如何做到招聘要求中的「要有扎实的 **Java** 基础」

来历

本文来自于一次和群里猿友的交流，具体的情况且听LZ慢慢道来。

一日，LZ在群里发话，“招人啦。”

然某群友曰，“群主，俺想去。”

LZ回之，“你年几何？”

群友曰，“两年也。”

LZ憾言之，“惜了，三至五为佳。然如汝有扎实之基础，且附一技之长，倒也并非不可呀。”

群友惑，问之，“何为一技之长？”

LZ抚须答曰，“皆可，吾之一技即为写。”

群友又问之，“何为基础扎实？”

LZ抚**答曰，“玩好JDK！”

引言

好了，LZ终于特么可以说人话了，请原谅LZ的文言文不好，这逼装的好累。其实很多人对于公司的招聘要求中写的，要有“扎实的Java基础”，都很迷茫。

这特么到底啥意思？什么东西算作Java基础？学到什么程度才算扎实？

这些问题的答案，LZ已经用文言文告诉你了，咳咳，答案就是玩好JDK就可以了。

不过JDK这东西包含的东西实在太多，群里的猿友还是拎不清到底应该学哪个，所以，LZ就专门写一篇文章，来讲一讲JDK应该怎么玩。

或许有的猿友会问，“林萧是谁？”

恩，问得好！林萧就是某程序员小说的主角啦，传言无所不知，看看你就知道咯。

玩好**JDK**

在详细讲解JDK之前，LZ首先要强调下，本文的内容，都是LZ个人的主观判断。因此，各位猿友务必结合自己的判断之后，再决定是否要学习，以及学习到什么程度。

此外，本文只是告诉你学什么，学到什么程度，本文不会包含太多具体的技术细节讲解。

JDK其实就是Java SE Development Kit的缩写，要玩好这东西可不简单。JDK主要包含了三部分，第一部分就是**Java**运行时环境，这其实就是JVM。此外，第二部分就是**Java**的基础类库，这个类库的数量还是非常可观的。最后，第三部分就是**Java**的开发工具，它们都是辅助你更好的使用Java的利器。

那么很显然，要玩好JDK，就是要玩好JDK的这三部分。接下来，咱们就逐个的来说一下，每一个部分要学什么，学到什么程度。

第一部分：Java运行时环境

这一部分其实就是常说的jre，而它的核心其实就一个东西，就是JVM。

JVM这个东西，它的重要性LZ不想再强调了，在之前的那篇《如何准备阿里社招面试，顺谈 Java 程序员学习中各阶段的建议》中，LZ就说过，JVM那本书甚至比《Thinking in java》还重要，这已经足见LZ多么看重JVM了。

当然了，只是LZ看重，当然没什么卵用，但只要Java稍微高级一点点的职位，这部分基本上都是面试必问内容，这更加说明了JVM的重要性。

所以，对于**JVM**，没什么可说的，就是往死里学，往死里研究，能有多深就多深！

第二部分：Java的基础类库

Java的基础类库从你刚开始学Java就开始接触了，但是，直到你结束整个Java生涯，你都不一定能把所有的类都见一遍。

这说明了什么？是不是说明了Java基础类库的庞大？

错！大错特错！这其实真正说明的是，有很多类你完全不必要care它们，因为它们从你接触Java到放弃，都不一定能接触到。

所以，搞清楚哪些类重要，哪些类不重要，就非常有必要了。这可以让你以最短的时间，做最有价值的事。

首先我们来看看，Java基础类库的包都有哪些，为了使得本文更有代表性，我们取JDK6的包列表。

```
java.applet
java.awt
java.awt.color
java.awt.datatransfer
java.awt.dnd
java.awt.event
java.awt.font
java.awt.geom
java.awt.im
java.awt.im.spi
java.awt.image
java.awt.image.renderable
java.awt.print
java.beans
java.beans.beancontext
java.io
java.lang
java.lang.annotation
java.lang.instrument
java.lang.management
java.lang.ref
java.lang.reflect
java.math
java.net
java.nio
java.nio.channels
java.nio.channels.spi
java.nio.charset
java.nio.charset.spi
java.rmi
java.rmi.activation
java.rmi.dgc
java.rmi.registry
java.rmi.server
java.security
java.security.acl
java.security.cert
java.security.interfaces
java.security.spec
java.sql
java.text
java.text.spi
java.util
java.util.concurrent
java.util.concurrent.atomic
java.util.concurrent.locks
java.util.jar
java.util.logging
java.util.prefs
java.util.regex
java.util.spi
java.util.zip

javax.accessibility
```

```
javax.activation
javax.activity
javax.annotation
javax.annotation.processing
javax.crypto
javax.crypto.interfaces
javax.crypto.spec
javax.imageio
javax.imageio.event
javax.imageio.metadata
javax.imageio.plugins.bmp
javax.imageio.plugins.jpeg
javax.imageio.spi
javax.imageio.stream
javax.jws
javax.jws.soap
javax.lang.model
javax.lang.model.element
javax.lang.model.type
javax.lang.model.util
javax.management
javax.management.loading
javax.management.modelmbean
javax.management.monitor
javax.management.openmbean
javax.management.relation
javax.management.remote
javax.management.remote.rmi
javax.management.timer
javax.naming
javax.naming.directory
javax.naming.event
javax.naming.ldap
javax.naming.spi
javax.net
javax.net.ssl
javax.print
javax.print.attribute
javax.print.attribute.standard
javax.print.event
javax.rmi
javax.rmi.CORBA
javax.rmi.ssl
javax.script
javax.security.auth
javax.security.auth.callback
javax.security.auth.kerberos
javax.security.auth.login
javax.security.auth.spi
javax.security.auth.x500
javax.security.cert
javax.security.sasl

javax.sound.midi
```

```
javax.sound.midi.spi
javax.sound.sampled
javax.sound.sampled.spi
javax.sql
javax.sql.rowset
javax.sql.rowset.serial
javax.sql.rowset.spi
javax.swing
javax.swing.border
javax.swing.colorchooser
javax.swing.event
javax.swing.filechooser
javax.swing.plaf
javax.swing.plaf.basic
javax.swing.plaf.metal
javax.swing.plaf.multi
javax.swing.plaf.synth
javax.swing.table
javax.swing.text
javax.swing.text.html
javax.swing.text.html.parser
javax.swing.text.rtf
javax.swing.tree
javax.swing.undo
javax.tools
javax.transaction
javax.transaction.xa
javax.xml
javax.xml.bind
javax.xml.bind.annotation
javax.xml.bind.annotation.adapters
javax.xml.bind.attachment
javax.xml.bind.helpers
javax.xml.bind.util
javax.xml.crypto
javax.xml.crypto.dom
javax.xml.crypto.dsig
javax.xml.crypto.dsig.dom
javax.xml.crypto.dsig.keyinfo
javax.xml.crypto.dsig.spec
javax.xml.datatype
javax.xml.namespace
javax.xml.parsers
javax.xml.soap
javax.xml.stream
javax.xml.stream.events
javax.xml.stream.util
javax.xml.transform
javax.xml.transform.dom
javax.xml.transform.sax
javax.xml.transform.stax
javax.xml.transform.stream

javax.xml.validation
```

```
javax.xml.ws
javax.xml.ws.handler
javax.xml.ws.handler.soap
javax.xml.ws.http
javax.xml.ws.soap
javax.xml.ws.spi
javax.xml.ws.wsaddressing
javax.xml.xpath
org.ietf.jgss
org.omg.CORBA
org.omg.CORBA_2_3
org.omg.CORBA_2_3.portable
org.omg.CORBA.DynAnyPackage
org.omg.CORBA.ORBPackage
org.omg.CORBA.portable
org.omg.CORBA.TypeCodePackage
org.omg.CosNaming
org.omg.CosNaming.NamingContextExtPackage
org.omg.CosNaming.NamingContextPackage
org.omg.Dynamic
org.omg.DynamicAny
org.omg.DynamicAny.DynAnyFactoryPackage
org.omg.DynamicAny.DynAnyPackage
org.omg.IOP
org.omg.IOP.CodecFactoryPackage
org.omg.IOP.CodecPackage
org.omg.Messaging
org.omg.PortableInterceptor
org.omg.PortableInterceptor.ORBInitInfoPackage
org.omg.PortableServer
org.omg.PortableServer.CurrentPackage
org.omg.PortableServer.POAManagerPackage
org.omg.PortableServer.POAPackage
org.omg.PortableServer.portable
org.omg.PortableServer.ServantLocatorPackage
org.omg.SendingContext
org.omg.stub.java.rmi
org.w3c.dom
org.w3c.dom.bootstrap
org.w3c.dom.events
org.w3c.dom.ls
org.xml.sax
org.xml.sax.ext
org.xml.sax.helpers
```

怎么样？是不是被吓到了？这么多包，而且还这么多陌生的包名，有的连见都没见过，这特么怎么玩？

不要着急，LZ先带你把这些包给分下级别，LZ将这些包一共分为四个级别。

第一级别：精读源码

该级别包含的包如下：

```
java.io
java.lang
java.util
```

精读源码，这是要求最高的级别。但是，要求你精读源码并不意味着，这些类就是最重要的。而是因为，LZ觉得这些类比较常用，而且比较简单，看看它们的源码有助于锻炼你看源码的感觉，也了解一下大神们写代码的风格。

看这些源码的目的，更多是为了增加你的阅读代码能力。而且，LZ这里必须要强调一下，像Exception和Error这一类的，就不用读源码了，亲。

其实上面三个包都有一个共同点，那就是这三个包，基本上都是你最常用的了。lang包不用说了，你随便写点啥都得用到，io包和util包也是你平时读写文件和使用数据结构必不可少的。

看源码从这些常用的包下手找手感，LZ个人觉得再合适不过。

第二级别：深刻理解

该级别包含的包如下：

```
java.lang.reflect
java.net
javax.net.*
java.nio.*
java.util.concurrent.*
```

深刻理解，这个看似比精读源码要求低的级别，其实恰恰是最重要的。这个级别要求的类，全都是一些进阶到高级所必须了解的。

当然了，这里要强调一点的是，LZ说这些类要深刻理解，而没说要看它们源码，并不是说这些类的源码不能看，或者看了没用。而是这些类的源码往往非常复杂，要了解清楚细节花费的时间是非常多的，因此，花费巨量的时间去研究这么复杂的代码其实没必要的。

不过，如果你在使用这些类的过程中，遇到了问题，这个时候如果看它们源码可以解决的话，那就不要再矜持了，果断看看源码解决你的问题吧，这是最适合的看源码的契机了。

小小的透露一下，LZ看过的JDK源码，基本上全是这么看过来的。遇到了问题不要百度和谷歌，看源码能解决你90%的问题。

此外，看到这四个包的名字，不难看出它们各自代表了什么。reflect代表了反射，net代表了网络IO，nio代表了非阻塞io，concurrent代表了并发。

这四个家伙可以说每一个都够面试官问上半天的，而且，这四个包的内容，如果你要深刻理解的话，其实还牵扯了很多其它的知识。

举个例子，反射你要了解清楚的话，你是不是要搞明白JVM的类加载机制？网络IO要搞清楚的话，你是不是要清楚TCP/IP和HTTP、HTTPS？包括并发包，如果你要搞清楚的话，是不是要了解并发的相关知识？

因此，这四个包要彻底搞清楚，还是需要花费一定时间和精力的。

但是，请相信LZ，这绝对是值得的，甚至可以说，这四个包用的够不够叼，基本决定了一个Java程序员所处的档次。

第三级别：会用即可

该级别包含的包如下：

```
java.lang.annotation
javax.annotation.*
java.lang.ref
java.math
java.rmi.*
javax.rmi.*
java.security.*
javax.security.*
java.sql
javax.sql.*
javax.transaction.*
java.text
javax.xml.*
org.w3c.dom.*
org.xml.sax.*
javax.crypto.*
javax.imageio.*
javax.jws.*
java.util.jar
java.util.logging
java.util.prefs
java.util.regex
java.util.zip
```

会用即可，这个级别的要求很显然了，就是会用就可以了。这些包大部分都是在特定的情况下会用到，但却不会时刻用到。

就像sql包和transaction包，就是操作数据库时用到的。而xml、dom和sax这些，都是操作xml时用到的。其它的包也都是类似的，有使用注解时用的，有远程方法调用时用的，也有涉及到加密时用到的等等。

这些包在面试时一般不会问到，所以它们的重要性自然要低很多，而且也不太需要刻意的去学，用到了研究一下，会用即可。

第四级别：请无视它

该级别包含了所有以上没有提到的包。

这个级别就更不用解释了，到现在还没提到的包，基本上可以忽略，因为你基本上不太可能用到它们，比如swing、awt这些玩意儿。

什么？你说你经常用它们？

如果真的是这样的话，那么LZ只能遗憾的告诉你，赶紧跳槽吧，你待在这公司没啥前途的，0-0。

第三部分：**Java**的开发工具

这些开发工具主要就是辅助你开发的了，javac应该是最常用的一个了，虽然你几乎不用手动执行它。

此外，其实还有一些比较实用的工具，可以帮助你排查问题。而且有的面试官，也会问你这类问题，比如问你平时都用什么工具排查问题。

LZ觉得比较实用的几个工具主要有jmap、jconsole、jstack、jvisualvm，至于这几个工具有什么作用，LZ这里就不提了，如果你要了解这些命令的详细内容，可以去谷歌或者官网上找，还是非常好找的。

当然，如果你有兴趣的话，也可以自己去JDK的bin目录下找找，看有没有什么更好玩的工具。

小结

好了，到这里基本上就把“玩好JDK”这件事说完了。

总的来说，第一级别和第二级别是最重要的。

更简单的说，可以把第一级别的那些包称作基础，第二级别的那些包称为进阶。至于第三级别和第四级别的那些包，就没什么可说的了。

当然了，最重要的还是那万年不变的JVM，请记住了，**JVM**才是你**Java**根基的根本，就是再牛逼的类，没了JVM它也就是一个无用的class文件而已。

如果你想拥有扎实的Java基础，那就抓紧玩好JDK吧。玩好JDK以后，不要忘了学个一技之长，就可以来LZ的公司面试咯。

嗯，这波招聘广告插入的，很隐晦，没毛病，0-0。

下期咱们屌程见了，各位！