@Value

Yes, the `@Value` annotation can work even if the class is not a Spring bean. You can use `@Value` to inject values into fields of any class, whether it's a Spring-managed bean or not. However, when using `@Value` in a non-bean class, you need to ensure that the instance of that class is managed or instantiated by Spring, otherwise the `@Value` annotation will not be processed.

For example, if you have a regular Java class and you want to inject a value into one of its fields using `@Value`, you can do so by instantiating the class within a Spring-managed bean or component, and then the `@Value` annotation will be processed by Spring.

Here's an example:

```java
public class MyClass {

    @Value("${my.property}")
    private String myProperty;

    public String getMyProperty() {
        return myProperty;
    }
}

@Configuration
public class AppConfig {

    @Bean
    public MyClass myClass() {
        return new MyClass();
    }
}
```

In this example, `MyClass` is a regular Java class, not annotated as a Spring component or bean. However, it has a field `myProperty` annotated with `@Value("${my.property}")`. When you instantiate `MyClass` as a bean in the `AppConfig` class, Spring will process the `@Value` annotation and inject the value of `${my.property}` into the `myProperty` field.