**1、简述 private、 protected、 public、 internal 修饰符的访问权限。**

private : 私有成员, 在类的内部才可以访问。

protected : 保护成员，该类内部和继承类中可以访问。

public : 公共成员，完全公开，没有访问限制。

internal: 当前程序集内可以访问。

**2、ADO.NET中的五个主要对象**

Connection：主要是开启程序和数据库之间的连接。没有利用连接对象将数据库打开，是无法从数据库中取得数据的。Close和Dispose的区别，Close以后还可以Open，Dispose以后则不能再用。

Command：主要可以用来对数据库发出一些指令，例如可以对数据库下达查询、新增、修改、删除数据等指令，以及调用存在数据库中的存储过程等。这个对象是架构在Connection 对象上，也就是Command 对象是透过连接到数据源。

DataAdapter：主要是在数据源以及DataSet 之间执行数据传输的工作，它可以透过Command 对象下达命令后，并将取得的数据放入DataSet 对象中。这个对象是架构在Command对象上，并提供了许多配合DataSet 使用的功能。

DataSet：这个对象可以视为一个暂存区（Cache），可以把从数据库中所查询到的数据保留起来，甚至可以将整个数据库显示出来，DataSet是放在内存中的。DataSet 的能力不只是可以储存多个Table 而已，还可以透过DataAdapter对象取得一些例如主键等的数据表结构，并可以记录数据表间的关联。DataSet 对象可以说是ADO.NET 中重量级的对象，这个对象架构在DataAdapter对象上，本身不具备和数据源沟通的能力；也就是说我们是将DataAdapter对象当做DataSet 对象以及数据源间传输数据的桥梁。DataSet包含若干DataTable、DataTableTable包含若干DataRow。

DataReader：当我们只需要循序的读取数据而不需要其它操作时，可以使用DataReader 对象。DataReader对象只是一次一笔向下循序的读取数据源中的数据，这些数据是存在数据库服务器中的，而不是一次性加载到程序的内存中的，只能（通过游标）读取当前行的数据，而且这些数据是只读的，并不允许作其它的操作。因为DataReader 在读取数据的时候限制了每次只读取一笔，而且只能只读，所以使用起来不但节省资源而且效率很好。使用DataReader 对象除了效率较好之外，因为不用把数据全部传回，故可以降低网络的负载。

ADO.NET 使用Connection 对象来连接数据库，使用Command 或DataAdapter对象来执行SQL语句，并将执行的结果返回给DataReader 或 DataAdapter ,然后再使用取得的DataReader 或DataAdapter 对象操作数据结果。

**3、列举ASP.NET 页面之间传递值的几种方式。**

1.使用QueryString, 如....?id=1; response. Redirect()....

2.使用Session变量

3.使用Server.Transfer

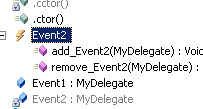
4.Cookie传值

**4、C#中的委托是什么？事件是不是一种委托？事件和委托的关系。**

委托可以把一个方法作为参数代入另一个方法。

委托可以理解为指向一个函数的指针。

委托和事件没有可比性，因为委托是类型，事件是对象，下面说的是委托的对象（用委托方式实现的事件）和（标准的event方式实现）事件的区别。事件的内部是用委托实现的。因为对于事件来讲，外部只能“注册自己+=、注销自己-=”，外界不可以注销其他的注册者，外界不可以主动触发事件，因此如果用Delegate就没法进行上面的控制，因此诞生了事件这种语法。事件是用来阉割委托实例的，类比用一个自定义类阉割List。事件只能add、remove自己，不能赋值。事件只能+=、-=，不能= 。加分的补充回答：事件内部就是一个private的委托和add、remove两个方法



面试聊：用Reflector查看.Net的类的内部实现，解决问题。

**5、override与重载（overload）的区别**

重载是方法的名称相同。参数或参数类型不同，进行多次重载以适应不同的需要。重载（overload）是面向过程的概念。

Override 是进行基类中函数的重写。Override是面向对象的概念

**6、C#中索引器是否只能根据数字进行索引？是否允许多个索引器参数？**

参数的个数和类型都是任意的。加分的补充回答：用reflector反编译可以看出，索引器的内部本质上就是set\_item、get\_item方法。

基础知识：

索引的语法：

public string this[string s]，通过get、set块来定义取值、赋值的逻辑

索引可以有多个参数、参数类型任意

索引可以重载。

如果只有get没有set就是只读的索引。

索引其实就是set\_Item、get\_Item两个方法。

**7、属性和public字段的区别是什么？调用set方法为一个属性设值，然后用get方法读取出来的值一定是set进去的值吗？**

属性可以对设值、取值的过程进行非法值控制，比如年龄禁止设值负数，而字段则不能进行这样的设置。虽然一般情况下get读取的值就是set设置的值，但是可以让get读取的值不是set设置的值的，极端的例子。Public Age{get{return 100;}set{}}。加分的补充回答：用reflector反编译可以看出，属性内部本质上就是set\_\*\*\*、get\_\*\*\*方法

class Person

{

public int Age

{

get

{

return 3;

}

set

{

}

}

}

Person p1 = new Person();

p1.Age = 30;

p1.Age++;

Console.Write(p1.Age);//输出3

必须手写掌握的代码（既包含拿电脑写，拿笔写）：

1. 手写三层架构
2. 手写冒泡排序
3. 手写AJAX：XMLHttpRequest
4. 手写增删改查、SQLHelper

**8、三层架构**

通常意义上的三层架构就是将整个业务应用划分为：表现层（UI）、业务逻辑层（BLL）、数据访问层（DAL）。区分层次的目的即为了“高内聚，低耦合”的思想。表现层（UI）：通俗讲就是展现给用户的界面，即用户在使用一个系统的时候的所见所得。业务逻辑层（BLL）：针对具体问题的操作，也可以说是对数据层的操作，对数据业务逻辑处理。数据访问层（DAL）：该层所做事务直接操作数据库，针对数据的增添、删除、修改、更新、查找等每层之间是一种垂直的关系。三层结构是N层结构的一种，一般来说，层次之间是向下依赖的，下层代码未确定其接口（契约）前，上层代码是无法开发的，下层代码接口（契约）的变化将使上层的代码一起变化。

优点： 分工明确，条理清晰，易于调试，而且具有可扩展性。

缺点： 增加成本。

**9、什么是装箱(boxing)和拆箱(unboxing)？ （\*）**

Object是引用类型，但是它的子类Int32竟然不能去Object能去的“要求必须是引用类型”

的地方，违反了继承的原则，所以需要把Int32装在Object中才能传递。

装箱：从值类型接口转换到引用类型。

拆箱：从引用类型转换到值类型。

object obj = null;//引用类型

obj = 1;//装箱，boxing。把值类型包装为引用类型。

int i1 = (int)obj;//拆箱。unboxing

2）下面三句代码有没有错，以inboxing或者unboxing为例，解释一下内存是怎么变化的

int i=10;

object obj = i;

int j = obj;

分析：在inboxing（装箱）时是不需要显式的类型转换的，不过unboxing（拆箱）需要显式的类型转换，所以第三行代码应该改为：

3 int j = (int)obj;

要掌握装箱与拆箱，就必须了解CTS及它的特点：

NET重要技术和基础之一的CTS(Common Type System)。CTS是为了实现在应用程序声明和使用这些类型时必须遵循的规则而存在的通用类型系统。.Net将整个系统的类型分成两大类 ：值类型和引用类型。

CTS中的所有东西都是对象；所有的对象都源自一个基类——System.Object类型。值类型的一个最大的特点是它们不能为null，值类型的变量总有一个值。为了解决值类型不可以为null，引用类型可以为null的问题，微软在.Net中引入了装箱和拆箱：装箱就是将值类型用引用类型包装起来转换为引用类型；而从引用类型中拿到被包装的值类型数据进行拆箱。

**10、CTS、CLS、CLR分别作何解释（\*）把英文全称背过来。**

C#和.Net的关系。

C#只是抽象的语言，可以把C#编译生成Java平台的二进制代码，也可以把Java代码编译生成.Net平台的二进制代码。所以C#只是提供了if、while、+-\*/、定义类、int、string等基础的语法，而Convert.ToInt32、FileStream、SqlConnection、String.Split等都属于.Net的东西。深蓝色是C#的，浅蓝色是.Net的。

C# new→IL:newobj

C# string →.Net中的String

类型的差别：.net中的Int32在C#中是int，在VB.Net中是Integer。String、Int32等公共类型。

语法的差别：IL中创建一个对象的方法是L\_0001: newobj instance void 索引.C1::.ctor()

C#中是new C1();VB.net中是 Dim c1 As New C1

CTS：Common Type System 通用类型系统。Int32、Int16→int、String→string、Boolean→bool。每种语言都定义了自己的类型，.Net通过CTS提供了公共的类型，然后翻译生成对应的.Net类型。

CLS：Common Language Specification 通用语言规范。不同语言语法的不同。每种语言都有自己的语法，.Net通过CLS提供了公共的语法，然后不同语言翻译生成对应的.Net语法。

CLR：Common Language Runtime 公共语言运行时，就是GC、JIT等这些。有不同的CLR，比如服务器CLR、Linux CLR（Mono）、Silverlight CLR(CoreCLR)。相当于一个发动机，负责执行IL。

**11、在dotnet中类(class)与结构(struct)的异同？**

Class可以被实例化,属于引用类型,是分配在内存的堆上的。类是引用传递的。

Struct属于值类型,是分配在内存的栈上的。结构体是复制传递的。加分的回答：Int32、Boolean等都属于结构体

**12、堆和栈的区别？**

栈是编译期间就分配好的内存空间，因此你的代码中必须就栈的大小有明确的定义；局部值类型变量、值类型参数等都在栈内存中。

堆是程序运行期间动态分配的内存空间，你可以根据程序的运行情况确定要分配的堆内存的大小。

**13、能用foreach遍历访问的对象的要求**

需要实现IEnumerable接口或声明GetEnumerator方法的类型。

**14、GC是什么? 为什么要有GC?**

C/C++中由程序员进行对象的回收像学校食堂中由学生收盘子，.Net中由GC进行垃圾回收像餐馆中店员去回收。

GC是垃圾收集器（Garbage Collection）。程序员不用担心内存管理，因为垃圾收集器会自动进行管理。GC只能处理托管内存资源的释放，对于非托管资源则不能使用GC进行回收，必须由程序员手工回收，一个例子就是FileStream或者SqlConnection需要程序员调用Dispose进行资源的回收。

要请求垃圾收集，可以调用下面的方法：GC.Collect()一般不需要手动调用GC.Collect()。当一个对象没有任何变量指向（不再能使用）的时候就**可以**被回收了。

基础知识：当没有任何变量指向一个对象的时候对象就可以被回收掉了，但不一定会立即被回收。

object obj = new object();//只有new才会有新对象

Console.WriteLine(obj);

object obj2 = obj;

obj = null;

obj2 = null;

Console.WriteLine();

**15、值类型和引用类型的区别？**

1.将一个值类型变量赋给另一个值类型变量时，将复制包含的值。引用类型变量的赋值只复制对对象的引用，而不复制对象本身。

2.值类型不可能派生出新的类型：所有的值类型均隐式派生自 System.ValueType。但与引用类型相同的是，结构也可以实现接口。

3.值类型不可能包含 null 值：然而，可空类型功能允许将 null 赋给值类型。

4.每种值类型均有一个隐式的默认构造函数来初始化该类型的默认值。

**16、C#中的接口和类有什么异同。**

不同点：

不能直接实例化接口。

接口不包含方法的实现。

接口可以多继承，类只能单继承。

类定义可在不同的源文件之间进行拆分。

相同点：

接口、类和结构都可以从多个接口继承。

接口类似于抽象基类：继承接口的任何非抽象类型都必须实现接口的所有成员。

接口和类都可以包含事件、索引器、方法和属性。

基础知识：接口只能定义方法（只能定义行为，不能定义实现也就是字段），因为事件、索引器、属性本质上都是方法，所以接口中也可以定义事件、索引器、属性。

**17、abstract class和interface有什么区别?**

相同点：

都不能被直接实例化，都可以通过继承实现其抽象方法。

不同点：

接口支持多继承；抽象类不能实现多继承。

接口只能定义行为；抽象类既可以定义行为，还可能提供实现。

接口只包含方法(Method)、属性(Property)、索引器(Index)、事件(Event)的签名，但不能定义字段和包含实现的方法；

抽象类可以定义字段、属性、包含有实现的方法。

接口可以作用于值类型(Struct)和引用类型(Class)；抽象类只能作用于引用类型。例如，Struct就可以继承接口，而不能继承类。

加分的补充回答：讲设计模式的时候SettingsProvider的例子。

**18、是否可以继承String类?**

String类是sealed类故不可以继承。

**19、int、DateTime、string是否可以为null?**

null表示“不知道”，而不是“没有”。没有年龄和不知道年龄是不一样。

数据库中null不能用0表示。0岁和不知道多少岁不一样。

Sex is zero。《色即是空》

//int i1 = null;

//int? i2 = null;//值类型后加?就成了可空数据类型

////int i3 = i2;//所以把可空的赋值给一定不能为空的会报错。

//int i4 = (int)i2;//可以显式转换，由程序员来保证“一定不为空”

//int? i5 = i4;//一定会成功！

//using()→try{]catch{}finally{}

//int?是微软的一个语法糖。是一种和int没有直接关系的Nullable类型

Nullable<int> d1 = new Nullable<int>();//int? d1=null;

Nullable<int> d2 = new Nullable<int>(3);//int? d2=3;

Console.WriteLine(d1==null);

int、DateTime不能，因为其为Struct类型，而结构属于值类型，值类型不能为null,只有引用类型才能被赋值null。string可以为null。

C#中int等值类型不可以为null、而数据库中的int可以为null，这就是纠结的地方。int?就变成了可空的int类型。bool?、DateTime?

int i1 = 3;

int? i2 = i1;

//int i3 = i2;//不能把可能为空的赋值给一定不能为空的变量

int i3 = (int)i2;//显式转换

可空数据类型经典应用场景：三层中的Model属性的设计。

int?翻译生成.Net的Nullable<int>，CTS。

**20、using关键字有什么用？什么是IDisposable？**

using可以声明namespace的引入，还可以实现非托管资源的释放，实现了IDisposiable的类在using中创建，using结束后会自动调用该对象的Dispose方法，释放资源。加分的补充回答：using其实等价于try……finally，用起来更方便。

**21、XML 与 HTML 的主要区别**

1. XML是区分大小写字母的，HTML不区分。

2. 在HTML中，如果上下文清楚地显示出段落或者列表键在何处结尾，那么你可以省略</p>或者</li>之类的结束 标记。在XML中，绝对不能省略掉结束标记。

HTML：<img src="1.jpg"><br><br>

XML：<img src="1.jpg"></img><br/><br/>

3. 在XML中，拥有单个标记而没有匹配的结束标记的元素必须用一个 / 字符作为结尾。这样分析器就知道不用 查找结束标记了。

4. 在XML中，属性值必须分装在引号中。在HTML中，引号是可用可不用的。

5. 在HTML中，可以拥有不带值的属性名。在XML中，所有的属性都必须带有相应的值。

XML是用来存储和传输数据的

HTML是用来显示数据的

如果使用了完全符合XML语法要求的HTML，那么就叫做符合XHTML标准。符合XHTML标准的页面有利于SEO。

**22、string str = null 与 string str =””说明其中的区别。**

答：string str = null 是不给他分配内存空间,而string str = \"\" 给它分配长度为空字符串的内存空间。 string str = null没有string对象，**string str = “”有一个字符串对象。**

string s3 = string.Empty;//反编译发现，string.Empty就是在类构造函数中 Empty = "";

**23、写出一条Sql语句：取出表A中第31到第40记录（SQLServer,以自动增长的ID作为主键,注意：ID可能不是连续的。**

答：解1: select top 10 \* from A where id not in (select top 30 id from A)

演变步骤：

1）select top 30 id from T\_FilterWords--取前条

2）select \* from T\_FilterWords

where id not in (select top 30 id from T\_FilterWords)--取id不等于前三十条的

--也就是把前条排除在外

3）select top 10 \* from T\_FilterWords

where id not in (select top 30 id from T\_FilterWords)

--取把前条排除在外的前条，也就是-40条

解2: select top 10 \* from A where id > (select max(id) from (select top 30 id from A )as A)

解答3：用ROW\_NUMBER实现

**24、在.Net中所有可序列化的类都被标记为**

[serializable]

**25、什么是code-Behind技术。**

就是代码隐藏，在ASP.NET中通过ASPX页面指向CS文件的方法实现显示逻辑和处理逻辑的分离，这样有助于web应用程序的创建。比如分工，美工和编程的可以个干各的，不用再像以前asp那样都代码和html代码混在一起，难以维护。**code-Behind是基于部分类技术实现的，在我的项目的三层代码生成器中用到了部分类。**

**26、接口是一种引用类型，在接口中可以声明（   a），但不可以声明公有的域或私有的成员变量。**   
  a) 方法、属性、索引器和事件；

  b) 索引器和字段；       
  c) 事件和字段；

解读：接口中不能声明字段只能声明方法，属性、索引器和事件 最终都编译生成方法。因为字段属于实现层面的东西，只有存取值的时候才会用到字段，所以中接口中不能定义字段。

**27、在ADO.NET中，对于Command对象的ExecuteNonQuery()方法和ExecuteReader()方法，下面叙述错误的是（c）。**   
  a) insert、update、delete等操作的Sql语句主要用ExecuteNonQuery()方法来执行；     
  b) ExecuteNonQuery()方法返回执行Sql语句所影响的行数。     
  c) Select操作的Sql语句只能由ExecuteReader()方法来执行；     
  d) ExecuteReader()方法返回一个DataReader对象；

拿SQLHelper实现一下。

**28、StringBuilder 和 String 的区别？**

答：String 在进行运算时（如赋值、拼接等）会产生一个新的实例，而 StringBuilder 则不会。所以在大量字符串拼接或频繁对某一字符串进行操作时最好使用 StringBuilder，不要使用 String

 如果要操作一个不断增长的字符串，尽量不用String类,改用StringBuilder类。两个类的工作原理不同:String类是一种传统的修改字符串的方式，它确实可以完成把一个字符串添加到另一个字符串上的工作没错,但是在.NET框架下，这个操作实在是划不来。因为系统先是把两个字符串写入内存，接着删除原来的String对象，然后创建一个String对象，并读取内存中的数据赋给该对象。这一来二去的，耗了不少时间。而使用System.Text命名空间下面的StringBuilder类就不是这样了，它提供的Append方法，能够在已有对象的原地进行字符串的修改，简单而且直接。当然，一般情况下觉察不到这二者效率的差异，但如果你要对某个字符串进行大量的添加操作，那么StringBuilder类所耗费的时间和String类简直不是一个数量级的。

**29、请叙述属性与索引器的区别。 （\*）**

属性 索引器

通过名称标识。 通过签名标识。

通过简单名称或成员访问来访问。 通过元素访问来访问。

可以为静态成员或实例成员。 必须为实例成员。

属性的   get   访问器没有参数。 索引器的   get   访问器具有与索引器相同的形参表。

属性的   set   访问器包含隐式   value   参数。 除了   value   参数外，索引器的 set 访问器还具有与索引器相同的形参表。

**30、请解释ASP。NET中的web页面与其隐藏类之间的关系？**

一个ASP.NET页面一般都对应一个隐藏类,一般都在ASP.NET页面的声明中指定了隐藏类例如一个页面Tst1.aspx的页面声明如下

<%@ Page language="c#" Codebehind="Tst1.aspx.cs" AutoEventWireup="false" Inherits="T1.Tst1" %>

Codebehind="Tst1.aspx.cs" 表明经编译此页面时使用哪一个代码文件

Inherits="T1.Tst1" 表用运行时使用哪一个隐藏类

aspx页面会编译生成一个类，这个类从隐藏类继承。

**31、您在什么情况下会用到虚方法？它与接口有什么不同？**

答案：子类重新定义父类的某一个方法时，必须把父类的方法定义为virtual

在定义接口中不能有方法体，虚方法可以。

实现时,子类可以不重新定义虚方法，但如果一个类继承接口，那必须实现这个接口。

**32、DataReader和DataSet的异同？**

DataReader使用时始终占用SqlConnection,在线操作数据库

每次只在内存中加载一条数据,所以占用的内存是很小的

是只进的、 只读的

DataSet则是将数据一次性加载在内存中.抛弃数据库连接..读取完毕即放弃数据库连接(非连接模式)

DataSet将数据全部加载在内存中.所以比较消耗内存...但是确比DataReader要灵活..可以动态的添加行,列,数据.对数据库进行 回传更新操作(动态操作读入到内存的数据)

**33、public static const int A = 1;这段代码有错误么？**

错误：const不能被修饰为static ；因为定义为常量 （const ）后就是静态的（static ）。

**34、传入某个属性的set方法的隐含参数的名称是什么？**

value，它的类型和属性所声名的类型相同。

**35、C#支持多重继承么？**

类之间不支持，接口之间支持。类对接口叫做实现，不叫继承。 类是爹、接口是能力，能有多个能力，但不能有多个爹。

**36、C#中所有对象共同的基类是什么？**

System.Object

**37、通过超链接怎样传递中文参数？**

答：用URL编码，通过QueryString传递，用urlencode编码 用urldecode解码

**38、string、String；int、Int32；Boolean、bool的区别**

String、Int32、Boolean等都属于.Net中定义的类，而string、int、bool相当于C#中对这些类定义的别名。CTS。

**39、Server.Transfer和Response.Redirect的区别是什么？（常考）**

答:Server.Transfer仅是服务器中控制权的转向，在客户端浏览器地址栏中不会显示出转向后的地址；Response.Redirect则是完全的跳转，浏览器将会得到跳转的地址，并重新发送请求链接。这样，从浏览器的地址栏中可以看到跳转后的链接地址。

Server.Transfer是服务器请求资源，服务器直接访问目标地址的URL，把那个URL的响应内容读取过来，然后把这些内容再发给浏览器，浏览器根本不知道服务器发送的内容是从哪儿来的，所以它的地址栏中还是原来的地址。 这个过程中浏览器和Web服务器之间经过了一次交互。

Response.Redirect就是服务端根据逻辑,发送一个状态码,告诉浏览器重新去请求那个地址，一般来说浏览器会用刚才请求的所有参数重新请求。这个过程中浏览器和Web服务器之间经过了两次交互。

Server.Transfer不可以转向外部网站，而**Response.Redirect可以。**

Server.Execute效果和Server.Transfer类似，但是是把执行的结果嵌入当前页面。

**40、不是说字符串是不可变的吗？string s="abc";s="123"不就是变了吗？**

String是不可变的在这段代码中，s原先指向一个String对象，内容是 "abc"，然后我们将s指向"123"，那么s所指向的那个对象是否发生了改变呢？答案是没有。这时，s不指向原来那个对象了，而指向了另一个 String对象，内容为"123"，原来那个对象还存在于内存之中，只是s这个引用变量不再指向它了。

**41、是否可以从一个static方法内部发出对非static方法的调用？**

不可以。因为非static方法是要与对象关联在一起的，必须创建一个对象后，才可以在该对象上进行方法调用，而static方法调用时不需要创建对象，可以直接调用。也就是说，当一个static方法被调用时，可能还没有创建任何实例对象，如果从一个static方法中发出对非static方法的调用，那个非static方法是关联到哪个对象上的呢？这个逻辑无法成立，所以，一个static方法内部不能发出对非static方法的调用。

**42、说出一些常用的类、接口，请各举5个**

要让人家感觉你对.Net开发很熟，所以，不能仅仅只列谁都能想到的那些东西，要多列你在做项目中涉及的那些东西。就写你最近写的那些程序中涉及的那些类。

常用的类：StreamReader、WebClient、Dictionary<K,V>、StringBuilder、SqlConnection、FileStream、File、Regex、List<T>

常用的接口：IDisposable、IEnumerable、IDbConnection、IComparable、ICollection、IList、IDictionary

要出乎意料！不要仅仅完成任务！笔试不是高考！处处要显出牛！

说出几个开源软件？MySQL、Linux、 Discuz、Apache、Paint.Net、Android、Chrome、Notepad++……

开源项目有一些是开发包。开源软件指的是可以直接用的。Jquery、NPOI、ASP.Net MVC、Silverlight Toolkit、AJAX toolkit、json.net

**43、编写一个单例（Singleton）类。**

把构造函数设置为private，设置一个public、static的对象实例

public FileManager

{

private FileManager(){}

public readonly static FileManager Instance = new FileManager();

}

扩展：搜“C# Singleton”，有线程安全的更牛B的实现

**44、什么是sql注入？如何避免sql注入？**

用户根据系统的程序构造非法的参数从而导致程序执行不是程序员期望的恶意SQL语句。使用参数化的SQL就可以避免SQL注入。

详细参考复习ppt。举例子，摆事实！

1' or 1=1

**45、数据库三范式是什么?**

**用自己的话解释，而不是背概念。**

第一范式：字段不能有冗余信息，所有字段都是必不可少的。

第二范式：满足第一范式并且表必须有主键。

第三范式：满足第二范式并且表引用其他的表必须通过主键引用。

员工内部→自己的老大→外部的老大

记忆顺序：自己内部不重复→别人引用自己→自己引用别人。

**46、post、get的区别**

get的参数会显示在浏览器地址栏中，而post的参数不会显示在浏览器地址栏中；

使用post提交的页面在点击【刷新】按钮的时候浏览器一般会提示“是否重新提交”，而get则不会；

用get的页面可以被搜索引擎抓取，而用post的则不可以；

用post可以提交的数据量非常大，而用get可以提交的数据量则非常小(2k)，受限于网页地址的长度。

用post可以进行文件的提交，而用get则不可以。

参考阅读：http://www.cnblogs.com/skynet/archive/2010/05/18/1738301.html

**47、.Net、ASP.Net、C#、VisualStudio之间的关系是什么？**

答：.Net一般指的是.Net Framework，提供了基础的.Net类，这些类可以被任何一种.Net编程语言调用，.Net Framework还提供了CLR、JIT、GC等基础功能。

ASP.Net是.Net中用来进行Web开发的一种技术，ASP.Net的页面部分写在aspx 文件中，逻辑代码通常通过Code-behind的方式用C#、VB.Net等支持.Net的语言编写。

C#是使用最广泛的支持.Net的编程语言。除了C#还有VB.Net、IronPython等。

VisualStudio是微软提供的用来进行.Net开发的集成开发环境（IDE），使用VisualStudio可以简化很多工作，不用程序员直接调用csc.exe等命令行进行程序的编译，而且VisualStudio提供了代码自动完成、代码高亮等功能方便开发。除了VisualStudio，还有SharpDevelop、MonoDevelop等免费、开源的IDE，VisualStudio Express版这个免费版本。

**48、AJAX解决什么问题？如何使用AJAX？AJAX有什么问题需要注意？项目中哪里用到了AJAX？**

答：AJAX解决的问题就是“无刷新更新页面”，用传统的HTML表单方式进行页面的更新时，每次都要将请求提交到服务器，服务器返回后再重绘界面，这样界面就会经历：提交→变白→重新显示这样一个过程，用户体验非常差，使用AJAX则不会导致页面重新提交、刷新。

AJAX最本质的实现是在Javascript中使用XMLHttpRequest进行Http的请求，开发中通常使用UpdatePanel、JQuery等方式简化AJAX的开发，UpdatePanel的方式实现AJAX最简单，但是数据通讯量比较大，因为要来回传整个ViewState，而且不灵活，对于复杂的需求则可以使用JQuery提供的ajax功能。

UpdatePanel的内部原理。

AJAX最重要的问题是无法跨域请求(www.rupeng.com →so.rupeng.com)，也就是无法在页面中向和当前域名不同的页面发送请求，可以使用在当前页面所在的域的服务端做代理页面的方式解决。

在如鹏网项目中发帖的时候显示相关帖的功能、站内搜索项目中显示搜索Suggestion、数据采集项目中都用到了AJAX。

常考：不用任何框架编写一个AJAX程序。XHR:XmlHttpRequest。背也要背下来！

如果面试的时候谈AJAX谈到UpdatePanel的时候，就是NB的时候！！！先侃UpdatePanel的原理！引出为什么Dom操作的动态效果在用UpdatePanel提交刷新以后没有了，以及CKEditor被套在UpdatePanel中提交以后也变成了textarea，为什么把Fileupload放到Updatepanel中无法实现无刷新上传。说成是公司内部的一个菜鸟用UpdatePanel遇到这样问题，由于我懂XHR、UpdatePanel的原理，所以轻松解决！UpdatePanel生成的上万行JS脚本，不适合于互联网项目。“WebForm怎么可能把开发人员编程傻子呢！不明白原理苦命呀！还是MVC好呀，MVC。。。。。。。”

**49、Application 、Cookie和 Session 两种会话有什么不同？**

答：Application是用来存取整个网站全局的信息，而Session是用来存取与具体某个访问者关联的信息。Cookie是保存在客户端的，机密信息不能保存在Cookie中，只能放小数据；Session是保存在服务器端的，比较安全，可以放大数据。

谈到Session的时候就侃Session和Cookie的关系：Cookie中的SessionId。和别人对比说自己懂这个原理而给工作带来的方便之处。

**50、开放式问题：你经常访问的技术类的网站是什么？**

博客园（www.cnblogs.com）、csdn、codeplex、codeproject、msdn文档、msdn论坛（遇到问题先到网上搜解决方案，还不行就问同事，同事也解决不了就去MSDN论坛提问，一定能得到解决）。Cnbeta.com。

**51、你对.net的GC的理解**

GC是.Net的垃圾收集器，可以进行内存资源的回收，程序员无需关心资源的回收，当一个对象没有任何引用的时候就可以被回收了。一个对象可以被回收并不意味着一定会被立即回收，GC会选择时机进行回收。可以调用GC.Collect()让GC立即回收。GC不能回收非托管资源，对于非托管资源一般都实现了IDisposable接口，然后使用using关键字进行资源的回收。

**52、请你简单的说明数据库建立索引的优缺点**

使用索引可以加快数据的查询速度，不过由于数据插入过程中会建索引，所以会降低数据的插入、更新速度，索引还会占磁盘空间。

**53、不用JQuery等框架实现AJAX无刷新登录。（常考）**

提示：用纯XMLHttpRequest实现。

**54、Http状态码各是什么意思。**

301（永久重定向）、302 ：重定向  
404 ：页面不存在

500：服务器内部错误

**55、Session有什么重大BUG，微软提出了什么方法加以解决？（常考）**

iis中由于有进程回收机制，系统繁忙的话Session会丢失，IIS重启也会造成Session丢失。这样用户就要重新登录或者重新添加购物车、验证码等放到Session中的信息。可以用State server或SQL Server数据库的方式存储Session不过这种方式比较慢，而且无法捕获Session的END事件。但是（一定要写上，这样才够拽）我认为这不是Bug，只能说是In-Proc方式存储Session的缺陷，缺陷是和Bug不一样的，In-Proc方式存储Session会由服务器来决定什么时候释放Session，这是By Design，In-Proc方式不满足要求的话完全可以用StateServer和数据库的方式。

StateServer还可以解决集群Session共享的问题。

**56、asp.net中<%%>、<%=%>、<%#%>的区别是什么**

<%%>是执行<%%>中的C#代码，<%=%>是将=后表达式的值输出到Response中，<%#%>是数据绑定，一般用在ListView、GridView、Repeater等控件的绑定中。数据绑定分为：Eval：单向绑定和Bind：双向绑定。

**57、说说在软件设计中你遇到的以空间换时间的例子？**

参考回答：谈Cache(缓存)、索引这些例子。用额外的磁盘、内存空间的消耗来提高执行速度。Lucene.Net也是一个例子。

**58、asp.net中的错误机制。（常考）**

参考：

定制错误页来将显示一个友好的报错页面。

页面中未捕获一样会触发Ｐａｇｅ\_Error（不常用），应用程序中的未捕获异常会触发Application\_Error。通过HttpContext.Current.Server.GetLastError()、HttpContext.Current.Error;拿到未捕捉异常，记录到Log4Net日志中**。**

**59、介绍几个使用过的开源的项目**

Lucene.net、NPOI、JQuery、ASP.Net AJAX toolkit、Quartz.Net、JqueryUI、ASP.Net MVC。在CodePlex、SourceForge等网站上有更多的开源项目

**60、说出五个集合类**

List、Dictionary、Set、Stack（先入后出）、Queue（先入先出）、Tree等。

**61、有一个10个数的数组，计算其中不重复数字的个数。{3,5,9,8,10,5,3}**

**工程化的非最优解答：用HashSet或者List**

int[] values = { 3, 5, 9, 8, 10, 5, 3 };

HashSet<int> set = new HashSet<int>();

foreach (int i in values)

{

set.Add(i);

}

Console.WriteLine(set.Count);

**62、说一下SQLServer中索引的两种类型（常考！！！）**

参考：聚簇(或者叫做聚集，cluster)索引和非聚簇索引。

字典的拼音目录就是聚簇(cluster)索引，笔画目录就是非聚簇索引。这样查询“G到M的汉字”就非常快，而查询“6划到8划的字”则慢。

聚簇索引是一种特殊索引，它使数据按照索引的排序顺序存放表中。聚簇索引类似于字典，即所有词条在字典中都以字母顺序排列。聚簇索引实际上重组了表中的数据，所以你只能在表中建立一个聚簇索引。  
当数据按值的范围查询时，聚簇索引就显得特别有用。因为所有SQLServer都必需先找到所查询范围的第一行，然后依次下去，直到该范围的最后一个值找到为止，并且保证了所有其他值也落在这个范围内。举一个例子，一个应用程序要查找首字母位于G和P之间的姓名列表，SQLServer首先找到以字母G开头的名字，取出所有记录，直到找到以字母P开头的名字为止，这种方法使得查询过程非常高效。

进行大量数据改动的表不适宜用聚簇索引，因为SQLServer将不得不在表中维护行的次序。如果要索引的值极少，例如一个列包含的全都是1和0，创建聚簇索引就不是个好主意。如果表经常由一个指定的列来排序，该列将是簇索引的最佳候选列。这是因为表中的数据已经为你排好序了。如果访问一个表并使用BETWEEN、<、>、>=或<=操作符来返回一个范围的值时，应该考虑使用聚簇索引。

**63、能否脱离VS用类似editplus的工具手写代码？你觉得如何才能提高代码的效率和性能？**

可以，使用CSC.exe 来编译.cs文件！

可以根据业务流程、业务数据的特点进行优化，比如可以采用缓存、索引、表驱动等来提升代码的效率和性能，同时不要进行无意义的代码优化，重点优化系统的性能瓶颈。

**64、如何提高页面的显示速度？假如一个页面的加载时间是10.89s，你会用什么样的方式进行优化？**

回答要大气！显得很牛！

提示：分而治之的思想。

首先要找出问题所在，是服务器端运行的速度慢还是服务器端到客户端的下载慢还是页面在浏览器的加载速度慢。

如果是服务器端运行速度慢，则找是数据库的原因还是算法的问题，如果是数据库的问题则尝试添加索引、优化SQL语句，如果是算法的问题，则优化算法。如果对于一些不经常改动的页面可以使用静态页技术！对于一些数据不需要及时更新的而且取数据的过程非常耗时可以使用缓存。页面中的内容可以按需加载（比如说可以像verycd网站的评论那样当用户需要看的时候再加载其内容），可以在图片需要显示的时候再进行加载。

如果是服务器端到客户端的下载慢则看是页面体积过于臃肿还是网络问题，如果是页面体积过于臃肿，则优化HTML代码，去掉无用的标签，压缩JS、CSS，可以用CSS Spirit技术将多个图片放到一个图片中，减少向服务器的请求。如果是网络问题，则尝试在不同的网络、地区部署服务器，然后使用CDN技术加速访问。

如果是页面中的JavaScript运行复杂导致运行速度慢，则优化JavaScript。

webkaka：测网速。

如何处理百万条数据的优化？

对于经常进行检索的字段创建索引，对于经常进行范围查询的一个字段创建聚集索引；

当有大量数据进行插入的时候进行批量插入；

一些代码用存储过程进行重写（当一个逻辑有多行SQL要执行的时候用存储过程可以优化速度，可以避免客户端和SQLServer之间多次交互）；

(\*)集群、读写分离；

(\*)分区、分表；

**65、在程序编码的时候，你会对Sql注入漏洞的防范采取什么样的措施？**

尽量不要拼SQL语句！使用参数化查询或存储过程可以防止SQL注入攻击！在必须用SQL拼接的地方对用户输入的内容进行检查、过滤。

**66、你会采用什么样的策略和方法来实现系统缓存？**

在ASP.NET中 缓存有 页面缓存，数据源缓存，和一些自己定义的缓存！

对于那些整个页面不经常变化的我们可以使用页面缓存，而对于那些执行非常耗时的SQL语句并且数据的及时性要求不高的我们可以使用数据源缓存。

对于页面缓存、数据源缓存等都不满足要求的情况下采用直接操作HttpRuntime.Cache来自定义缓存策略。如果需要用多台Web服务器作为一个集群来承载网站的情况，则可以部署专门的分布式缓存服务器，比如Memcached

**67、网站想要实现文件防盗链的功能，说说你的解决方案。**

读取HTTP报文头中的UrlReferrer在Application\_BeginRequest中我们可以判断用户的请求是否来源于本网站。如果不是我们可以终止用户的请求（Response.End();）！

**68、ref与out的区别？**

Ref和Out这样个2个关键字都能够提供相似的功效，其作用也很像C中的指针变量。稍有不相同之处是：  
用Ref型参数时，传入的参数必须先被初始化。而Out则不要要，对Out而言，就必须在方法中对其完成初始化。  
用Ref和Out时都必须注意，在方法的参数和执行方法时，都要加Ref或Out关键字。以满足匹配。  
Out更适合用在要要Return多个返回值的地方，而Ref则用在要要被调出使用的方法修改调出使用者的引用的时候。

**69、MVC路由理解？**

1、首先我们要理解MVC中路由的作用：url Routing的作用是将浏览器的URL请求映射到特定的MVC控制器动作。

2、当我们访问http://localhost:8080/Home/Index这个地址的时候，请求首先被UrlRoutingModule截获，截获请求后，从Routes中得到与当前请求URL相符合的RouteData对象，将RouteData对象和当前URL封装成一个RequestContext对象，然后从Requestcontext封装的RouteData中得到Controller名字，根据Controller的名字，通过反射创建控制器对象，这个时候控制器才真正被激活，最后去执行控制器里面对应的action。

**70、谈谈你觉得做的不错系统，大概介绍下用到了哪些技术？**

就拿我之前做过的一个项目为例来简单说明一下吧。项目分为客户端和服务端，客户端分为BS客户端和CS客户端，BS客户端采用MVC 5.0的框架，CS客户端是Winform项目，服务端使用WebApi统一提供服务接口，考虑以后可能还要扩展手机端，所以服务接口的参数和返回值使用通用的Json格式来传递数据。

1、服务端采用的面向接口编程，我们在软件架构的过程中，层和层之间通过接口依赖，下层不是直接给上层提供实现，而是提供接口，具体的实现以依赖注入的方式在运行的时候动态注入进去。MEF就是实现依赖注入的一种组件。它的使用使得UI层不直接依赖于BLL层，而是依赖于中间的一个IBLL层，在程序运行的时候，通过MEF动态将BLL里面的实现注入到UI层里面去，这样做的好处是减少了层与层之间的耦合。服务端的异常里面、权限验证、日志记录等通用功能使用了AOP拦截的机制统一管理，项目中使用的是Postsharp这个组件，很好地将通用需求功能从不相关的类当中分离出来，提高了代码的可维护性。

2、BS的客户端采用的jquery+bootstrap 的方式，所有页面采用流式布局，能更好适应各种不同的终端设备（PC、手机）。项目中使用了各种功能强大的bootstrap组件，能适应各种复杂的业务需求。

**71、谈谈你对设计模式的认识？结合你用得最多的一种设计模式说说它的使用。**

**72、IIS的工作原理？**

1、当客户端发送HTTP Request时，服务端的HTTP.sys（可以理解为IIS的一个监听组件） 拦截到这个请求；

2、HTTP.sys 联系 WAS 向配置存储中心请求配置信息。

3、然后将请求传入IIS的应用程序池。

4、检查请求的后缀，启动aspnet\_isapi.dll这个dll，这个dll是.net framework里面的，也就是说到这一步，请求进入了.net framework的管辖范围。

5、这个时候如果是WebForm，开始执行复杂的页面生命周期（HttpRuntime→ProcessRequest→HttpContext→HttpHandler）；如果是MVC，则启动mvc的路由机制，根据路由规则为URL来指定HttpHandler。

6、httpHandler处理请求后，请求结束，给出Response，客户端处理响应，整个过程结束。

**73、Http协议**

1、http协议是浏览器和服务器双方共同遵循的规范，是一种基于TCP/IP应用层协议。

2、http是一种典型的请求/响应协议。客户端发送请求，请求的内容以及参数存放到请求报文里面，服务端收到请求后，做出响应，返回响应的结果放到响应报文里面。通过F12可以查看请求报文和响应报文。

3、http协议是”无状态”的，当客户端向服务端发送一次http请求后，服务端收到请求然后返回给客户端相应的结果，服务器会立即断开连接并释放资源。在实际开发过程中，我们有时需要“保持”这种状态，所以衍生出了Session/Cookie这些技术。

4、http请求的方式主要有get/post。

5、http状态码最好记几个，博主有一次面试就被问到了。200（请求成功）、404（请求的资源不存在）、403（禁止访问）、5xx（服务端错误）

**74、数据库优化经验（后端工程师非常常见）**

1、数据库运维方面的优化：启用数据库缓存。对于一些比较常用的查询可以采用数据库缓存的机制，部署的时候需要注意设置好缓存依赖项，防止“过期”数据的产生。

2、数据库索引方面的优化：比如常用的字段建索引，联合查询考虑联合索引。（PS：如果你有基础，可以敞开谈谈聚集索引和非聚集索引的使用场景和区别）

3、数据库查询方面的优化：避免select \* 的写法、尽量不用in和not in 这种耗性能的用法等等。

4、数据库算法方面的优化：尽量避免大事务操作、减少循环算法，对于大数据量的操作，避免使用游标的用法等等。

**75、关于代码优化你怎么理解？你会考虑去代码重构吗？**

1、对于代码优化，之前的公司每周会做代码审核，审核的主要作用就是保证代码的正确性和执行效率，比如减少代码的层级结构、避免循环嵌套、避免循环CURD数据库、尽量避免一次取出大量数据放在内存中（容易内存溢出）、优化算法等。

2、对于陈旧代码，可能很多地方有调用，并且开发和维护人员很有可能不是同一个人，所以重构时要格外小心，如果没有十足的把握，不要轻易重构。如果必须要重构，必须做好充分的单元测试和全局测试。

**76、关于服务器端 MVC 架构的技术实现，您是怎样理解的？这种架构方式有什么好处？您在项目中是如何应用这一架构的？**

参考答案：MVC，顾名思义，Model、View、Controller。所有的界面代码放在View里面，所有涉及和界面交互以及URL路由相关的逻辑都在Controller里面，Model提供数据模型。MVC的架构方式会让系统的可维护性更高，使得每一部分更加专注自己的职责，并且MVC提供了强大的路由机制，方便了页面切换和界面交互。然后可以结合和WebForm的比较，谈谈MVC如何解决复杂的控件树生成、如何避免了复杂的页面生命周期。

**77、网站优化：网站运行慢，如何定位问题？发现问题如何解决？**

浏览器F12→网络→查看http请求数以及每个请求的耗时，找到问题的根源，然后依次解决，解决方案可以参考问题一里面的Web优化方案。

**78、说说你最擅长的技术？并说说你是如何使用的？**

 简单谈谈MEF在我们项目里面的使用吧。

在谈MEF之前，我们必须要先谈谈DIP、IOC、DI

依赖倒置原则（DIP）：一种软件架构设计的原则（抽象概念）

控制反转（IoC）：一种反转流、依赖和接口的方式（DIP的具体实现方式）。

依赖注入（DI）：IoC的一种实现方式，用来反转依赖（IoC的具体实现方式）。

什么意思呢？也就是说，我们在软件架构的过程中，层和层之间通过接口依赖，下层不是直接给上层提供实现，而是提供接口，具体的实现以依赖注入的方式在运行的时候动态注入进去。MEF就是实现依赖注入的一种组件。它的使用使得UI层不直接依赖于BLL层，而是依赖于中间的一个IBLL层，在程序运行的时候，通过MEF动态将BLL里面的实现注入到UI层里面去，这样做的好处是减少了层与层之间的耦合。这也正是面向接口编程方式的体现。

**79、自己写过多线程组件吗？简要说明！**

参考答案：可以参考<http://www.cnblogs.com/Alexander-Lee/archive/2009/10/31/1593647.html>

**80、谈谈ERP、CRM、MES、OA类型系统的不同**

ERP：Enterprise Resource Planning，企业资源计划。是指建立在信息技术基础上，以系统化的[管理思想](http://baike.baidu.com/view/141611.htm)，为企业决策层及员工提供决策运行手段的管理平台。

CRM：Customer Relationship Management，客户关系管理。是一种以"客户关系一对一理论"为基础，旨在改善企业与客户之间关系的新型管理机制。

MES：manufacturing execution system，生产执行系统。是一套面向制造企业车间执行层的生产信息化管理系统

OA：Office Automation,办公自动化。将[现代化办公](http://baike.baidu.com/view/2218242.htm)和计算机网络功能结合起来的一种新型的办公方式

**81、如何处理几十万条并发数据？**

答：用存储过程或事务。取得最大标识的时候同时更新..注意主键不是自增量方式这种方法并发的时候是不会有重复主键的..取得最大标识要有一个存储过程来获取.

**82、ASP.Net页面生命周期简单描述**

**[复制代码](javascript:void(0);)**

每个页面的生命周期为用户的每一次访问，也就是说每一次客户端与服务器之间的一个往返过程.全局变量的生命周期在此之间.

1. Page\_Init();

2. Load ViewState and Postback data;

3. Page\_Load();

4. Handle control events;

5. Page\_PreRender();

6. Page\_Render();

7. Unload event;

8. Dispose method called;