

Manual do Programador para o Projeto de Princípios de Programação Procedimental



David Gomes 2013136061
João Craveiro 2013136429

Professor Tiago Baptista, PPP 2013/2014 Turma

Introdução

O principal objetivo deste manual é descrever a estrutura do código usada no nosso programa e clarificar as razões por detrás de algumas das nossas escolhas. O programa permite uma gestão dinâmica e eficiente das reservas e pré-reservas de uma estação de serviço, recorrendo a listas ligadas e ficheiros. Apesar de os dados serem gravados em ficheiros entre execuções, durante a execução do mesmo estes ficam alocados em listas ligadas.

Documentação Doxygen

Uma maneira simples de analisar o nosso programa é explorando a documentação gerada pelo Doxygen. Primeiro, deve-se instalar o Doxygen e depois correr, na diretoria-mãe do nosso projeto:

```
doxygen Doxyfile
```

Depois, o ficheiro `html/index.html` serve de página inicial para a versão HTML da documentação.

Por outro lado, colocamos *online* a mesma documentação aqui:

<http://student.dei.uc.pt/~drgomes/ppp-project/>.

Estrutura Geral

Para termos um programa dinâmico e com classes reutilizáveis, optamos por uma abordagem mais abstrata à criação das nossas estruturas. Para exemplificar o que foi anteriormente dito, podemos analisar a nossa estrutura `llist` (lista ligada):

http://student.dei.uc.pt/~drgomes/ppp-project/llist_8h_source.html

Usamos um ponteiro do tipo `void*` para os valores dos nós da lista ligada de forma a podermos usar a mesma estrutura para guardar vários tipos de valores.

Por outro lado, simulamos a existência de *namespaces* para as funções correspondentes a diferentes módulos prefixando `module_name_` antes do nome de cada função pertencente ao módulo `module_name`.

Classe `xtime`

Para todas as datas no nosso programa, usamos uma estrutura nossa chamada `xtime` que é bastante mais simples que a classe `time_t`. Claro, usar a classe `time_t` dar-nos-ia um programa mais portátil e mais independente já que a classe `time_t` está disponível livremente em diversas plataformas. No entanto, com o objetivo de simplificarmos o nosso código e

termos controlo total no mesmo, consideramos que usarmos a nossa própria estrutura foi uma decisão mais apropriada.

No entanto, é importante referir que acabamos por usar o `time.h` e o `time_t` para obter a hora atual ao registar novas reservas.

Utilitários

Para facilitar diversos componentes do nosso programa, criamos um *set* de funções utilitárias, abstratas e reutilizáveis. A documentação dessas funções pode ser vista aqui:

http://student.dei.uc.pt/~drgomes/ppp-project/utils_8h.html.

Algumas destas funções, como o `get_int_input` ou o `ask_date` servem para ler *input* e protegê-lo apenas uma vez para todo o programa.

Ficheiros

Devido à estrutura do nosso programa foi necessário a criação de funções diferentes para a escrita e leitura dos clientes e das reservas e pré-reservas.

As funções de escrita abrem o ficheiro em modo de escrita e percorrem as listas ligadas escrevendo através da função `fprintf` os dados de cada `node` linha a linha.

A leitura dos ficheiros é muito parecida em ambas as funções, lendo o ficheiro linha a linha e retirando a informação necessária. Primeiro começamos por ler os clientes e recriar essa `llist`, depois ao ler os ficheiros de reservas e pre-reservas, começamos por ler o nome do cliente e como a `client_list` já está ligada podemos associar cada reserva ao respectivo cliente retirando de seguida o resto da informação como as datas ou o tipo de reserva.

Os ficheiros são escritos sempre que necessário para evitar a perda de dados.

Além disto, todos os ficheiros de dados são guardados na pasta `data/`.

Coloração do terminal

Para tornarmos o nosso programa mais simples de utilizar decidimos colorar certas partes do mesmo. Embora possa ser mais ou menos abstraído, a nossa implementação das cores apenas funcionará em sistemas Unix-like (OS X, GNU/Linux).

Definimos um `colors.h` com algumas cores e até mesmo uma função `reset_color` para reiniciar a cor a branco.

Pré-Reservas

As pré-reservas do nosso programa servem para quando se tenta registrar uma pré-reserva por cima de uma já existente. Quando tal acontece, na lista ligada de pré-reservas é acrescentada uma nova reserva sendo que esta lista ligada está ordenada por prioridade (a reserva mais prioritária encontra-se no nó raíz).

Porque não usar uma *queue*? Começamos por pensar em usar uma *queue* mas dada a possibilidade de poder remover elementos que não a *front* da *queue* (um cliente cancelar a sua pré-reserva) decidimos que uma lista ligada seria mais apropriada.

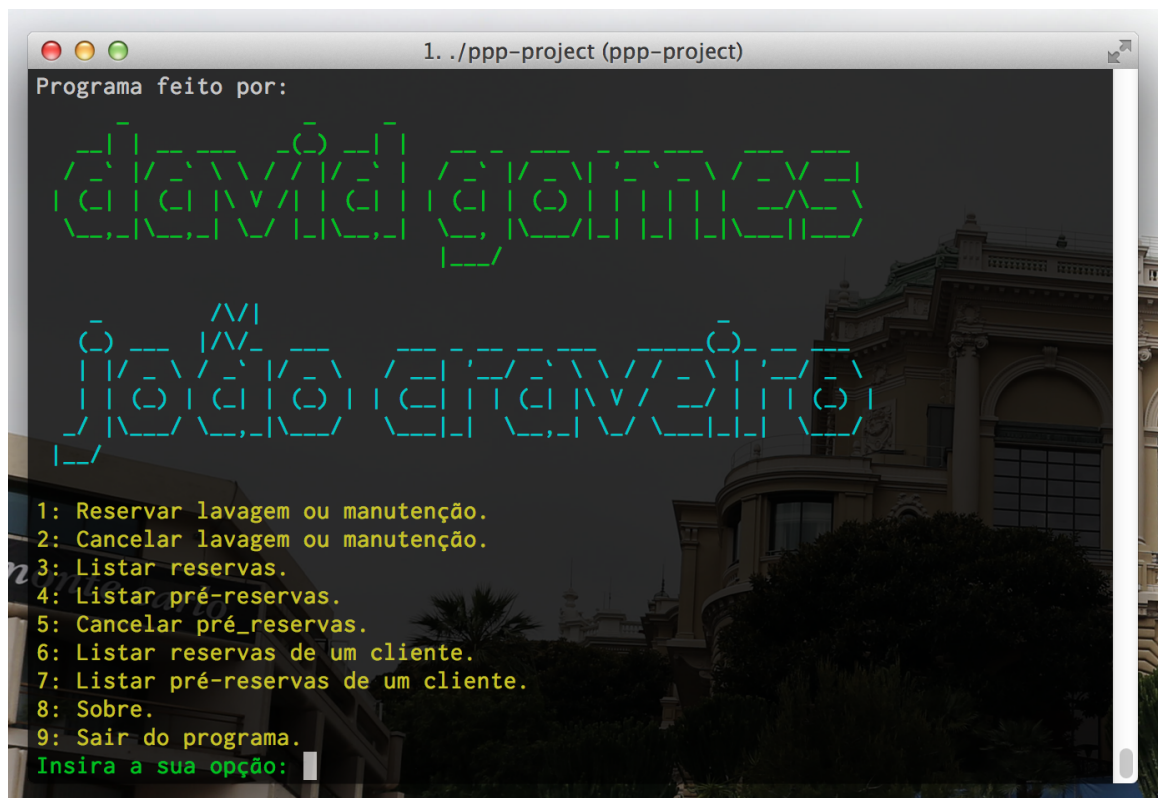
Compilação

Para compilar o nosso programa, siga os passos do ficheiro INSTALL.md incluído no código-fonte do programa.

Convenção de retornos

No nosso programa temos como convenção o retorno de 0 quando a função conclui sem encontrar algum problema, e retorno de 1 no caso contrário. Sempre que há um retorno de 2 o programa regressa ao main menu.

<https://github.com/davidgomes/ppp-project>

A screenshot of a terminal window titled "1. ./ppp-project (ppp-project)". The terminal displays a menu for a program made by "davidgomes" (in green ASCII art) and "davidgomes" (in cyan ASCII art). The menu lists nine options: 1: Reservar lavagem ou manutenção., 2: Cancelar lavagem ou manutenção., 3: Listar reservas., 4: Listar pré-reservas., 5: Cancelar pré-reservas., 6: Listar reservas de um cliente., 7: Listar pré-reservas de um cliente., 8: Sobre., 9: Sair do programa. The prompt "Insira a sua opção:" is followed by a cursor.

```
1. ./ppp-project (ppp-project)
Programa feito por:
davidgomes
davidgomes
1: Reservar lavagem ou manutenção.
2: Cancelar lavagem ou manutenção.
3: Listar reservas.
4: Listar pré-reservas.
5: Cancelar pré-reservas.
6: Listar reservas de um cliente.
7: Listar pré-reservas de um cliente.
8: Sobre.
9: Sair do programa.
Insira a sua opção: 
```

O nosso programa em execução.