std::shared_ptr**, mas não participa da contagem de referência;

Quando usar o std::weak_ptr?

Quando você quer se referir ao seu objeto a partir de múltiplos lugares - para aquelas referências para as quais pode-se ignorar e desalocar.

std::weak_ptr

- Dependência Cíclica (Problemas com **std::weak_ptr**): Vamos considerar um cenário em que temos duas classes A e B, ambas possuem ponteiros para outras classes. Então, é sempre como A está apontando para B e B está apontando para A. Assim, **use_count** nunca chegará a zero e nunca será deletado.