Spring

* Java Framework for making Enterprise Level applications.
* Framework vs Library
  + Framework
    - You write your code to fit into the application.
    - The application calls your code.
    - You HAVE TO code the way Spring wants you to.
      * You are building a Spring application.
  + Library
    - You use someone else’s code in your application.
  + Usually web applications but not strictly.
  + Came out in 2009\*\*\*
    - I did not check
  + Before Spring there were EJB Entity Java Beans.
    - Were terrible.
    - Very buggy and hard to work with.
    - It was proprietary software that you had to pay Oracle for.
  + Spring killed the old enterprise software of Java.
    - It was free. (Open Source)
    - It was better.
* The heart of spring is the IoC (Inversion of Control) container
  + (Literally called Spring Core)
  + It’s main purpose is to control the lifecycle and management of Spring Beans.
    - Lifecycle
      * How many instances of this bean exist in the application?
      * When do they get created.
    - Management
      * Fulfilling of dependencies in beans.
      * Dependency Injection
  + Spring Beans are objects that are managed by the IoC container
  + In newer version of Spring the IoC container is called application context.
    - Previously the Bean Factory
* Spring is Modular by design.
  + Spring CORE
    - IoC Container
  + Spring WEB
    - Creating controllers
  + Spring DATA
    - Creating a persistence layer (DAOS)
  + Spring AOP
    - Aspect Oriented Programming
  + Spring Actuator
    - Metrics for your application
  + Spring Security
    - Everything Security related