Spring Stereotype Annotations

For Spring 4.3.x

Component Scan

- One of the ways we can configure beans is by using annotations. The annotations we use to declare a bean are called Stereotype Annotations
- In order to use annotations, we have to allow component scanning in Spring. This means Spring will look through our Java files for annotated classes, and then build beans appropriately based on those annotations.
- XML is fairly common for Spring Framework, but many other Spring projects have fully embraced annotations

@Component

- Basic stereotype annotation
- Lets the ApplicationContext know to build this class as a bean
- Generic; used in the definition of the other annotations

@Controller

- Stereotype for presentation layer, used with spring-mvc
- Only in @Controller will spring make use of @RequestMapping annotations
    - Stay tuned for @RequestMapping… :)

@Service

- Stereotype for service layer
- Not functionally different from @Component
- Services are typically stateless singletons that implement some logic on other objects.

@Repository

- Stereotype for persistence layer
- Used for repositories and DAOs
- In @Repository, Spring will replace specific data access exceptions with its own general unchecked exceptions
    - This functionality makes use of the PersistenceExceptionTranslationPostProcessor bean, that you have to define in your XML or elsewhere.

Annotations for Dependency Injection, Spring

- These requires <context: component-scan /> or similar
  - <mvc: annotation-driven />
- @Autowired
  - Automatically finds the appropriate dependency using either the bean name or the type
  - Used with fields (boo!), constructors (good!), and setters (meh?)
- @Primary
  - Makes this bean the first choice when there is ambiguity in autowiring / DI
- @Lazy
  - Makes this bean lazily loaded

Annotations for Dependency Injection, JSR 330

- These annotation CAN AND SHOULD be used in Spring
- In addition, they come from javax, so they can be used outside of Spring
- Similar to @Entity from ORM, which can be used in multiple frameworks

- @Inject : javax dependency injection, similar to @Autowired
- @Named : specify a name to specify which object to inject
- @Qualifier : more general specification, similar to @Named
- @Scope : set scope of bean by giving appropriate argument