

# utad

UNIVERSIDADE  
DE TRÁS-OS-MONTES  
E ALTO DOURO



## EXERCÍCIOS EXTRA AULA DE PROGRAMAÇÃO MULTIPLATAFORMA

LICENCIATURA EM ENGENHARIA INFORMÁTICA E LICENCIATURA EM  
TECNOLOGIAS DA INFORMAÇÃO E COMUNICAÇÃO

EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

## TPCs C#.NET

Após o final de cada aula será sugerido que implemente um exercício que deve conter parte ou a totalidade da matéria lecionada na aula e poderá ainda ter alguma componente de inovação.

TPCs:

Exercício 1 .....	2
Exercício 2 .....	3
Exercício 3 .....	4
Exercício 4 .....	6
Exercício 5 .....	8
Exercício 6 .....	9
Exercício 7 .....	13
Exercício 8 .....	15
Exercício 9 .....	17
Exercício 10 .....	19

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 1

Desenvolva uma aplicação WPF (*Windows Presentation Foundation*) que implemente uma máquina de calcular com as operações aritméticas básicas (adição, subtração, multiplicação e divisão).

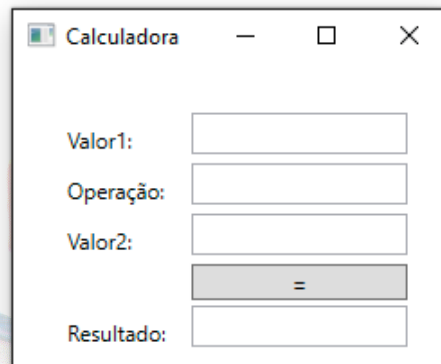


Diagrama de uma janela de calculadora com o título "Calculadora". A janela contém os seguintes elementos:

- Valor1:
- Operação:
- Valor2:
- Botão de igualdade:
- Resultado:

A janela da aplicação deve conter um formulário semelhante ao ilustrado na figura anterior.

No funcionamento da aplicação, quando o utilizador pressionar (clique) o botão, deve ser identificada a operação inserida pelo utilizador e calculado o resultado correspondente. O resultado deve ser colocado no *TextBox* correspondente.

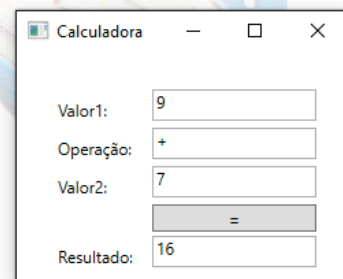


Diagrama de uma janela de calculadora com o título "Calculadora". A janela contém os seguintes elementos:

- Valor1:
- Operação:
- Valor2:
- Botão de igualdade:
- Resultado:

Tenha em atenção que:

- Sempre que o utilizador indicar uma operação não válida (diferente de +, -, \* e /) o resultado deve ser "Operador errado".
- Se o utilizador tentar calcular uma divisão com o "Valor 2" preenchido com o valor zero o resultado deve ser "Divisão por zero";
- Todos os restantes possíveis erros provenientes do cálculo dos resultados deverão ser ignorados;

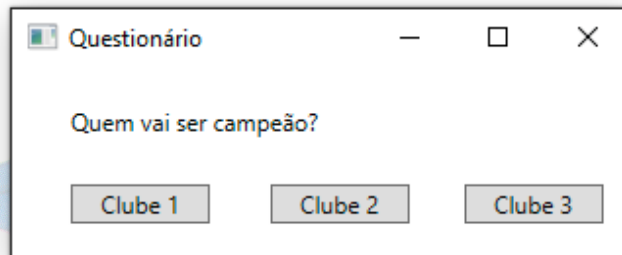
**Dicas:** para poder efetuar o cálculo numérico a partir do texto inserido, utilize a classe **Convert**. Todas as classes implementam o método **ToString** que permite representar qualquer grandeza em formato *string*.



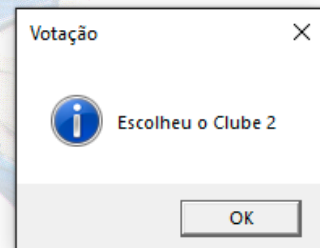
## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 2

Desenvolva uma aplicação WPF que implemente um mini questionário sobre quem vai ser o próximo campeão. As possíveis respostas devem ser representadas por botões (ver figura seguinte).

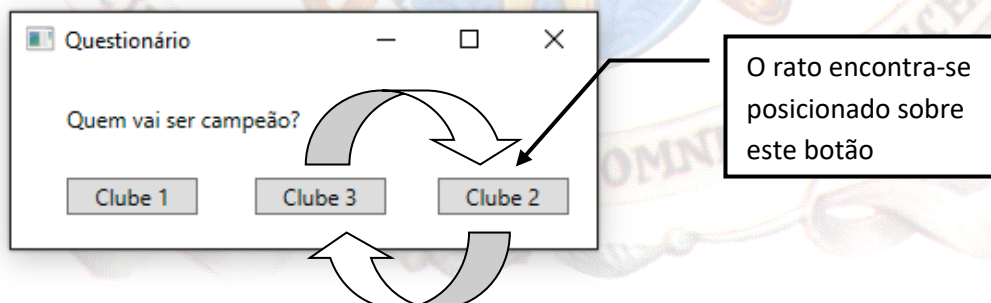


A escolha do utilizador é “registrada” através do clicar de um dos botões. Após a escolha feita, deve surgir, em formato de mensagem a escolha que foi efetuada.



Modifique o funcionamento do programa falseie o questionário e apenas aceite uma das possíveis escolhas. Para isso, sempre que o utilizador tentar colocar o rato sobre um dos botões que não seja a escolha pretendida, o programa deve “trocar” os botões de forma que o botão associado à resposta que se pretende ser a escolhida fique colocado por baixo do rato.

No exemplo da figura seguinte, pretende-se que a resposta seja sempre “Clube 2”. O utilizador ao tentar clicar na escolha “Clube 3” provocou a troca dos botões correspondentes.



**Dicas:** utilize a classe *MessageBox* para fazer surgir a mensagem resultante da escolha da votação. Para controlar o posicionamento dos botões utilize o evento *MouseEnter*.

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 3

Desenvolva uma aplicação WPF, com uma arquitetura MVVM simplificada, que permita efetuar a gestão do “fundo de manei” de um qualquer estudante da UTAD.

A(s) classe(s) que implementa(m) o *Model* deve(m) cumprir os seguintes requisitos:

- Manter sempre atualizado o valor monetário disponível;
- Registrar movimentos, que podem ser de despesa ou receita, que são representados por uma data, uma descrição e por um valor monetário;
- Lançar um evento de “Saldo Baixo – Nível 1” sempre que, depois de um qualquer movimento, o saldo ficar abaixo dos 20€;
- Lançar um evento de “Saldo Baixo – Nível 2” sempre que, depois de um qualquer movimento, o saldo ficar abaixo dos 5€;
- Lançar um evento de “Erro” sempre que, se tentar registar um movimento de despesa quando o saldo for inferior à quantia do movimento.
- **EXTRA:** Faça uma pesquisa na internet sobre o mecanismo de tratamento de exceções do C# e tente implementar o ponto anterior lançando uma exceção adequada;

Abandonar o programa.

Registrar novo movimento.

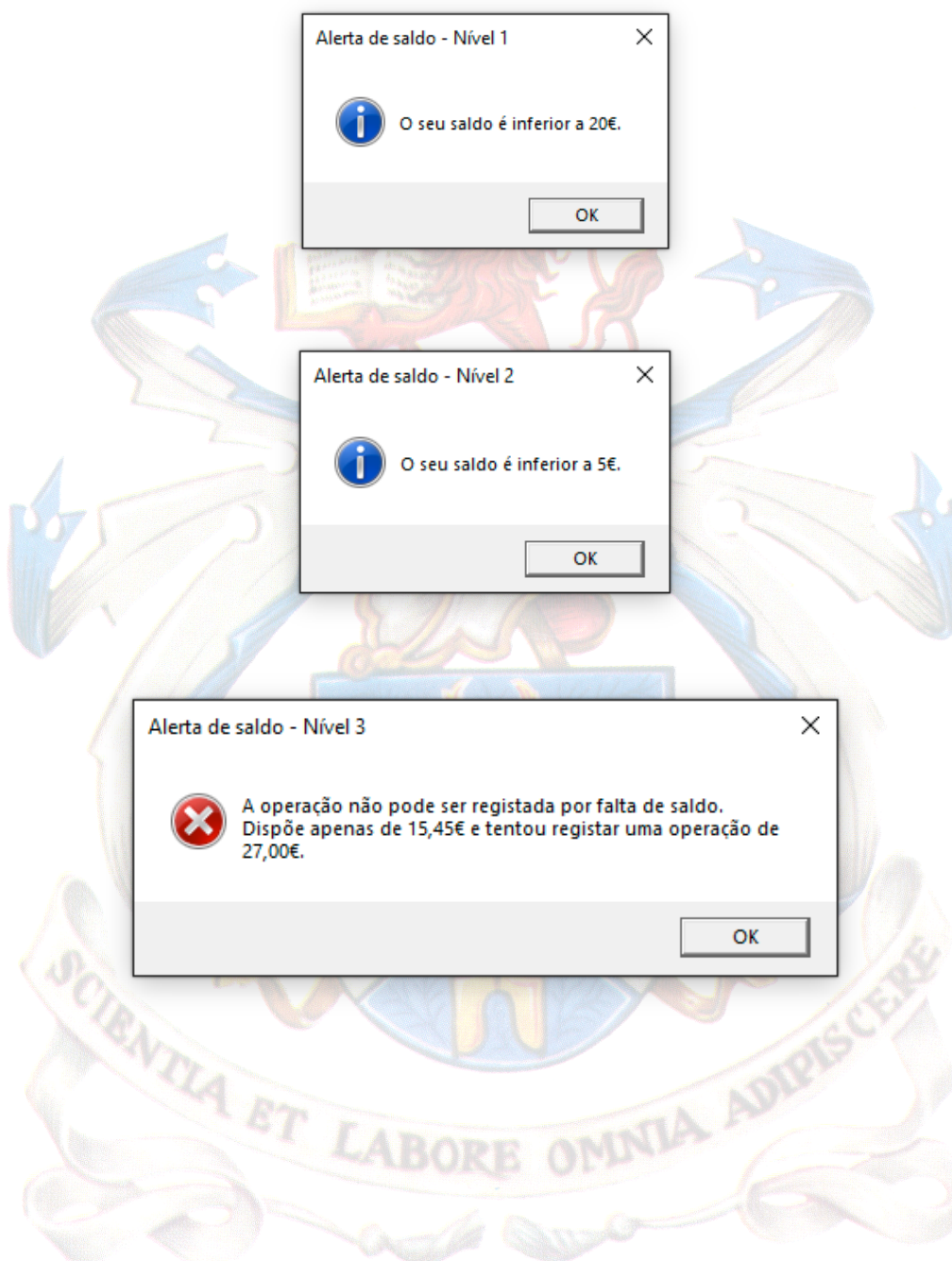
Data	Descrição	Quantia
23-01-2010	Almoço	5,20
23-01-2010	Fotocópias	12,00
23-01-2010	Autocarro	0,80
23-01-2010	Jantar	3,75
23-01-2010	Cinema	5,50
23-01-2010	Lanche	1,20
23-01-2010	Mesada	150,00
23-01-2010	Gasóleo	47,80
23-01-2010	Almoço	5,20
23-01-2010	Lavandaria	28,00

A(s) classe(s) que implemente(m) o *View* deve(m) permitir ao utilizador aceder à informação dos movimentos, dispondo das seguintes funcionalidades:

- Listar todos os movimentos registados (as receitas devem ser diferenciadas das despesas);
- Mostrar o saldo disponível;
- Registrar um novo movimento;
- Abandonar a aplicação;

EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

- Apresentar mensagens de alerta diferenciadas (Nível 1, Nível 2 e Erro) como ilustrado nas figuras seguintes;





## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 4

Desenvolva uma aplicação WPF, com uma arquitetura MVVM simplificada, que permita visualizar e alterar o conteúdo de um ficheiro de texto.

O ficheiro de texto contém o número, nome e curso de um qualquer aluno, com um campo em cada linha de texto.

ficheiro.txt

```
123456
Artur Aniceto Alves
Licenciatura em Pesca Desportiva
```

A aplicação deve logo de início ler a informação do ficheiro, mostrando-a ao utilizador no formato representado na figura seguinte:

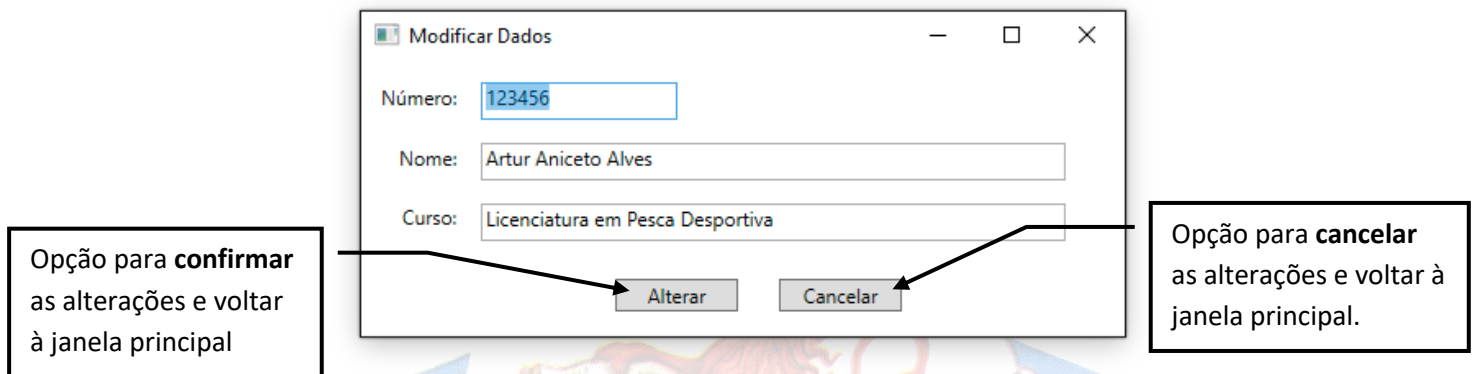
The screenshot shows a window titled "Janela Principal" with a title bar containing standard Windows window controls. Inside the window, there is a section titled "Informação" containing three disabled text boxes: "Número:" with the value "123456", "Nome:" with the value "Artur Aniceto Alves", and "Curso:" with the value "Licenciatura em Pesca Desportiva". To the right of the "Número" box is a button labeled "Modificar". At the bottom center of the window is a button labeled "Sair".

Annotations with arrows pointing to the interface elements:

- An arrow points from the "Número" text box to a box containing the text: "Informação retirada do ficheiro. (TextBox desabilitadas)".
- An arrow points from the "Modificar" button to a box containing the text: "Opção para alterar a informação."
- An arrow points from the "Sair" button to a box containing the text: "Opção para abandonar a aplicação."

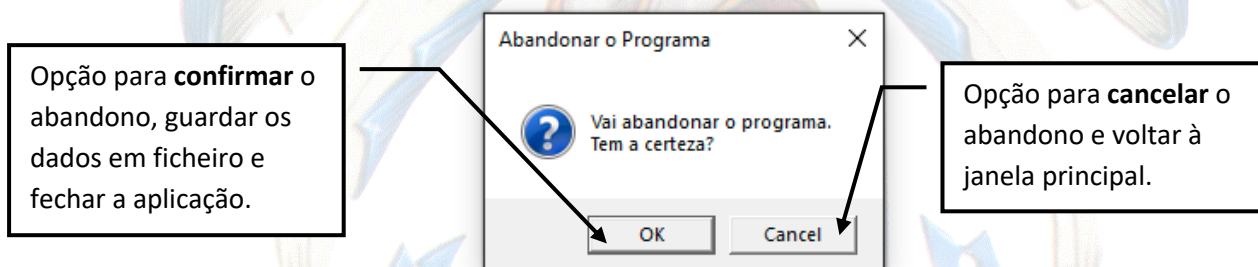
Quando o utilizador clicar na opção **Modificar**, deve surgir uma nova janela onde é apresentado o mesmo conteúdo mas num formato editável. Este conteúdo deve ser proveniente diretamente da janela principal.

EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA



Após a modificação dos dados, o utilizador pode escolher entre confirmar as alterações ou cancelar a operação. Caso escolha confirmar, todas as alterações feitas na informação deverão ser repercutidas na janela principal.

Na janela principal, quando o utilizador optar por Sair do programa, deve surgir um diálogo que questione o utilizador sobre a operação escolhida, permitindo cancelar a mesma (ver figura seguinte).



Quando o utilizador confirmar o abandono do programa, a informação que está na janela principal deve ser guardada (sobrepota) no mesmo ficheiro de onde foi retirada.

**Dicas:** utilize o evento *Loaded* da janela principal para ler a informação do ficheiro. Utilize a classe *MessageBox* para fazer surgir a mensagem de confirmação do abandono da aplicação. A janela de modificação dos dados deve conter propriedades para a transferência (bidirecional) da informação. Utilize a classe *File* para o acesso ao ficheiro de texto.

**EXTRA:** Crie a mesma aplicação em *Windows Forms*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) Model(s).



## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 5

Desenvolva uma aplicação de consola, com uma arquitetura MVVM simplificada, que proceda à leitura do seguinte texto a partir de um ficheiro para um objeto do tipo *String*.

A unidade curricular Programação Multiplataforma pretende servir como charneira no processo de aprendizagem de programação dos alunos. Os alunos, por contacto e utilização de um leque alargado de recursos de programação distintos, deverão efetuar a transição do código planeado e criado de raiz, para o código planeado e criado por recurso a componentes já existentes (quer em bibliotecas, quer por código aberto disponível na Web).

Os alunos deverão, igualmente, desenvolver a capacidade de abstração da estrutura do código, aprendendo conceitos de padrões arquitetónicos estruturantes do código.

Pretende-se igualmente que os alunos iniciem o contacto com o desenvolvimento de interfaces visuais de utilizador, com programação por objetos que incorpore utilização intensa de eventos e exceções.

1. Desenvolva um conjunto de métodos que permita retirar do texto (calcular) as seguintes características:
  - a. Número de parágrafos;
  - b. Número de palavras;
  - c. Número de caracteres (incluindo os separadores brancos);
  - d. Quantas vezes surge a palavra “alunos”;
2. Desenvolva um conjunto de métodos que permita efetuar as seguintes operações sobre o texto:
  - a. Substituir as palavras “Programação Multiplataforma” por “PM”;
  - b. Eliminar do texto todos os caracteres que se encontrem entre parênteses curvos (incluindo os parênteses);
3. Para cada uma das operações anteriores mostre o resultado no ecrã.

**EXTRA:** Crie a mesma aplicação em *WPF*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) Model(s).

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 6

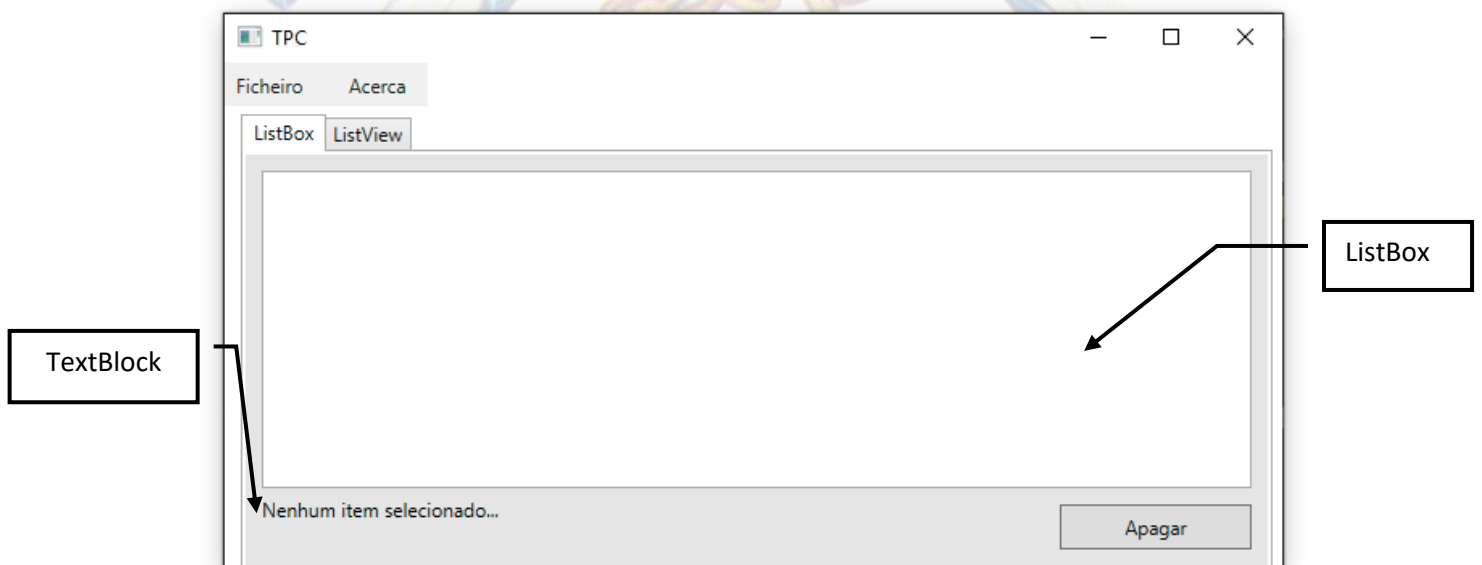
Desenvolva uma aplicação WPF, com uma arquitetura MVVM simplificada, que permita visualizar e alterar o conteúdo de um ficheiro de texto.

O ficheiro de texto contém a lista de alunos inscritos na UC “Programação Multiplataforma”. O ficheiro tem a informação estruturada, com os dados de cada aluno por cada linha de texto. Os dados de cada aluno são o número, nome e curso separados pelo carácter ‘#’ (ver ficheiro exemplo - alunos.txt).

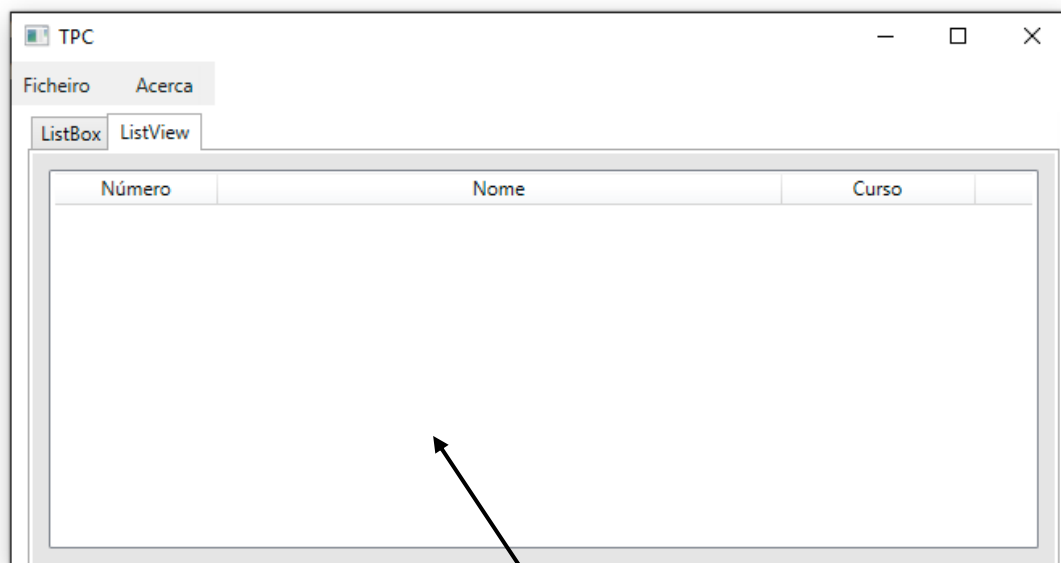
#### alunos.txt

```
28979#AGOSTINHO Gil Simão Ribeiro Carvalho CARDOSO#LEI
28560#ALBERTO Damião Canelas FRAGA#LEI
30528#ALEXANDRE Paiva PEREIRA#LEI
26802#ANA Cristina Salgado Teixeira Lima PEREIRA#LTIC
22361#ANA Cristina Teixeira Pinto MAGALHÃES#LEI
31904#ANDRÉ Gonçalves ALBERTO#LEI
...
```

A janela da aplicação contém um *Menu*, com as opções de acesso às funcionalidades que a aplicação fornece, e também um *TabControl* constituído por duas *Tabs*. A primeira *Tab* contém um *ListBox*, um *TextBlock* e um *Button*. A segunda *Tab* contém um *ListView*.



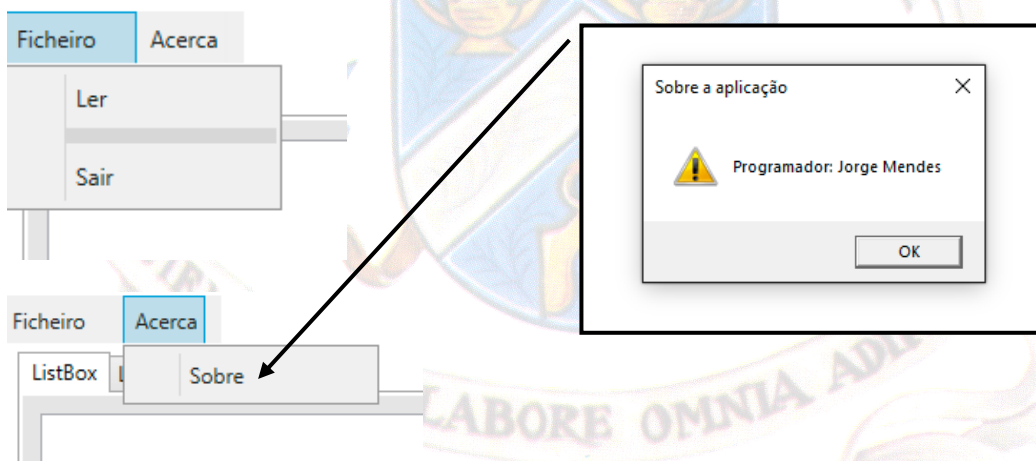
EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA



O menu contém três opções ativas:

- **Sobre**: faz surgir uma janela com o autor da aplicação;
- **Ler**: permite ler o ficheiro de texto;
- **Sair**: permite abandonar a aplicação.

ListView



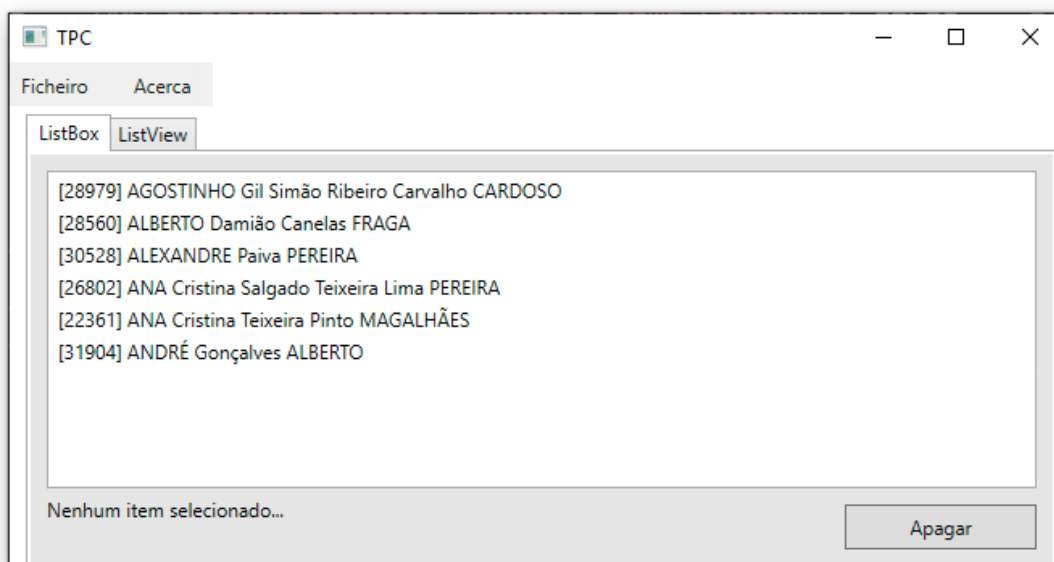
A opção **Ler** (do menu) desencadeia a leitura do ficheiro de texto (use um *OpenFileDialog* para a escolha do ficheiro). Nesta operação, todas as linhas de texto são lidas e, em cada uma delas, a informação de cada aluno é separada (número, nome e curso).

A informação dos alunos é colocada no *ListBox* e no *ListView*. Cada *Item* destes dois controlos contém informação referente a cada um dos alunos.

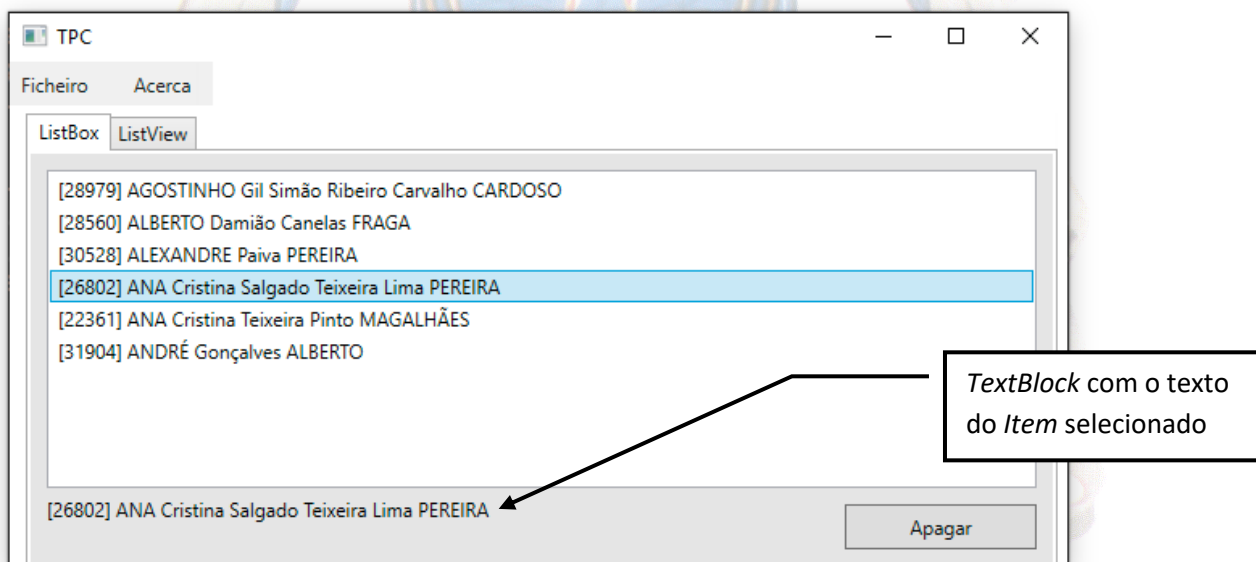


### EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

No *ListBox*, cada *Item* é constituído pela combinação do número e nome de cada aluno (ver figura seguinte).

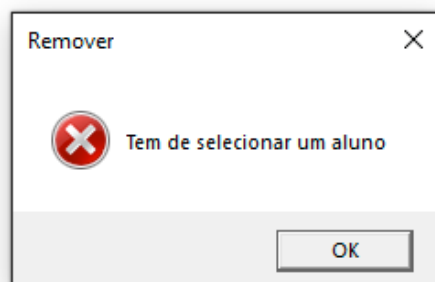


O *ListBox* permite que o utilizador selecione um *Item* da listagem. Após a seleção, a informação do *Item* selecionado aparece no *TextBlock* que está presente no *Tab*.

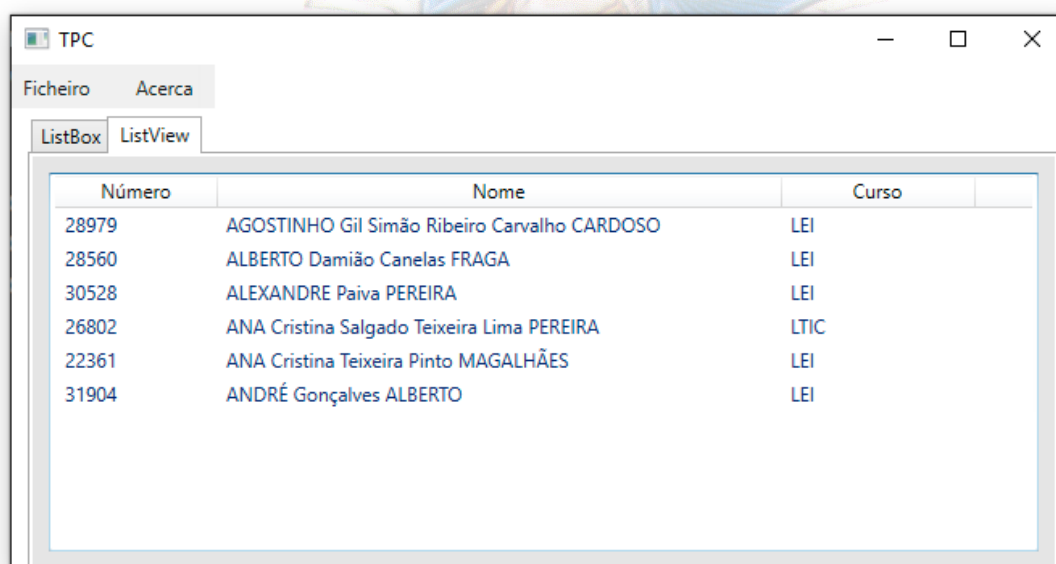


O *Button* (Apagar) permite que o utilizador elimine, do *ListBox*, o *Item* que está selecionado. Se esta opção for desencadeada e não houver nenhum *Item* selecionado, deve surgir uma mensagem ao utilizador conforme a figura seguinte.

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA



O controlo *ListView* é preenchido com a informação do ficheiro em simultâneo com o *ListBox*. No *ListView*, a informação está organizada por colunas.



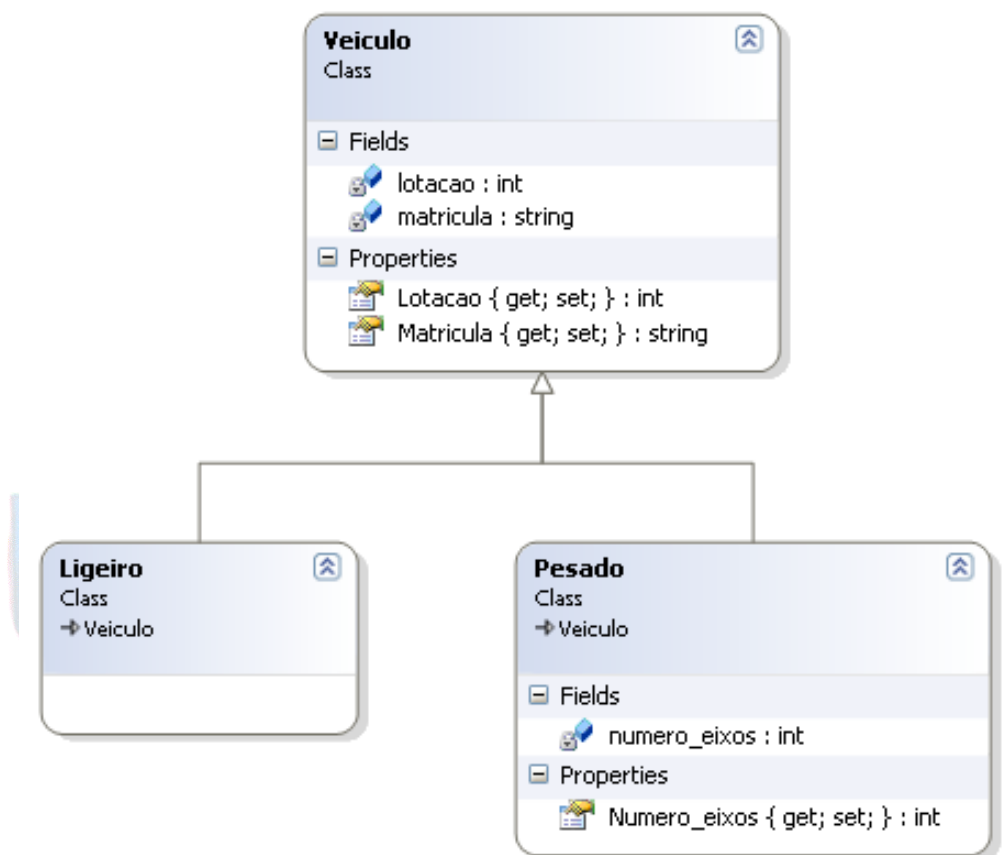
**Dicas:** utilize o método *Split* (da classe *string*) para separar os dados de cada aluno (número, nome e curso) nas linhas de texto.

**EXTRA:** Crie a mesma aplicação em *Windows Forms*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) Model(s).

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 7

Implemente uma aplicação WPF, com uma arquitetura MVVM simplificada, no qual o *Model* implementa as classes ilustradas nas figuras seguintes:



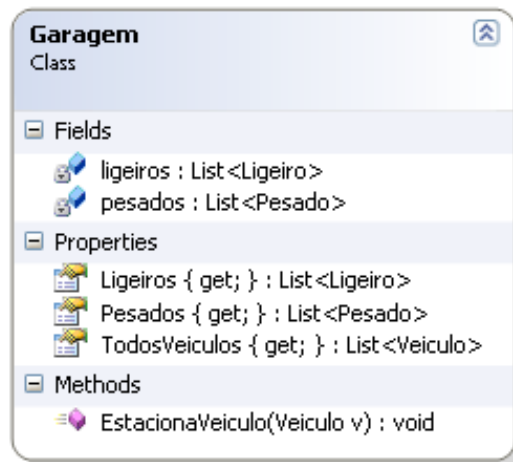
Na classe **Veiculo**, as propriedades *Lotacao* e *Matricula* fazem verificações aquando da operação **set**. A **lotacao** nunca pode ser inferior a 1. A **matricula** deve ter o formato da matrícula Portuguesa. A tentativa de armazenar valores que não respeitem estas condições provoca o lançamento de exceções de um tipo adequado à situação.

Para verificar a matrícula, considere as seguintes configurações de letras e algarismos.

Por exemplo: “76-AR-33” ou “77-41-FX” ou “AA-01-02” ou “AB-23-RT”



## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA



A classe **Garagem** permite “estacionar” veículos ligeiros e veículos pesados (instâncias das classes **Ligeiro** ou **Pesado**). O estacionamento, em ambos os casos, é efetuado sempre pelo método **EstacionaVeículo**.

Ainda na classe **Garagem**, a propriedade *Ligeiros* fornece uma lista dos veículos ligeiros estacionados e a propriedade *Pesados* fornece uma lista dos veículos pesados estacionados. A propriedade *TodosVeiculos* fornece a lista combinada de todos os veículos estacionados.

A aplicação a desenvolver deve apenas testar o *Model* implementado. Por exemplo, fazer estacionar vários veículos ligeiros e pesados e listar os veículos estacionados. Devem ser testadas as possíveis situações de exceção.

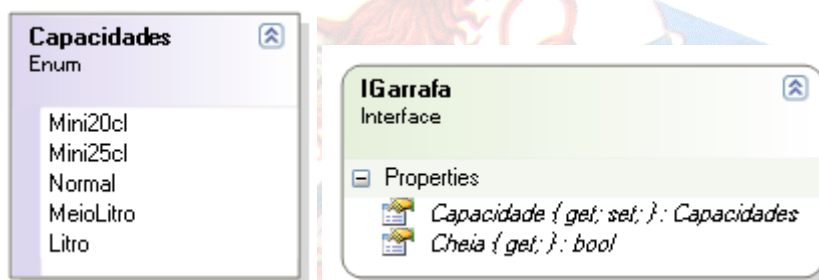
**EXTRA:** Crie a mesma aplicação em *Windows Forms*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) *Model(s)*.

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

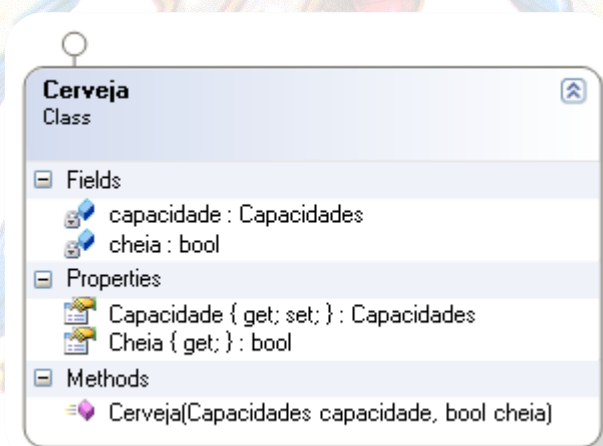
### Exercício 8

Desenvolva uma aplicação WPF, com uma arquitetura MVVM simplificada, que implemente a gestão de uma grade de cerveja. A aplicação deve ser baseada em duas classes, a cerveja e a grade.

Considere o tipo enumerado **Capacidades** e o interface **IGarrafa**. O tipo enumerado define quais os tipos de garrafas existentes. O interface define as regras para a declaração de uma embalagem garrafa.

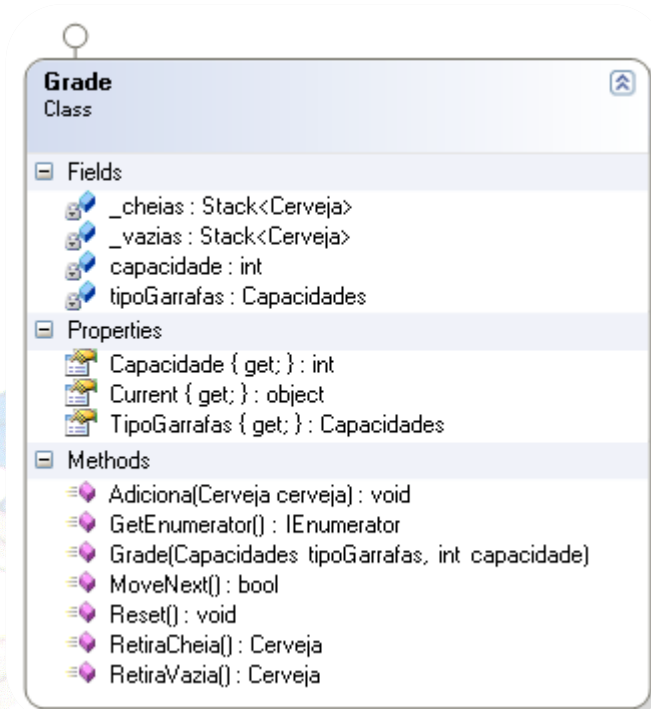


A classe **Cerveja** implementa o interface **IGarrafa**. No construtor desta classe é definida qual a capacidade da garrafa e se esta se encontra cheia ou vazia.



Para o armazenamento das garrafas de cerveja é implementada a classe **Grade**. Esta classe contém um construtor onde se define qual o tipo de garrafas a que se destina e qual o número de garrafas que comporta (capacidade).

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA



Esta classe armazena internamente as garrafas cheias e vazias em coleções diferentes. Para além destes requisitos, a classe implementa igualmente as seguintes funcionalidades:

- Método **Adiciona**: recebe uma cerveja, que tem de ser do mesmo tipo para o qual a grade foi declarada, e guarda-a nas coleções internas. A garrafa é “encaminhada” para a coleção adequada ao seu estado (vazia ou cheia). A capacidade de armazenamento da grade nunca é ultrapassada. A tentativa de guardar uma garrafa com a grade cheia é assinalada com uma exceção adequada.
- Método **RetiraCheia**: retira da grade uma garrafa cheia. Se a grade não possuir nenhuma garrafa cheia retorna o valor *null*.
- Método **RetiraVazia**: retira da grade uma garrafa vazia. Se a grade não possuir nenhuma garrafa vazia retorna o valor *null*.

A classe Grade implementa os interfaces *IEnumerator* e *IEnumerable*. Ambas as implementações providenciam o acesso às garrafas, vazias e cheias, como se tratasse de uma só coleção.

**Dicas:** utilize a coleção **Stack** (namespace *System.Collections*) para a implementação das coleções das garrafas vazias e das garrafas cheias.

**EXTRA:** Crie a mesma aplicação em *Windows Forms*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) Model(s).

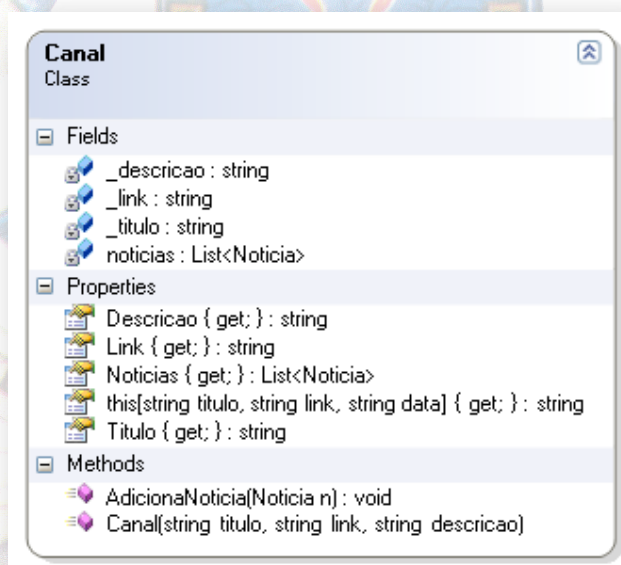
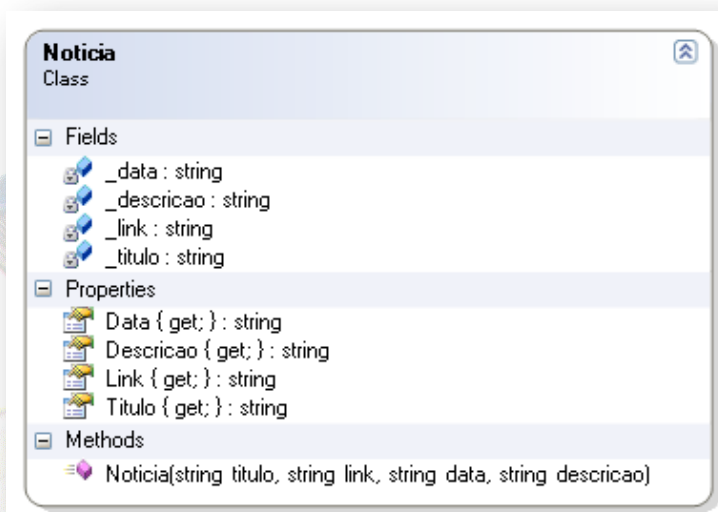


## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 9

Desenvolva a componente *Model* de uma aplicação WPF, com uma arquitetura MVVM simplificada, que permita efetuar a leitura de *streams* XML provenientes de fontes RSS.

A componente deve ser constituída pelas classes ilustradas nas figuras seguintes.



O *Model* deve ainda conter um método que receba o URL de um *feed* RSS e que retire e armazene as notícias/avisos provenientes desse mesmo *feed*.

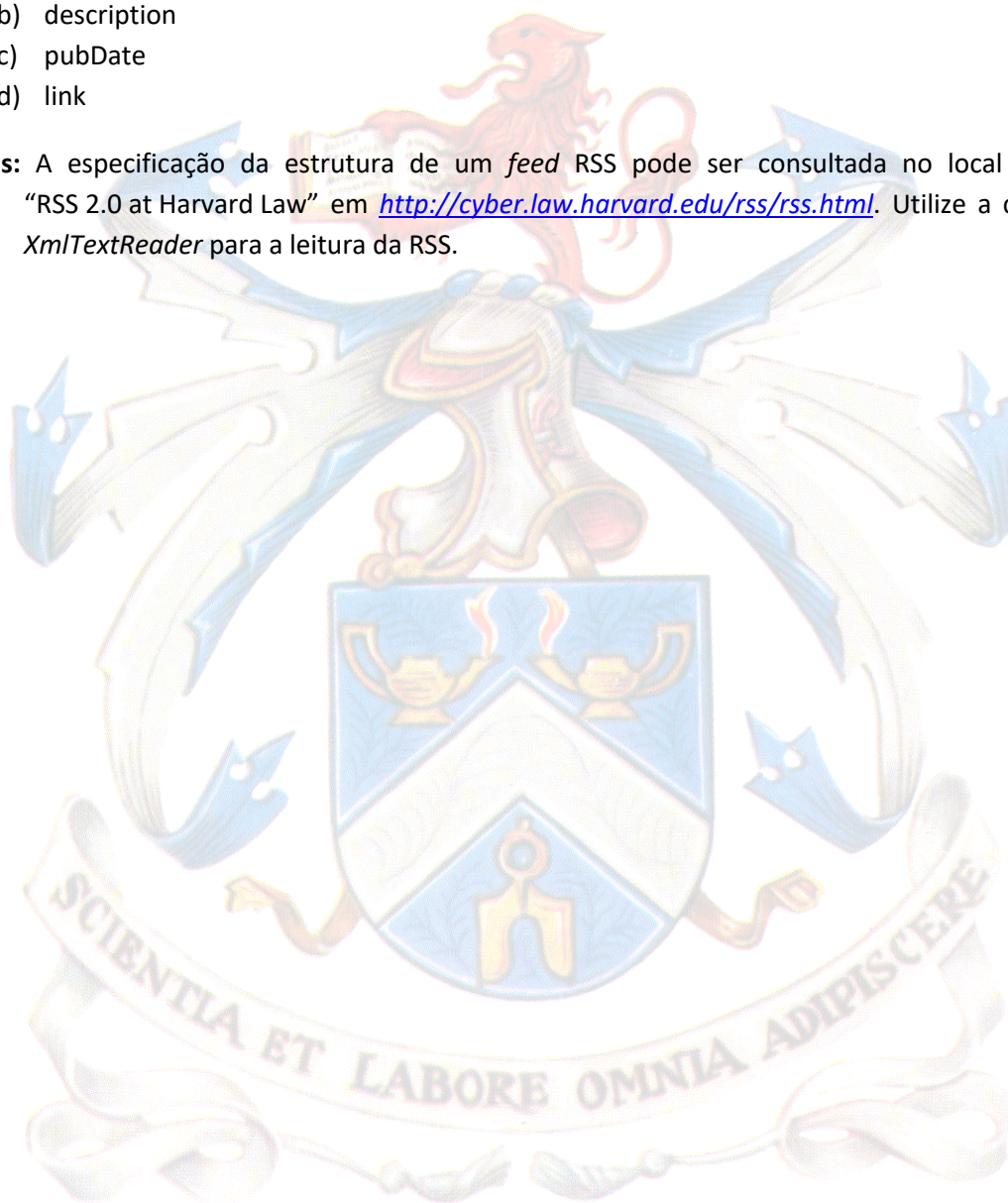
Na interpretação da RSS tenha em consideração apenas as seguintes TAGs:

## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

---

- 1) Para o Canal (channel):
  - a) title
  - b) description
  - c) link
- 2) Para cada notícia (item)
  - a) title
  - b) description
  - c) pubDate
  - d) link

**Dicas:** A especificação da estrutura de um *feed* RSS pode ser consultada no local Web “RSS 2.0 at Harvard Law” em <http://cyber.law.harvard.edu/rss/rss.html>. Utilize a classe *XmlTextReader* para a leitura da RSS.



## EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

### Exercício 10

Desenvolva uma aplicação WPF, com uma arquitetura MVVM simplificada, que permita efetuar a leitura de *streams* XML provenientes de fontes RSS.

A aplicação deve conter como componente *Model* as classes desenvolvidas no exercício anterior.

A aplicação deve ter uma interface com o utilizador igual à ilustrada na figura em baixo.

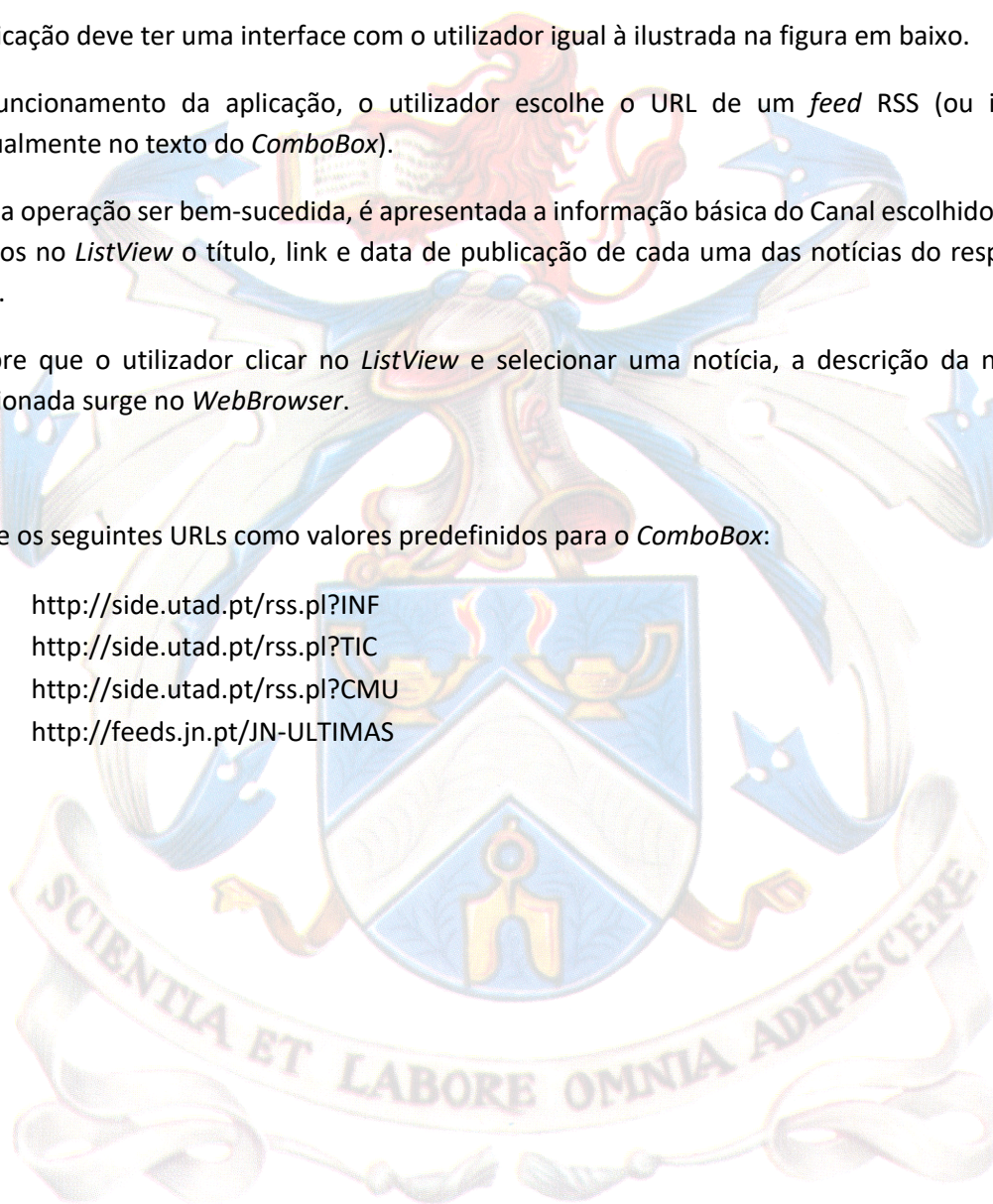
No funcionamento da aplicação, o utilizador escolhe o URL de um *feed* RSS (ou insere manualmente no texto do *ComboBox*).

Após a operação ser bem-sucedida, é apresentada a informação básica do Canal escolhido e são listados no *ListView* o título, link e data de publicação de cada uma das notícias do respetivo canal.

Sempre que o utilizador clicar no *ListView* e selecionar uma notícia, a descrição da notícia selecionada surge no *WebBrowser*.

Utilize os seguintes URLs como valores predefinidos para o *ComboBox*:

- <http://side.utad.pt/rss.pl?INF>
- <http://side.utad.pt/rss.pl?TIC>
- <http://side.utad.pt/rss.pl?CMU>
- <http://feeds.jn.pt/JN-ULTIMAS>





EXERCÍCIOS PARA IMPLEMENTAÇÃO EXTRA-AULA

The screenshot shows an application window titled "RSS Reader". It features a text box for the "RSS Feed" URL, currently set to "http://side.utad.pt/rss.pl?TIC". Below this, there are two text blocks displaying information from the feed: "Avisos do SIDE - Licenciatura em Tecnologias da Informação e Comunicação" and "Avisos do Sistema de Informação de Apoio ao Ensino". A table lists several news items with columns for "Titulo", "Url", and "Data". Below the table, there is a text block stating "O protocolo do Trabalho Experimental encontra-se na zona de download." and a "WebBrowser" control showing a detailed description of the selected news item.

**Annotations:**

- ComboBox com URLs de feeds RSS pré-definidos e com a possibilidade de inserir um de forma manual:** Points to the "RSS Feed" text box.
- TextBlocks com informação obtida do Canal:** Points to the two text blocks above the table.
- ListView com as notícias do canal selecionado:** Points to the table of news items.
- WebBrowser com a descrição da notícia selecionada no ListView:** Points to the "WebBrowser" control at the bottom.

Titulo	Url	Data
Dia da Universidade	http://side.utad.pt/cursos/tecno	Mon, 15 Mar 2020 11:52:58 GMT
Programa TANDEM	http://side.utad.pt/cursos/tecno	Sun, 28 Mar 2020 23:45:49 GMT
Técnicas Avançadas de Bases de Dados - Trabalho Experin	http://side.utad.pt/cursos/tecno	Fri, 26 Mar 2020 20:00:47 GMT
Laboratório de TIC III - Turma TP2 - Fecho de turma	http://side.utad.pt/cursos/tecno	Fri, 26 Mar 2020 16:30:18 GMT
Prémios APDA - Ensino Superior 2020	http://side.utad.pt/cursos/tecno	Fri, 19 Mar 2020 14:29:08 GMT
Inglês e Técnicas de Comunicação II - TIC English I	http://side.utad.pt/cursos/tecno	Wed, 24 Feb 2020 17:12:59 GMT
Convite à participação - IBM Lotus U-CRE8! Mashup Mani	http://side.utad.pt/cursos/tecno	Wed, 10 Mar 2020 22:48:09 GMT

**EXTRA:** Crie a mesma aplicação em *Windows Forms*, com uma arquitetura MVVM simplificada, reutilizando o código do(s) Model(s).