

```
        return S_OK;
    }
    */
    if (pRs->ADOEOF)
    {
        *x_UserInfo = ::SysAllocString(L"N#Couldn't find
        records!");
        return S_OK;
    }

    CComVariant m_varData;
    CComBSTR m_bstrInfo;
    m_bstrInfo.Append("Y#");    //返回的字符串中以#作为边界符
    //获得字段名为 USERNAME 的值
    pRs->GetFields()->GetItem("USERNAME")->
    get_Value(&m_varData);
    if(m_varData.vt == VT_BSTR)
    {
        m_bstrInfo.AppendBSTR(m_varData.bstrVal );
        m_bstrUsername.Empty();
        m_bstrUsername.AppendBSTR(m_varData.bstrVal);
    }
    m_bstrInfo.Append("#");
    //获得 Password 字段的值
    pRs->GetFields()->GetItem("PASSWORD")->
    get_Value(&m_varData);
    if(m_varData.vt == VT_BSTR)
    {
        m_bstrInfo.AppendBSTR(m_varData.bstrVal );
        m_bstrPassword.Empty();
        m_bstrPassword.AppendBSTR(m_varData.bstrVal);
    }
    m_bstrInfo.Append("#");
    //获得字段名 Name 的值
```

```
pRs->GetFields()->GetItem("NAME")->get_Value(&m_varData);
if(m_varData.vt == VT_BSTR)
{
    m_bstrInfo.AppendBSTR(m_varData.bstrVal );
    m_bstrName.Empty();
    m_bstrName.AppendBSTR(m_varData.bstrVal);
}
m_bstrInfo.Append("#");
//获得字段 School 的值
pRs->GetFields()->GetItem("SCHOOL")->get_Value(&m_varData);
if(m_varData.vt == VT_BSTR)
{
    m_bstrInfo.AppendBSTR(m_varData.bstrVal );
    m_bstrSchool.Empty();
    m_bstrSchool.AppendBSTR(m_varData.bstrVal);
}
m_bstrInfo.Append("#");
//获得字段 DEPARTMENT 的值
pRs->GetFields()->GetItem("DEPARTMENT")->
get_Value(&m_varData);
if(m_varData.vt == VT_BSTR)
{
    m_bstrInfo.AppendBSTR(m_varData.bstrVal );
    m_bstrDepartment.Empty();
    m_bstrDepartment.AppendBSTR(m_varData.bstrVal);
}
m_bstrInfo.Append("#");
//获得学号字段值

if(x_nType==0)
{
    pRs->GetFields()->GetItem("STUDENTNC")->
    get_Value(&m_varData);
    if(m_varData.vt == VT_BSTR)
```

```
{
    m_bstrInfo.AppendBSTR(m_varData.bstrVal );
    m_bstrStudentNO.AppendBSTR(m_varData.bstrVal);
}
m_bstrInfo.Append("#");
//获得班级字段值
pRs->GetFields()->GetItem("CLASS")->
get_Value(&m_varData);
if(m_varData.vt == VT_BSTR)
{
    m_bstrInfo.AppendBSTR(m_varData.bstrVal );
    m_bstrClass.Empty();
    m_bstrClass.AppendBSTR(m_varData.bstrVal);
}
m_bstrInfo.Append("#");
}
if(x_nType==1)
{
    //获得 EMAIL 字段值
    pRs->GetFields()->GetItem("EMAIL")->
    get_Value(&m_varData);
    if(m_varData.vt == VT_BSTR)
    {
        m_bstrInfo.AppendBSTR(m_varData.bstrVal );
        m_bstrClass.Empty();
        m_bstrClass.AppendBSTR(m_varData.bstrVal);
    }
    m_bstrInfo.Append("#");
}

*x_UserInfo = m_bstrInfo;

pRs->Close();
pConn->Close();
```

```

        return S_OK;
    }

```

11.3 Exam 组件的实现

Exam 组件主要实现考试过程涉及的所有操作, 包括产生试卷、计算分数, 保存学生写的答案, 以及返回考试状态信息 (如考试剩余时间、是否完成等)。Exam 组件通过调用 QuestionzInfo 来实现它的功能的, 这是在介绍 Exam 组件时将重点介绍的地方。在 Exam 组件中实现了四个接口: IExamState, IGenerateExam, ICaculateScore, ISaveCurrentAnswer。IexamState 接口完成考试状态返回的功能; IgenerateExam 实现试题产生的功能; ICaculateScore 完成分数的计算; 而 ISaveCurrentAnswer 实现学生答题时, 答案的保存。

首先介绍 IexamState 的实现。IexamStat 包括了四个方法: CheckExist, CheckFinish, GetAnswer, GetCurrentQuestion, GetExamDetail, GetExamInfo。

CheckExist 方法检查某考生是否已参加考试。这主要是针对考生在做题过程中突然断电, 或出现其他故障时, 该考生可重新登录继续做题。该方法的输入参数是学号, 返回值表示该学生是否参与考试。代码如程序 11-3-1 所示:

程序 11-3-1:

```

STDMETHODIMP CExamState::CheckExist(int x_nSID, BSTR *x_Success)
{
    if (FAILED(pConn.CreateInstance(__uuidof(Connection))))
        //建立 Connection 对象
    {
        *x_Success = ::SysAllocString(L"Couldn't create connection
        component!");
        return S_OK;
    }

    if (FAILED(pConn->
    Open("Provider=MSDASQL.1;Password=;PersistSecurity Info=True;
    User ID=SA;Data Source=EXAM;Initial Catalog=EXAM", "", "", -1)))
        //打开 Connection 对象
    {

```

```
        *x_Success = ::SysAllocString(L"Couldn't  
open connection");  
        return S_OK;  
    }  
  
    CComBSTR m_bstrSql;  
  
    char m_strBuf[10];  
    _itoa(x_nSID, m_strBuf, 10);  
  
    m_bstrSql.Append("SELECT SID FROM TESTPROC WHERE SID='");  
    m_bstrSql.Append(m_strBuf);  
  
    _bstr_t m_bstrResult(m_bstrSql, FALSE);  
  
    CComVariant m_varNum;  
    //执行 SQL 语句  
    pRs = pConn->Execute(m_bstrResult, &m_varNum, -1);  
  
    if (m_varNum.lVal == 0)  
    {  
        *x_Success = ::SysAllocString(L"N");  
        return S_OK;  
    }  
  
    *x_Success = ::SysAllocString(L"Y");  
  
    pRs->Close();  
    pConn->Close();  
    return S_OK;  
}
```

CheckFinish 方法实现检查某学生是否完成了考试。它的输入参数是学生学

号, 输出参数是该学生是否完成考试的标志。代码如程序 11-3-2 所示:

程序 11-3-2:

```
STDMETHODIMP CExamState::CheckFinish(int x_nSID, BSTR *x_Success)
{
    // TODO: Add your implementation code here

    if (FAILED(pConn.CreateInstance(__uuidof(Connection))))
    {
        *x_Success = ::SysAllocString(L"Couldn't create connection
        component!");
        return S_OK;
    }

    if (FAILED(pConn->Open("Provider=MSDASQL.1;Password=;Persist
    Security Info=True;User ID=SA;Data Source=EXAM;Initial Catalog=EXAM",
    "", "", -1)))
    {
        *x_Success = ::SysAllocString(L"Couldn't
        open connection");
        return S_OK;
    }

    CComBSTR m_bstrSql;

    char m_strBuf[10];
    _itoa(x_nSID, m_strBuf, 10);
    //查询 Testproc 数据库表中是否有该考生的记录, FINISHFLAG 字符值是否是'y'
    m_bstrSql.Append("SELECT SID FROM TESTPROC WHERE SID=");
    m_bstrSql.Append(m_strBuf);
    m_bstrSql.Append(" AND FINISHFLAG = 'Y'");
}
```

```
_bstr_t m_bstrResult(m_bstrSql, FALSE);

CComVariant m_varNum;

pRs = pConn->Execute(m_bstrResult, &m_varNum, -1); //.....代码 1

if (m_varNum.lVal == 0)
{
    *x_Success = ::SysAllocString(L"N");
    return S_OK;
}

*x_Success = ::SysAllocString(L"Y");

pRs->Close();
pConn->Close();
return S_OK;

}
```

Connection 对象的 Execute 方法中, 第二个参数是返回执行 SQL 语句后受影响的记录数。所以在代码 1 中, 判断它的值是否为 0。如果为 0 说明没有符合条件的纪录, 也就是这个考生或者没参加考试或者考试没完成。如果不为 0, 说明考生完成考试了。

GetAnswer 是获得学生答题的答案。它的输入参数是学生 ID、题号, 返回值是学生做题的答案。答案的计算方式在前面数据库结构中已介绍, 在这不再重复。代码如程序 11-3-3 所示。

程序 11-3-3:

```
STDMETHODIMP CExamState::GetAnswer(int x_nSID, int x_nQID, long
*x_lAnswer)
{
    // TODO: Add your implementation code here
    try
    {
```

```

    if (FAILED(pConn.CreateInstance(__uuidof(Connection))))
    {
        *x_lAnswer = -1;
        return S_OK;
    }
    if (FAILED(pConn->Open("Provider=MSDASQL.1;Password=;
Persist Security Info=True;User ID=SA;Data Source=EXAM;
Initial Catalog=EXAM", "", "", -1)))
    {
        *x_lAnswer = -1;
        return S_OK;
    }
    char m_strBuf[10];
    _itoa(x_nSID, m_strBuf, 10);

    CComBSTR m_bstrSql;
    m_bstrSql.Append("SELECT * FROM TESTPROCDETAIL WHERE
SID=");
    m_bstrSql.Append(m_strBuf);
    m_bstrSql.Append(" and QNO=");
    _itoa(x_nQID, m_strBuf, 10);
    m_bstrSql.Append(m_strBuf);

    _bstr_t m_bstrResult(m_bstrSql, FALSE);

    CComVariant m_varNum;

    pRs = pConn->Execute(m_bstrResult, &m_varNum, -1);

    if (pRs->ADOEOF)
    {
        *x_lAnswer = -1;
        return S_OK;
    }

```

```
CComVariant m_varData;

char m_strFieldName[20];
_itoa(x_nQID, m_strBuf, 10);
strcpy(m_strFieldName, "A");
strcpy(m_strFieldName, m_strBuf);

pRs->GetFields()->GetItem(m_strFieldName)->
get_Value(&m_varData);
if(m_varData.vt != VT_NULL)
{
    *x_lAnswer = m_varData.iVal;
}

pRs->Close();
pConn->Close();

}
catch(...)
{
    *x_lAnswer = -1;
    return S_OK;
}

return S_OK;
}
```

GetCurrentQuestion 主要返回考生当前正在做的题的题目号。代码如程序 11-3-4 所示:

程序 11-3-4:

```
STDMETHODIMP CExamState::GetCurrentQuestion(int x_nSID, long
*x_lCurrentQuestion)
```

//x_nSID 表示学生 ID, x_lCurrentQuestion 用于保存返回的当前题目号

```
{
    // TODO: Add your implementation code here
    try
    {

        if
(FAILED(pConn.CreateInstance(__uuidof(Connection))))
        {
            *x_lCurrentQuestion = -1;
            return S_OK;
        }

        if (FAILED(pConn->Open("Provider=MSDASQL.1;
Password=;Persist Security Info=True;User ID=SA;
Data Source=EXAM;Initial Catalog=EXAM", "", "", -1)))
        {
            *x_lCurrentQuestion = -1;
            return S_OK;
        }
        //构造 SQL 语句
        char m_strBuf[10];
        _itoa(x_nSID, m_strBuf, 10);

        CComBSTR m_bstrSql;
        m_bstrSql.Append("SELECT * FROM TESTPROC WHERE SID=");
        m_bstrSql.Append(m_strBuf);

        _bstr_t m_bstrResult(m_bstrSql, FALSE);

        CComVariant m_varNum;

        pRs = pConn->Execute(m_bstrResult, &m_varNum, -1);
    }
}
```