Certainly! **Spring AOP (Aspect-Oriented Programming)** is a programming paradigm that enhances modularity by allowing the separation of cross-cutting concerns. It achieves this by adding additional behavior to existing code without directly modifying the code itself. [Instead, new behaviors are declared separately from the original code1](https://www.baeldung.com/spring-aop).

Here are **five free reference links** where you can learn more about Spring AOP:

1. **Baeldung**[: This tutorial provides an introduction to Spring AOP, covers pointcut expressions, and explains how to implement custom Spring AOP annotationsRead more1](https://www.baeldung.com/spring-aop).
2. **Spring Framework Documentation**[: The official Spring Framework documentation offers detailed information on AOP concepts, terminology, and usage](https://www.baeldung.com/spring-aop)[Explore here2](https://docs.spring.io/spring-framework/reference/core/aop/introduction-defn.html).
3. **Java Guides**[: This tutorial walks you through Spring AOP with annotations, including practical examples](https://www.baeldung.com/spring-aop)[Learn from Java Guides](https://www.javaguides.net/2021/08/spring-aop-tutorial-with-annotations.html)[3](https://jstobigdata.com/spring/complete-spring-aop-tutorial/).
4. **Javatpoint**[: Javatpoint’s tutorial covers Spring AOP implementation, aspects, and cross-cutting concerns](https://www.baeldung.com/spring-aop)[Check it out4](https://www.javatpoint.com/spring-aop-tutorial).
5. **Spring Framework Reference**[: Dive deeper into Spring AOP with AspectJ pointcuts and explore how to write custom aspects](https://www.baeldung.com/spring-aop)[Read the official documentation5](https://docs.spring.io/spring-framework/reference/core/aop.html).

Feel free to explore these resources to enhance your understanding of Spring AOP! 🌱