Annotations:

- **@Component**: indicates that a class has to be managed by Spring Framework
- **@Autowired**: indicates that a class is dependent on another class
- **@SpringBootApplication**: simplifies the initialization of our Spring-Based Applications

Terms:

- **Dependency**: something a class needs to be able to perform its functionality
  - In our example, GamingConsole implementation (i.e. MarioGame) is a dependency of GameRunner
- **Component Scan**: how Spring Framework finds component classes
  - This is defined as a part of @SpringBootApplication
  - As soon as we launch the SpringContext in the main class, a component scan is done
  - By default, a Component Scan is done on the same package where @SpringBootApplication application annotation present
- **Spring Bean**: an object managed by Spring Framework
- **Dependency Injection**: Identify beans, their dependencies, and wire them together
  - Provides *Inversion of Control* (*IOC*)
- **IOC**: moves the responsibility of creating objects and wiring them together from the programmer to the Spring Framework
- **IOC Container**: Manages the lifecycle of the beans and the dependencies
  - The technical component inside Spring Framework
    - *ApplicaitonContext* (complex, more frequently used)
    - *BeanFactory* (simpler features, rarely used)

- **Autowiring**: The process of wiring (injecting) in the dependencies for a Spring Bean