@DynamicPropertySource

The `@DynamicPropertySource` annotation is used in JUnit 5 tests in Spring to dynamically add properties to the Spring `Environment` during test execution. This annotation can be particularly useful when you need to mock or stub external services or configurations for your tests.

Here's how you can use `@DynamicPropertySource`:

1. **Import Necessary Dependencies**:
   Make sure you have the necessary dependencies for JUnit 5 and Spring Test in your project.

2. **Write Test with @DynamicPropertySource**:
   In your test class, annotate a method with `@DynamicPropertySource` to dynamically add properties to the Spring `Environment`. This method should have a `DynamicPropertyRegistry` parameter to register the properties. Inside the method, you can use the `DynamicPropertyRegistry` to add properties.

   ```java
   import org.junit.jupiter.api.Test;
   import org.junit.jupiter.api.extension.ExtendWith;
   import org.springframework.boot.test.context.SpringBootTest;
   import org.springframework.test.context.DynamicPropertyRegistry;
   import org.springframework.test.context.DynamicPropertySource;
   import org.springframework.test.context.junit.jupiter.SpringExtension;

   @ExtendWith(SpringExtension.class)
   @SpringBootTest
   public class MyTest {

       @DynamicPropertySource
       static void properties(DynamicPropertyRegistry registry) {
           registry.add("my.property", () -> "exampleValue");
       }

       @Test
       void testSomething() {
           // Your test logic here
       }
   }
   ```

   In this example, the `properties()` method dynamically adds the property `my.property` with the value `"exampleValue"` to the Spring `Environment`.

3. **Write Tests Using the Properties**:

Write your tests, making use of the properties added dynamically to the Spring `Environment` in the `properties()` method.

The properties added using `@DynamicPropertySource` will be available in the Spring `Environment` for the duration of the test execution. This can be helpful for configuring your application or mocking external dependencies in your tests.

Remember that `@DynamicPropertySource` is a feature of Spring's testing framework and is intended for use in integration tests. It allows you to customize the environment specifically for your tests without affecting the production environment.