Awaitility

- Stop sleeping, start awaiting!

Johan Haleby

JAYWAY

First try

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));


  assertThat(users.statusOf(userId), is(REGISTERED));
}
```

Second try

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  Thread.sleep(200);
  assertThat(users.statusOf(userId), is(REGISTERED));
}
```

What about Jenkins?

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  Thread.sleep(200);
  assertThat(users.statusOf(userId), is(REGISTERED));
}
```

JAYWAY

Third try

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  Thread.sleep(500);
  assertThat(users.statusOf(userId), is(REGISTERED));
}
```

Fourth try

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  Thread.sleep(2000);
  assertThat(users.statusFor(userId), is(REGISTERED));
}
```

Using Awaitility

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  await().atMost(2, SECONDS).until(userIsRegistered());
}
```

JAYWAY

Callable<Boolean>

```
private Callable<Boolean> userIsRegistered() throws Exception {
        return new Callable<Boolean>() {
            public Boolean call() throws Exception {
                return users.statusOf(userId) == REGISTERED;
            }
         }
}
```

Better re-use

```
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  await().atMost(2,SECONDS).until(userStatus(), is(REGISTERED));
}
```

Supplier

Matcher

Callable<StatusType>

```
private Callable<StatusType> userStatus() throws Exception{
        return new Callable<StatusType>() {
            public StatusType call() throws Exception {
                return users.statusOf(userId);
            }
        }
}
```

JAYWAY

Better re-use

```
@Test
public void updatesUserStatus() throws Exception {
 cmdPublisher.publish(new UpdateStatusCommand(userId,UNREGISTERED));

 await().atMost(2,SECONDS).until(userStatus(), is(UNREGISTERED));
}
```

Same supplier

Different Matcher value

Reduce verboseness

```java
@Test
public void updatesUserStatus() throws Exception {
  cmdPublisher.publish(new UpdateStatusCommand(userId, REGISTERED));

  await().untilCall(to(users).statusOf(userId), is(REGISTERED));
}
```

Proxy supplier

Field supplier

```
await().until(
   fieldIn(object).ofType(int.class).andWithName("value"), is(1)
);
```

Atomic

```
AtomicInteger atomic = new AtomicInteger(0);
..
await().untilAtomic(atomic, equalTo(1));
```

Atomic boolean

```
AtomicBoolean myBoolean = new AtomicBoolean(false);
..
await().untilTrue(myBoolean);
```

JAYWAY

Advanced

```
with().
      pollInterval(ONE_HUNDERED_MILLISECONDS).and().
      pollDelay(20, MILLISECONDS).
await("user registration").
      until(userStatus(), equalTo(REGISTERED));
```

Scala API

```
await until { numberOfReceivedMessages() > 3 }
```

Use trait AwaitilitySupport

Groovy API

```
await().until { numberOfReceivedMessages() > 3 }
```

Extend/Mixin AwaitilitySupport

More

- Exception handling
- Defaults

```
Awaitility.setDefaultTimeout(..)
Awaitility.setDefaultPollInterval(..)
Awaitility.setDefaultPollDelay(..)
```

Web Page & Contact

- Google for: awaitility
- Web page
  - http://code.google.com/p/awaitility/
- Twitter
  - johanhaleby
- Blog:
  - http://blog.jayway.com/author/johanhaleby/

JAYWAY

JAYWAY