Exame | Instruções para a realização de exame

LABORATÓRIO DE DESENVOLVIMENTO DE SOFTWARE | 21179

Data e hora de realização

28 de junho de 2022, às 15h00 de Portugal Continental

Duração da prova

120 minutos + 60 minutos de tolerância

Temática / Tema / Conteúdos

Matéria teórica do semestre e sua aplicação em código

Objetivos

- Aferir a compreensão dos temas teóricos estudados
- Aferir a compreensão da aplicação prática desses temas sobre o código, através do uso de eventos e delegados, exceções e interfaces.

Trabalho a desenvolver

1.ª Parte (4 Valores)

Código de referência

Este bloco de código foi extraído e adaptado dos exemplos da biblioteca PESocket, que permite estabelecer comunicações em rede, criando clientes e servidores, disponível em: https://github.com/PlaneZhong/PESocket/blob/master/Example/ConsoleProjects/ConsoleClient/ClientStart.cs:

```
1  //Classe de configurações
2  public class IPCfg {
3          public const string srvIP = "127.0.0.1";
4          public const int srvPort = 17666;
5  }

6  static void Main(string[] args) {
7      //Criar e configurar objeto de gestão das comunicações
8      cliente = new PESocket<ClientSession, NetMsg>();
9      cliente.StartAsClient(IPCfg.srvIP, IPCfg.srvPort);
10     //Explicar ao utilizador o modo de interação
11     Console.WriteLine("\nEscreva 'sair' para parar o cliente!");
12     while(true) {
13     //Obter comando
14         string ipt = Console.ReadLine();
15     //Interpretar comando e decidir o que fazer
16         if(ipt == "sair") {
17     //Encerrar o cliente de rede
18             cliente.Close();break;
19         }
20         else {
21     //Enviar mensagem pela rede
22             cliente.session.SendMsg(new NetMsg {text = ipt});
23         }
24     }
25 }
```

As alíneas seguintes são todas de resposta curta.

1.

a) Indique as linhas do código de referência onde há operações de *input*.

b) Indique as linhas do código de referência onde há operações de *output.*

Caso tenha tido dúvidas de interpretação do código que lhe afetaram a resposta, exponha-as por escrito.

2. Suponha que queria reescrever o código de referência segundo o estilo arquitetónico MVC. Indique como distribuiria pelos componentes as tarefas descritas nos comentários desse código. Por ex., se considerar que a tarefa "`//Classe de configurações`" (linha 1) corresponde ao Controller, marque a célula C1, se acha que corresponde ao Model, marque a célula M1, para indicar que corresponde à View, marque a célula V1.

a) Segundo a abordagem de Krasner & Pope (1988).

Linha → / ↓ Componente	1	7	10	13	15	17	21
M							
V							
C							

b) Segundo a abordagem de Curry & Grace (2008).

Linha → / ↓ Componente	1	7	10	13	15	17	21
M							
V							
C							

c) Explique as dúvidas ou dilemas com que se debateu para responder às alíneas a) e b) e justifique as principais opções que tomou ao dar as suas respostas.

3. Suponha que a linha 22, `cliente.session.SendMsg`, não obtinha resposta do servidor a confirmar a receção da mensagem.

a) Imagine que, como consequência disso, o programa ficava preso nessa linha, sem voltar ao `Console.ReadLine`. Nesta situação específica, indique o que é a falta, o erro e a falha.

b) Imagine um bloco `try-catch` em redor da linha 22. Este problema não se alterou, o programa continua preso na linha. Explique se é normal o `try-catch` não resolver a situação.

4. No código original, `if(ipt == "sair")` chama diretamente a um método do objeto `cliente`, com os parâmetros necessários. Como poderia aumentar a independência entre a interpretação do texto do comando e a execução de componentes posteriores?

2.ª Parte (8 Valores)

5. Reescreva o código de referência segundo o modelo MVC, de acordo com as seguintes alíneas. Como comentário no início do código, indique se está a usar a sua resposta 2a ou 2b como base para esta parte.

a) Para não ter de usar a biblioteca PESocket, crie uma classe `PESocket` com métodos (vazios de código) usados no exemplo. Nota: não é necessário ter a funcionalidade de *template* <…> no construtor. Estamos apenas a evitar ter de usar a biblioteca.

b) Crie classes para os componentes Controller, View e Model e dentro delas métodos (código vazio, apenas indicação dos parâmetros de entrada e tipo de retorno) correspondentes à distribuição de responsabilidades que fez na sua resposta 2a/2b.

c) Defina uma exceção para o caso de *timeout* na comunicação pela rede no envio de mensagem e lance-a dentro do objeto cliente (pode apenas fazer `if (true) //teste de timeout` para simular a deteção do *timeout*).

3.ª Parte (8 Valores)

6. Com a sua resposta à pergunta 5, reescreva o código para obter independência de componentes e separação de interesses, sem ter de ser código funcional. Por ex., não é necessário que a classe `PESocket` que criou na alínea efetue comunicações, basta que o simule (por ex. escrevendo no ecrã).

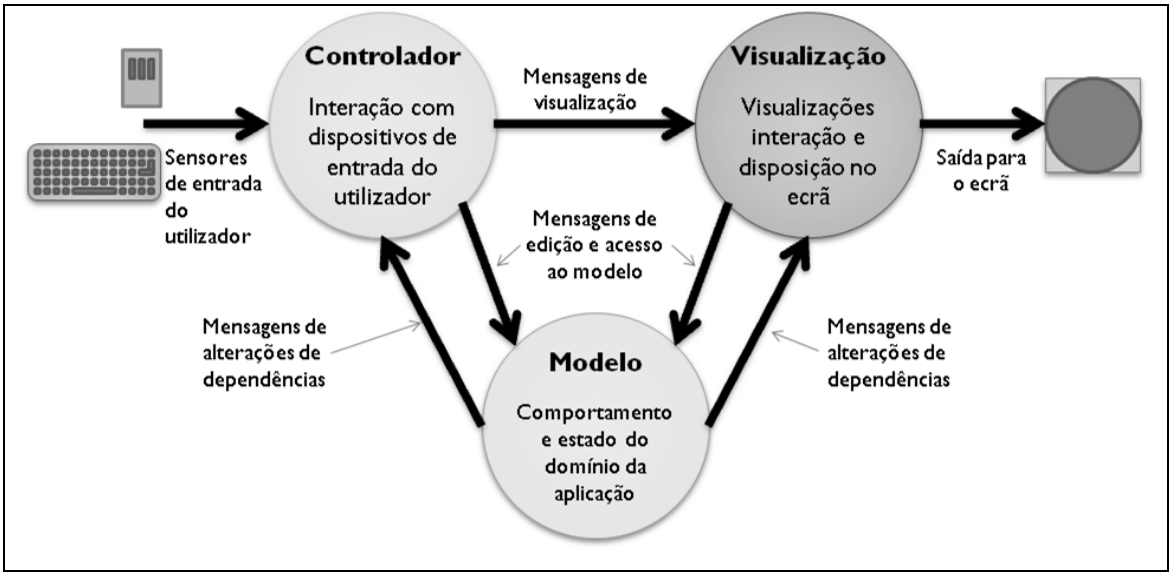
a) Defina (não é preciso implementar, apenas apresentar as linhas de definição) eventos e delegados que serão usados para reportar alterações entre componentes.

b) No componente Controller, ligue os eventos dos componentes aos delegados respetivos.

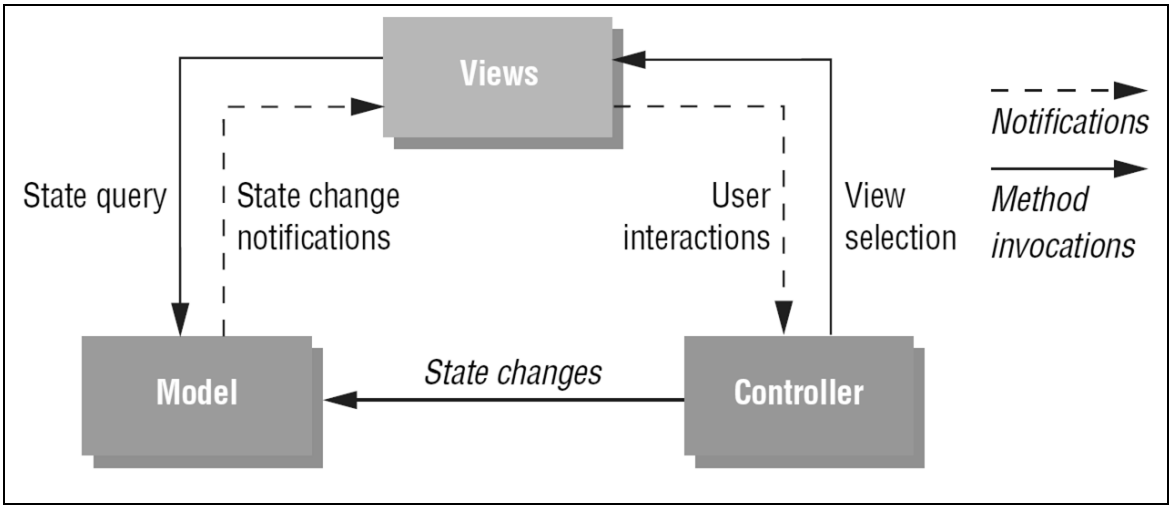
c) Faça um `try-catch` no Controller que possa apanhar a exceção implementada na resposta à alínea 5c. No `catch` quer-se viabilizar uma reação prestável ao utilizador, respeitando as responsabilidades de cada componente no MVC. Indique em comentário, dentro do `catch`, o que faria para o concretizar.

Recursos

KRASNER & POPE (1988)



CURRY & GRACE (2008)



APOIOS DE SINTAXE

```
throw new ClasseDaExcecao(ParametroDaExcecao);
catch (ClasseDaExcecao VariavelDaExcecao) { }
public delegate TipoDevolvido
    NomeDoTipoDeDelegado(Parametro1,Parametro2);
NomeDoTipoDeDelegado delegado = new
    NomeDoTipoDeDelegado(MetodoAtribuido);
public event NomeDoTipoDeDelegado NomeDoEvento;
NomeDoEvento += delegado;
NomeDoEvento += new NomeDoTipoDeDelegado(MetodoAtribuido);
```

Critérios de avaliação e cotação

Critérios de avaliação e cotação

1.ª parte: 4 valores, distribuídos da seguinte forma:

1. a) 0,2 valores
   b) 0,2 valores
2. a) 0,5 valores
   b) 0,5 valores
   c) 0,2 valores
3. a) 0,5 valores
   b) 0,4 valores
4. 1,5 valores

2.ª parte: 8 valores, distribuídos da seguinte forma:

a) 1 valor
b) 4 valores
c) 3 valores

3.ª parte: 8 valores, distribuídos da seguinte forma:

a) 3 valor
b) 3 valor
c) 2 valor

Total: 20 valores

Normas a respeitar

Deve redigir o seu Exame na Folha de Resolução disponibilizada na turma e preencher todos os dados do cabeçalho.

Se necessitar de anexar ficheiros de código-fonte, compacte a folha de resolução e esses ficheiros de código-fonte num único ficheiro .zip.

Todas as páginas do documento devem ser numeradas.

Nomeie o ficheiro com o seu número de estudante, seguido da identificação do Exame, segundo o exemplo apresentado: 000000exame.

Deve carregar o referido ficheiro para a plataforma no dispositivo Exame até à data-limite e hora-limite de entrega. Evite a entrega próximo da hora-limite para se precaver contra eventuais problemas.

O ficheiro a enviar não deve exceder 8 MB.

Votos de bom trabalho!

Leonel Morgado