

Java：一个帝国的诞生

C语言帝国的统治

现在是公元1995年，C语言帝国已经统治了我们20多年，实在是太久了。

1972年，随着C语言的诞生和Unix的问世，帝国迅速建立统治，从北美到欧洲，从欧洲到亚洲，无数程序员臣服在他的脚下。

帝国给我们提供了极好的福利：贴近硬件，运行极快，效率极高。

使用这些福利，程序员们用C开发了很多系统级软件，操作系统，编译器，数据库，网络系统.....

但是帝国也给我们安上了两个沉重的枷锁：指针和内存管理

虽然指针无比强大，能直接操作内存，但是帝国却没有给我们工具去做越界的检查，导致很多新手程序员轻易犯错。

至于内存管理，帝国更完全是放任的态度：你自己分配的空间，自己去释放！

更要命的是这些问题在编译期发现不了，在运行时才会突然暴露，常常让我们手忙脚乱，昏天黑地去调试。

我们的大量时间和宝贵的精力都被浪费在小心翼翼的处理指针和内存分配上。

每个程序员都被这两个东西搞的焦头烂额！

帝国宣称的可移植性骗了我们，他宣称我们在一个机器上写的程序，只要在另外一个机器上编译就可以了，实际上不是这样。他要求我们尽量用标准的C函数库。其次，如果遇到了一些针对特定平台的调用，需要对每个平台都得写一份！有一点点小错误，都会导致编译失败。

1982年，帝国又推出了一门新的语言C++，添加了面向对象的功能，兼容C，有静态类型检查，性能也很好。

但是这门新的语言实在是太复杂了，复杂到比我聪明的多的人都没有办法完全掌握这门语言，它的很多特性复杂的让人吃惊。

C++在图形领域和游戏上取得了一些成功，但是我一直学不好它。

反抗

我决定反抗这个庞大的帝国，我偷偷的带领着一帮志同道合的兄弟离开了，我们要新建一块清新自由的领地。

为了吸引更多的程序员加入我们，我们要建立一个新的语言，这个语言应该有这样的特性：

语法有点像C，这样大家容易接受

没有C语言那样的指针

再也不要考虑内存管理了，实在受不了了

真正的可移植性，编写一次，到处运行

面向对象

类型安全

还有，我们要提供一套高质量的类库，随语言发行。

我想把这个语言命名为C++-，即C++减减，因为我想在C++的基础上改进，把它简化。

后来发现不行，设计理念差别太大。

干脆重启炉灶。

我看到门口的一棵橡树，就把这个语言叫做Oak。

但是后来发布的时候，发现Oak已经被别人用了，我们讨论很久，最终决定把这门新的语言叫做Java。

为了实现跨平台，我们在操作系统和应用程序之间增加了一个抽象层：Java 虚拟机

用Java写的程序都跑在虚拟机上，除非个别情况，都不用看到操作系统。

一鸣惊人

为了吸引更多的人加入我们的新领地，我们决定搞一个演示，向大家展示Java 的能力。

出世未久的Java其实还远不完善。搞点什么呢？

我们把眼光盯上了刚刚兴起的互联网，1995年的网页简单而粗糙，缺乏互动性。于是我们在浏览器上弄了个小插件，把java 运行环境放了上去。

然后在上面开发了一个图形界面的程序(Applet)，让它看起来美轮美奂，震撼人心。

每一个看到他的程序员都会发出“Wow”的惊叹！为之倾倒。

Java 活了！

通过Applet，无数的程序员看到了Java这门语言，了解了这门语言特性以后，很多无法忍受C帝国暴政的程序员，很快加入了我们，我们的领地开始迅速扩大。

连C语言帝国里的一些商业巨头也纷纷来和我们合作，其中就包括Oracle，微软这样的巨头，微软的头领Bill Gates还说：这是迄今为止设计的最好的语言！

但是Bill Gates非常的不地道，买了我们的Java 许可以后，虽然在自家的浏览器上也支持Applet,但是他们却偷偷的试图修改Java,想把Java绑死在自家的操作系统上赚钱，Java会变的不可移植。

这是我们难于忍受的，我们和微软发起了一场旷日持久的游击战争，逼着微软退出了Java领域，开发了自己的.NET，这是后话。

开拓疆土

从1995年到1997年，我们依靠Java 不断的攻城略地，开拓疆土，我们王国的子民不断增加，达到了几十万之众，已经是一个不可忽视的力量了。

但是大家发现，Java除了Applet, 以及一些小程序之外，似乎干不了别的事情。

C帝国的人还不断的嘲笑我们慢，像个玩具。

到了1998年，经过密谋，我们Java 王国决定派出三只军队向外扩展：

Java 2 标准版(J2SE)：去占领桌面

Java 2 移动版(J2ME)：去占领手机

Java 2 企业版(J2EE)：去占领服务器

其中的两只大军很快败下阵来。

J2SE 的首领发现，开发桌面应用的程序员根本接受不了Java，虽然我们有做的很优雅的Swing 可以开发界面，但是开发出的界面非常难看，和原生的桌面差距很大。尤其是为了运行程序还得安装一个虚拟机，大家都受不了。

J2ME也是，一直不受待见，当然更重要的原因是乔布斯还没有重新发明手机，移动互联网还没有启动。

失之东隅，收之桑榆，J2EE赶上了好时候，互联网大发展，大家忽然发现，Java简直是为写服务器端程序所发明的！

强大，健壮，安全，简单，跨平台！

在J2EE规范的指导下，特别适合团队开发复杂的大型项目。

我们授权BEA公司第一个使用J2EE许可证，推出了Weblogic, 凭借其集群功能，第一次展示了复杂应用的扩展性和高可用性。【三高：高可用，高性能，高并发】

这个后来被称为中间件的东西把程序员从事务管理，安全管理，权限管理等方面解放出来，让他们专注于业务开发。这立刻捕获了大量程序员的心。

很快Java 王国的子民就达到数百万之众。

榜样的力量是无穷的，很快其他商业巨头也纷纷入场，尤其是IBM，在Java 上疯狂投入，不仅开发了自己的应用服务器 Websphere, 还推出了Eclipse这个极具魅力的开源开发平台。

当然IBM利用java 获得了非常可观的效益，软件+硬件+服务 三驾马车滚滚向前，把IBM推向了一个新的高峰。

帝国的诞生

大家也没有想到，除了商业巨头以外，程序员们也会对Java王国 这么热爱，他们基于Java 开发了巨多的平台，系统，工具，例如：

构建工具: Ant, Maven, Jenkins

应用服务器: Tomcat, Jetty, Jboss, Websphere, weblogic

Web开发: Struts, Spring, Hibernate, myBatis

开发工具: Eclipse, Netbean, IntelliJ IDEA, JBuilder

。。。。等等等等。。。。

并且绝大部分都是开源的！

微软眼睁睁的看着服务器端的市场被Java 王国占据，岂能善罢甘休？他们赶紧推出.NET来对抗，但我们已经不在乎了，因为他的系统是封闭的，所有的软件都是自家的：

开发工具是Visual Studio, 应用服务器是IIS, 数据库是SQL Server，只要你用.NET，基本上就会绑定微软。

另外他们的系统只能运行在Windows服务器上，这个服务器在高端市场的占有率实在是太低了。

2005年底，一个新的王国突然崛起，他们号称开发效率比Java 快5-10倍，由此吸引了大批程序员前往加盟。

这个新的王国叫做Ruby on Rails, 它结合了PHP体系的优点（快速开发）和Java体系的优点（程序规整），特别适合快速的开发简单的Web网站。

虽然发展很快，但没有对Java 王国产生实质性的威胁，使用Ruby on Rails搭建大型商业系统的还很少。

除了Ruby on Rails，还有PHP，Python，都适合快速开发不太复杂的Web系统。但是关键的，复杂的商业系统开发还是Java王国的统治之下。所以我们和他们相安无事。

2006年，一只叫Hadoop的军队让Java王国入侵了大数据领域，由于使用Java语言，绝大多数程序员在理解了Map/Reduce，分布式文件系统在Hadoop中的实现以后，很快就能编写处理海量数据的程序，Java王国的领地得到了极大的扩展。

2008年，一个名叫Android的系统横空出世，并且随着移动互联网的爆发迅速普及，运行在Android之上的正是Java！

Java王国在Google的支持下，以一种意想不到的方式占领了手机端，完成了当年J2ME壮志未酬的事业！

到今年为止，全世界估计有1000万程序员加入了Java王国，它领土之广泛，实力之强大，是其他语言无法比拟的。

Java占据了大部分的服务器端开发，尤其是关键的复杂的系统，绝大多数的手机端，以及大部分的大数据领域。

一个伟大的帝国诞生了。

Java的特性和优势

简单性

就是c++语法的纯净版。没有头文件，指针运算，结构，联合，操作符重载，虚基类等等。由于语法基于c，因此学习起来完全不费力。

面向对象

面向对象是一种程序设计技术，他将重点放在数据（即对象）和对象之间的接口上。模拟人的思维写程序，万物皆对象！

可移植性(跨平台性)

这是JAVA的一个重要的优势。JAVA代码或者说字节码、二进制码可以跨平台的移植，而不用管具体的操作系统和硬件环境。

“一次编写，随处运行”：“write once, run anywhere”

JAVA在设计时就注重移植和跨平台性。比如：JAVA的Int永远都是32位。不像c++可能是16，32，可能是根据编译器厂商规定的变化。这样的话程序的移植就会非常麻烦。

高性能

JIT(JUST IN TIME)即时编译。将一些“热点”字节码编译成本地机器码，并将结果缓存起来，在需要的时候重新调用。这样的话，使JAVA程序的执行效率大大提高，某些代码甚至接近c++的效率。随着这种技术的一天完善，也许有一天会超越编译代码的效率。

分布式

JAVA是为internet的分布式环境设计的，因为他能够处理tcp/ip协议。事实上，通过url访问一个网络资源和访问本地文件是一样简单的。Java还支持远程方法调用(RMI,remote method Invocation)，使程序能够通过网络调用方法。

动态性

就是在需要时将某些代码添加到正在运行的程序中。反射机制。当需要把某些代码添加到正在运行的程序中时，动态性是一个非常重要的特性。Java的动态特性是其面向对象设计方法的扩展。它允许程序动态地装入运行过程中所需要的类，这是C++语言进行面向对象程序设计所无法实现的

多线程

多线程的使用可以带来更好的交互响应和实时行为。多线程的简单性是Java成为主流服务器端开发语言的主要原因之一。

安全性

Java适合于网络/分布式环境，为了达到这个目标，在安全性方面投入了很大的精力，使Java可以构建防病毒，防篡改的系统。

健壮性

Java是一种健壮的语言，吸收了C/C++ 语言的优点，但去掉了其影响程序健壮性的部分（如：指针、内存的申请与释放等）。Java程序不可能造成计算机崩溃。Java系统仔细检测对内存的每次访问，确认它是合法的，而且不致引起任何问题。不过，即使Java程序也可能有错误。如果出现某种出乎意料之事，程序也不会崩溃，而是把该例外抛弃。再通过异常处理机制，程序就会发现这类例外，并加以处理。

当然，我们学习他这门语言，最初的目的，是因为使用的人多，我们需要去掌握学习，但是当我们慢慢的越来越深入的时候，你就会爱上他，发现他的无穷魅力并为之沉醉！

JAVA语言为什么能够成功

一个产品的成功和一个人的成功是一个道理。除了自身具备过硬的优势外，还需要那么一点点狗屎运，甚至狗屎运非常关键。就是我们所讲的天时地利人和。JAVA的成功除了自己具备跨平台特性外，更重要的是正好踩中了互联网发展的节奏。微软的成功除了windows好用外，也是正好踩中了个人电脑发展的节奏。所以，我们除了自己努力外，也需要出去走走努力才踩一坨狗屎才行！

学技术，也需要知道很多事情和历史对吧，我们来聊聊硅谷。

斯坦福大学依靠自身庞大的校区，创建了斯坦福科技园(硅谷的前身)，初期采用房租免费等方式，鼓励自己的毕业生在科技园创业、鼓励社会企业入驻。由于免费的方式再加上其他优惠的政策，吸引了大量的创业者加入进来。最终大家添柴加火，造就了今天的硅谷。这才是最高明的房地产商啊！说到房地产，我们现在这个社会是不是就很难受了，年薪20万，买一套房子180万，你算算，是不是大概下来就需要十年了，你说你一个大好青年，因为一个房子，就一下变成中年大叔了！哎，人间不值得啊，所以我们需要通过不断的学习来提升自己，从而进步！让自己的人生更有价值，而不是活着而已。

[斯坦福大学百科](#)

看着这么多的牛人，各位，是不是我们也得努努力了，少刷点抖音，少玩点王者，吃鸡（玩得时候带带我~）多跟着我敲敲代码，没准以后咱们的同学也能搞出一个改变世界的东西是吧。

好了，不多说话，我们先来认识一下我们即将要学习的最终要的阶段，JavaSE。

Java三大版本

首先，新人在刚入行的时候，不论是抱着什么目的（当然最后是因为兴趣，因为兴趣是一个很强的推动力），我们要了解学习什么东西，或者说从什么东西学起---JAVA。作为一款高级程序设计语言，它的学习难度上面，至少我认为要比C语言简单许多，再加上如今市场对这方面的人才需求非常大，所以很多人将它作为了首选。目前来说JAVA的应用比较偏向于WEB的设计，也是我们很多学习JAVA的同志，日久就业的一个方向。

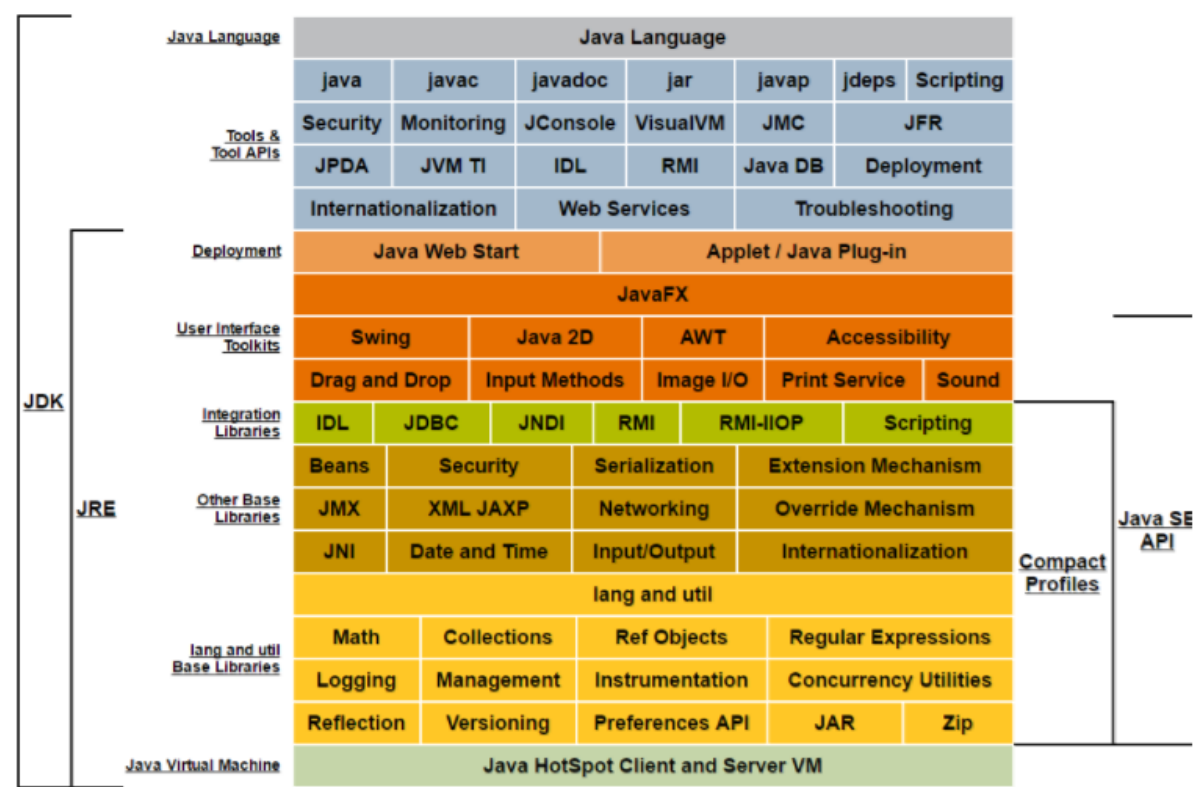
JAVA最大的特点：

Java的主要优势在于其做出的WORA：即一次编写（Write Once）、随处运行（Run Anywhere）。简单来讲，这意味着开发团队能够利用Java编写一款应用程序，并将其编译为可执行形式，而后将其运行在任何支持Java的平台之上。这显然能够极大提高编程工作的实际效率，这种优势来源于Java Virtual Machine(JAVA虚拟机的缩写)，JVM是一种用于计算设备的规范，它是一个虚构出来的计算机，是通过在实际的计算机上仿真模拟各种计算机功能来实现的。JAVA语言的一个非常重要的特点就是与平台的无关性，而使用Java虚拟机是实现这一特点的关键。

JAVA的三大版本：

- 1. JAVA SE：它是JAVA的标准版，是整个JAVA的基础和核心，这是我们主要学习的一个部分，也是JAVAAEE和JAVAME技术的基础，主要用于开发桌面应用程序。学会后可以做一些简单的桌面应用如：扫雷，连连看等。
- 2. JAVA ME：它是JAVA的微缩版，主要应用于嵌入式开发，比如手机程序的开发。目前来说就业范围不是很广，在一些城市可能相对的不好找工作。
- 3. JAVA EE：也叫JAVA的企业版，它提供了企业级应用开发的完整解决方案，比如开发网站，还有企业的一些应用系统，是JAVA技术应用最广泛的领域。主要还是偏向于WEB的开发，而JAVA EE的基础就是JAVA SE，所以我们在学习JAVA SE的时候，基础一定要打好，因为这是最基本的，也是最核心的。

Java SE:

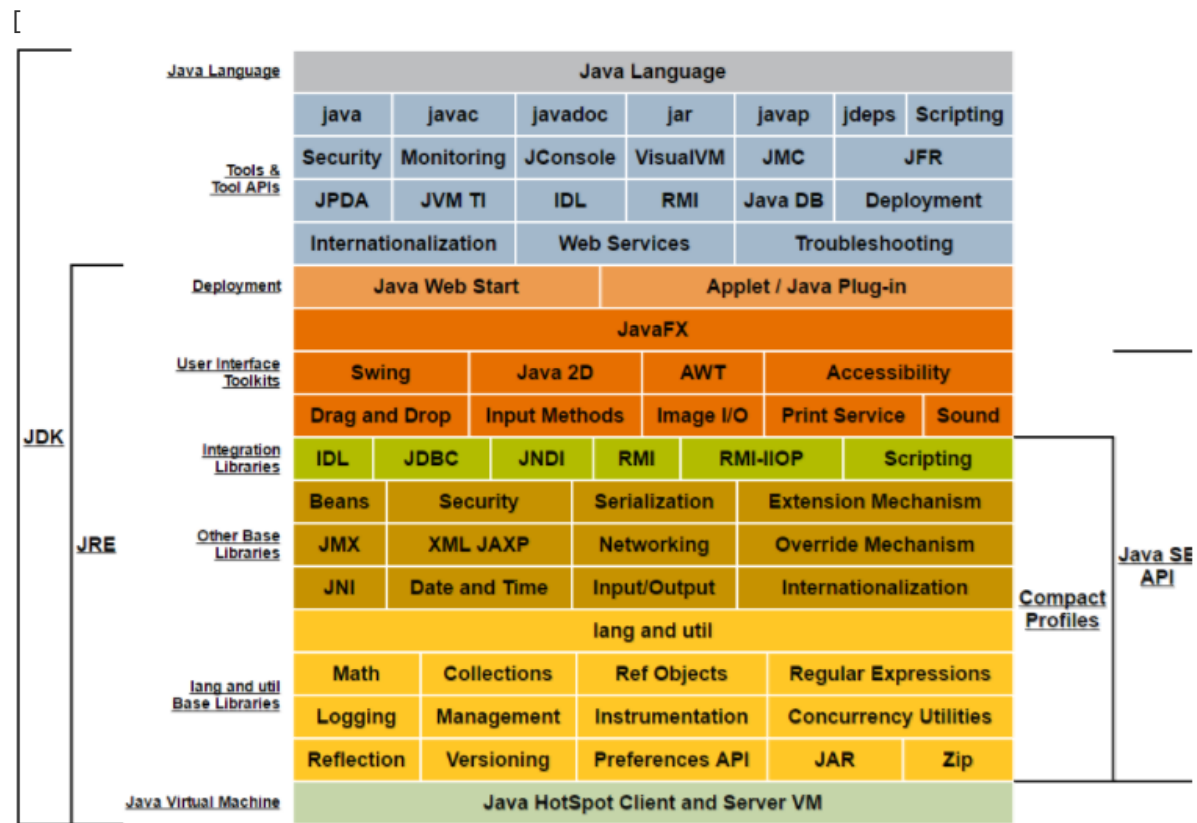


经过这么多的介绍，相信大家已经很明白，如果学习JAVA的目的是为了就业，我们应该先学好JAVA SE，然后继续学习JAVA EE。精髓在于多思考，多练习，不要怕会出错，没有完美的程序，都是不断的在改进，不断的在完善，出错是好事，这样你可以了解更多的异常情况和突发情况，可以为日后工作中出现问题时及时解决积累足够的经验。

最后希望大家在学习程序开发的时候，要有一定勇于探索的精神，搞不懂的问题可以自己先思考（程序开发的核心也是在于不断思考），实在搞不懂可以在网上查询，也可以和大家讨论，但是不要太过于较真，毕竟学海无涯，你永远都学不完所有的东西，要劳逸结合，不要让自己过于疲惫，为了学习更多的东西牺牲健康，那是伟人才会做的事情，得不偿失。

JDK 和 JRE

我们回头看看我们JavaSE的图。



JDK

Java 2 SDK (Development Kit)包含：JRE的超集，包含编译器和调试器等用于程序开发的文件

JRE

Java Runtime Environment (JRE) 包含：Java虚拟机、库函数、运行Java应用程序和Applet所必须文件

Java运行环境的三项主要功能：

- 加载代码：由class loader 完成；
- 校验代码：由bytecode verifier 完成；
- 执行代码：由 runtime interpreter完成。

区别和联系：

sdk（也就是jdk）是jre的超集，是在jre的基础上增加了编译器及其他一些开发工具。

jre就是java运行时环境，包括了jvm和其它一些java核心api,任何一台电脑，只有安装了jre才可以运行java程序。

如果只是想运行JAVA程序，之需要JRE就可以。JRE通常非常小，也包含了JVM。

如果要开发JAVA程序，就需要安装JDK。

初识JVM

(JAVA Virtual Machine)

JVM是一种规范，可以使用软件来实现，也可以使用硬件来实现，就是一个虚拟的用于执行bytecodes字节码的计算机。他也定义了指令集、寄存器集、结构栈、垃圾收集堆、内存区域。

JVM负责将java字节码解释运行，边解释边运行，这样，速度就会受到一定的影响。JAVA提供了另一种解释运行的方法JIT（just in time），可以一次解释完，再运行特定平台上的机器码，高级的JIT可以只能分析热点代码，并将这些代码转成本地机器码，并将结果缓存起来，下次直接从内存中调用，这样就大大提高了执行JAVA代码的效率。这样就实现了跨平台、可移植的功能。

1. JVM是指在一台计算机上由软件或硬件模拟的计算机；它类似一个小巧而高效的CPU。
2. byte-code代码是与平台无关的是虚拟机的机器指令。
3. java字节代码运行的两种方式:

1)方式interpreter(解释)

2)Just-in-time(即时编译):由代码生成器将字节代码转换成本机的机器代码,然后可以以较高速度执行。

JAVA的跨平台实现的核心是不同平台使用不同的虚拟机

不同的操作系统有不同的虚拟机。Java 虚拟机机制屏蔽了底层运行平台的差别，实现了“一次编译，随处运行”。

欲工善其事必先利其器，我们先把我们的环境搭建好，才能开始咱们的征程！

Java开发环境搭建

百度搜索JDK，找到下载地址，浏览Oracle的网站。

JDK下载地址：<https://www.oracle.com/technetwork/java/javase/downloads/jdk8-downloads-2133151.html>

1. 选择版本（我们采用JDK8）
2. 同意协议（流氓协议）
3. 查看电脑的环境（64位，还是32位等等）
4. 下载安装包
5. 安装JDK，JRE（改安装目录，注意规范，作为一个程序人，规范很重要。所以给电脑上环境单独分类最好，）
6. 配置电脑的环境变量：
 1. 我的电脑-->右键-->属性
 2. 环境变量-->添加 JAVA_HOME（JDK的安装地址：D:\Environment\javajdk1.8.0_201）
 3. 配置path：%JAVA_HOME%\bin；%JAVA_HOME%\jre\bin
7. 测试是否安装成功：
 1. 打开cmd（命令行窗口）
 2. 输出 java -version 查看版本信息，是否成功输出！
8. 安装notepad++，或者一个好用的文本编辑器，最好不要用记事本，很不方便！

HelloWorld

环境安装好了，测试代码一定要写HelloWorld！代表你向这个世界的呐喊，仪式感很重要，就像你生活中和家人，朋友，妻子在节日中或者纪念日一定要做一些事情，这就是仪式感，所以各位来跟着我写哈！

1. 我们先随便建立一个文件夹，放我们的代码
2. 新建文件 Hello.java

3. 编写我们的HelloWorld程序!

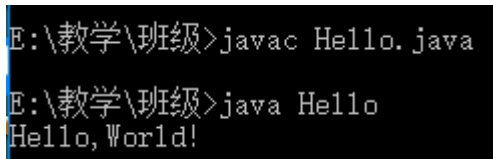
```
1 public class Hello{
2     public static void main(String[] args){
3         System.out.println("Hello,world!");
4     }
5 }
```

4. 保存文件，打开命令行，利用javac编译!

```
1 javac Hello.java
2 # 如果没有报错，查看文件夹下是否有新的一个文件
3 # Hello.class
4 # 如果没有出现，恭喜！说明你遇到了你在学Java当中的第一个Bug
```

5. java 执行!

```
1 java Hello
2 # 成功输出Hello, world!
```



```
E:\教学\班级>javac Hello.java
E:\教学\班级>java Hello
Hello,World!
```

1. 如果出现错误，检查字母大小写是否有错误，或者是否标点符号错误，文件名错误等等，一定要确保成功输出，我们之后再研究，它是怎么输出的!

编写 Java 程序时，应注意以下几点：

- **大小写敏感**：Java 是大小写敏感的，这就意味着标识符 Hello 与 hello 是不同的。
- **类名**：对于所有的类来说，类名的首字母应该大写。如果类名由若干单词组成，那么每个单词的首字母应该大写，例如 **MyFirstJavaClass**。
- **方法名**：所有的方法名都应该以小写字母开头。如果方法名含有若干单词，则后面的每个单词首字母大写。
- **源文件名**：源文件名必须和类名相同。当保存文件的时候，你应该使用类名作为文件名保存（切记 Java 是大小写敏感的），文件名的后缀为 **.java**。（如果文件名和类名不相同则会导致编译错误）。
- **主方法入口**：所有的 Java 程序由 **public static void main(String []args)** 方法开始执行。

JAVA程序运行机制

计算机的高级编程语言类型: 编译型，解释型. Java 语言是两种类型的结合；

【科普：编译型，解释型】

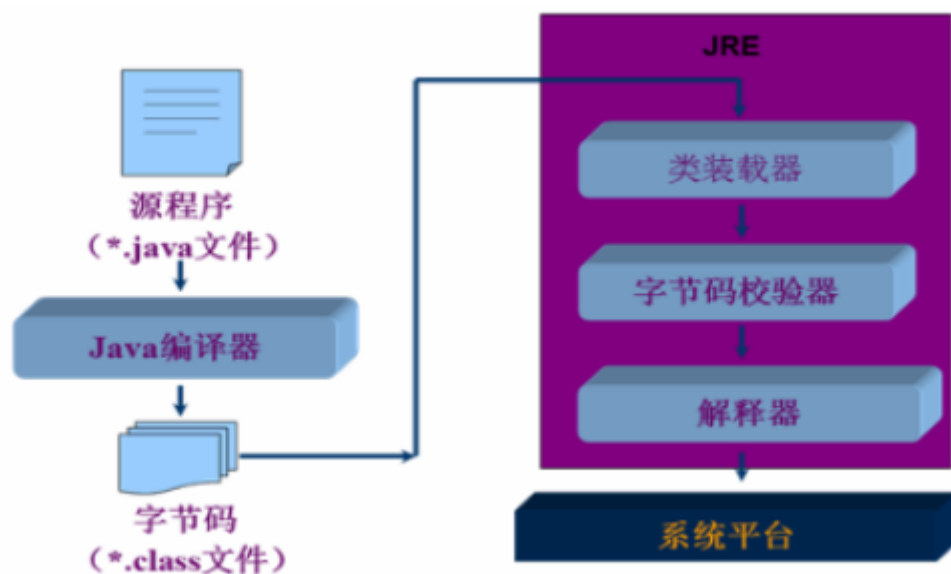
```
1 //从字面上看，“编译”和“解释”的确都有“翻译”的意思，它们的区别则在于翻译的时机安排不大一样。
2
3 //我们先看看编译型：有一个负责翻译的程序来对我们的源代码进行转换，生成相对应的可执行代码。这个过程说得专业一点，就称为编译（Compile），而负责编译的程序自然就称为编译器（Compiler）。就类似于把一本中文书直接翻译成英文版进行出售。
4
5 //现在再看看解释型：打个比方：假如你打算阅读一本外文书，而你不知道这门外语，那么你可以找一名翻译，给他足够的时间让他从头到尾把整本书翻译好，然后把书的母语版交给你阅读（编译型）；或者，你也立刻让这名翻译辅助你阅读，让他一句一句给你翻译，如果你想往回看某个章节，他也得重新给你翻译。（解释型）
6
7 //编译型与解释型，两者各有利弊。前者由于程序执行速度快，同等条件下对系统要求较低，因此像开发操作系统、大型应用程序、数据库系统等时都采用它，像C/C++、Pascal/Object Pascal（Delphi）、VB等基本都可视为编译语言，而一些网页脚本、服务器脚本及辅助开发接口这样的对速度要求不高、对不同系统平台间的兼容性有一定要求的程序则通常使用解释性语言，如Java、JavaScript、VBScript、Perl、Python等等。
8
9 //Java语言虽然比较接近解释型语言的特征，但在执行之前已经预先进行一次预编译，生成的代码是介于机器码和Java源代码之间的中介代码，运行的时候则由JVM（Java的虚拟机平台，可视为解释器）解释执行。它既保留了源代码的高抽象、可移植的特点，又已经完成了对源代码的大部分预编译工作，所以执行起来比“纯解释型”程序要快许多。
10
11 //总之，随着设计技术与硬件的不断发展，编译型与解释型两种方式的界限正在不断变得模糊。
```

第一步：编译

利用编译器（javac）将源程序编译成字节码à 字节码文件名：源文件名.class

第二部：运行

利用虚拟机（解释器，java）解释执行class字节码文件。



总结

- 我们了解了计算机的发展史
- 看到了Java帝国的诞生
- 了解了Java这门语言的特性
- 跟着历史看懂了Java的三大版本
- 安装了Java的开发环境

- 写了自己的第一个程序HelloWorld!
- 知道了编译型和解释型语言的区别

当然，我认为这些都不是最重要的，关键是，今天我们踏出了第一步，我看过这样一个故事：

一个原始人抬头仰望星空，宇宙发出了警告！人类问为什么，宇宙使者说：从他抬头仰望星空开始，整个人类的发展就不过是为了实现他的愿望而已！

所以，只要确定了方向，只要不停下脚步，我们终会抵达终点，实现自己的梦想！

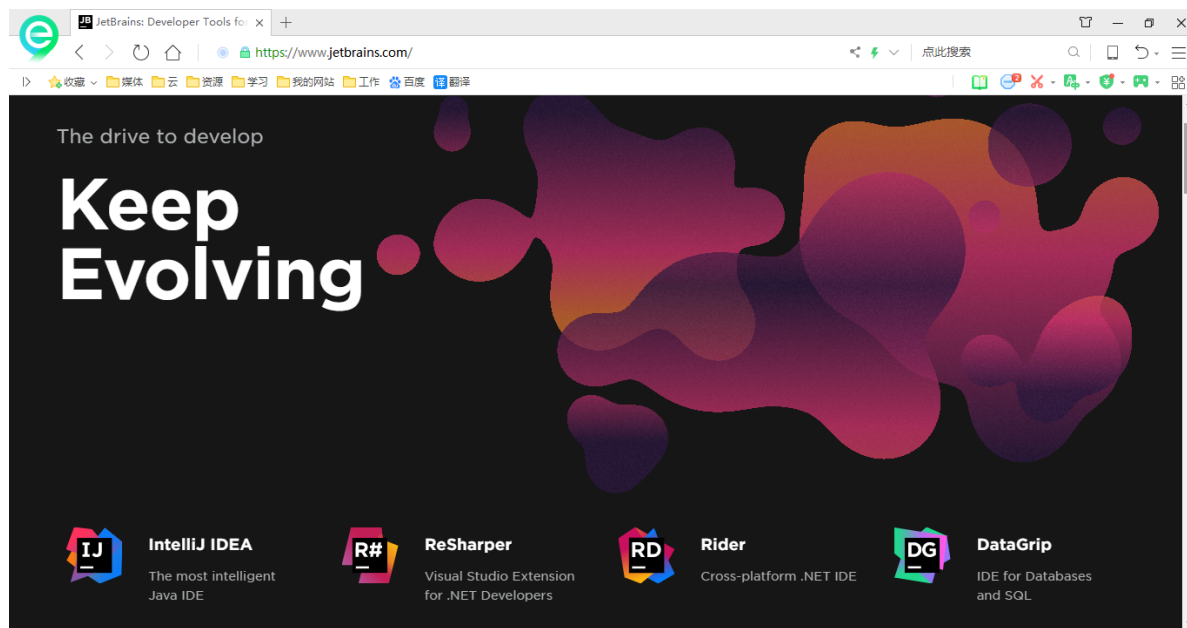
安装IDEA

每次都要利用记事本去编写代码的话，那将是十分麻烦的，工程量大，不易于调试，所以我们迫切需要一款智能的开发工具，于是Eclipse，myEclipse，netbeans，IntelliJ IDEA等智能的IDE诞生了。早时候学习都是利用的Eclipse，不过现在我们学习和工作大部分都利用IDEA了。这也是互联网的残酷性，每个时代都有自己的宠儿。所以我们无论什么时候都需要有危机感，不能忘记学习！

我们来看看IDEA，它是JetBrains公司下的Java集成开发环境，在业界被公认为是最好的Java开发工具之一；JetBrains是捷克的一家软件公司，该公司总部位于捷克共和国的首都布拉格，开发人员以严谨著称的东欧程序员为主，旗下开发了多款软件开发工具；

官方网站：<https://www.jetbrains.com/>

演示：浏览并介绍jetbrains的官网



目前IntelliJ IDEA有免费的社区版（功能相对较少），和收费的旗舰版（功能比较全面）；

我们看自己需要和能力下载，有条件的话，用人家的东西还是可以用正版的。

IDEA破解版下载地址：

- 1 链接：https://pan.baidu.com/s/1_9CDV4jL3BTzi6omT79u1Q
- 2 提取码：xmxq

官方下载地址：<http://www.jetbrains.com/idea/download/#section=windows>

安装步骤：

1. 直接安装完成
2. 下载破解补丁：<http://idea.lanyus.com/>

3. 得到 JetBrainsCrack-2.6.10-release-enc.jar 我们重命名去掉 -release-enc 然后放在idea安装目录的bin文件夹下
4. 在IDEA安装目录bin目录里找到 idea.exe.vmoptions 和 idea64.exe.vmoptions 两个文件打开，在最后一行添加 -javaagent:D:\IDEA\IntelliJ IDEA 2017.2.5\bin\JetBrainsCrack-2.6.10.jar
5. 打开IDEA激活码填写

```
1 ThisCrackLicenseId-{
2   "licenseId":"ThisCrackLicenseId",
3   "licenseeName":"idea",
4   "assigneeName":"kuangshen",
5   "assigneeEmail":"24736743@qq.com",
6   "licenseRestriction":"For This Crack, Only Test! Please support genuine!!!",
7   "checkConcurrentUse":false,
8   "products":[
9     {"code":"II","paidUpTo":"2099-12-31"},
10    {"code":"DM","paidUpTo":"2099-12-31"},
11    {"code":"AC","paidUpTo":"2099-12-31"},
12    {"code":"RS0","paidUpTo":"2099-12-31"},
13    {"code":"WS","paidUpTo":"2099-12-31"},
14    {"code":"DPN","paidUpTo":"2099-12-31"},
15    {"code":"RC","paidUpTo":"2099-12-31"},
16    {"code":"PS","paidUpTo":"2099-12-31"},
17    {"code":"DC","paidUpTo":"2099-12-31"},
18    {"code":"RM","paidUpTo":"2099-12-31"},
19    {"code":"CL","paidUpTo":"2099-12-31"},
20    {"code":"PC","paidUpTo":"2099-12-31"}
21  ],
22   "hash":"2911276/0",
23   "gracePeriodDays":7,
24   "autoProlongated":false
25 }
```

1. 复制汉化文件到lib：（汉化有BUG，不建议大家汉化，顺便也能提高英语水平！）
2. IDEA优化配置参考：<https://jingyan.baidu.com/article/dca1fa6f6af95af1a540527d.html>
 1. 调节字体
 2. 鼠标滚动大小
 3. 注释颜色
 4. 字符编码设置
 5. 常用快捷键
 6. 项目属性浏览
3. 利用IDEA编写HelloWorld！查看控制台输出结果！