如何快速学习新技术?

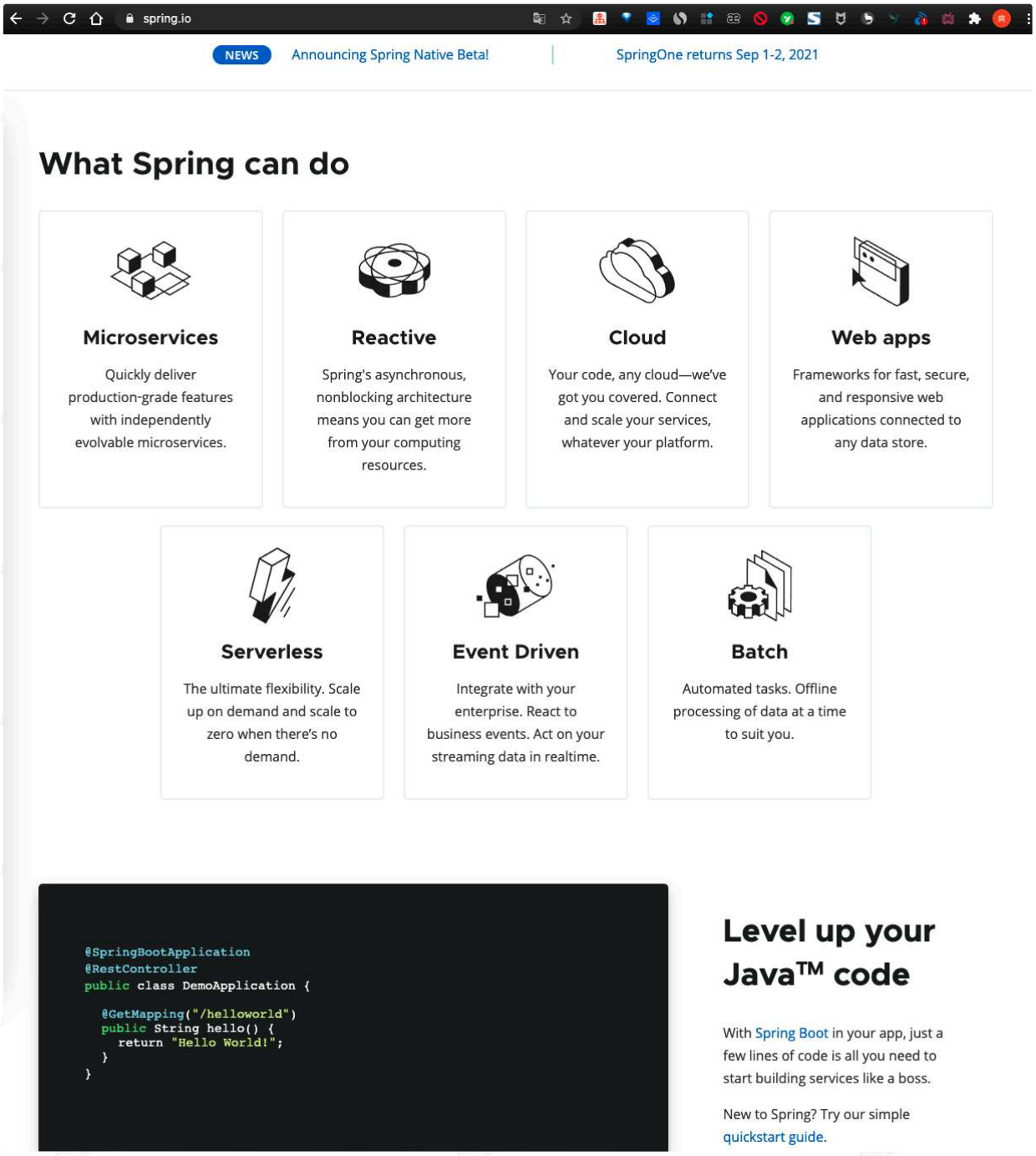
很多时候，我们因为工作原因需要快速学习某项技术，进而在项目中应用。或者说，我们想要去面试的公司要求的某项技术我们之前没有接触过，为了应对面试需要，我们需要快速掌握这项技术。

作为一个人纯自学出生的程序员，这篇文章简单聊聊自己对于如何快速学习某项技术的看法。

文章内容仅代表个人观点，如果你有更好的学习方法，还请在评论区多多和我交流。希望我们都能有所收货！

学习任何一门技术的时候，一定要先搞清楚这个技术是为了解决什么问题的。深入学习这个技术的之前，一定先从全局的角度来了解这个技术，思考一下它是由哪些模块构成的，提供了哪些功能，和同类的技术想必它有什么优势。

比如说我们在学习 Spring 的时候，通过 Spring 官方文档你就可以知道 Spring 最新的技术动态，Spring 包含哪些模块 以及 Spring 可以帮你解决什么问题。



再比如说我在学习消息队列的时候，我会先去了解这个消息队列一般在系统中有什么作用，帮助我们解决了什么问题。消息队列的种类很多，具体学习研究某个消息队列的时候，我会将其和自己已经学习过的消息队列作比较。像我自己在学习 RocketMQ 的时候，就会先将其和自己曾经学习过的第 1 个消息队列 ActiveMQ 进行比较，思考 RocketMQ 相对于 ActiveMQ 有了哪些提升，解决了 ActiveMQ 的哪些痛点，两者有哪些相似的地方，又有哪些不同的地方。

学习一个技术最有效最快的办法就是将这个技术和自己之前学到的技术建立连接，形成一个网络。

然后，我建议你先去看看官方文档的教程，运行一下相关的 Demo ，做一些小项目。

不过，官方文档通常是英文的，通常只有国产项目以及少部分国外的项目提供了中文文档。并且，官方文档介绍的往往也比较粗糙，不太适合初学者作为学习资料。

如果你看不太懂官网的文档，你也可以搜索相关的关键词找一些高质量的博客或者视频来看。 **一定不要一上来就想着要搞懂这个技术的原理。**

就比如说我们在学习 Spring 框架的时候，我建议你在搞懂 Spring 框架所解决的问题之后，不是直接去开始研究 Spring 框架的原理或者源码，而是先实际去体验一下 Spring 框架提供的核心功能 IoC（Inverse of Control:控制反转） 和 AOP(Aspect-Oriented Programming:面向切面编程)，使用 Spring 框架写一些 Demo，甚至是使用 Spring 框架做一些小项目。

一言以蔽之， **在研究这个技术的原理之前，先要搞懂这个技术是怎么使用的。**

这样的循序渐进的学习过程，可以逐渐帮你建立学习的快感，获得即时的成就感，避免直接研究原理性的知识而被劝退。

研究某个技术原理的时候，为了避免内容过于抽象，我们同样可以动手实践。

比如说我们学习 Tomcat 原理的时候，我们发现 Tomcat 的自定义线程池挺有意思，那我们自己也可以手写一个定制版的线程池。再比如我们学习Dubbo 原理的时候，可以自己动手造一个简易版的 RPC 框架。

另外，学习项目中需要用到的技术和面试中需要用到的技术其实还是有一些差别的。

如果你学习某一项技术是为了在实际项目中使用的话，那你的侧重点就是学习这项技术的使用以及最佳实践，了解这项技术在使用过程中可能会遇到的问题。你的最终目标就是这项技术为项目带来了实际的效果，并且，这个效果是正面的。

如果你学习某一项技术仅仅是为了面试的话，那你的侧重点就应该放在这项技术在面试中最常见的一些问题上，也就是我们常说的八股文。

很多人一提到八股文，就是一脸不屑。在我看来，如果你不是死记硬背八股文，而是去所思考这些面试题的本质。那你在准备八股文的过程中，同样也能让你加深对这项技术的了解。

最后，最重要同时也是最难的还是 **知行合一！知行合一！知行合一！** 不论是编程还是其他领域，最重要不是你知道的有多少，而是要尽量做到知行合一。