Ficha nº 3 – *DLL – Bibliotecas de ligação dinâmica*

Âmbito da matéria

Ao longo desta ficha será possível:

- Criar e usar bibliotecas de ligação dinâmica (DLL) por ligação explícita e ligação implícita.

Pressupostos

- Conhecimento de programação, em linguagem C e C++, e de funções biblioteca standard destas linguagens.
- Conhecimento da matéria das aulas anteriores e das aulas teóricas.

Referências bibliográficas

- Material das aulas teóricas e incluindo documentos de apoio ao tópico de bibliotecas dinâmicas
- Capítulo 5 do Livro *Windows System Programming* (da Bibliografia) (pags. 167 em diante)
- MSDN:

  Run-Time Dynamic Linking (*Overview* geral)

  https://docs.microsoft.com/en-us/windows/win32/dlls/run-time-dynamic-linking

  Using Run-Time Dynamic Linking (Inclui exemplos)

  https://docs.microsoft.com/en-us/windows/win32/dlls/using-run-time-dynamic-linking

  Referência do API para lidar com DLL (*libloaderapi*)
  https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/

Introdução e contexto

Nesta ficha vai i) **usar bibliotecas dinâmicas** (".dll") de **forma explícita**, ii) de **forma implícita**, e iii) **criar** e usar bibliotecas dinâmicas suas.

Recorra aos conhecimentos das aulas teóricas, dos slides apresentados e do material de preparação para esta aula prática. Os conceitos são demasiado extensos para serem vistos pela primeira vez em 5 minutos no início da aula e é importante que tenha já presente os conceitos envolvidos, nomeadamente:

- ***Bibliotecas estáticas*** (ligada de forma permanente ao ficheiro executável do programa que a usa) *vs*. ***bibliotecas dinâmicas*** (ficheiro independente carregado em *runtime* para os processos que dela precisam).

- ***Ligação explícita*** (o programador que usa biblioteca controla a carga, a procura e uso dos recursos, e libertação da biblioteca) vs. ***ligação implícita*** (a carga, procura de recursos e libertação da biblioteca são feitas de forma transparente para o programador que usa a biblioteca).

- Ficheiros envolvidos fornecidos pelo autor da biblioteca: ***.dll*** (o código "executável" da biblioteca dinâmica); ***.lib*** (código de uma pequena biblioteca estática que faz a ponte entre o programa cliente e a biblioteca na ligação implícita, e que é adicionada ao projecto do programa que usa a biblioteca – a forma como isto funciona foi descrita nas aulas teóricas), ***.h*** (ficheiro *header* com protótipos das funções na biblioteca, necessário essencialmente no caso da ligação implícita, também a usar no projecto que usa a biblioteca); ***.c/.cpp*** (código fonte da biblioteca que não é fornecido nem faz falta a quem usa a biblioteca).

O link abaixo permite aceder o material de referência do API mais focado neste tema.

> **Referência do API para lidar com DLL**
> https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/

Esta introdução faz parte e consta nos documentos das aulas teóricas. É apresentado aqui como parte da ficha de exercícios para maior comodidade de consulta e incentivo de preparação para a aula. **De modo algum a inclusão deste conteúdo aqui dispensa as aulas teóricas**.

Exercícios

1. Utilização de biblioteca feita por terceiros, sem acesso ao seu código fonte.

   Obtenha o ficheiro "SO2 - 2022 - Ficha 3 - DLL.zip" que acompanha esta ficha. Contém ficheiros relativos a uma biblioteca dinâmica que lhe é disponibilizada na forma habitual: código binário, um *header* file e nenhum código fonte. Os ficheiros disponibilizados são:

   - Versão para arquitectura de **32 bits**: pasta ***x86*** , ficheiros *SO2_F3_DLL.**dll*** e *SO2_F3_DLL.**lib***
   - Versão para arquitectura de **64 bits**: pasta ***x64*** , ficheiros *SO2_F3_DLL.**dll*** e *SO2_F3_DLL.**lib***
   - *Header* file, comum a ambas as arquiteturas 32 e 64 bits: *SO2_F3_DLL**.h***

   a) Analise o ficheiro *header* disponibilizado e identifique os recursos que estão a ser disponibilizados (exportados) pela biblioteca. Trata-se de uma função e uma variável.
   A função devolve um valor que lhe é passado por parâmetro multiplicado pelo valor que está armazenado na variável.

2. Construa um projecto novo no IDE e prepare-o para utilização da biblioteca por **ligação explícita** da biblioteca que lhe foi disponibilizada.

a) Adicione um ficheiro com função *main* e usando as funções **`LoadLibrary`**, **`GetProcAddress`**, **`FreeLibrary`**, faça uso da biblioteca e dos recursos por ela disponibilizados de forma a que o seu programa consiga fazer:

- Pedir um valor ***a*** ao utilizador e colocá-lo na variável exportada pela DLL.
- Imprimir no ecrã o valor que está armazenado na variável na DLL, consultando directamente essa variável e não uma outra variável auxiliar qualquer. Confirme que lá está o valor que foi introduzido pelo utilizador.
- Pedir ao utilizador um segundo valor ***b***, passando-o como parâmetro à função da biblioteca e depois apresentar no ecrã o resultado da invocação da função, permitindo verificar que o resultado é o esperado.
- Este procedimento repete-se até que o valor ***a*** pedido ao utilizador seja -1.

Garanta que faz um uso eficiente e lógico da biblioteca, carregando-a apenas quando necessária, removendo-a apenas quando já não precisa dela. Não deve carregar/remover a biblioteca de cada vez que faz uso de um dos seus recursos, e, de igual forma, só deve procurar os recursos uma única vez (a primeira vez que os usa).

b) Execute o seu programa em simultâneo com ele próprio e em cada uma das execuções armazene valores diferentes na variável que é exportada pela DLL. Fazendo uso dos momentos que que o programa mostra o valor da variável e o resultado do seu uso, confirme que cada execução tem a sua cópia independente da variável, ou seja: o código da DLL poderá ser partilhado entre os processos, dependendo de várias circunstâncias descritas nas aulas teóricas, mas cada processo tem a sua cópia da variável exportada.

c) **Para fazer fora do tempo de aula caso já tenha decorrido mais de metade do tempo da aula**. Pode usar o ficheiro *header*, mas aqui, se não o usar, o impacto é mínimo. Experimente usar/não-usar o ficheiro *.h* e identifique qual a informação de que precisa e como pode avançar sem esse ficheiro.

d) **Para fazer fora do tempo da aula**: experimente correr o seu programa mudando o nome do ficheiro DLL e analise as circunstâncias em que o programa deixa de poder correr por falta da DLL. Use pausas ou leituras de caracter para controlar a temporização dessa experiência.

3. Repita todo o exercício 2 num novo projecto, mas desta vez usando a biblioteca por **ligação <u>implícita</u>**. Verifique que confirma e entende as razões dos seguintes aspectos:

a) Desta vez o projecto terá que ter a dependência para o ficheiro .lib, que deve estar disponível durante a linkagem.

b) O *header* file é agora muito mais relevante do que no caso anterior. Entenda o porquê disto.

c) Há diferenças quanto às conclusões da experiência descrita na alínea d) do exercício anterior.

4. Assuma que lhe foi revogada a licença de utilização da biblioteca disponibilizada anteriormente. Decide então meter mãos à obra e fazer a sua própria biblioteca dinâmica.

a) Faça uma biblioteca dinâmica que ofereça os mesmos recursos e funcionalidade daquela que esteve a usar até agora. Tome atenção que existem diferenças na construção do projecto no IDE quando se trata de fazer DLL, e não esqueça a questão de 32/64 bits. Dê um nome diferente aos ficheiros da sua biblioteca para não haver confusão com a anterior. A análise do ficheiro *header* que lhe foi dado no contexto da outra biblioteca poderá ajudá-lo a fazer o seu.

b) Modifique o projecto do exercício 2 de forma a usar a sua biblioteca e não a que lhe foi disponibilizada antes.

c) Repita o procedimento da alínea anterior para o projecto do exercício 3.

Esta ficha de exercícios não pode ser simplificada. É, na verdade, muito curta, e tudo o que é pedido aqui é relevante.

- Se não conseguir completar estes exercícios no tempo de aula, deverá continuar em casa/biblioteca e considere isso como tempo de estudo, como é natural em qualquer disciplina. Traga eventuais dúvidas pontuais que possam ter ocorrido para a aula seguinte.
- Se sentir que ficou muito aquém na resolução da ficha, poderá haver alguma falha quanto aos conhecimentos pressupostos. Procure identificar essa falha e tente colmatá-la o mais rápido possível. Poderá falar com um dos docentes da disciplina e pedir conselhos quanto a isso.

Listagem de Programas

Não existe nem é necessário nenhum código de partida nesta ficha. Quando muito, poderá precisar das linhas típicas de configuração da consola quanto a UNICODE. Se for mesmo preciso, copie essas linhas de um projecto anterior.

Resumo das funções API mais centrais a estes exercícios

- Existem mais funções que:
    - Pode usar em substituição das que foram mencionadas aqui.
    - Que podem ser necessárias a outros cenários maiores do que os desta ficha, por exemplo para encontrar recursos na DLL de outros tipos para além de funções e variáveis.
- A algumas das funções aplicam-se as questões de char/wchar, a outras não. Deve conseguir identificar essas situações e agir em conformidade. Neste documento são normalmente apresentadas as versões “A” (char), mas isso não significa que deve usar essas.
- A informação foi directamente obtida do site oficial da Microsoft. Tem um objectivo secundário de mostrar como se obtém este tipo de informação e irá ser dado progressivamente mais incentivo à sua consulta directamente no site e cada vez menos nestes documentos.
- A informação é apresentada no original em inglês, não só porque não faz sentido traduzir este tipo de material, mas também como forma de promover a competência de domínio de língua inglesa, cada vez mais necessária nas entrevistas de emprego. Esta observação vai deixar de ser feita.

Este resumo de API consta ou é mencionado em documentos do contexto de aulas teóricas. Esta lista não substitui as aulas teóricas e vai sendo apresentado na ficha de exercícios apenas para maior comodidade de consulta.

LoadLibraryExA

Loads the specified module into the address space of the calling process. The specified module may cause other modules to be loaded.

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-loadlibraryexa

```
HMODULE LoadLibraryExA(
  LPCSTR lpLibFileName,
  HANDLE hFile,
  DWORD  dwFlags
);
```

FreeLibrary

Frees the loaded dynamic-link library (DLL) module and, if necessary, decrements its reference count. When the reference count reaches zero, the module is unloaded from the address space of the calling process and the handle is no longer valid.

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-freelibrary

```
BOOL FreeLibrary(
  HMODULE hLibModule
);
```

GetProcAddress

Retrieves the address of an exported function or variable from the specified dynamic-link library (DLL).

https://docs.microsoft.com/en-us/windows/win32/api/libloaderapi/nf-libloaderapi-getprocaddress

```
FARPROC GetProcAddress(
  HMODULE hModule,
  LPCSTR  lpProcName
);
```