《职坐标：JavaEE 岗位就业标准化手册 V2.0》

定位职场坐标，打造 IT 达人

Revision Time：2017 年 5 月 25 日

Dept：网校教学部

Version：V2.0

第一部分 就业流程与要求

1.就业流程

1.1 面试题发布

在项目期结束后，由班级技术老师将模拟面试题库（此文档）发布到班级群，模拟面试会针对此文档中列出的高频面试题、项目开发流程、常用开发工具中内容中随机抽取（但不仅限于此文档）。

1.2 简历指导

1.3 就业指导

1.4 模拟面试

1.5 简历投递

要求：每天至少投递 200 份简历。

2.模拟面试要求

2.1 模拟面试意义

（1）临门一脚，增强面试经验，提高面试成功率。

（2）提前熟悉面试流程，避免因个人心态紧张、环境陌生等心理素质问题，导致发挥不出实际应有水平。

（3）提前熟悉和掌握企业项目开发流程和团队协作常用工具。

2.2 模拟面试要求

（1）模拟面试重要性：模拟面试作为整个学习和岗位认证考核的一部分。如果没有及格需要延迟就业时间。模拟面试通过方可进入就业环节。

（2）模拟面试考核标准：人事面试和技术面试，需要按照模拟面试评分表进行打分，60 分通过。

（3）技术面试考核要求：针对老师提供的《岗位就业标准化手册》文档，老师会从中随机抽取至少 10 道面试题，随机抽取的面试题需要覆盖该岗位所有课程，至少答对 6 道，技术面试才算通过。

第二部分 人事面试常见问题

常见人事面试题	
一、基本情况考察	
1.请谈一下你自己的一些情况	**考察：1、表达流程；2、凸出亮点、优点** 【思路】：建议大家用 2 分钟得自我介绍，面试官较喜欢的自我介绍（1）有亮点，每一小段都有一个亮点，而不是平铺直叙（2） 有互动：每一小段都会和面试官互动，而不是自说自话，但是切记，这种互动并不需要面试官配合，绝对不要总是直勾勾地盯着面试官的眼睛逼着人家配合你，要知道面试官最恐怖的经历就是被申请人从头盯到尾！（3） **相关**
二、大学情况考察	
1.你对自己的学习成绩满意吗	【思路】：有的毕业生成绩比较好，这样的问题就很好回答，但对于那些成绩不太好的毕业生，可以表明自己的态度，并给予一个合适的理由，但不能找客观原因，如“老师教得不好”，显得你是推卸责任的人，同时最好突出一个自己好的方面，以免让人觉得你一无是处。 正确回答：**比较满意，我想我在学校学到的不仅是课本知识，还学到很多为人处世的经验。**
2.你觉得大学生活使你收获了什么？	【思路】：这是一个总体概括性的问题， 很明显， 面试官期待你给出一个总体概括性的回答。 范例：回答示范 1： 我觉得大学生活使我学会了与人沟通， 可能您会觉得， 十个大学生有九个会强调自己善于与人沟通， 不过我依然觉得这是我大学里面最大的收获。您从简历上看得出来， 我大学里在学生会工作了两年半， 从干事一直到副主席， 这使我有机会锻炼与年龄、 背景完全不同的人交流， 从学生到老师， 从学校的领导到校外的公司， 每一种沟通的方式和方法都不同， 的确使我学到了很多。回答示范 2： 我觉得大学是我迄今为止成长最快的时期， 有很多收获。首先是知识和技能方面的， 我修了"地理科学"和"管理学"两个专业， 虽然我不打算从事地理科学方面的工作， 但是我掌握了很多必要的工作技能， 比如搜索信息、 分析信息、 独立思考等。除了知识，我还提升了自己的综合素质。就拿我担任班长这件事来说， 我觉得提升很快， 首先是要竞选， 竞选成功以后要策划吸引人的班级活动， 策划活动的过程中要调动同学一块参与等等， 每一个环节都很锻炼人。
3.你的专业课程中，哪	【思路】你在学校最喜欢的科目一般而言也是你最学有所长、学有所得的科目。如果你觉得这个科目能对你当前应聘的工作产生积极作用，就抓住机会予以强调，作深入细致

些课程最让你感兴趣了？	阐述，否则不妨淡化处理。在回答在校期间最不喜欢的科目时注意把握如下原则：**第一，要懂得如何避重就轻；第二，假如你不喜欢的科目恰好与所应聘的工作密切相关，那你就需要巧妙改变主题；第三，要有幽默感。关于这类问题，面试应届大学毕业生和毕业研究生时主考官常会提起，因此，如果你是一个青年学生，在面试前就不妨琢磨琢磨，用心准备一番。**
4. 介绍一下你的课外活动。你为什么愿意从事那些课外活动？通过那些课外活动，你都学了些什么？	【思路】：这个问题考查是你的业余倾向，还有你的学习能力，**切记不要说一些娱乐的业余课外活动，选一些积极向上的比**如：学习小组、志愿者等等，其次叙述你在其中体现了哪些人身价值，从中锻炼了哪方面，积累了哪些经验，学到了什么！
5. 大学时，你为什么选择专业？	【思路】：如果你应聘得岗位和你所学得专业对口的话很容易回答，但如果和所学专业不对口得话，回答起来就要慎重看了。比如说是父母选的或是其他同学都添的这个专业等，但上学后觉得不适合自己，在休完本专业的情况下，努力学习适合自己的动画行业，毕业之后又到专门的培训机构学习，并有项目经验等
6. 我想知道，你在大学时遇到的最有挑战性的事情是什么？为什么你认为那件事对你最具有挑战性？	【思路】：**面试者往往通过这个问题考查你的信心，信心是应聘者在面试者面前是否具有吸引力的一个非常重要的因素。**有信心的人往往在办事、说话和判断中，以及在对自己的能力方面表现出强烈的信心。有信心的人善于对他们自己的决定和行为的后果承担责任。此外，他们往往把冲突视为是发展的机会
7. 如果你在大学（学院）做过兼职工作的话，你认为哪种兼职工作对你最有意思？为什么？	【思路】：当面试刚刚走出校门的毕业生时（就是那些几乎没有工作经验的应聘者），企业希望录用那些要么学习很快，要么有领导（管理）潜力的毕业生。希望对方有决定能力、毅力（时间加努力等于成功），或是能够看清人的能力，**在介绍兼职时着重突出你的学习能力、工作能力、工作业绩等**
三、工作态度、企业忠诚度考察	
1. 你是否有过工作经验？	【思路】：最好的回答是如实回答自己的真实情况。如果学员没有工作经验的话，依照岗位要求的技术方向以及岗位职责，把重点转移到做过的类似的项目经验上去，让面试官确信你的技术能够胜任这个岗位。 有过工作经验的学员，要把以前的工作经验描述的和应聘的岗位有相同之处 正确回答：之前曾经在 XX 网络科技公司做过网络管理员这个职位，在工作中我充分利用所学知识，对公司的网络环境进行了净化，同时对公司的网络进行了安全加固……，

	在工作当我学到了很多的东西，甚至包括之前我所没有接触过的方面
2“您在前一家公司的离职原因是什么？”	正确回答：我离职是因为这家公司倒闭。我在公司工作了三年多，有较深的感情。从去年始，由于市场形势突变，公司的局面急转直下。到眼下这一步我觉得很遗憾，但还要面对显示，重新寻找能发挥我能力的舞台。 【思路】：1、 最重要的是：应聘者要使找招聘单位相信，应聘者在过往的单位的“离职原因”在此家招聘单位里不存在。 2、避免把“离职原因”说得太详细、太具体。 3、不能掺杂主观的负面感受，如“太幸苦”、“人际关系复杂”、“管理太混乱”、“公司不重视人才”、“公司排斥我们某某的员工”等。 4、但也不能躲闪、回避，如“想换换环境”、“个人原因”等。 5、 不能涉及自己负面的人格特征，如不诚实、懒惰、缺乏责任感、不随和等。 6、尽量使解释的理由为应聘者个人形象添彩。 7、同一个面试问题并非只有一个答案，而同一个答案并不是在任何面试场合都有效，关键在于应聘者掌握了规律后，对面试的具体情况进行把握，有意识地揣摩面试官提出问题的心理背景，然后投其所好。
3.之前的薪水有多少？	正确回答：在以前的单位，差不多一个月薪水是在 1500 到 2000 之间吧 ，因为奖金是看绩效考核的，不过我一般情况的薪水都在 1800 到 1900 之间。 【思路】：之前的薪水可据实以报，切勿写不实数据，因为有些公司会去查证，万一得知所言不实，就可能会丧失工作机会。
4.从你的前任工作中，你所学到的最有意义的两到三件事是什么？	【思路】：你这里要说的就是工作心得或是工作经验，比如使你工作效率提高的例子
5.讲述一些对你的发展贡献最大的事件或事情。请说说你从那些些事件或事故中学到了什么，并且是怎样把所学的知识应用到后来的实际工作中的？	【思路】：这里你所要讲的事情是你遇到的比较积极的事情，比如你看了一位专业成功人士的个人传记，从他的身上懂得了怎么刻苦学习、为人处事、和其他同事的关系以及对工作的狂热，从而运用到自身，把他的经验运用到自己的工作当中，少走了很多弯路，提高了工作效率。
四、性格考察	
1.你有什么优缺点	**考察：自我认识、了解多少。注意缺点讲述。** 【思路】：优点在这个问题上， 面试官关注的问题有两点。第一， 应聘人没有撒谎，而是真实地阐述了自己的优点。第二， 他所阐述的优点， 恰好是这个职位所需要的素质。有很多时候， 对于一个岗位而言的优点， 会成为另一个岗位的缺点。所以， 在

	回答这个问题的时候， 要遵从以下步骤： （1） 找出自己的三至五个优点； （2） 每个优点找出 N 多个例子， 举例最好来自学习、 工作和生活等三个方面， 而不是仅仅来自一个方面； （3） 在这三到五个优点之中， 精选出一两个和所申请职位最吻合的优点。 缺点：与工作无关的缺点。如：1、在开发这块我缺乏工作经验，2、与陌生人讲话紧张。之后补充：我现在意识到这个缺点，正在努力改进当中。
2.谈一谈你的一次失败经历”	**考察：面对困难，挫折时的心态** 【思路】：1、 不宜说自己没有失败的经历。 2、 不宜把那些明显的成功说成是失败。3、不宜说出严重影响所应聘工作的失败经历， 4、 所谈经历的结果应是失败的。 5、宜说明失败之前自己曾信心白倍、尽心尽力。 6、说明仅仅是由于外在客观原因导致失败。 7、 失败后自己很快振作起来，以更加饱满的热情面对以后的工作。如果是外企，你就可以说失败的经历就是追女孩子失败之类~~ 如果是国企，最好编一个故事，随便编什么，比如说学校的运动会，你参加 800 米的比赛啊，你准备成绩跑到多少秒，之前也进行了精心准备，练习，但最后没有跑到那个成绩，所以你发愤练习，终于在第二年的运动会跑进了前三，破了开始预定的目标，，哈哈，我是瞎编一个，我的意思就是告诉你，**虽然让你讲失败的经历，但你讲了之后要让人家听了，虽然失败了，但是你很有精神，另外一种意义上是一种成功，明白了吧，但要注意哦，要编的圆满一点，特别涉及倒数字，一定要准确，不要闹笑话~~~**
3.“你有什么业余爱好？”	正确回答：说起业余爱好，我还真不是很多 ，除了读书，我就比较喜爱打球，在大学的时候我是我们系篮球队的 ，我打的位置是中锋，不过最近几年因为工作的关系渐渐的打的少了 【思路】：1、**业余爱好能在一定程度上反映应聘者的性格、观念、心态**，这是招聘单位问该问题的主要原因。 2、 最好不要说自己没有业余爱好。 3、不要说自己有那些庸俗的、令人感觉不好的爱好。 4、 最好不要说自己仅限于读书、听音乐、上网，否则可能令面试官怀疑应聘者性格孤僻。 5、最好能有一些户外的业余爱好来“点缀”你的形象。
4.谈谈你的家庭	正确回答：我的父亲是做教师工作，母亲是做财会工作的，从小父母对我的学习情况就特别重视。…….. 【思路】 ：了解求职者的性格、观念、心态等，这是该问题的主要作用 2 简单地罗列家庭人口 **3 强调温馨和睦的家庭氛围 4 强调父母对自己教育的重视** 5 强调各位家庭成员的良好状况 **6 强调家庭成员对自己工作的支持** 7 强调自己对家庭的责任感
5. “谈谈你的家庭情况”	【思路】：**1、 况对于了解应聘者的性格、观念、心态等有一定的作用，这是招聘单位问该问题的主要原因。 2、 简单地罗列家庭人口。 3、宜强调温馨和睦的家庭氛围。 4、宜强调父母对自己教育的重视。 5、 宜强调各位家庭成员的良好状况。 6、 宜强调家庭成员对自己工作的支持。 7、 宜强调自己对家庭的责任感。** 正确回答：我父亲是位铁路工作者，母亲是做会计工作的。我从他们身上学到了严谨还有细心，…….. 【思路】：这个问题是招聘方考察你的个人表达能力和认识能力的问题。你可以突出自己的一些优点，但是不要说得过于直白，同时要强调自己的能力比较适合做***工作，这里技巧很重要。同时说话要机智，也可以适当加些小幽默，显示你天生有与人交际的能力和创新能力，这对于***工作来说是很重要的。
6.你的朋友如何评价你	

7.“你的座右铭是什么？”	正确回答：只为成功找方法，不为失败找借口 【思路】：1、座右铭能在一定程度上反映应聘者的性格、观念、心态，这是面试官问这个问题的主要原因。 2、不宜说那些医引起不好联想的座右铭。 3、不宜说那些太抽象的座右铭。 4、不宜说太长的座右铭。 5、座右铭最好能反映出自己某种优秀品质。
8.你的实力是什么？	【思路】：**描述你得专业技能、两三项和该工作最相关的工作经验或项目经验。**避免老生常谈或过分笼统，应提供详细的例子。描述能把这些技能运用于新职位的新方法。例如，指出你仅有他们要求的工作经验是明智的，面试官肯定很重视这些，另外你不仅指出你以前的技能、相关得工作经验，还要说明你能胜任这份工作。
9.你的私人生活是否存在对你的职业造成影响的问题？	【思路】：使面试官深信你能把私人的感受和职业行为分开来。例如，假如你不能积极处理好危急关头的事情，你就不可能真正帮助你的病人面对创伤问题，因为那会引发你太多个人的强烈情绪 正确回答：我总是很注意把自己的私人生活与职业生活分开。我认为在给病人治病时，重要的是保持客观。一个临床医生应该能够控制自己的感情而不带偏见地给病人看病。假如你把个人的问题或感受发泄到病人的身上，那只能对病人造成伤害。这个问题对于在医疗服务领域工作的人尤为重要。面试官想知道作为一个独立个体，你是如何完善你的人格的。假如你有不能克服的个人问题，这些问题就会影响你和你病人的关系，也有可能影响你对病人的诊断，同时影响你的治疗方案和治疗建议。
10. 最能概括你自己的三个词是什么？	【思路】：概括自己得三个词**要结合你所应聘岗位来写**，然后用一个自身例子分贝来来解释这三个词，比如：学习能力强、创新能力强、有团队精神、有责任心等
11.你最崇拜的人是谁	【思路】：对于这个问题，**最好是说一个本行业的知名人士**，你能从这个知名人士身上能学到什么东西，
五、 工作能力考察	
1.“对这项工作，你有哪些可预见的困难？”	正确回答：工作中出现一些困难是正常的，也是难免的，但是只要有坚忍不拔的毅力、良好的合作精神以及事前周密而充分的准备，任何困难都是可以克服的。” 【思路】：1、 不宜直接说出具体的困难，否则可能令对方怀疑应聘者不行。 2、可以尝试迂回战术，说出应聘者对困难所持有的态度
2.“与上级意见不一是，你将怎么办？”	正确回答：首先呢，作为一个员工我的是不会和上级产生争执的，**如果真有意见不一致的时候，我想我会服从领导安排的** 【思路】： 1、一般可以这样回答“我会给上级以必要的解释和提醒，在这种情况下，我会服从上级的意见。” 2、如果面试你的是总经理，而你所应聘的职位另有一位经理，且这位经理当时不在场，可以这样回答：“对于非原则性问题，我会服从上级的意见，对于涉及公司利益的重大问题，我希望能向更高层领导反映。”
3.“我们为什么要录用你？”	正确回答：**我符合贵公司的招聘条件，凭我目前掌握的技能、高度的责任感和良好的饿适应能力及学习能力 ，完全能胜任这份工作。**我十分希望能为贵公司服务，如果贵公司给我这个机会，我一定能成为贵公司的栋梁！” 【思路】：1、应聘者最好站在招聘单位的角度来回答。 2、 招聘单位一般会录用这样的应聘者：基本符合条件、对这份共组感兴趣、有足够的信心
4.“你能为我们做什	正确回答：首先我应聘的是网络管理员，如果贵公司能够录用我的话，首先，我会对公司整个网络环境进行净化，对公司的网络安全状况进行加固……

么？”	【思路】：1、 基本原则上“投其所好”。 2、 回答这个问题前应聘者最好能“先发制人”，了解招聘单位期待这个职位所能发挥的作用。 3、 应聘者可以根据自己的了解，结合自己在专业领域的优势来回答这个问题。
5.“你希望与什么样的上级共事？”	正确回答：做为刚步入社会新人，**我应该多要求自己尽快熟悉环境、适应环境，而不应该对环境提出什么要求，只要能发挥我的专长就可以了。** 【思路】：1、通过应聘者对上级的“希望”可以判断出应聘者对自我要求的意识，这既上一个陷阱，又上一次机会。 2、 最好回避对上级具体的希望，多谈对自己的要求。
6.“你是应届毕业生，缺乏经验，如何能胜任这项工作？”	正确回答：作为应届毕业生，在工作经验方面的确会有所欠缺，**因此在读书期间我一直利用各种机会在这个行业里做兼职**。我也发现，实际工作远比书本知识丰富、复杂。但我有较强的责任心、适应能力和学习能力，而且比较勤奋，所以在兼职中均能圆满完成各项工作，从中获取的经验也令我受益非浅。请贵公司放心，学校所学及兼职的工作经验使我一定能胜任这个职位。 【思路】：1、 如果招聘单位对应届毕业生的应聘者提出这个问题，说明招聘单位并不真正在乎“经验”，关键看应聘者怎样回答。 2、对这个问题的回答最好要体现出应聘者的诚恳、机智、果敢及敬业。
7.你如何规划你个人的职业生涯	正确回答：我对自己进行了一个长期的规划，以及 4 个短期的规划，现在这个阶段我主要是对自己的内职业生涯进行提升 【思路】：**分长期和短期来进行回答。毕业生在求职前一定要对这样的问题有所考虑，**并不仅仅是因为面试时可能被问到，对这个问题的思考有助于为个人树立目标。
8.你希望 5 年后达到什么成就？	A.做一天和尚敲一天钟，尽人事听天命、顺其自然。 B.依我的机灵及才干，晋升到部门经理是我的中期目标。 C.自己独当一面开公司。 D.**“全力以赴”是我的座右铭，希望能随着经验的增加，被赋予更多的职责及挑战。** 解答：最理想的回答是 D。
9.5 年之内你想处于什么位置？	正确回答：我希望有机会在工厂或国内办事处工作。我也希望通过管理一个小团体发展我的管理技能。 【思路】：不要给出具体的时限或工作头衔。谈你喜欢的东西，你天生的技能，实际的问题和在你所选的领域或行业里你希望有什么机会，你希望从那些经验中学点什么。不要谈论你在那些与你所应聘的工作无关的领域或行业里的目标。这是听起来很明显的道理，但是很多求职者会犯这个错误。不经意间你就表现出了对当前的领域或行业缺乏真正的兴趣。不用说，一失言马上就会把你从进一步的考虑中淘汰掉。
10.描述你的理想职业	正确回答：不管发生什么事情，我都愿意在与 IT 有关的领域里工作。和在大学中教学的工作相比，我还是对 IT 感兴趣。但我相信从事 IT 是我的天性，我擅长销售是因为我愿意花时间去教我的客户。现在我热切盼望我能培训那些新招聘进来的人。 【思路】：谈及你喜欢的东西，你天生的技能，实际的问题或在这份特定工作或行业里你所期盼的机会，你希望从那些经验里学点什么东西。避免谈具体的时限和工作头衔。
六、企业忠诚度考察	
1.说说你对行业、技术发展趋势的看法？	【思路】:企业对这个问题很感兴趣，只有有备而来的求职者能够过关。求职者可以直接在网上查找对你所申请的行业部门的信息，只有深入了解才能产生独特的见解。企业认为最聪明的求职者是对所面试的公司预先了解很多，包括公司各个部门，发展情况，在面试回答问题的时候可以提到所了解的情况，企业欢迎进入企业的人是“知己”，而不是“盲人”。

2. 如果上司对你提出了不公平的批评，你会怎么样？	**我觉得无论事情是由谁引起的，如何解决矛盾才是最重要的。**因此，在我们都心平气和的时候，我会主动去找我的上司，**用他能够接受的方式来表达我的想法**。首先，我对他的批评中可能正确的部分表示接受然后再以咨询或征求意见的态度与上司进行沟通，这样矛盾也许会更容易解决一些。”
3. 你找工作最重要的考虑因素是什么	正确回答：我觉得应该是发展和提升，再者就是公司的培训，每个人都有自己的职业规划和定位，二我的规划就是在这段时间内无论是外职业生涯还是内职业生涯，我希望都有所发展。 【思路】：可以结合你正在应聘的工作，侧重谈你的兴趣、你对于取得事业上的成就的渴望、施展你的才能的可能性、未来的发展前景等方面来谈
4. 你认为你适合什么样的工作	正确回答：我大学的专业是计算机应用，主要学习的是网络方面的知识，在我毕业以后我又进行了网络工程师系统的培训，所以我觉得在网络方面的工作我会做的很好 【思路】：**结合你的长处或者专业背景回答，也许单位是结合未来的工作安排来提问，也许只是一般性地了解你对自己的评价，**不要说不知道，也不要说什么都行
5. 你了解我们单位吗	正确回答：之前我在网络上以及朋友的口中了解过咱们单位，咱们单位在网络行业是石家庄的巨头，……. 【思路】 ：只要毕业生提前做些准备，**从多种途径收集用人单位的信息，这样的问题就比较容易回答**，如果答非所问或张口结舌，场面可能会很尴尬
6 “你为什么选择我们公司？”	正确回答：在几家应聘单位中我之所以选择咱们公司，是因为看重咱们公司在业内的影响力以及咱们公司的实力 【思路】：1、 面试官试图从中了解你求职的动机、愿望以及对此项工作的态度。 2、**建议从行业、企业和岗位这三个角度来回答。** 3、 参考答案——“我十分看好贵公司所在的行业，我认为贵公司十分重视人才，而且这项工作很适合我，相信自己一定能做好。”
7. 你会为公司服务多久？	正确回答：只要公司与我双方都觉得我在公司运营中发挥作用，在不断成长成熟，**在公司实现预期目标中作出贡献，我乐意为公司效力。** 【思路】： 把公司的发展与你个人发展结合到一起来说这个问题
8. 希望待遇多少？	正确回答：刚到公司，还是按照公司的岗位工资我就满意了，后期我想咱们单位应该是能力与待遇成正比的 ，对吗? 【思路】：常被问到希望待遇时，最好能诚实回答，考虑年龄、经验及能力等客观条件来决定，对某些企业而言，这也是评论应征者的能力及经验的参考，一般要求比以前一工作薪水高出百分之十是合理范围。
9. 何时可以到职？	**正确回答：如果被录用的话，可按公司规定时间上班** 【思路】： 大多数企业会关心就职时间，最好是回答“如果被录用的话，可按公司规定时间上班，”但如果还未辞去上一个工作、上班时间又太近，似乎有些强人所难，因为交接至少要一个月的时间，应进一步说明原因，录取公司应该会通融的 正确回答：如果被录用的话，可按公司规定时间上班
七、 承受压力考察	
1. 你对加班的看法。	正确回答：**首先我想确认下，是何种性质的加班？** 如果是我个人的工作量是在规定的时间内没有完成的话，这种情况是不会发生的，我是个注重工作效率的人。其次如果是

	公司业务量临时增加的话，我会接受加班。 【思路】：首先，明确的告诉对方，如果是因为自己在规定的时间内没有完成工作任务的话 ，需要加班的情况是几乎不可能出现的，“我是个注重工作效率的人”其次，如果是因为公司业务情况或者其他的一些紧急工作的话是可以适应加班的。
2.经常需要出差的工作适合你的生活方式吗？	正确回答：对我或我的家庭来说，这种经常需要出差的顾问工作是完全没有问题的。不会在意出差的艰辛，相反我会以此为荣。我非常喜欢这份工作，我觉得自己更看重的是这一点。
3. 就你的能力而言，如何让我相信你能够胜任这份工作？	正确回答：就我之前的工作经验而言，我觉得我能够胜任这份工作，网络管理员主要负责公司内部局域网的……..，我在以前的单位，做的正是这些。 【思路】：**这个问题要从所应聘的岗位职责和要求入手，阐述你在这方面的实力**

第三部分 技术面试常见问题

1.JavaSE 常见面试题（50 题）

1、谈谈final, finally, finalize的区别。

final修饰符（关键字）

如果一个类被声明为final，意味着它不能再派生出新的子类，不能作为父类被继承。因此一个类不能既被声明为abstract，又被声明为final。将变量或方法声明为final，可以保证它们在使用中不被改变。其初始化可以在两个地方：一是其定义处，也就是说在final变量定义时直接给其赋值；二是在构造函数中。这两个地方只能选其一，要么在定义时给值，要么在构造函数中给值，不能同时既在定义时给了值，又在构造函数中给另外的值，而在以后的引用中只能读取,不可修改。被声明为final的方法也同样只能使用,不能重写(override)。

finally

在异常处理时提供finally块来执行任何清除操作。如果抛出一个异常，那么相匹配的catch子句就会执行，然后控制就会进入finally块（如果有的话）。

finalize

方法名。Java 技术允许使用finalize()方法在垃圾收集器将对象从内存中清除出去之前做必要的清理工作。这个方法是由垃圾收集器在确定这个对象没有被引用时对这个对象调用的。它是在Object类中定义的，因此所有的类都继承了它。子类覆盖finalize()方法以整理系统资源或者执行其他清理工作。finalize()方法是在垃圾收集器删除对象之前对这个对象调用的。

2、HashMap和Hashtable的区别。

都属于Map接口的类，实现了将惟一键映射到特定的值上。

HashMap 类没有分类或者排序。它允许一个 null 键和多个 null 值。

Hashtable 类似于 HashMap，但是不允许 null 键和 null 值。它也比 HashMap 慢，因为它是同步的。

Hashtable继承自Dictionary类，而HashMap是Java1.2引进的Map interface的一个实现。

HashMap允许将null作为一个entry的key或者value,而Hashtable不允许,还有就是,HashMap把Hashtable的contains方法去掉了，改成containsvalue(Returns true if this map maps one or more keys to the specified value)和containsKey(Returns true if this map contains a mapping for the specified key)。因为contains(Tests if some key maps into the specified value in this hashtable)方法容易让人引起误解。

最大的不同是，Hashtable的方法是Synchronize的，而HashMap不是，在多个线程访问Hashtable时，不需要自己为它的方法实现同步，而HashMap 就必须为之提供外同步。

Hashtable和HashMap采用的hash/rehash算法都大概一样，所以性能不会有很大的差异。

3、Overload和Override的区别。Overloaded的方法是否可以改变返回值的类型？

方法的重写Overriding和重载Overloading是Java多态性的不同表现。重写Overriding是父类与子类之间多态性的一种表现，重载Overloading是一个类中多态性的一种表现。如果在子类中定义某方法与其父类有相同的名称和参数，我们说该方法被重写 (Overriding)。子

类的对象使用这个方法时，将调用子类中的定义，对它而言，父类中的定义如同被"屏蔽"了。如果在一个类中定义了多个同名的方法,它们或有不同的参数个数或有不同的参数类型，则称为方法的重载(Overloading)。Overloaded的方法是可以改变返回值的类型。但是不能通过返回值进行overload。

4、abstract class 和 interface 有什么区别？

声明方法的存在而不去实现它的类被叫做抽象类（abstract class），它用于要创建一个体现某些基本行为的类，并为该类声明方法，但不能在该类中实现该类的情况。不能创建 abstract 类的实例。然而可以创建一个变量，其类型是一个抽象类，并让它指向具体子类的一个实例。不能有抽象构造函数或抽象静态方法。Abstract 类的子类为它们父类中的所有抽象方法提供实现，否则它们也是抽象类为。取而代之，在子类中实现该方法。知道其行为的其它类可以在类中实现这些方法。

接口（interface）是抽象类的变体。在接口中，所有方法都是抽象的。多继承性可通过实现这样的接口而获得。接口中的所有方法都是抽象的，没有一个有程序体。接口只可以定义 static final 成员变量。接口的实现与子类相似，除了该实现类不能从接口定义中继承行为。当类实现特殊接口时，它定义（即将程序体给予）所有这种接口的方法。然后，它可以在实现了该接口的类的任何对象上调用接口的方法。由于有抽象类，它允许使用接口名作为引用变量的类型。通常的动态联编将生效。引用可以转换到接口类型或从接口类型转换，instanceof 运算符可以用来决定某对象的类是否实现了接口。

5、运行时异常与一般异常有何异同？

异常表示程序运行过程中可能出现的非正常状态,运行时异常表示虚拟机的通常操作中可能遇到的异常，是一种常见运行错误。java编译器要求方法必须声明抛出可能发生的非运行时异常，但是并不要求必须声明抛出未被捕获的运行时异常。

6、事务是什么？

事务是作为一个逻辑单元执行的一系列操作,一个逻辑工作单元必须有四个属性,称为 ACID（原子性、一致性、隔离性和持久性）属性，只有这样才能成为一个事务：

原子性

事务必须是原子工作单元；对于其数据修改，要么全都执行，要么全都不执行。

一致性

事务在完成时，必须使所有的数据都保持一致状态。在相关数据库中，所有规则都必须应用于事务的修改，以保持所有数据的完整性。事务结束时，所有的内部数据结构（如 B 树索引或双向链表）都必须是正确的。

隔离性

由并发事务所作的修改必须与任何其它并发事务所作的修改隔离。事务查看数据时数据所处的状态，要么是另一并发事务修改它之前的状态，要么是另一事务修改它之后的状态，事务不会查看中间状态的数据。这称为可串行性，因为它能够重新装载起始数据，并且重播一系列事务，以使数据结束时的状态与原始事务执行的状态相同。

持久性

事务完成之后，它对于系统的影响是永久性的。该修改即使出现系统故障也将一直保持。

7、说出ArrayList,Vector, LinkedList的存储性能和特性

ArrayList 和 Vector 都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，它们都允许直接按序号索引元素，但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，Vector 由于使用了 synchronized 方法（线程安全），通常性能上较 ArrayList 差，而 LinkedList 使用双向链表实现存储，按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

8、同步和异步有何异同，在什么情况下分别使用他们？举例说明。

如果数据将在线程间共享。例如正在写的数据以后可能被另一个线程读到，或者正在读的数据可能已经被另一个线程写过了，那么这些数据就是共享数据，必须进行同步存取。当应用程序在对象上调用了一个需要花费很长时间来执行的方法，并且不希望让程序等待方法的返回时，就应该使用异步编程，在很多情况下采用异步途径往往更有效率。

9、GC是什么？为什么要有GC?

GC 是垃圾收集的意思（Gabage Collection），内存处理是编程人员容易出现问题的地方，忘记或者错误的内存回收会导致程序或系统的不稳定甚至崩溃，Java 提供的 GC 功能可以自动监测对象是否超过作用域从而达到自动回收内存的目的，Java 语言没有提供释放已分配内存的显示操作方法。

10、设计 4 个线程，其中两个线程每次对 j 增加 1，另外两个线程对 j 每次减少 1。写出程序。

以下程序使用内部类实现线程，对 j 增减的时候没有考虑顺序问题。

```
public class ThreadTest1{private int j;  public static void main(String args[])
{  ThreadTest1 tt=new ThreadTest1();  Inc inc=tt.new Inc(); Dec dec=tt.new De
c();  for(int i=0;i<2;i++){  Thread t=new Thread(inc);  t.start();  t=new T
hread(dec);  t.start();  } }  private synchronized void inc(){  j++;  Sy
stem.out.println(Thread.currentThread().getName()+"-inc:"+j);  }
  private synchronized void dec(){ j--;  System.out.println(Thread.currentThr
ead().getName()+"-dec:"+j);  }
class Inc implements Runnable{  public void run(){  for(int i=0;i<100;i++)
{  inc();  } }  } class Dec implements Runnable{public void run(){  for(int
 i=0;i<100;i++){ dec(); } }  } }
```

11、给我一个你最常见到的runtime exception。

ArithmeticException, ArrayStoreException, BufferOverflowException, BufferUnderflowException, CannotRedoException, CannotUndoException, ClassCastException, CMMException, ConcurrentModificationException, DOMException, EmptyStackException, IllegalArgumentException, IllegalMonitorStateException, IllegalPathStateException, IllegalStateException, ImagingOpException, IndexOutOfBoundsException, MissingResourceException, NegativeArraySizeException, NoSuchElementException, NullPointerException, ProfileDataException, ProviderException, RasterFormatException, SecurityException, SystemException, UndeclaredThrowableException, UnmodifiableSetException, UnsupportedOperationException

12、Static Nested Class 和 Inner Class 的不同

Nested Class （一般是 C++的说法），Inner Class (一般是 JAVA 的说法)。Java 内部类与 C++嵌套类最大的不同就在于是否有指向外部的引用上。注： 静态内部类（Inner Class）意味着 1 创建一个 static 内部类的对象，不需要一个外部类对象，2 不能从一个 static 内部类的一个对象访问一个外部类对象

13、STRING 与 STRINGBUFFER 的区别。

STRING 的长度是不可变的，STRINGBUFFER 的长度是可变的。如果你对字符串中的内容经常进行操作，特别是内容要修改时，那么使用 StringBuffer，如果最后需要 String，那么使用 StringBuffer 的 toString()方法

14、面向对象的特征有哪些方面

主要有以下四方面：

1.抽象：

抽象就是忽略一个主题中与当前目标无关的那些方面，以便更充分地注意与当前目标有关的方面。抽象并不打算了解全部问题，而只是选择其中的一部分，暂时不用部分细节。抽象包括两个方面，一是过程抽象，二是数据抽象。

2.继承：

继承是一种联结类的层次模型，并且允许和鼓励类的重用，它提供了一种明确表述共性的方法。对象的一个新类可以从现有的类中派生，这个过程称为类继承。新类继承了原始类的特性，新类称为原始类的派生类（子类），而原始类称为新类的基类（父类）。派生类可以从它的基类那里继承方法和实例变量，并且类可以修改或增加新的方法使之更适合特殊的需要。

3.封装：

封装是把过程和数据包围起来，对数据的访问只能通过已定义的界面。面向对象计算始于这个基本概念，即现实世界可以被描绘成一系列完全自治、封装的对象，这些对象通过一个受保护的接口访问其他对象。

4. 多态性：

多态性是指允许不同类的对象对同一消息作出响应。多态性包括参数化多态性和包含多态性。多态性语言具有灵活、抽象、行为共享、代码共享的优势，很好的解决了应用程序函数同名问题。

15、垃圾回收的优点和原理。并考虑 2 种回收机制

Java 语言中一个显著的特点就是引入了垃圾回收机制，使 c++程序员最头疼的内存管理的问题迎刃而解，它使得 Java 程序员在编写程序的时候不再需要考虑内存管理。由于有个垃圾回收机制，Java 中的对象不再有"作用域"的概念，只有对象的引用才有"作用域"。垃圾回收可以有效的防止内存泄露，有效的使用可以使用的内存。垃圾回收器通常是作为一个单独的低级别的线程运行，不可预知的情况下对内存堆中已经死亡的或者长时间没有使用的对象进行清楚和回收，程序员不能实时的调用垃圾回收器对某个对象或所有对象进行垃圾回收。回收机制有分代复制垃圾回收和标记垃圾回收，增量垃圾回收。

16、JAVA 语言如何进行异常处理，关键字：throws,throw,try,catch,finally 分别代表什么意义？在 try 块中可以抛出异常吗？

Java 通过面向对象的方法进行异常处理，把各种不同的异常进行分类，并提供了良好的接口。在 Java 中，每个异常都是一个对象，它是 Throwable 类或其它子类的实例。当一个方法出现异常后便抛出一个异常对象，该对象中包含有异常信息，调用这个对象的方法可以捕获到这个异常并进行处理。Java 的异常处理是通过 5 个关键词来实现的：try、catch、throw、throws 和 finally。一般情况下是用 try 来执行一段程序，如果出现异常，系统会抛出

（throws）一个异常，这时候你可以通过它的类型来捕捉（catch）它，或最后（finally）由缺省处理器来处理。
用 try 来指定一块预防所有"异常"的程序。紧跟在 try 程序后面，应包含一个 catch 子句来指定你想要捕捉的"异常"的类型。
throw 语句用来明确地抛出一个"异常"。
throws 用来标明一个成员函数可能抛出的各种"异常"。
Finally 为确保一段代码不管发生什么"异常"都被执行一段代码。
可以在一个成员函数调用的外面写一个 try 语句，在这个成员函数内部写另一个 try 语句保护其他代码。每当遇到一个 try 语句，"异常"的框架就放到堆栈上面，直到所有的 try 语句都完成。如果下一级的 try 语句没有对某种"异常"进行处理，堆栈就会展开，直到遇到有处理这种"异常"的 try 语句。

17、什么是 java 序列化，如何实现 java 序列化？

序列化就是一种用来处理对象流的机制，所谓对象流也就是将对象的内容进行流化。可以对流化后的对象进行读写操作，也可将流化后的对象传输于网络之间。序列化是为了解决在对对象流进行读写操作时所引发的问题。
序列化的实现：将需要被序列化的类实现 Serializable 接口，该接口没有需要实现的方法，implements Serializable 只是为了标注该对象是可被序列化的，然后使用一个输出流(如：FileOutputStream)来构造一个 ObjectOutputStream(对象流)对象，接着，使用 ObjectOutputStream 对象的 writeObject(Object obj)方法就可以将参数为 obj 的对象写出(即保存其状态)，要恢复的话则用输入流。

18、垃圾回收器的基本原理是什么？垃圾回收器可以马上回收内存吗？有什么办法主动通知虚拟机进行垃圾回收？

对于 GC 来说，当程序员创建对象时，GC 就开始监控这个对象的地址、大小以及使用情况。通常，GC 采用有向图的方式记录和管理堆(heap)中的所有对象。通过这种方式确定哪些对象是"可达的"，哪些对象是"不可达的"。当 GC 确定一些对象为"不可达"时，GC 就有责任回收这些内存空间。可以。程序员可以手动执行 System.gc()，通知 GC 运行，但是 Java 语言规范并不保证 GC 一定会执行。

19、abstract class和interface有什么区别?

声明方法的存在而不去实现它的类被叫做抽象类（abstract class），它用于要创建一个体现某些基本行为的类，并为该类声明方法，但不能在该类中实现该类的情况。不能创建 abstract 类的实例。然而可以创建一个变量，其类型是一个抽象类，并让它指向具体子类的一个实例。不能有抽象构造函数或抽象静态方法。Abstract 类的子类为它们父类中的所有抽象方法提供实现，否则它们也是抽象类为。取而代之，在子类中实现该方法。知道其行为的其它类可以在类中实现这些方法。接口（interface）是抽象类的变体。在接口中，所有方法都是抽象的。多继承性可通过实现这样的接口而获得。接口中的所有方法都是抽象的，没有一个有程序体。接口只可以定义 static final 成员变量。接口的实现与子类相似，除了该实现类不能从接口定义中继承行为。当类实现特殊接口时，它定义（即将程序体给予）所有这种接口的方法。然后，它可以在实现了该接口的类的任何对象上调用接口的方法。由于有抽象类，它允许使用接口名作为引用变量的类型。通常的动态联编将生效。引用可以转换到接口类型或从接口类型转换，instanceof 运算符可以用来决定某对象的类是否实现了接口。

20、Static Nested Class 和 Inner Class 的不同，说得越多越好

Nested Class （一般是 C++的说法），Inner Class (一般是 JAVA 的说法)。Java 内部类与 C++嵌套类最大的不同就在于是否有指向外部的引用上。具体可见 http:

//www.frontfree.net/articles/services/view.asp?id=704&page=1

注： 静态内部类（Inner Class）意味着 1 创建一个 static 内部类的对象，不需要一个外部类对象，2 不能从一个 static 内部类的一个对象访问一个外部类对象

21、&和&&的区别。

&是位运算符。&&是布尔逻辑运算符。

22、short s1 = 1; s1 = s1 + 1;有什么错? short s1 = 1; s1 += 1;有什么错?

short s1 = 1; s1 = s1 + 1;有错，s1 是 short 型，s1+1 是 int 型,不能显式转化为 short 型。可修改为 s1 =(short)(s1 + 1) 。short s1 = 1; s1 += 1 正确。

23、sleep() 和 wait() 有什么区别? 搞线程的最爱

sleep()方法是使线程停止一段时间的方法。在 sleep 时间间隔期满后，线程不一定立即恢复执行。这是因为在那个时刻，其它线程可能正在运行而且没有被调度为放弃执行，除非

(a)“醒来”的线程具有更高的优先级

(b)正在运行的线程因为其它原因而阻塞。

wait()是线程交互时，如果线程对一个同步对象 x 发出一个 wait()调用，该线程会暂停执行，被调对象进入等待状态，直到被唤醒或等待时间到。

24、error 和 exception 有什么区别?

error 表示恢复不是不可能但很困难的情况下的一种严重问题。比如说内存溢出。不可能指望程序能处理这样的情况。

exception 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。

25、启动一个线程是用 run()还是 start()?

启动一个线程是调用 start()方法，使线程所代表的虚拟处理机处于可运行状态，这意味着它可以由 JVM 调度并执行。这并不意味着线程就会立即运行。run()方法可以产生必须退出的标志来停止一个线程。

26、当一个对象被当作参数传递到一个方法后，此方法可改变这个对象的属性，并可返回变化后的结果，那么这里到底是值传递还是引用传递?

是值传递。Java 编程语言只由值传递参数。当一个对象实例作为一个参数被传递到方法中时，参数的值就是对该对象的引用。对象的内容可以在被调用的方法中改变，但对象的引用是永远不会改变的。

编程题: 写一个Singleton出来。

Singleton模式主要作用是保证在Java应用程序中，一个类Class只有一个实例存在。

一般Singleton模式通常有几种种形式:

第一种形式: 定义一个类，它的构造函数为private的，它有一个static的private的该类变量，在类初始化时实例话，通过一个public的getInstance方法获取对它的引用,继而调用其中的方法。

```
public class Singleton {
    private Singleton(){}
    //在自己内部定义自己一个实例，是不是很奇怪?
    //注意这是private 只供内部调用
    private static Singleton instance = new Singleton();
    //这里提供了一个供外部访问本class的静态方法，可以直接访问
    public static Singleton getInstance() {
        return instance;
     }
```

```
}
```

第二种形式:

```
public class Singleton {
    private static Singleton instance = null;
    public static synchronized Singleton getInstance() {
    //这个方法比上面有所改进，不用每次都进行生成对象，只是第一次
    //使用时生成实例，提高了效率!
    if (instance==null)
        instance＝new Singleton();
return instance;     }
}
```

其他形式:

定义一个类，它的构造函数为private的，所有方法为static的。

一般认为第一种形式要更加安全些

27、int 和 Integer 有什么区别

Java 提供两种不同的类型：引用类型和原始类型（或内置类型）。Int 是 java 的原始数据类型，Integer 是 java 为 int 提供的封装类。Java 为每个原始类型提供了封装类。

原始类型封装类

booleanBoolean

charCharacter

byteByte

shortShort

intInteger

longLong

floatFloat

doubleDouble

引用类型和原始类型的行为完全不同，并且它们具有不同的语义。引用类型和原始类型具有不同的特征和用法，它们包括：大小和速度问题，这种类型以哪种类型的数据结构存储，当引用类型和原始类型用作某个类的实例数据时所指定的缺省值。对象引用实例变量的缺省值为 null，而原始类型实例变量的缺省值与它们的类型有关。

28、String 和 StringBuffer 的区别

JAVA 平台提供了两个类：String 和 StringBuffer，它们可以储存和操作字符串，即包含多个字符的字符数据。这个 String 类提供了数值不可改变的字符串。而这个 StringBuffer 类提供的字符串进行修改。当你知道字符数据要改变的时候你就可以使用 StringBuffer。典型地，你可以使用 StringBuffers 来动态构造字符数据。

29、说出 ArrayList,Vector, LinkedList 的存储性能和特性

ArrayList 和 Vector 都是使用数组方式存储数据，此数组元素数大于实际存储的数据以便增加和插入元素，它们都允许直接按序号索引元素，但是插入元素要涉及数组元素移动等内存操作，所以索引数据快而插入数据慢，Vector 由于使用了 synchronized 方法（线程安全），通常性能上较 ArrayList 差，而 LinkedList 使用双向链表实现存储，按序号索引数据需要进行前向或后向遍历，但是插入数据时只需要记录本项的前后项即可，所以插入速度较快。

30、Collection 和 Collections 的区别。

Collection 是集合类的上级接口，继承与他的接口主要有 Set 和 List.

Collections 是针对集合类的一个帮助类，他提供一系列静态方法实现对各种集合的搜索、

排序、线程安全化等操作。

31、error 和 exception 有什么区别?

error 表示恢复不是不可能但很困难的情况下的一种严重问题。比如说内存溢出。不可能指望程序能处理这样的情况。

exception 表示一种设计或实现问题。也就是说，它表示如果程序运行正常，从不会发生的情况。

32、数组有没有 length()这个方法? String 有没有 length()这个方法?

数组没有 length()这个方法，有 length 的属性。String 有有 length()这个方法。

33、swtich 是否能作用在 byte 上，是否能作用在 long 上，是否能作用在 String 上?

switch（expr1）中，expr1 是一个整数表达式。因此传递给 switch 和 case 语句的参数应该是 int、 short、 char 或者 byte。long,string 都不能作用于 swtich。

34、try {}里有一个 return 语句，那么紧跟在这个 try 后的 finally {}里的 code 会不会被执行，什么时候被执行，在 return 前还是后?

会执行，在 return 前执行。

35、描述一下 JVM 加载 class 文件的原理机制?

JVM 中类的装载是由 ClassLoader 和它的子类来实现的,Java ClassLoader 是一个重要的 Java 运行时系统组件。它负责在运行时查找和装入类文件的类。

36、排序都有哪几种方法？请列举。用 JAVA 实现一个快速排序。

排序的方法有：插入排序（直接插入排序、希尔排序），交换排序（冒泡排序、快速排序），选择排序（直接选择排序、堆排序），归并排序，分配排序（箱排序、基数排序）

快速排序的伪代码。

/ /使用快速排序方法对 a[0 :n- 1]排序

从 a[0 :n- 1]中选择一个元素作为 m i d d l e，该元素为支点

把余下的元素分割为两段 left 和 r i g h t，使得 l e f t 中的元素都小于等于支点，而 right 中的元素都大于等于支点

递归地使用快速排序方法对 left 进行排序

递归地使用快速排序方法对 right 进行排序

所得结果为 l e f t + m i d d l e + r i g h t

37、java中有几种方法可以实现一个线程？用什么关键字修饰同步方法? stop()和 suspend()方法为何不推荐使用？

有两种实现方法，分别是继承Thread类与实现Runnable接口

用synchronized关键字修饰同步方法

反对使用stop()，是因为它不安全。它会解除由线程获取的所有锁定，而且如果对象处于一种不连贯状态，那么其他线程能在那种状态下检查和修改它们。结果很难检查出真正的问题所在。suspend()方法容易发生死锁。调用suspend()的时候，目标线程会停下来，但却仍然持有在这之前获得的锁定。此时，其他任何线程都不能访问锁定的资源，除非被"挂起"的线程恢复运行。对任何线程来说，如果它们想恢复目标线程，同时又试图使用任何一个锁定的资源，就会造成死锁。所以不应该使用suspend()，而应在自己的Thread类中置入一个标志，指出线程应该活动还是挂起。若标志指出线程应该挂起，便用wait()命其进入等待状态。若标志指出线程应当恢复，则用一个notify()重新启动线程。

38、java中有几种类型的流？JDK为每种类型的流提供了一些抽象类以供继承，请说出他们分别是哪些类？

字节流，字符流。字节流继承于InputStream \ OutputStream，字符流继承于InputStreamReader \ OutputStreamWriter。在java.io包中还有许多其他的流，主要是

为了提高性能和使用方便。

39、java 中会存在内存泄漏吗，请简单描述。

会。如：int i,i2; return (i-i2); //when i 为足够大的正数,i2 为足够大的负数。结果会造成溢位，导致错误。

40、java 中实现多态的机制是什么？

方法的重写 Overriding 和重载 Overloading 是 Java 多态性的不同表现。重写 Overriding 是父类与子类之间多态性的一种表现，重载 Overloading 是一个类中多态性的一种表现。

41、说出一些常用的类，包，接口，请各举 5 个

常用的类：
BufferedReader BufferedWriter FileReader FileWirter String Integer

常用的包：java.lang java.awt java.io java.util java.sql

常用的接口：Remote List Map Document NodeList

42、什么时候用 assert。

assertion(断言)在软件开发中是一种常用的调试方式,很多开发语言中都支持这种机制。在实现中，assertion 就是在程序中的一条语句，它对一个 boolean 表达式进行检查，一个正确程序必须保证这个 boolean 表达式的值为 true；如果该值为 false，说明程序已经处于不正确的状态下，系统将给出警告或退出。一般来说，assertion 用于保证程序最基本、关键的正确性。assertion 检查通常在开发和测试时开启。为了提高性能，在软件发布后，assertion 检查通常是关闭的。

43、Java 有没有 goto?

java 中的保留字，现在没有在 java 中使用。

44、接口是否可继承接口？抽象类是否可实现(implements)接口？抽象类是否可继承实体类(concrete class)？

接口可以继承接口。抽象类可以实现(implements)接口，抽象类是否可继承实体类，但前提是实体类必须有明确的构造函数。

45、abstract 的 method 是否可同时是 static,是否可同时是 native，是否可同时是 synchronized?

都不能

46、数组有没有 length()这个方法? String 有没有 length()这个方法？

数组没有 length()这个方法，有 length 的属性。String 有有 length()这个方法。

47、两个对象值相同(x.equals(y) == true)，但却可有不同的 hash code，这句话对不对?

不对，有相同的 hash code。

48、请说出你所知道的线程同步的方法。

wait():使一个线程处于等待状态，并且释放所持有的对象的 lock。

sleep():使一个正在运行的线程处于睡眠状态，是一个静态方法，调用此方法要捕捉 InterruptedException 异常。

notify():唤醒一个处于等待状态的线程，注意的是在调用此方法的时候，并不能确切的唤醒某一个等待状态的线程，而是由 JVM 确定唤醒哪个线程，而且不是按优先级。

Allnotity():唤醒所有处入等待状态的线程，注意并不是给所有唤醒线程一个对象的锁，而是让它们竞争。

49、简述 synchronized 和 java.util.concurrent.locks.Lock 的异同 ？

主要相同点：Lock 能完成 synchronized 所实现的所有功能

主要不同点：Lock 有比 synchronized 更精确的线程语义和更好的性能。synchronized 会自

动释放锁，而 Lock 一定要求程序员手工释放，并且必须在 finally 从句中释放。

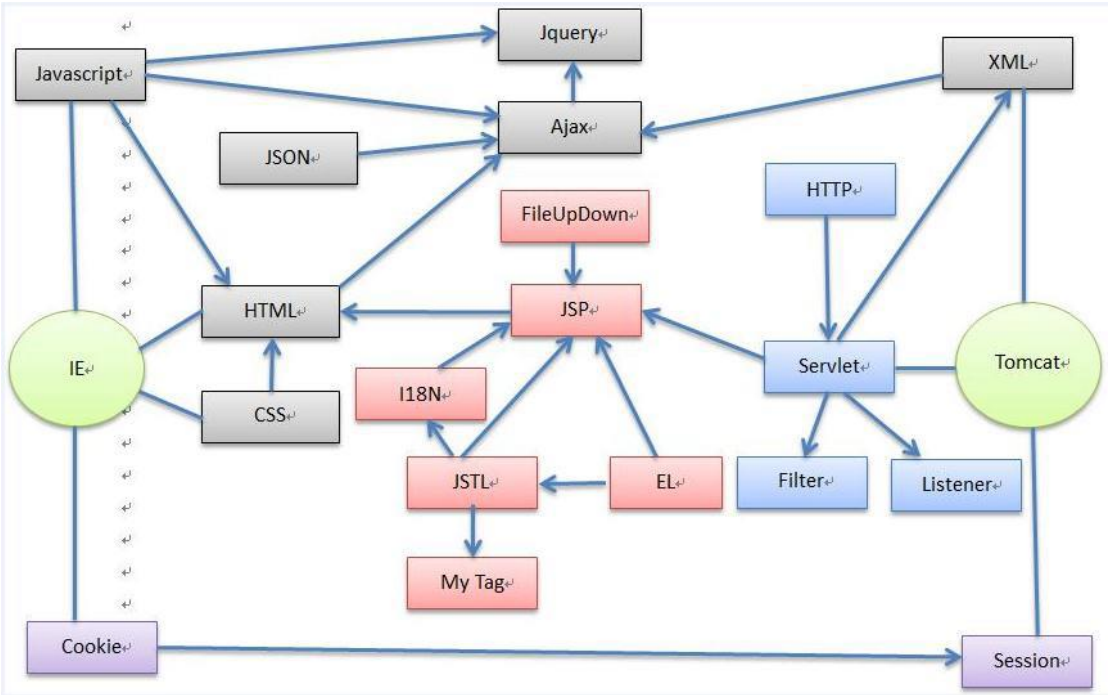
50、Jdo 是什么?

JDO 是 Java 对象持久化的新的规范，为 java data object 的简称,也是一个用于存取某种数据仓库中的对象的标准化 API。JDO 提供了透明的对象存储，因此对开发人员来说，存储数据对象完全不需要额外的代码（如 JDBC API 的使用）。这些繁琐的例行工作已经转移到 JDO 产品提供商身上，使开发人员解脱出来，从而集中时间和精力在业务逻辑上。另外，JDO 很灵活，因为它可以在任何数据底层上运行。JDBC 只是面向关系数据库（RDBMS）JDO 更通用，提供到任何数据底层的存储功能，比如关系数据库、文件、XML 以及对象数据库（ODBMS）等等，使得应用可移植性更强。

2.JavaWEB 常见面试题（34 题）

1. Javaweb 技术的结构

1.1. Javaweb 技术结构图



1.2. 结构图说明:

整体分为四个部分:

1. 黑线: JavaScript 相关技术路线
2. 蓝线: Servlet 相关技术路线
3. 红线: Jsp 相关技术路线
4. 紫线: Web 会话相关技术路线

2. JavaScript 相关技术路线(黑线)

此部分包括: JavaScript, Jquery, Ajax, XML, JSON 和HTML 等技术.

2.1. 列举BOM 中常用的几个全局变量和全局方法?

全局对象: window

全局变量： document location history navigator screen

全局方法：alert() confirm() prompt() open() close()

2.2. 在js 中如何创建一个对象?

```
var p1 = {name: "Tom", "my age" : 12};

function Person(name, age) {
    this.name = name;
    this.age = age;
}
var p2 = new Person("Jack", 14);
```

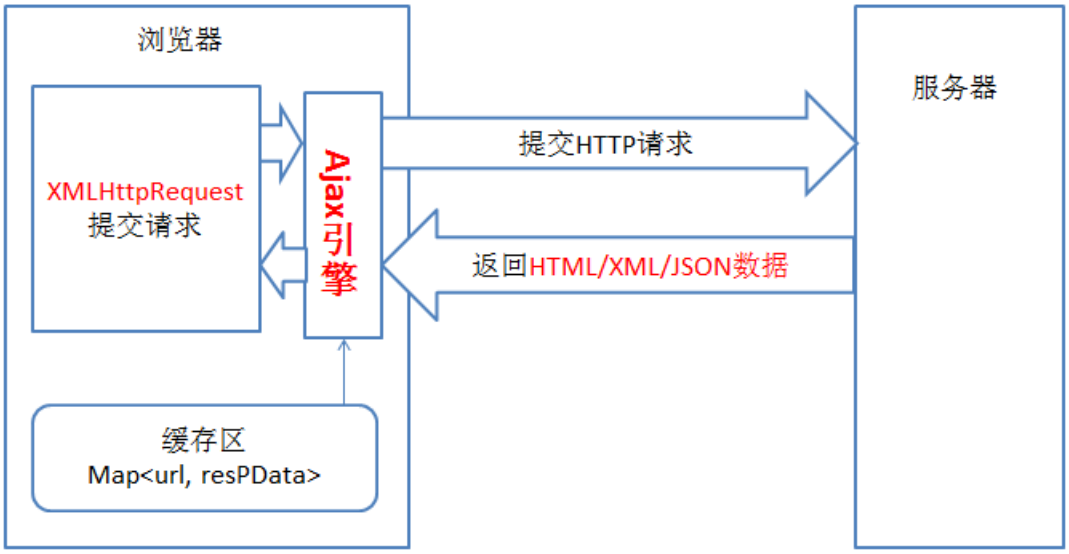
2.3. 在js 中如何得到对象的属性?

```
var age = p2.age
//alert(age);
age = p1["my age"];
alert(age);
```

2.4. 谈谈Ajax 技术

Ajax 原理

Ajax 的原理简单来说通过XmlHttpRequest 对象来向服务器发异步请求，从服务器获得数据，然后用javascript 来操作DOM 而更新页面的局部显示。



Ajax 的优点：

1.最大的一点是页面无刷新，给用户的体验非常好。

2.使用异步方式与服务器通信，不需要打断用户操作，具有更加迅速的响应能力。

3.ajax 的原则是“按需取数据”，最大程度的减少冗余请求，减少服务器的负荷。

Ajax 的缺点：

1.破坏浏览器后退按钮的正常行为。在动态更新页面后，用户无法回到前一个页面的状态.

2.使用JavaScript 作Ajax 的引擎，JavaScript 的兼容性和Debug 本身就让人头大。

Ajax 的应用场景：

1.文本输入提示（自动完成）的场景(注册)

2.对数据进行联动过滤的场景(三级联动)

2.5. 你觉得jquery 有哪些好处?

jQuery 是轻量级的javascript 框架

强大的选择器

出色的DOM 操作的封装

可靠的事件处理机制
完善的ajax 封装
出色的浏览器的兼容性
支持链式操作，隐式迭代
支持丰富的插件
jquery 的文档也非常的丰富

2.6. jquery 对象和dom 对象如何转换？

1. jquery 转DOM 对象:
jQuery 对象是一个数组对象，可以通过[index]的丰富得到DOM 对象还可以通过get[index]去得到相应的DOM 对象。
2. DOM 对象转jQuery 对象:
$(DOM 对象)

2.7. jquery 中$.get()提交和$.post()提交的区别？

1. $.get() 方法使用GET 方式提交请求,而$.post()使用POST 方式。
2. GET 方式传输的数据大小不能超过2KB 而POST 要大的多
3. GET 方式请求的数据会被浏览器缓存起来，因此有安全问题。

2.8. $(document).ready()方法和window.onload 区别？

答: 两个方法有相似的功能，但是在实行时机方面是有区别的。
1 window.onload 方法是在网页中所有的元素(包括元素的所有关联文件)完全加载到浏览器后才执行的。
2 $(document).ready() 方法可以在DOM 载入就绪时就对其进行操纵，并调用执行绑定的函数。

2.9. xml 有哪些解析技术?区别是什么？

答:有DOM,DOM4j,SAX,PULL 等
DOM: 一次性将整个文档加载到内存中，生成一个对象树，在处理大型文件时其性能下降的非常厉害。
DOM4J: 对DOM 的进一步封装，API 使用更简洁
SAX:基于事件驱动的方法回调机制。每读取一小部分数据时就会回调事件处理器对象的方法，但解析一旦开始就不能停止.
PULL: 也是基于事件驱动，只是需要手动控制读取下一部分数据,这样得到想要的数据后就可以停止解析.

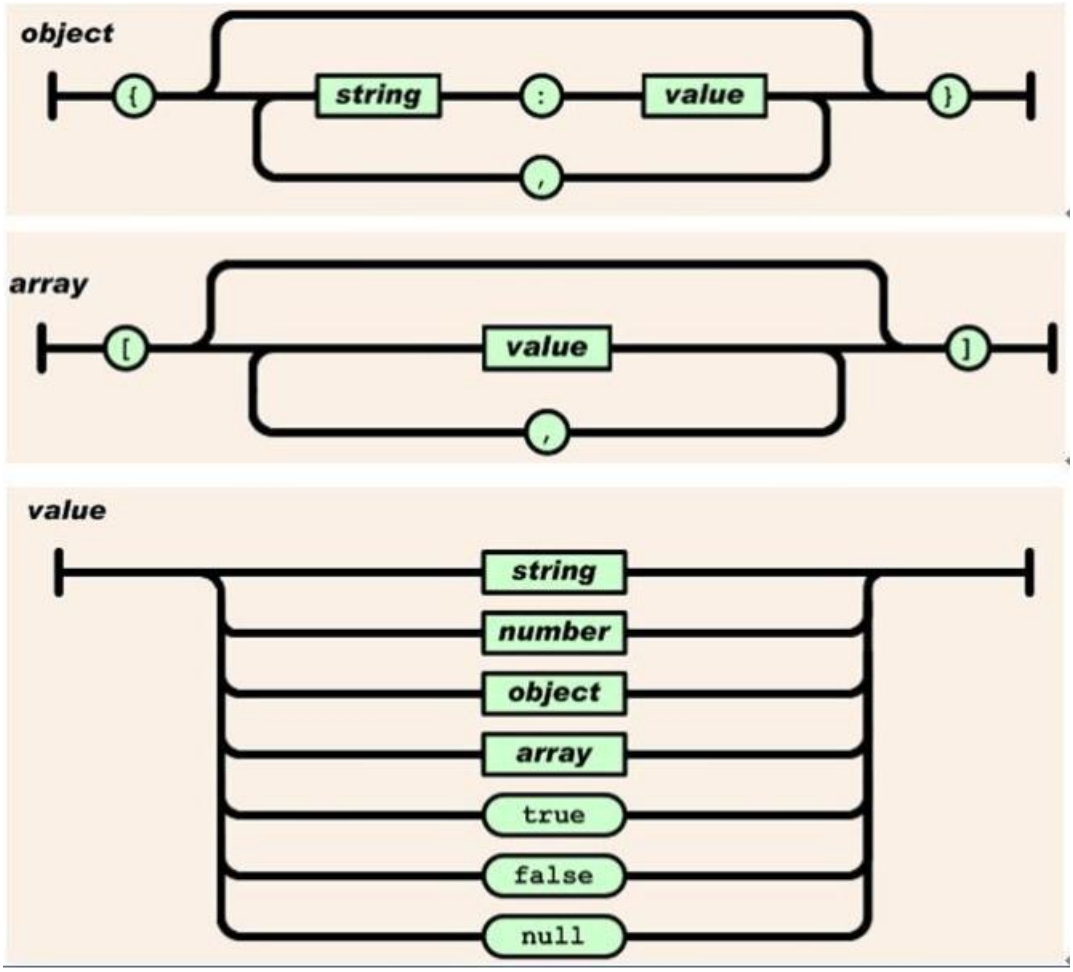
2.10. 你在项目中用到了xml 技术的哪些方面?如何实现的?

答:用到了数据存贮，信息配置两方面。在做数据交换平台时，将不能数据源的数据组装成XML文件，然后将XML文件压缩打包加密后通过网络传送给接收者，接收解密与解压缩后再同XML 文件中还原相关信息进行处理。在做软件配置时，利用XML 可以很方便的进行，软件的各种配置参数都存贮在XML 文件中。

2.11. 说说你对JSON 的理解

JSON(JavaScript Object Notation) 是一种轻量级的数据交换格式。它基于标准JavaScript 的一个子集,是一个Js 对象或数组结构的字符串
JSON 有三类数据
1. 单个数据
有number, string, boolean 和null 四种类型数据
2. 多个有序的数据: 数组
用[]包含起来，其元素可以是三类数据中的任意一种，元素之间用,号隔开

3. 多个无序的数据：对象
用{}包含起来，其元素必须由key-value 组成，key 是一个字符串，value 可以是任意类型数据，key 与value 之间用:号隔开，两个key-value 之间用,号隔开.



3.Servlet 相关技术路线(蓝线)

此部分包括：Servlet，Filter，Listener 和HTTP 协议

3.1. 解释一下什么是servlet?

答：我们可以从下面二个方面去看Servlet：
1. API：有一个接口Servlet，它是Servlet 规范中定义的用来处理客户端请求的程序需要实现的顶级接口
2. 组件：服务器端用来处理客户端请求的组件，需要在web.xml 请求中配置

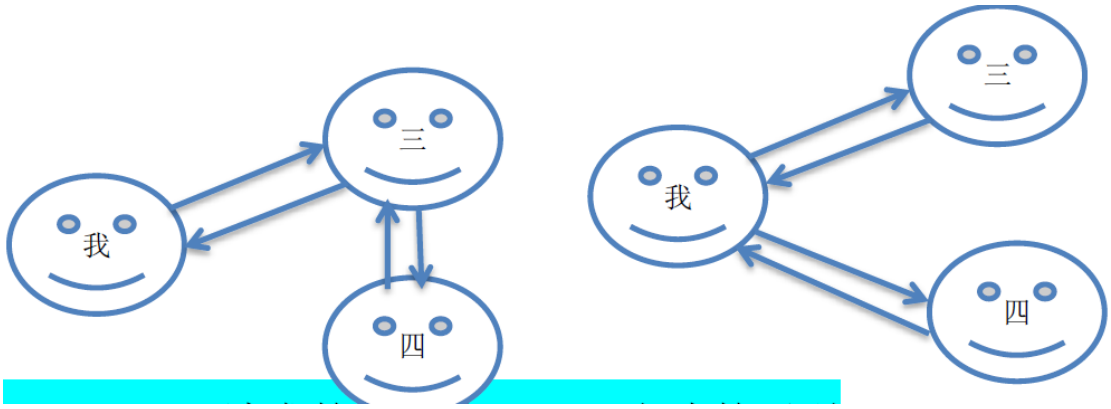
3.2. 说一说Servlet 的生命周期?

答：Servlet 生命周期分为三个阶段：
1，初始化阶段 调用init()方法
2，响应客户请求阶段 调用service()方法- doGet/doPost()
3，终止阶段 调用destroy()方法

3.3. 区别请求的转发与重定向?

答：可以从以下三个方面进行比较
1.地址栏：
转发：显示的是请求的URL
重定向：显示的不是请求的URL，而是重定向指向的新的URL

2.浏览器发了几次请求?
转发：1 次请求
重定向：2 次请求
3. 是否可以进行Request 的数据共享?
转发：两个资源之间是同一个request 对象，可以共享request 中的数据
重定向：两个资源之间不是同一个request 对象，不可以共享
经典现实案例：



3.4. HTTP 请求的GET 与POST 方式的区别

答：可以从以下几个方面去回答：
1. 携带请求参数的方式
GET：通过请求行携带参数，参数会显示在地址栏
POST：通过请求体来携带参数，参数不会显示在地址栏
2. 服务器端处理请求的方法
GET：会调用Servlet 的doGet()来处理请求
POST：会调用Servlet 的doPost()来处理请求
3. 数据大小与安全性
GET：大小有限制(小于2k)，不安全
POST：大小没有限制，安全

3.5. 比较一下Servlet 与Filter

1. Filter 是一种特别的Servlet，它们的作用是完全不一样的. Servlet 是用来处理请求的，而Filter 是用来过滤检查请求的.
经典现实案例：
假如我们要去坐地铁去天安门，我们需要先在检票机上刷票后才能进站坐上地铁，请求问：在这个实际业务中，哪个是Servlet?哪个是Filter 呢?

4. Jsp 相关技术路线(红线)

此部分包括：JSP，EL，JSTL，My Tag，I18N，FileUpDown

4.1. jsp 有哪些内置对象?作用分别是什么?

答:JSP 共有以下9 个内置的对象：
request：用户端请求，此请求会包含来自GET/POST 请求的参数
response：网页传回用户端的回应
pageContext：网页的属性是在这里管理
session：与请求有关的会话期
application：与当前应用对应的ServletContext 对象，应用中只有一个

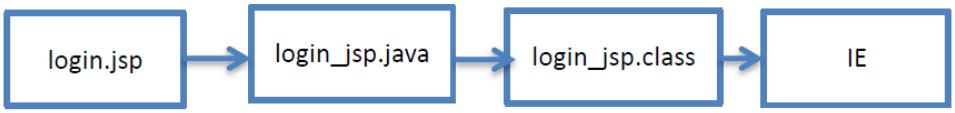
out：用来传送回应的输出 {}<%=%>
config：与jsp 配置对象的对象，一般无用
page：jsp 对应的Servlet 对象
exception：针对错误网页，未捕捉的异常对象

4.2. jsp 有哪些动作?作用分别是什么?

答:JSP 共有以下6 种基本动作
jsp:include：在页面被请求的时候引入一个文件。
jsp:forward：把请求转到一个新的页面。
jsp:useBean：寻找或者实例化一个JavaBean。
jsp:setProperty：设置JavaBean 的属性。
jsp:getProperty：输出某个JavaBean 的属性。
jsp:plugin：根据浏览器类型为Java 插件生成OBJECT 或EMBED 标记

4.3. JSP 的常用指令

答:主要有下面3 种指令
1. page 指令：指定页面的的一些属性，常用属性：
contentType="text/html; charset=utf-8" //向浏览器端输出数据的编码
pageEncoding="utf-8" //jsp 文件被编译成java 文件时所用的编码
session="true" //是否自动创建session



2. include 指令：包含别一个jsp 页面
3. taglib 指令：引入一个标签库

4.4. JSP 中动态INCLUDE 与静态INCLUDE 的区别?

答:
1. 动态包含：用<jsp:include>，包含的动作是在jsp 对应的Serlet 处理请求时去执行的,每次请求都会执行.
2. 静态包含：用include 指令，包含的动作是在jsp 被编译成java 文件时执行的，只有第一次请求时执行.

4.5. JSP 和Servlet 有哪些相同点和不同点，他们之间的联系是什么?

答:
JSP 的优点是擅长于网页制作，生成动态页面比较直观，缺点是不容易跟踪与排错。
Servlet 是纯Java 语言，擅长于处理流程和业务逻辑，缺点是生成动态网页不直观。

4.6. EL 的功能，为什么要用EL?

EL 的功能包括:
1. 从四个域对象中取出属性数据显示
2. 取出请求参数数据显示
为什么要用EL?
在页面中用jsp 脚本和jsp 表达式来获取数据显示比较麻烦
1. 需要条件判断
login.jsp login_jsp.java login_jsp.class IE
2. 可能需要强转

4.7. JSTL 的功能，为什么要用JSTL?

JSTL 的功能

JSTL 全名为JavaServer Pages Standard Tag Library，主要用于基本输入输出、流程控制、循环、XML 文件剖析、数据库查询及国际化和文字格式标准化的应用等

为什么要用JSTL?

在jsp 页面做条件判断或循环操作并输出时，比较费力

4.8. 为什么要用自定义标签?，MyTag 如何实现?

为什么要用?

1. 不想在Jsp 中编写java 代码
2. JSTL 标签库不能满足实际项目的需求

自定义标签定义和使用的流程

1. 编写标签处理器类(SimpleTagSupport 的实现类)

a) 重写doTag()

2. 编写标签库文件(WEB-INF/xxx.tld)

a) 整个文件的定义：<short-name> <uri>

b) 标签的定义：<tag>

3. 在jsp 页面使用标签：

a) 导入标签库(xxx.tld/)

b) 使用标签

5. Web 会话相关技术路线(紫线)

此部分包括：Cookie 和Session 技术

5.1. 说说你对Cookie 与Session 技术的理解?

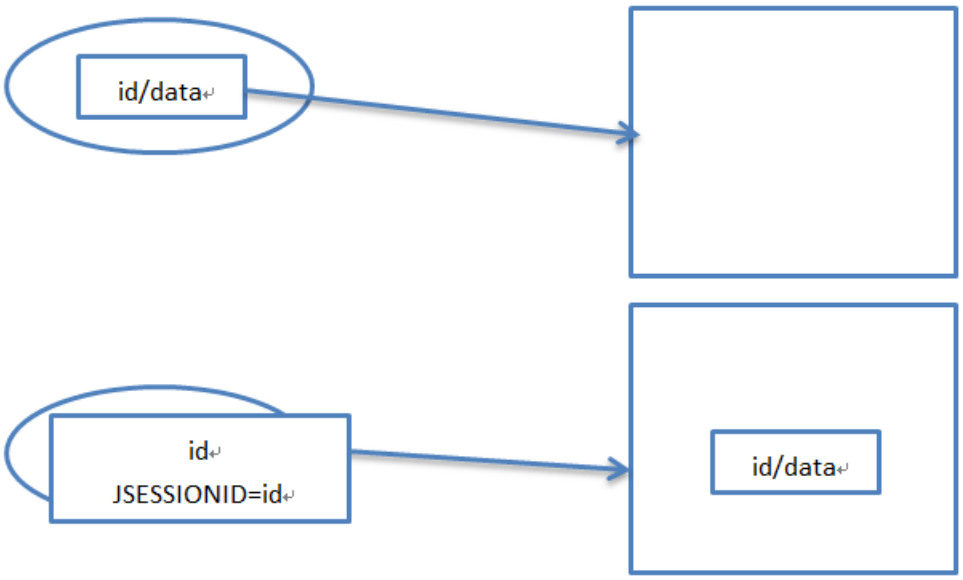
1. cookie 是一种浏览器端的缓存技术，而Session 是一种服务器端的缓存技术(依赖cookie)

经典现实案例：

某咖啡厅推出了一个优惠活动：累计喝五杯咖啡可以免费赠送一杯。他们该如何实现呢？

方法一：咖啡厅办卡(id,count)，交给消费者，消费者下次再来消费时，必须带上卡，消费一次由咖啡厅来更新卡上的数据，再次交给消费者

方法二：咖啡厅办卡(id)，id 和count 都保存在咖啡厅的电脑中的表中，将卡(id)交给消费者;消费者下次再来消费时，必须带上卡，消费一次由咖啡厅来更新表中的数据，再次交给消费者



5.2. 说说自动登陆功能的编码实现?

1. 登陆功能是用Session 实现的,就是向Session 对象中保存当前用户的对象
2. 自动的功能用Cookie 实现，就是登陆时将用户的信息保存为持久化Cookie
3. 下次访问时，读取请求中如果有用户信息的Cookie 就可以自动登陆

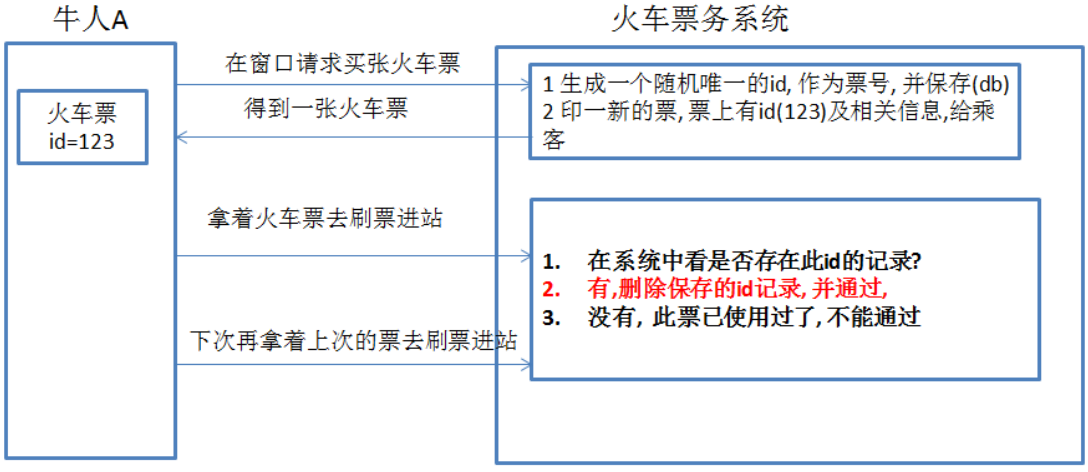
5.3. 如何防止表单重复提交?

答：使用Session 技术:

1. 在regist.jsp 页面中生成一个唯一随机值，将其保存到Session 中，同时将其保存为表单的隐藏域的值
2. 在处理注册的请求时,获取Session 中值,获取请求参数的值,比较两者是否相同，如果相同说明不是重复提交,请求通过同时删除session 中保存的值，如果不相同则是重复提交，不能通过.

经典现实案例:

一位乘客在北京火车站买了一张去天津的火车票(直接刷的那种)，他刷票进站坐火车去了天津，回来后过了几天，他又需要去天津这次他不想再买票，直接拿上次的票去进站口刷，检票机提示“此火车票已使用过了”，不能进站.



6. 其它

此部分包括: MVC, WebService 和Mybatis

6.1. MVC 的各个部分都有那些技术来实现?如何实现?

答: MVC 是Model－View－Controller 的简写。
Model 代表的是应用的业务逻辑（通过JavaBean，EJB 组件实现），
View 是应用的表示面（由JSP 页面产生），
Controller 是提供应用的处理过程控制（一般是一个Servlet），
通过这种设计模型把应用逻辑，处理过程和显示逻辑分成不同的组件实现。这些组件可以进行交互和重用。

6.2. WEB SERVICE 相关名词解释

Web Service
Web Service 是基于网络的、分布式的模块化组件，它执行特定的任务，遵守具体的技术规范，这些规范使得Web Service 能与其他兼容的组件进行互操作。
JAXM(Java API for XML Messaging)
是为SOAP 通信提供访问方法和传输机制的API。
WSDL:
是一种 XML 格式，用于将网络服务描述为一组端点，这些端点对包含面向文档信息或面向过程信息的消息进行操作。这种格式首先对操作和消息进行抽象描述，然后将其绑定到具体的网络协议和消息格式上以定义端点。相关的具体端点即组合成为抽象端点（服务）。
SOAP:
即简单对象访问协议(Simple Object Access Protocol)，它是用于交换XML 编码信息的轻量级协议。

6.3. WebService 技术的本质是使用哪几种技术实现的?

HTTP + XML + Schema

6.4. 如何编码发布一个WebService?

1. 定义SEI: 使用@Webservice 和@Webmethod
2. 定义SEI 的实现类: 使用@Webservice
3. 发布:使用JDK 中的Endpoint,或者使用CXF 框架基于Spring 的配置来发布

6.5. 如何编码请求一个WebService?

1. 根据wsdl 文档生成客户端代码.
2. 利用客户端代码编写调用webservice 的代码.

6.6. 比较一下JDBC, dbutils, Mybatis 和Hibernate

1. JDBC: 原生访问数据库的方式, 其它三个都是对JDBC 不同程度的包装访问数据库比较麻烦, 代码重复度极高
2. dbutils: 是对jdbc 进行了相对简单的包装, 主要就是能自动封装查询结构集, 需要在代码中写sql 语句
3. Mybatis: 进一步封装jdbc, Sql 语句写在配置文件中, 面向对象操作, 有一二级缓存功能
4. Hibernate: 对 jdbc 封装得最彻底的框架, 纯面向对象, 可以不用写 SQL__

3.SpringMVC 常见面试题（28 题）

1、讲下 SpringMvc 和 Struts1,Struts2 的比较的优势

性能上 Struts1>SpringMvc>Struts2 开发速度上 SpringMvc 和 Struts2 差不多,比 Struts1 要高

2、讲下 SpringMvc 的核心入口类是什么,Struts1,Struts2 的分别是什么

SpringMvc 的是 DispatchServlet,Struts1 的是 ActionServlet,Struts2 的是 StrutsPrepareAndExecuteFilter

3、SpringMvc 的控制器是不是单例模式,如果是,有什么问题,怎么解决

是单例模式,所以在多线程访问的时候有线程安全问题,不要用同步,会影响性能的,解决方案是在控制器里面不能写字段

4、SpingMvc 中的控制器的注解一般用那个,有没有别的注解可以替代

一般用@Conntroller 注解,表示是表现层,不能用用别的注解代替.

5、@RequestMapping 注解用在类上面有什么作用

用于类上，表示类中的所有响应请求的方法都是以该地址作为父路径。

6、怎么样把某个请求映射到特定的方法上面

直接在方法上面加上注解@RequestMapping,并且在这个注解里面写上要拦截的路径

7、如果在拦截请求中,我想拦截 get 方式提交的方法,怎么配置

可以在@RequestMapping 注解里面加上 method=RequestMethod.GET

8、如果在拦截请求中,我想拦截提交参数中包含"type=test"字符串,怎么配置

可以在@RequestMapping 注解里面加上 params="type=test"

9、我想在拦截的方法里面得到从前台传入的参数,怎么得到

直接在形参里面声明这个参数就可以,但必须名字和传过来的参数一样

10、如果前台有很多个参数传入,并且这些参数都是一个对象的,那么怎么样快速得到这个对象

直接在方法中声明这个对象,SpringMvc 就自动会把属性赋值到这个对象里面

11、怎么样在方法里面得到 Request,或者 Session

直接在方法的形参中声明 request,SpringMvc 就自动把 request 对象传入

12、SpringMvc 中函数的返回值是什么.

返回值可以有很多类型,有 String, ModelAndView,当一般用 String 比较好

13、SpringMvc 怎么处理返回值的

SpringMvc 根据配置文件中 InternalResourceViewResolver 的前缀和后缀,用前缀+返回值+后缀组成完整的返回值

14、SpringMVC 怎么样设定重定向和转发的

在返回值前面加"forward:"就可以让结果转发,譬如"forward:user.do?name=method4"
在返回值前面加"redirect:"就可以让返回值重定向,譬如"redirect:http://www.baidu.com"

15、SpringMvc 用什么对象从后台向前台传递数据的

通过 ModelMap 对象,可以在这个对象里面用 put 方法,把对象加到里面,前台就可以通过 el 表达式拿到

16、SpringMvc 中有个类把视图和数据都合并的一起的,叫什么

叫 ModelAndView

17、怎么样把 ModelMap 里面的数据放入 Session 里面

可以在类上面加上@SessionAttributes 注解,里面包含的字符串就是要放入 session 里面的 key

18、SpringMvc 怎么和 AJAX 相互调用的

通过 Jackson 框架就可以把 Java 里面的对象直接转化成 Js 可以识别的 Json 对象

具体步骤如下

1）加入 Jackson.jar

2）在配置文件中配置 json 的映射

3）在接受 Ajax 方法里面可以直接返回 Object,List 等,但方法前面要加上@ResponseBody 注解

19、当一个方法向 AJAX 返回特殊对象,譬如 Object,List 等,需要做什么处理

要加上@ResponseBody 注解

20、SpringMvc 里面拦截器是怎么写的

有两种写法,一种是实现接口,另外一种是继承适配器类,然后在 SpringMvc 的配置文件中配置拦截器即可:

```
<!-- 配置 SpringMvc 的拦截器 -->
<mvc:interceptors>
    <!-- 配置一个拦截器的 Bean 就可以了 默认是对所有请求都拦截 -->
    <bean id="myInterceptor" class="com.et.action.MyHandlerInterceptor"></bean>
    <!-- 只针对部分请求拦截 -->
    <mvc:interceptor>
       <mvc:mapping path="/modelMap.do" />
       <bean class="com.et.action.MyHandlerInterceptorAdapter" />
    </mvc:interceptor>
</mvc:interceptors>
```

21、springmvc 工作原理：

1) springmvc 请所有的请求都提交给 DispatcherServlet,它会委托应用系统的其他模块负责负责对请求进行真正的处理工作。

2) DispatcherServlet 查询一个或多个 HandlerMapping,找到处理请求的 Controller.

3) DispatcherServlet 请请求提交到目标 Controller

4) Controller 进行业务逻辑处理后，会返回一个 ModelAndView

5) Dispathcher 查询一个或多个 ViewResolver 视图解析器,找到 ModelAndView 对象指定的视图对象

6) 视图对象负责渲染返回给客户端。

22、请你谈谈 SSH 整合：

SSH

Struts（表示层）+Spring（业务层）+Hibernate（持久层）

Struts

Struts 是一个表示层框架，主要作用是界面展示，接收请求，分发请求。

在 MVC 框架中，Struts 属于 VC 层次，负责界面表现，负责 MVC 关系的分发。（View：沿用 JSP，HTTP，Form，Tag，Resourse ；Controller：ActionServlet，struts-config.xml，Action）

Hibernate

Hibernate 是一个持久层框架，它只负责与关系数据库的操作。

Spring

Spring 是一个业务层框架，是一个整合的框架，能够很好地黏合表示层与持久层。

23、Spring 里面 applicationContext.xml 文件能不能改成其他文件名？

ContextLoaderListener 是一个 ServletContextListener,它在你的 web 应用启动的时候初始化。缺省情况下， 它会在 WEB-INF/applicationContext.xml 文件找 Spring 的配置。 你可以通过定义一个元素名字为”contextConfigLocation”来改变 Spring 配置文件的位置。示例如下：

```
<listener>
   <listener-class>org.springframework.web.context.ContextLoaderListener
    <context-param>
       <param-name>contextConfigLocation</param-name>
        <param-value>/WEB-INF/xyz.xml</param-value>
    </context-param>
  </listener-class>
</listener>
```

24、如何在 web 应用里面配置 spring?

在 web.xml 中加入如下同容,在启动 web 服务器时加载/WEB-INF/applicationContext.xml 中的内容。

```
<servlet>
<servlet-name>context</servlet-name>
<servlet-class>
org.springframework.web.context.ContextLoaderServlet
</servlet-class>
```

```
<load-on-startup>1</load-on-startup>
</servlet>
```

通过如下类得到 ApplicationContext 实例
WebApplicationContextUtils.getWebApplicationContext

25、如何在 spring 中实现国际化?

在 applicationContext.xml 加载一个 bean

```
<beanid=”messageSource” class=”org.springframework.context.support.ResourceBu
ndleMessageSource” >
   <property name=”basename” >
    <value>message</value>
   </property>
</bean>
```

1) 在 src 目录下建多个 properties 文件

2) 对于非英文的要用 native2ascii-encoding gb2312 源目转化文件相关内容

3) 其命名格式是 message *语言*国家。

4) 页面中的中显示提示信息，键名取键值。

5) 当给定国家，系统会自动加载对应的国家的 properties 信息。

6) 通过 applictionContext.getMessage(“键名”,”参数”,”区域”)取出相关的信息。

26、什么是 Spring，它有什么特点？

Spring 是一个轻量级的控制反转(IoC)和面向切面(AOP)的容器框架。

- 轻量——从大小与开销两方面而言 Spring 都是轻量的。完整的 Spring 框架可以在一个大小只有 1MB 多的 JAR 文件里发布。并 且 Spring 所需的处理开销也是微不足道的。此外，Spring 是非侵入式的：典型地，Spring 应用中的对象不依赖于 Spring 的特定类。

- 控制反转——Spring 通过一种称作控制反转(IoC)的技术促进了松耦合。当应用了 IoC，一个对象依赖的其它对象会通过被动的方式传递进来，而不是这个对象自己创建或者查找依赖对象。你可以认为 IoC 与 JNDI 相反——不 是对象从容器中查找依赖，而是容器在对象初始化时不等对象请求就主动将依赖传递给它。

- 面向切面——Spring 提供了面向切面编程的丰富支持，允许通过分离应用的 业务逻辑与系统级服务（例如审计（auditing）和事务（）管理）进行内聚性的开发。应用对象只实现它们应该做的——完成业务逻辑——仅此而已。它们 并不负责(甚至是意识)其它的系统级关注点，例如日志或事务支持。

- 容器——Spring 包含并管理应用对象的配置和生命周期，在这个意义上它是 一种容器，你可以配置你的每个 bean 如何被创建——基于一个可配置原型（prototype），

你的 bean 可以创建一个单独的实例或者每次需要时都生 成一个新的实例——以及它们是如何相互关联的。然而，Spring 不应该被混同于传统的重量级的 EJB 容器，它们经常是庞大与笨重的，难以使用。

- 框架——Spring 可以将简单的组件配置、组合成为复杂的应用。在 Spring 中，应用对象被声明式地组合，典型地是在一个 XML 文件里。Spring 也提供了很多基础功能（事务管理、持久化框架集成等等），将应用逻辑的开发留给了你。

27、AOP 里面重要的几个名词概念解释：

- 切面（Aspect）： 一个关注点的模块化，这个关注点可能会横切多个对象。事务管理是 J2EE 应用中一个关于横切关注点的很好的例子。 在 Spring AOP 中，切面可以使用通用类（基于模式的风格）或者在普通类中以 @Aspect 注解（@AspectJ 风格）来实现。

- 连接点（Joinpoint）： 在程序执行过程中某个特定的点，比如某方法调用的时候或者处理异常的时候。 在 Spring AOP 中，一个连接点 总是代表一个方法的执行。 通过声明一个 org.aspectj.lang.JoinPoint 类型的参数可以使通知（Advice）的主体部分获得连接点信息。

- 通知（Advice）： 在切面的某个特定的连接点（Joinpoint）上执行的动作。通知有各种类型，其中包括“around”、“before”和“after”等通知。 通知的类型将在后面部分进行讨论。许多 AOP 框架，包括 Spring，都是以拦截器做通知模型， 并维护一个以连接点为中心的拦截器链。

- 切入点（Pointcut）： 匹配连接点（Joinpoint）的断言。通知和一个切入点表达式关联，并在满足这个切入点的连接点上运行（例如，当执行某个特定名称的方法时）。 切入点表达式如何和连接点匹配是 AOP 的核心：Spring 缺省使用 AspectJ 切入点语法。

- 引入（Introduction）： （也被称为内部类型声明（inter-typedeclaration））。声明额外的方法或者某个类型的字段。Spring 允许引入新的接口（以及一个对应的实现）到任何被代理的对象。 例如，你可以使用一个引入来使 bean 实现 IsModified 接口，以便简化缓存机制。

- 目标对象（Target Object）： 被一个或者多个切面（aspect）所通知（advise）的对象。也有人把它叫做 被通知（advised） 对象。 既然 Spring AOP 是通过运行时代理实现的，这个对象永远是一个 被代理（proxied） 对象。

- AOP 代理（AOP Proxy）： AOP 框架创建的对象，用来实现切面契约（aspect contract）（包括通知方法执行等功能）。在 Spring 中，AOP 代理可以是 JDK 动态代理或者 CGLIB 代理。 注意：Spring 2.0 最新引入的基于模式（schema-based）风格和@AspectJ 注解风格的切面声明，对于使用这些风格的用户来说，代理的创建是透明的。

- 织入（Weaving）： 把切面（aspect）连接到其它的应用程序类型或者对象上，并创建一个被通知（advised）的对象。这些可以在编译时（例如使用 AspectJ 编译器），类加载时和运行时完成。 Spring 和其他纯 JavaAOP 框架一样，在运行时完成织入。

通知的类型：

- 前置通知（Before advice）： 在某连接点（join point）之前执行的通知，但这个通知不能阻止连接点前的执行（除非它抛出一个异常）。

- 返回后通知（Afterreturning advice）：在某连接点（join point）正常完成后执行的通知：例如，一个方法没有抛出任何异常，正常返回。
- 抛出异常后通知（Afterthrowing advice）：在方法抛出异常退出时执行的通知。
- 后通知（After(finally) advice）：当某连接点退出的时候执行的通知（不论是正常返回还是异常退出）。
- 环绕通知（Around Advice）： 包围一个连接点（joinpoint）的通知，如方法调用。这是最强大的一种通知类型。 环绕通知可以在方法调用前后完成自定义的行为。它也会选择是否继续执行连接点或直接返回它们自己的返回值或抛出异常来结束执行。
- 环绕通知是最常用的一种通知类型。大部分基于拦截的 AOP 框架，例如 Nanning 和 JBoss4，都只提供环绕通知。
- 切入点（pointcut）和连接点（join point）匹配的概念是 AOP 的关键，这使得 AOP 不同于其它仅仅提供拦截功能的旧技术。 切入点使得定位通知（advice）可独立于 OO 层次。 例如，一个提供声明式事务管理的 around 通知可以被应用到一组横跨多个对象中的方法上（例如服务层的所有业务操作）。

28、SpringMVC 与 Struts2 的主要区别？

1) springmvc 的入口是一个 servlet 即前端控制器，而 struts2 入口是一个 filter 过虑器。

2) springmvc 是基于方法开发，传递参数是通过方法形参，可以设计为单例或多例(建议单例)，struts2 是基于类开发，传递参数是通过类的属性，只能设计为多例。

3) Struts 采用值栈存储请求和响应的数据，通过 OGNL 存取数据， springmvc 通过参数解析器是将 request 对象内容进行解析成方法形参，将响应数据和页面封装成 ModelAndView 对象，最后又将模型数据通过 request 对象传输到页面。 Jsp 视图解析器默认使用 jstl。

4.Spring 常见面试题（56 题）

1、什么是 spring?

Spring 是个 java 企业级应用的开源开发框架。Spring 主要用来开发 Java 应用，但是有些扩展是针对构建 J2EE 平台的 web 应用。Spring 框架目标是简化 Java 企业级应用开发，并通过 POJO 为基础的编程模型促进良好的编程习惯。

2、使用 Spring 框架的好处是什么？

轻量：Spring 是轻量的，基本的版本大约 2MB。

控制反转：Spring 通过控制反转实现了松散耦合，对象们给出它们的依赖，而不是创建或查找依赖的对象们。

面向切面的编程(AOP)：Spring 支持面向切面的编程，并且把应用业务逻辑和系统服务分开。

容器：Spring 包含并管理应用中对象的生命周期和配置。

MVC 框架：Spring 的 WEB 框架是个精心设计的框架，是 Web 框架的一个很好的替代品。

事务管理：Spring 提供一个持续的事务管理接口，可以扩展到上至本地事务下至全局事务（JTA）。

异常处理：Spring 提供方便的 API 把具体技术相关的异常（比如由 JDBC，Hibernate or JDO 抛出的）转化为一致的 unchecked 异常。

3、 BeanFactory - BeanFactory 实现举例。

Bean 工厂是工厂模式的一个实现，提供了控制反转功能，用来把应用的配置和依赖从正真的应用代码中分离。

最常用的 BeanFactory 实现是 XmlBeanFactory 类。

4、XMLBeanFactory

最常用的就是 org.springframework.beans.factory.xml.XmlBeanFactory ，它根据 XML 文件中的定义加载 beans。该容器从 XML 文件读取配置元数据并用它去创建一个完全配置的系统或应用。

5、解释 AOP 模块

AOP 模块用于发给我们的 Spring 应用做面向切面的开发， 很多支持由 AOP 联盟提供，这样就确保了 Spring 和其他 AOP 框架的共通性。这个模块将元数据编程引入 Spring。

6、什么是 Spring IOC 容器？

Spring IOC 负责创建对象，管理对象（通过依赖注入（DI），装配对象，配置对象，并且管理这些对象的整个生命周期。

7、IOC 的优点是什么？

IOC 或 依赖注入把应用的代码量降到最低。它使应用容易测试，单元测试不再需要单例和 JNDI 查找机制。最小的代价和最小的侵入性使松散耦合得以实现。IOC 容器支持加载服务时的饿汉式初始化和懒加载。

8、ApplicationContext 通常的实现是什么？

FileSystemXmlApplicationContext ：此容器从一个 XML 文件中加载 beans 的定义，XML Bean 配置文件的全路径名必须提供给它的构造函数。

ClassPathXmlApplicationContext：此容器也从一个 XML 文件中加载 beans 的定义，这里，你需要正确设置 classpath 因为这个容器将在 classpath 里找 bean 配置。

WebXmlApplicationContext：此容器加载一个 XML 文件，此文件定义了一个 WEB 应用的所有 bean。

【依赖注入部分】

9、什么是 Spring 的依赖注入？ **

依赖注入，是 IOC 的一个方面，是个通常的概念，它有多种解释。这概念是说你不用创建对象，而只需要描述它如何被创建。你不在代码里直接组装你的组件和服务，但是要在配置文件里描述哪些组件需要哪些服务，之后一个容器（IOC 容器）负责把他们组装起来。

10、有哪些不同类型的 IOC（依赖注入）方式？

构造器依赖注入：构造器依赖注入通过容器触发一个类的构造器来实现的，该类有一系列参数，每个参数代表一个对其他类的依赖。

Setter 方法注入：Setter 方法注入是容器通过调用无参构造器或无参 static 工厂 方法实例化 bean 之后，调用该 bean 的 setter 方法，即实现了基于 setter 的依赖注入。

11、哪种依赖注入方式你建议使用，构造器注入，还是 Setter 方法注入？

你两种依赖方式都可以使用，构造器注入和 Setter 方法注入。最好的解决方案是用构造器参数实现强制依赖，setter 方法实现可选依赖。

【Spring Beans 部分】

12、什么是 Spring beans?

Spring beans 是那些形成 Spring 应用的主干的 java 对象。它们被 Spring IOC 容器初始化，装配，和管理。这些 beans 通过容器中配置的元数据创建。比如，以 XML 文件中 的形式定

义。

Spring 框架定义的 beans 都是单件 beans。在 bean tag 中有个属性”singleton”，如果它被赋为 TRUE，bean 就是单件，否则就是一个 prototype bean。默认是 TRUE，所以所有在 Spring 框架中的 beans 缺省都是单 件。

13、一个 Spring Bean 定义 包含什么？

一个 Spring Bean 的定义包含容器必知的所有配置元数据，包括如何创建一个 bean，它的生命周期详情及它的依赖。

14、如何给 Spring 容器提供配置元数据?

这里有三种重要的方法给 Spring 容器提供配置元数据。

XML 配置文件。

基于注解的配置。

基于 java 的配置。

15、你怎样定义类的作用域?

当定义一个 在 Spring 里，我们还能给这个 bean 声明一个作用域。它可以通过 bean 定义中的 scope 属性来定 义。如，当 Spring 要在需要的时候每次生产一个新的 bean 实例，bean 的 scope 属性被指定为 prototype。另一方面，一个 bean 每次 使用的时候必须返回同一个实例，这个 bean 的 scope 属性 必须设为 singleton。

16、解释 Spring 支持的几种 bean 的作用域。

Spring 框架支持以下五种 bean 的作用域：

singleton : bean 在每个 Spring ioc 容器中只有一个实例。

prototype：一个 bean 的定义可以有多个实例。

request：每次 http 请求都会创建一个 bean，该作用域仅在基于 web 的 Spring ApplicationContext 情形下有效。

session：在一个 HTTP Session 中，一个 bean 定义对应一个实例。该作用域仅在基于 web 的 Spring ApplicationContext 情形下有效。

global-session：在一个全局的 HTTP Session 中，一个 bean 定义对应一个实例。该作用域仅在基于 web 的 Spring ApplicationContext 情形下有效。

缺省的 Spring bean 的作用域是 Singleton.

17、Spring 框架中的单例 bean 是线程安全的吗?

不，Spring 框架中的单例 bean 不是线程安全的。

18、解释 Spring 框架中 bean 的生命周期。

Spring 容器 从 XML 文件中读取 bean 的定义，并实例化 bean。

Spring 根据 bean 的定义填充所有的属性。

如果 bean 实现了 BeanNameAware 接口，Spring 传递 bean 的 ID 到 setBeanName 方法。

如果 Bean 实现了 BeanFactoryAware 接口， Spring 传递 beanfactory 给 setBeanFactory 方法。

如果有任何与 bean 相关联的 BeanPostProcessors，Spring 会在 postProcesserBeforeInitialization() 方法内调用它们。

如果 bean 实现 IntializingBean 了，调用它的 afterPropertySet 方法，如果 bean 声明了初始化方法，调用此初始化方法。

如果有 BeanPostProcessors 和 bean 关联，这些 bean 的 postProcessAfterInitialization() 方法将被调用。

如果 bean 实现了 DisposableBean，它将调用 destroy()方法。

19、哪些是重要的 bean 生命周期方法？ 你能重载它们吗？

有两个重要的 bean 生命周期方法，第一个是 setup ， 它是在容器加载 bean 的时候被调用。第二个方法是 teardown 它是在容器卸载类的时候被调用。

The bean 标签有两个重要的属性（init-method 和 destroy-method）。用它们你可以自己定制初始化和注销方法。它们也有相应的注解（@PostConstruct 和@PreDestroy）。

20、什么是 Spring 的内部 bean?

当一个 bean 仅被用作另一个 bean 的属性时，它能被声明为一个内部 bean，为了定义 inner bean，在 Spring 的 基于 XML 的 配置元数据中，可以在 或 元素内使用 元素，内部 bean 通常是匿名的，它们的 Scope 一般是 prototype。

21、在 Spring 中如何注入一个 java 集合?

Spring 提供以下几种集合的配置元素：

类型用于注入一列值，允许有相同的值。

类型用于注入一组值，不允许有相同的值。

类型用于注入一组键值对，键和值都可以为任意类型。

类型用于注入一组键值对，键和值都只能为 String 类型。

22、什么是 bean 装配?

装配，或 bean 装配是指在 Spring 容器中把 bean 组装到一起，前提是容器需要知道 bean 的依赖关系，如何通过依赖注入来把它们装配到一起。

23、什么是 bean 的自动装配?

Spring 容器能够自动装配相互合作的 bean，这意味着容器不需要和配置，能通过 Bean 工厂自动处理 bean 之间的协作。

24、解释不同方式的自动装配 。

有五种自动装配的方式，可以用来指导 Spring 容器用自动装配方式来进行依赖注入。

no：默认的方式是不进行自动装配，通过显式设置 ref 属性来进行装配。

byName：通过参数名 自动装配，Spring 容器在配置文件中发现 bean 的 autowire 属性被设置成 byname，之后容器试图匹配、装配和该 bean 的属性具有相同名字的 bean。

byType:：通过参数类型自动装配，Spring 容器在配置文件中发现 bean 的 autowire 属性被设置成 byType，之后容器试图匹配、装配和该 bean 的属性具有相同类型的 bean。如果有多个 bean 符合条件，则抛出错误。

constructor：这个方式类似于 byType， 但是要提供给构造器参数，如果没有确定的带参数的构造器参数类型，将会抛出异常。

autodetect：首先尝试使用 constructor 来自动装配，如果无法工作，则使用 byType 方式。

25、自动装配有哪些局限性 ？

自动装配的局限性是：

重写： 你仍需用 和 配置来定义依赖，意味着总要重写自动装配。

基本数据类型：你不能自动装配简单的属性，如基本数据类型，String 字符串，和类。

模糊特性：自动装配不如显式装配精确，如果有可能，建议使用显式装配。

26、你可以在 Spring 中注入一个 null 和一个空字符串吗？

可以。

【Spring 注解部分】

27、什么是基于 Java 的 Spring 注解配置? 给一些注解的例子.

基于 Java 的配置，允许你在少量的 Java 注解的帮助下，进行你的大部分 Spring 配置而非通过 XML 文件。

以@Configuration 注解为例，它用来标记类可以当做一个 bean 的定义，被 Spring IOC 容器使用。另一个例子是@Bean 注解，它表示此方法将要返回一个对象，作为一个 bean 注册进

Spring 应用上下文。

28、什么是基于注解的容器配置?

相对于 XML 文件，注解型的配置依赖于通过字节码元数据装配组件，而非尖括号的声明。开发者通过在相应的类，方法或属性上使用注解的方式，直接组件类中进行配置，而不是使用 xml 表述 bean 的装配关系。

29、怎样开启注解装配?

注解装配在默认情况下是不开启的，为了使用注解装配，我们必须在 Spring 配置文件中配置 元素。

30、 @Required 注解

这个注解表明 bean 的属性必须在配置的时候设置，通过一个 bean 定义的显式的属性值或通过自动装配，若 @Required 注解的 bean 属性未被设置，容器将抛出 BeanInitializationException。

31、@Autowired 注解

@Autowired 注解提供了更细粒度的控制，包括在何处以及如何完成自动装配。它的用法和 @Required 一样，修饰 setter 方法、构造器、属性或者具有任意名称和/或多个参数的 PN 方法。

32、@Qualifier 注解

当有多个相同类型的 bean 却只有一个需要自动装配时，将@Qualifier 注解和@Autowire 注解结合使用以消除这种混淆，指定需要装配的确切的 bean。

【Spring 数据访问】

33、在 Spring 框架中如何更有效地使用 JDBC?

使用 SpringJDBC 框架，资源管理和错误处理的代价都会被减轻。所以开发者只需写 statements 和 queries 从数据存取数据，JDBC 也可以在 Spring 框架提供的模板类的帮助下更有效地被使用，这个模板叫 JdbcTemplate （例子见这里 here）

34、JdbcTemplate

JdbcTemplate 类提供了很多便利的方法解决诸如把数据库数据转变成基本数据类型或对象，执行写好的或可调用的数据库操作语句，提供自定义的数据错误处理。

35、Spring 对 DAO 的支持

Spring 对数据访问对象（DAO）的支持旨在简化它和数据访问技术如 JDBC，Hibernate or JDO 结合使用。这使我们可以方便切换持久层。编码时也不用担心会捕获每种技术特有的异常。

36、使用 Spring 通过什么方式访问 Hibernate?

在 Spring 中有两种方式访问 Hibernate：

控制反转 Hibernate Template 和 Callback。

继承 HibernateDAOSupport 提供一个 AOP 拦截器。

37、Spring 支持的 ORM

Spring 支持以下 ORM：

Hibernate

iBatis

JPA (Java Persistence API)

TopLink

JDO (Java Data Objects)

OJB

38、如何通过 HibernateDaoSupport 将 Spring 和 Hibernate 结合起来?

用 Spring 的 SessionFactory 调用 LocalSessionFactory。集成过程分三步：

配置 the Hibernate SessionFactory。
继承 HibernateDaoSupport 实现一个 DAO。
在 AOP 支持的事务中装配。

39、Spring 支持的事务管理类型

Spring 支持两种类型的事务管理：
编程式事务管理：这意味你通过编程的方式管理事务，给你带来极大的灵活性，但是难维护。
声明式事务管理：这意味着你可以将业务代码和事务管理分离，你只需用注解和 XML 配置来管理事务。

40、Spring 框架的事务管理有哪些优点？

它为不同的事务 API 如 JTA，JDBC，Hibernate，JPA 和 JDO，提供一个不变的编程模式。
它为编程式事务管理提供了一套简单的 API 而不是一些复杂的事务 API 如
它支持声明式事务管理。
它和 Spring 各种数据访问抽象层很好得集成。

41、你更倾向用那种事务管理类型？

大多数 Spring 框架的用户选择声明式事务管理，因为它对应用代码的影响最小，因此更符合一个无侵入的轻量级容器的思想。声明式事务管理要优于编程式事务管理，虽然比编程式事务管理（这种方式允许你通过代码控制事务）少了一点灵活性。

【Spring 面向切面编程（AOP）】

42、解释 AOP

面向切面的编程，或 AOP， 是一种编程技术，允许程序模块化横向切割关注点，或横切典型的责任划分，如日志和事务管理。

43、Aspect 切面

AOP 核心就是切面，它将多个类的通用行为封装成可重用的模块，该模块含有一组 API 提供横切功能。比如，一个日志模块可以被称作日志的 AOP 切面。根据需求的不同，一个应用程序可以有若干切面。在 Spring AOP 中，切面通过带有@Aspect 注解的类实现。

44、在 Spring AOP 中，关注点和横切关注的区别是什么？

关注点是应用中一个模块的行为，一个关注点可能会被定义成一个我们想实现的一个功能。
横切关注点是一个关注点，此关注点是整个应用都会使用的功能，并影响整个应用，比如日志，安全和数据传输，几乎应用的每个模块都需要的功能。因此这些都属于横切关注点。

45、连接点

连接点代表一个应用程序的某个位置，在这个位置我们可以插入一个 AOP 切面，它实际上是个应用程序执行 Spring AOP 的位置。

46、通知

通知是个在方法执行前或执行后要做的动作，实际上是程序执行时要通过 SpringAOP 框架触发的代码段。
Spring 切面可以应用五种类型的通知：
before：前置通知，在一个方法执行前被调用。
after: 在方法执行之后调用的通知，无论方法执行是否成功。
after-returning: 仅当方法成功完成后执行的通知。
after-throwing: 在方法抛出异常退出时执行的通知。
around: 在方法执行之前和之后调用的通知。

47、切点

切入点是一个或一组连接点，通知将在这些位置执行。可以通过表达式或匹配的方式指明切入点。

48、什么是引入?

引入允许我们在已存在的类中增加新的方法和属性。

49、什么是目标对象?

被一个或者多个切面所通知的对象。它通常是一个代理对象。也指被通知（advised）对象。

50、什么是代理?

代理是通知目标对象后创建的对象。从客户端的角度看，代理对象和目标对象是一样的。

51、有几种不同类型的自动代理？

BeanNameAutoProxyCreator

DefaultAdvisorAutoProxyCreator

Metadata autoproxying

52、什么是织入。什么是织入应用的不同点？

织入是将切面和到其他应用类型或对象连接或创建一个被通知对象的过程。

织入可以在编译时，加载时，或运行时完成。

53、解释基于 XML Schema 方式的切面实现。

在这种情况下，切面由常规类以及基于 XML 的配置实现。

54、解释基于注解的切面实现

在这种情况下(基于@AspectJ 的实现)，涉及到的切面声明的风格与带有 java5 标注的普通 java 类一致。

55、Spring 的事物有几种方式？谈谈 spring 事物的隔离级别和传播行为？

- 声明式事务 使用 spring 声明式事务，spring 使用 AOP 来支持声明式事务，会根据事务属性，自动在方法调用之前决定是否开启一个事务，并在方法执行之后决定事务提交或回滚事务。
- 编程式事务管理

Spring 的编程式事务与声明式事务区别

- 编程式事务需要你在代码中直接加入处理事 务的逻辑,可能需要在代码中显式调用 beginTransaction()、commit()、rollback()等事务管理相关的方法,如在执行 a 方 法时候需要事务处理,你需要在 a 方法开始时候开启事务,处理完后。在方法结束时候,关闭事务.
- 声明式的事务的做法是在 a 方法外围添加注解或者直接在配置文件中定义,a 方法需要事务处理,在 spring 中会通过配置文件在 a 方法前后拦截,并添加事务.
- 二者区别.编程式事务侵入性比较强，但处理粒度更细.

事务的隔离级别：

数据库系统提供了 4 种事务隔离级别，在这 4 种隔离级别中，Serializable 的隔离级别最高，Read Uncommitted 的隔离级别最低；

- Read Uncommitted 读未提交数据；（会出现脏读）
- Read Committed 读已提交数据；
- Repeatable Read 可重复读；
- Serializable 串行化

事务的传播属性包括：

- REQUIRED 这个是默认的属性 Support a current transaction, create a new one if none exists. 如果存在一个事务，则支持当前事务。如果没有事务则开启一个新的事务。 被设置成这个级别时，会为每一个被调用的方法创建一个逻辑事务域。如果前面的方法已经创建了事务，那么后面的方法支持当前的事务，如果当前没有事务会重新建立事务。
- MANDATORY Support a current transaction, throw an exception if none exists.支持当前

事务，如果当前没有事务，就抛出异常。

- NEVER Execute non-transactionally, throw an exception if a transaction exists. 以非事务方式执行，如果当前存在事务，则抛出异常。
- NOT_SUPPORTED Execute non-transactionally, suspend the current transaction if one exists. 以非事务方式执行操作，如果当前存在事务，就把当前事务挂起。
- REQUIRES_NEW Create a new transaction, suspend the current transaction if one exists. 新建事务，如果当前存在事务，把当前事务挂起。
- SUPPORTS Support a current transaction, execute non-transactionally if none exists.

支持当前事务，如果当前没有事务，就以非事务方式执行。

- NESTED Execute within a nested transaction if a current transaction exists, behave like PROPAGATION_REQUIRED else. 支持当前事务，新增 Savepoint 点，与当前事务同步提交或回滚。 嵌套事务一个非常重要的概念就是内层事务依赖于外层事务。 外层事务失败时，会回滚内层事务所做的动作。而内层事务操作失败并不会引起外层事务的回滚。
- PROPAGATION_NESTED 与 PROPAGATION_REQUIRES_NEW 的区别 它们非常 类似,都像一个嵌套事务，如果不存在一个活动的事务，都会开启一个新的事务。使用 PROPAGATION_REQUIRES_NEW 时，内层事务与外层事务就像两个独立的事务一样，一旦内层事务进行了提交后，外层事务不能对其进行回滚。两个事务互不影响。两个事务不是一个真正的嵌套事务。同时它需要 JTA 事务管理器的支持。 使用 PROPAGATION_NESTED 时，外层事务的回滚可以引起内层事务的回滚。而内层事务的异常并不会导致外层事务的回滚，它是一个真正的嵌套事务。

56、Spring 的优点和缺点？

Spring 的优点：

1) Spring 能有效地组织你的中间层对象，不管你是否选择使用了 EJB；
2) Spring 能消除在许多工程中常见的对 Singleton 的过多使用。(因为它降低了系统的可测试性和面向对象的程度)；
3) 通过一种在不同应用程序和项目间一致的方法来处理配置文件，Spring 能消除各种各样自定义格式的属性文件的需要。Inversion of Control 的使用帮助完成了这种简化；
4) 通过把对接口编程而不是对类编程的代价几乎减少到没有，Spring 能够促进养成好的编程习惯；
5) Spring 被设计为让使用它创建的应用尽可能少的依赖于他的 APIs。在 Spring 应用中的大多数业务对象没有依赖于 Spring；
6) 使用 Spring 构建的应用程序易于单元测试；
7) Spring 能使 EJB 的使用成为一个实现选择，而不是应用架构的必然选择。你能选择用 POJOs 或 local EJBs 来实现业务接口，却不会影响调用代码；
8) Spring 帮助你解决许多问题而无需使用 EJB。Spring 能提供一种 EJB 的替换物，他们适用于许多 web 应用。例如：Spring 能使用 AOP 提供声明性事务管理而不通过 EJB 容器；
9) Spring 为数据存取提供了一个一致的框架不论使用的是 JDBC 还是 O/R mapping 产品；

Spring 的缺点：

1) 使用人数不多，jsp 中要写很多代码；
2) 控制器过于灵活，缺少一个公用控制器。

5.MyBatis 常见面试题（25 题）

1、什么是 MyBatis 的接口绑定，有什么好处？

接口映射就是在 IBatis 中任意定义接口然，后把接口里面的方法和 SQL 语句绑定，我们直接调用接口方法就可以，这样比起原来了 SqlSession 提供的方法我们可以有更加灵活的选择和设置。

2、接口绑定有几种实现方式，分别是怎么实现的？

接口绑定有两种实现方式，

1) 一种是通过注解绑定，就是在接口的方法上面加上@Select@Update 等注解里面包含 Sql 语句来绑定。
2) 另外一种就是通过 xml 里面写 SQL 来绑定，在这种情况下，要指定 xml 映射文件里面的 namespace 必须为接口的全路径名。

3、什么情况下用注解绑定，什么情况下用 xml 绑定？

1) 当 Sql 语句比较简单时候，用注解绑定。
2) 当 SQL 语句比较复杂时候，用 xml 绑定，一般用 xml 绑定的比较多。

4、MyBatis 实现一对一有几种方式？具体怎么操作的？

1) 有联合查询和嵌套查询，联合查询是几个表联合查询，只查询一次，通过在 resultMap 里面配置 association 节点配置一对一的类就可以完成；
2) 嵌套查询是先查一个表，根据这个表里面的结果的外键 id，去再另外一个表里面查询数据，也是通过 association 配置，但另外一个表的查询通过 select 属性配置。

5、MyBatis 里面的动态 Sql 是怎么设定的？用什么语法？

MyBatis 里面的动态 Sql 一般是通过 if 节点来实现，通过 OGNL 语法来实现，但是如果要写的完整，必须配合 where，trim 节点，where 节点是判断包含节点有内容就插入 where，否则不插入，trim 节点是用来判断如果动态语句是以 and 或 or 开始，那么会自动把这个 and 或者 or 去掉。

6、讲下 MyBatis 的缓存

MyBatis 的缓存分为一级缓存和二级缓存：

1) 一级缓存放在 session 里面，默认就有。
2) 二级缓存放在它的命名空间里，默认是打开的，使用二级缓存属性类需要实现 Serializable 序列化接口(可用来保存对象的状态)，可在它的映射文件中配置。

7、MyBatis(IBatis)的好处是什么？

1) ibatis 把 sql 语句从 Java 源程序中独立出来，放在单独的 XML 文件中编写，给程序的维护带来了很大便利。
2) ibatis 封装了底层 JDBC API 的调用细节，并能自动将结果集转换成 Java Bean 对象，大大简化了 Java 数据库编程的重复工作。

3) 因为 Ibatis 需要程序员自己去编写 sql 语句，程序员可以结合数据库自身的特点灵活控制 sql 语句，因此能够实现比 hibernate 等全自动 orm 框架更高的查询效率，能够完成复杂查询。

8、#{}和${}的区别是什么？

${}是 Properties 文件中的变量占位符，它可以用于标签属性值和 sql 内部，属于静态文本替换，比如${driver}会被静态替换为 com.mysql.jdbc.Driver。

#{}是 sql 的参数占位符，Mybatis 会将 sql 中的#{}替换为?号，在 sql 执行前会使用 PreparedStatement 的参数设置方法，按序给 sql 的?号占位符设置参数值，比如 ps.setInt(0, parameterValue)，#{item.name}的取值方式为使用反射从参数对象中获取 item 对象的 name 属性值，相当于 param.getItem().getName()。

9、Xml 映射文件中，除了常见的 select|insert|updae|delete 标签之外，还有哪些标签？

<resultMap>、<parameterMap>、<sql>、<include>、<selectKey>，加上动态 sql 的 9 个标签，trim|where|set|foreach|if|choose|when|otherwise|bind 等，其中<sql>为 sql 片段标签，通过<include>标签引入 sql 片段，<selectKey>为不支持自增的主键生成策略标签。

10、最佳实践中，通常一个 Xml 映射文件，都会写一个 Dao 接口与之对应，请问，这个 Dao 接口的工作原理是什么？Dao 接口里的方法，参数不同时，方法能重载吗？

答：Dao 接口，就是人们常说的 Mapper 接口，接口的全限名，就是映射文件中的 namespace 的值，接口的方法名，就是映射文件中 MappedStatement 的 id 值，接口方法内的参数，就是传递给 sql 的参数。

Mapper 接口是没有实现类的，当调用接口方法时，接口全限名+方法名拼接字符串作为 key 值，可唯一定位一个 MappedStatement.

举例：

com.mybatis3.mappers.StudentDao.findStudentById，可以唯一找到 namespace 为 com.mybatis3.mappers.StudentDao 下面 id = findStudentById 的 MappedStatement。

在 Mybatis 中，每一个<select>、<insert>、<update>、<delete>标签，都会被解析为一个 MappedStatement 对象。

Dao 接口里的方法，是不能重载的，因为是全限名+方法名的保存和寻找策略。

Dao 接口的工作原理是 JDK 动态代理，Mybatis 运行时会使用 JDK 动态代理为 Dao 接口生成代理 proxy 对象，代理对象 proxy 会拦截接口方法，转而执行 MappedStatement 所代表的 sql，然后将 sql 执行结果返回。

11、Mybatis 是如何进行分页的？分页插件的原理是什么？

答：Mybatis 使用 RowBounds 对象进行分页，它是针对 ResultSet 结果集执行的内存分页，而非物理分页，可以在 sql 内直接书写带有物理分页的参数来完成物理分页功能，也可以使用分页插件来完成物理分页。

分页插件的基本原理是使用 Mybatis 提供的插件接口，实现自定义插件，在插件的拦截方法内拦截待执行的 sql，然后重写 sql，根据 dialect 方言，添加对应的物理分页语句和物理分页参数。

举例：select * from student，拦截 sql 后重写为：select t.* from （select * from student）t limit 0，10

12、简述 Mybatis 的插件运行原理，以及如何编写一个插件。

答：Mybatis 仅可以编写针对 ParameterHandler、ResultSetHandler、StatementHandler、Executor 这 4 种接口的插件，Mybatis 使用 JDK 的动态代理，为需要拦截的接口生成代理对象以实现接口方法拦截功能，每当执行这 4 种接口对象的方法时，就会进入拦截方法，具体就是 InvocationHandler 的 invoke()方法，当然，只会拦截那些你指定需要拦截的方法。

实现 Mybatis 的 Interceptor 接口并复写 intercept()方法，然后在给插件编写注解，指定要拦截哪一个接口的哪些方法即可，记住，别忘了在配置文件中配置你编写的插件。

13、Mybatis 执行批量插入，能返回数据库主键列表吗？

答：能，JDBC 都能，Mybatis 当然也能。

14、Mybatis 动态 sql 是做什么的？都有哪些动态 sql？能简述一下动态 sql 的执行原理

答：Mybatis 动态 sql 可以让我们在 Xml 映射文件内，以标签的形式编写动态 sql，完成逻辑判断和动态拼接 sql 的功能，Mybatis 提供了 9 种动态 sql 标签 trim|where|set|foreach|if|choose|when|otherwise|bind。

其执行原理为，使用 OGNL 从 sql 参数对象中计算表达式的值，根据表达式的值动态拼接 sql，以此来完成动态 sql 的功能。

15、Mybatis 是如何将 sql 执行结果封装为目标对象并返回的？都有哪些映射形式？

1) 第一种是使用标签，逐一定义列名和对象属性名之间的映射关。

2) 第二种是使用 sql 列的别名功能，将列别名书写为对象属性名，比如 T_NAME AS NAME，对象属性名一般是 name，小写，但是列名不区分大小写，Mybatis 会忽略列名大小写，智能找到与之对应对象属性名，你甚至可以写成 T_NAME AS NaMe，Mybatis 一样可以正常工作。

3) 有了列名与属性名的映射关系后，Mybatis 通过反射创建对象，同时使用反射给对象的属性逐一赋值并返回，那些找不到映射关系的属性，是无法完成赋值的。

16、Mybatis 能执行一对一、一对多的关联查询吗？都有哪些实现方式，以及它们之间的区别。

能，Mybatis 不仅可以执行一对一、一对多的关联查询，还可以执行多对一，多对多的关联查询，多对一查询，其实就是一对一查询，只需要把 selectOne()修改为 selectList()即可；多对多查询，其实就是一对多查询，只需要把 selectOne()修改为 selectList()即可。

关联对象查询，有两种实现方式。

1) 一种是单独发送一个 sql 去查询关联对象，赋给主对象，然后返回主对象。

2) 另一种是使用嵌套查询，嵌套查询的含义为使用 join 查询，一部分列是 A 对象的属性值，另外一部分列是关联对象 B 的属性值，好处是只发一个 sql 查询，就可以把主对象和其关联对象查出来。

那么问题来了，join 查询出来 100 条记录，如何确定主对象是 5 个，而不是 100 个？其去重复的原理是标签内的子标签，指定了唯一确定一条记录的 id 列，Mybatis 根据列值来完成 100 条记录的去重复功能，可以有多个，代表了联合主键的语意。

同样主对象的关联对象，也是根据这个原理去重复的，尽管一般情况下，只有主对象会有重复记录，关联对象一般不会重复。

举例：下面 join 查询出来 6 条记录，一、二列是 Teacher 对象列，第三列为 Student 对象列，Mybatis 去重复处理后，结果为 1 个老师 6 个学生，而不是 6 个老师 6 个学生。

t_id	t_name	s_id
1	teacher	38
1	teacher	39
1	teacher	40
1	teacher	41
1	teacher	42
1	teacher	43

17、Mybatis 是否支持延迟加载？如果支持，它的实现原理是什么？

答：Mybatis 仅支持 association 关联对象和 collection 关联集合对象的延迟加载，association 指的就是一对一，collection 指的就是一对多查询。在 Mybatis 配置文件中，可以配置是否启用延迟加载 lazyLoadingEnabled=true|false。

它的原理是，使用 CGLIB 创建目标对象的代理对象，当调用目标方法时，进入拦截器方法，比如调用 a.getB().getName()，拦截器 invoke()方法发现 a.getB()是 null 值，那么就会单独发送事先保存好的查询关联 B 对象的 sql，把 B 查询上来，然后调用 a.setB(b)，于是 a 的对象 b 属性就有值了，接着完成 a.getB().getName()方法的调用。这就是延迟加载的基本原理。

当然了，不光是 Mybatis，几乎所有的包括 Hibernate，支持延迟加载的原理都是一样的。

18、Mybatis 的 Xml 映射文件中，不同的 Xml 映射文件，id 是否可以重复？

答：不同的 Xml 映射文件，如果配置了 namespace，那么 id 可以重复。如果没有配置 namespace，那么 id 不能重复。毕竟 namespace 不是必须的，只是最佳实践而已。

原因就是 namespace+id 是作为 Map<String, MappedStatement>**的 key 使用的，如果没有 namespace，就剩下 id，那么，id 重复会导致数据互相覆盖。有了 namespace，自然 id 就可以重复，namespace 不同，namespace+id 自然也就不同。**

19、Mybatis 中如何执行批处理？

答：使用 BatchExecutor 完成批处理。

20、Mybatis 都有哪些 Executor 执行器？它们之间的区别是什么？

答：Mybatis 有三种基本的 Executor 执行器，SimpleExecutor、ReuseExecutor、BatchExecutor。

1) SimpleExecutor：每执行一次 update 或 select，就开启一个 Statement 对象，用完立刻关闭 Statement 对象。

2) ReuseExecutor：执行 update 或 select，以 sql 作为 key 查找 Statement 对象，存在就使用，不存在就创建，用完后，不关闭 Statement 对象，而是放置于 Map 内，供下一次使用。简言之，就是重复使用 Statement 对象。

3) BatchExecutor：执行 update（没有 select，JDBC 批处理不支持 select），将所有 sql 都添加到批处理中（addBatch()），等待统一执行（executeBatch()），它缓存了多个 Statement 对象，每个 Statement 对象都是 addBatch()完毕后，等待逐一执行 executeBatch()批处理。与 JDBC 批处理相同。

4) 作用范围：Executor 的这些特点，都严格限制在 SqlSession 生命周期范围内。

21、Mybatis 中如何指定使用哪一种 Executor 执行器？

答：在 Mybatis 配置文件中，可以指定默认的 ExecutorType 执行器类型，也可以手动给 DefaultSqlSessionFactory 的创建 SqlSession 的方法传递 ExecutorType 类型参数。

22、Mybatis 是否可以映射 Enum 枚举类？

答：Mybatis 可以映射枚举类，不单可以映射枚举类，Mybatis 可以映射任何对象到表的一列上。

映射方式为自定义一个 TypeHandler，实现 TypeHandler 的 setParameter()和 getResult()接口方法。TypeHandler 有两个作用：

1) 一是完成从 javaType 至 jdbcType 的转换。

2) 二是完成 jdbcType 至 javaType 的转换，体现为 setParameter()和 getResult()两个方法，分别代表设置 sql 问号占位符参数和获取列查询结果。

23、Mybatis 映射文件中，如果 A 标签通过 include 引用了 B 标签的内容，请问，B 标签能否定义在 A 标签的后面，还是说必须定义在 A 标签的前面？

答：虽然 Mybatis 解析 Xml 映射文件是按照顺序解析的，但是，被引用的 B 标签依然可以定义在任何地方，Mybatis 都可以正确识别。

原理是，Mybatis 解析 A 标签，发现 A 标签引用了 B 标签，但是 B 标签尚未解析到，尚不存在，此时，Mybatis 会将 A 标签标记为未解析状态，然后继续解析余下的标签，包含 B 标签，待所有标签解析完毕，Mybatis 会重新解析那些被标记为未解析的标签，此时再解析 A 标签时，B 标签已经存在，A 标签也就可以正常解析完成了。

24、简述 Mybatis 的 Xml 映射文件和 Mybatis 内部数据结构之间的映射关系？

答：Mybatis 将所有 Xml 配置信息都封装到 All-In-One 重量级对象 Configuration 内部。在 Xml 映射文件中，<parameterMap>标签会被解析为 ParameterMap 对象，其每个子元素会被解析为 ParameterMapping 对象。<resultMap>标签会被解析为 ResultMap 对象，其每个子元素会被解析为 ResultMapping 对象。每一个<select>、<insert>、<update>、<delete>标签均会被解析为 MappedStatement 对象，标签内的 sql 会被解析为 BoundSql 对象。

25、为什么说 Mybatis 是半自动 ORM 映射工具？它与全自动的区别在哪里？

1) Hibernate 属于全自动 ORM 映射工具，使用 Hibernate 查询关联对象或者关联集合对象时，可以根据对象关系模型直接获取，所以它是全自动的。

2) 而 Mybatis 在查询关联对象或关联集合对象时，需要手动编写 sql 来完成，所以，称之为半自动 ORM 映射工具。

第四部分 企业级项目开发流程

1.软件生命周期（软件开发流程）

软件生命周期(SDLC，Systems Development Life Cycle,SDLC)是软件的产生直到报废或停止使用的生命周期。旧的解释是周期内有问题定义、可行性分析、总体描述、系统设计、编码、调试和测试、验收与运行、维护升级到废弃等阶段，这种按时间分程的思想方法是软件工程中的一种思想原则，即按部就班、逐步推进，每个阶段都要有定义、工作、审查、形成文档以供交流或备查，以提高软件的质量。

随着新的面向对象的设计方法和技术的成熟，早期软件生命周期设计方法的指导意义正在逐步减少或需要调整。不过从另一种意义来说，面向对象本身也是一种软件生命周期，传统的软件生命周期的概念仍是所有软件工程师非常重要的知识基础和工作指导。

软件生命周期的解释也应当调整。

以上旧的解释与下文的生命周期模型是不相容的，只与瀑布型生命周期模型及其衍生模型（比如 V 模型，W 模型）相符合，而与迭代为基本特征的生命周期模型是不符合的。新的情况应当是把迭代加入到阶段当中，如下：软件生命周期内有问题定义、可行性分析、总体描述、系统设计、编码、调试和测试、验收与运行、维护升级到废弃等阶段，也有将以上阶段的活动组合在内的迭代阶段，即迭代作为生命周期的阶段。

中文名

软件生命周期

外文名

SDLC

概　念

软件的产生直到报废的生命周期

别　称

软件生存周期或系统开发生命周期

思想方法

软件工程中的一种思想原则

用　途

确定软件的开发目标及其可行性

2.简介

软件生命周期又称为软件生存周期或系统开发生命周期，是软件的产生直到报废的生命周期，周期内有问题定义、可行性分析、总体描述、系统设计、编码、调试和测试、验收与运行、维护升级到废弃等阶段，这种按时间分程的思想方法是软件工程中的一种思想原则，即按部就班、逐步推进，每个阶段都要有定义、工作、审查、形成文档以供交流或备查，以提高软件的质量。但随着新的面向对象的设计方法和技术的成熟，软件生命周期设计方法的指导意义正在逐步减少。 生命周期的每一个周期都有确定的任务，并产生一定规格的文档（资料），提交给下一个周期作为继续工作的依据。按照软件的生命周期，软件的开发不再只单单强调“编码”，而是概括了软件开发的全过程。软件工程要求每一周期工作的开始只能必须是建立在前一个周期结果“正确”前提上的延续；因此，每一周期都是按“活动 —— 结果 —— 审核 —— 再活动 —— 直至结果正确”循环往复进展的。

3.阶段

同任何事物一样，一个软件产品或软件系统也要经历孕育、诞生、成长、成熟、衰亡等阶段，一般称为软件生存周期（软件生命周期）。把整个软件生存周期划分为若干阶段，使得每个阶段有明确的任务，使规模大，结构复杂和管理复杂的软件开发变的容易控制和管理。通常，软件生存周期包括：

一，问题定义。要求系统分析员与用户进行交流，弄清“用户需要计算机解决什么问题”然后提出关于“系统目标与范围的说明”，提交用户审查和确认。

二，可行性研究。一方面在于把待开发的系统的目标以明确的语言描述出来，另一方面从经济、技术、法律等多方面进行可行性分析。

三，需求分析。弄清用户对软件系统的全部需求，编写需求规格说明书和初步的用户手册，提交评审。

四，开发阶段。开发阶段由三个阶段组成：

1，设计

2，实现：根据选定的程序设计语言完成源程序的编码。

3，测试

五，维护：维护包括四个方面

1，改正性维护：在软件交付使用后，由于开发测试时的不彻底、不完全、必然会有一部分隐藏的错误被带到运行阶段，这些隐藏的错误在某些特定的使用环境下就会暴露。

2，适应性维护：是为适应环境的变化而修改软件的活动。

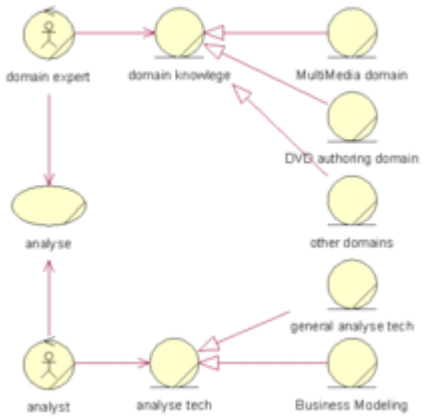
3，完善性维护 ：是根据用户在使用过程中提出的一些建设性意见而进行的维护活动。

4，预防性维护：是为了进一步改善软件系统的可维护性和可靠性，并为以后的改进奠定基础。

3.1 问题的定义及规划

此阶段是软件开发方与需求方共同讨论，主要确定软件的开发目标及其可行性。

3.2 需求分析



软件生命周期之需求分析

在确定软件开发可行的情况下，对软件需要实现的各个功能进行详细分析。需求分析阶段是一个很重要的阶段，这一阶段做得好，将为整个软件开发项目的成功打下良好的基础。"唯一不变的是变化本身。"，同样需求也是在整个软件开发过程中不断变化和深入的，因此我们必须制定需求变更计划来应付这种变化，以保护整个项目的顺利进行。软件需求定义是软件设计开发阶段的输入，为需求被翻译成为可以使软件建构功能的代码发挥作用。

3.3 软件设计

软件生命周期之软件设计

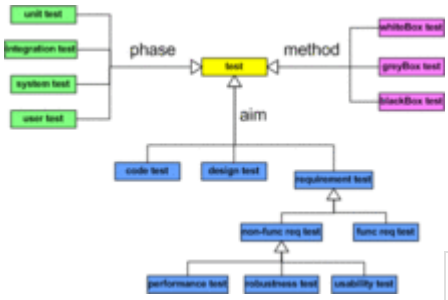
此阶段主要根据需求分析的结果，对整个软件系统进行设计，如系统框架设计，数据库设计等等。软件设计一般分为总体设计和详细设计。好的软件设计将为软件程序编写打下良好的基础。软件设计的核心在于把握好那些决定“服务质量”的因素，比如软件的性能，可扩展性，安全性，怎样划分模块的组成，怎样组织和封装软件的组件，以及其他一些虽然不作为软件主要应用的方面但会对其支持方面有所影响的方方面面。软件设计的原理包括抽象，分解和模块化，耦合和内聚，封装，充分性，完整性和原始性。软件设计主要关注软件的兼容性、可扩展性、容错性、可维护性、模块化、可靠性、可重用性、健壮性、安全性、

可用性和互操作性。耦合和内聚是两个用来评估软件设计质量的方法。

3.4 程序编码

此阶段是将软件设计的结果转换成计算机可运行的程序代码。在程序编码中必须要制定统一，符合标准的编写规范。以保证程序的可读性，易维护性，提高程序的运行效率。

3.5 软件测试



软件生命周期之软件测试

在软件设计完成后要经过严密的测试，以发现软件在整个设计过程中存在的问题并加以纠正。整个测试过程分单元测试、组装测试以及系统测试三个阶段进行。测试的方法主要有白盒测试和黑盒测试两种。在测试过程中需要建立详细的测试计划并严格按照测试计划进行测试，以减少测试的随意性。

3.6 软件交付

在软件测试证明软件达到要求后，软件开发者应向用户提交开发的目标安装程序、数据库的数据字典、《用户安装手册》、《用户使用指南》、需求报告、设计报告、测试报告等双方合同约定的产物。

《用户安装手册》应详细介绍安装软件对运行环境的要求、安装软件的定义和内容、在客户端、服务器端及中间件的具体安装步骤、安装后的系统配置。

《用户使用指南》应包括软件各项功能的使用流程、操作步骤、相应业务介绍、特殊提示和注意事项等方面的内容，在需要时还应举例说明。

3.7 软件验收

用户验收。

3.8 运行维护

软件维护是软件生命周期中持续时间最长的阶段，根据用户需求的变化或环境的变化，对应用程序进行全部或部分的修改。在软件开发完成并投入使用后，由于多方面的原因，软件不能继续适应用户的要求。要延续软件的使用寿命，就必须对软件进行维护。软件的维护包括纠错性维护和改进性维护两个方面。

4 周期模型

任何办公的流程处理；设计一种商务信函打印系统并投放市场。这个概念是不清晰的，但却是最高层的业务需求的原型。这个概念都会伴随着一个目的，例如在一个"银行押汇系统" 的目的是提高工作的效率。这个目的将会成为系统的核心思想，系统成败的评判标准。99 年政府部门上了大量的 OA 系统，学过一点 Lotus Notes 的人都发了财（IBM 更不用说了），但是更普遍的情况是，许多的政府部门原有的处理模式并没有变化，反而又加上了自动化处理的一套流程。提高工作效率的初衷却导致了完全不同的结果。这样的软件究竟是不是成功的呢？

从概念提出的那一刻开始，软件产品就进入了软件生命周期。在经历需求、分析、设计、实现、部署后，软件将被使用并进入维护阶段，直到最后由于缺少维护费用而逐渐消亡。这样的一个过程，称为"生命周期模型"（Life Cycle Model）。

典型的几种生命周期模型包括瀑布模型、快速原型模型、迭代模型。

4.1 瀑布模型

（Waterfall Model）首先由 Royce 提出。该模型由于酷似瀑布闻名。在该模型中，首先确定需求，并接受客户和 SQA 小组的验证。然后拟定规格说明，同样通过验证后，进入计划阶段…可以看出，瀑布模型中至关重要的一点是只有当一个阶段的文档已经编制好并获得 SQA 小组的认可才可以进入下一个阶段。这样，瀑布模型通过强制性的要求提供规约文档来确保每个阶段都能很好的完成任务。但是实际上往往难以办到，因为整个的模型几乎都是以文档驱动的，这对于非专业的用户来说是难以阅读和理解的。想象一下，你去买衣服的时候，售货员给你出示的是一本厚厚的服装规格说明，你会有什么样的感触。虽然瀑布模型有很多很好的思想可以借鉴，但是在过程能力上有天生的缺陷。

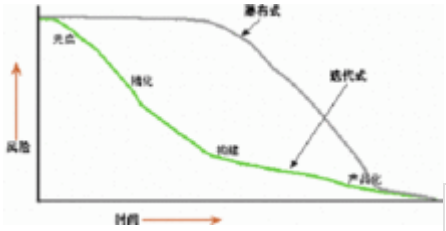
然而轻易抛弃瀑布模型的观点也是非常错误的，瀑布模型还是所有软件开发模型的基础，体现了软件开发的本质过程。对于一些大型 的软件项目，试图过于简化瀑布的前期的需求和设计阶段，用一个简单的原型或者迭代来模拟未来的系统，并试图帮助确认和挖掘客户的需求，是不可能的，不仅此时离客户的最终需求和隔山万千重，系统的架构也会随着过程而有很大被抛弃和大幅调整的过程，原型也就起不到原型的作用，成本和时间反而浪费，

所以前期的功课还是少不了的，尤其对于复杂系统。即使对于简单如定制一件衣服，在给客户提出修改的时候，它要基本是一件衣服，而不是几块布片，否则客户无从提出进一步的需求，前期的功夫也是白费的。

4.2 迭代式模型

迭代式模型

迭代式模型是是 RUP（Rational Unified Process，统一软件开发过程，统一软件过程)推荐的周期模型，也是我们在这个系列文章讨论的基础。在 RUP 中，迭代被定义为：迭代包括产生产品发布（稳定、可执行的产品版本）的全部开发活动和要使用该发布必需的所有其他外围元素。所以，在某种程度上，开发迭代是一次完整地经过所有工作流程的过程：（至少包括）需求工作流程、分析设计工作流程、实施工作流程和测试工作流程。实质上，它类似小型的瀑布式项目。RUP 认为，所有的阶段（需求及其它）都可以细分为迭代。每一次的迭代都会产生一个可以发布的产品，这个产品是最终产品的一个子集。迭代的思想如图所示。



迭代和瀑布的区别

迭代和瀑布的最大的差别就在于风险的暴露时间上。“任何项目都会涉及到一定的风险。如果能在生命周期中尽早确保避免了风险，那么您的计划自然会更趋精确。有许多风险直到已准备集成系统时才被发现。不管开发团队经验如何，都绝不可能预知所有的风险。”

由于瀑布模型的特点（文档是主体），很多的问题在最后才会暴露出来，为了解决这些问题的风险是巨大的。"在迭代式生命周期中，您需要根据主要风险列表选择要在迭代中开发的新的增量内容。每次迭代完成时都会生成一个经过测试的可执行文件，这样就可以核实是否已经降低了目标风险。"

4.3 快速原型模型

快速原型（Rapid Prototype）模型在功能上等价于产品的一个子集。注意，这里说的是功能上。瀑布模型的缺点就在于不够直观，快速原型法就解决了这个问题。一般来说，根据客户的需要在很短的时间内解决用户最迫切需要，完成一个可以演示的产品。这个产品只

是实现部分的功能（最重要的）。它最重要的目的是为了确定用户的真正需求。在我的经验中，这种方法非常的有效，原先对计算机没有丝毫概念的用户在你的原型面前往往口若悬河，有些观点让你都觉得非常的吃惊。在得到用户的需求之后，原型将被抛弃。因为原型开发的速度很快，设计方面是几乎没有考虑的，如果保留原型的话，在随后的开发中会为此付出极大的代价。至于保留原型方面，也是有一种叫做增量模型是这么做的，但这种模型并不为大家所接受，不在我们的讨论之内。 上述的模型中都有自己独特的思想，其实现在的软件组织中很少说标准的采用那一种模型的。模型和实用还是有很大的区别的。

软件生命周期模型的发展实际上是体现了软件工程理论的发展。在最早的时候，软件的生命周期处于无序、混乱的情况。一些人为了能够控制软件的开发过程，就把软件开发严格的区分为多个不同的阶段，并在阶段间加上严格的审查。这就是瀑布模型产生的起因。瀑布模型体现了人们对软件过程的一个希望：严格控制、确保质量。可惜的是，现实往往是残酷的。瀑布模型根本达不到这个过高的要求，因为软件的过程往往难于预测。反而导致了其它的负面影响，例如大量的文档、繁琐的审批。因此人们就开始尝试着用其它的方法来改进或替代瀑布方法。例如把过程细分来增加过程的可预测性。

4.4 螺旋模型

1988 年，Barry Boehm 正式发表了软件系统开发的"螺旋模型"，它将瀑布模型和快速原型模型结合起来，强调了其他模型所忽视的风险分析，特别适合于大型复杂的系统。

螺旋模型沿着螺线进行若干次迭代，图中的四个象限代表了以下活动：

（1） 制定计划：确定软件目标，选定实施方案，弄清项目开发的限制条件；

（2） 风险分析：分析评估所选方案，考虑如何识别和消除风险；

（3） 实施工程：实施软件开发和验证；

（4） 客户评估：评价开发工作，提出修正建议，制定下一步计划。

螺旋模型由风险驱动，强调可选方案和约束条件从而支持软件的重用，有助于将软件质量作为特殊目标融入产品开发之中。但是，螺旋模型也有一定的限制条件，具体如下：

（1） 螺旋模型强调风险分析，但要求许多客户接受和相信这种分析，并做出相关反应是不容易的，因此，这种模型往往适应于内部的大规模软件开发。

（2） 如果执行风险分析将大大影响项目的利润，那么进行风险分析毫无意义，因此，螺旋模型只适合于大规模软件项目。

（3） 软件开发人员应该擅长寻找可能的风险，准确地分析风险，否则将会带来更大的风险

一个阶段首先是确定该阶段的目标，完成这些目标的选择方案及其约束条件，然后从风险角度分析方案的开发策略，努力排除各种潜在的风险，有时需要通过建造原型来完成。如果某些风险不能排除，该方案立即终止，否则启动下一个开发步骤。最后，评价该阶段的结果，并设计下一个阶段。

第五部分 JavaEE 企业开发常用工具

1.Java 集成化开发工具

1、Eclipse；
2、MyEclipse；
3、IntelliJ IDEA；
4、Jbuilder；
5、NetBeans；

2.常用几种 Java Web 容器

Web 服务器是运行及发布 Web 应用的容器，只有将开发的 Web 项目放置到该容器中，才能使网络中的所有用户通过浏览器进行访问。开发 Java Web 应用所采用的服务器主要是与 JSP/Servlet 兼容的 Web 服务器，比较常用的有 Tomcat、Resin、JBoss、WebSphere 和 WebLogic 等，下面将分别进行介绍。

1、Tomcat 服务器（市场占有率）：目前最为流行的 Tomcat 服务器是 Apache-Jarkarta 开源项目中的一个子项目，是一个小型、轻量级的支持 JSP 和 Servlet 技术的 Web 服务器，也是初学者学习开发 JSP 应用的首选。Tomcat 的安装市场占有率超过 50% 大关，达到 58.6%，是毫无疑问的最大赢家。

2、JBoss 服务器：JBoss 是一个种遵从 JavaEE 规范的、开放源代码的、纯 Java 的 EJB 服务器，对于 J2EE 有很好的支持。JBoss 采用 JML API 实现软件模块的集成与管理，其核心服务又是提供 EJB 服务器，不包含 Servlet 和 JSP 的 Web 容器，不过它可以和 Tomcat 完美结合。市场占有率 15.7%。

3、Oracle WebLogic 服务器：WebLogic 是 BEA 公司的产品，可进一步细分为 WebLogic Server、WebLogic Enterprise 和 WebLogic Portal 等系列，其中 WebLogic Server 的功能特别强大。WebLogic 支持企业级的、多层次的和完全分布式的 Web 应用，并且服务器的配置简单、界面友好。对于那些正在寻求能够提供 Java 平台所拥有的一切应用服务器的用户来说，WebLogic 是一个十分理想的选择。市场占有率接近 10%。

4、Jetty：市场占有率为 8%

5、GlassFish：市场占有率为 5%。

6、Resin 服务器：Resin 是 Caucho 公司的产品，是一个非常流行的支持 Servlet 和 JSP 的服务器，速度非常快。Resin 本身包含了一个支持 HTML 的 Web 服务器，这使它不仅可以显示动态内容，而且显示静态内容的能力也毫不逊色，因此许多网站都是使用 Resin 服务器构建。

7、WebSphere 服务器：WebSphere 是 IBM 公司的产品，可进一步细分为 WebSphere Performance Pack、Cache Manager 和 WebSphere Application Server 等系列，其中 WebSphere Application Server 是基于 Java 的应用环境，可以运行于 Sun Solaris、

Windows NT 等多种操作系统平台，用于建立、部署和管理 Internet 和 Intranet Web 应用程序。

3.常见的版本控制管理工具

1、Git（开源分布式）：Git 是一个开源的分布式版本控制系统，用以有效、高速的处理从很小到非常大的项目版本管理。
2、SVN（常用）：SVN（Subversion） 是一种版本管理系统，其前身是 CVS。SVN 是根据 CVS 的功能为基础来设计的，它除包括了 CVS 的大多数特点外，还有一些新的功能，如：文件目录可以方便的改名、基于数据库的版本库、操作速度提升、权限管理更完善等。
3、CVS：CVS 是一个典型的服务器/客户端软件，有 Unix 版本的 CVS 、Linux 版本的 CVS 和 Windows 版本的 CVS。CVS 支持远程管理，项目组分布开发时一般都采用 CVS。
4、VSS：Visual SourceSafe：微软的版本控制工具，仅支持 Windows 操作系统。虽然简单好用，但是仅适用于团队级开发，不能胜任企业级的开发工作。
5、其他版本控制工具：ClearCase（闭源集中式）、StarTeam（闭源集中式）、Mercurial（开源分布式）、Monotone（开源分布式）

4.Java 程序员常用的 10 大构建工具

1、Apache Maven：主要用于构建 Java 项目的自动化工具。
2、Hudson：用 Java 编写的持续集成（CI）工具。
3、Jenkins：用 Java 编写的一个开源持续集成工具。项目是在和 Oracle 发生争执后的来自于 Hudson 的分支。
4、Gradle：一个开源的自动化构建系统，建立在 Apache Ant 和 Maven Apache 概念的基础上，并引入了基于 Groovy 的特定领域语言（DSL），而不是使用 Apache Maven 宣布的项目配置 XML 形式。
5、Apache Ant：用于自动化软件构建过程的软件工具，源于 2000 年初的的 Apache Tomcat 项目。
6、SBT：用于 Scala 和 Java 项目的开源构建工具，类似于 Java 的 Maven 和 Ant。
7、TeamCity：来自于 JetBrains 的一个基于 Java 构建的管理和持续集成服务器。
8、Grape：嵌入到 Groovy 的 JAR 依赖项管理器。
9、Ivy：Apache Ant 项目的一个子项目，一个可传递的依赖项管理器。
10、Leiningen：一个自动化构建和依赖性管理工具，用于使用 Clojure 编程语言写的软件项目。

5.常见数据库

1、SQL Server
2、MySQL
3、DB2
4、Oracle
5、Sybase

6.数据库可视化管理软件

1、PL/SQLDeveloper：用于管理 Oracle 数据库
2、navicat for mysql：用于 MySql 数据库

7.其他开发中常用工具

1、PowerDesigner；
2、Xmind；
3、有道云笔记；
4、Jad 用于反编译 Java 类。你可以用 Jad 以纯文本的形式命令和阅读代码；
5、Notepad++；
6、EditPlus；
7、Javadoc：Javadoc 是 Oracle 提供的文档生成器。它可以将特殊格式的注释解析为 HTML 文档。
8、Junit：JUnit 是用于编写和运行单元测试的开源框架。
9、StarUML：StarUML 是一款开放源码的 UML 开发工具，是由韩国公司主导开发出来的产品，是一种创建 UML 类图，生成类图和其他类型的统一建模语言(UML)图表的工具。