Using @Primary and @Qualifier Annotations

Spring Feature Description

The `@Primary` annotation indicates that a particular bean should be given preference when multiple candidates are qualified to autowire a single-valued dependency. The `@Qualifier` annotation allows you to specify which bean to inject when there are multiple candidates.

Why Do We Need It in Flight Application?

In our application, we may have multiple implementations of a service or component (e.g., different data loaders for various formats). By using `@Primary` and `@Qualifier`, we can control which bean gets injected without altering our application logic.

Refactoring Application

We will introduce two different data loaders, one for CSV files and another for database interactions, and use these annotations to manage them.

Update The Code

- **Introduce DataLoader Interface**

Define a common interface for all data loaders to standardize behavior across different implementations.

- **Define Multiple Implementations**

Create separate classes for each data loader, ensuring they implement the DataLoader interface. Use `@Component` to register them as beans.

- **Inject Data Loader**

Update the main application class to use the DataLoader interface when retrieving the data loader bean.

This way, when the application starts, it will use the DataLoader interface to load data from either CSV or a database, depending on the implementation injected.

- **Use @Primary to Set Default Loader**

Annotate one of the data loader classes with `@Primary` to designate it as the default when no `@Qualifier` is specified.

- **Specify the @Qualifier**

Annotate each data loader class with `@Qualifier` to give each a distinct name, enabling you to specify which loader to use when injecting.

- **Inject with @Qualifier**

Use `@Qualifier` in your service or component to inject the specific data loader you want. This provides clarity about which implementation is being used.

Benefits of Using @Primary and @Qualifier

- **Fine-Grained Control**: Specify exactly which bean to inject when there are multiple candidates.
- **Maintainability**: Easily swap implementations without changing the code that uses them.

Further Advices

Use `@Primary` sparingly to avoid confusion. Whenever possible, opt for `@Qualifier` to make dependencies explicit and understandable.