The Spring Framework application development is becoming a very popular approach for developing applications for the Java EE platform. There are several benefits to using the Spring Framework. For this assignment you are to take a light overview of the Spring Framework.

Select **one** of the topics below. Then, answer the “what,” “how,” and “why” questions below about your selected topic. Provide a simple code or code snippet example to further illustrate your thoughts.

* Dependency Injection (DI)
* Aspect Orientated Programming (AOP)

**“What,” “How,” and “Why” Questions**

* **What:** What are you writing about? Give the audience a brief overview of the topic by providing them with foundational information (history, background information, etc.).
* **How:** How is the information relevant? Apply personal knowledge (this can be through research or actual practiced knowledge) to build trust with the audience.
* **Why:** Justify your position and/or course of action. The audience needs proof the information you are presenting is creditable and actionable.

Spring Framework is becoming more common. Within the Spring Framework, there are key features such as Dependency Injection, Aspect-Oriented Programming, Transaction Management, Spring MVC, Spring Security, Spring Data, Spring Batch, and integration with other frameworks (GeeksforGeeks, 2025a). This discussion post will focus on dependency injection (DI).

Spring Dependency Injection is considered the main functionality of Spring IOC (GeeksforGeeks, 2025b). Constructor and setter methods are used to inject dependencies where Java classes are held independently and follow the exact requirements of a JavaBean (GeeksforGeeks, 2019b). Dependency Injection enhances the reusability of code and makes testing it easier by resolving dependent classes (GeeksforGeeks, 2019b). There are two types of Dependency Injection: "Setter Dependency Injection (SDI)" and "Constructor Dependency Injection (CDI)" (GeeksforGeeks, 2019b). SDIs inject dependencies through setter methods while CDIs inject dependencies through constructors (GeeksforGeeks, 2019b). Field injection may increase code readability, but using setter and constructor injections is the preferred way to handle this (GeeksforGeeks, 2019b).

This information is particularly relevant to what we have been learning and working with thus far. There will be times when classes that we have created need to work together, so understanding how dependency injection can be applied makes the process easier.

This information all comes from a reputable source, the Geeks for Geeks website. This information can be directly applied to our ongoing projects to improve how they perform. It also directly relates to the resources provided by our professor for this module.

Here is a code example of Spring Dependency Injection (GeeksforGeeks, 2019b):

package com.geeksforgeeks.org;  
  
import com.geeksforgeeks.org.IGeek;  
import org.springframework.beans.factory.annotation.Autowired;  
  
public class GFG {  
  
 // The object of the interface IGeek  
 private IGeek geek;  
  
 // Setter method for property geek with @Autowired annotation  
 @Autowired  
 public void setGeek(IGeek geek) {  
 this.geek = geek;  
 }  
}

<beans   
xmlns="http://www.springframework.org/schema/beans"  
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"  
xsi:schemaLocation="http://www.springframework.org/schema/beans  
http://www.springframework.org/schema/beans/spring-beans-2.5.xsd">  
  
 <bean id="GFG" class="com.geeksforgeeks.org.GFG">  
 <property name="geek">  
 <ref bean="CsvGFG" />  
 </property>  
 </bean>  
   
<bean id="CsvGFG" class="com.geeksforgeeks.org.impl.CsvGFG" />  
<bean id="JsonGFG" class="com.geeksforgeeks.org.impl.JsonGFG" />  
   
</beans>

**References**

GeeksforGeeks. (2025a, January 16). *Introduction to Spring Framework*. GeeksforGeeks. https://www.geeksforgeeks.org/introduction-to-spring-framework/

GeeksforGeeks. (2025b, April 23). *Spring Dependency Injection with Example*. GeeksforGeeks. https://www.geeksforgeeks.org/advance-java/spring-dependency-injection-with-example/

**Assignment Requirements and Grading:**

* An initial post of approximately 250 words is due by **Thursday, 11:59 p.m., CST**.
* Submit your post by clicking on the assignment link above, then Create Thread. You must create a thread in order to view your peers' posts. Tip: Create your post in a Word document and then copy and paste your work into the thread.
* A minimum of three (3) responses, to the original threads of other students, of 100-200 words each are due by **Sunday, 11:59 p.m., CST**.
* This discussion board is worth **25 Points**.
* To view the rubric grading criteria, click on the following link: [Discussion Board Grading Rubric.](https://content.bellevue.edu/cst/csd/rubricdbv3.pdf)

Hey, Lea! I really enjoyed reading your post for this week. You did a nice job explaining what dependency injection is, how it is relevant, and why we should use this course of action. I do not believe I have read or referenced the Built-in site you used as a resource before. The information provided was insightful, and after looking over the website, I realized it can be convenient in the future since it can match you with tech jobs! It is great to know that there are resources dedicated to finding tech jobs, and according to their site, 105,571 tech companies.

Hi there, Samir! You did a great job of explaining dependency injection and providing a code example of how it works. I agree that DI is a main functionality of the Spring Framework. It is a very loaded framework, so focusing on a feature at a time helps fully understand it and how it fits with the framework as a whole. DI really is great at improving code and making it easier for developers. It is important to note that there are two types of DI: Setter and Constructor. Along with DI, the Spring Framework also involves Aspect-Oriented Programming, Transaction Management, Spring MVC, Spring Security, Spring Data, and Spring Batch.

Hey, Megan! I think you did a comprehensive job of covering aspect-oriented programming. The included code example adds to your explanation and shows how to implement aspect-oriented programming. AOP seems to be very handy when developing code. OOP differs from AOP since the main unit is the class, while AOP focuses on the aspect. Like Dependency Injection, AOP is a core component of the Spring Framework. However, it is not a required part of the process, but it goes well together. I completely agree that the relevance of AOP is vital. By implementing AOP, it helps fight against conflicting code deployments