**JDK的稳定版本**

Java这个语言，1995 年发展至今，生态方面就不多说了，没有强大的生态，在科技快速发展的今天，是不可能被互联网企业与开发者认同的。

我们都知道，要使用Java语言，那么JDK就必不可少。

而经历了一代一代的变迁，JDK也慢慢更新换代，今年**Oracle**也是直接发布了**JDK18**，而到目前为止，我们的JDK存在了三个版本的**长期支持版**。

分别是**JDK8**、**JDK11**和**JDK17**。

而我们所说的长期支持版是什么呢？长期支持版本我们可以理解成**一个项目会长期维护更新**，也就是说，这三个版本是**Oracle长期维护**的JDK版本。而企业里面，就是需要长期支持版本，因为企业的项目需要稳定的开发环境。

**如何选择**

那JDK版本如此之多，我们改如何选择呢？毋庸置疑，我们肯定选择长期支持版本。

而我们长期支持版本中，JDK8是一个老熟人了，**基本上Java大部分的项目都是基于JDK8环境**。那既然这样，我们就来看看，JDK8的魅力。

**JDK8**

如果你在技术交流群或一群Java程序员中讨论JDK版本选择哪个，我相信，一般来说，他们都脱口而出：“JDK8”。

那么这个版本究竟为何受众面如此之广泛呢？

首先，回顾所有版本，JDK8刚出生时，他就冠名了一个相对完美的开发环境之称。因为相比以往的JDK7、JDK5这些，JDK8的生态很好，开发者使用起来也比之前的版本舒适，同时JDK8支持了几乎所有的开发功能需求，设计比较完善。

所以，JDK8也是迄今为止维护支持时间最长的版本，甚至比JDK17还长。

**JDK11**

科技在慢慢发展，在使用JDK8开发程序时，总存在一些弊端，也多了许许多多的需求，所以慢慢的，**Oracle**也发布了另一个长期支持更新维护版本——**JDK11**。

相比JDK8，用的人肯定没那么多，包括到现在，JDK8仍然是主流。

首先，JDK11对比JDK8，除了有很多内部的升级（比如**开销和时延更低的GC、TLS1.3加持**等等）之外，对于初学使用者来说也有一些语言使用层面的进化。

**变量类型推断**

新版Java引入了一个全新的类型关键字var，用var来定义的变量不用写具体类型，编译器能根据=右边的实际赋值来自动推断出变量的类型。

具体的话，就自己去尝试吧。

**官方HTTP Client加持**

现在JDK官方就自带HTTP Client了，位于java.net.http包下，支持发送同步、异步的HTTP请求，这样一来，以前咱们常用的HTTP请求客户端诸如：OKHttp、HttpClient这种现在都可以退下了！

**总结**

除了这两个明显的变化，其实还有很多内部结构方法的增加。就不多说了。

**实际是，我个人觉得，如果不是有对应需求，那还是老老实实的用JDK8吧。**

因为，说实话，JDK11的生态不太行，参杂在JDK8与JDK17之间。

**JDK17**

JDK17也是长期支持版本，相比11一般来说更具优势，但是，我查阅了一些网络上其他大佬的评价。

如下：

**Java 17的变化真不大**，不及java9的模块化，java11和12并最终在15里全面普及的zgc和shennadoah gc，16里的record类等。各个版本的发布情况和变化详情，我之前有整理过，见下面链接：

* **JDK1.0到17发布版本**： [https://shimowendang.com/docs/C1pC5q2Jff0DhCAz](https://link.zhihu.com/?target=https%3A//shimowendang.com/docs/C1pC5q2Jff0DhCAz)
* **JDK9-17 相关新功能和版本机制**：[https://shimowendang.com/docs/hW6ddKDwt6YPJkDD](https://link.zhihu.com/?target=https%3A//shimowendang.com/docs/hW6ddKDwt6YPJkDD)