## 项目经理/.NET高级开发工程师笔试题

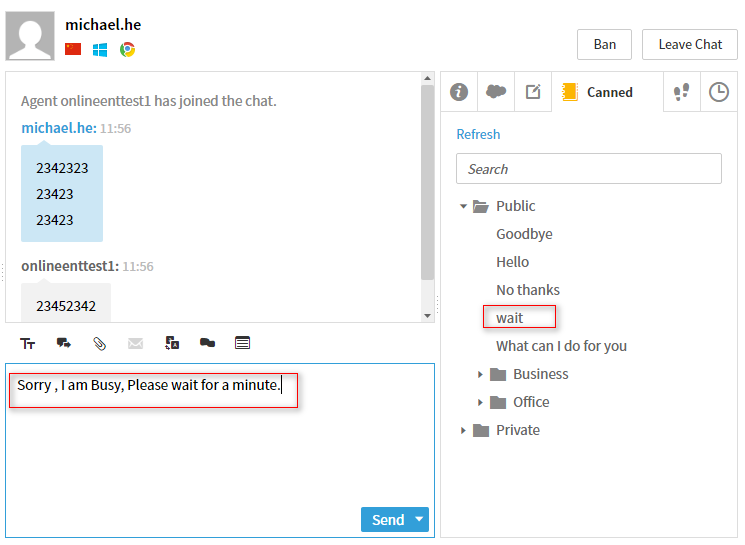
#### 一、C# &基础

某在线客服系统采用传统的三层架构

* **Data Access 层:** 负责构建SQL脚本, 直接与数据库进行交互.
* **Business 层:** 封装了实体的业务逻辑, 需要调用Data Access 层的方法才能执行对应的数据库操作.
* **Process 层:** 统一生成数据库连接、管理事务、调用Business 层的方法来处理UI层发送过来的请求.
* **UI层:** 与用户进行交互, 调用Process 层的方法完成各种UI请求.

客服在线聊天时会使用到Canned Message

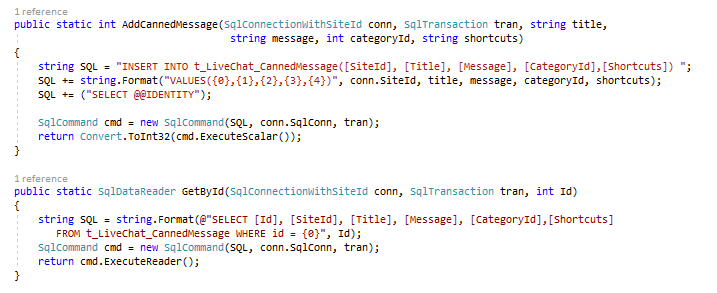
Canned Message : 预存储的常用聊天消息, 按Category 进行分类, 用于减轻客服在线聊天时的工作量. 包含Category, Title, Name, Message, Shortcuts 等属性.



举例: 客服在聊天过程中单击了Tab[Canned] 下Name为”wait”的canned message, 系统将会自动发送对应的消息: “Sorry, I am Busy, Please wait for a minute”.

以下是Canned Message 实体的相关代码:

* Data Access 层:



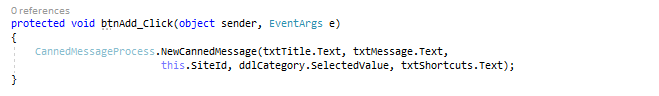
* Business 层：



* Process 层:



* UI层:



试找出上述代码的存在的错误和可优化点

答：

1：UI层 发送预存储的消息采用单击发送容易产生误操作，建议改为DoubleClick双击或者增加发送按钮等。

2：Data Access层 对数据库进行增改查询等操作时，没有对Title，Message等字段进行SQL关键字或敏感字符的特殊处理，容易产生系统异常等，甚至威胁到数据库安全。建议对这些字段进行预处理（Encode等）。同时需要增加try/catch处理异常，避免系统崩溃。（+3）

3：AddCannedMessage方法内sql语句 “VALUES(…..)”括号内的字段若为字符类型需要用单引号包围。（+3） “select @@ IDENTITY” 前需要加空格（+3）。

4：AddCannedMessage方法使用了事务tran，需要先Begin开始事务，再提交事务tran.Commit()，同时若发生错误需要回滚事务。处理完成后建议Close关闭并Dispose销毁SqlCommand和SqlTransaction。

5：Business 层 的Add方法建议增加对同一Category内的相同内容消息进行判断，内容相同时不插入数据库，同时UI层回馈给用户（+3）。同时GetById方法对数据库操作时建议增加try/catch处理异常。

6：Process 层 建议最后Close关闭并Dispose销毁SqlTransaction

7：UI 层 未处理New Message抛出的异常，建议增加try/catch块并提示用户出现的错误或者写入错误日志。（+3）

#### 二、数据库

有一Ticket（个案或工单）系统，用于处理来自于客户的邮件、在线表单等问题。其中，个案的编号、标题、描述、创建时间、状态、负责人、部门等固定字段30个。用户会根据自己的业务需求添加或删除其它自定义字段，比如产品、渠道、咨询类别等。

自定义字段类型包括：文本框，下拉框，复选框，复选框列表等。其中，下拉框/复选框列表是指用户可以预定义数据，然后在具体新建或编辑Ticket的时候，客服可以选择其中的一个值。比如，用户可以加一咨询类别，预定义类别为：售前咨询，产品质量，支付问题等。在客服在新建或修改产品的时候，客服可以选择一个或多个选项。

业务系统的使用或数据量需求：

1. 我们假定系统有非常多的Ticket，比如>100万个Ticket。
2. 系统设计应该支持客户针对Ticket字段的频繁查询。比如客户定义自己的查询为“部门为客服，负责人是自己，状态为Open，自定义字段-资讯类别为产品质量”的Ticket，经常使用
3. 用户一般在使用初期会根据业务需求自定义字段，之后在系统使用过程自定义字段新建和修改会比较少；
4. 用户查询和修改一月之前的Ticket的概率会比最近时间的Ticket的概率低很多，大概1：100；

**问题：**在关系型数据库的基础上，请定义能实现自定义和存储Ticket的表结构；

#### 三、设计

在线客服系统是一款基于Web的实时聊天系统，用户帮助企业的客服人员回答网站访客的问题。请根据以下需求场景描述，对系统进行类设计：

1. 聊天（Chat）的双方为访客（Visitor）和客服（Agent）；
2. 客服为系统预设，属性包含Id、Name；
3. 一个聊天只能有一个访客参加，可以有多个客服参加；
4. 聊天由访客请求发起，客服人员接受请求后，双方可以聊天；
5. 聊天过程中，另外的客服人员可以加入；
6. 聊天中访客和客服都可以发送和接收聊天消息（Message）；

类设计要求体现需求描述内容和对象关系，要求包含类的定义和成员；

不限制语言，不要求实现函数体内容，不要求体现需求未描述内容；

/\*\*

/用户类型

\*\*/

Public enum UType

{

PVisitor,

PAgent,

Other

}

/\*\*

/用户基类

\*\*/

Public abstract class Person

{

Public UType UserType;

Public long UserId;

Public String Name;

Public void SendMessage(String msg);

Public void GetNewestMessages(long lastmsg);

}

/\*\*

/访客类

\*\*/

Public class Visitor: Person

{

Public Visitor()

{

This.UType= PVisitor;

}

Public bool SendAddChatRequese(long userid)

{

……

}

……

}

/\*\*

/客服类

\*\*/

Public class Agent: Person

{

Public deleget bool GetAddChatResponse(long chatid);

Public event GetAddChatResponseHandle;

Public Agent ()

{

This.UType= PAgent;

GetAddChatResponseHandle += DoGetAddChatResponse;

}

DoGetAddChatResponse(long chatid)

{

…

}

……

}

/\*\*

/聊天组类

\*\*/

Public class Chat

{

Public readonly long ChatId;

Public readonly List< Person> ChatMembers;

Public readonly List<Message> ChatMessages;

Public readonly long CreatedTime ;

Public Chat(int chatid)

{

This.ChatId=chatid;

ChatMembers=new List< Person>();

ChatMessages =new List< Message>();

CreatedTime = DateTime.Now.Ticket;

}

……

}

/\*\*

/聊天内容类

\*\*/

Public class Message

{

Public readonly long MessageId;

Public String MsgText {get;set}

Public long MsgTimeTickt {get; set;}

Public int MsgFrom {get;set}

……

……

}