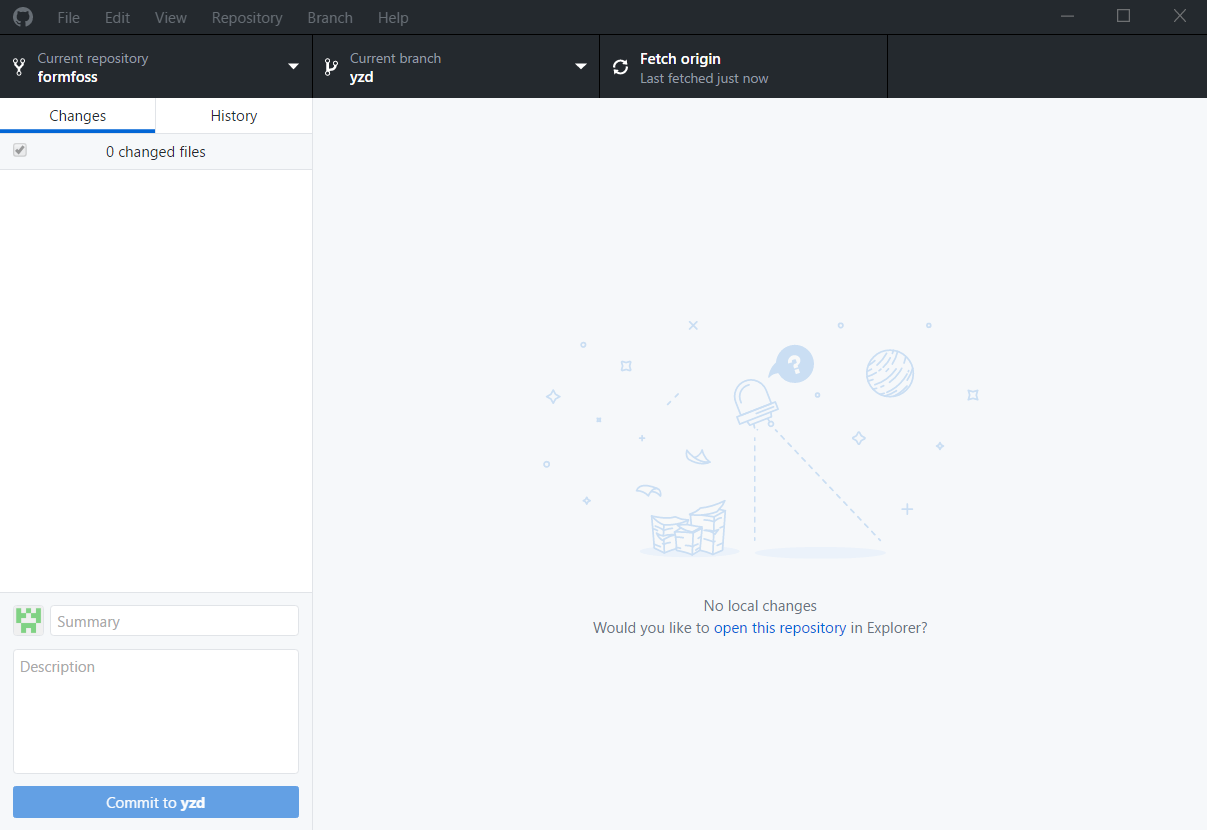
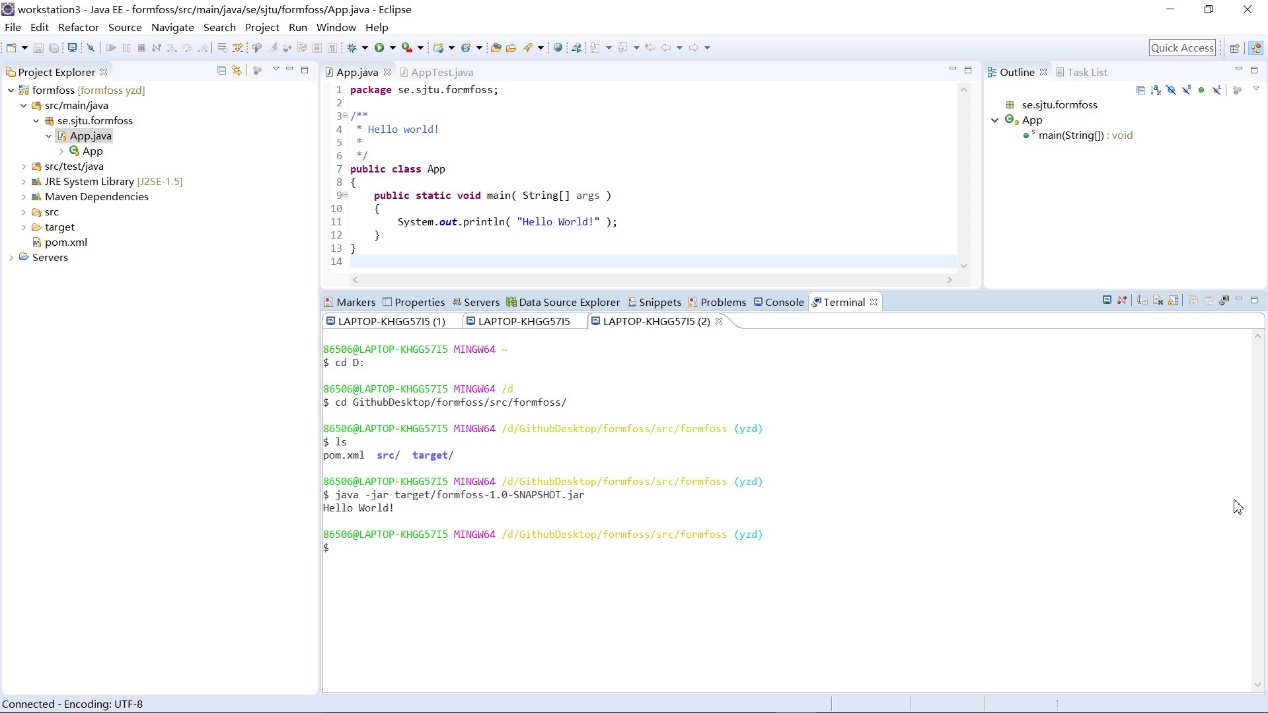
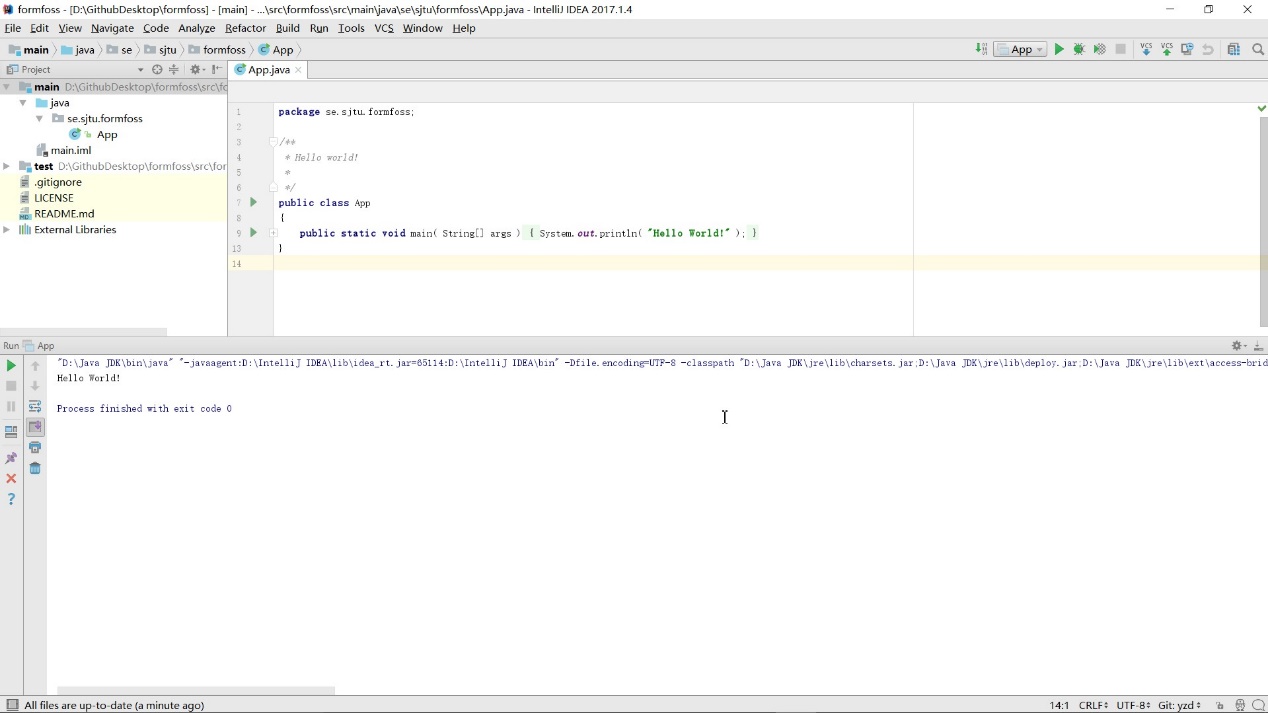
学习笔记

1. 熟练使用GithubDesktop，将小组leader的项目fork到自己的github上之后通过pull request向leader提交新版本。尝试了Git Command 操作后发现使用Git GUI可以方便很多，今后将一直采用GithubDesk进行版本管理。



1. 在Eclipse和Intellij idea中分别尝试着建立了一个基于Spring Boot框架的应用（Hello，world！）。Spring Boot的理念是让项目快速运行起来，使用Spring Boot可以很容易的创建一个独立运行（运行jar，内嵌Servlet容器）、准生产级别的基于Spring框架的项目。使用Spring Boot可以使我不用或者很少进行Spring的配置。





尝试了一下发现启动项目的方法有两种，一种是找到main函数所在的方法后直接Run，另一种方法是将maven打包的jar文件通过java -jar运行。网络上还有一种方法：使用命令 mvn spring-boot:run”在命令行启动该应用。

1. Spring Boot的几个特点：

* 自动配置：针对很多Spring应用程序常见的应用功能，Spring Boot能自动提供相关配置
* 起步依赖：告诉Spring Boot需要什么功能，它就能引入需要的库。
* 命令行界面：这是Spring Boot的可选特性，借此你只需写代码就能完成完整的应用程序，无需传统项目构建。
* Actuator：让你能够深入运行中的Spring Boot应用程序，一探究竟。

1. 简单学习了Angular的一些基础知识

* Angular 是一个用 HTML 和 JavaScript 或者一个可以编译成 JavaScript 的语言（例如 Dart 或者 TypeScript ），来构建客户端应用的框架。用 Angular 扩展语法编写 HTML 模板， 用组件类管理这些模板，用服务添加应用逻辑， 用模块打包发布组件与服务。然后，我们通过引导根模块来启动该应用。 Angular 在浏览器中接管、展现应用的内容，并根据我们提供的操作指令响应用户的交互。
* Angular应用中8个主要的构造块



* 模块 (module)：
* 每个 Angular 应用至少有一个模块（根模块），习惯上命名为AppModule。
* 根模块在一些小型应用中可能是唯一的模块，大多数应用会有很多特性模块，每个模块都是一个内聚的代码块专注于某个应用领域、工作流或紧密相关的功能。
* Angular 模块（无论是根模块还是特性模块）都是一个带有@NgModule装饰器的类。
* NgModule是一个装饰器函数，它接收一个用来描述模块属性的元数据对象
* Angular 提供了一组 JavaScript 模块。可以把它们看做库模块。每个 Angular 库的名字都带有@angular前缀。
* 组件 (component)：
* 组件负责控制屏幕上的一小块区域，我们称之为视图。
* 我们在类中定义组件的应用逻辑，为视图提供支持。 组件通过一些由属性和方法组成的 API 与视图交互。
* 模板 (template)：
* 我们通过组件的自带的模板来定义组件视图。模板以 HTML 形式存在，告诉 Angular 如何渲染组件。
* 元数据 (metadata)：
* 元数据告诉 Angular 如何处理一个类。
* 数据绑定 (data binding)：
* Angular 支持数据绑定，一种让模板的各部分与组件的各部分相互合作的机制。 我们往模板 HTML 中添加绑定标记，来告诉 Angular 如何把二者联系起来。
* 指令 (directive)：
* Angular 模板是动态的。当 Angular 渲染它们时，它会根据指令提供的操作对 DOM 进行转换；
* 组件是一个带模板的指令。
* 服务 (service)
* 依赖注入 (dependency injection)：
* “依赖注入”是提供类的新实例的一种方式，还负责处理好类所需的全部依赖。大多数依赖都是服务。 Angular 使用依赖注入来提供新组件以及组件所需的服务。
* Angular 通过查看构造函数的参数类型得知组件需要哪些服务。