let's move
the java world

Hurdle run through EJB testing

Jakub www.marchwicki.pl / @kubem

This is not only about testing

Neither about EJB

Nor TDD

Just somewhere in between

Who is this guy?!

Business intelligence and data modelling
Inteligent marketing
Data mining

I’m part of Automotive Group
Working on Operation Data Store
Realtime view for Data Warehouse

So let’s start with EJB

Is it a a plane?

Is it a bird?

Is it a framework?

JEE takes all these technologies

Puts it together

In a single server

Big server

And we are enterprise (from now on)

According to the spec

You are now about to
Develop
Assemble
Deploy
Run
an Enterprise Solution

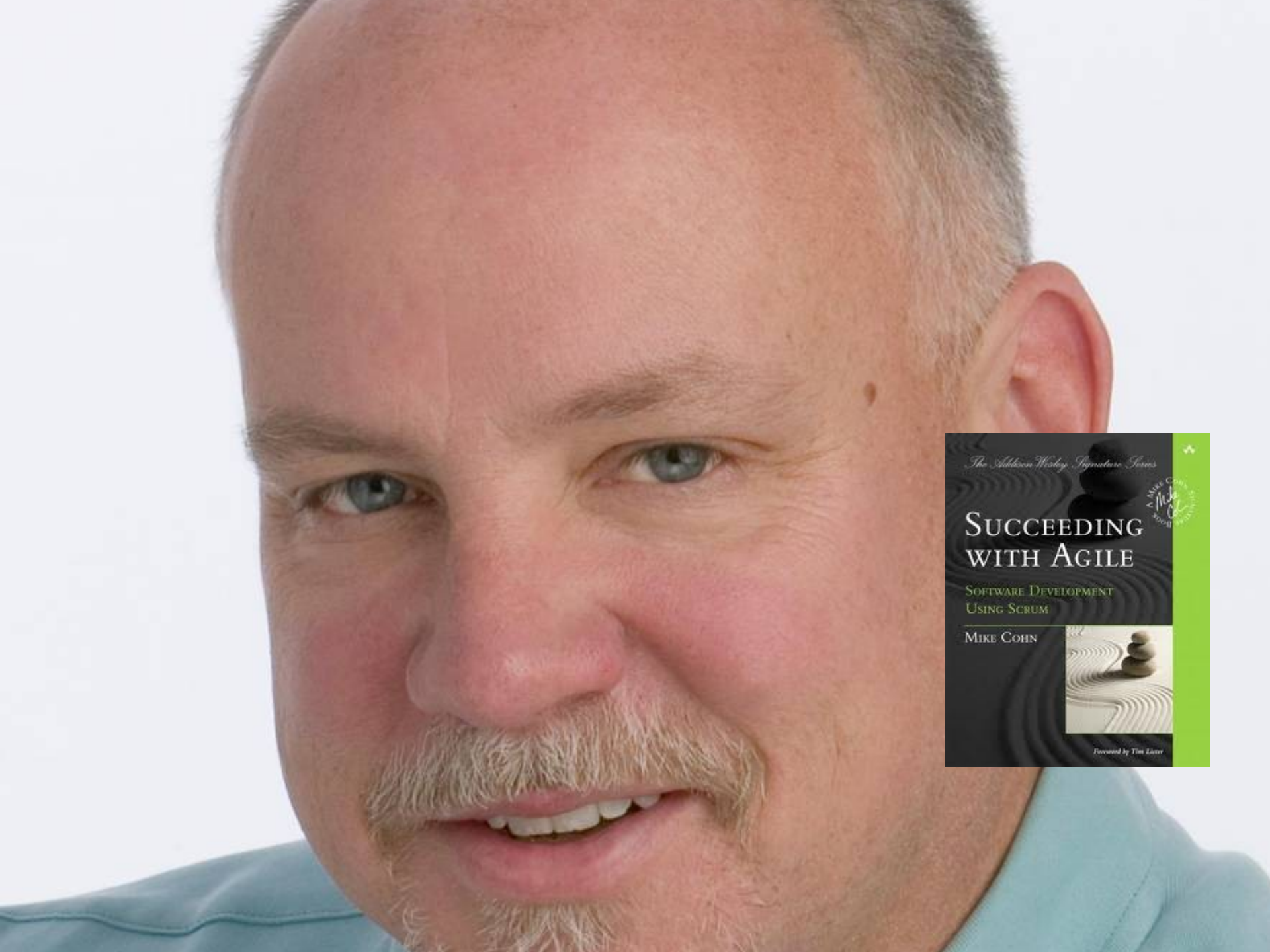
Emmm...

- Are we testing anything here, sir?
- Nah... Test are for testers, n00b!

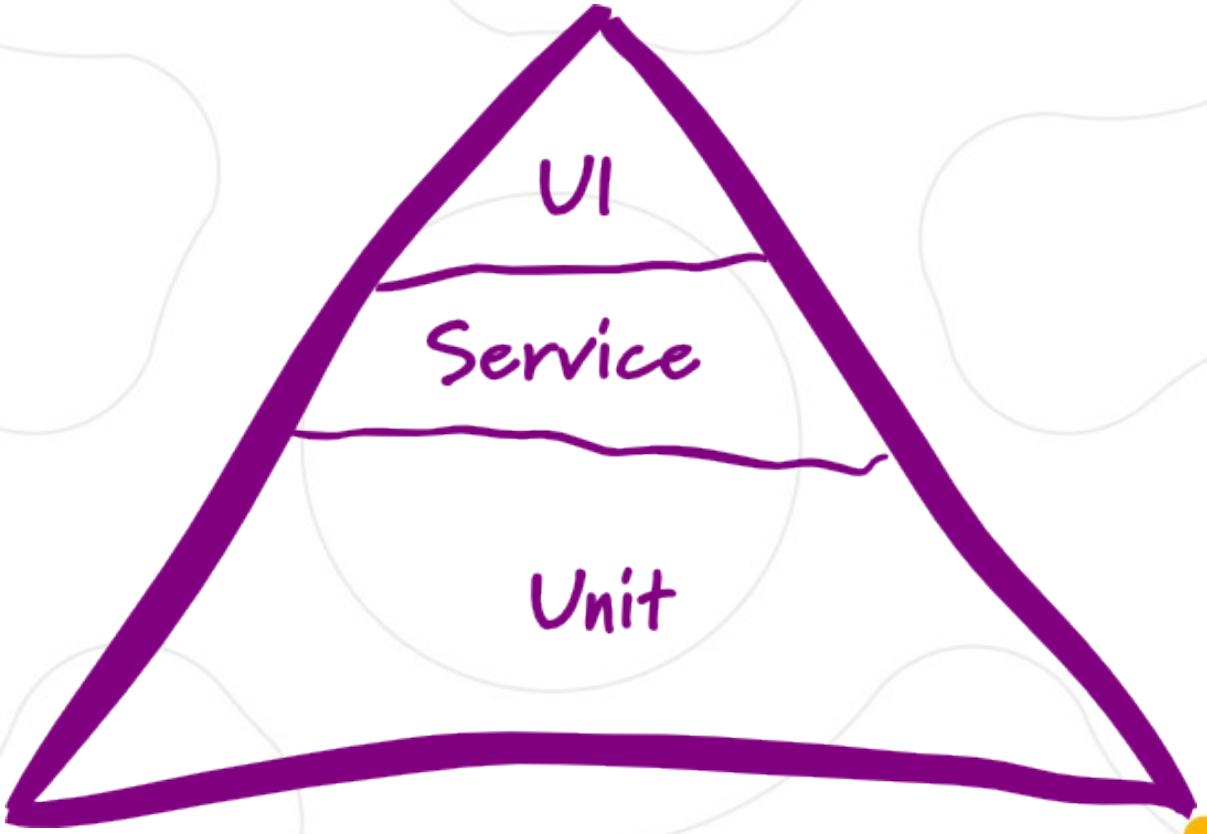
TDD

It’s a Tester Driven Development

… bitch please!


The Addison Wesley Signature Series
Succeeding with Agile
Software Development Using Scrum
Mike Cohn
Foreword by Tim Lister

But seriously – Test Pyramid



Tests…?

But tests slows me down

They need maintenance

This is too simple to test

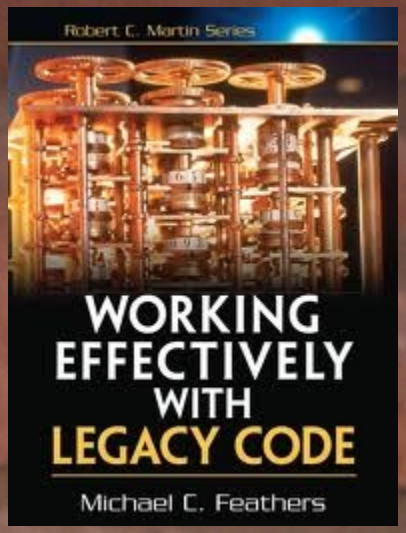
And I don’t make mistakes afterall

This is legacy code, there are no tests

Build takes time… more time

2012 geecon

http://www.slideshare.net/sebrose/common-objections-to-tdd-and-their-refutations


Robert C. Martin Series
WORKING
EFFECTIVELY
WITH
LEGACY CODE
Michael C. Feathers

Start from the bottom

Unit tests don’t:

talk to the database

communicate over network

touch the file system

need to be run independently

(tests can run simultaneously)

need special environment configuration

No tests in EJB?

If these are integretion tests...

It means...

There are no tests in EJB...

Woohoo!

Quick win!

Ekhmmm...

Is EJB the only framework?
How do other tackle testing?

You know when it’s Spring

Rod Johnson says it’s all about testing

Dependency Injection

```
@ContextConfiguration
@RunWith(SpringJUnit4ClassRunner.class)
public class ExampleConfigurationTests {
    @Autowired
    ApplicationContext ctx;

    @Autowired
    MyService service;

    @Test
    public void servletTest() throws Exception {

       //.. yada yada
    }

}
```

Database tests

```
<beans>
    <jdbc:embedded-database id="dataSource" type="HSQL">
        <jdbc:script location="import.sql"/>
    </jdbc:embedded-database>
</beans>
```

Spring + Web

```
@Autowired
ApplicationContext ctx;

@Autowired
TaskController controller;

@Test
public void servletTest() throws Exception {
   MockHttpServletRequest req = new MockHttpServletRequest("GET", "/tasks");
   MockHttpServletResponse resp = new MockHttpServletResponse();
   HandlerAdapter handlerAdapter =
ctx.getBean(AnnotationMethodHandlerAdapter.class);
   final ModelAndView model = handlerAdapter.handle(req, resp, controller);

   assertViewName(model, "tasks");
   assertAndReturnModelAttributeOfType(model, "task", Task.class);
   assertAndReturnModelAttributeOfType(model, "tasks", List.class);

   //.. create testTask object

   assertModelAttributeValue(model, "tasks", Arrays.asList(testTask));
   assertModelAttributeValue(model, "task", new Task());
}
```

Spring + Web (2)

```
MockMvcBuilders.annotationConfigSetup(TestConfiguration.class)
    .build()
    .perform(get("/resources/Spring.js"))
.andExpect(content().type("application/octet-stream"))
.andExpect(content().string(containsString("Spring={};")));
```

RestEASY

```
MockHttpRequest request = MockHttpRequest.get("/user/1234");
MockHttpResponse response = new MockHttpResponse();

Dispatcher dispatcher = MockDispatcherFactory.createDispatcher();
dispatcher.invoke(request, response);
assertEquals(response.getStatus(), HttpServletResponse.SC_METHOD_NOT_ALLOWED);


URI uri = UriBuilder.fromPath("/user/{arg1}/delete").build(clientId);

MockHttpRequest request = MockHttpRequest.get(uri.toString());
MockHttpResponse response = new MockHttpResponse();

dispatcher.invoke(request, response);
assertEquals(response.getStatus(), HttpServletResponse.SC_OK);
assertTrue(response.getContentAsString().contains("def"),
    "Response should contain 'def' result.");
assertTrue(response.getContentAsString().contains("xx"),
    "Response should contain 'xx' result.");
assertTrue(response.getContentAsString().contains("123"),
    "Response should contain '123' result.");
```

There are frameworks...

... and frameworks

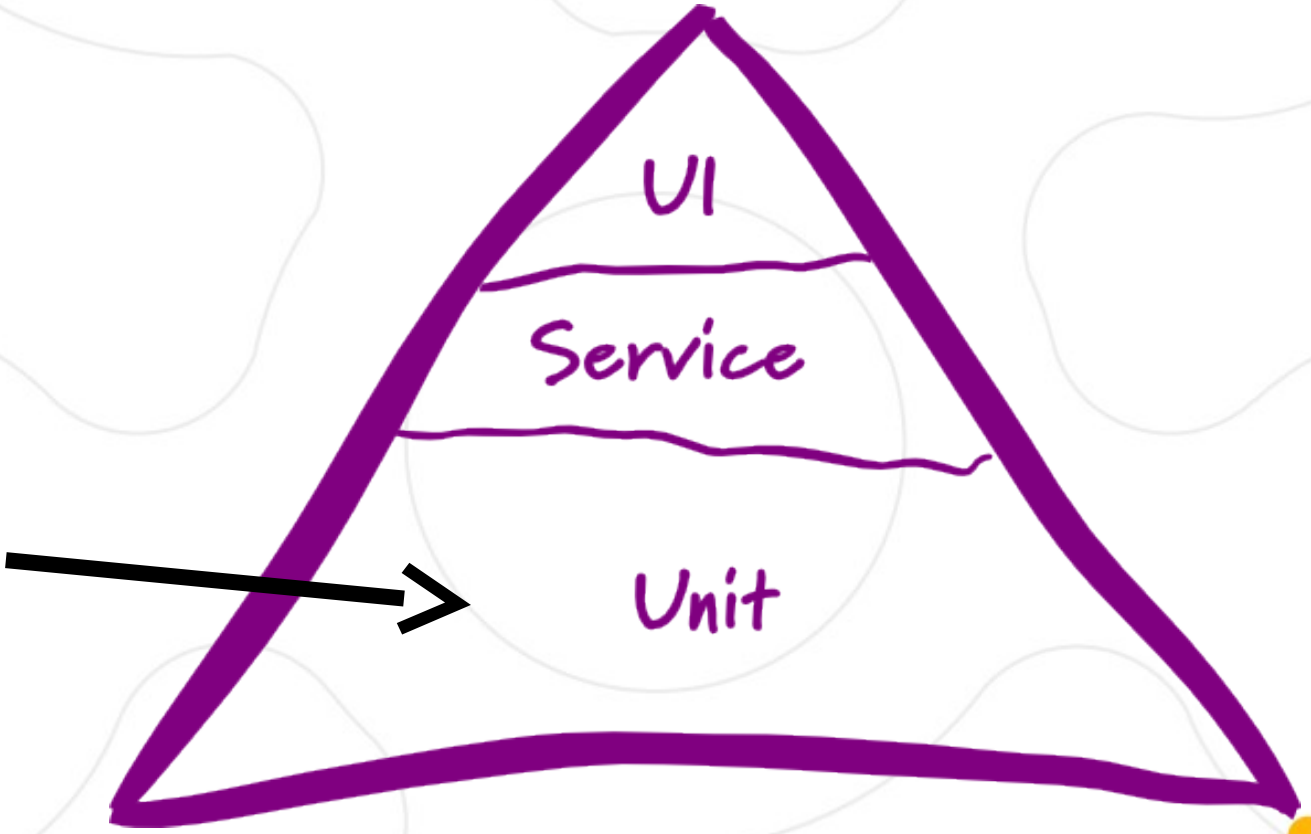
Some things are built with testing in mind

But we are to talk about EJB

... available since 2006 (EJB 3.0)

Vendors deliver a platform
And they don’t really care about testing
Just until now... 6 f**kn years
EJB 3.1 is also live
Still noone cares about tests
(nearly noone)

Back to The Pyramid



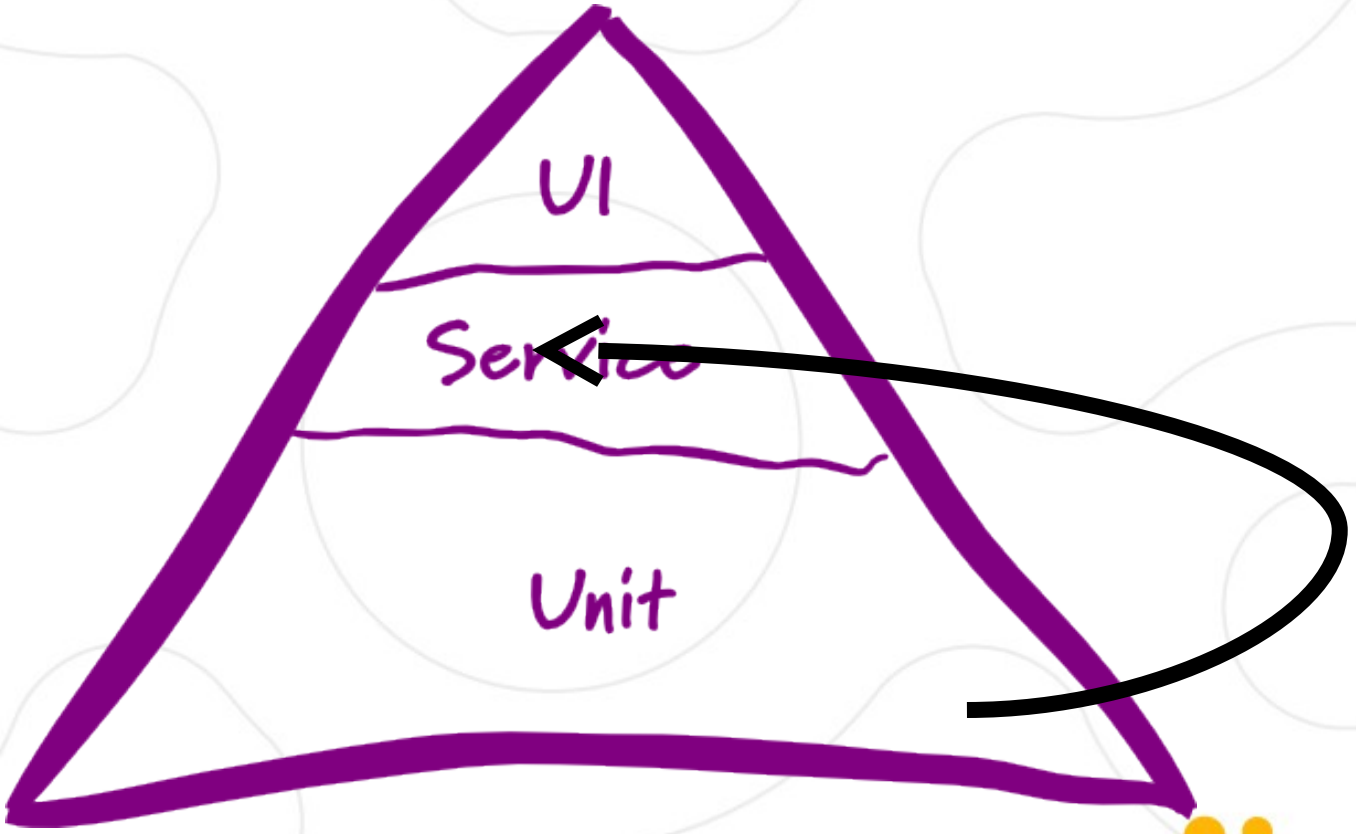
Let’s have a look

Welcome the sample application

https://github.com/kubamarchwicki/geecon-ejbtest-workshop

Mate! It’s IDE time!

Mocks are tactical



Fully blown integration

There is a hassle with maintaing DI

Hello… calling dr. Container

It’s your job afterall

Are there other options?

Embed a container

bootsrtap whole application

test end2end

Mate! It’s IDE time!

What’s else - Open EJB

yet another embedded container

What’s else - Open EJB

PROS

It just works!

Selective classpath scanning!

What’s else - Open EJB

PROS

It just works!

Selective classpath scanning!

CONS

Different than your production container

What’s else (2) - Arquillian

wooow, so there is a test framework for EJB

What’s else (2) - Arquillian

PROS

Closer to actual production environment

Independent from the actual container

What’s else (2) - Arquillian

PROS

Closer to actual production environment

Independent from the actual container

CONS

Just graduated

Our container’s adapter is still in alpha

Arquillian seems nice

Manage lifecycle for you

Shrinkwrap creates an archive

Adapters link between any container and arquillian

Remote testing / embedded testing

Mate! It’s IDE time!

???

Jakub www.marchwicki.pl
@kubem

2012 geecon