快速入门Java后端开发的正确姿势

原创 Keegan小钢 Keegan小钢 2019-08-23

这篇文章本是我的收费专栏里的一篇文章，发现有很多人会付费阅读这篇文章，因此决定将其免费分享出来，希望可以将其扩散帮到更多想快速入门Java的开发人员。

---------- **以下是正文**----------

前面三篇文章我们已经聊完了如何快速入门 Android、iOS、Web 前端开发，本篇轮到最后一个端——Java 后端了。Java 后端有着非常庞大的生态圈，也涉及到很多复杂的问题，如分布式事务、分布式查询、微服务、高并发、容灾、容器化等等，涉及到的技术栈和框架就更多了。不过，我们目前只是为了快速入门，我们只会学习最必要的那些知识，能支撑到我们进行实际项目的开发即可。那么，我们需要学习的核心知识主要包括 **Java**、**Servlet**、**JDBC**、**MySQL**、**Redis**、**Spring**、**MyBatis** 等。Java 基础部分在聊入门 Android开发[1]时已经讲过，这里就不重复讲了。

开发工具

先聊下 Java 后端的开发工具吧，我推荐使用 **IntelliJ IDEA**，这是被公认为最好用的 Java 开发工具，熟悉 Android Studio 的 Android 开发人员对它应该不陌生，其强大的功能和特性可以说是完虐 **Eclipse** 和 **MyEclipse**，尤其我最喜欢它的智能代码提示功能。IDEA 用熟了之后绝对大大提升开发效率，目前我们公司开发人员也是全面转向 IDEA 了。这里我向大家推荐一套很详细的中文专题教程，该教程在 github 上是开源的，也经常更新，以下是地址：

- **IntelliJ-IDEA-Tutorial**：https://github.com/judasn/IntelliJ-IDEA-Tutorial

如果英文比较好，建议还是直接阅读官方的帮助文档：

- 官方帮助文档：https://www.jetbrains.com/help/idea/meet-intellij-idea.html

Servlet

Servlet 是 Java Web 的底层技术，虽然现在因为有 **Spring** 框架，我们已经不需要直接编写 Servlet 了，但作为底层技术，我们还是要了解其原理，这样我们才能更好地理解在其之上

的 Spring 框架等。

Servlet 其实就是 **Server Applet** —— 服务器小程序，即运行在服务器上的一个小程序，也是一个基于 Java 技术的 Web 组件。编写一个 Servlet，实际上也是按照 Servlet 规范编写一个 Java 类。只是这个 Java 类是不能独立运行的，它并没有 main 函数，它必须被部署到 **Servlet 容器**中，由容器来实例化和调用 Servlet 的方法。Servlet 容器也称为 **Web 容器**，目前主要就是 **Tomcat** 与 **Jetty**，两者各有优劣，Jetty 更满足公有云的分布式环境的需求，而 Tomcat 更符合企业级环境。我们大部分时候还是处于开发企业级应用阶段，因此，还是优先选择 Tomcat。

目前 Servlet 规范已经演进到 4.0 版本，相比之前的版本，主要是新增了对 **HTTP/2** 的支持。另外，3.0 版本相比之前的 2.5 版本，最重要的则是新增了对 **WebSocket** 的支持，3.0 版本之前的 Servlet 只支持 **HTTP** 请求。那么，对于目前的 Servlet 来说，需要学习哪些知识点呢？这次，我先把核心的知识点罗列出来，后面再来推荐一些相关的学习资源。

- **环境配置**：包括**开发环境**和**部署环境**，要熟悉这两种环境的搭建，开发环境主要就是 JDK 和 IDEA 的安装配置，部署环境则是 Web 容器 **Tomcat** 的安装配置。另外，Java Web 普遍使用 **Maven** 作为构建工具，所以也要了解下 Maven 的用法。虽然 Gradle 比 Maven 更先进，但在 Java 后端大部分项目目前依然还在使用 Maven。
- **生命周期**：要了解 Servlet 的生命周期，并了解 Servlet 容器对 Servlet 实例是如何管理的。
- **API**：Servlet 的常用 API 要熟悉，主要包括 Servlet 接口、ServletConfig、ServletContext、RequestDispatcher、ServletRequest、ServletResponse、HttpServlet、HttpServletRequest、HttpServletResponse，尤其是最后两个类 **HttpServletRequest**、**HttpServletResponse** 必须要非常熟悉，这是在实际开发中会频繁使用到的。
- **过滤器**：Servlet 过滤器可以拦截请求和响应，并进行一些处理。主要用于对用户请求进行鉴权、做日志记录、对数据进行过滤或替换、对数据进行解码或解密等等，多个过滤器可以组成一条过滤器链。
- **监听器**：Servlet 监听器主要用来监听 ServletContext、HttpSession、ServletRequest 对象的生命周期事件，以及这些对象的属性改变事件。
- **WebSocket**：需要注意的是，网上有些继承 WebSocketServlet 的实现方式已经是 Deprecated（过期）的，根据 Oracle 发布的 Java 的 WebSocket 的 JSR356规范[2]所展示的实现方式有两种，一种是使用 @ServerEndpoint[3] 注解，另一种是继承 **javax.websocket.Endpoint** 类，推荐使用注解。关于 WebSocket 的开发入门可以看这篇文章：java WebSocket开发入门WebSocket[4]

关于对 HTTP/2 的支持，目前还没大范围使用，相关资料也是非常少，可以以后再了解。

接着，再来推荐一些学习资源。首推 Head First 的一本书《**Head First Servlets & JSP**》，还是保持一贯通俗易懂的作风，非常适合入门。不过，其中，第 7/8/9 三章关于 JSP 的内

容可以略过，毕竟 JSP 早已经过时不用了。第 14 章讲模式部分则非常值得看，应该说是每个人都必看，当你理解了里面所讲的内容后，才有可能设计出一个更好的系统。

另外，我的 Servlet 入门书籍**《Servlet/JSP深入详解——基于Tomcat的Web开发》**也是本不错的入门书，只是太老旧已经停印了，所以在亚马逊、京东、当当等都买不到了，要看只能到网上找电子版了。比较新的一本书叫**《Servlet、JSP和Spring MVC学习指南》**，2016 年底出版，不过我没看过，不知道怎么样。

菜鸟有一套简明教程，其实也是翻译自国外的一套英文教程，也可以简单看看，有个基本认识。以下是地址：

- **Servlet 教程**：http://www.runoob.com/servlet/servlet-tutorial.html

视频资源方面，我找到一份 Servlet 3.0 特性专题视频，也分享给大家：

- 百度网盘链接：https://pan.baidu.com/s/1qYn2ws0 密码：p85n

JDBC

JDBC = Java DataBase Connectivity（Java 数据库连接），是一套访问数据库的标准 Java API。通过调用这套 API，就可以连接到数据库，可以执行 SQL 语句，对数据库里的数据进行处理。不过，JDBC API 只是一套规范接口，真正与数据库进行连接的则是 JDBC 驱动程序，不同的数据库采用不同的 JDBC 驱动程序。

和 Servlet 一样，因为有更方便的框架，我们也不直接使用 JDBC 的 API 进行数据库编程了。不过，**Hibernate** 和 **MyBatis** 框架的底层还是 JDBC，因此，我们还是要了解 JDBC 的基本原理，才能更快地上手 MyBatis 等框架。好在这部分的内容很少，所以我们只需要投入很少时间就可以掌握了。也找了一份在线的 JDBC 教程分享给大家，以下是地址：

- **JDBC 教程**：http://www.yiibai.com/jdbc/

JDBC 的核心接口和类就只有几个：**Driver**、**DriverManager**、**Connection**、**Statement**、**ResultSet**、**SQLException**。要了解这几个接口和类的用法，以及了解使用 JDBC 的流程，简单来说就是五个步骤：

1. 注册 JDBC 驱动程序；
2. 建立与数据库的连接；
3. 执行 SQL 语句；
4. 获取结果数据；
5. 清理资源。

另外，也要了解下对事务的处理，包括事务的提交和回滚。以及了解如何对 SQL 进行批量处理。所有这些内容在上面的 JDBC 教程中都有讲述，我就不展开说了。另外，我在上面介绍的**《Servlet/JSP深入详解——基于Tomcat的Web开发》**一书中也有一个章节专门讲了 JDBC，也可以看一看。

数据库

我们知道，数据库分为**关系型数据库**和**非关系型数据库**两大类。关系型数据库主要就是 **MySQL**、**Oracle**、**DB2**、**PostgreSQL** 等，标准数据查询语言 **SQL** 就是一种基于关系型数据库的语言。非关系型数据库也称 **NoSQL**（Not only SQL），主要包括 **Redis**、**MongoDB**、**HBase** 等。如今，很多项目都会使用一种以上的数据库，其中，**Redis** 可以说是必选的。另外，**MySQL** 作为最流行的关系型数据库，开源而且入门简单，因此我们就先熟悉这两个数据库。

先介绍下关系型数据库设计工具吧，非关系型数据库好像没什么专门的设计工具就不介绍了。首先，Windows 下无疑 **PowerDesigner** 是最强大也是最流行的，可惜的是它并没有推出 Mac 版本。Mac 下我推荐使用 **Navicat**，既是好用的关系数据库客户端，也是不错的数据库建模工具，建模完成后可以导出为一个 SQL 文件，很方便。

对于 **MySQL**，目前我们也还不需要去研究比较深入的东西，比如存储引擎、存储过程、触发器、事务隔离级别、分布式事务、锁机制等等，最初级的主要了解以下内容：

- **安装配置**：这不用说，最最基本的
- **SQL**：对 SQL 肯定要非常熟悉，这也是最基本的
- **数据类型**：要了解 MySQL 有哪些数据类型，以及不同数据类型之间有哪些差异
- **约束**：了解并熟悉各种约束的用法，包括主键约束、非空约束、唯一性约束、自增约束、默认值约束等
- **索引**：索引很重要，必须要懂，当然，如何用好索引也是门学问
- **函数**：了解一些常用的函数，如 count、sum、avg、max、min、now 等等

这些基础方面的知识就不用推荐什么学习资源了，网上随便找一大把。不过，对于掌握这些基础知识之后的深入学习，我推荐两本书：**《高性能 MySQL》**和**《MySQL 技术内幕：InnoDB 存储引擎》**。另外，官方文档[5]也是必不可少的，只是内容比较多。

对于 **Redis**，因为是基于内存的，所以性能非常高，很多项目用它作缓存。虽然是一个 **key - value** 数据库，但 value 可以是**字符串(String)**、**哈希(Map)**、**列表(List)**、**集合(Sets)**、**有序集合(Sorted Sets)** 等类型。推荐一个菜鸟的简明教程，可以简单入门：

- **Redis教程**：http://www.runoob.com/redis/redis-tutorial.html

书籍方面，推荐看**《Redis实战》**，是一本非常不错且广受好评的入门书籍。

另外，Redis 的 Java 库叫 **Jedis**，也要简单了解下其用法。

Spring

Spring 发展到如今，已经成为一个很庞大的生态圈，我们所了解的 **Spring Framework** 只是整个 Spring 生态圈的一个核心项目，除此之外还有很多其他项目，包括 **Spring Boot**、**Spring Cloud**、**Spring Data**、**Spring For Android** 等 20 多个项目，你可以在 Spring 官网[6]看到所有项目。不过，我们入门只先了解两个项目：**Spring Boot** 和 **Spring Framework**。其中，Spring Framework 是我们要学习的核心框架，但也不是要了解这个项目的全部内容，只先了解一些基础的东西，能够支撑到我们进行入门级的开发就足够了。

那么，先来聊聊 **Spring Framework**，目前最新版本是 Spring Framework 5.0。Spring Framework 包含有20多个模块，包括 **spring-core**、**spring-aop**、**spring-beans**、**spring-context**、**spring-jdbc**、**spring-web**、**spring-webmvc**、**spring-websocket** 等等，具体列表可以查看 Spring Framework 的 Github 项目[7]。不过，一般我们不需要将所有模块都引入到项目中，根据所需去引入即可。

学习 Spring Framework，有两个核心设计思想是要掌握的，就是 **控制反转（Inversion of Control，简称 IoC）** 和 **面向切面编程（Aspect Oriented Programming，简称 AOP）**。说到控制反转，还有一个概念也要理解，叫 **依赖注入（Dependency Injection，简称 DI）**，区别就是，控制反转是一种设计思想，而依赖注入则是其中一种实现的方式，还有另一个实现方式叫**依赖查找（Dependency Lookup）**。不过，大部分都是采用依赖注入，Spring 使用的也是依赖注入的方式。另外，Spring 核心还有一个**控制反转容器（IoC Container）**，主要就是通过配置文件以及利用反射在运行时创建所需要的实现类。要理解这几个概念，推荐看知乎上的一个回答，深入浅出，非常好理解：

- Spring IoC有什么好处呢？ - Sevenvidia的回答[8]

AOP 同样也是一种设计思想，主要是为了分离出一些非业务逻辑的代码，如日志记录、性能统计，安全控制，事务处理，异常处理等。在 Spring 底层是通过动态代理的方式实现了 AOP 的内部机制，另外，上层也支持 AspectJ 的注解方式。AOP 中还有几个相关概念需要理解：**Aspect**、**Joint point**、**Pointcut**、**Advice**、**Weave**、**Introduction**、**Target Object**、**AOP Proxy**。那么，为了帮助大家理解 AOP 这些概念和原理，以及 Spring 中要如何使用 AOP，我找了两篇文章分享给大家学习：

- AOP概念，原理，应用介绍[9]
- Spring中AOP的配置从1.0到5.0的演进[10]

接着，再来了解下 **Spring Boot**，Spring Boot 能够简化基于 Spring 的应用的搭建和开发，Spring Cloud 就是基于 Spring Boot 的。Spring Boot 从根本上来说就是一些库的集合，这些库的 **artifactId** 一般都是以 **spring-boot-starter-** 为前缀。使用 Spring Boot 搭建项目也是非常简单的，我就分享《Spring Cloud微服务实战》一书的作者翟永超博客的两篇文章给大家吧：

- Spring Boot快速入门[11]
- 使用Intellij中的Spring Initializr来快速构建Spring Boot/Cloud工程[12]

要相对系统地入门 Spring，我推荐评价最高的两本书：**《Spring实战(第4版)》**和**《精通Spring 4.x：企业应用开发实战》**。两本书都是基于 Spring 4 的，基于 Spring 5 的书籍还没有，不过这对于入门来说并没什么影响。另外，《精通Spring 4.x：企业应用开发实战》这本书中还加了 Spring Boot 的章节。Spring Framework 的官方文档[13]也可以看，内容毕竟是最新也是最详细的，只是内容太多了，据说有600多页，所以一时半会是看不完的，一般也不推荐初学者直接看官方文档，只建议对某些点理解得还不是很深的情况下可以去翻来看看，加深理解。Spring Boot 的官方文档[14]也是一样。

MyBatis

前面我们知道，Java 访问数据库的标准 API 是 JDBC，只是，直接使用 JDBC API 进行编码操作数据库的话，代码非常繁琐，可扩展性非常低，因此，才会衍生出 **Hibernate**、**MyBatis**、**SpringJDBC** 等对 JDBC 进行了封装的 ORM 框架，以简化开发。那么，为什么选择 MyBatis，而不是 Hibernate 或 SpringJDBC 呢？主要是因为目前最流行的还是 MyBatis，另外 MyBatis 也容易上手。

先来了解下 ORM 这个概念，**ORM = Object Relational Mapping**，称为**对象关系映射**，即业务实体对象和关系数据库之间建立起一种映射关系，最简单的映射关系就是：一个类对应一张数据表，类的每个实例对应表中的一条记录，类的每个属性则对应表的每个字段。Hibernate 是一个标准的 ORM 框架，在实体类和数据库之间建立了完整的映射关系，基本不需要开发人员自己写 SQL，内部会自动生成 SQL 语句，因此，对于数据量非常大的应用，想要优化 SQL 语句就比较困难。MyBatis 则是一个半自动化的 ORM 框架，不同于 Hibernate 提供了从实体类到数据库表的全套映射机制，SQL 是自动生成的；而 MyBatis 则只是在实体类和 SQL 之间建立映射关系，并不自动生成 SQL 语句，因此，开发人员是需要自己写 SQL 的。这样，对 SQL 进行优化则比较方便了。

要了解 MyBatis 的用法，直接看官方文档即可，内容不多，且还有中文版本，以下是链接地址：

- MyBatis 官方文档：http://www.mybatis.org/mybatis-3/zh/index.html

实战入门

前面的内容非常多，要通过实战项目将所有知识点全部串起来还真的比较难，主要是如今已经找不到项目会直接编写 Servlet 和 JDBC 了，而这两块是 Java Web 的底层技术，必须要理解透彻。那么，我们只能拆开来学习，Servlet 和 JDBC 原生部分，建议就按照书籍上的 Demo 去练习，主要目的是要通过编码对那些核心的知识点形成深刻理解。Spring、MyBatis、数据库这些，则可以通过一些开源项目进行学习，我找到一个基于 Spring Boot 的不错的学习项目，推荐给大家：

- springboot-learning-example[15]

里面的每个小项目都非常简单，建议重点看 **springboot-mybatis-redis** 这个小项目，最贴合我们需要学习的内容。因为项目太简单，建议对其进行扩展，比如，自己多增加一个表，并通过增加新的 REST 接口对其进行增删改查；或者添加用户注册登录功能，并使用过滤器对用户作统一鉴权；或者再添加对 WebSocket 的支持。最好是将其扩展成一个相对完整的 Java Web 项目，比如做一个简化的电商项目，不需要界面，只需要提供接口。当你能完成这个完整的 Java Web 项目之后，也算真正入门了。

思考与实践

如果用 Gradle 作为构建工具，要如何配置？与 Android 的配置有何不同？SpringJDBC 要怎么使用？与 MyBatis 比较如何？通过项目实战去感受这些知识点的差异吧。

References

[1] 入门 Android开发: *https://xiaozhuanlan.com/topic/3489260715*
[2] JSR356规范: *http://www.oracle.com/technetwork/articles/java/jsr356-1937161.html*
[3] @ServerEndpoint: *https://xiaozhuanlan.com/u/ServerEndpoint*
[4] java WebSocket开发入门WebSocket: *http://www.jianshu.com/p/d79bf8174196*
[5] 官方文档: *https://dev.mysql.com/doc/refman/5.7/en/*
[6] Spring 官网: *https://spring.io/projects*
[7] Spring Framework 的 Github 项目: *https://github.com/spring-projects/spring-framework*
[8] Spring IoC有什么好处呢？ - Sevenvidia的回答: *https://www.zhihu.com/question/23277575/answer/169698662*
[9] AOP概念，原理，应用介绍: *http://cxis.me/2017/04/12/AOP概念，原理，应用介绍/*
[10] Spring中AOP的配置从1.0到5.0的演进: *http://cxis.me/2017/04/10/Spring中AOP的配置从1.0到5.0的演进/*
[11] Spring Boot快速入门: *http://blog.didispace.com/spring-boot-learning-1/*
[12] 使用Intellij中的Spring Initializr来快速构建Spring Boot/Cloud工程: *http://www.jianshu.com/p/a774fc3b5e37*

[13] Spring Framework 的官方文档: *https://docs.spring.io/spring/docs/5.0.1.RELEASE/spring-framework-reference/index.html*
[14] Spring Boot 的官方文档: *https://docs.spring.io/spring-boot/docs/current/reference/html/*
[15] springboot-learning-example: *https://github.com/JeffLi1993/springboot-learning-example*

阅读原文