# .NET高薪面试高频题库

值类型和引用类型的区别

1. 值类型分配在线程栈上（管理由操作系统负责），引用类型分配在托管堆上（管理由垃圾回收器GC负责）。管理指内存空间的分配和释放
2. 值类型继承自valueType,valueType继承自System.Object；引用类型直接继承自System.Object
3. 值类型在作用域内结束时，会被操作系统自释放，减少托管堆压力；引用类型则靠GC。因此值类型在性能上由优势
4. 值类型是密封的，不能作为基类。引用类型一般具有继承性
5. 值类型不能为null，默认初始化为该类型的默认值；引用类型默认初始化为null
6. 值类型作为参数传递时，不影响本身。引用类型作为参数传递时，会改变最终该变量的值
7. 装箱：值类型转为引用类型
8. 拆箱：引用类型转为值类型

简述 C#中重写、重载和隐藏的概念

1. 重写是指用 override 关键字重新实现基类中的虚方法，在运行的过程中，无论通过哪个类型的引用，真正对象类型的方法将被调用。
2. 隐藏是指用 new关键字重新实现基类中的方法，在运行的过程中通过引用的类型判断应该调用哪个类型的方法。
3. 重载是指多个方法共享同一个名字并且拥有相同的返回值，但是却可以拥有不同的参数列表。

Serializable 特性有何作用   
通过为类型添加Serialization特性，可以使对象申明为可被序列化，简单说，就是数据通信时，并不是所有的数据类型都支持的，这时就需要双方都支持的一种格式，把你的对象转换成能通讯的格式就是序列化，反之就是反序列化。

请解释委托的基本原理   
委托是一类继承自System.Delegate 的类型，每个委托对象至少包含了一个指向某个方法的指针，该方法可以是实例方法，也可以是静态方法。委托实现了回调方法的机制，能够帮助程序员设计更加简洁优美的面向对象程序。

什么是链式委托   
链式委托是指一个由委托串成的链表，当链表上的一个委托被回调时，所有链表上该委托的后续委托将会被顺序执行。

请解释事件的基本使用方法   
事件是一种使对象或类能够提供通知的成员。客户端可以通过提供事件处理程序为相应的事件添加可执行代码。事件是一种特殊的委托。

请解释反射的基本原理和其实现的基石   
反射是.NET中的重要机制，通过反射，可以在运行时获得程序或程序集中每一个类型（包括类、结构、委托、接口和枚举等）的成员和成员的信息。有了反射，即可对每一个类型了如指掌。另外还可以直接创建对象，即使这个对象的类型在编译时还不知道。  
如何利用反射来实现工厂模式   
使用反射可以实现灵活性较高的工厂模式，其关键在于动态地查找产品所包含的所有零件，而不需要通过代码来逐一分析使用者的需求。反射工厂模式具有灵活性高，运行效率相对较低的特点。

什么是线程   
线程（Thread）是进程中的基本执行单元，是操作系统分配CPU时间的基本单位，一个进程可以包含若干个线程，在进程入口执行的第一个线程被视为这个进程的主线程。在.NET应用程序中，都是以Main()方法作为入口的，当调用此方法时系统就会自动创建一个主线程。

线程和进程的关系

windows系统是一个多线程的操作系统。一个程序至少有一个进程,一个进程至少有一个线程。进程是线程的容器，一个C#客户端程序开始于一个单独的线程，CLR(公共语言运行库)为该进程创建了一个线程，该线程称为主线程。例如当我们创建一个C#控制台程序，程序的入口是Main()函数，Main()函数是始于一个主线程的。它的功能主要 是产生新的线程和执行程序。C#是一门支持多线程的编程语言，通过Thread类创建子线程。

前台线程与后台线程

C#中的线程分为前台线程和后台线程，线程创建时不做设置默认是前台线程。即线程属性IsBackground=false。  
Thread.IsBackground = false;

两者的区别就是：应用程序必须运行完所有的前台线程才可以退出；而对于后台线程，应用程序则可以不考虑其是否已经运行完毕而直接退出，所有的后台线程在应用程序退出时都会自动结束。一般后台线程用于处理时间较短的任务，如在一个Web服务器中可以利用后台线程来处理客户端发过来的请求信息。而前台线程一般用于处理需要长时间等待的任务，如在Web服务器中的监听客户端请求的程序  
如何使用.NET的线程池   
线程池是一种多线程处理形式，处理过程中将任务添加到队列，然后在创建线程后自动启动这些任务。线程池线程都是后台线程。每个线程都使用默认的[堆栈](http://baike.baidu.com/view/93201.htm)大小，以默认的优先级运行，并处于多线程单元中。如果某个线程在[托管代码](http://baike.baidu.com/view/159896.htm)中空闲（如正在等待某个事件）,则线程池将插入另一个[辅助线程](http://baike.baidu.com/view/262200.htm)来使所有处理器保持繁忙。如果所有线程池线程都始终保持繁忙，但队列中包含挂起的工作，则线程池将在一段时间后创建另一个辅助线程但线程的数目永远不会超过最大值。超过最大值的线程可以排队，但他们要等到其他线程完成后才启动。

C#中的 lock 关键字有何作用   
C#中的lock关键字实质是调用 Monitor.Enter 和Monitor.Exit两个方法的简化语法，功能上其实现了进入和退出某个对象的同步。通常情况下，可以通过 lock一个私有的引用成员变量来完成成员方法内的线程同步，而通过lock一个私有的静态引用成员变量来完成静态方法内的线程同步。

GET请求和 POST 请求有何区别   
HTTP协议中有两种常用的请求：GET 和POST。

GET 请求把表单的数据显示地放在 URI 中，并且对长度和数据值编码有所限制。

POST 请求把表单数据放在 HTTP请求体中，并且没有长度的限制。

 如何防止 SQL注入式攻击   
SQL 注入式攻击时常见的一种攻击方法，主要利用的是系统设计的弊端。程序员在设计时需要考虑到注入式攻击的问题，避免直接使用用户输入拼接 SQL 语句，适当使用加密数据进行存储，并且在合适的场合使用存储过程。

Jquery选择器有哪些？

元素选择器、ID选择器、类选择器、属性选择器、过滤选择器、层级选择器、表单选择器

什么是跨域请求

1.浏览器从一个域名的网页去请求另一个域名的资源时，域名、端口、协议任一不同，都是跨域

2.只要协议、域名、端口有任何一个不同，都被当作是不同的域，之间的请求就是跨域操作。

数据库多表联建查询几种方式

交叉连接，内连接，自连接，外连接

如果表的数据量比较大

1. 合理使用索引；
2. 避免或简化排序；
3. 消除对大型表行数据的顺序存取；
4. 避免相关子查询；
5. 避免困难的正规表达式；
6. 使用临时表加速查询。

视图和表的区别

1、表只用物理空间而视图不占用物理空间，视图只是逻辑概念的存在，表可以及时对它进行修改，但视图只能有创建的语句来修改。

2、视图是查看数据表的一种方法，可以查询数据表中某些字段构成的数据，只是一些[SQL语句](https://www.baidu.com/s?wd=SQL%E8%AF%AD%E5%8F%A5&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)的集合。从安全的角度说，视图可以不给用户接触数据表，从而不知道表结构。

3、表属于全局模式中的表，是实表；视图属于局部模式的表，是虚表。 联系：视图（view）是在基本表之上建立的表，它的结构（即所定义的列）和内容（即所有数据行）都来自基本表，它依据基本表存在而存在。一个视图可以对应一个基本表，也可以对应多个基本表。视图是基本表的抽象和在逻辑意义上建立的新关系。

**视图可以操作数据么？**

要通过视图更新基本表数据，必须保证视图是可更新视图，即可以在INSET、UPDATE或DELETE等语句当中使用它们。对于可更新的视图，在视图中的行和基表中的行之间必须具有一对一的关系。还有一些特定的其他结构，这类结构会使得视图不可更新。   
如果视图包含下述结构中的任何一种，那么它就是不可更新的：  
（1）聚合函数；  
（2）DISTINCT关键字；  
（3）GROUP BY子句；  
（4）ORDER BY子句；  
（5）HAVING子句；  
（6）UNION运算符；  
（7）位于选择列表中的子查询；  
（8）FROM子句中包含多个表；  
（9）SELECT语句中引用了不可更新视图；  
（10）WHERE子句中的子查询，引用FROM子句中的表；  
（11）ALGORITHM 选项指定为TEMPTABLE（使用临时表总会使视图成为不可更新的）。

**工厂模式**

工厂方法模式：工厂方法模式定义一个用于创建对象的接口，让子类决定实例化哪一个类。工厂方法使一个类的实例化延迟到其子类。工厂模式分为简单工厂模式，工厂方法模式和抽象工厂模式，它们都属于设计模式中的创建型模式。其主要功能都是帮助我们把对象的实例化部分抽取了出来，目的是降低系统中代码耦合度，并且增强了系统的扩展性。

**接口和类的区别**

1、接口类似于类，但接口的成员都没有执行方式，它只是方法、属性、事件和索引的组合而已，并且也只能包含这四种成员；类除了这四种成员之外还可以有别的成员(如字段)。  
2、不能实例化一个接口，接口只包括成员的签名；而类可以实例化(abstract类除外)。  
3、接口没有[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)，类有[构造函数](https://www.baidu.com/s?wd=%E6%9E%84%E9%80%A0%E5%87%BD%E6%95%B0&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)。  
4、接口不能进行运算符的重载，类可以进行[运算符重载](https://www.baidu.com/s?wd=%E8%BF%90%E7%AE%97%E7%AC%A6%E9%87%8D%E8%BD%BD&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)。  
5、接口的成员没有任何修饰符，其成员总是公共的，而类的成员则可以有修饰符(如：虚拟或者静态)。  
6、派生于接口的类必须实现接口中所有成员的执行方式，而从类派生则不然。

**谈一下对EF的理解 使用它的好处**

EF是一种通过映射操作实现数据交互的ORM框架技术

优点：

1.简洁的Linq to Sql语句大大提高了开发人员的效率，不要再写复杂的sql语句；

2.不再需要再管应用程序如何去连接数据库；

3.EF可以用作用于数据服务和OData Service的基础设施；

**ajax的底层原理**

ajax是一种异步通信技术，AJAX 的核心是 XMLHttpRequest 对象。使用XMLHttpRequest来和服务器进行异步通信

**Api的接口和我们平时定义的接口有什么不同**

从狭义上讲，接口指的是借由 interface 定义的结构，接口中只对方法做定义，不做实现。具体实现由最终实现接口的类提供。 interface 作为一种类型，可以用于定义方法，我们只关心类实现了接口声明的方法，并不关心实际上引用的是哪个类的对象。

接口是应用程序和模块之间相互沟通的方式。提供某些功能的应用程序或者模块为了与其它程序相互沟通，对外公开了某些功能的调用方式方法，这一套方式方法就是 API 。

C#接口是狭义的接口，只是编程中的一个规范，像模式、AOP之流，局限于某个领域

api是一个自闭的最小模块向外部提供访问的一个公开途径，是公众称呼

广义的接口是指对外提供交互，如api接口、usb接口、rest接口

**OOP是怎么理解的？**

OOP是面向对象编程，特征分别是封装、继承、多态、抽象。

学过别的语言吗？

学过，C语言，JS，Java也了解一点

如果是自己开发，会选择的哪种ORM框来进行数据库交互？

Dapper或者EF

**问题：****WebApi和WebService的区别是什么?**

WebService的数据格式是XML,消息体比较大,它的应用协议是SOAP(简单对象访问协议)，在某些业务复杂，稳定性和正确性要求较高的领域，WebService还是有用武之地的。

WebApi是网络应用程序接口，用的是HTTP协议，可以对接各种客户端，移动设备，构建http服务的框架。

**问题：MVC的特点是什么？**

耦合性低，重用性高，部署快，可维护性高，前后端分离，有利于软件工程化管理。

**谈谈你对索引的理解？**

SQL索引有两种，聚集索引和非聚集索引，索引主要目的是提高了SQL Server系统的性能，加快数据的查询速度与减少系统的响应时间 聚集索引存储记录是物理上连续存在，而非聚集索引是逻辑上的连续，物理存储并不连续SQL索引有两种，聚集索引和非聚集索引，索引主要目的是提高了SQL Server系统的性能，加快数据的查询速度与减少系统的响应时间 聚集索引存储记录是物理上连续存在，而非聚集索引是逻辑上的连续，物理存储并不连续

**对异步加载的理解**

1. Ajax异步加载，减少渲染阻塞的js和css
2. 推迟js的解析
3. 不重要的脚本可推迟执行解析。

**对队列的特点？**

先进先出，像一些高并发的也都处理过 "队列是一种只允许在表的一端进行插入，在另一端进行删除操作的线性表，因此具有先进先出的特性。队列中允许进行插入的一端称为队尾，允许删除的另一端成为对头，当队列中没有元素时称为空对。

队列的基本操作有：1.构造一个空的队列。2.在队尾部插入元素3.在对头删除元素。4.得到对头元素。5.判断队列是否为空。

**面试问题：简单叙述EF的核心思想**

将表实体的变化，映射到表的变化；

对于ADO.net的使用在博客中进行了总结：.Net连接SQL Server。

在ADO .NET中用于存取和管理资料的两个主要元件是.NET Framework 资料提供者和 DataSet，5大对象实现应用 程序和数据库之间的操作。而在EF中

1.封装了数据库的统一入口，EF上下文；

2.把对实体的操作放入上下文中；

3.把对实体操作产生的变化生成Sql脚本执行到数据库中

面试问题：WebAPI通常来干什么使用

问题解答：

一般作为接口来使用

面试问题：你们现在服务端的分层一般是什么结果

问题解答：

我们一般就是三层+工厂+接口+依赖注入

面试问题：Redis这东西有什么用

问题解答：

 通常局限点来说，Redis也以消息队列的形式存在，作为内嵌的List存在，满足实时的高并发需求。而通常在一个电商类型的数据处理过程之中，有关商品，热销，推荐排序的队列，通常存放在Redis之中，期间也包扩Storm对于Redis列表的读取和更新。

面试问题：Redis常用的五种方式

问题解答：

字符串（String）、哈希/散列/字典（Hash）、列表（List）、集合（Set）、有序集合（sorted set）五种

面试问题：什么叫做多线程

问题解答：

多线程是为了使得多个线程并行的工作以完成多项任务，以提高系统的效率。线程是在同一时间需要完成多项任务的时候被实现的。

面试问题：分层结构是哪几层

问题解答：界面层（User [Interface](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=3389638&ss_c=ssc.citiao.link" \t "_blank) layer）、[业务逻辑层](https://baike.sogou.com/lemma/ShowInnerLink.htm?lemmaId=7792526&ss_c=ssc.citiao.link" \t "_blank)（Business Logic Layer）、数据访问层（Data access layer）

面试问题：VUE是什么

问题解答：是一套构建用户界面的 渐进式框架。与其他重量级框架不同的是，Vue 采用自底向上增量开发的设计。Vue 的核心库只关注视图层，并且非常容易学习，非常容易与其它库或已有项目整合

面试问题：后端拿到数据有哪些过程

问题解答：请求数据，解析，发送请求，查找路径，查询到控制器中的方法。

面试问题：存储过程接触过吗

问题解答： 固化在SQL数据库系统内部的SQL语句，这样做的好处是可以提高效率、增强数据库的安全性、减少网络流量。

面试问题：缓存失效是什么意思

问题解答：有时候如果网站并发访问高，一个缓存如果失效，可能出现多个进程同时查询DB，同时设置缓存的情况，如果并发确实很大，这也可能造成DB压力过大，还有缓存频繁更新的问题。

我现在的想法是对缓存查询加锁，如果KEY不存在，就加锁，然后查DB入缓存，然后解锁；其他进程如果发现有锁就等待，然后等解锁后返回数据或者进入DB查询。

这种情况和刚才说的预先设定值问题有些类似，只不过利用锁的方式，会造成部分请求等待。

面试问题：支付模块独立开发（开放原则）和解耦

问题解答：

首先要开发一个独立松耦合的支付模块的话，根据我目前所掌握的技术的话，我觉得用工厂模式是必须的，我们开一个支付的接口，其他模块可以通过反射找到这个方法，让其他模块的来实现这个接口功能，也可以采用通知的模式来进行开发

[String和StringBuiler的区别](https://www.cnblogs.com/xiangkejin/p/6444400.html)?

String 是不可变的对象，因此在每次对String类型进行改变的时候其实都等同于生成了一个新的 String对象，然后将指针指向新的String对象，所以经常改变内容的字符串最好不要用 String，因为每次生成对象都会对系统性能产生影响.

StringBuilder 上来先设置个缓冲区，然后开始写入字符串，超过了缓存区就二倍增容。

可不可以介绍下MVC是什么？

MVC全名是Model View Controller，是模型(model)－视图(view)－控制器(controller)的缩写，一种软件设计典范，用一种业务逻辑、数据、界面显示分离的方法组织代码，将业务逻辑聚集到一个部件里面，在改进和个性化定制界面及用户交互的同时，不需要重新编写业务逻辑。MVC被独特的发展起来用于映射传统的输入、处理和输出功能在一个逻辑的图形化用户界面的结构中。

 MVC就是类似三层的一种[架构](http://lib.csdn.net/base/architecture" \t "https://www.cnblogs.com/shidan/p/_blank" \o "大型网站架构知识库)，主要还是采用封装（分层）的思想，来降低耦合度，从而使我们的系统更加的灵活，扩展性更好。

路由是什么？

WebApi和WebSerive的区别？

Ref和Out的区别？

1、使用ref型参数时，传入的参数必须先被初始化，对out而言，必须在方法中对其完成初始化。

2、使用ref和out时，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。

3、out适合用在需要retrun多个返回值的地方，而ref则用在需要被调用的方法修改调用者的引用的时候。

4、ref传进去的参数在函数内部可以直接使用，而out不可。

5、系统对ref的限制是更少一些的。

6、若要使用 ref 参数，必须将参数作为 ref 参数显式传递到方法，ref 参数的值被传递到 ref 参数。

7、当希望方法返回多个值时，声明 out 方法非常有用；使用 out 参数的方法仍然可以返回一个值.

out：标识一个参数值会受影响的参数，但在传入方法时，该参数无需先初始化。

ref：标识一个参数值可能会受影响的参数。

对Sql的优化的理解？

**1、**在表中建立索引，优先考虑where、group by使用到的字段。

2、尽量避免使用select \*，返回无用的字段会降低查询效率。

3、尽量避免使用in 和not in，会导致数据库引擎放弃索引进行全表扫描。

4、尽量避免使用or，会导致数据库引擎放弃索引进行全表扫描。

5、尽量避免在字段开头模糊查询，会导致数据库引擎放弃索引进行全表扫描。

6、尽量避免进行null值的判断，会导致数据库引擎放弃索引进行全表扫描。

7、尽量避免在where条件中等号的左侧进行表达式、函数操作，会导致数据库引擎放弃索引进行全表扫描。

8、当数据量大时，避免使用where 1=1的条件。通常为了方便拼装查询条件，我们会默认使用该条件，数据库引擎会放弃索引进行全表扫描。

对泛型的理解？

泛型是 2.0 版 C# 语言和公共语言运行库 (CLR) 中的一个非常重要的新功能。

泛型（Generic） 允许您延迟编写类或方法中的编程元素的数据类型的规范，直到实际在程序中使用它的时候。换句话说，泛型允许您编写一个可以与任何数据类型一起工作的类或方法。您可以通过数据类型的替代参数编写类或方法的规范。当编译器遇到类的构造函数或方法的函数调用时，它会生成代码来处理指定的数据类型。

面试问题：数据库为什么不使用IN和ONTIN优化

问题解答 因为在建立索引后进行IN 和NOT IN会使数据库放弃建立的索引进行全表扫描

面试问题：控制反转

问题解答 也就是依赖注入，就是创建调用者的工作不再有被调用者来创建，有Sping容易来实现，然后注入调用者

面试问题：webAPI通信安全

问题解答 可以再API使用token和签名验证来加密API的安全性

http和https的主要区别

http的安全性很低,他的链接是很简单,https 他是可以加密的,可以进行身份验证,安全性比http要安全的多,

http协议

Http 是客户端和服务器的一个应答标准,通常由客户端发起一个请求 发送到服务器,服务器收到请求以后,给客户端返回一个请求 比如说200 然后客户端收到请求 向服务器发送一个收到请求 这就是三次握手

多继承

一个子类同时有多个父类这样的关系称之为 多继承 多继承可以看做是单继承的扩展.

IOC

Ioc 他不是一个技术 是一种思想, 他可以指导我们如何设计出 松耦合,更优良的程序,传统应用程序都是由于我们在类内部主动创建依赖对象 从而导致类与类之间 高耦合,难于测试,有了ioc容器以后我们可以吧创建依赖对象的控制权交给了ioc 容器 由容器注入组合对象,所以对象和对象之间是松耦合的 便于测试 最重要的是 使各个功能之间变得非常灵活

依赖注入

所谓依赖注入就是说 在运行期 由外部容器直接将动态对象注入到组件中,当spring 容器初始化,创建并且管理bean对象,以及销毁他, 所以我们只需要从外部容器直接获取bean对象,而不用编写代码创建bean 对象, 这种现象就叫做控制反转,就是应用本身不负责以来对象的创建和维护,依赖对象的创建和维护是由外部容器控制的,这样控制权就从应用本身转移

mvc 中 string 和 String 有什么区别

String是 Framework中的 ， string 是 c#中的 ， string是String的别名， 一般来说写string的时候编译的时候 ， 会把string 转为 String ， 而用String编译的时候会少做一点工作.

作业

作业他是sql server 自动执行的指定操作 比如 可以实现 数据库每天十点进行备份

.NET中集合类型有哪些？

动态数组（ArrayList） 它代表了可被单独索引的对象的有序集合。

哈希表（Hashtable） 它使用键来访问集合中的元素。

排序列表（SortedList）它可以使用键和索引来访问列表中的项。

堆栈（Stack） 它代表了一个后进先出的对象集合。

队列（Queue） 它代表了一个先进先出的对象集合。

点阵列（BitArray） 它代表了一个使用值 1 和 0 来表示的二进制数组。

字典类型的说明

回 答： 字典是Key—value类型的，是个数组类型的，是个键值对类型的集合；

解 答： 必须包含名空间System.Collection.Generic，Dictionary里面的每一个元素都是一个键值对(由二个元素组成：键和值) ， 键必须是唯一的,而值不需要唯一的，键和值都可以是任何类型(比如：string, int, 自定义类型，等等)，通过一个键读取一个值的时间是接近O(1)，键值对之间的偏序可以不定义。

.NET 数据类型float、double和decimal区别

回 答： float 单精度浮点，double 双精度浮点，decimal是高精度浮点型；

解 答： float 单精度浮点 32bit，double 双精度浮点64bit，decimal是高精度 128bit，浮点型，float double 是 基本类型（primitive type），decimal不是。decimal的[有效位数](https://www.baidu.com/s?wd=%E6%9C%89%E6%95%88%E4%BD%8D%E6%95%B0&tn=SE_PcZhidaonwhc_ngpagmjz&rsv_dl=gh_pc_zhidao" \t "https://zhidao.baidu.com/question/_blank)很大，达到了28位，但是表示的数据范围却比float和double类型小。  
使用的时候会对计算时的性能有影响。decimal所能储存的数比double大，从double到decimal的类型转换不会出现任何问题。

高并发怎么处理？  
尽量使用Redis缓存，包括用户缓存，信息缓存等，多花点内存来做缓存，可以大量减少与数据库的交互，提高性能。  
使用Redis队列List，Push和POP方法保证了原子性的实现。即使有很多用户同时到达，也是依次执行。缓存也是可以应对写请求的，比如我们就可以把数据库中的库存数据转移到Redis缓存中，所有减库存操作都在Redis中进行，然后再通过后台进程把Redis中的用户请求同步到数据库中。

问题:方法重写、抽象类是什么意思？  
重写要求方法名、参数和返回值相同，重写是为了增强类的重用性和复用性，扩展性；重写是对类中方法的扩充。要被重写的基方法必须声明为为virtual、abstract。  
抽象类是类，只能被继承，但是不能实例化。可以包含抽象成员，也可以包含非抽象成员，抽象成员只能出现在抽象类中，抽象方法不能有方法体语句，可以有参数列表，如果子类不是抽象类，子类必须实现抽象类所有抽象方法。  
  
问题:什么是依赖注入？  
在程序运行期间，由外部容器动态的将依赖对象注入到组件中。当spring容器启动后，spring容器初始化，创建并管理bean对象，以及销毁它。我们只需要从容器中直接获取bean对象，不用编写代码来创建，这种现象叫做依赖注入。  
所谓依赖注入，是指程序运行过程中，如果需要调用另一个对象协助时，无须在代码中创建被调用者，而是依赖于外部的注入。Spring的依赖注入对调用者和被调用者几乎没有任何要求，完全支持对POJO之间依赖关系的管理。

Webapi异常捕抓：

局部的话采用try...cache

全局可以使用异常过滤器

C#中var与js中var的区别：

在c#中必须初始化，定义时不能为空值，初始化完成后就不能再赋值，

在JS中var是弱类型的可以当类型使用，也可以当引用类型使用，如果为控不会报错而是默认为undefined。

lambda表达式：

lambda是一种运算符=>，lambda表达式相当于一个匿名函数

它的写法更加的简单灵活，

使用过哪些前段框架？

答：使用过html,css,js,vue等框架。

什么是回调函数？

答：字面上的理解，回调函数就是一个参数，将这个函数作为参数传到另一个函数 里面，当那个函数执行完之后，再执行传进去的这个函数。这个过程就叫做回调。

问题六：EF怎么怎么执行SQL语句

答：1、使用SqlQuery在已知的实体上执行SQL查询语句

2、使用ExcuteSqlCommand在未知实体上执行增删改操作

秒杀的业务逻辑？

答：①计时交互：通过后端请求拿到当前系统时间；将start，end，now作为参数 传进countdown方法作时间判断，now>end-->秒杀结束;now<start-->

秒杀未开始，计时事件绑定（countdown），并在计时结束后，获取秒杀地址，控 制现实逻辑，执行秒杀[handleSeckill（exposer）]，即出现按钮；now>start&&now<end :执行秒杀操作[handleSeckill（exposer）]

②执行秒杀：获取exposer，判断是否开启秒杀，是：给出秒杀地址，并且绑定 一次点击事件以执行秒杀请求（1.禁用按钮；2.发送秒杀请求），根据实际情况判断 秒杀成功与秒杀异常（重复秒杀，秒杀结束，系统异常等）；否：获取now，start， end，再次执行countdown操作。

问题四：cookie和Session的区别

答：1、cookie数据存放在客户的浏览器上，session数据放在服务器上。

1. cookie不是很安全，别人可以分析存放在本地的COOKIE并进.

问题五：cookie放在那里，session放在那里？

答：cookie存放在客户端，session存放在服务器

问题六：BS和CS的区别

答：1．硬件环境不同 2．对安全要求不同 3．对程序架构不同 4．软件重用不同 5．系统维护不同 6．处理问题不同 7．用户接口不同 8．信息流不同 。

存储过程和一般sql语句那个效率高

数据库会对存储过程进行预编译，因此存储过程的速度比较快一些、效率比较高

mvc 路由机制

首先URL会进入到路由表中再通过路由表内的路由找到匹配的方法

四大过滤器(filters)拦截器

授权过滤器、异常过滤器、结果过滤器、行为过滤器

代码重构

改善软件质量、性能，使他的设计模式、架构更加的合理，提高扩展性和维护性

token验证

每一次访问API都有一个token令牌，token令牌进行三次判断，验证是否存、第二次验证是否过期、第三次验证是否有效合法

B/S架构优先访问的文件

Global.asax

方法重写

重写是编译时的一种多态，根据参数的不同、类型的不同、顺序的的不同调用不同的方法

消息队列

主要用于解耦、异步调用、消息通讯、流量削峰等问题，实现高性能、可伸缩、最终一致性的架构

oop面对对象

核心思想就是：封装、继承、多态。是一种面向对象的设计模式，主要目标就是重用性、灵活性、扩展性。

Ef的核心思想

是Ado.net中的一套支持面向数据的应用程序的技术属于ORM框架

DI和IOC

IOC（控制反转）：将自身对象中的一个内置对象的控制反转，反转后不再有自己操作创建，而是由第三方系统来完成。DI（依赖注入）：自身对象中的内置对象是通过注入的方式进行创建。

面试问题：值类型后面加?

问题解答：

在值类型后面加？是可以为NULL值的

依赖注入

我的回答：

依赖注入(DI)和控制反转(IOC)是同一个概念。具体含义是:当某个角色需要另一个角色的协助时，在一般的程序设计过程中，通常由调用者来创建被调用者的实例。但在Spring里，创建被调用者的工作不再由调用者来完成，因此称为控制反转;创建被调用者 实例的工作通常由Spring容器来完成，然后注入调用者，因此也称为依赖注入。依赖注入（Dependency Injection）是用于实现控制反转（Inversion of Control）的最常见的方式之一（控制反转还有一种常见的实现方式称为依赖查找）