Algolab

Das kleine Unit-Test-Dilemma

Michael Wiedeking

Java Forum Stuttgart – 4. Juli 2019

```
public class ResourceTest {

    @Test
    public void testWithResource() {

        try (Resource resource = new Resource()) {

            checkIt(resource);

        }

    }

}
```

```
private Resource resource;


@Test
public void testWithResource() {
    checkIt(resource);
}
```

```
@BeforeEach
public void openResource() {
    resource = new Resource();
}


@AfterEach
public void closeResource() {
    resource.close();
}
```

```
test ResourceTest is

    test "test with resource" is

        with resource :← Resource() do

            checkIt(resource)

        end

    end

end
```

resource : Resource?

test “test with resource” **is**

    if *resource* ≠ NIL **then**

        checkIt(*resource*) ¶ Vorsicht!

    end

end

method forEach(*test* : () → TestStatus) → () **is**

    resource ← Resource()

    test()

    resource?.close() ¶ Vorsicht!

end

resource : Resource?

test “test with resource” **is**

    assume *resource* ≠ NIL

    checkIt(*resource*)

end

method forEach(*test* : () → TestStatus) → () **is**

    with *r* :← Resource() **do**

        resource ← *r*

        test()

    end

end

```
test "test with resource" (r : Resource) is
    checkIt(r)
end
```

```
test "test with resource" (r : Resource) is
    checkIt(r)
end
```

test : (Resource) → TestStatus

```
test "test with resource" (r : Resource) is
    checkIt(r)
end
```

```
method forEach(test : (Resource) → TestStatus) → () is
    with resource :← Resource() do
        test(resource)
    end
end
```

```
test "test with resource" (r : Resource) is
    checkIt(r)
end
```

```
method forGroup(group : (Resource) → TestStatus) → () is
    with resource :← Resource() do
        group(resource)
    end
end
```

```
test "two resources" (r : Resource, f : File) is
    checkIt(r, f)
end
```

test “two resources” (r : Resource, f : File) **is**

checkIt(r, f)

end

test : (Resource, File) → TestStatus

```
test “two resources” (r : Resource, f : File) is
    checkIt(r, f)
end
```

```
        method forEach(
            test : (Resource, File) → TestStatus
        ) → () is
            with resource : Resource(); file :← File() do
                test(resource, file)
            end
        end
```

```
test "two resources" (r : Resource, f : File) is
    checkIt(r, f)
end
```

```
        method forEach(
            resource : Resource,
            test : (Resource, File) → TestStatus
        ) → () is
            with file :← File() do
                test(resource, file)
            end
        end
```

```
test "two resources" (r : Resource, f : File) is
    checkIt(r, f)
end
```

```
method forEach(
    resource : Resource,
    test : (Resource, File) → TestStatus
) → () is
    with file :← File() do
        test(resource, file)
    end
end
```

```
method forGroup(
    group : (Resource) → TestStatus
) → () is
    with resource :← Resource() do
        group(resource)
    end
end
```

```
test "two resources" (r : Resource, f : File) is
    checkIt(r, f)
end
```

```
test "other resources" (r : Resource, s : String) is
    checkIt(r, s)
end
```

```
method forEach(
    resource : Resource,
    test : (Resource, File) → TestStatus
) → () is
    with file :← File() do
        test(resource, file)
    end
end
```

```
test "two resources" (r : Resource, f : File) is
    checkIt(r, f)
end

    method forEach(
        resource : Resource,
        test : (Resource, File) → TestStatus
    ) → () is
        with file :← File() do
            test(resource, file)
        end
    end
```

```
test "other resources" (r : Resource, s : String) is
    checkIt(r, s)
end

    method forEach(
        resource : Resource,
        test : (Resource, String) → TestStatus
    ) → () is
        with string :← String() do
            test(resource, string)
        end
    end
```

```java
public class JavaTest {
    @Test
    public void testWithResource(Resource r) {
        checkIt(r);
    }
}
```

```java
public void forEach(Testable test) {
    try (Resource resource = new Resource()) {
        test.execute(resource);
    }
}
```

```java
public abstract class JavaTest {
    @Test
    public void testWithResource(Resource r) {
        checkIt(r);
    }

    public abstract TestResult execute(
        Resource resource
    );
}
```

```java
public void forEach(JavaTest test) {
    try (Resource resource = new Resource()) {
        test.execute(resource);
    }
}
```

Vielen Dank!

Mehr über Aalgola unter www.aalgolab.de